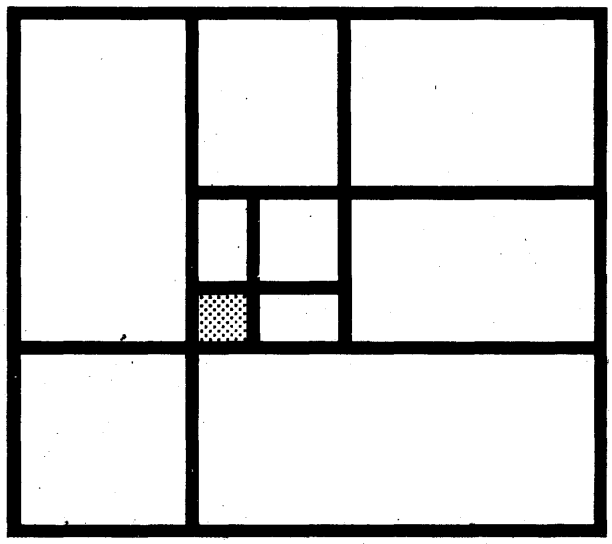
PCSI- 650

PROCESS COMPUTER

GE PAC® 4010



GENERAL ELECTRIC

UTILITY AND PROCESS AUTOMATION
PRODUCTS DEPARTMENT
PHOENIX, ARIZONA

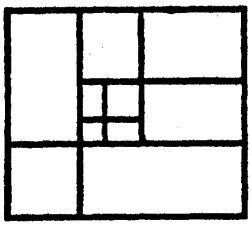
GE-NPGCD CAROLINA I (AZNA) REQN: 282-L3285

The GE PAC® 4010

Process Computer System

VOLUME I

SYSTEM DESCRIPTION
SYSTEM POWER
CORE MEMORY
ARITHMATIC UNIT
AUTOMATIC PROGRAM INTERRUPTS
I/O EXPANDER/COUPLER



This instruction book is provided as a basic source of technical information on this system. If problems arise in installation, operation, or maintenance that are not covered in these instructions, the matter should be referred to General Electric Company, Process Computer Products Department, Phoenix, Arizona, U.S.A. Attn: Technical Publications.

General Electric reserves the right to make changes in the equipment or software, and its characteristics or functions, at any time without notice.

GENERAL ELECTRIC

4780E CSU POWER SUBSYSTEM

TABLE OF CONTENTS

INTRODUCTION

GENERAL DESCRIPTION

The Central System Unit Power Subsystem provides the following functions:

- Distributes AC power as required throughout the CSU.
- Controls the application of AC power to the CSU power supply modules.
- Controls the application of power to external cabinets and, indirectly, to the bulk memory units.
- Converts the primary AC power to regulated DC power for the CSU.
- Monitors the Logic and Memory Power Supply outputs.
- Monitors temperatures at several points in the CSU.
- Provides an orderly shut-down of sequenced system power in the event of an out of tolerance voltage, excessive temperature, or an operator initiated shut-down.
- Provides the Automatic Restart Option, if implemented.
- Provides the Alternate Source Timer Option, if implemented.
- Provides the CSU Power Surge Limiter Option, if implemented.

The principal modules of the power subsystem are depicted on Fig. INT. 1.

REFERENCES

The following GE drawings provide schematics, logic, and other information useful in understanding and maintaining the CSU Power Subsystem:

Drawing No.	Description
70C180914	Logic, 4780E Power Modules
70C180955	Logic, Arithmetic Unit; sheet 96.2 - Auto Restart " 140 - BLM " 141 - Power On/Off Circuit " 142 - Optional Alarm Outputs " 144 - Alternate Source Timer " 145 - Line Frequency Timer

The Power Supplies section of the Computer Maintenance Manual provides preventive maintenance, test, adjusting, and troubleshooting procedures for the CSU Power Subsystem, as well as other power supply modules used in the GE-PAC* 4010 Computer System.

*Registered Trademark of General Electric Company

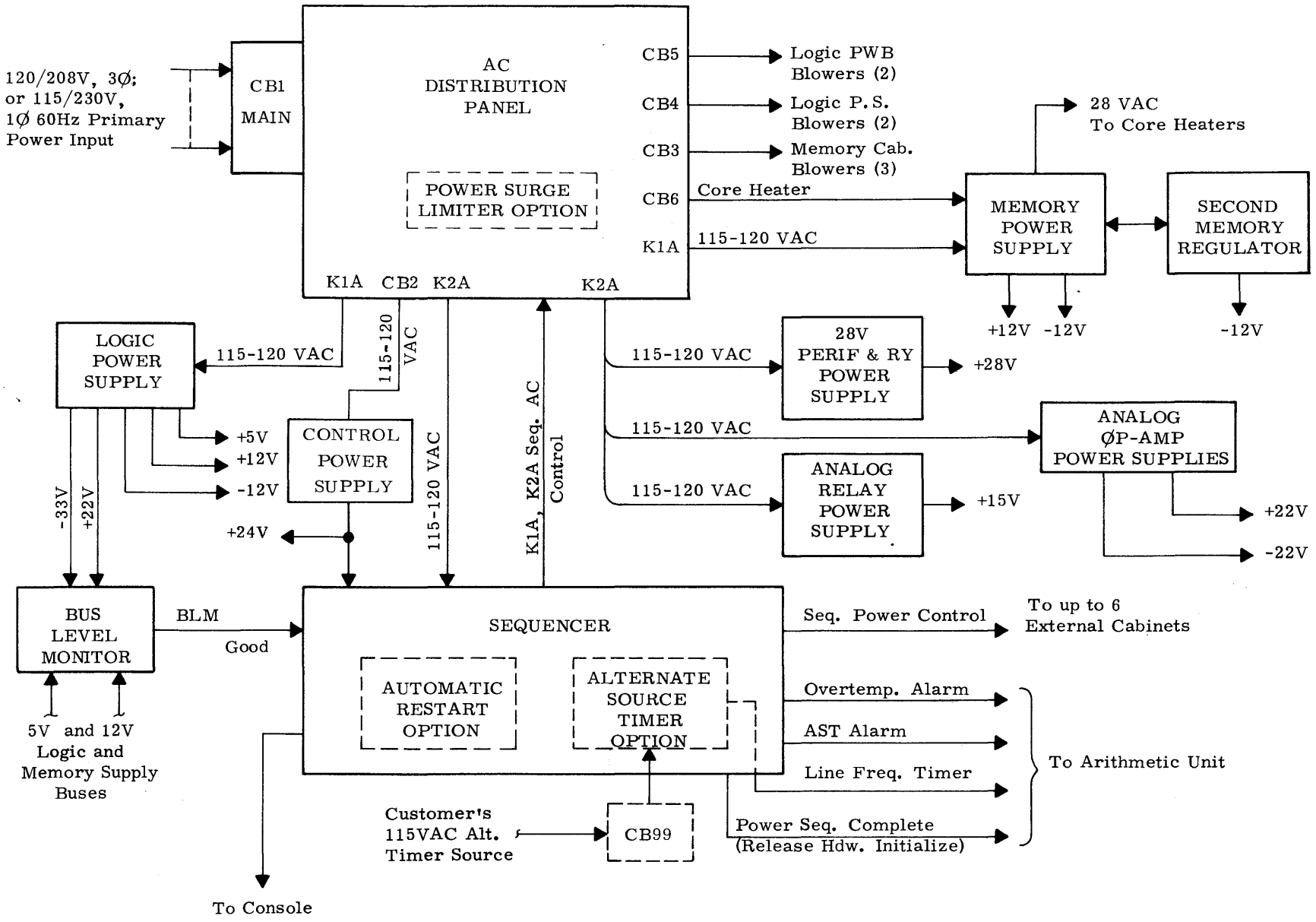


Fig. INT. 1 Block Diagram, CSU Power Subsystem

THEORY OF OPERATION

SEQUENCER AND POWER CONTROLS

The Sequencer and the power control elements provide an orderly start-up of the computer system and, if necessary, an orderly shut-down. Power subsystem wiring and components are protected by several circuit breakers. Temperatures at several important CSU points are monitored by the Sequencer circuits and a power shut-down is initiated when any temperature exceeds its limit. These circuits also shut down sequenced power when the Bus Level Monitor (BLM) detects an out of tolerance voltage and when the Off pushbutton on the Programming and Maintenance Console is pushed.

The majority of components and circuits involved in sequencing and power control are located in the AC Distribution Panel and the Sequencer. These modules are located at the rear of the CSU, at the lower left-hand corner, as viewed from the rear. The AC panel is located just above the Main circuit breaker, CB1, and the Sequencer is immediately above the AC panel.

Primary Power Input

The customer's primary power source is connected to CB1 as shown on sheet 10 of the Power Module logic (70C180914). The chassis ground point shown near CB1 on sheet 10 is the safety ground connection point and must not be connected to any current carrying ground such as the neutral wire. For further information on the primary power source and connections, refer to the GE-PAC 4010 Site Planning Manual GET-6075.

If the primary input power is 208V $\pm$10%, 3-phase, 60 Hz $\pm$3%, the nominal line to neutral voltage (A, B, or C, to N) will be 120VAC. In this case, lines A, B, and C on the logic drawing are phases A, B, and C, respectively.

If the primary input power is 115/230V $\pm$10%, single phase, 60 Hz $\pm$3%, the nominal line to neutral voltage will be 115VAC. In this case, lines A and B are connected to one side of the 230V line, and C is connected to the opposite side of the 230V line.

Power On Sequence

In the following discussion, it is assumed that all circuit breakers are closed and that the primary input power is on. Fig. THEORY. 1 is a simplified schematic of the Sequencer and the associated AC distribution circuits. Circuit elements which appear on several pages of the -914 logic are combined into a working schematic on Fig. THEORY. 1, with connector and pin number details left out. Refer also to the "Turn On Sequence" flow chart on sheet 3 of the -914 logic.

1. Non-sequenced AC power is applied to the CSU blowers, the Core Memory heater supply, and to the +24V Control Power Supply in the Sequencer (sh. 14 of -914 logic and Fig. THEORY. 1).

2. A start or restart can be attempted only after relay K6 has energized. This will occur either 1.5 seconds after AC power is first applied, or 1.5 seconds after sequenced power is shut down. Relay K6 provides this restart delay to ensure that all of the protective circuits including the overvoltage crowbars have completely stabilized in the "off" state and all capacitors have been discharged. Prior to a start sequence, all relays are de-energized. If unsequenced power is present, or when it returns, the 1.5 second timer on the ground return side of K6 is started, and K6 picks up at the end of the 1.5 second period. The 1.5 second timer is similar to several which are used in the Sequencer. Refer to the "Unijunction Timers" heading.

3. With the Programming and Maintenance Console enabled and in Manual mode, a start or restart may be attempted by pushing the On pushbutton. If the Automatic Restart option is implemented, relay K7 will be present and an automatic restart will be attempted if the Automatic mode is selected at the console (refer to the "Automatic Restart" heading). In either case, +6V is applied through the Off pushbutton, and through thermostats on the logic and memory supplies, to a relay driver, which is enabled, energizing K1.

4. The one-second timer on the ground return side of relay K1T is started when the K1 contacts open. This timer will stop the power-on sequence, if the logic and memory supply outputs are not "in tolerance" within one second. Refer to the "Suicide Timer" heading.

5. Relay K1A in the AC Panel is energized and AC power is applied to the Logic Power Supply and the Memory Power Supply.

6. When the logic and memory supply voltage outputs are within tolerance (about 115 milliseconds after K1 picks up), a relay driver (on the SBMA1 BLM board in the B Panel) is enabled which picks up K2. Contacts on K2 energize the +24V (Seq.) voltage source and energize K2A in the AC panel. The K2A contacts apply AC power to the 28V Peripheral and Relay Power Supply, to the analog power supplies (if present), to the console transformer, and to the line frequency timer. At this time, the operator may determine if the sequenced power is on, as the Programming and Maintenance Console indicators

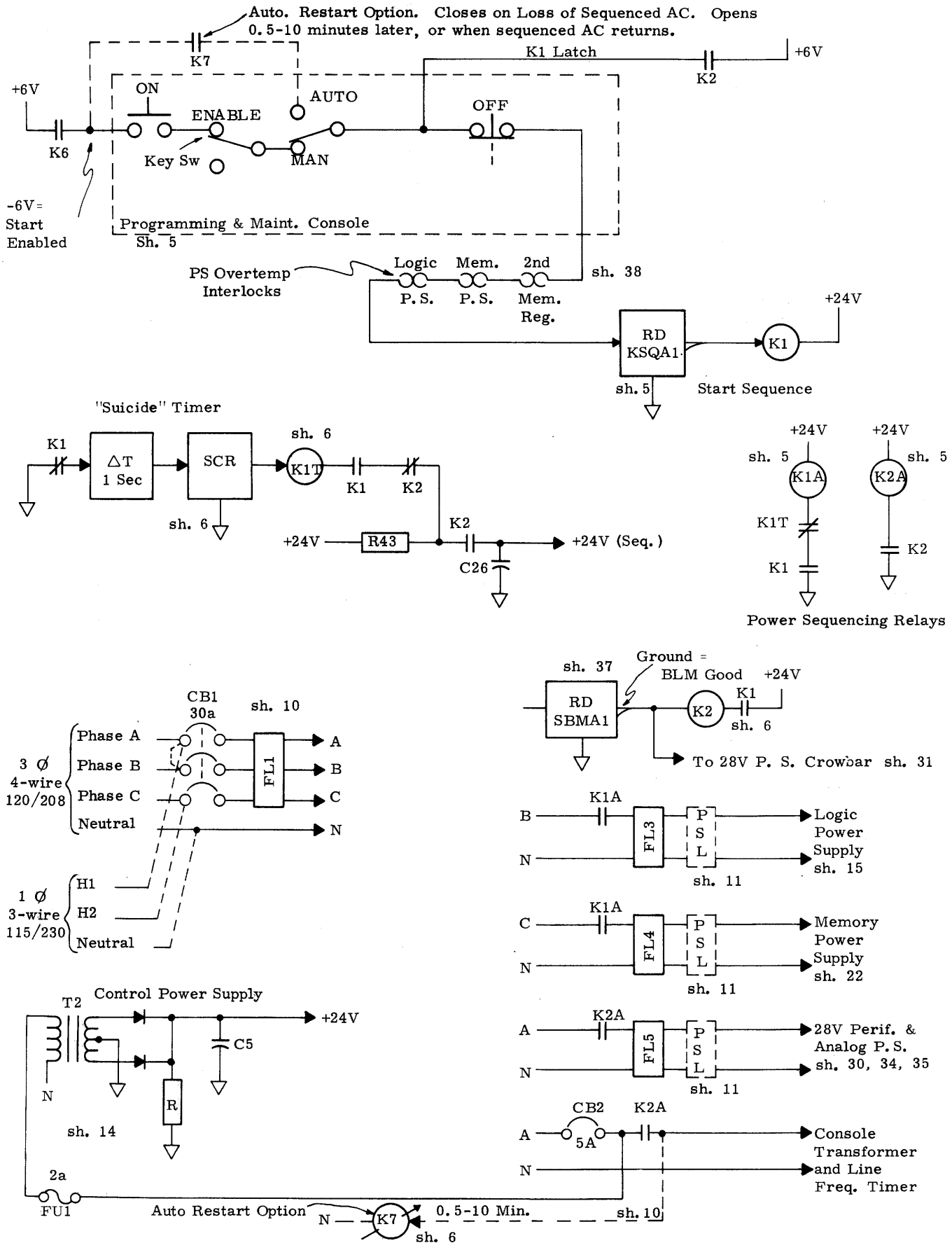
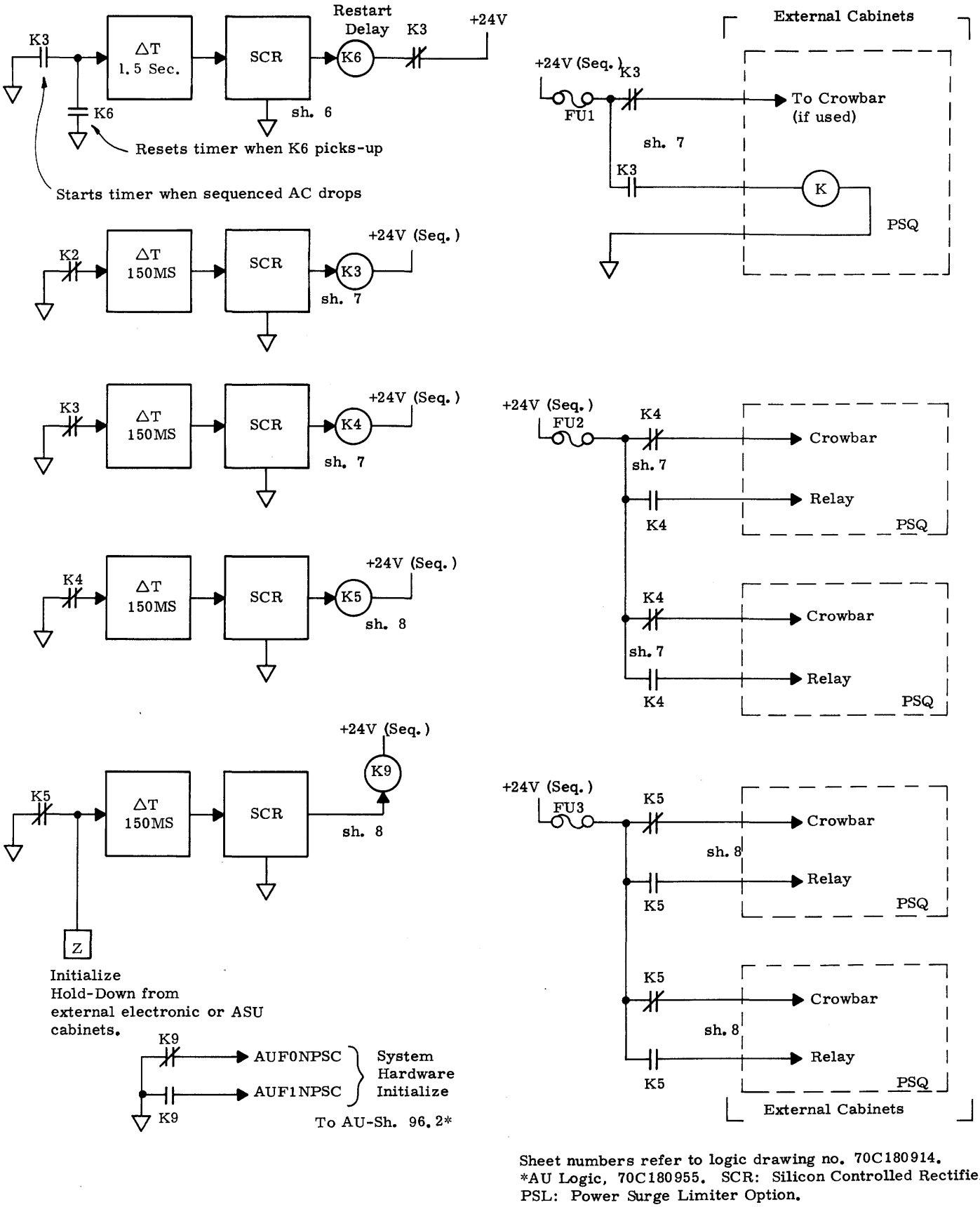


Fig. THEORY.1 Simplified Schematic, Sequencer and AC Power Distribution



Sheet numbers refer to logic drawing no. 70C180914.
*AU Logic, 70C180955. SCR: Silicon Controlled Rectifier
PSL: Power Surge Limiter Option.

Fig. THEORY.1 Simplified Schematic, Sequencer and AC Power Distribution (Cont'd)

will be illuminated because the 18VAC console transformer output is present (sh. 13 of -914 logic). A set of K2 contacts in the +24V line to K1T is now open, preventing the pick-up of K1T when the Suicide Timer runs out.

7. The 150 millisecond timer on the ground return side of K3 is started when the K2 contacts open. 150 milliseconds later, K3 picks up, and its contacts pick up the sequenced power control relay in the first external cabinet, applying sequenced AC power to the loads in that cabinet.

8. The timer on relay K4 is started when a K3 contact-set opens. 150 milliseconds later, K4 energizes, its contacts pick up the control relays in external cabinets 2 and 3, and sequenced AC power is applied to the loads in those cabinets.

9. 150 milliseconds later, K5 energizes, and sequenced AC power is applied to the loads in external cabinets 4 and 5.

10. 150 milliseconds after K5 picks up, K9 is energized, if no initialize hold-down signal (ground at point "Z") is present. If a hold-down signal is present, K9 picks up 150 ms after the hold-down signal is removed. The hold-down signal would be applied only by external electronics or ASU cabinets. When K9 picks up, the power sequence complete flip-flop, F1NPSC, in the Arithmetic Unit is set, releasing the system hardware initialization signal, and allowing the system to operate.

Sequenced AC power is applied to the first drum or disc unit in a system when the BLM B signal goes "good", or about 70 milliseconds after the K2 relay driver is enabled (step 6). The "BLM good" signal is routed to bulk memory unit 0 through the Arithmetic Unit and the Dual Bulk Controller. Sequenced AC power is brought on in each subsequent bulk memory unit in a system, by the power control circuits in those units, as the power-on sequence in the previous bulk memory unit is completed.

Power Off Sequence

A power-off sequence begins 1) when the Off pushbutton on the Programming and Maintenance Console is pushed, 2) when the primary AC power is lost, 3) when the Bus Level Monitor senses an out of tolerance voltage, or 4) when the high temperature limit of the Logic Power Supply or one of the Memory Power Supplies is exceeded.

A. Off pushbutton is pushed:

1. During normal operation, K1 is held energized by contacts on K2 which apply +6V to the K1 relay driver through the Off pushbutton and the power supply thermostats. Pushing the Off pushbutton breaks that circuit, dropping K1. Refer to Fig. THEORY. 1.

2. K1A drops as a K1 contact-set opens, removing AC power from the logic and memory supplies.

3. As the logic and memory regulator output voltages begin to decay, the BLM senses the undervoltage, the BLM relay driver turns off, allowing the cathode of CR3 in the 28V crowbar circuit (sh. 31 of -914 logic) to swing in the positive direction. This allows zener diode CR2 to conduct, pulling the gate of the SCR in the crowbar circuit up to the firing point. The conducting SCR drops the 28V output to zero almost immediately, preventing further operation of the peripheral devices and circuitry using the 28V supply.

4. Relay K2 has dropped. A K2 contact-set opens, dropping K2A and removing AC power from the 28V power supply, the analog power supplies, the console transformer, and the line frequency timer.

5. +24V is removed from the +24V (Seq.) line, dropping the power sequencing relays and resetting all of the 150 millisecond timers.

6. As +24V (Seq.) goes off, K9 drops. When power is reapplied, K9 will remain de-energized until the power-on sequence is again complete, and F1NPSC in the AU will be reset, holding down the hardware initialize signal, until power is restored.

B. Loss of primary AC power:

1. Within a few milliseconds of the loss of power, one or more of the logic and memory supply outputs will decay to the lower limit, disabling the BLM relay driver, and dropping K2. Refer to Fig. THEORY. 1.

2. A K2 contact-set opens, breaks the K1 latch circuit, and drops K1.

3. The remainder of the sequence is as under steps 2 through 6 in paragraph A, above.

C. BLM senses an out of tolerance voltage:

1. This event may occur because of a loss of primary power, because of a fault in a power supply or load causing an undervoltage, or because the upper limit of a power supply output was exceeded, causing a crowbar to fire. Once a crowbar has fired, the BLM senses the output as undervoltage. Refer to Fig. THEORY. 1.

2. The BLM relay driver is disabled and K2 drops. A K2 contact set opens, breaks the K1 latch circuit, and drops K1. The remainder of the sequence is as under steps 2 through 6 in paragraph A on page THEORY-4.

D Power supply high temperature limit exceeded:

1. The power supply overtemperature shutdown interlocks are shown on sheet 38 of the -914 logic. The following thermostats are connected in series in the K1 latch circuit:

   Logic Power Supply S2 - opens when the temperature of the +5V heat sink exceeds +287°F ±10°F.

   Memory Power Supply S1 - opens when the temperature of the -12V heat sink exceeds +287°F ±10°F.

   2nd Memory Regulator S1 - opens when the temperature of the -12V heat sink on the -12V regulator for the 2nd 16K core stack exceeds +287°F ±10°F. If the 2nd 16K core stack is not implemented, a dummy plug is used in place of this thermostat.

2. If any of the thermostats opens, the K1 relay driver is disabled and K1 drops. The remainder of the sequence is as under steps 2 through 6 in paragraph A. Refer to Fig. THEORY. 1.

Unijunction Timers

Six unijunction transistor timer circuits are used in the Sequencer to control the time at which various events in the power-on sequence occur. All of the timers use the same type of circuit and all are located on the KSQA1 printed wire board. A schematic of the timer circuit appears at the upper left corner of sheet 6 of the -914 logic.

The timers are reset when a relay contact-set connected between the emitter of the unijunction transistor and ground is closed. When the contacts open, the capacitor begins to charge through a resistor toward the +12V supply voltage. As the capacitor charges, a point is eventually reached where the emitter to base-1 resistance drops low enough to discharge the capacitor. This produces a positive pulse on the gate of the SCR which causes the SCR to fire. Once fired, the SCR provides a ground return for the relay solenoid. If +24V is available at the other end of the solenoid winding, the relay energizes. The resistors which are connected across each relay coil (sh. 6 of -914 logic) provide sufficient holding current for the SCR, while the current through the coil is building up. Resistors and capacitors such as C27 and R43 are provided to absorb voltage transients created when the field in the coil collapses, as the relay is de-energized. The diodes across the coils also aid in absorbing this energy. The timer is reset by the closing of the contacts across the timing capacitor, which discharges the capacitor. If it is conducting, the SCR will continue to conduct until some other event, such as the opening of a contact-set or removal of +24V reduces the current below the holding-current level of the SCR.

The value of the timing capacitor and the resistor through which it charges determine the time delay time, as indicated by the table just below the typical circuit schematic on sheet 6 of the logic.

Suicide Timer

In normal operation, sequenced AC power is held on only while the Bus Level Monitor (BLM) and the overtemperature thermostats allow Sequencer relays K1 and K2 to remain energized. The Suicide Timer allows AC power to be applied to the logic and memory supplies by K1A, while the output voltages are building up to the proper levels. The timer and associated relay, K1T, appear on sheet 6 of the -914 logic and on Fig. THEORY. 1.

The timer and K1T allow up to one second for the BLM signal to go "good". When the BLM signal does go "good", K2 picks up, which allows the remainder of the power-on sequence to continue, including the application of sequenced AC to the analog and +28V power supplies, the line frequency timer, the console transformer, and the external cabinets. A K2 contact set breaks the +24V line to K1T.

If the one-second timer fires the SCR before K2 picks up, K1T energizes, its contacts open the ground return for K1A, K1A drops, AC power is removed from the logic and memory supplies, and the power-on sequence is discontinued.

A manual or automatic restart may be attempted. If the time-out of K1T was due to a power supply fault, and the primary AC remained on, a second automatic restart attempt will not be made. This is because relay K1 will be held by the contacts on the automatic restart relay, K7. If K1 is not dropped, K1T remains energized, and it will be necessary to go to Manual mode, which drops K1, and resets the suicide timer. A manual restart may then be attempted.

Automatic Restart Option

The Automatic Restart Option initiates a power-on sequence in the event of a momentary interruption of the primary AC power, or in the event of a shut-down due to a transient power supply fault. In some process applications, automatic restarts without operator intervention cannot be made safely. Where such restarts can be made, and where the customer provides appropriate recovery software, the computer can resume control of the process even though no operator is present. Refer to Fig. THEORY.1 and sheets 5 and 6 of the -914 logic.

Pneumatic time delay relay, K7, measures the time period in which restart attempts without operator intervention may be made. This allowable period is determined by appropriate personnel for each site incorporating the automatic restart option, and is adjusted to a value from 1/2 minute to 10 minutes, through a dial setting on K7.

When a shut-down occurs, relays K2 and K2A drop. When K2A drops, power is removed from the K7 coil and the K7 contacts close for the period indicated by the dial setting. If primary AC power is still present, K6 picks up in 1-1/2 seconds. If primary AC power was interrupted, K6 picks up 1-1/2 seconds after the power returns. If K7 has not timed-out, the K6 contacts apply +6V DC to the K1 relay driver through the K7 contacts, through the Auto side of the Auto/Man switch, through the Off pushbutton, and through the thermostats. K1 then picks up, initiating a power-on sequence.

If the primary AC power is interrupted again during the power-on sequence, the timers will be reset. If the power returns before K7 times-out, another restart attempt may be made when K6 picks up. Each time power returns the K7 timer is reset.

If the primary AC power was not interrupted, and if the suicide timer times-out and K1T picks up, before K2 picks up, K1 and K1T will remain energized until the K7 contacts open, or the Auto/Man switch is placed in the Manual position.

In the event of a successful restart, K9 will energize when the power-on sequence is complete, releasing the hardware initialize signal. Once the initialize signal is removed, the logic on sheet 96.2 of the Arithmetic Unit logic drawing, 70C180955, generates API no 200_8. This is the first API level and has the highest priority. The response location in core normally contains a BRU instruction which causes an unconditional branch to a corrective restart program.

Power Surge Limiter Option

The inrush current drawn from the primary AC supply during the power-on sequence is reduced to an acceptable level for many installations by the application of power to the loads in steps, as described under the Power On Sequence heading. The inrush current to the CSU may be reduced to no more than 150% of the rated continuous operating current by implementation of the Power Surge Limiter option. With the Power Surge Limiter, the inrush current to the CSU is 55 amperes peak, or less, for the first six cycles after K1A closes. Without the limiter, the inrush current to the CSU may be as high as 400 amperes peak during the first half-cycle.

The Power Surge Limiter limits the inrush current to the logic and memory supplies as K1A in the AC panel closes, and to the 28V and analog power supplies as K2A in the AC panel closes. Three separate but similar limiter circuits are employed. The limiter module is mounted in the AC distribution panel and connected in series with FL3, FL4, and FL5, as shown on sheet 11 of the -914 logic.

The inrush current to the loads in series with the three limiters is minimized by applying power to the loads at one of the points on the AC voltage waveform where minimum inrush current results. These points are at the peaks of the voltage waveform, where the rate of voltage change is least, as illustrated by Fig. THEORY. 2.

If the switching time is not controlled, it may occur near one of the zero volt crossings of the voltage waveform, where the rate of voltage change is maximum. This can cause the transformer core flux to rise from its residual flux level to beyond the saturation point for one or more cycles. If the core is allowed to saturate, very high current peaks may occur. If however, the switching time occurs at the peak of the voltage waveform, the core will not saturate and the inrush current will be minimized.

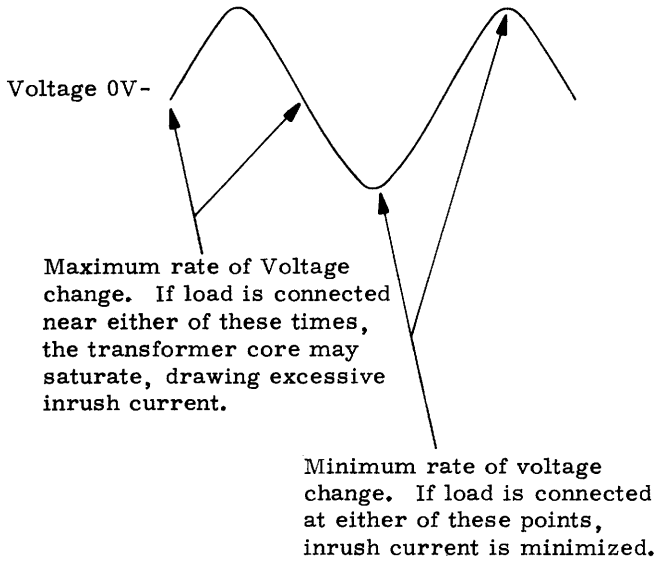


Fig. THEORY.2 Effect of Switching vs. Phase

The switching of power onto the loads is accomplished by three triac (triode-AC) semiconductor switches. The triacs operate as if they were two parallel- inverted SCR's which, once turned-on, conduct in either direction. When conducting, the triacs complete the neutral return line for the step-down transformers in the power supplies. Triacs may be turned on by a negative or a positive voltage on the gate. In the Power Surge Limiter, the triacs are turned on by a voltage which is negative with respect to the AC neutral line. The turn-on time for the three triacs is controlled by three phase control circuits. A simplified schematic of a typical limiter circuit appears on Fig. THEORY. 3. Complete schematic details are provided on sheets 11, 12, and 12.1 of the -914 logic.

When a power-on sequence starts, relay K1 in the sequencer is energized and a K1 contact set picks up three Power Surge Limiter enabling relays, K101, K102, and K103. Contacts on these relays connect the phase control circuit outputs to the gate circuits on the three triacs.

When power is applied to each of the circuits by the closing of contacts on K1A or K2A, the transformer and half-wave rectifier are energized, producing about 16 V DC across the phase control circuit. R2, in the emitter circuit of the unijunction transistor, is adjusted and sealed at the factory to cause the unijunction transistor to fire at the peak of each half-cycle of the AC waveform. When the unijunction transistor fires, it discharges the one microfarad capacitor in the emitter circuit, producing a positive spike on base-1 of the transistor. Since the relay contacts across the gate circuit of the SCR are still closed, the firing of the unijunction transistor at this time has no effect on the SCR.

When the 22 microfarad capacitance across the phase control circuit relay coil has charged sufficiently, relay Kx is energized. This requires at least one full AC cycle (16.67 milliseconds) from the time power is applied to the phase control circuit. When the relay is energized, the next pulse out of the unijunction transistor circuit fires the SCR, pulling the triac gate in a negative direction, and firing the triac. The triac has therefore been fired at the peak of the first half-cycle, following at least one full cycle from the time the K1A or K2A contacts closed.

The 300 microfarad capacitance stores sufficient energy at the DC voltage peaks to maintain the SCR and triac gate current between peaks.

Since power is first applied to the transformer primary at one of the peaks of the AC waveform, the core is not driven from its residual flux point into saturation, and excessive peak current is not drawn from the primary supply. Since the surge limiter which is in series with the AC lines to the 28V and analog supplies is energized several milliseconds after the first two limiters are energized, the inrush current for that load is drawn at a later time, which contributes to the spread of the overall inrush current demand.

During a power-off sequence, sequencer relay K1 drops, dropping K101, K102, and K103, opening the contacts in the triac gate circuits and removing the negative voltages from the gates. With no gate current, the triac turns off when the current through it next drops below the holding current for the device. Before a restart can be attempted, K104, K105, and K106 will have dropped, re-closing the contacts across the SCR gate circuit.

LINE FREQUENCY TIMER AND ALTERNATE SOURCE TIMER

The Line Frequency Timer is a functional component of the Arithmetic Unit and appears on sheet 145 of the AU logic, 70C180955. The timer generates API no. 232_8, and the response location contains a DMT instruction which is used for system time keeping.

If the Alternate Source Timer option is not implemented, the input to the Line Frequency Timer is provided by the console lamp transformer in the Sequencer, as shown on sheet 13 of the -914 logic. Two 18V RMS AC signals are provided. Each is 180° out of phase with the other.

The Alternate Source Timer is used when the primary computer power is derived from a source, such as a motor-generator set, which does not provide an accurate, stable line frequency. The alternate timer signal is derived from an accurate, stable supply, such as is normally available on commercial power lines. When the option is implemented, the alternate timer source is connected to CB1A, which is located next to the main circuit breaker, CB1, at the rear of the CSU, in the lower left-hand corner, as viewed from the rear.

Sequenced AC power from the primary source is applied to relay K1 on the Alternate Source Timer board, which is located in the Sequencer, and is shown on sheet 13 of the -914 logic. Two sets of K1 contacts close when the sequenced AC is on, and the alternate AC source is applied to a bridge rectifier. The DC output of the rectifier energizes K3 and K2. A set of K2 contacts connects logic ground to the center tap on the secondary of transformer T1, which provides a conduction path for the diodes on the T1 secondary. The alternate AC source, at about 18V RMS, is applied to the Line Frequency Timer in the AU.

Should the alternate source be turned off or fail, K3 and K2 will drop. Closure of the K3 contacts produces an "Alternate Source Timer Failure" alarm output from the optional CSU alarm board (refer to AU logic, 70C180955, sheet 142). When K2 is dropped, the center tap on the secondary of T2 is grounded, providing a conduction path for the diodes connected to that transformer, and applying the primary AC signal to the Line Frequency Timer.

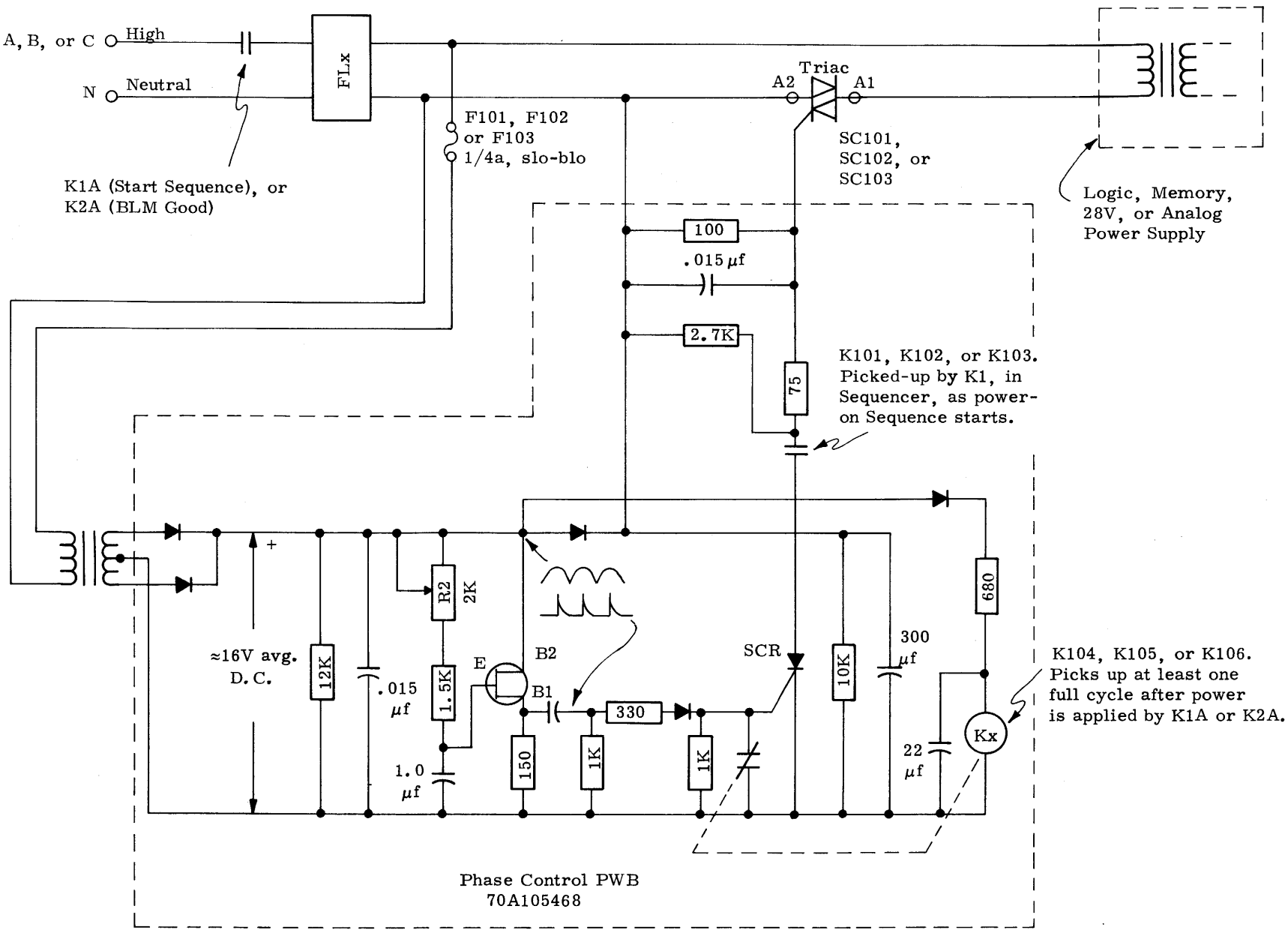


Fig. THEORY. 3 Typical Power Surge Limiter

LOGIC POWER SUPPLY

The Logic Power Supply provides +5V, +12V, and -12V DC voltages to the DC busses which distribute logic power throughout the CSU. The power supply module is located at the rear of the CSU, just inside the left-hand wall of the 60 inch cabinet, as viewed from the rear. In addition to the internal power supply components, which include the step-down transformer, rectifiers, filter capacitors, amplifier PWB's (PC1, 2, and 3), and heat sinks; three crow-bar circuits are mounted outside the module, on the end nearest the center of the CSU, just below the bus bars.

In addition to the logic voltages already mentioned, the Logic Power Supply provides unregulated +22V DC and -33V DC to the Bus Level Monitor.

Schematics parts lists, and component lay-out sketches for the Logic Power Supply are provided on sheets 15 through 21 of the Power Module Logic 70C180914.

AC Input

The logic power supply schematic on sheet 15 of the -914 logic indicates that the AC input may be 115V or 230V, and 50 Hz or 60 Hz. In this application, the AC input is 115V AC at 60 Hz, as supplied through K1A contacts and filter, FL3, in the AC panel.

Step-down transformer, T1, incorporates a ferro-resonant winding (sh. 15 of -914 logic) which improves the line regulation of the transformer. C14, C15, and C16 form a resonant circuit with the inductance of the winding.

Rectifiers

The four T1 secondary windings drive a total of five rectifiers:

- CR9, CR10, CR16, and CR17; a full-wave bridge rectifier supplying unregulated DC to the +5V regulator.
- CR1 and CR2; a full-wave rectifier supplying unregulated +22V DC as a BLM power supply.
- CR3 and CR4; a full-wave rectifier supplying unregulated DC to the +12V regulator.
- CR5 and CR6; a full wave rectifier supplying unregulated DC to the -12V regulator.
- CR7 and CR8; a full-wave rectifier, provides approximately 15V DC which is referenced to the output of CR5 and CR6 and supplies approximately -33V DC as a BLM power supply.

+5V Regulator

The +5V Regulator filters and regulates the raw DC voltage at the output of the CR9, CR10, CR16, CR17 full wave rectifier to +5V DC, nominal. Ripple on the the rectifier output is reduced to a level acceptable to the regulator by four large electrolytic capacitors. Bleeder resistors are connected across the rectifier output to reduce the range of no-load to full load voltage which the regulator must accommodate.

With the incoming air to the Logic Power Supply module at +29°C, and when adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the output voltage, including noise and ripple, should not vary more than ±3%, while the load current is between 32 and 80 amperes. The regulator has a negative temperature coefficient of voltage; as the temperature rises, the voltage drops.

Each of the regulators in the Logic Power Supply controls the output voltage by dynamically varying the series impedance of from two to twenty series regulator transistors. The output voltage is compared with a stable reference voltage, in a differential amplifier. The differential amplifier output signal is buffered from the relatively low impedance of the series regulators by one or two emitter followers and then used to control the series impedance, regulating the output. Refer to Fig. THEORY. 4 and sheet 15 of the -914 logic.

The reference voltage for the +5V regulator is derived from a voltage divider which is connected across the +5.6V reference for the +12V regulator. The +4V reference is applied to the base of Q43. The +5V output voltage is applied across a voltage divider, and the signal for comparison with the reference is tapped off the divider by the output adjustment potentiometer and applied to the base of Q42.

The emitter to collector current for both stages of the differential amplifier flows through the common emitter resistor, R63. Any drift in the operating point in one of the stages will produce an approximately equal opposite drift in the other stage, because the total current flow through R63 remains virtually constant. The differential amplifier is, therefore, quite stable, and well suited for this application.

As an example of the operation of the regulator, assume that due to a decrease in the load current, or an increase in the line voltage, the output voltage tends to increase:

1. The voltage on the base of Q42 increases, increasing the conduction of Q42 and reducing the conduction of Q43.
2. The voltage drop across Q42's collector load resistor, R37, increases, and the collector voltage decreases (less positive).

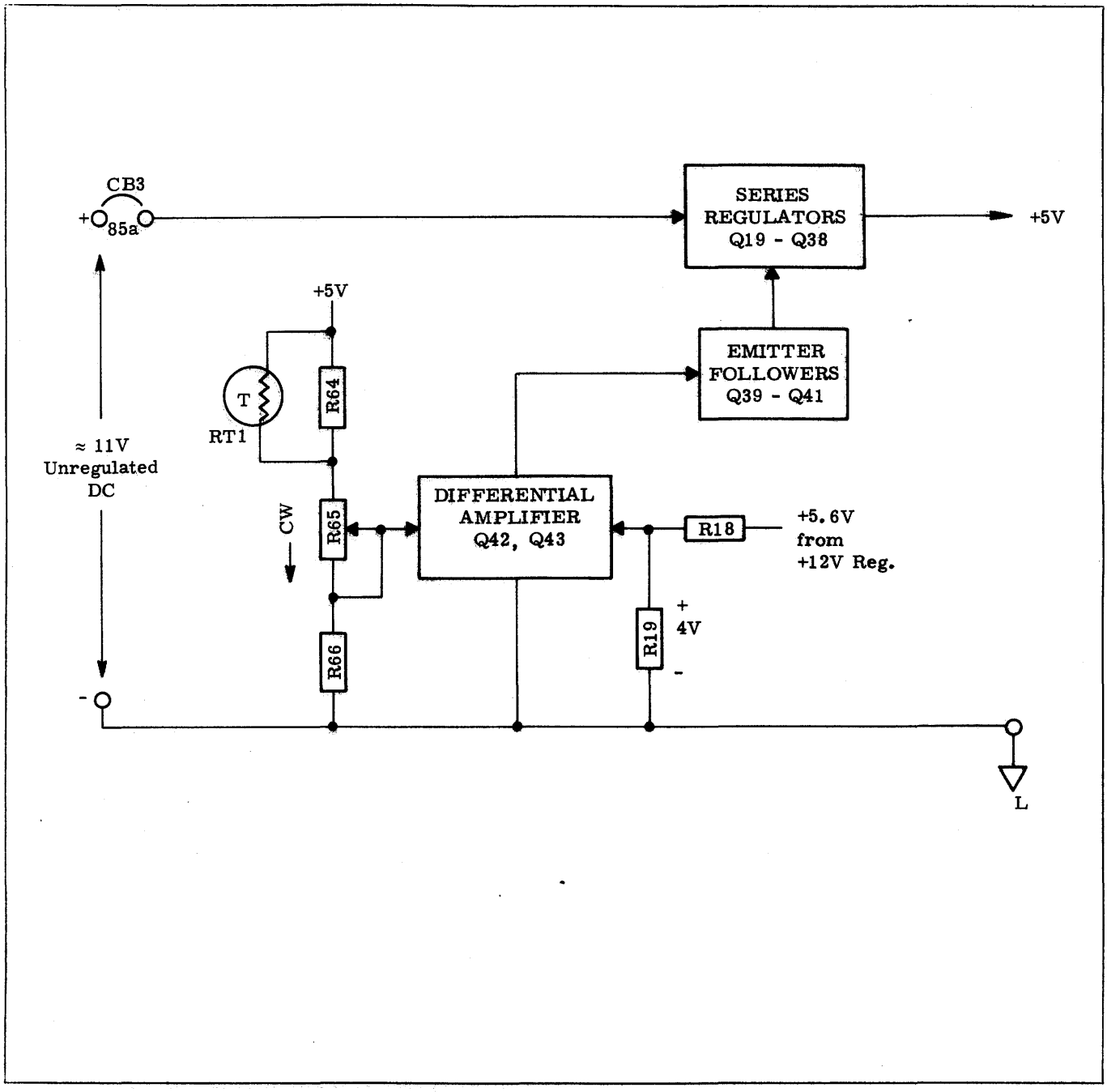


Fig. THEORY.4 +5V Logic Regulator

3. The emitters of Q39 through Q41 become less positive, decreasing the conduction of the series regulator transistors, increasing their impedance, and holding the output voltage within tolerance.

As is typically required in regulators of this type, several components are included to limit the high frequency and transient response of the regulator. Without such compensation, under various conditions of line voltage and load impedance, such regulators may oscillate. Such components are: R72 and C23 on the base of Q42; R71 and C22 from the collector to the base of Q41; and C24 from the collector to the base of Q39.

Thermistor, RT1, in the output sensing voltage divider, provides the negative temperature coefficient of voltage. At 29°C, RT1 has a value of about 82 ohms. As temperature increases, the resistance of the thermistor decreases, pulling the base voltage on Q42 up, and decreasing the output voltage.

Each of the series regulator transistors has a small resistance in series with its emitter. These resistors provide a small amount of degeneration, which helps to equalize the current carried by each transistor. If two or more such transistors were connected directly in parallel, the one with the highest current gain would try to carry most of the current, until it overheated and destroyed itself.

Clamp diode, CR15, between the +5V output terminals, prevents the positive terminal from swinging negative due to a fault in the regulator or load or during the power-on or power-off sequence. Negative voltage on the +5V bus could damage components.

+12V Regulator

When adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the +12V regulated voltage, including noise and ripple, should not vary more than ±3%, while the load current is between 9 and 30 amperes.

The operation of the +12V Regulator is virtually the same as the +5V Regulator. Refer to Fig. THEORY. 5 and sheet 16 of the -914 logic.

This regulator has no intentional voltage-temperature coefficient and the output sensing voltage divider does not incorporate a thermistor. The reference voltage is derived from a shunt regulator using a 5.6V Zener diode. R18 and R19 serve as a voltage divider to tap off the reference voltage for the +5V regulator.

The collector supply for Q11 and Q10 is provided by the +22V BLM power supply, through a set of switch contacts incorporated in the CB1 circuit breaker. (Note: If the +22V fuse, F1, blows, both the +12V and +5V outputs will go to zero.)

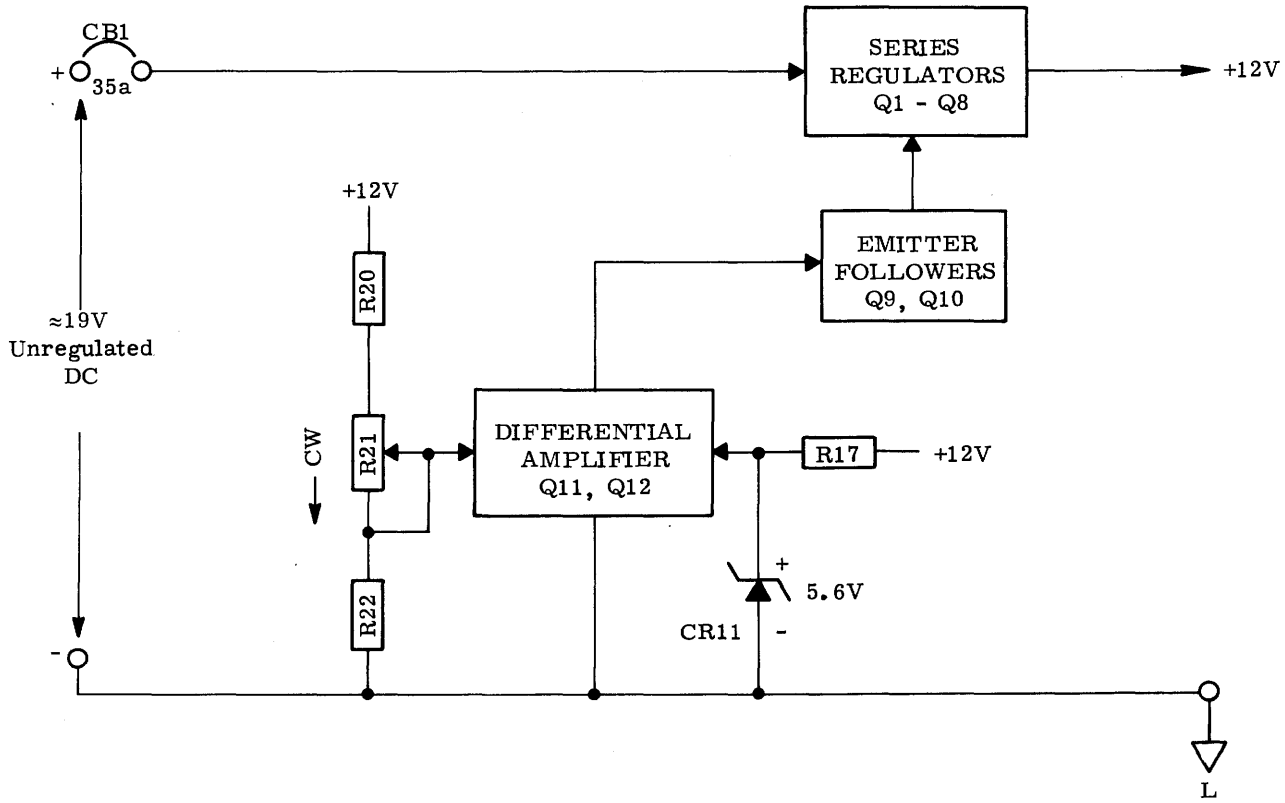


Fig. THEORY. 5 +12V Logic Regulator

Transient response and frequency response compensation is provided by C17, C18, R15, and C2.

Since the series regulators do not dissipate as much power in this regulator as in the +5V Regulator, eight series regulators are used. Each uses a small emitter resistor to help equalize the current carried by each transistor.

Clamp diode, CR18, between the +12V output terminals, prevents the positive terminal from swinging negative due to a fault in the regulator or load. Negative voltage on the +12V bus could damage components.

-12V Regulator

When adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the -12V regulated voltage, including noise and ripple, should not vary more than ±3%, while the load current is between 2.1 and 7 amperes.

The operation of the -12V Regulator is similar to the +5V and +12V regulators, except that the output is of the opposite polarity, which requires that the series regulator transistor connections be reversed. PNP transistors are used in the differential amplifier, rather than NPN. Fig. THEORY. 6 is a simplified schematic of the regulator. The complete schematic is on sheet 17 of the -914 logic.

The -33V BLM power supply is used as a collector supply for the differential amplifier and to supply operating current for Q15. CR13 and Zener diode, CR14, are connected in series with R32. The -33V supply draws current through CR13, CR14, and R32. CR13 is , therefore, forward biased holding the emitter of Q16 slightly more negative than -12V. The voltage drop across the Zener diode holds the junction of R32 and R33 at approximately -18V, which is used as the collector supply for the differential amplifier. (Note: If the -33V fuse, F2, flows, the -12V output will go to zero.)

As an example of the operation of this regulator, assume that due to a decrease in the load current, or an increase in the line voltage, the output voltage tends to increase (become more negative):

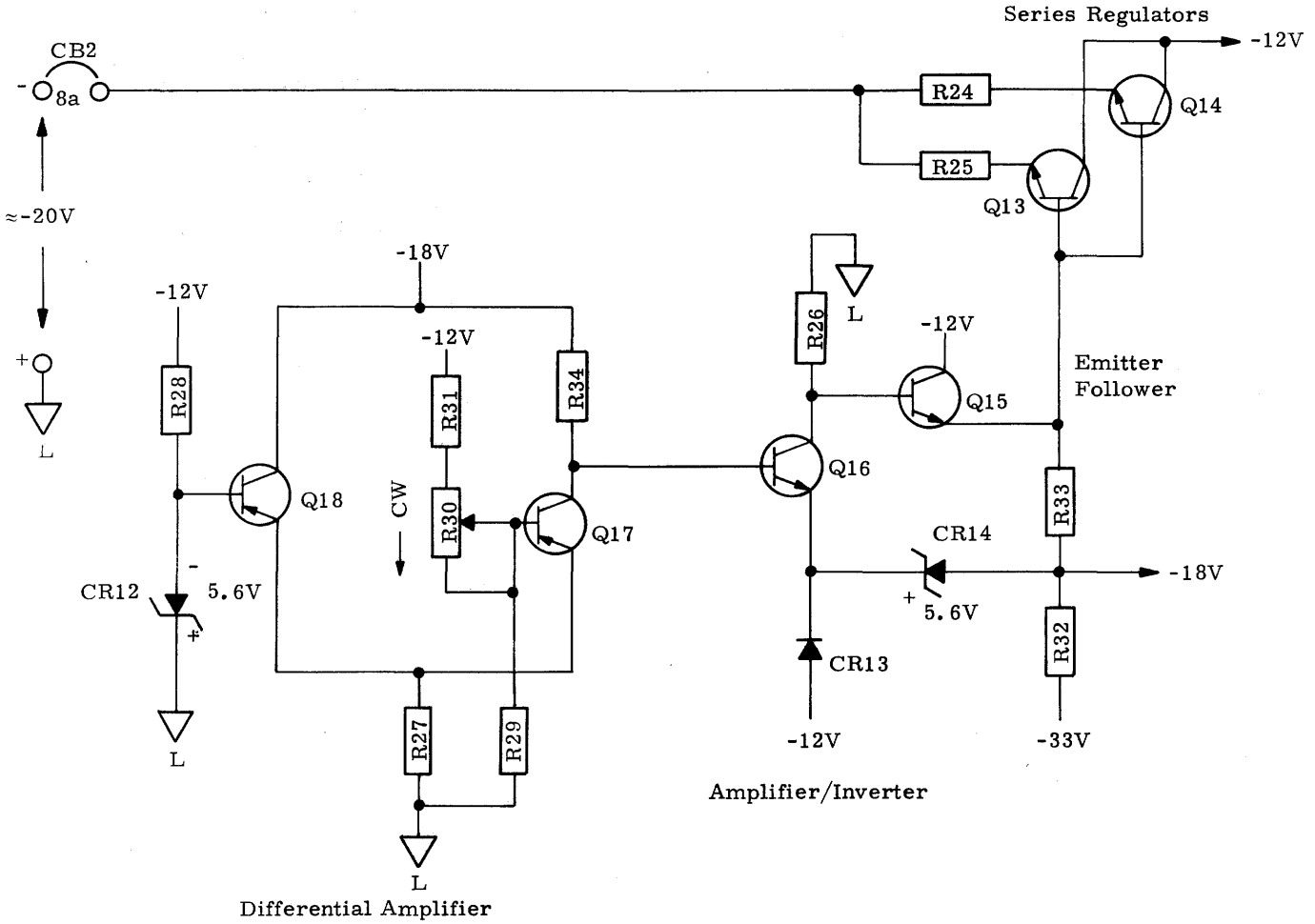


Fig. THEORY. 6 Simplified Schematic, -12V Logic Regulator

1. The voltage on the base of Q17 increases, increasing its conduction, and causing a corresponding decrease in the conduction of Q18.

2. The voltage drop across R34 increases and the voltage on the collector of Q17 decreases (becomes less negative).

3. As the base of Q16 becomes less negative, its conduction increases.

4. The increased voltage drop across R26 swings the base of Q15 in the negative direction. The bases of the series regulator transistors become more negative, increasing their impedance, and holding the output voltage within tolerance.

Clamp diode, CR19, between the -12V output terminals, prevents the negative terminal from swinging positive due to a fault in the regulator or load. Positive voltage on the -12V bus could damage components.

Logic Voltage Crowbars

Three crowbar circuits are provided on the exterior of the Logic Power Supply module. These circuits prevent the bus voltages from increasing to a point where components in the load circuitry could be damaged. The operation of all three circuits is the same. Refer to sheet 21 of the -914 logic.

The transistor and the SCR in each crowbar circuit are normally not conducting. The zener diodes in the base circuit of each transistor hold the bases at a fairly constant voltage with respect to the negative bus. Should the voltage between two buses increase to the trip point, the emitter of the transistor will be more positive than the base, turning the transistor on. The conducting transistor raises the gate of the SCR to the firing point, which places a virtual short across the busses, and the DC circuit breaker in the power supply trips. The BLM senses the "crowbarred" voltage as an undervoltage and the Sequencer initiates a shut-down.

Firing voltages and circuit breakers tripped are as follows:

Bus	Firing Point	Circuit Breaker
+12V	+13V to +16V	CB1
-12V	-13V to -16V	CB2
+5V	+5.5V to +6.0V	CB3

MEMORY POWER SUPPLY

The Memory Power Supply provides +12V DC and from -9V to -14V DC to the Core Memory. The negative DC output is adjusted as required for optimum Core Memory operation, and therefore, may not be at the nominal -12V DC level. The Memory Power Supply module is located at the rear of the CSU, just above the Sequencer. If a second 16K core stack is implemented, a second -12V memory regulator is mounted just above the basic Memory Power Supply.

The Memory Power Supply also provides transformer T2 (sh. 22 of -914 logic), which converts non-sequenced 115VAC to 28VAC for the core stack heaters.

Schematics, parts lists, and component lay-out sketches for the Memory Power Supply are provided on sheets 22 through 29 of the Power Module Logic, 70C180914.

AC Input and Rectifiers

The memory power supply schematic on sheet 22 of the -914 logic indicates that the AC input may be 115V or 230V, and 50 Hz or 60 Hz. In this application, the AC input is 115VAC at 60 Hz, as supplied via CB1, K1A contacts, and filter, FL4, in the AC panel.

Step-down transformer, T1, incorporates a ferroresonant winding which improves the line regulation of the transformer. C10 forms a resonant circuit with the inductance of the winding. Each of the four T1 secondaries drives a full-wave rectifier. Large filter capacitors are connected across the rectifier outputs to reduce ripple. Unregulated DC for the regulators is provided by one of the rectifiers, while the remaining rectifier provides bias voltages for the amplifier circuits in the regulators.

+12V Regulator

When adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the +12V regulated voltage, including noise and ripple, should not vary more than ±3%, while the load current is between 0 and 12 amperes.

The operation of the +12V Regulator is virtually the same as the +5V and +12V regulators in the Logic Power Supply. The schematic is provided on sheet 22 of the -914 logic. Fig. THEORY. 7 is a block diagram of the regulator.

-12V Regulator

When adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the -12V regulator output voltage, including noise and ripple, should not vary more than ±3% from the adjusted value (-9V to -14V), while the load current is between 0 and 12 amperes.

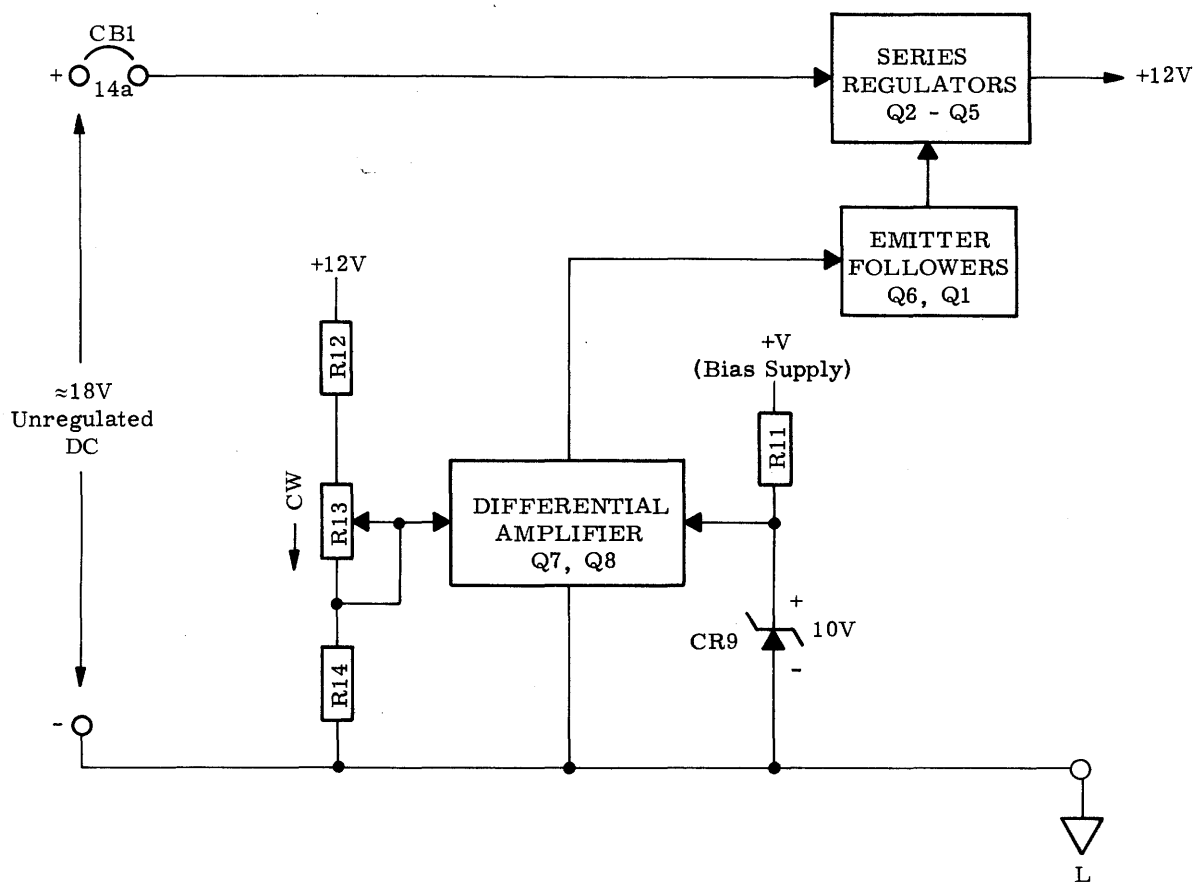


Fig. THEORY. 7 +12V Memory Regulator

The operation of the -12V Regulator is virtually the same as the -12V Regulator in the Logic Power Supply. As in the case of the -12V Logic Regulator, the series regulator transistors are connected with the collectors supplying the output, and PNP transistors are used in the differential amplifier. Fig. THEORY. 8 is a block diagram of the regulator.

Second Memory Regulator

When a second 16K core stack is implemented, a second -12V regulator is required to provide a separately adjustable negative operating voltage (-9V to -14V) to that stack. The schematic of the second regulator is on sheet 27 of the -914 logic.

The second regulator is electronically identical to the primary -12V Regulator. The unregulated (bulk) DC voltage is supplied by the same rectifier, CR5 and CR6. The AC voltage for the bias supply is supplied by the same transformer secondary as supplies the primary bias supply AC.

Memory Voltage Crowbars

An overvoltage crowbar is connected between the +12V and -12V memory supply outputs. If the second memory regulator is implemented, a second crowbar is connected between the +12V and -12V outputs of that supply. The primary crowbar is installed in the Memory Power Supply module and the schematic is on sheet 22 of the -914 logic. The second crowbar is in the second regulator module and its schematic is on sheet 27 of the logic.

In the primary crowbar, when the sum of the two output voltages is great enough to cause the 25.6V zener diode string to go into reverse conduction, the base of the transistor swings sufficiently negative to cause the transistor to conduct, raising the voltage on the gate of the SCR to the firing point.

In the second crowbar, the zener diode string goes into reverse conduction when the sum of the output voltages is excessive, raising the gate of the SCR to the firing point.

When either of the SCR's fires, both of the bulk DC circuit breakers on the memory supply module trip. The BLM senses the "crowbarred" voltages as undervoltages and the Sequencer initiates a shutdown. The crowbars fire when the sum of the output voltages is between 27 and 31V.

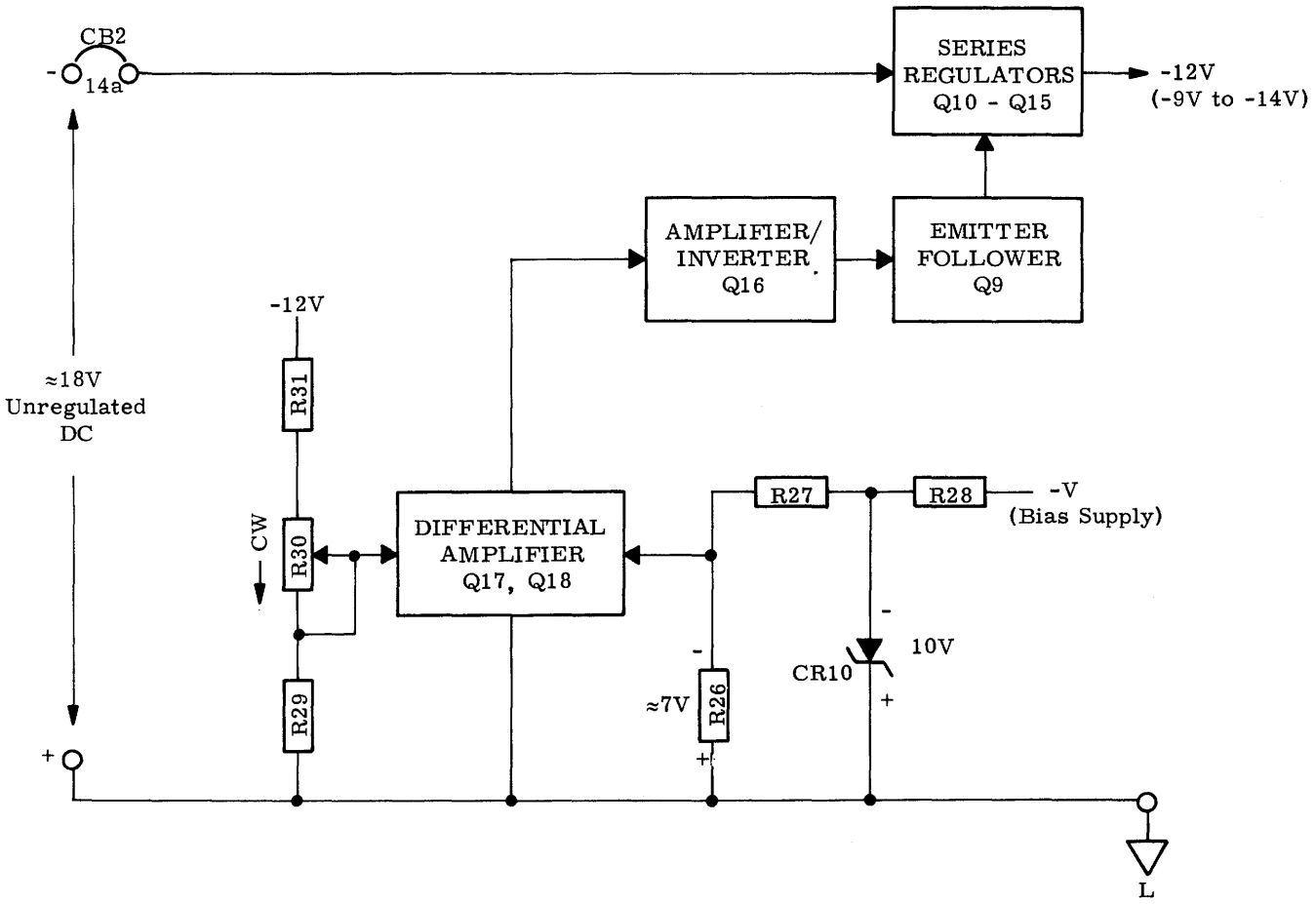


Fig. THEORY. 8 -12V Memory Regulator

BUS LEVEL MONITOR

The Bus Level Monitor (BLM) senses the regulated outputs of the Logic Power Supply and the unregulated (bulk) DC inputs to the Memory Power Supply. If one or more of the monitored voltage decreases to a point where the operation of the computer may be adversely affected, the BLM inhibits Core Memory accesses, disables Core Memory write current, and initiates a shut-down of sequenced AC power. A block diagram of the BLM appears on sheet 140 of the Arithmetic Unit logic, 70C180955.

The BLM is also involved in the power on sequence, in that it withholds the Sequencer K2 relay enable signal until the monitored supplies are in tolerance.

Four undervoltage sensor circuits are employed. Three of these monitor the three Logic Power Supply outputs, +5V, +12V, and -12V. The fourth sensor monitors the sum of the bulk DC memory regulator supply voltages. The sensor outputs are applied to a BLM control circuit which generates and controls the timing of the four BLM signal outputs.

The BLM circuitry is installed on a single PX1000SBMA1 board, which is located in slot 17 of B panel in the CSU.

BLM Power Supply

The BLM power supply is independent of the supplies which the BLM monitors. +22V DC and -33V DC, which originate in the Logic Power Supply, are shunt regulated to +12V DC and -20V DC by Zener diodes as shown on sheet 36 of the Power Module Logic, 70C180914.

Undervoltage Sensors

The undervoltage sensors consist of four differential amplifiers which compare the monitored voltages with a stable reference. A schematic of the sensor circuits appears on sheet 36 of the -914 logic.

The undervoltage trip points for all of the sensors are adjusted by means of multi-turn trimmer potentiometers. The adjustment procedures are provided in the Power Supplies section of the Computer Mainte-

nance manual. The potentiometer in each of the sensors, except the -12V sensor, is adjusted so that with the monitored voltage in tolerance, the odd numbered transistor is biased on. In the case of the -12V sensor, the odd numbered transistor is biased off. During normal operation, then, Q11, Q1, Q3 and Q10 are conducting.

During normal operation, the current drawn across the common emitter resistor by the conducting transistor drops enough voltage to hold the opposite transistor off. Q12, Q2, Q4, and Q9, therefore, are normally not conducting.

The sensor outputs are connected from the collectors of the conducting transistors to the BLM control circuit. Any sensor output at +8.5V or less appears to the control as an "in tolerance" signal. When conducting, the collector of Q11 rests at about -12V. Each of the other sensors rests at about +5V to +7V.

When the voltage monitored by a sensor decreases to the trip point, there is no longer sufficient base/emitter current to hold the normally conducting transistor on. As the normally conducting transistor goes off, the opposite transistor turns on, which increases the reverse bias on the first transistor and maintains about the same current flow through the common emitter resistor. The normally conducting transistor then makes a very abrupt transistion from on to off, its collector rises rapidly to about +11V, and the BLM control initiates a controlled shut-down.

BLM Control

The BLM control participates with the Sequencer in the power on sequence and the shut-down sequence. The control also provides three "BLM good/BLM bad" signals to Core Memory and the Arithmetic Unit. The "good" or "bad" state of the four outputs is determined by the four undervoltage sensor inputs. An output timing diagram appears on sheet 37 of the logic. On that diagram, the lower level is the "bad" state and the upper level is the "good" state.

During the power on sequence, the control progresses from the point where no voltages are available to operate the BLM to the period when all four outputs are in the "good" state. None of the outputs can be "good" except when relay K1 is energized because a set of contacts holds the A, B, and C outputs at ground when the relay is not energized. The driver which pulls K1 is transistor Q13. Q13 also pulls the "BLM good" relay, K2, in the Sequencer.

The five inputs to the control are applied through an OR gate. If any of the inputs is above about +8.5V, the diode is forward biased, and Q19 cannot conduct. When all of the inputs are below 8V, Q19 does conduct. Q19, then, begins to conduct when all of the sensors are in the "in tolerance" state. In turn, Q5 turns on, Q16 off, and Q13 turns on, allowing K1 to energize. When K1 does energize, in one or two milliseconds, BLM A goes "good". During turn-on, the regulators may overshoot their "in tolerance" range briefly. Capacitor C9 delays the turn-on of Q5 for about 13 milliseconds to allow the voltage to stabilize.

Q21 and Q24 are initially conducting, which grounds the BLM B and BLM C outputs. The rise of these two outputs is delayed until Q21 and Q24 are turned off. During the turn-on sequence, when Q16 turns off, its rising collector voltage turns on Q6. The negative swing at the collector of Q6 is coupled across C14 to the base of Q7, holding Q7 off until C14 has charged sufficiently to allow the base of Q7 to rise above the turn-on point. This takes about 70 milliseconds. After the delay, Q7 turns on, Q8 goes off, Q20 goes on, and Q22 goes off, allowing BLM B to go "good". Q8 also pulls Q23 on and Q24 off, allowing BLM C to go "good".

Should a sensor detect an undervoltage, Q19 turns off, and all of the conditions previously established are reversed. Q21 is provided to insure that BLM B goes down immediately to prevent further accesses to core. As soon as Q6 goes off, Q21 goes on, grounding the B output. The A and C outputs go down about 2 milliseconds later when K2 drops, closing the grounded contacts. The Q24 turn-on cannot occur that soon, because the discharge of C14 holds Q7 on for several milliseconds.

PERIPHERAL AND RELAY POWER SUPPLY

The Peripheral and Relay Power Supply provides +28V DC for use throughout the CSU as a relay and control power supply. The +28V output is also used in the I/O Buffer peripheral device drives and controls to supply relay control signals to the peripheral devices. The Peripheral and Relay Power Supply module is located at the rear of the CSU, near the lower right-hand corner, as viewed from the rear.

When adjusted and maintained per the Power Supplies section of the Computer Maintenance manual, the +28V regulated output voltage should not vary more than ±3%.

Schematics, parts lists, and component lay-out sketches for the power supply are provided on sheets 30 through 33 of the Power Module Logic, 70C180914.

AC Input and Rectifier

The +28V power supply schematic on sheet 30 of the -914 logic indicates that the AC input may be 115V or 230V, and 50 Hz or 60 Hz. In this application, the AC input is 115VAC at 60 Hz, as supplied via a set of K2A contacts, and filter, FL5 in the AC panel.

Step-down transformer, T1, incorporates a ferro-resonant winding which improves the line regulation of the transformer. C1 forms a resonant circuit with the inductance of the winding. Each of the two T1

secondaries drives a full-wave rectifier. The output of CR1 and CR2 is added to the +28V output to provide approximately +38V to the amplifier in the regulator. CR3 and CR4 provide approximately 39V DC as the unregulated supply for the regulator. The capacitors across each of the rectifier outputs reduce ripple to a level acceptable to the amplifier and the regulator.

+28V Regulator

As with the regulators discussed previously, the +28V output is regulated by series regulators transistors whose series impedance is dynamically adjusted by an output sensing amplifier. In this regulator, a single amplifier stage is employed.

A 5.1V Zener diode regulates and stabilizes the amplifier reference voltage. The amplifier operating point and the output voltage are adjusted by a potentiometer in a voltage divider connected across the regulated output.

For an example of the regulator operation, assume that the output voltage tends to increase due to a decrease in load current or an increase in line voltage. The increased voltage on the base of Q12 increases its conduction. The voltage drop across R1 increases, lowering the Q21 collector voltage, and the voltage on the base and emitter of Q11. The decreased voltage on the bases of the ten series regulator transistors increases their impedance, holding the +28V within tolerance.

The ten series regulator transistors have small resistances in series with their emitters to provide a small amount of degeneration and distribute the load current more evenly among the transistors.

In this application, the negative side of the 28V supply is designated as the peripheral supply common. The common and the +28V output pass through the 28V crowbar to be distributed to the 28V loads in the CSU.

28V Crowbar

The 28V crowbar provides two functions: 1) Shorts the 28V output and trips circuit breaker, CB1, if the output voltage is excessive. 2) Abruptly terminates the 28V output when K2 in the Sequencer drops, indicating that the logic and or memory voltages are out of tolerance. The first function will occur only in the event of a 28V supply or load fault. The second occurs in either an intentional or unintentional shut-down, and prohibits faulty peripheral or control circuit operation due to out of tolerance voltages in the computer. The crowbar schematic is on sheet 31 of the -914 logic.

Silicon controller rectifier, SC1, is the sequenced shut-down crowbar. During normal operation, the BLM relay driver, which is holding sequencer relay K2, is also holding the cathode of CR3 at ground. CR3 then conducts, dropping 28V across the 200 ohm resistor, zener diode CR2 does not conduct, and the gate on SC1 remains near ground potential. Early in a power-off sequence, the relay driver turns off, and CR3 is reverse biased. This causes the zener to go into reverse conduction, pulling SC1's gate to the firing point. SC1 pulls the 28V output virtually to zero volts within one or two microseconds. SC1 will conduct so long as the voltage is sufficient to provide holding current. It does not conduct long enough to damage any power components, because sequenced AC is removed from the input to the 28V regulator shortly after the crowbar fires.

SC2 provides overvoltage protection. Should the power supply voltage reach about 34 volts, Zener diode, CR1, will go into reverse conduction, drawing current across R3, and raising the SCR gate to the firing point. The SCR places a virtual short across the output, tripping CB1. CB1 will also trip if the load current exceeds 10 amperes.

The red lamp, which appears on sheet 31 of the logic, extends through the rear panel of the 28V power supply module and is visible from the rear of the CSU. It is illuminated when 28V is available to the loads.

ANALOG RELAY POWER SUPPLY

The Analog Relay Power Supply provides +15V DC to the Analog Input Subsystem for use in relay and control circuitry. This supply features line regulation better than $\pm$0.1%, load regulation better than $\pm$0.1%, and noise and ripple output of 3 millivolts peak-to-peak or less. This supply is included only in CSU's implementing analog inputs. The schematic appears on sheet 35 of the -914 logic.

The analog relay supply module (F) is located at the rear of the 60 inch section of the CSU, near the upper right-hand corner, as you face the rear of the CSU.

Step-down transformer, T1, provides power to two rectifiers; a half-wave bias supply rectifier, CR7, and the main rectifier, a full-wave bridge consisting of CR8 through CR11.

The unregulated DC, at about 27V, is dropped to a regulated 15V by five series regulator transistors, Q8 through Q12. Driver stages, Q5 and Q6, are emitter followers which couple the control signal from the amplifiers to the series regulators. Q1 and Q2

are the voltage regulation amplifiers and Q3 is an output current limiting amplifier. R1 is the output voltage adjustment and R17 is the current limit adjustment.

Zener diodes, CR1 and CR6, are shunt regulators, which regulate the DC output from the bias rectifier, CR7, to approximately +21V and +24V, respectively, with respect to the -V output terminal (15V common). These voltages supply the amplifiers.

The voltage regulator (error) amplifier monitors the output voltage which is across the voltage divider in the Q1 base circuit. The amplifier corrects the operating point and series impedance of the series regulator transistors to maintain the output voltage within tolerance.

The current limit amplifier, Q3, is normally not conducting. Should the load current increase to the limit, (approximately 15.4 amps. at 40° C) the base of Q3 will be raised to the point where the transistor begins to conduct, lowering its collector voltage, and increasing the impedance of the series regulators. If the load impedance continues to decrease, Q3 continues to decrease both the output voltage and the current. If the load current is again reduced below the limit, Q3 will cease conducting and the voltage regulator will again take over.

The current limit is adjusted at the factory and sealed by soldering the wiper of R17 in the final position. This adjustment is not normally made in the field.

A 10 amp. fuse in the AC input line provides protection from internal power supply failures. A thermostat in the AC line removes power from the transformer if the maximum temperature of the series transistor heatsink is exceeded. The thermostat contacts close again when the temperature returns normal.

22V ANALOG POWER SUPPLIES

The 22V Analog Power Supply module provides +22V and -22V DC to the Analog Input Subsystem for use by the A/D Converter and Gain Ranging Amplifier. The two supplies are contained in a single module (FF) at the rear of the 60 inch section of the CSU, near the upper right-hand corner, as you face the rear of the CSU. The two supplies are designated as the "master" (+22V) and the "slave" (-22V). They share the same step-down transformer and use identical circuitry except that the slave supply has one additional diode (CR112). This module is included only in CSU's implementing analog inputs. The schematics of the master and slave supplies appear on sheets 34 and 34.1, respectively, of the -914 logic.

These supplies feature line regulation better than ±0.05%, load regulation better than ±0.05%, and noise and ripple output of one millivolt peak-to-peak or less. The voltage outputs are adjustable but are set at the factory to 22.0V and sealed.

Fig. THEORY.9 is a block diagram of the +22V master supply. The diagram applies to the -22V slave supply if the analog common symbol and the 22V output are exchanged. The selection of positive and negative voltage outputs from the supplies is made in the same manner; the negative output is tied to analog common on the master, and the positive output is tied to analog common on the slave.

For the remainder of this discussion, references are to the master (+22V) supply, and are equally applicable to the slave supply if 100 is added to the reference designators (CR1 becomes CR101).

The transformer secondaries drive two rectifiers on each supply; the half-wave bias supply rectifier CR1, and the full-wave main rectifier consisting of CR14 and CR15. The bias supply provides DC voltage to operate much of the circuitry in the integrated circuit module, IC1, which contains much of the active electronics for the regulator. The main rectifier provides the raw DC power to the drivers and the series regulator transistor, Q3.

When the supplies are operating normally with the load current below the upper limit, the voltage comparator signal controls the operating point of the amplifier, which in turn corrects the series impedance of Q3 to maintain the output voltage within tolerance.

Should the load current increase to the limit (about 660 ma at 40° C), the output of the current comparator takes over control of the amplifier. If the current goes beyond the limit, the current comparator signal causes the amplifier to increase the series impedance of Q3, which reduces the output voltage and the current. If the load impedance returns to normal, the voltage comparator again takes over, providing the regulated 22V output.

R13 is adjusted for the proper current limit at the factory and sealed. This adjustment is not normally made in the field.

A 1-1/2 amp. fuse in the negative output lead provides protection from internal power supply failures. A thermostat in the AC line removes power from the transformer if the maximum temperature of power supply heatsink is exceeded. The thermostat contacts close again when the temperature returns to normal.

ANALOG POWER SUPPLY CROWBARS

A small crowbar assembly is mounted near the output terminal board on each of the three analog power supplies (+15V, +22V, and -22V). The crowbars protect the load circuits from excessive voltages on the power supply outputs, which might damage components in the load circuitry. Schematics of the crowbars appear on sheet 35.1 of the -914 logic.

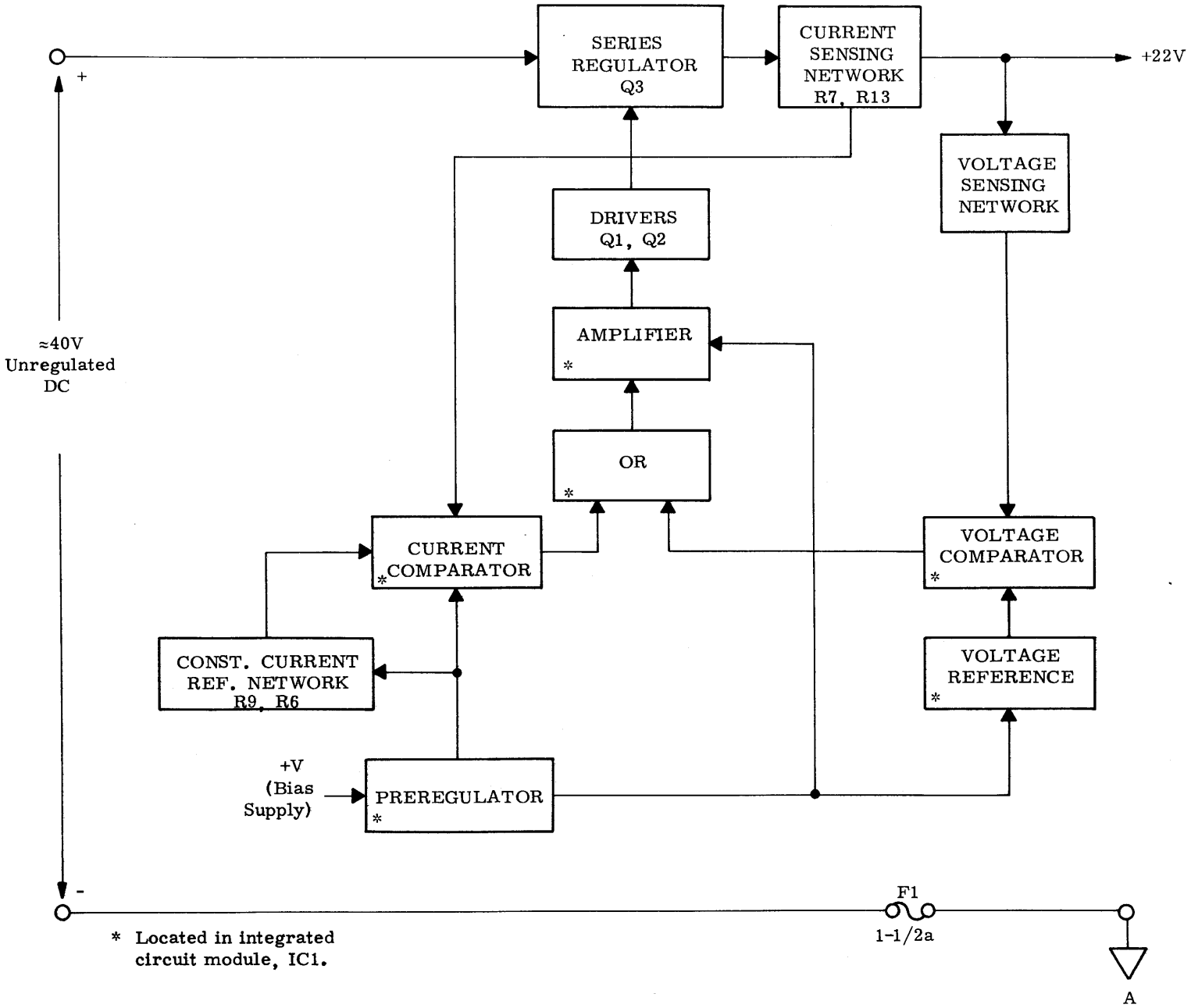


Fig. THEORY. 9 +22V Analog Power Supply

All three circuits operate identically. The only differences are the component values. The 22V circuit is used twice and the 15V circuit is used once. During normal operation, both transistors are off. If the power supply output voltage exceeds the limit, sufficient current is drawn through the base and emitter of Q1 to turn it on. This lowers the voltage on the collector of Q1 sufficiently to turn Q2 on, raising the gate of the SCR to the firing point. The conducting SCR places a virtual short across the supply output, the current limiter in the supply takes over, the output voltage decreases, but the current through the SCR remains above the holding current for the device.

Once a crowbar is fired, operation cannot be restored without turning the supply off, because the SCR cannot be turned off until the current through it drops below the holding level. If power is turned off, the fault cleared, and power restored, the crowbar circuit will revert to the normal state.

The trip points for all three crowbars are adjusted by a trimmer resistor in the base circuit of Q1. R4 is a resistor with a negative temperature coefficient which corrects the bias on Q1 to compensate for changes in the current gain of the transistor with temperature.

The procedure for adjusting the crowbar trip points is in the Power Supplies section of the Computer Maintenance Manual. Trip points for the two voltages are:

+22V, -22V; 23.6V.

+15V; 18.2V.

AIR FLOW SENSORS

Seven electronic air flow sensors monitor the flow of cooling air from the seven blowers in the Central System Unit. One of the seven sensors is a master containing a relay driver transistor, which holds sequencer relay K11 energized when all seven blowers are running. A schematic of the sensor circuitry appears on sheet 38 of the -914 logic.

The master sensor is installed in the blower 1 housing, just below the logic power supply. The slave sensor outputs are connected to a transistor base circuit in the master through OR gating diodes. Each sensor, including the one in the master, is simply a voltage divider, consisting of a fixed resistor and a thermistor. Enough power is dissipated in each thermistor to cause it to produce heat. If the blower is not running and carrying the heat away, the resistance of the thermistor increases to a point where the OR diode starts conducting, turning the first transistor on, the second transistor off, dropping K11, and opening its contacts.

If the K11 contacts open, the CAB TEMP light on the Programming and Maintenance Console lights. The same light comes on if any of the thermostats on the logic or memory supplies opens, as they are in series with the K11 contacts. The lamp driver logic appears on sheet 142 of the Arithmetic Unit logic drawing, 70C180955.

GROUNDING

The cabling, shielding, and grounding scheme for the computer system has been carefully designed to reduce noise throughout the system to the greatest extent possible. A schematic of the grounding system in the CSU is provided on sheet 42 of the -914 logic. The routing and connections shown on that schematic should not be altered, except as permitted by the notes on the schematic. External cables and grounding should be installed per the drawings shipped with each system, and should not be changed unless such changes are approved by the cognizant Headquarters Engineer.

A detailed description of the primary power connection and system grounding requirements is provided in the Site Planning Manual.

In general, there are three basic requirements for a proper grounding system:

1. It must protect personnel safety. This means that while complying with all other grounding system requirements, all conducting surfaces on the equipment which are subject to personal contact must ultimately be returned to earth by one or more conductors.

2. Ground loops must be avoided. This means that any conductor or conducting surface should be returned to ground at one point only. Loop currents can be produced in conductors which are grounded at more than one point, due to differing potentials at the two points. Such currents can produce excessive noise in the circuitry.

3. Grounding subsystems of differing quality or noise levels, while ultimately returning to earth, should be kept separate. Note that on sheet 42 of the logic, and throughout the logic, differing symbols for grounds and power supply commons are used, i.e. analog common, peripheral common, and logic common.

REFERENCE POWER SUPPLIES, 42 VOLTS DC

Power Supplies, Model No. 4396A and 4847AS07, are low voltage, medium current, highly regulated DC power supply modules, with high long term stability. These supplies are used principally in the analog input subsystem, for such purposes as Resistance-Temperature Device supplies, slide wire supplies, analog generator supplies, etc.

Two versions of this module are available:

Model No. 4396A100 - 42V DC, 1.0 amperes, no overvoltage protection; part no. 68A8455P51011

Model No. 4396A101* - 42V DC, 1.0 amperes, with overvoltage protection; part no. 68A8455P51111

The rectifying and regulation circuits are contained within a hermetically sealed, non-repairable module. A bracket is attached to this module which contains a potentiometer, for adjustment of the output voltage to the exact value required, and a terminal board, to which external connections such as, primary AC power, DC output, and remote sensing lines are made. A small fan is installed near the module, so that the heat sink may be maintained below its maximum temperature.

* This version is designated model no. 4847AS07 for use in 4010 computer systems, were it serves as the analog output reference supply.

Either the positive or negative output of these supplies may be connected to logic ground, providing either -42VDC or +42VDC at the output.

Model 4396A101 has an overvoltage protection circuit which trips at approximately 48.5V, setting the output to zero. Removal of the cause of the overvoltage, internal or external to the power supply, will allow normal operation to resume.

Both supplies have shorted output protection. Removal of a short from the output will allow normal operation to resume.

The essential characteristics of both supplies are as follows:

Regulation, Line and Load; 0.05% of nominal output voltage, maximum.

Ripple; 0.002% of nominal output voltage, maximum.

Transients; 10% of initial output, maximum, no load to full load.

Wiring external to the sealed module is shown on drawing no. 68B995217.

INSTRUCTION MANUAL

OVERVOLTAGE PROTECTORS

LMOV-1, LMOV-2 AND LMOV-3

LAMBDA

LAMBDA ELECTRONICS CORP.–MELVILLE, L. I., N. Y.

GENERAL INFORMATION

The Lambda overvoltage protector prevents damage to the load caused by excessive power supply output voltage due to improper adjustment, improper connection, or failure of the power supply. Load protection is accomplished automatically by effectively short circuiting the output terminals of the power supply when a preset limit voltage has been exceeded.

Although designed specifically for use with Lambda LM series power supply packages A through E, the overvoltage protector can be used with other similarly rated current-limited power supplies when certain basic precautions are observed. When mounted on a 1/16 inch thick aluminum heat sink of at least 20 square inches, in an ambient temperature of up to 80°C, the unit can withstand a fault current of 35 amperes for a period of one minute, or of 80 amperes for a period of one second. Consult the factory for specific other-model application information.

The overvoltage protector will not provide protection against overvoltage conditions caused by storage batteries or other power sources used in conjunction with the Lambda power supply. Whenever additional power sources must be used with the Lambda power supply and overvoltage protector combination, consult the factory for proper installation information.

The adjustable voltage range of each overvoltage protector for specific power supply output voltage ranges are listed below.

	ADJUSTABLE VOLTAGE RANGE	SUGGESTED P.S. OUTPUT VOLTAGE RANGE
LMOV-1	3 - 8v	1.8 -6.0v
LMOV-2	6 - 20v	4.2 - 16.0v
LMOV-3	18 - 70v	15.0 - 60.0v

THEORY OF OPERATION

Divider network R1, R2, R3, R4 and R7 attenuates the power supply output voltage. When the power supply increases above the overvoltage limit set by R1, the attenuated voltage of the divider network also increases, biasing transistor Q1 on. Q1 in turn biases Q2 on through resistor R10. Q2 supplies gate current to SCR-1, turning it on, causing an effective short circuit across the supply output terminals. This in turn causes the voltage at the power supply terminals to drop, protecting the load from excessive supply output voltage. SCR-1 conducts until the power supply is turned off, or until a fusible link in the circuit clears, causing power to be cut off.

INSTALLATION AND ADJUSTMENT

Install the Overvoltage Protector as Follows:

1. Position the protector on the power supply, figure 1 or 2, so that the unit is located adjacent to the supply output terminal block and the red and black leads are next to the terminal block.
2. Connect and solder red lead to the +V output terminal and the black lead to the -V output terminal of the power supply.
3. Align the two 6-32 captive screws located on the protector, with the two existing pre-tapped 6-32 holes in the LM power supply and firmly tighten screws to secure the protector in position. See figure 3.

Adjust the Overvoltage Protector Voltage-Protection Point as Follows:

1. Turn voltage adjust control on overvoltage unit fully clockwise.

2. The recommended voltage-protection point is 115% of normal power supply operating voltage plus one volt. Compute this value for the operating voltage being used.

3. Turn on the power supply, and raise the output voltage to the desired voltage-protection point; monitor the power supply output voltage to assure correct voltage. If the power supply does not have an adequate adjustment range omit steps 4 and 5, and continue with step 6 below.

4. Slowly turn the voltage adjust control of the Overvoltage Protector counterclockwise until the power supply output voltage, as indicated by the monitoring device, suddenly drops.

5. The voltage-protection point is now set. Remove power supply input power, so that overvoltage protector will reset. Reduce power supply output voltage setting before re-applying input power.

6. If the power supply output voltage adjustment range does not extend to the protection-point voltage computed in step 2, proceed as follows:

   a) Turn on power supply and raise output voltage to the normal operating voltage, taking care to monitor the power supply output voltage.

   b) Slowly turn the voltage adjust control of the Overvoltage Protector counterclockwise until the power supply voltage, as indicated by the monitor, drops suddenly.

   c) Refer to table below, select appropriate volts/turn ratio, and turn voltage adjust control on Overvoltage Protector clockwise by the number of turns equivalent to 1 volt plus 15% of the operating voltage.

MODEL	VOLTS/TURN
LMOV-1	0.3
LMOV-2	0.8
LMOV-3	2.7

For Example:

When using a power supply with an output voltage setting of 5 volts together with Overvoltage Protector LMOV-1, calculate as follows:

$$T = \frac{1 + .15\ (OV)}{V/T}$$

where

T = turns of voltage adjustment control on protector

OV = operating voltage of power supply

V/T = volts/turn ratio from table

$$T = \frac{1 + .15\ (5)}{0.3} = 5.8$$

Rotate voltage adjustment control 5.8 turns in clockwise direction to obtain a voltage setting 1.8 volts above the 5-volt power supply setting, or 6.8 volts.

   d) With voltage adjustment complete, momentarily remove input power to the supply so that the Overvoltage Protector will reset.

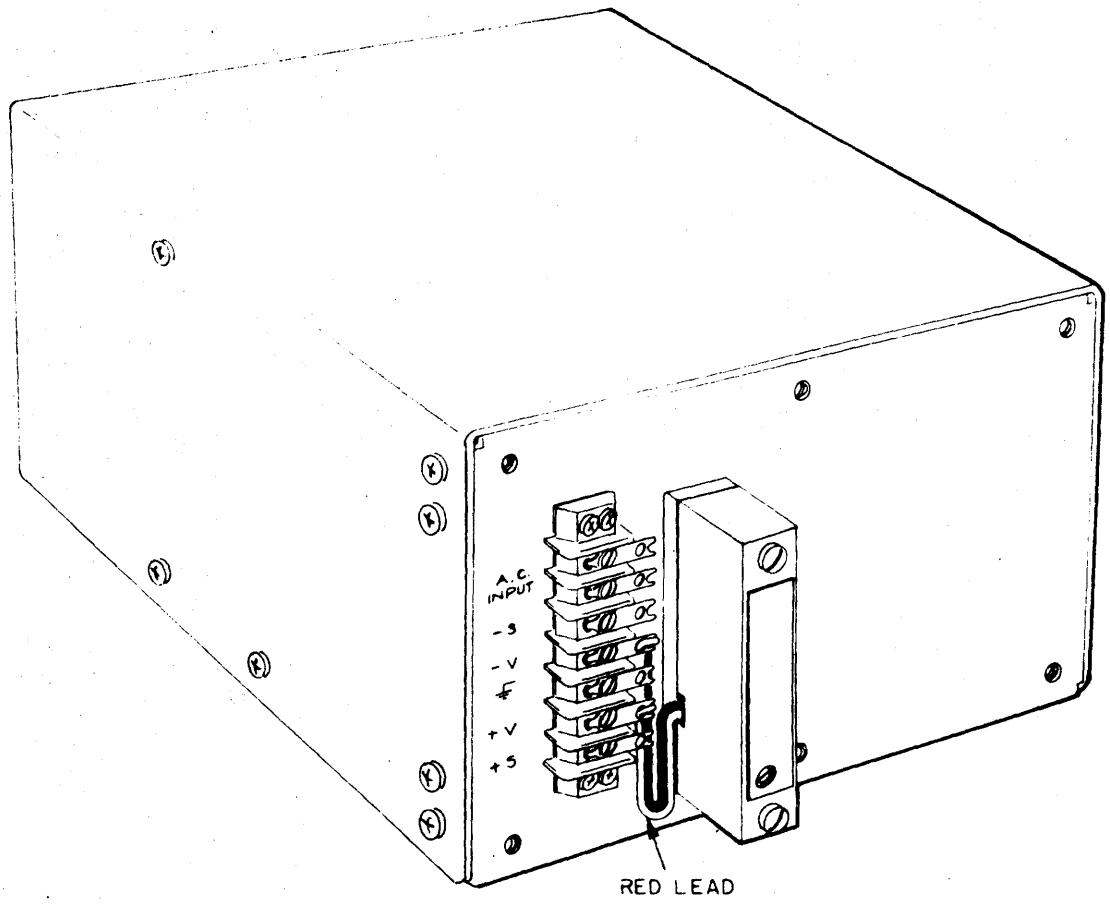
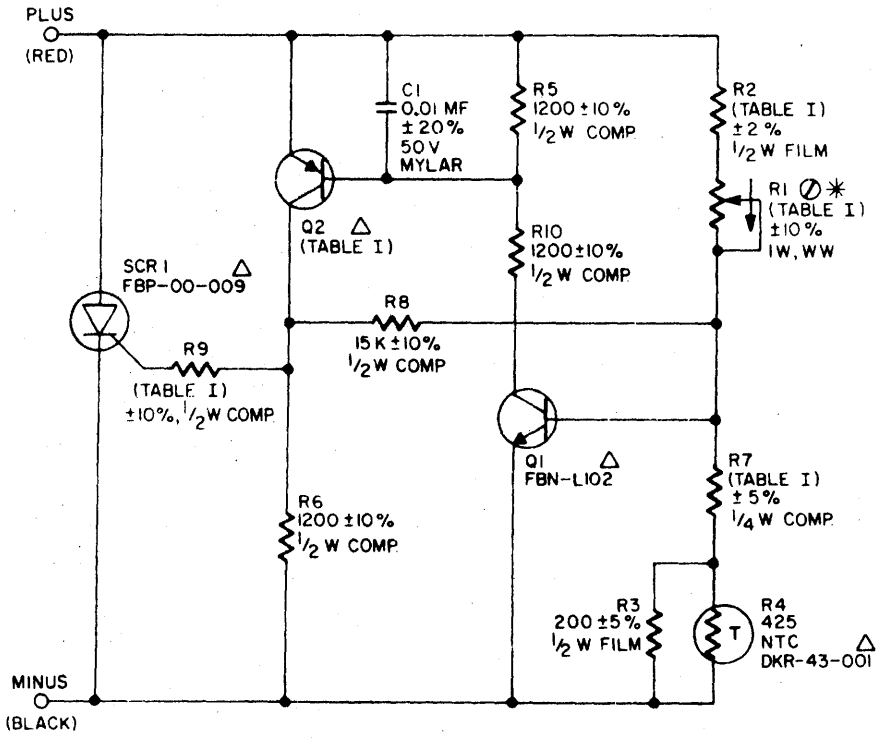


Figure 3. Overvoltage Protector, Typical Installation



THIS SCHEMATIC APPLIES TO UNITS BEARING SERIAL NO. PREFIX A

TABLE I			
	LMOV-1	LMOV-2	LMOV-3
R1	2K	5K	20K
R2	750	1300	4700
R7	33	33	39
R9	JUMPER	JUMPER	22
Q2	FBN-L103	FBN-L103	FBN-L114

NOTES

1. RESISTOR VALUES ARE IN OHMS.
2. SYMBOLS:
   - ↓ INDICATES CLOCKWISE ROTATION OF SHAFT.
   - ⊘ INDICATES ADJUSTMENT OR CALIBRATION CONTROL.
   - ✳ SEE INSTRUCTION MANUAL.
   - △ LAMBDA PART NUMBER.

SCHEMATIC DIAGRAM

OVERVOLTAGE PROTECTORS
LMOV-1, LMOV-2 & LMOV-3

LAMBDA
ELECTRONICS CORP.
MELVILLE NEW YORK

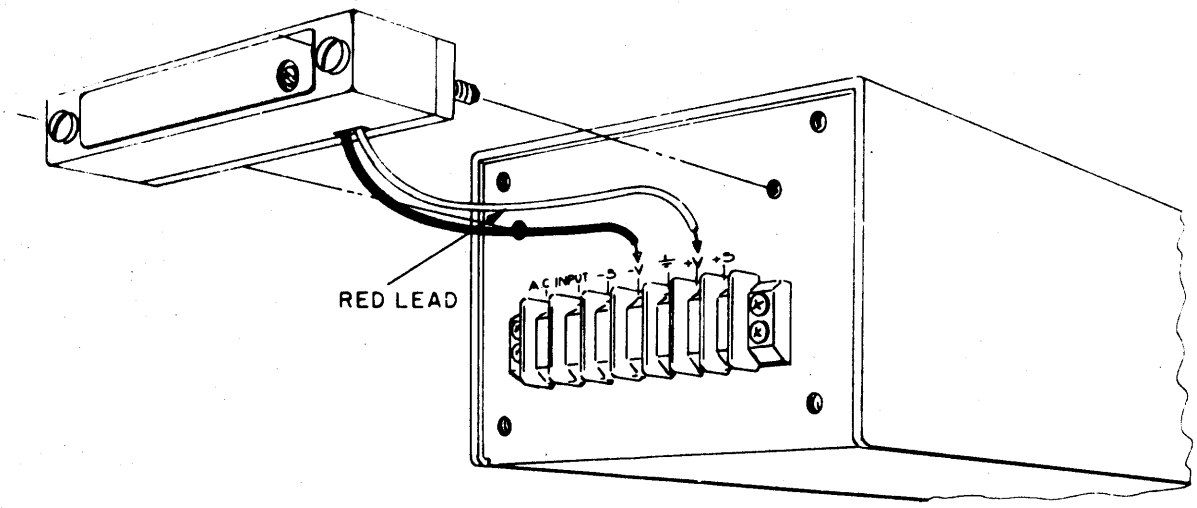


Figure 1. Overvoltage Protector, Typical Horizontal Mounting

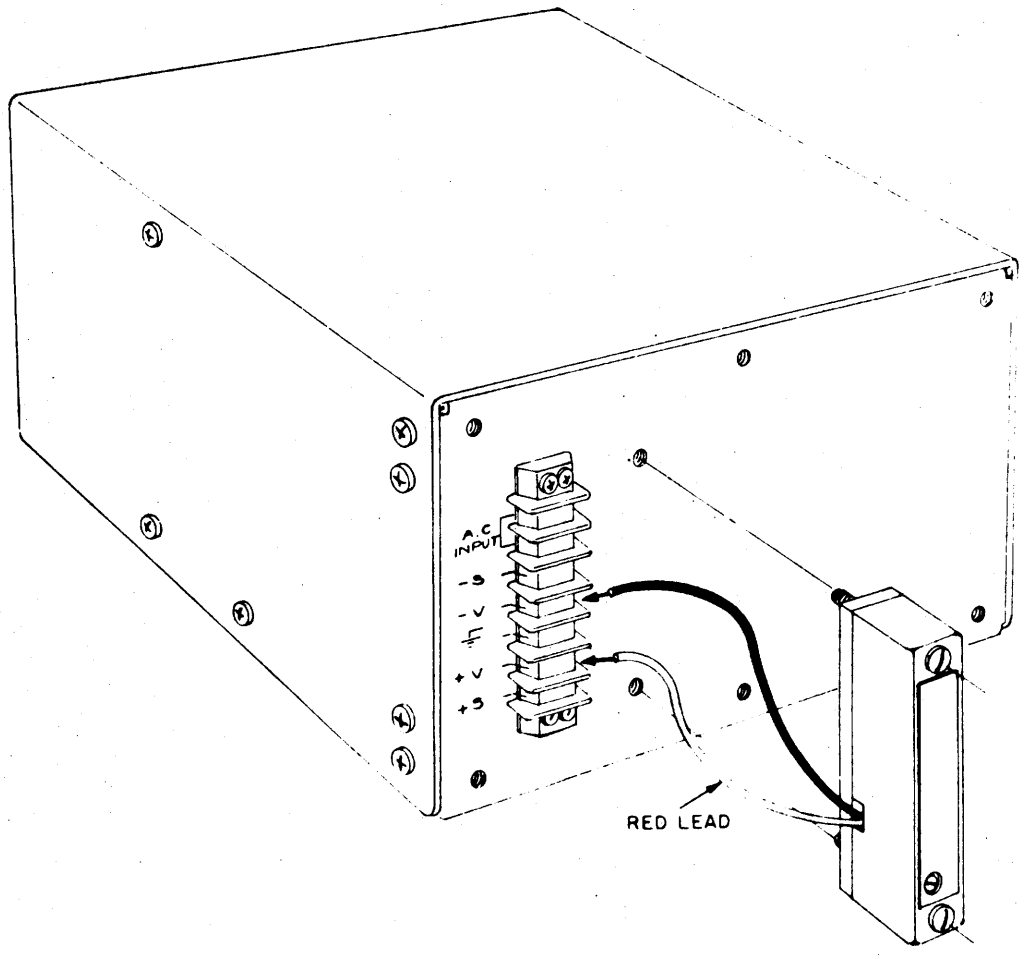


Figure 2. Overvoltage Protector, Typical Vertical Mounting

5-Year Guarantee

We warrant each instrument manufactured by us, and sold by us or our authorized agents, to be free from defects in material and workmanship, and that it will perform within applicable specifications for a period of five years after original shipment. Our obligation under this guarantee is limited to repairing or replacing any instrument or part thereof, (except tubes and fuses) which shall, within five years after delivery to the original purchaser, be returned to us with transportation charges prepaid, prove after our examination to be thus defective.

We reserve the right to discontinue instruments without notice, and to make modifications in design at any time without incurring any obligation to make such modifications to instruments previously sold.

LAMBDA ELECTRONICS CORP.
515 BROAD HOLLOW ROAD • MELVILLE, L. I., NEW YORK • 516 MYRTLE 4-4200

DIGITAL OUTPUT POWER SUPPLY, 125 VOLTS

The 125 volt power supply is identified under GE model 4491A. It is also identified under GE part number 68A8456P. The letter "P" in the part number is followed by three digits. The first digit "1" identifies it as a 10 ampere supply. The second digit identifies the input AC line frequency for which it is designed, with a "1" indicating 60 cycles and a "2" indicating 50 cycles. Most systems use a "1" as the third and last digit of the three digit number following the letter "P" in the part number. A "2" in this position indicates special features used with Micro-power units.

The schematic of the power supply is GE drawing 68C998191.

The schematic is self explanatory. Note that it incorporates ferro-resonant voltage regulation. Optional wiring exists for 115 or 230 input voltage. A relay, K1, is incorporated across the 115 volt input portion of the input transformer. The relay has a normally closed contact in series with a 50 ohm resistor across the output of the power supply which serves to discharge the capacitor bank when power is turned off. Application of AC power opens the relay contact during usage of the supply.

This power supply is usually used for "reading" customer contacts via the Digital Input feature used by some systems. Digital Inputs are discussed in Volume II. A relatively high voltage such as 125 volts insures better reading of contacts in operations where there is an inherent tendency for dust to collect on the contacts, example, cement plants. The 125 volts is reduced to logic levels through current limiting circuitry.

DIGITAL INPUT POWER PACKAGE 4847A-T

TABLE OF CONTENTS

DIGITAL INPUT POWER PACKAGE

Digital Input Termination Cabinets or Digital Input/ Output Termination Cabinets, model no. 4847A, employ a 4847AS04 Digital Input Power Supply Package. This package consists of power supply VD1, a 28V relay power supply; and VD2, a -12V supply which provides logic voltage to the selection gates on the digital input termination boards.

The external connections to these power supplies are shown on the Digital Input Logic drawing, 70C180056, which is located under the "Process Digital I/O" tab in the system drawings book set. The 28V supply is a model 4290A120 Power Supply, and the schematic is on drawing no. 68A998331. The -12V power supply is a Lambda Electronics Corp. model LM-E12 or a North Electric N15012.

The theory of operation for the 28V supply is described in publication no. 4290A which is in this section. Preventive maintenance, adjustments, and troubleshooting information for both supplies is in the Power Supplies section of the Computer Maintenance manual. The remainder of this publication describes the theory of operation of the -12V supply.

-12V LM-E12 POWER SUPPLY

The -12V supply features line regulation better than ±0.1%, load regulation better than ±0.1%, and noise and ripple output of 3 millivolts peak-to-peak or less. A schematic of the LM-E12 appears in Fig. DIP. 1.

Step-down transformer T1 provides power to two rectifiers; a half-wave bias supply rectifier, CR7, and the full-wave rectifier, CR8 and CR11.

The unregulated DC, at about 23V, is dropped to a regulated 12V by five series regulator transistors, Q8 through Q12. Driver stages, Q5 and Q6, are emitter followers which couple the control signal from the amplifiers to the series regulators. Q1 and Q2 are the voltage regulation amplifiers and Q3 is an output current limiting amplifier, R1 is the output voltage adjustment and R17 is the current limit adjustment.

Zener diodes, CR1 and CR6, are shunt regulators, which regulate the DC output from the bias rectifier, CR7, to approximately +8.1V and +9.1V, respectively, with respect to the +V output terminal (common). These voltages supply the amplifiers.

The voltage regulator (error) amplifier monitors the output voltage which is across the voltage divider in the Q1 base circuit. The amplifier corrects the operating point and series impedance of the series regulator transistors to maintain the output voltage within tolerance.

The current limit amplifier, Q3, is normally not conducting. Should the load current increase to the limit, (approximately 15.4 amps. at 40°C) the base of Q3 will be raised to the point where the transistor begins to conduct, lowering its collector voltage, and increasing the impedance of the series regulators. If the load impedance continues to decrease, Q3 continues to decrease both the output voltage and the current. If the load current is again reduced below the limit, Q3 will cease conducting and the voltage regulator will again take over.

The current limit is adjusted at the factory and sealed by soldering the wiper of R17 in the final position. This adjustment is not normally made in the field.

An 8 amp. fuse in the AC input line provides protection from internal power supply failures. A thermostat in the AC line removes power from the transformer if the maximum temperature of the series transistor heatsink is exceeded. The thermostat contacts close again when the temperature returns normal.

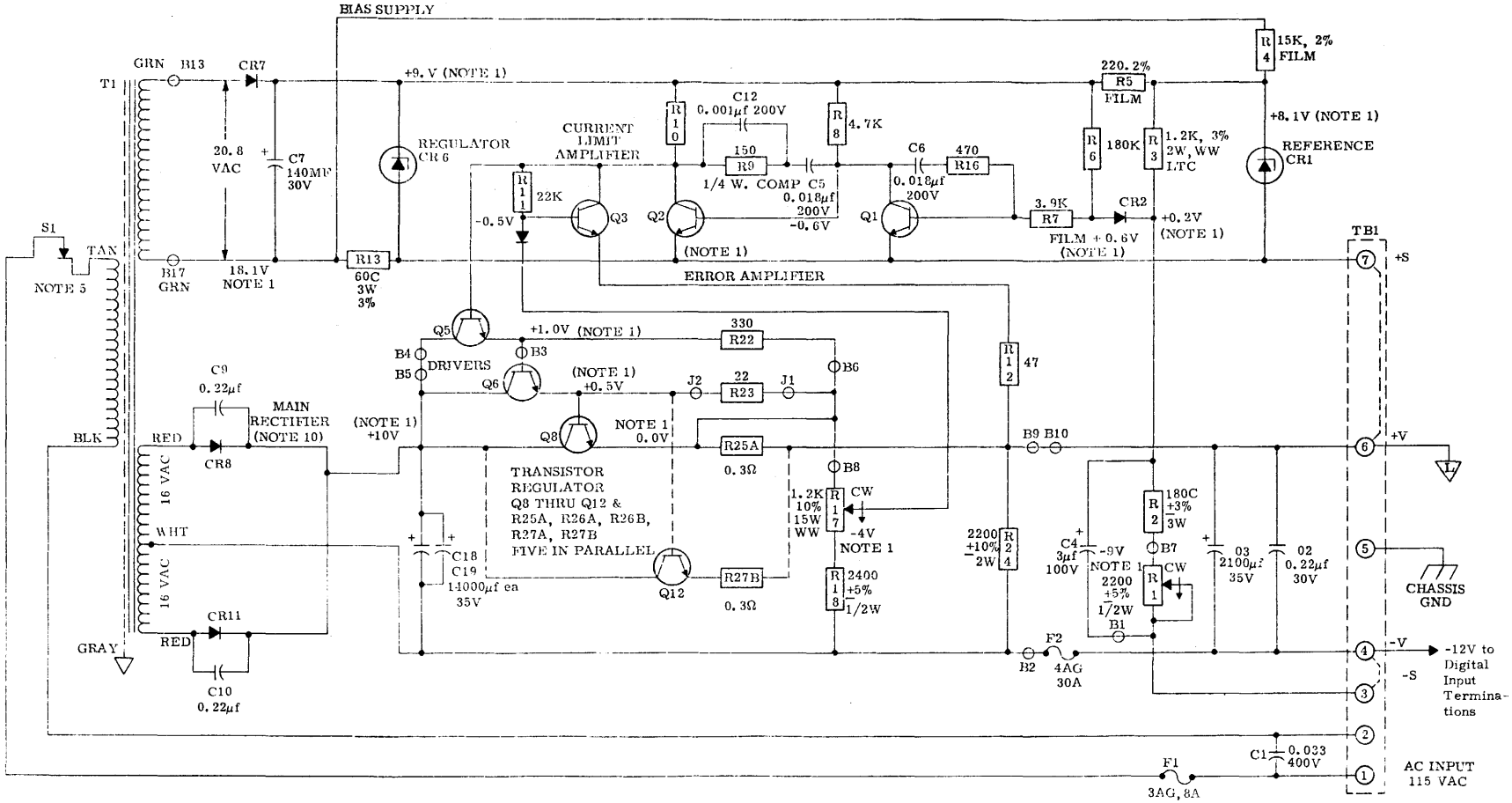
Overvoltage Crowbar

A small crowbar assembly is mounted near the output terminal board on the -12V supply. The crowbar protects the load circuits from excessive power supply output voltage. Fig. DIP. 2 is a schematic of the crowbar circuit.

During the normal operation, both transistors are off. If the power supply output voltage exceeds the limit, sufficient current is drawn through the base and emitter of Q1 to turn it on. This lowers the voltage on the collector of Q1 sufficiently to turn Q2 on, raising the gate of the SCR to the firing point. The conducting SCR places a virtual short across the supply output, the current limiter in the supply takes over, the output voltage decreases, but the current through the SCR remains above the holding current for the device.

Once a crowbar is fired, operation cannot be restored without turning the supply off, because the SCR cannot be turned off until the current through it drops below the holding level. If power is turned off, the fault cleared, and power restored, the crowbar circuit will revert to the normal state.

The crowbar trip point is adjusted to 14.8V by a trimmer resistor in the base circuit of Q1. R4 is a resistor with a negative temperature coefficient of resistance, which corrects the bias on Q1 to compensate for changes in the current gain of the transistor with temperature. The adjustment procedure is in the Power Supplies Computer Maintenance manual.



NOTES:

1. DC measurements taken with 20,000 OHMS/V voltmeter between +S (Term. 7) & indicated points unless noted.
2. Coat both sides of insulating wafer with dow corning no. 340 silicone grease.
3. If Q5 is replaced, always retain radiator and install with new transistor.
4. Θ indicates terminal on PWB (B) or terminal board (J).
5. S1 is automatic resetting, opens at 8.6 amps DC load @ 71°C.
6. R1 is voltage adjustment. See Maintenance Manual Section 2. R17 is current limit adjustment.

Fig. DIP. 1 Lambda -12A Digital Input Power Supply

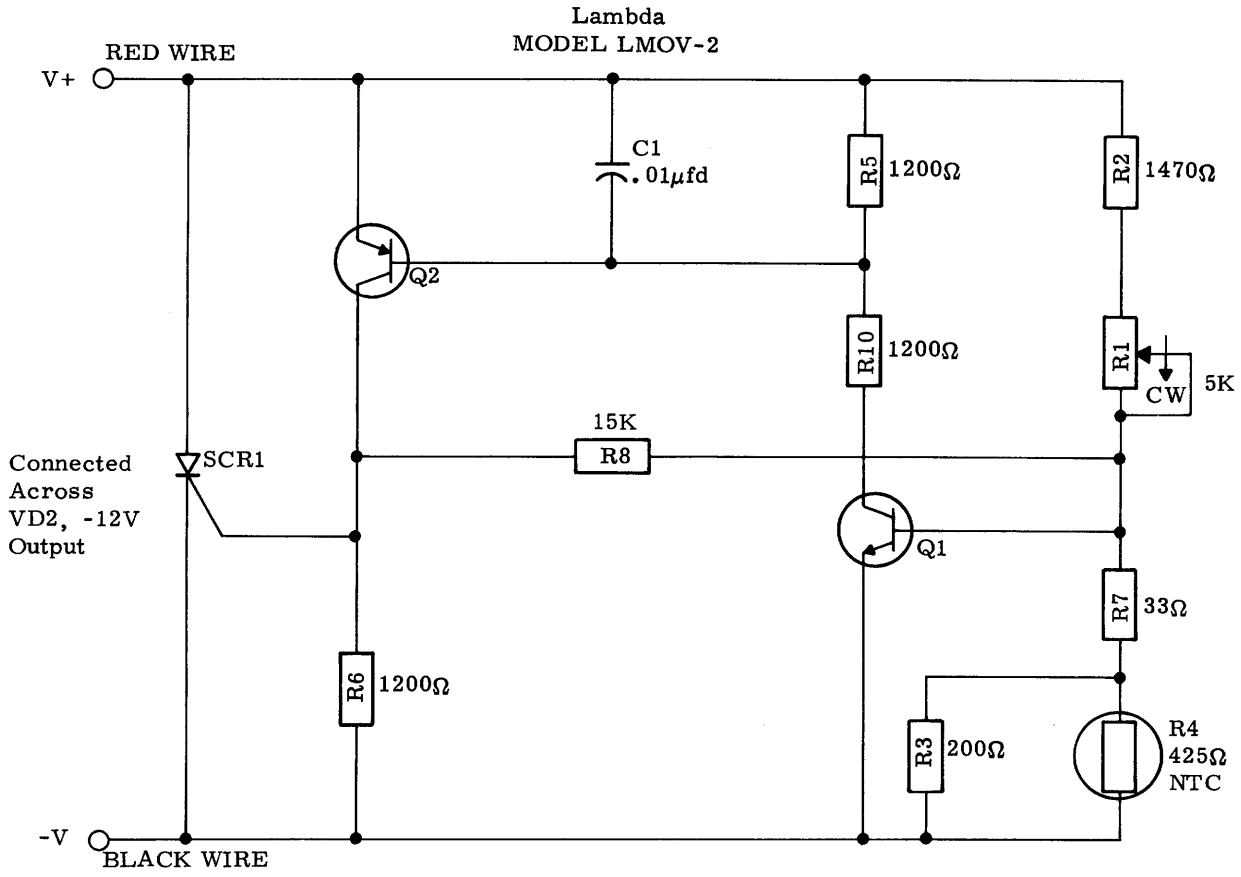


Fig. DIP. 2 -12V Crowbar

-12V N15012 POWER SUPPLY

Single phase input power is applied to the primary of a step-down transformer T1 through fuse F1, which open-circuits the primary when blown, and through thermal switch S1, which open-circuits the primary when the heat sink temperature reaches approximately 125°C. T1 has two secondaries which are full-wave rectified and capacitive filtered to produce the voltages indicated on the schematic, Fig. DIP. 3.

After rectification and filtering the bulk voltage is applied to the collectors of the Darlington Q3, Q4-Q10. This bulk voltage is then regulated, as described below, to provide a precision output which can be adjusted by using R16 potentiometer via rear of unit.

The bias voltage is regulated by R5 and VR1 to provide a constant voltage to driver transistor Q1 and integrated circuit VR3 (which contains a stable reference, detector, amplifier and current limiting circuit).

The output voltage is sensed by the resistive divider R1, R2, and compared to the stable reference of VR3. If the output voltage increases by DV, an error voltage is applied to Q1 which decreases its collector current and increases the impedance of Q4-Q10 enough to drop DV across its collector-emitter. Conversely if the output voltage drops by DV, the impedance of Q4-Q10 is decreased enough to allow an additional DV to appear across the load. The stabilization network for the loop is made up of C2, C4, R7 and the output capacitor C8. R17 and R14 are used in the higher voltage units as bleeder resistors to eliminate a possible shock hazard.

The current fold-back circuitry is made up of an emitter resistor for each parallel pass transistor, R13, potentiometer R12 and the internal circuit of VR3. Changes in output current are sensed by the emitter resistors. When the output current reaches approximately 130% of the 40°C current rating, the output voltage and current begin to decrease linearly such that a safe power level is maintained for the power-supply.

> NOTE
>
> R12 potentiometer is factory set and determines the maximum current output. For adjustment of R12, refer to Power Supply Testing.

Troubleshooting Guide

When trouble occurs thoroughly inspect the following before going to internal components.

A. See that the proper A.C. input voltage is applied to A.C. input terminals TB-1 & TB1-2.

B. See that F1 and/or F2 fuse is not blown (if blown, replace with fuse of same value).

C. Check for an over-heating condition, which will open-circuit S1 thermo switch.

D. See that power-supply load is properly connected.

E. Check power-supply sensing mode (local or remote which ever is applicable).

F. See that an overload condition does not exist on power-supply output terminals.

If conditions A thru F have been checked satisfactorily, then proceed to the troubleshooting list:

> NOTE
>
> Schematic diagram Fig. DIP. 3 indicates voltage reading used in troubleshooting of power-supply.

Indication: Low or No Output Voltage.

Trouble 1. Supply operating under a current limit condition, check for over-load, short circuit or improper connection of load leads.

2. Check R16 voltage adjust potentiometer for correct voltage setting.

> NOTE
>
> Voltage control potentiometer R16 adjustment is at rear of unit via access hole in cover.

3. Check for correct voltage on VR1 zener diode. Reference schematic for voltage level.

Indication: High Output Voltage.

Trouble 1. Check R16 voltage, adjust potentiometer for correct output voltage setting.

2. Check R3, R4, R16 and (-) sense lead for "open".

3. Check VR3 and Q1 (see schematic) for shorts.

4. If all items check O.K. and still have high output voltage, remove R8 resistor if output remains high either Q3 or one or all of the pass transistors are shorted.

   If removing R8, the output voltage goes low, the defective component is probably VR3 or Q1.

Indication: High Output Ripple.

Trouble 1. Check for over-load condition on power-supply.

2. Check current limit pot R12 for correct setting.

3. Check for loose connections at sense and/or load terminal.

4. Check for proper input to terminals TB1-1 and TB1-2 (Reference schematic).

5. Check VR1 zener diode for open.

Indication: Poor or Incorrect Regulation.

Trouble 1. Meter measurements must be taken as follows:

Local sense - connect meters at TB1-7 and TB1-3 for readings.

Remote Sense - connect meters at the Sense Lead termination point.

2. Check R12 current limit pot for correct setting.

3. Check for loose connections at load and/or sense terminals.

Procedure For Setting Current Limit Control R12

Whenever any components are replaced, check R12 current limit potentiometer setting.

> NOTE
>
> The adjustment procedure requires that the power-supply be removed from associated equipment.

A. Remove A.C. input power to the power-supply.

B. Break the seal of R12 from potentiometer housing and turn to full CCW direction being careful not to overload or short the supply while R12 is in the full CCW direction.

C. Operate the power-supply for constant voltage with local sensing, with no external load.

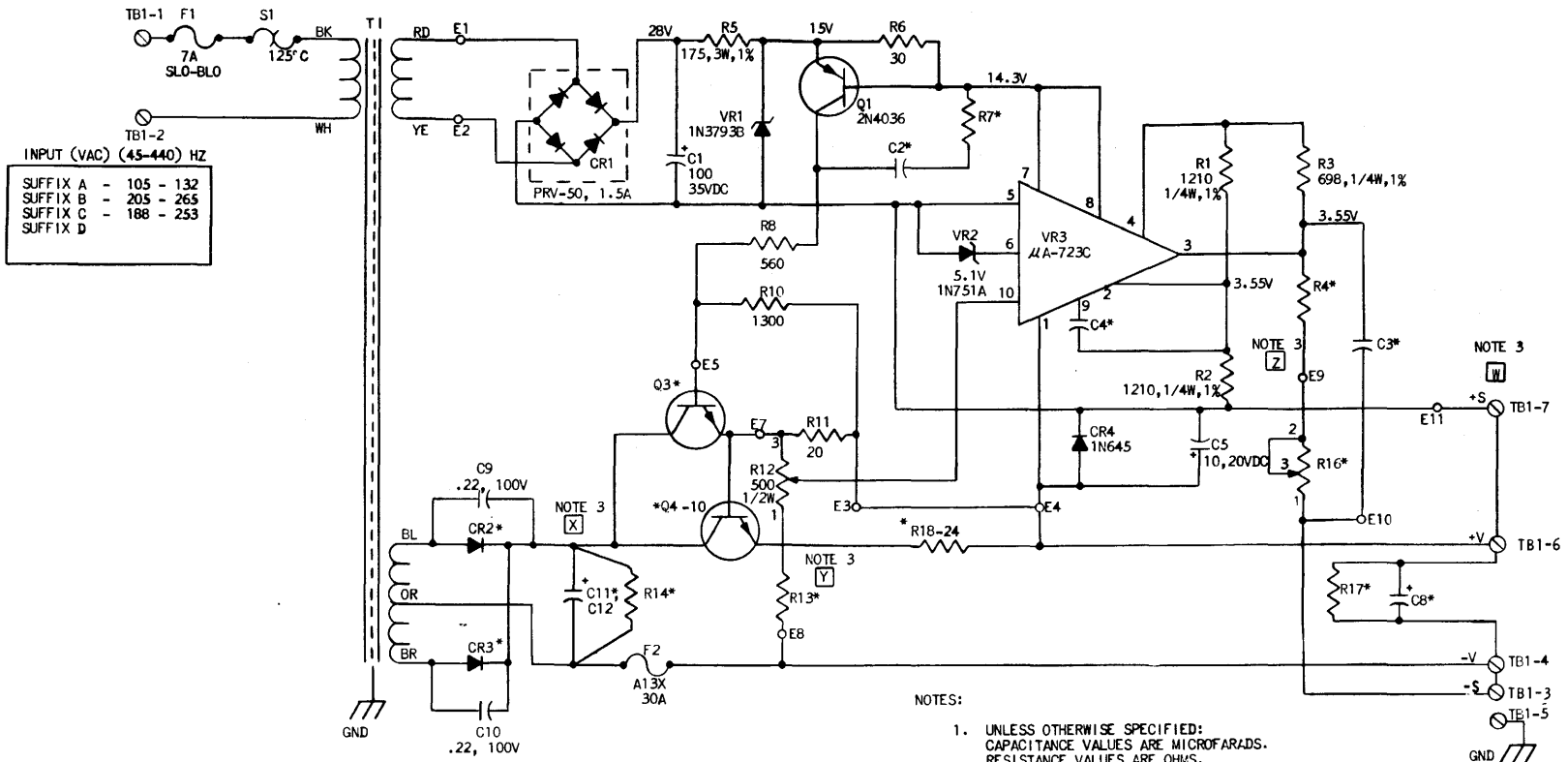
D. Turn voltage adjust control R16 CCW until minimum rated output voltage is obtained. (Reference Schematic for voltage range).

E. Apply load so that output current is 110% of 40 C rating for unit (see schematic for rated load).

F. Using an oscilloscope on unit output, observe output ripple while adjusting R12 in a CW direction. Adjust R12 until the output ripple increases to approximately 5 MV on oscilloscope. At this point the unit will start into a current limit condition.

G. Reduce the load resistance; the output current should decrease as the load resistance is reduced toward short circuit. At short circuit the output voltage will be approximately "0" volts.

H. Check setting and repeat adjustment procedure if required, and reseal current limit pot R12.



MODEL NO.	VOLTAGE RANGE OUTPUT D.C.	MAX. CURRENT (AMPS) 40°C	50°C	60°C	71°C	SCHEMATIC VOLTAGE W	X	Y	Z
N15012	12.0 VDC ± 5%	15.0	13.6	12.3	9.5	+12.0	+26.45	+10.65	+1.56

NOTES:

1. UNLESS OTHERWISE SPECIFIED:
   CAPACITANCE VALUES ARE MICROFARADS.
   RESISTANCE VALUES ARE OHMS.
   RESISTORS ARE 1/2 W., ±5%.
2. VOLTAGE MEASUREMENTS ARE TYPICAL AND TAKEN AT NO LOAD WITH NOMINAL INPUT AND A 20,000 OHMS/VDC VOLTMETER BETWEEN THE INDICATED POINT AND THE (-S) TERMINAL FOR CHARTED VOLTAGES AND (S) FOR ALL OTHER VOLTAGES.
3. SEE CHART FOR VOLTAGE MEASUREMENT.
4. WE RESERVE THE RIGHT TO CHANGE CIRCUITRY, OR SOURCES WITHOUT NOTICE.
5. * SEE CHART ON SHEET (2) OF SCHEMATIC FOR DESCRIPTION.

MODEL NO.	A1 CARD ASS'Y.	C2		C3		C4		C8		C11,12		R4			R7			R13			R14			R16			R17			R18 - 24			Q3-10	CR2 & CR3	UNIT REV.
	PART NO.	MFD	VDC	MFD	VDC	MFD	VDC	MFD	VDC	MFD	VDC	OHMS	W	%	OHMS	W	%	OHMS	W	%	OHMS	W	%	OHMS	W	%	OHMS	W	%	OHMS	W	%			
N15012 12.0 VDC	618 2033	.068	100	.1	100	.033	100	3100	25	25,000	35	2870	1/4	1	910	1/2	5	2700	1/2	5	NOT USED			500	2	10	100	10	5	.3	3	1	2N3055	1N1185A	–

Fig. DIP. 3 Schematic (N15012)

28 VOLT POWER SUPPLIES

The model 4290A 28 Volt Power Supplies* convert the primary AC power to regulated 28V DC. Two versions of this module are available. They and their corresponding GE schematic drawings are:

- 4290AX1Z 28V Power Supply, 5 amperes, Dwg. # 68C998942.
- 4290AX2Z 28V Power Supply, 10 amperes, Dwg. # 68C998331.

After having been installed, tested, and adjusted as indicated in the Computer Maintenance Manual, in Section 2. Power Supplies; the overall performance of the module should be such that the output voltage, including noise and ripple, does not vary from the original value set, by more than plus or minus three percent. This specification should be met under all conditions of line, load, and environment specified for the system. If it is not met, servicing of the 28V Power Supply is indicated.

The remaining undefined digits in the complete model number specify additional options as follows:

- X = 1; 60 Hz primary power.
- X = 2; 50 Hz primary power.
- Z = 0; no overvoltage crowbar.
- Z = 1; overvoltage crowbar provided.

While these power supplies may be used for any application for which their 28V output is suitable, they are primarily used to supply those circuits within the computer system which have widely variable loads such as relays, relay drivers, lamps, etc.

These supplies are usually designated by markings adjacent to the module and on the system drawings according to their application as follows:

- VP; peripheral supply. Supplies 28V to peripherals through the Peripheral or I/O Buffer, or directly to the peripherals. May also be used for the multiple output controller.
- VD; digital supply. Supplies 28V to the digital input terminations.
- VG; analog supply. Supplies 28V to the analog converter and controller.
- VX; miscellaneous supply. Supplies 28V to special loads such as process controllers, special consoles, etc.

* In 4010 computer systems these supplies are included in power packages 4847AS04 and 4847AS07 and are installed in digital I/O termination cabinets.

Some variation in these typical applications may occur. Reference to the DC power distribution sheets of the system power distribution drawing will provide information as to the actual loads for the 28V supplies in a given system. The system block diagram indicates the location of each 4290 module within the system cabinets.

In some systems, 2 model 4290's may be connected in series to provide 56V output.

TRANSFORMER AND RECTIFIER

Either 230V AC or 115V AC may be applied to the power supply transformer. Usually, transformer terminals 2 and 3 are jumpered, and 230V AC is applied. A chart on the schematic indicates the method of connection for 115V AC. A ferroresonant winding is provided on the transformer, which is adjusted to provide optimum line regulation at the input frequency. The input frequency option determines the value of the capacitor connected across the ferroresonant winding, as indicated on the schematic.

Each of the two center tapped transformer secondaries drives a conventional full wave rectifier. The 120 Hz (or 100 Hz) ripple at the rectifier outputs is reduced to a level tolerable by the regulator circuit by large value capacitors.

Ten volts DC is produced at the output of the CR1 - CR2 rectifier. The return side of this circuit is connected to the +28V regulated output, with the result that the positive output of the ten volt supply provides a bias supply to the voltage amplifier in the regulator circuit at +38V DC, with respect to the negative output terminal of the module.

The CR3 - CR4 rectifier provides a nominal 39V DC output which is controlled and smoothed by the regulator to provide the specified 28V DC output.

REGULATOR

Regulation of the 28V output is accomplished by varying the impedance of five parallel transistor stages in the case of the 5 amp. supply, and ten parallel stages in the case of the 10 amp. supply. A fixed emitter bias voltage is applied to the voltage amplifier (Q7 in 5 a., Q12 in 10 a.). The input to the voltage amplifier is applied to the base, from the output adjustment potentiometer. The output of the voltage amplifier, buffered by an emitter follower (Q6 in 5 a., Q11 in 10 a.), is applied to the parallel stages, where the signal regulates the output.

To illustrate the operation of the regulator, assume that, due to a drop in line voltage or an increase in load current, the output tends to decrease. Due to the resultant decrease in the base bias voltage, the conduction of the voltage amplifier transistor decreases. The voltage at the collector of the amplifier

therefore increases, as well as the voltage at the emitter of the emitter follower. The base-emitter current of each of the parallel transistor stages increases, decreasing their impedance, and holding the output voltage nearly constant.

Note that each of the parallel transistor stages has a 0.2 ohm resistor in series with the output from its emitter. These resistors tend to equalize the current flowing through each of the parallel stages, to ensure that individual transistors with higher current gain, do not carry most of the load.

Since no point in the rectifier of regulator circuits is connected to ground, except by external wiring, either the positive or the negative output terminal may be grounded. Usually, the negative terminal is grounded providing +28V DC at the output.

OVERVOLTAGE CROWBAR

The optional overvoltage crowbar circuit is installed on a small chassis, external to the 4290 power supply module. The 28V power supply output is brought from the module, through the crowbar chassis, and on to the loads.

Zener diode CR1, and silicon controlled rectifier SC1 are normally not conducting, and the 28V power passes through a circuit breaker, and on to the loads.

If, due to some malfunction the voltage increases to approximately 33V or more, CR1 conducts, applying positive voltage to the gate on SC1. SC1 then turns "on" and will remain "on" until the voltage has decayed sufficiently, that the current is less than the holding current for the SCR. Since, when conducting, the SCR appears as a virtual short, the circuit breaker will trip.

If the circuit breaker is reset after the voltage returns to normal, operation can resume.

28V SEQUENCER OPTION

Where the Model No. 4290A option code Z is equal to 2, a 28V Sequencer assembly is provided in a location near the 28V Power Supply, which causes a very abrupt termination of the 28V output when the 4797 Sequencer shuts down sequenced AC power. The 28V output is dropped to zero in a few microseconds to prevent erroneous operation of peripheral devices or process inputs or outputs utilizing the 28V supply, due to the decaying logic voltages.

The 28V Sequencer schematic is provided on sheet 1.3 of the power supply schematic drawing (68C998942 for 5a, or 68C998331 for 10a). The 28V DC output from the 28V Power Supply is routed through the 28V Sequencer to the load circuitry. When 28V power is on, a red light, DS1, on the 28V Sequencer panel is lit.

Q1 and SC2 are the primary components of a crowbar circuit which will trip circuit breaker CB1 if the 28V output goes above about 35V. Should this occur, the Zener diode string on the base of Q1 will start to conduct, lowering the base voltage sufficiently to turn Q1 on. When Q1 conducts, the gate of SC2 goes positive, turning the SCR on, shorting the output, and tripping the circuit breaker.

Q2, T1, and SC1 are the principal components in the circuit which abruptly terminates the 28V output in the case of 4797 Sequencer shutdown. A cable interconnects the Sequencer and the 28V Sequencer as shown in Fig. +28V. 1. The shutdown signal is a positive pulse of about 18V peak and several milliseconds duration. The signal line rests at about -10V in the absence of the pulse.

The positive pulse turns on CR2 and 3.9V Zener diode CR7. The positive pulse is developed on one of the primaries of T1, coupled to the secondary without inversion, and applied to the gate of SC1, turning it on. SC1 shorts out the 28V output and begins to discharge the filter capacitors in the 28V Power Supply within a few microseconds.

The duration of the gate pulse to SC1 is limited by Q2. Within a few microseconds after the leading edge of the shutdown pulse, C1 will have charged sufficiently to allow the base of Q1 to swing positive, cutting it off, producing a negative swing across T1, and terminating the gate pulse.

SC1 conducts as long as the power supply capacitors provide sufficient holding current for the SCR. During the conduction period, the sequenced AC power contactor relays drop, removing the primary supply.

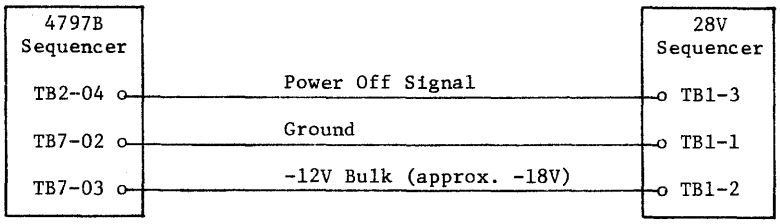


Fig. 28V. 1. Sequencer to 28V Sequencer Connections

READER COMMENTS

The General Electric Company solicits your comments on publications covering Process Computer equipment. Please explain any "no" responses in the COMMENTS section. Your comments and suggestions become the property of the General Electric Company.

- Name of Manual: _____
- What is your computer application: _____

  _____
- How is this publication used:

  Familiarization ☐ Reference ☐

  Training ☐ Maintenance ☐

  Other (Explain) _____

	YES	NO
• Does this publication meet your requirements	☐	☐
• Is the material:		
1) Presented in clear text	☐	☐
2) Conveniently organized	☐	☐
3) Adequately detailed	☐	☐
4) Adequately illustrated	☐	☐
5) Presented at appropriate technical level	☐	☐

- Please provide specific text references (page number, line, etc.) with your comments.

NAME _____ DATE _____

TITLE _____

COMPANY NAME _____

AND ADDRESS _____

COMMENTS:

No postage necessary if mailed in the U. S. A.

Staple

Communications concerning Technical Publications should be directed to:

Manager, Technical Publications
Utility and Process Automation Systems Operation
2255 West Desert Cove Road
Phoenix, Arizona 85029

Fold Fold

FIRST CLASS
Permit No. 4091
Phoenix, Arizona

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

GENERAL ELECTRIC COMPANY
UTILITY and PROCESS AUTOMATION
SYSTEMS OPERATION
2255 West Desert Cove Road
Phoenix, Arizona 85029

Attention: Technical Publications

Fold Fold

Cut Along Line

Additional Comments:

4016B/5174 CORE MEMORY

TABLE OF CONTENTS

INTRODUCTION

The 4016B/5174 (70D195174) Core Memory Unit is that part of the GE-PAC* 4010 Central Systems Unit used to store the program instructions and data. The memory is a random access device operating at a 1.6 micro-second memory cycle time. One memory cycle is the time from any one event in one cycle to the same event in the following cycle. Memory sizes of 16,384, 24,576, or 32,768 words are available.

Each memory word consists of 25 bits of which 24 are data and one is used as an odd parity bit. The parity bit is generated prior to storing (writing) a word into memory and then checked when the word is read from memory. The parity bit is used to flag the running program when a word is read from memory that does not contain an odd number of "one" bits. If an odd number of "one" bits is not contained by the word read from memory, it indicates an error has occurred in either parity check or generate circuits, memory operation, or data transfers. If a parity error is detected, the running program may monitor the error (JNP), visual alarms are provided, and further accesses to memory may be inhibited using the STOP ON PARITY switch.

The arithmetic unit Bulk Memory Controller, and two optional devices may communicate directly (one at a time) with the core memory module. Access to memory is on a priority basis with the Bulk Memory Controller having top priority user number 2 next, then user number 3, and then the arithmetic unit lowest priority.

Fig. INT. 1 illustrates the location of the core memory within the 4010 Central System Unit.

SPECIFICATION SUMMARY

Features of the Core Memory module are summarized below:

- Type - 2 1/2 D, Coincident Current, Destructive Read.
- Capacity - 16,384 (16K), 24,576 (24K), or 32,768 (32K) words.
- Word Size - 24 information bits plus 1 parity bit.
- Memory Cycle Time - 1.6 microseconds.
- Access Time - 800 nanoseconds.
- Communication - Parallel.
- Random Access - All memory cells are directly addressable and all require the same time for reading and writing.
- Protection - Word Select currents are disabled during AC power failures and power turn-on so as to protect core cells from arbitrary data destruction.
- Temperature Stability - Temperature controlled core stack allows operation of core at 55°C. This is done to maintain operation throughout ambient temperature of 0° to 55°C.
- Operating Mode - Asynchronous. The Memory enters a dynamic mode only after it has received a request.
- Error Checking - Core Parity error, Transmission Parity error for channel 1, 2, and 3 users, and core stack temperature outside the operating temperature range.

  The Core Parity error signifies that an even number of ones were read from the addressed core memory location even though an odd number of ones were stored in the addressed location. When an error is detected, the running program may monitor the error (JNP), visual alarms are provided on the programming and maintenance console and further assesses by the AU may be inhibited by using the Stop On Parity switch. The data is restored in the same configuration as read.
- A transmission parity error signal signifies that the parity bit associated with the data sent to the memory from user devices other than the AU and the parity bit generated by the memory for the data received at the memory did not compare. The memory does not hold this information; instead a signal is applied to the user indicating the error occurred.

  Memory operation is halted and no memory request is honored when a parity error occurs while the core memory module is outside its normal temperature range (45°C to 65°C). Requests are honored when the core memory module is outside its normal operating temperature range as long as a parity error is not detected.
- Instructions may be executed with an address greater than implemented memory size. True memory wrap around occurs in 16K and 32K systems. In systems with 24K of core memory "zeros" are read back with no parity errors when addresses of > 24K are executed.

MODEL NUMBERS

The following model numbers are associated with the core memory module:

4DP4800BS01 - Basic Central Systems Unit (Specifies a 16K word Core Memory system)

*Registered Trademark of General Electric Company

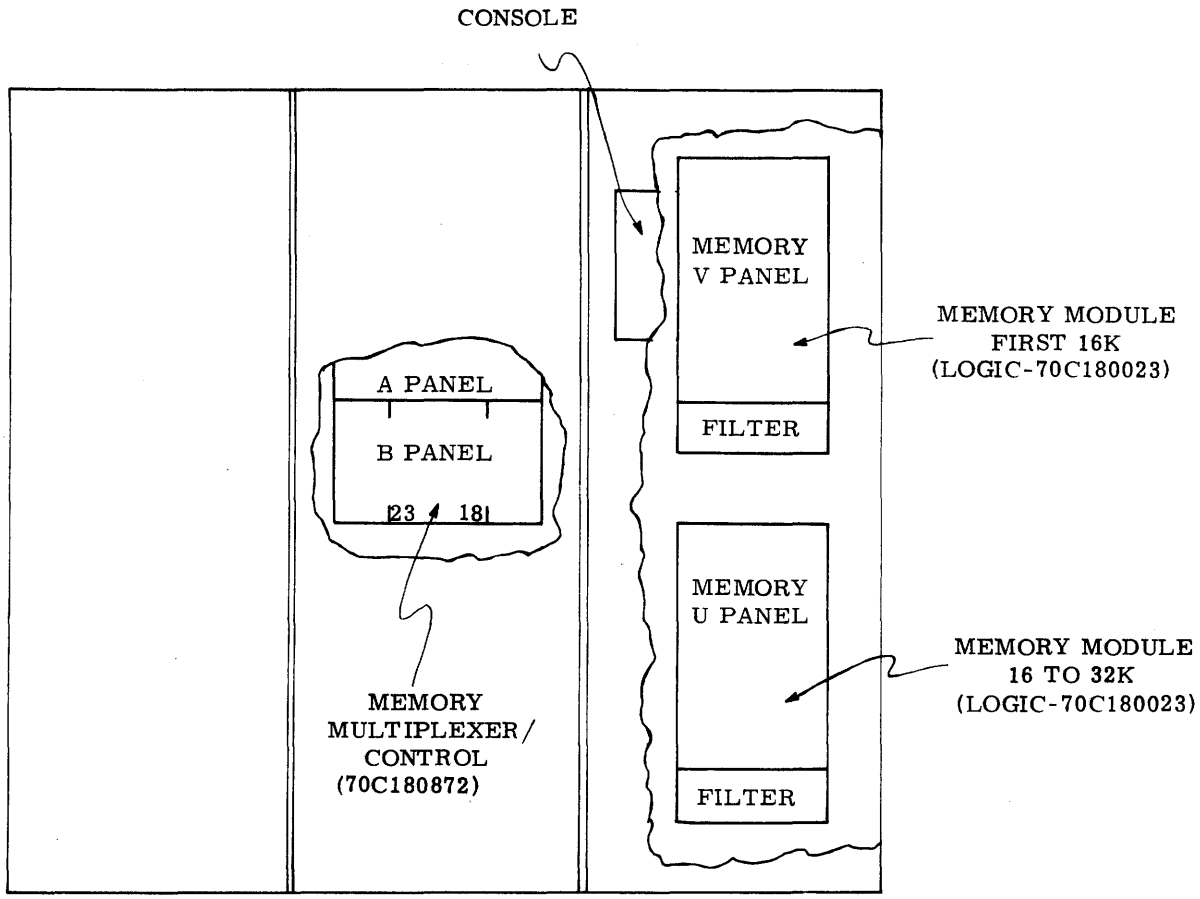


Fig. INT. 1 Core Memory Location

4DP4800BS02 - 16 - 24K Core Adder (Expands a 16K Core Memory system to a 24K Core Memory system)

4DP4800BS03 - 24 - 32K Core Adder (Expands a 24K Core Memory system to a 32K Core Memory system)

4DP4800BS04 - 16 - 32K Core Adder (Expands a 16K Core Memory system to a 32K Core Memory system)

The Central Systems Unit drawing, 4DP4903BSID, defines the options contained in a particular system.

REFERENCE DOCUMENTS

Logic:

70C180872 - 4016B Memory Multiplexer/ Control

70C180023 - 5174 Memory Module

Interface Module Logic:

70C180955 - 4022D Arithmetic Unit
70C180908 - Bulk Memory Controller
70C180914 - Power Components

Maintenance:

GE-PAC 4010 Computer Maintenance Manual

MEMORY BLOCK DIAGRAM

Fig. INT. 2 illustrates the basic components of the Core Memory System. As illustrated in the block diagram, the memory system consists of the Memory Control/Multiplexer (GE logic drawing, 70C180872) and the Memory Module (GE logic drawing, 70C180023).

The Memory Control/Multiplexer Module acts as the communication link between the user devices, establishes the priority of memory requests, multiplexes the data and address bits from the user devices to the memory, and provides parity generation and parity error detection.

The memory module contains the core stack to store the data bits, the address register to control the selection of the desired core address, a data register to hold the data read from memory or to be stored in memory, and the circuits required to retrieve data from, or store data into, the addressed memory location.

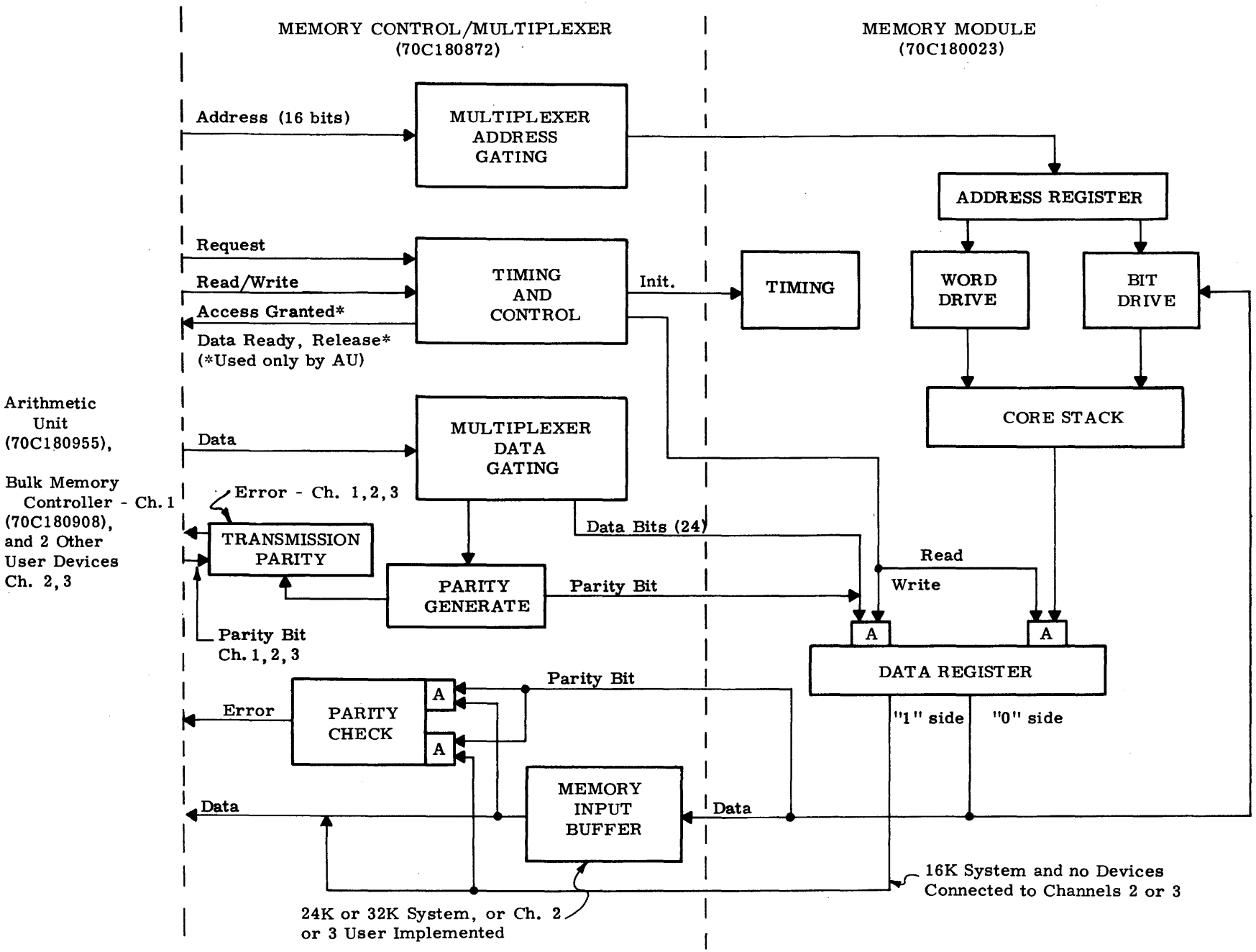


Fig. INT. 2 Memory Module Block Diagram

There are two basic operational modes of the memory module: Read (read/restore) data from memory and store (clear/write) data into memory. Therefore, to access memory the requesting user device must transfer, in addition to a request signal, a signal which specifies the read or write operation and the bits to specify the desired memory address. Also, if the operation is write, the data bits to be stored in the addressed location must be transferred to the memory module.

Actual operation of the memory is very similar for both a read and write operation. The basic differences are: (1) Parity is checked during a read operation and parity is generated during a write operation, and (2) During a read operation, the data sensed from core is gated to the data register and during write, the data from the user device is gated to the data register. In either mode, the contents of the data register are stored in the addressed memory location.

If it is a read operation, this data will be the data previously read from core. If it is a write operation, this data will be the data transferred from the user device.

Read/Restore

The Read/Restore mode of memory operation is used to obtain the contents of a memory word and to restore the same data in the memory word. The read/restore cycle is initiated by a request from the user device accompanied by a read ($\overline{\text{store}}$) signal from the device. If more than one user requests memory, the memory control unit determines which unit shall be granted access. The bulk memory controller has highest priority and the AU lowest. Any memory cycle which has been started before a higher request is received, will be completed. The memory control section then sends a signal to initiate the operation of the memory module and transfers the address of the desired memory word to the memory address register of the memory module. Decoding of this memory address enables the Word Drive and Bit Drive selection lines to switch all cores of the addressed word that are ones to the zero state. Switching these cores to the zero state induces a voltage in the sense line. This voltage is then amplified and gated to the data register. From the data register, the data read from the addressed memory word is transferred to the user device. If an even number of one bits is contained by the data word, the parity check circuitry generates an error signal which is applied to the arithmetic unit. A data ready signal is applied from the memory control logic to the user device to gate the data to the user device. The data contained by the data register is also applied to the Bit Drive Selection circuitry. During the write (restore) portion of the memory cycle, the corresponding Bit Drive Select line is driven only if a one is to be stored in the corresponding bit position. A memory release signal is generated by the memory control logic to indicate the end of the memory cycle. The memory release signal is only used by the AU.

Clear/Write

The Clear/Write mode of memory operation is used to store new data in an addressed memory word. The Clear/Write mode is initiated by a request signal from the user device along with the write (store) signal. The priority of the request signal is determined by the memory control logic and when access is granted, a signal is sent to the user device. The core memory address is then applied to the memory unit along with an initiate signal to start the memory cycle. The address is decoded from the address register and enables the corresponding Word Drive and Bit Drive selection lines. Driving the select lines flips the cores of the addressed word to the zero state. Since this is a write (store) operation, the voltage induced in the sense lines is not used. Instead, the data from the user device to be stored in the addressed memory word is gated through the memory control unit where parity is generated. The data word to be stored along with the parity bit is gated to the data register of the memory unit. The write portion of the memory cycle is then entered. The contents of the data register are then used to control the Bit Drive select lines. Each one bit in the data register will enable the addressed Bit Drive line and set the corresponding core to the one position. In this manner, the data bits from the user device are stored in the addressed memory location.

BASIC CORE THEORY

The magnetic properties of ferrite cores make them ideal high-speed storage elements. If, as shown by Fig. INT. 3, a DC current is applied to a wire threading the core, the magnetic flux around the wire induces a similar flux pattern in the core. According to the left-hand rule, the direction of the induced flux field depends on the direction of current flow in the wire. Thus, the core may be forced to a known magnetic state by driving current in one direction and this state may be reversed by reversing the current. Since the core is capable of retaining an induced magnetic field indefinitely, it possesses all the bistable characteristics necessary to store binary information.

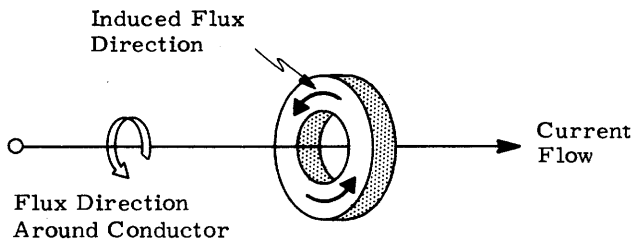


Fig. INT. 3 Flux Induction in a Ferrite Core

A certain amount of energy is required to change a core from one magnetic state to the other. If flux density (β) within the core is plotted as a function of drive current (h), a graph similar to that shown by Fig. INT. 4 is obtained. The area bounded by the curve represents the power expended to overcome hysteresis losses in the core material; for this reason, the βh curve is commonly called a "hysteresis loop".

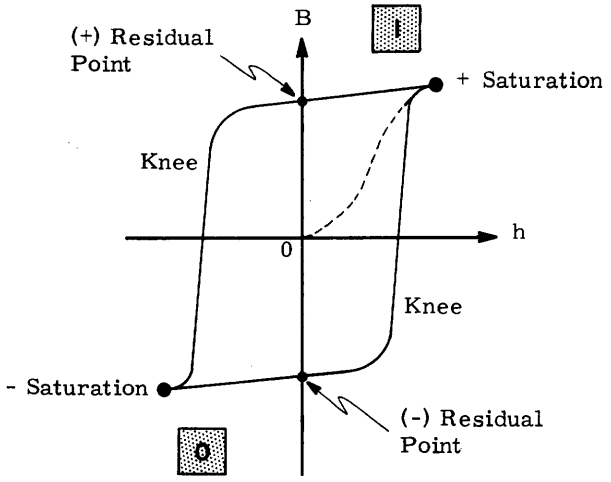


Fig. INT. 4 Basic βh Curve

Assuming the first drive current applied to the core is of sufficient magnitude in the positive h direction, the core follows the broken line from the origin to the point of positive β saturation. When this drive current is removed, the core retains most of the induced flux as shown by the slight slope of the curve as it returns to the (+) residual point of zero h.

If current is now increased in the negative h direction, flux density decreases slightly until the threshold point or "knee" of the hysteresis loop is reached. At this point a slight increase in h is sufficient to cause a rapid reversal of the magnetic field and the core is "flipped" from one state to the other. A similar sequence is followed when the core is moved from the (-) residual point to the (+) saturation point by a (+h) current. Notice that a (+h) current applied to the core has little effect if the core lies at the (+) residual point, since it only forces the core a little farther into saturation; the same is true if a (-h) current is applied to the core which is at the (-) saturation point.

When the core is magnetized to the upper portion of its hysteresis loop it is said to contain a 1; if it lies on the negative portion of the curve it contains a 0. The amount of drive current necessary to flip the core from one magnetic state to the other is called a "full-select" current.

In order to read information from the core, an additional wire, called a sense winding, is threaded through the core. When information is to be read, a negative full select current is passed through the core and the resultant change of flux is sampled by the sense winding. If the core contains a zero, the flux change is negligible; if the core contains a one, however, the large change of flux as the core flips induces a voltage pulse in the sense winding. A high-gain amplifier, called a Sense Amplifier, monitors the output of the sense winding to detect the presence of a one-bit. Since reading sets the core to zero, the stored information is essentially lost; hence, the term "destructive read" is applied to core memories. In order to retain information, a complete memory cycle must consist of both a read and write operation.

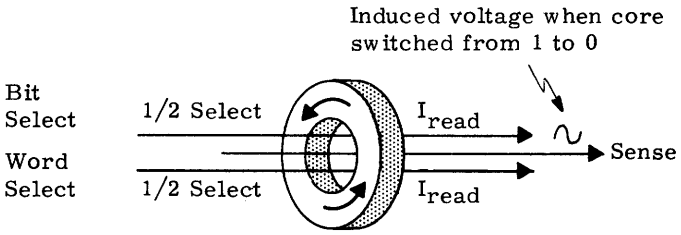
The method of writing a one-bit is simply to drive a full-select current through the core. Since the cores are cleared by the read current, writing a zero is accomplished by inhibiting the full select current through the core. Therefore, a zero is written into core by not writing a one.

2 ½ D Coincident Current Selection

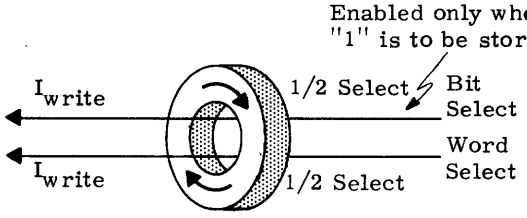
In the preceding discussion of Basic Core Theory, selection of a memory cell for reading or writing is shown to be possible through the aid of a full select current applied through a core and a sense wire to detect a change of flux in the core. The complexity of selecting a core with a single drive line is prohibitive, however, in the light of 16,384 separate selections necessary for even the smallest memory size.

In order to minimize the address selection logic, each core is threaded by two drive windings, designated Bit Select and Word Select, as shown in Fig. INT. 5.

The core is selected by coincident half-select currents in the Bit and Word Select lines rather than by a full-select current on the single line previously described.



(a) Basic 2 1/2 D Read



(b) Basic 2 1/2 D Write

Fig. INT. 5 Basic 2 1/2 D Operation

Fig. INT. 6 illustrates the effect of various combinations of half-select currents on a memory core. Notice the ratio of the sensed output signal, V sense, for a full-selected current core compared to the sensed output V_h for a half-selected memory core. This relationship permits one Bit Select line and one Word Select line to be threaded through many cores in a matrix. The only core affected being the one in which the Bit and Word select lines meet with current flowing in the same direction.

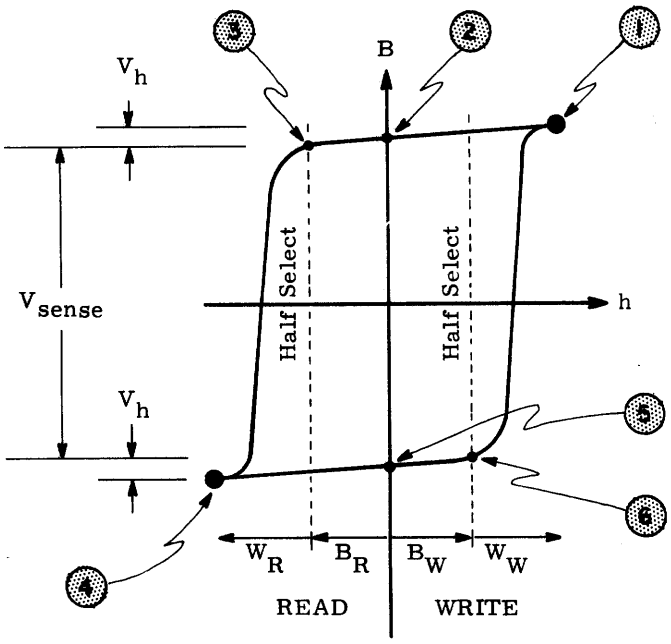
During a write cycle, current is reversed in the Bit and Word select lines to induce flux in the one direction if a one is to be stored. Control of whether a one is to be stored or a zero is to be stored is accomplished by enabling the Bit Select current only when a one is to be written. Inhibiting current flow during the write cycle in the Bit Select line leaves the core in the zero state.

Figs. INT. 7 and INT. 8 illustrate two bits of an 8-word memory. Fig. INT. 7 illustrates the direction of current flow in the Bit and Word Select lines to read the contents of two different words. Notice that only one word (bits 0 and 1) is selected by having both the Bit Select and Word Select current in the same direction. The locations, 00_8 and 10_8, are provided with arbitrary addresses to illustrate that by changing the direction of current flow through the select lines, different addresses may be selected. Therefore, when current is flowing in the direction of the solid arrows, address 00_8 will be addressed and the bit 0 and bit 1 cores will be flipped, if previously a one, inducing a signal in the sense lines for interpretation as stored ones in these positions. Addressing cell 10_8 will cause the bit 0 and bit 1 cores for this address to flip, as indicated by the unshaded arrows.

Fig. INT. 8 illustrates the same 2-bit, 8-word memory for writing a one back in both bit positions of arbitrary addresses 00_8 and 10_8. Again, the solid arrows indicate the current flow for address 00_8 and the unshaded arrows indicate the direction of current flow for address 10_8. Notice that during the write cycle, the Bit Select lines are controlled by the desired data bit to be stored in each bit position. Current will flow in the Bit Select line only if a one is to be stored in the addressed bit. If a zero is to be stored current only flows in the Word Select lines leaving the core in the zero state.

MEMORY RETENTION

The magnetic core array does not require power to provide its static memory capability. A pulse of power is required to switch cores from one state to the other, but the pulse is not necessary to hold cores in their respective states. All cores remain in the state to which they have been switched because of the retentivity of the core magnetic material. If power is removed or lost, the magnetic core array retains stored information indefinitely.



W_R = Word Read current
B_R = Bit Read current
B_W = Bit Write Current
W_W = Word Write current

1. "1" Saturation Point - Reached from 3 during write cycle when B_W and W_W provide current in the same direction.
2. "1" Residual Point - Reached from 1 or 3 after current is removed.
3. Read Half Select Point - Reached from 2 by W_R or B_R.
4. "0" Saturation Point - Reached from 2 or 5 during read cycle when B_R and W_R provide current in the same direction.
5. "0" Residual Point - Reached from 4 or 6 after current is removed.
6. Write Half Select Point - Reached from 5 by W_W or B_W.

V_{sense} = Voltage sensed proportional to the change in the B axis.

Fig. INT. 6 βh Curve, Coincident Current Effects

MEMORY WRAP AROUND

In systems using less than 32K words of core memory storage, instructions may be executed with an address greater than the implemented core size. If an address greater than the implemented core size is specified, a corresponding location within the implemented memory will be addressed or zeros will be read with no parity error detected.

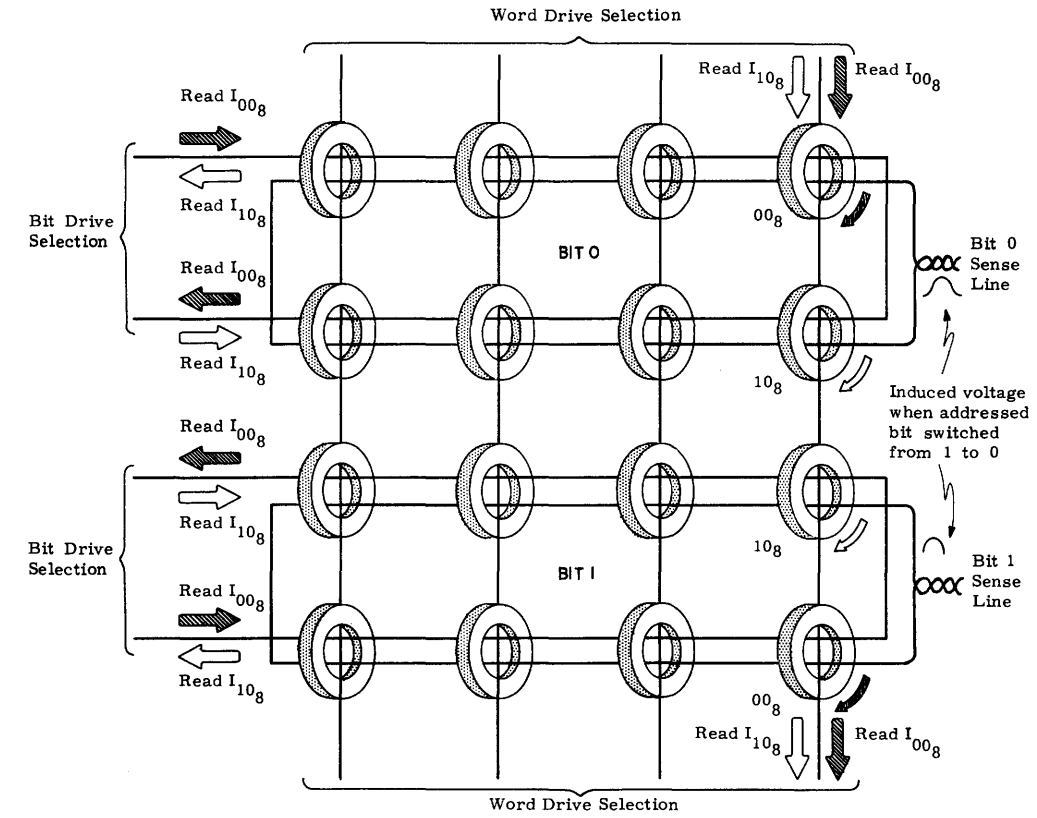


Fig. INT. 7 Read Coincident Current 2 1/2 D Selection, Basic Operation

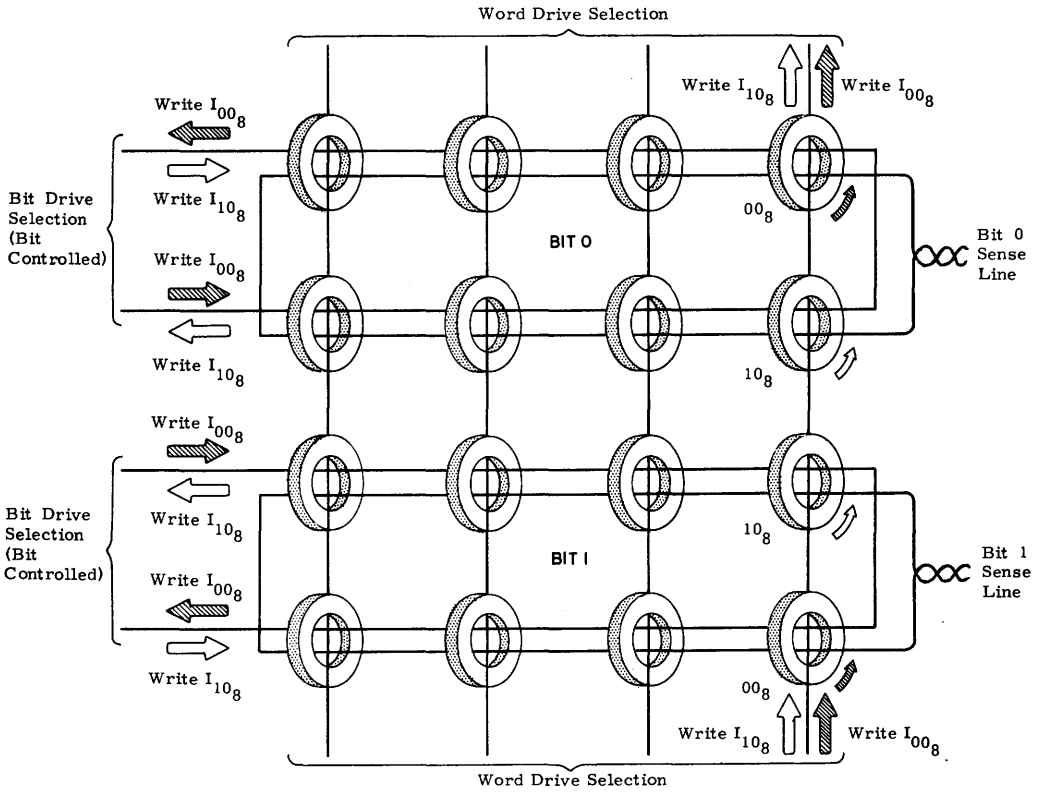


Fig. INT. 8 Write Coincident Current 2 1/2 D Selection, Basic Operation

The following chart illustrates the 4K memory block addressed and the corresponding 4K block affected for the various implemented memory sizes.

4K Block Addressed	Implemented Memory 16K	24K	32K
0- 4K	0- 4	0- 4	0- 4
4- 8K	4- 8	4- 8	4- 8
8-12K	8-12	8-12	8-12
12-16K	12-16	12-16	12-16
16-20K	0- 4	16-20	16-20
20-24K	4- 8	20-24	20-24
24-28K	8-12	*	24-28
28-32K	12-16	*	28-32

* A zero word is read with no parity error. A write operation would go undetected unless Memory Protect is used.

Memory wrap around control is provided by inserting jumper pins on the RMTB1 board in slot B18AK as shown on sheet 22.1 of logic, 70C180872. These jumper pins disable address bits not required to address the maximum core size and provide control of the parity error signal. No jumper pins are inserted when 16K of core memory is implemented.

DEDICATED MEMORY ADDRESSES

Certain memory locations are pre-assigned for specific use. These memory locations should be used only for the purpose intended unless hardware is not included to make use of the specific reserved address. The reserved memory locations and the pre-assigned uses are listed as follows:

Octal Address		Common Usage
000	-	Bulk Memory Pointer Word
001	-	SPB Link Storage
002	-	Quasi Operand Storage
003		
004		
005	-	Index Register
006		
007		
010	-	Q Register
011		
012	-	Reserved
013		
014		
015	-	Unused
016		
017		
020	-	Memory Protect Branch Vector
021	-	Memory Protect/Watchdog Instruction Storage
022		
023	-	Reserved
024	-	Watchdog Branch Vector
025		
037	-	Reserved
040		
077	-	Quasi Branch Vectors
100		
177	-	Memory Protect Status Map
200		
377	-	Interrupt Response Location
400		
	-	Quasi Package

MEMORY INTERFACE

Fig. INT. 9 illustrates the data and control signal interface associated with the core memory subsystem. Timing associated with these signals is provided in the THEORY Section of this publication.

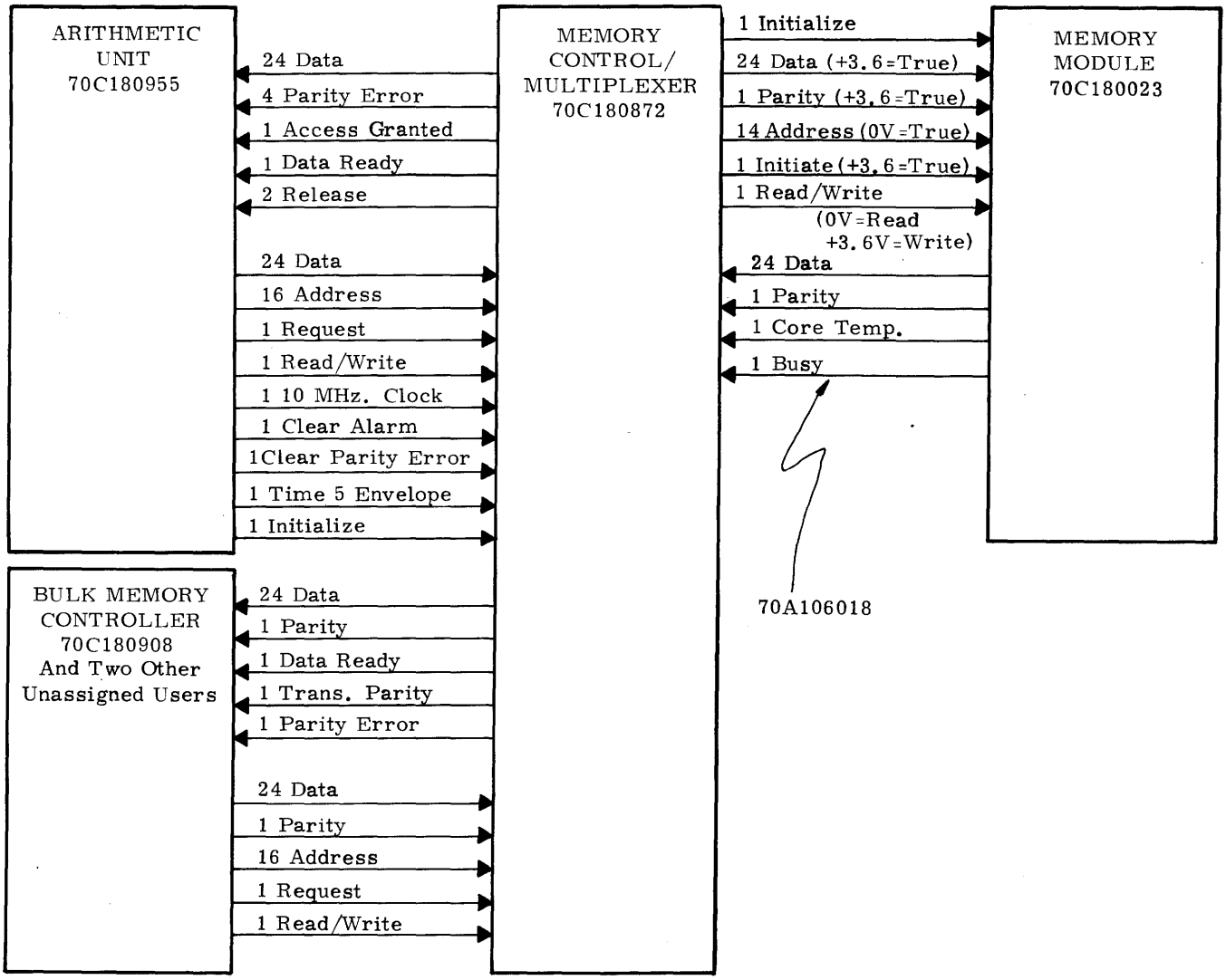


Fig. INT. 9 Memory Interface

THEORY OF OPERATION

A functional block diagram of the memory system is contained in Fig. THEORY. 1. As shown in the diagram, the Arithmetic Unit, Bulk Memory Controller, and as many as two other users may communicate directly with the memory system. Communications between these two devices and the core memory module is on a true cycle stealing basis with the Bulk Memory Controller having highest priority and the Arithmetic Unit having lowest priority.

The user device, whether the Arithmetic Unit, Bulk Memory Controller, or other user device, applies a memory request signal, specifies the operation (write into core or read from core) to be performed, and specifies the memory address to the memory control/multiplexer portion of the memory system.

If more than one device initiates simultaneous requests, the priority logic of the memory controller/multiplexer will grant access on a priority basis. A channel making requests will obtain successive accesses to memory, provided requests from higher priority channels are not present. In any case, the memory controller/multiplexer generates an access granted signal to the user granted access.

After the access priority has been established, a signal is applied to the core memory module to indicate the type of operation and to initiate the memory cycle.

If the operation is write (Clear/Write) into core, the data transferred from the user device is gated through the Multiplexer Data Gates. The parity bit, if required, is generated and the data is transferred to the Data Register of the memory module. The address of the core location to be affected is multiplexed from the user device to the Address Register of the memory module. The address is decoded and the corresponding word and bit drive lines are enabled. Enabling the Word and Bit Drive lines will cause the cores in this memory location to clear. Since this is a write into core operation, the data sensed from this location is not used. The "one" data bits contained in the Data Register then control the decoded Bit Address lines to set the corresponding data bits of the addressed location. Setting the core cells of the addressed location to the configuration of "one" bits in the Data Register completes the memory cycle. When the Arithmetic Unit has access to memory, a Memory Release signal is applied from the Memory Control logic to the AU near the end of the cycle. At the end of each memory cycle, priority is re-established for subsequent memory requests.

Read (Read/Restore) core memory operations require the same determination of priority after receipt of the memory request signal. The multiplexed address is again decoded to clear the bits of the corresponding memory location. The data sensed during this clearing operation is applied through the sense amplifiers and gated into the Data Register. The data contained in the Data Register is then used to control the Bit Drive lines and, thereby, store the same data back in the addressed location. The data in the Data Register is also applied through the Memory Input Buffer or directly to the user device. A Data Ready signal is provided by the Memory Control Unit to set this data in the user device.

The data read from core is checked for odd parity. If a parity error exists, a parity error signal is provided. This parity error signal lights the Core Parity and Alarm indicators on the console, provides for program detection of the error (JNP), and may be optionally used to halt further accesses to memory by the AU using the Stop On Parity switch. If the parity error occurred during a data transfer to the Bulk Memory Controller, the MPLX CH1 indicator on the AU console is also lighted, a bit is set in the alert status word for program monitoring, and an indication is provided on the maintenance console.

If the user device is the Arithmetic Unit, a memory release signal is applied to the AU upon completion of the memory cycle.

The Memory Control Unit is capable of controlling up to 32K words of memory storage. A Memory Module contains up to 16K words of memory storage. Therefore, if 24K or 32K words of memory are required, two Memory Modules are used as shown in Fig. THEORY. 2. These Memory Modules are identical.

The Memory Control Unit will then initiate the proper 16K Memory Module addressed by using memory address bit 14. If memory address bit 14 is a "zero", then the lower 16K Memory Module operation is initiated. If memory address bit 14 is a "one", then the upper 16K Memory Module operation is initiated.

CORE STACK ARRANGEMENT

The core stack is a sealed unit which is maintained at a constant temperature - nominally 55°C. The capacity of the stack is sixteen thousand words. Its physical size is the same regardless of capacity - 12.5"wide x 20" high x 3" thick.

Within the encased unit are the magnetic cores, a heater element to hold the temperature constant, thermocouples for monitoring the internal temperature, and over/under temperature protection sensors used for alarming.

The magnetic stack contains four planes, a diode matrix, and the connections for the Word Drive lines, Bit Drive lines, and Sense lines. Each plane contains the magnetic cores required to store 4K (4096) words of data. Fig. THEORY. 3 illustrates the organization of the planes within the stack.

Fig. THEORY. 4 illustrates a 4K word plane including the Word and Bit Drive lines. As shown, a single plane contains 4096 cores (.03 inches in diameter) for each of the 25 bits of a word. The cores for each bit position are physically separated.

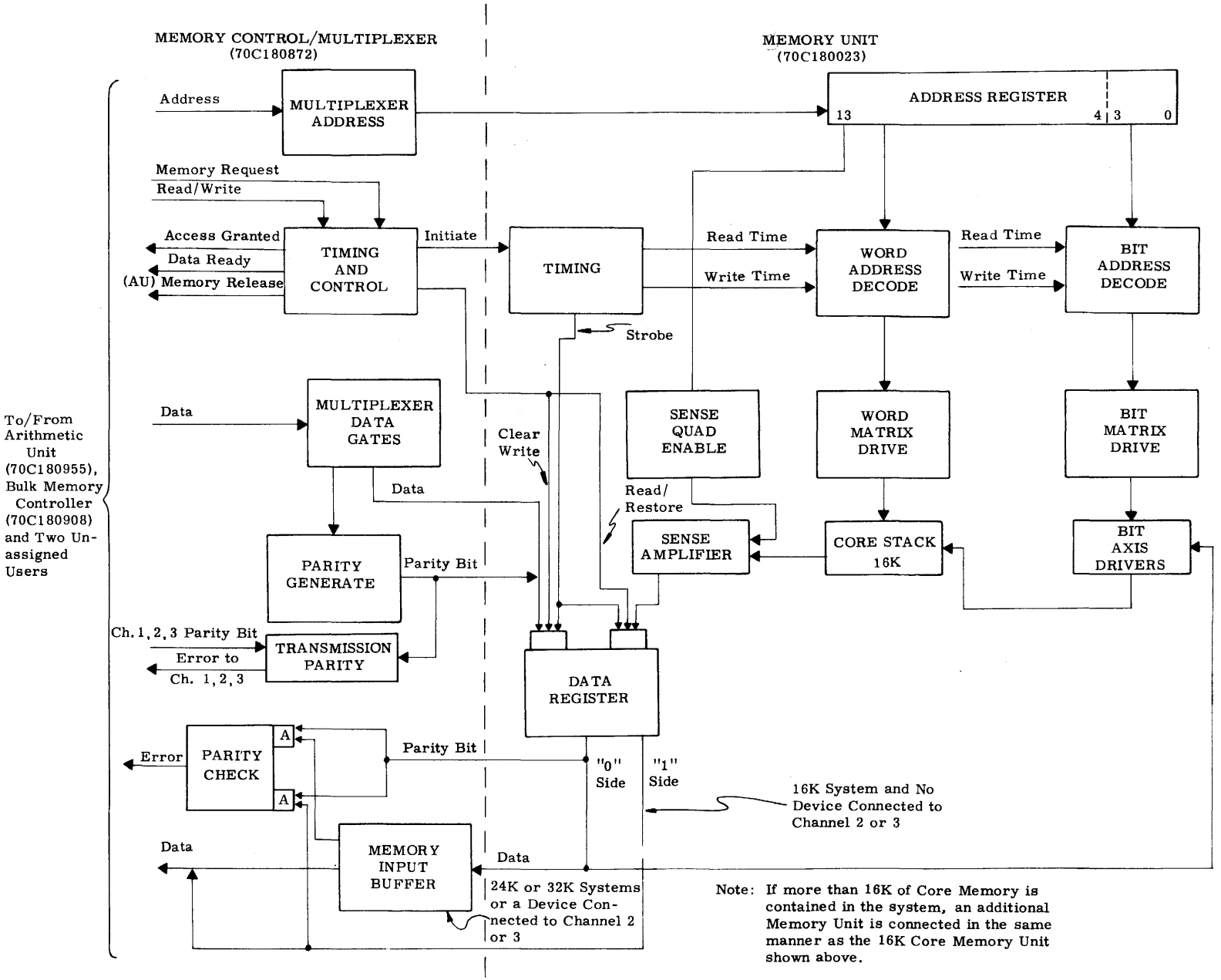


Fig. THEORY. 1 Memory System Functional Block Diagram

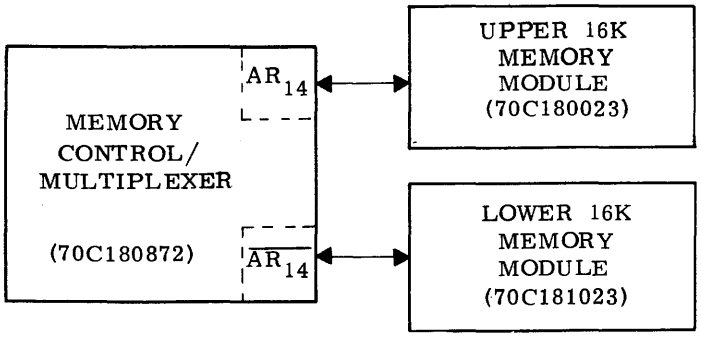


Fig. THEORY. 2 24K - 32K Word Memory, Block Diagram

The 4096 cores for each bit position are organized in a 16 by 256 matrix. To simplify the understanding of address decoding, the 16 by 256 matrix for each bit is further divided into a 16 by 16 group as shown in the figure.

A close study of Fig. THEORY. 4 will illustrate the basic method of addressing one core in each of the 25 bit positions. Notice that although a signal for selecting the Word Line Diode End is common to one of the 16 lines for each of the 16 groups, only one of these lines will be enabled by the Word Line Group End address signal. Therefore, the Word Line enabled passes through 16 cores of each bit position. Individual Bit Diode Matrix circuits are provided for each bit position. The Bit Diode Matrix circuits enable one of 16 lines to each bit position. The enabled Bit Line selects the addressed core cell from the 16 cores selected by the Word Line. Also, notice that current may flow in both directions in both the Word Line and the Bit Line. Current flows in one direction to read the contents of the addressed cell and current flows in the other direction to set the addressed core cell to the "one" position.

A detailed discussion of address selection is provided later in this section. The previous discussion is provided only as a basic understanding of the core stack organization and interconnections of the drive lines.

A single sense line is threaded through all cores of each bit position in a 4K plane. Therefore, each plane will contain 25 sense lines. Each sense line will be threaded through 4096 core cells. The sense windings are threaded through the 16 by 256 bit matrix of each bit position in a rectangular fashion with wire transpositions spaced at 128 core intervals as shown in Fig. THEORY. 5.

PRIORITY ACCESS

The memory control/multiplexer allows both the Arithmetic Unit, Bulk Memory Controller, and as many as two unassigned users to communicate directly with the memory module, one at a time. Since these modules obviously cannot use the memory at the same time the memory control unit must determine which unit should have access. If both the Arithmetic Unit and the Bulk Memory Controller request memory within an existing memory cycle, the Bulk Memory Controller will get access before the arithmetic unit. That is, Bulk Memory Controller requests take priority over arithmetic unit requests.

As shown on sheet 7 of 70C180872, requests from the Bulk Memory Controller are applied to channel 1 Priority Request flip-flop, F1MPR1, and requests from the arithmetic unit are applied to F1MPR4. Requests from unassigned user devices are applied to F1MPR2 and F1MPR3. When no requests are present, or at the completion of the current memory cycle (MT17) the Priority Request flip-flop corresponding to the request is set. Access priority is then established in N0MBR1, G0MBR2, G0MBR3, and D0MBR4.

$$N0MBR1 = F1MPR1$$
$$G0MBR2 = F1MPR2 \cdot \overline{F1MPR1}$$
$$G0MBR3 = F1MPR3 \cdot \overline{F1MPR2} \cdot \overline{F1MPR1}$$
$$D0MBR4 = F1MPR4 \cdot \overline{F1MPR1} \cdot \overline{F1MPR2} \cdot \overline{F1MPR3}$$

As shown by the logic equations, channel 1 requests have highest priority, and channel 4 requests have lowest priority. Any memory cycle which has started before a higher priority request is received will be completed. At the end of each memory cycle, priority is re-established. A channel making requests will obtain successive memory cycles as long as the requests are made and channels of higher priority are not making requests.

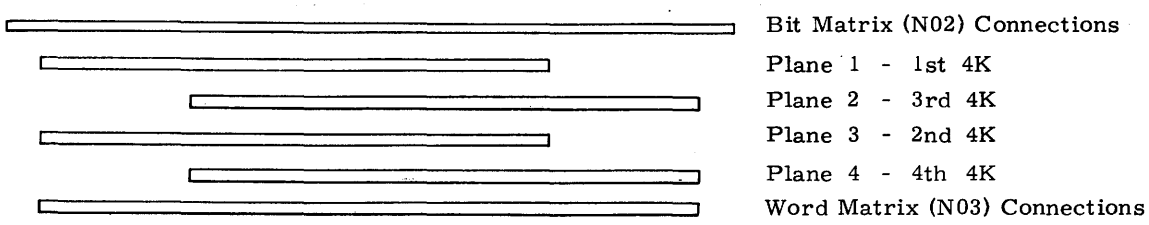


Fig. THEORY. 3 Core Stack Arrangement

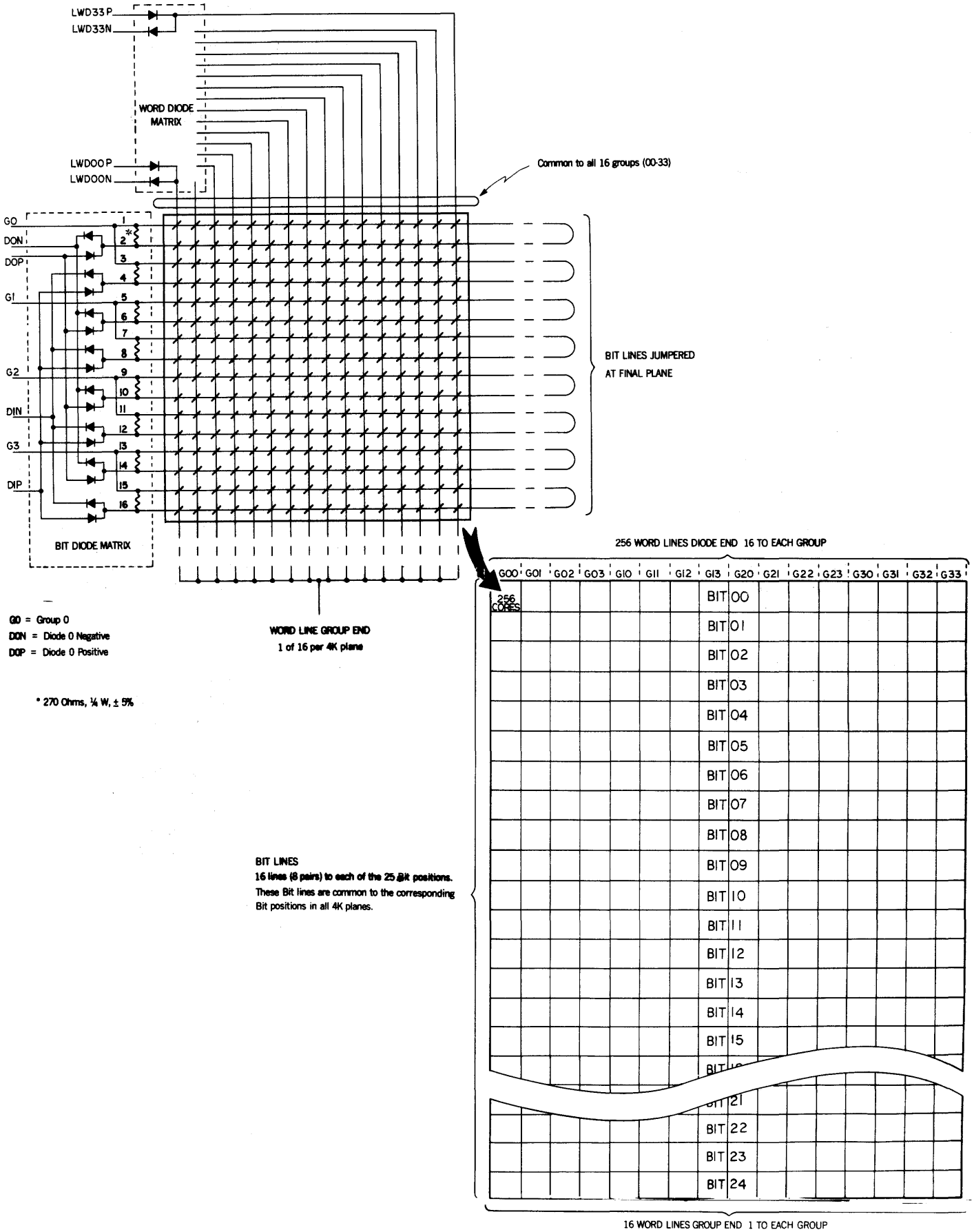
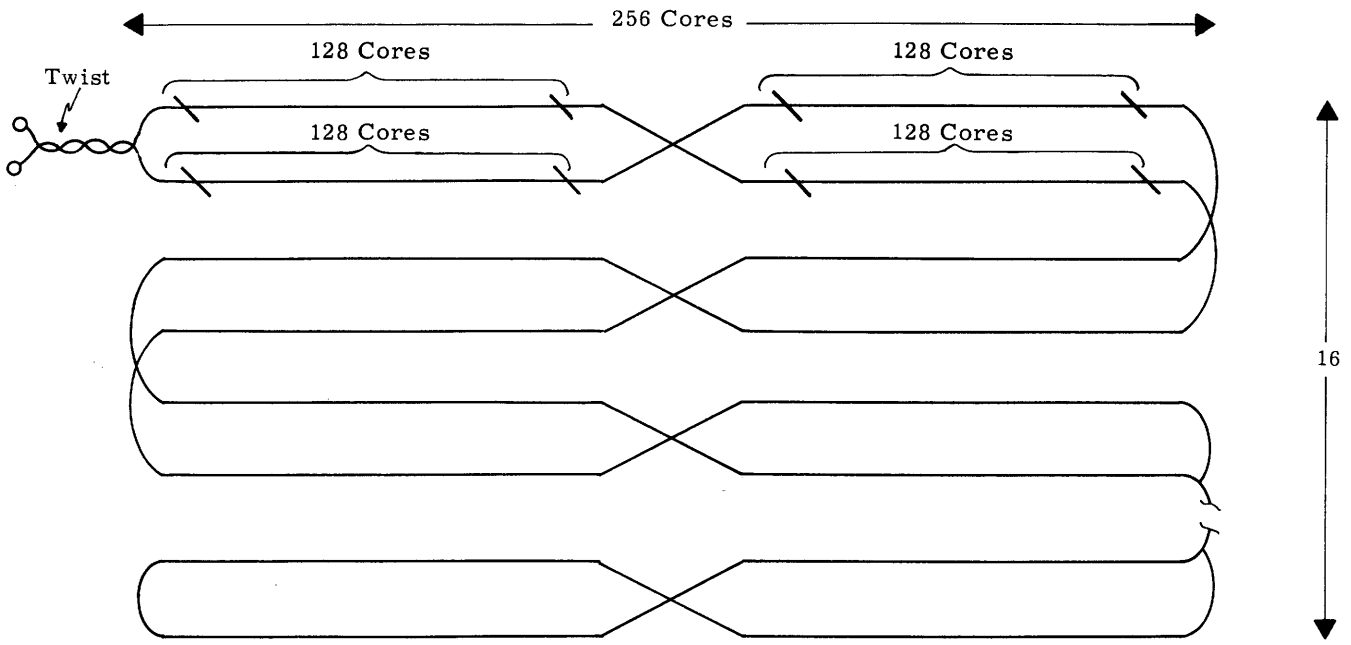


Fig. THEORY. 4 4K Plane Organization and Addressing



NOTE: A sense winding threads 4096 cores of a bit position.

Fig. THEORY. 5 Sense Winding

Enabling one of the priority gates starts the memory cycle and enables access to memory for the corresponding device. Address and data control of the memory is then allocated to the user device granted access.

MEMORY CONTROL TIMING

Fig. THEORY. 6 contains a timing diagram with logic equations of the memory control timing. The memory control unit timing is initiated by a request signal from a user device and then operates independent of the memory unit timing. Fig. THEORY. 6 illustrates a request signal originating from the arithmetic unit (AUG1SMRQ), however, the request signal from the Bulk Memory Controller or other user devices would be similar. The memory control timing applies an initiate signal to the memory module to start memory timing and applies a signal to indicate the memory operation, either read (read/restore) or write (clear/write). The memory control unit then multiplexes the address to the memory module from the requesting device and transfers data from the user device to the memory module (write) or transfers data from the memory module to the user device (read).

The first clock pulse following the request signal sets the Priority Request flip-flop (F1MPR4) and resets the Time Counter Enable flip-flop (F1MTCE) provided that the memory unit is not busy. Setting F1MPR4 enables D0MBR4 if there is no request from the other user devices.

Enabling D0MBR4 enables D1MBE4 and D1MAE4 to:

(1) Gate the memory address from the AU through the address multiplexer (G0MA14 - 00) to the stack selection circuitry (D1M1SL, U, and D1M2SL, U) and the memory module.

(2) Gate the data bits from the AU through the data multiplexer (G1MD23-00) to the memory module.

(3) Gate the read/write (store) signal from the AU to the memory module (G0MWRQ) to specify the mode of operation.

(4) Enable (G1MRRQ) the parity check circuitry during read ($\overline{\text{STORE}}$) memory operations, and

(5) Enable the Data Ready and Memory Release signals to the AU when they are generated.

The read/write signal (AUG0STOR) from the arithmetic unit enables G0MWRQ for write (store) operations and enables G0MRRQ for read ($\overline{\text{store}}$) memory operations. When G0MWRQ is enabled, the clear/write mode of memory operation is enabled. When G0MWRQ is disabled, the read/restore mode of memory operation is enabled.

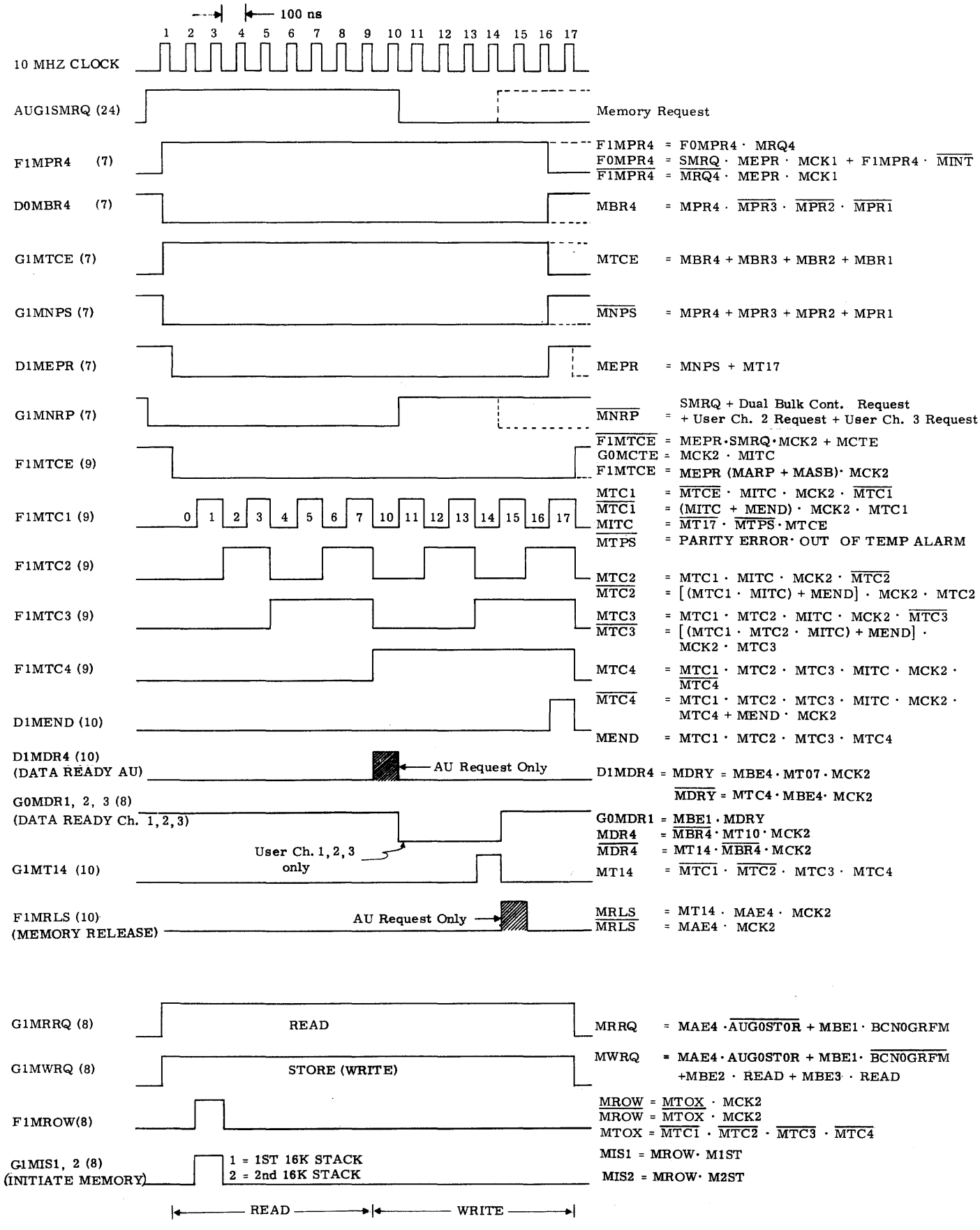


Fig. THEORY. 6 Memory Control Timing

The memory time counter (F1MTC1 - 4) is a binary counter incremented to a count of 17_8 by the 10 MHz clock pulses to provide control throughout the memory cycle. The time counter is enabled following access granted, provided: (1) there is no parity error while an out of temperature alarm exists (G0MSTF) and (2) the memory module is not busy due to a hardware error (G1MASB).

An initiate signal (G1MIS1) is applied to the memory module after access to memory has been granted. Applying this initiate signal to the memory module starts the memory module timing and the read or clear portion of the memory cycle is started. The memory module then operates independent of the memory control timing. The initiate signal is only applied to the 16K core stack addressed, as controlled by the memory address and implemented memory size for memory wrap-around.

The memory time counter continues to be incremented by each 10 MHz clock pulse until a count of 17_8 is reached. At this time the memory module has completed the write or restore portion of the cycle. Therefore, a new request may be serviced.

If the arithmetic unit is the user device, a release signal (F1MRLS) is applied to the arithmetic unit to terminate the current sequence state when the time counter is equal to 15_8. A new memory cycle will not be initiated until after the memory time counter has been incremented to 17_8 (D1MEPR).

Fig. THEORY. 7 illustrates the relationship of the memory control/multiplexer timing with that of the arithmetic unit. Notice that this relationship differs slightly when repetitive memory cycles occur as compared to memory cycles that follow a "rest" state.

MEMORY MODULE TIMING

Fig. THEORY. 8 illustrates the timing of the memory module. Because of the operational speed of the memory unit, circuit delays become important when considering the memory timing. These delays are taken into account in the timing diagram.

These signals are generated from the initiate signal (G1MIS1(2) generated in the memory control unit when access has been granted to a requesting user device. This initiate signal is ANDed with memory busy and the buss level monitor in G0TIMS. If the memory is not busy and the buss level monitor has not detected a power failure, the initiate memory cycle flip-flop (F1TIMC) is set. Setting F1TIMC initiates the memory cycle by applying a pulse to the time delay circuits and by setting the memory busy flip-flop (F1TMBU).

The timing pulse applied to the delay circuits provide precise timing signals for control during the memory cycle. Three delay circuits are provided as shown on sheet 8 of the memory logic (70C180023). Delay circuits 1 (DL1) and 2 (DL2) provide taps for obtaining delays of the timing pulse from 25 nanoseconds to 1650 nanoseconds in 25 nanosecond intervals. Delay circuit 3 (DL3) is used to generate the strobe pulse providing more precise delay increments of 5 nanoseconds from the timing pulse previously delayed in delay circuit 1.

The timing signals obtained from these delay circuits are wired to the delay tap pins to provide optimum operation of the memory module. These delays should not require changing unless component deterioration or component replacement occurs.

Each of the control signals illustrated in Fig. THEORY. 8 are described as to the function performed in the following text. A more detailed understanding of these signals will be obtained from the discussion of memory addressing, and the read/restore and clear/write modes of memory operation contained later in this section.

G1MIS1(2) - Initiate:

The initiate signal originates in the memory control unit when access has been granted to a requesting device. This signal is applied from G1MIS1 to the first 16K stack of memory storage. If the system contains more than 16K words of memory, the initiate signal is applied from G1MIS2 to the upper 16K stack. In systems with more than 16K words of storage, the initiate signal is applied only to the addressed 16K memory module.

This initiate signal is used to start the memory timing by applying a pulse to the delay circuits from which the remaining timing signals are obtained.

D1TGAB - Gate Address, Bit:

This signal gates address bits 3 through 0 from memory control to the address register (F1AR03-F1AR00) of the memory module. These bits are decoded to select the bit drive lines.

At the same time that D1TGAB is enabled, D1TGAL and D1TGAU are enabled to gate address bits 4 through 12 to the lower and upper 8K memory address register, respectively, of the memory module. These bits are decoded to select the word drive lines. D1TGAL is also used to gate address bits 12 and 13 to the memory address register from the memory control unit. These bits are decoded to select the sense quad enable signal (D1TSQ0-3) corresponding to the 4K plane addressed.

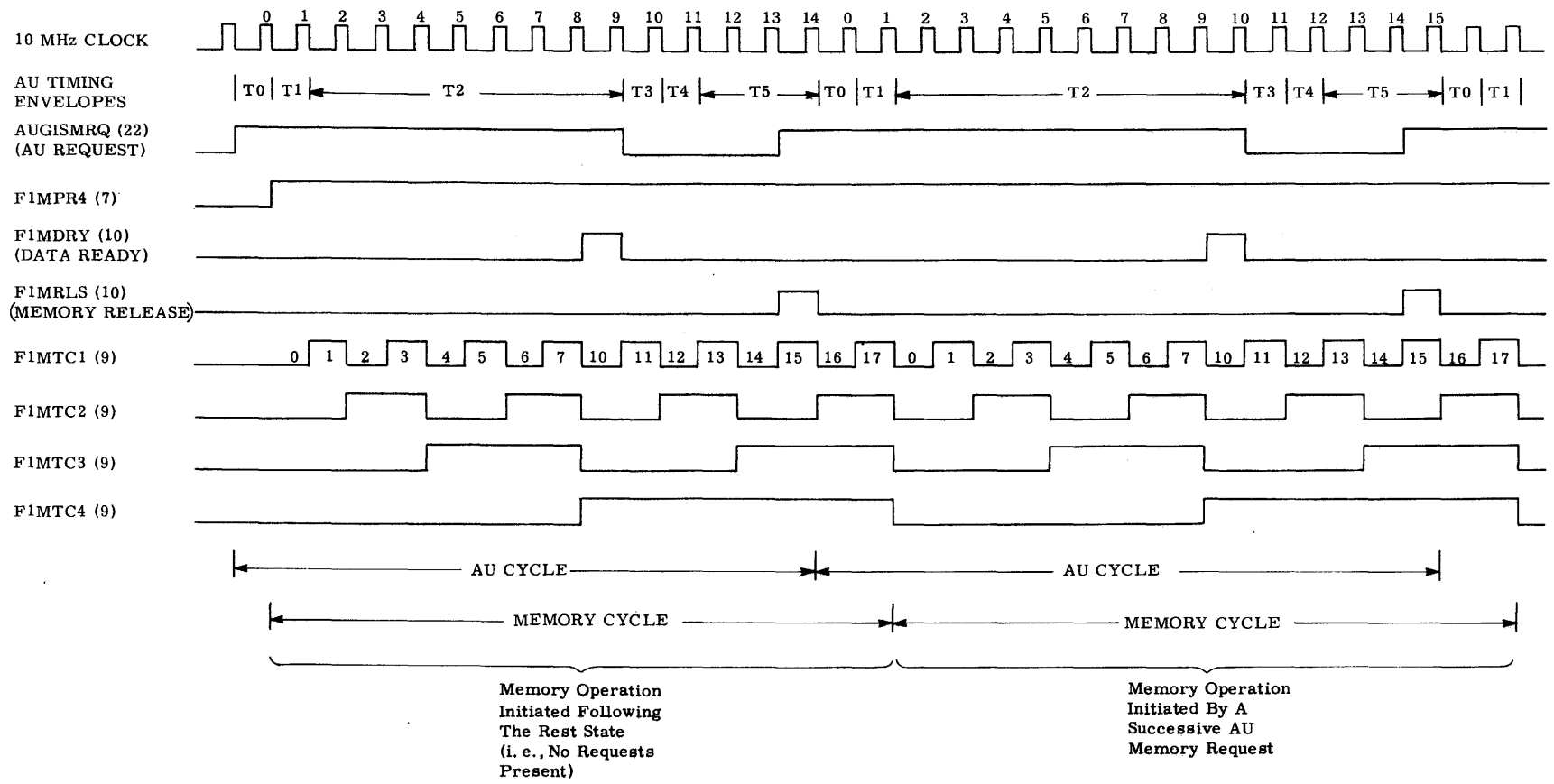


Fig. THEORY. 7 AU/Memory Cycle Relationship

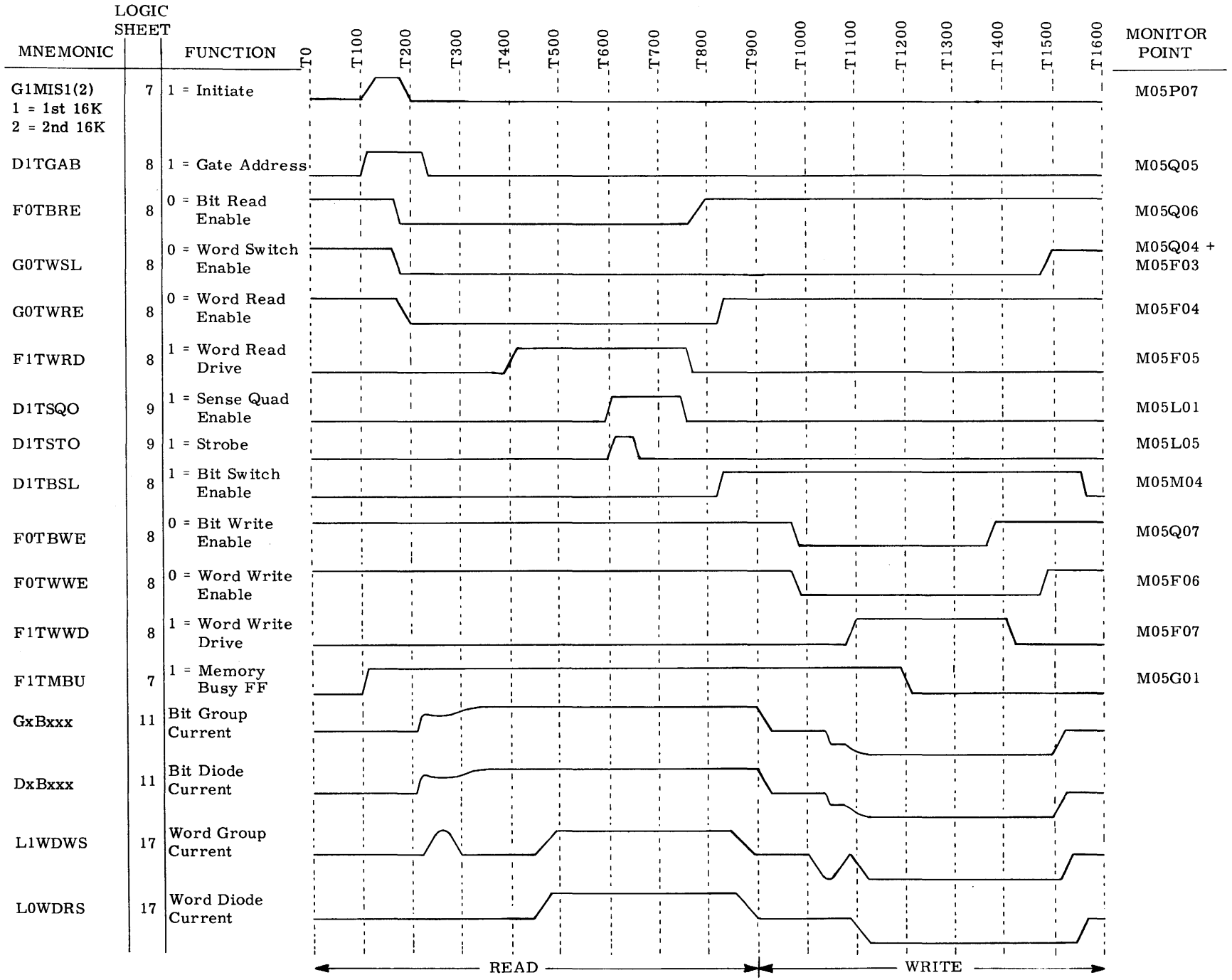


Fig. THEORY.8 Memory Timing

F0TBRE - Bit Read Enable:

The Bit Read Enable signal enables decoding of address bits 3 through 0 to enable the addressed bit drive lines for the read or clear portion of the memory cycle. F0TBRE controls the time that current is allowed to flow through the addressed bit drive line of the read portion of the memory cycle.

G0TWSL - Word Switch Enable:

G0TWSL supplies one input to enable decoding of address bits 12 through 4 for selection of the addressed word drive lines. The other enable signal required for selecting the word drive lines is the word read enable or word write enable signal described below.

G0TWRE - Word Read Enable:

This signal enables selection of the word drive lines for the read or clear portion of the memory cycle.

F1TWRD - Word Read Drive:

F1TWRD enables the voltage source for the word drive lines. Therefore, current will flow through the addressed word drive lines. In conjunction with the current flow through the bit drive lines, full select current will flow through the addressed word and the core will be flipped to the zero state.

D1TSQ0-3 Sense Quad Enable:

This signal enables the sense amplifier to detect the change in flux for a core cell flipped from the one to zero state. Only the 4K plane addressed by address bits 12 and 13 will be enabled. The output from the enabled sense amplifier will be gated to the memory data register if the operation is read.

D1TST0-3 Strobe

This signal will gate data into the memory data register during a clear/write operation. The data gated to the memory data register will be the data from the user device that is multiplexed through the memory control unit.

D1TBSL - Bit Switch Enable:

D1TBSL gates the contents of the memory data register to the bit matrix for controlling the bit drive lines during the write or restore portion of the memory cycle. Only the addressed bit lines corresponding to one bits in the memory data register will be enabled. This will flip the addressed cores corresponding to ones in the memory data register and leave the remaining cores in the zero state.

F0TBWT - Bit Write Enable:

F0TBWT enables the decoding of address bits 3 through 0 for selection of the bit drive lines for the write portion of the memory cycle.

F0TWWE - Word Write Enable·

F0TWWE enables decoding of address bits 12 through 0 for selection of the word drive lines for the write or restore portion of the memory cycle.

F1TWWD - Word Write Drive:

This signal enables the voltage source for the selected word drive lines and, therefore, allows current to flow through these lines. Therefore, with current flowing in the word drive lines and in the bit drive lines for those cells in which a one is to be stored, full select current flows, flipping them to the one state.

F1TMBU - Memory Busy:

This signal inhibits initiation of a memory cycle until the present cycle is completed. In this manner, protection is provided from destroying desired data.

Current Signals:

These signals are provided to illustrate the current waveforms on the drive line.

MEMORY ADDRESS SELECTION

The following text describes the word drive line and bit drive line selection logic used to enable full select current to flow through the core cells of the addressed memory word. Driving specific word and bit drive lines is required to read previously stored data from a specific memory location or to store new data in a specific memory location.

The word and bit drive lines enabled, correspond to the memory address specified by the user device granted access to memory. As shown in Fig. THEORY. 9, the 15 bits of the memory address from the user device is multiplexed through the memory control unit when access is granted.

> NOTE
>
> Address bit 15 is also applied to the memory control unit. This bit is disabled (logic sheet 22. 1) by not inserting a jumper pin, since it is not required to address the maximum (32K) memory in a GE-PAC 4010 system. Therefore, this discussion does not consider bit 15 as a part of the memory address.

The stack selection portion of the memory control logic determines which 16K stack is being addressed. The initiate memory signal (G1MIS1, 2) is only enabled to the addressed stack and data is only enabled from the addressed stack. Since the initiate memory signal is required to start a memory cycle, only the stack addressed will be operative.

Jumper pins are inserted (RMTC1 board in B18) in the stack selection logic according to the memory size implemented. Logic sheet 22.1 lists the jumper pin arrangement for each memory size. These jumper pins provide memory wrap around control and permit the interchanging of stacks in those systems with 32K of memory. That is, for maintenance purposes, the core stack associated with memory locations 0 through 16K may be easily interchanged with the stack normally associated with locations 16K through 32K.

From the multiplexer address gates of the memory control unit, address bits 13 through 0 are applied to the memory module for decoding. Fig. THEORY. 9 lists the control provided by each bit.

Refer to the memory block diagram contained on sheet 5 of the memory logic (70C180023) and to the 4K Plane Organization drawing, Fig. THEORY. 4, during the following discussion.

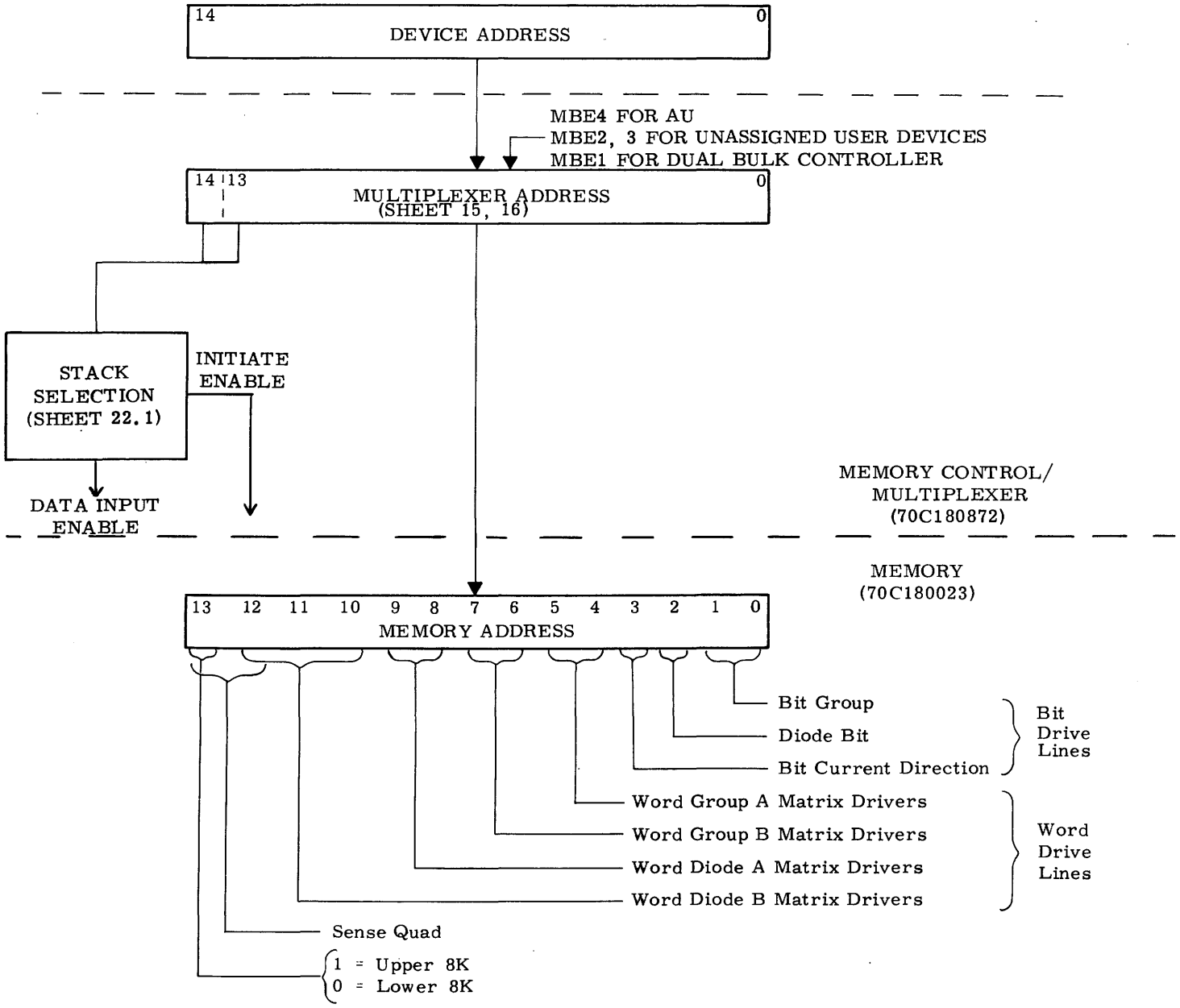


Fig. THEORY. 9 Memory Addressing

Bit 13 of the memory address is used to determine if the upper or lower 8K portion of a 16K stack is being addressed. Separate, identical decoding of each 8K portion of the stack is provided.

Bits 13 and 12 are used to determine which 4K plane is being addressed. From these bits one of four sense quad enable signals is generated to gate the data sensed from core during the read portion of the cycle through the sense amplifiers. Since each bit position of each 4K plane contains a separate sense line, the sense quad enable signal enables the sense amplifier corresponding to the sense line of the addressed plane.

Bits 12 through 4 are decoded to enable the addressed word drive lines.

Bits 3 through 0 are decoded to enable the addressed bit drive lines. During the write (restore) portion of the memory cycle the addressed bit drive lines are also controlled by the data bits to be stored in core.

Table THEORY. 1 lists the word drive line circuits enabled for all combinations of address bits 12 through 4. Both the group end and diode end circuits enabled are shown. Table THEORY. 2 lists the bit drive line circuits enabled for all combinations of address bits 3 through 0. Notice that in addition to the read or write portion of the memory cycle, the bit drive line circuits enabled also depend on whether even or odd numbered

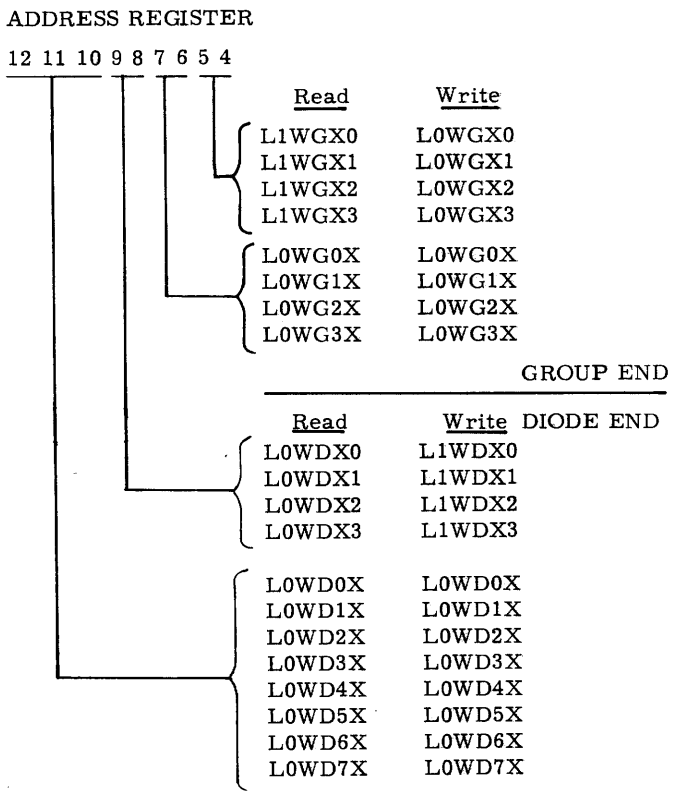


Table THEORY. 1 Word Matrix Decode

Address Bits				READ				WRITE			
				Even Bits		Odd Bits		Even Bits		Odd Bits	
3	2	1	0	Group End	Diode End	Group End	Diode End	Group End	Diode End	Group End	Diode End
0	0	0	0	G1B0XX	D0B0XX	G0B0XX	D1B0XX	G0B0XX	D1B0XX	G1B0XX	D0B0XX
0	0	0	1	G1B1XX	D0B0XX	G0B1XX	D1B0XX	G0B1XX	D1B0XX	G1B1XX	D0B0XX
0	0	1	0	G1B2XX	D0B0XX	G0B2XX	D1B0XX	G0B2XX	D1B0XX	G1B2XX	D0B0XX
0	0	1	1	G1B3XX	D0B0XX	G0B3XX	D1B0XX	G0B3XX	D1B0XX	G1B3XX	D0B0XX
0	1	0	0	G1B0XX	D0B1XX	G0B0XX	D1B1XX	G0B0XX	D1B1XX	G1B0XX	D0B1XX
0	1	0	1	G1B1XX	D0B1XX	G0B1XX	D1B1XX	G0B1XX	D1B1XX	G1B1XX	D0B1XX
0	1	1	0	G1B2XX	D0B1XX	G0B2XX	D1B1XX	G0B2XX	D1B1XX	G1B2XX	D0B1XX
0	1	1	1	G1B3XX	D0B1XX	G0B3XX	D1B1XX	G0B3XX	D1B1XX	G1B3XX	D0B1XX
1	0	0	0	G0B0XX	D1B0XX	G1B0XX	D0B0XX	G1B0XX	D0B0XX	G0B0XX	D1B0XX
1	0	0	1	G0B1XX	D1B0XX	G1B1XX	D0B0XX	G1B1XX	D0B0XX	G0B1XX	D1B0XX
1	0	1	0	G0B2XX	D1B0XX	G1B2XX	D0B0XX	G1B2XX	D0B0XX	G0B2XX	D1B0XX
1	0	1	1	G0B3XX	D1B0XX	G1B3XX	D0B0XX	G1B3XX	D0B0XX	G0B3XX	D1B0XX
1	1	0	0	G0B0XX	D1B1XX	G1B0XX	D0B1XX	G1B0XX	D0B1XX	G0B0XX	D1B1XX
1	1	0	1	G0B1XX	D1B1XX	G1B1XX	D0B1XX	G1B1XX	D0B1XX	G0B1XX	D1B1XX
1	1	1	0	G0B2XX	D1B1XX	G1B2XX	D0B1XX	G1B2XX	D0B1XX	G0B2XX	D1B1XX
1	1	1	1	G0B3XX	D1B1XX	G1B3XX	D0B1XX	G1B3XX	D0B1XX	G0B3XX	D1B1XX

XX = Bit position 00 - 24

Table THEORY. 2 Bit Matrix Decode

bits are being selected. The bit drive circuits enable current to flow in opposite directions for odd and even bits. This arrangement aids in preventing interaction between adjacent bit positions.

Fig. THEORY. 10 illustrates, in block diagram form, the address selection circuits affected for even and odd bits of memory location 00000_8. This diagram illustrates the direction of current flow and the corresponding circuits enabled for both the read and write portions of the memory cycle.

Notice that current flowing in both the bit and word drive lines provide full select current through the cores shown. During the read portion of the memory cycle, current flows in a direction that will flip cores in the "one" state to the "zero" state. The resultant flux is induced in the sense line, amplified by the sense amplifier and if the memory operation is read/restore, a "one" is strobed into the memory data register. If the memory operation is clear/write, this data bit will not be strobed into the memory data register. Instead, the data from the user device is gated to the memory data register. In either case, the read portion of the memory cycle will place all bits of the addressed memory location in the "zero" state.

Following the read portion of the memory cycle, the write portion of the cycle is entered. Current is enabled in the opposite direction during the write portion of the cycle to flip the cores, corresponding to "one" bits in the data register, to the "one" state. Only the bit drive lines corresponding to "one" bits in the memory data register are enabled.

Figs. THEORY. 11 and THEORY. 12 illustrate the detailed operation of the read and write cycle, respectively, for the bit and word drive lines associated with memory address 00000_8, bit 0. These drawings illustrate the relative potential, plus or minus, at the output of the circuit areas, and the direction of current flow. Using these diagrams, the operation of any word and bit drive line circuits may be determined.

READ/RESTORE OPERATION

The read/restore mode of memory operation retrieves data previously stored in a memory location and restores the same data back in the addressed memory location. The read/restore mode is initiated when a user device has been granted access to memory and the read (store) mode is specified by the device.

The read/write or store signal from the user device is applied to the memory control unit where G0MRRQ is enabled and G0MWRQ is disabled for the read/restore mode. G0MRRQ enables the parity check function of the parity check-generate logic. G0MWRQ applies a "one" to the memory unit enabling the read cycle enable gate D1TRRE. D1TRRE enables the data sensed from core to be gated to the memory data register for transfer to the user device and for control of the bit drive lines during the write portion of the memory cycle.

Fig. THEORY. 13 illustrates the data flow for the read/restore memory operation. During the read portion of the memory cycle, the bit and word drive lines are enabled according to the memory address specified by the user device. These drive lines provide full select current to flip the addressed core cells that are in the "one" state to the "zero" state. Flipping these cores to the "zero" state induces a voltage into the sense lines. The sense quad enable signal (D1TSQ_) is enabled corresponding to the 4K plane addressed. This sense quad enable signal gates the sense winding signal for each bit through the sense amplifiers. The sense amplifiers convert the sense winding signals to logic levels. From the sense amplifiers, the data read from core is gated into the memory data register since this is a read/restore operation (i.e., D1TRRE is enabled).

The data contained by the memory data register is then used to control the bit drive lines during the write portion of the memory cycle to restore the contents of the memory location addressed. Bits 23 through 0 of the memory data register are also transferred to the user device. Transfer to the user device occurs through the memory input buffer of the memory control unit (if the system contains 24K or 32K words of memory or if channel 1 or 2 input is used) or direct to the user device (if 16K words of memory and channel 1 and 2 inputs are not used). The data ready signal generated within the memory control unit is applied to the user device to gate the data to the B Register (G0BP23-00) if the AU is the user or to the data register of the Bulk Memory Controller, channel 1 or 2 user. Bit 24 of the memory data register is applied to the parity check circuitry of the memory control unit.

The data gated to the user is applied back through the multiplexer (or direct from the B Register in systems not requiring the multiplexer) to the parity check logic. In this manner, transmission parity is also checked. A detailed discussion of parity checking is provided later in this section.

As mentioned above, the data in the memory data register is used to control the bit drive lines during the write portion of the memory cycle. If a "one" is to be stored into a core cell, full select current is passed through the cell in a direction opposite to that of the read portion of the cycle. If the core is to remain in the "zero" state, no current flows in the bit drive lines, only one-half select current flows through the core from the word drive lines, and the core remains in the "zero" state.

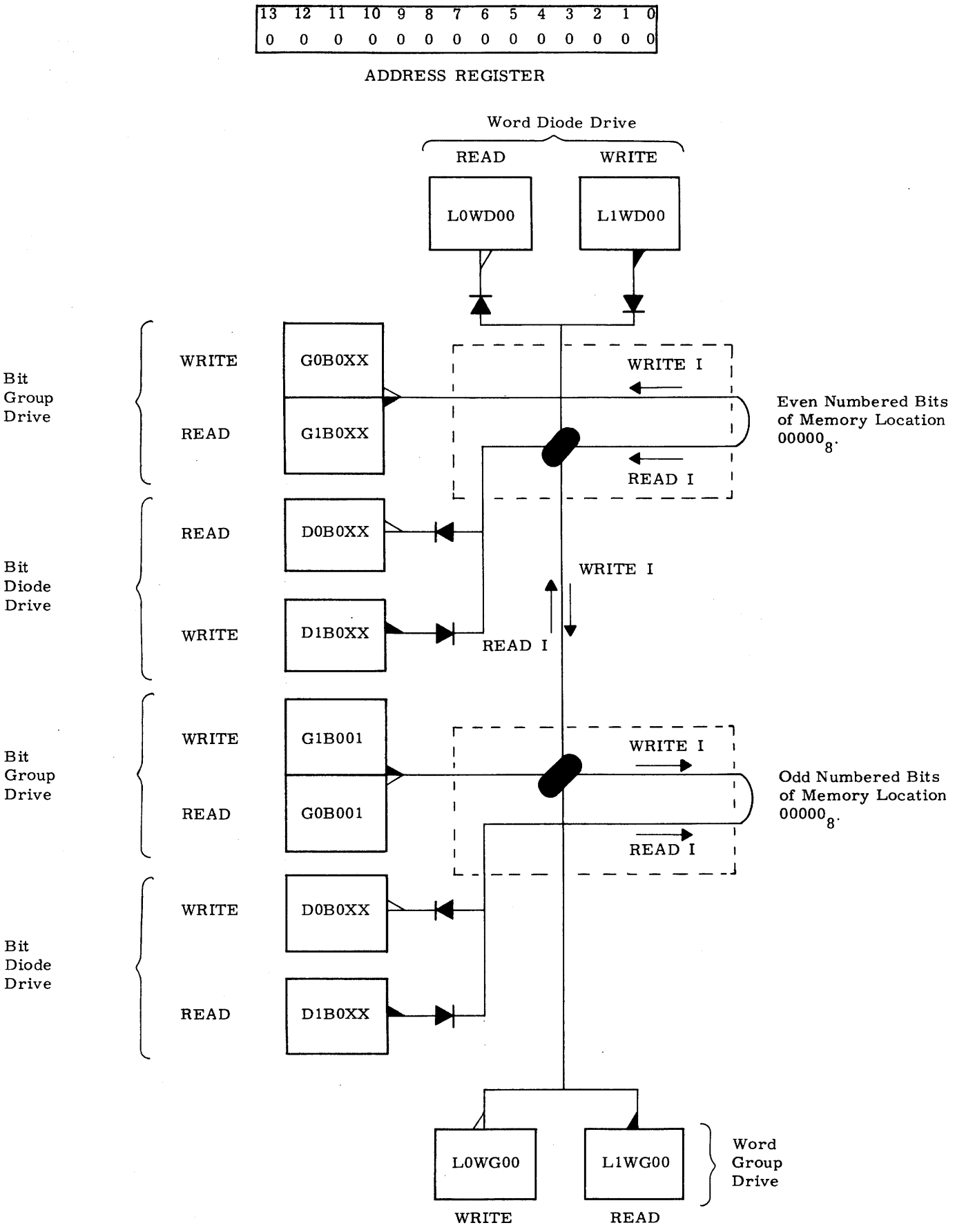


Fig. THEORY. 10 Address Selection Block Diagram

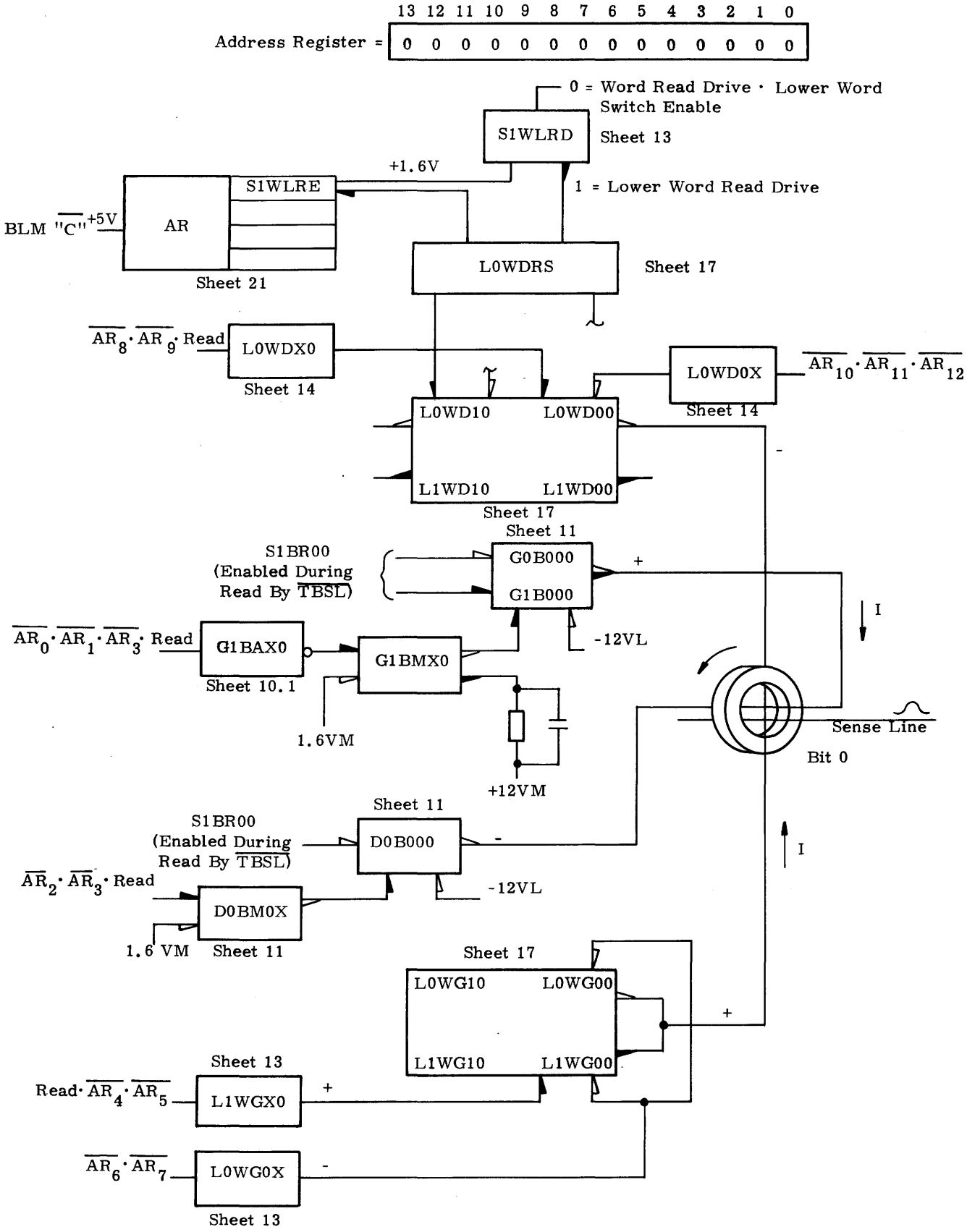


Fig. THEORY. 11 Read Drive Circuit

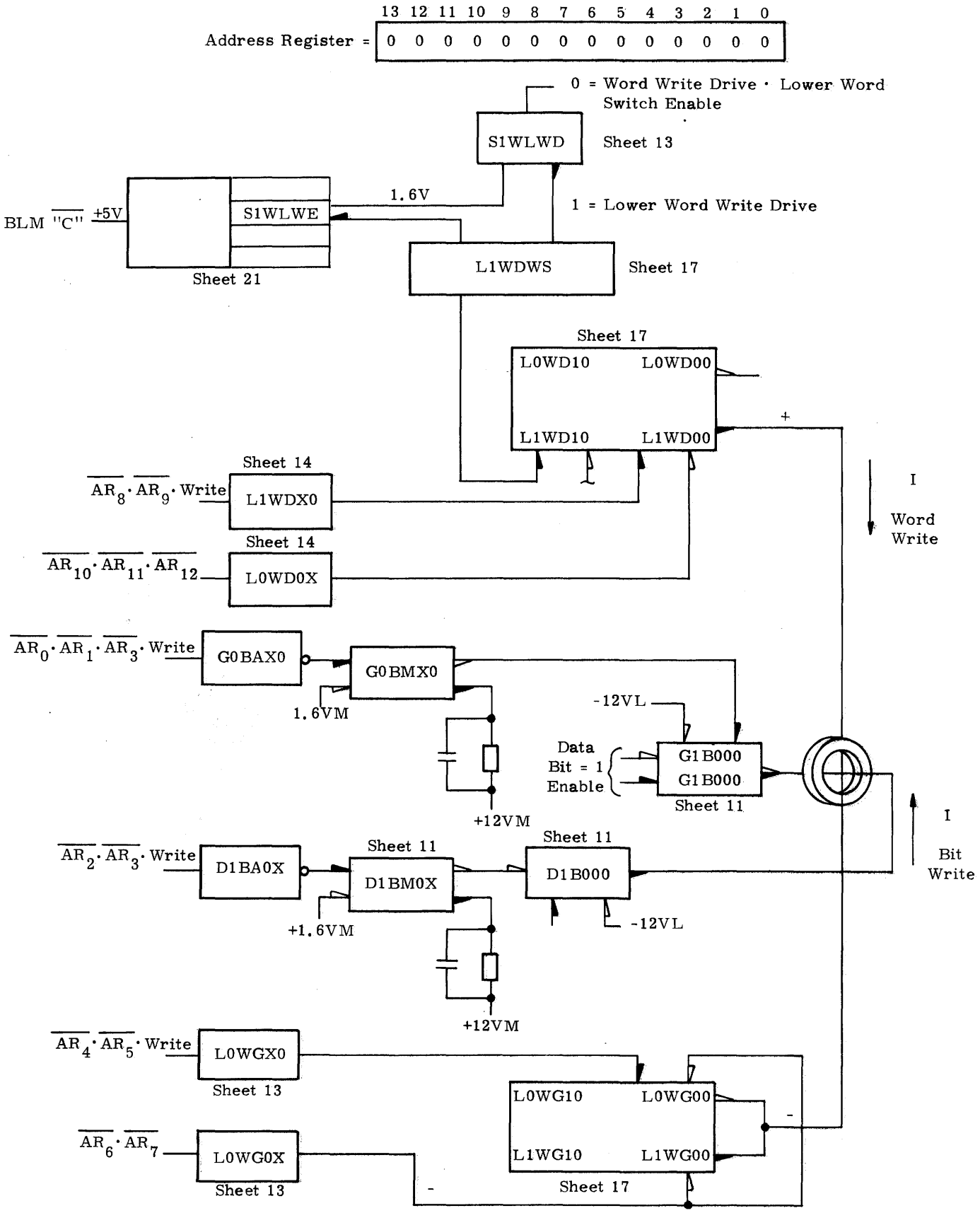


Fig. THEORY. 12 Write Drive Circuit

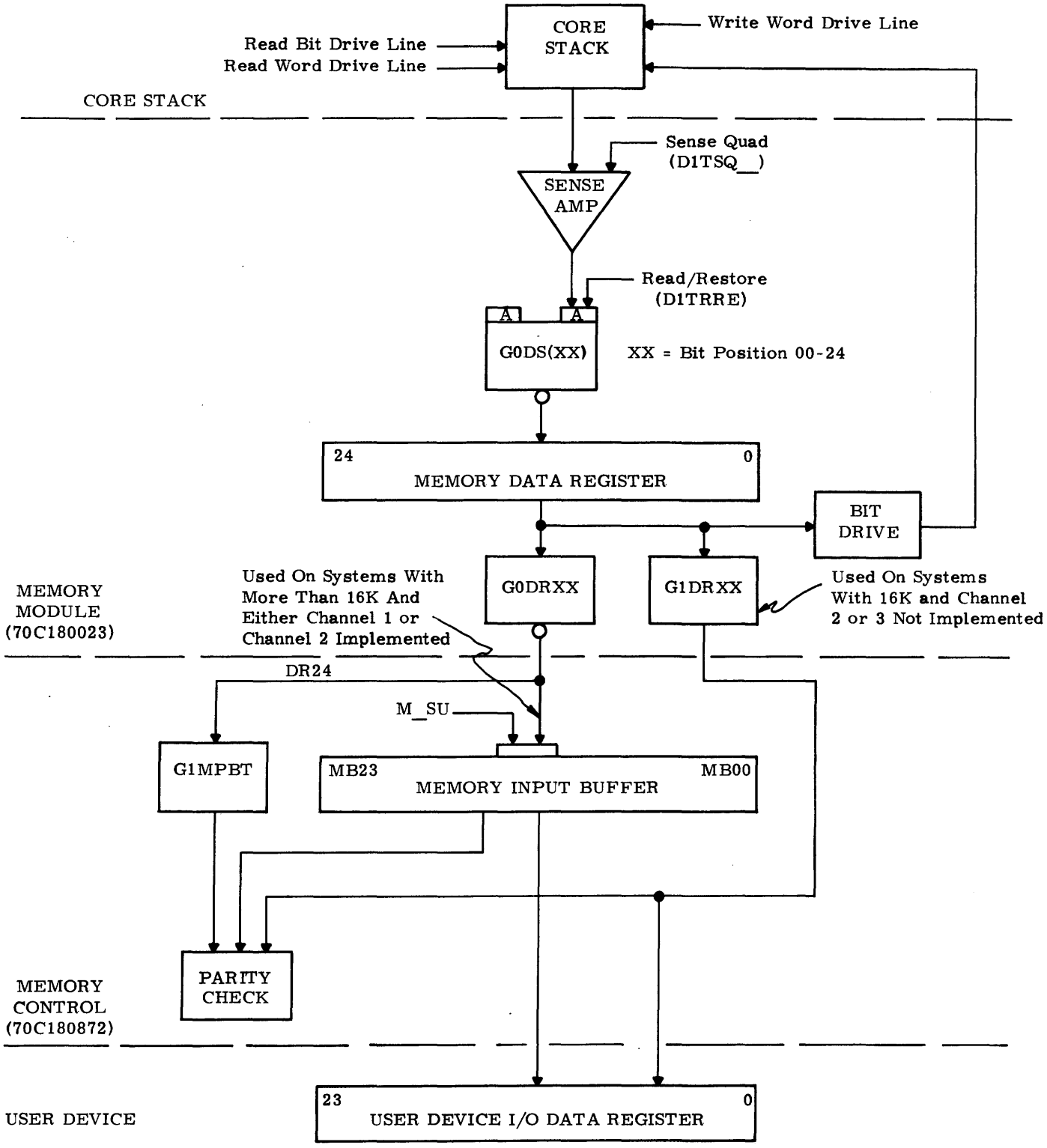


Fig. THEORY. 13 Read/Restore Block Diagram

CLEAR/WRITE OPERATION

The clear/write mode of memory operation stores new data in a memory location. The clear/write mode is enabled when the user device granted access to memory specifies the write or store mode of operation.

The clear/write mode of memory operation is very similar to that of the read/restore mode of operation. The basic differences are that the parity generate logic is enabled in the clear/write mode and the data sensed from core during the read portion of the memory cycle is not strobed to the data register in the clear/write mode. Instead, the data to be stored from the user device is strobed into the data register.

The user device specifies the clear/write mode by applying a read/write or store signal to the memory control unit. This signal enables G0MWRQ and inhibits G0MRRQ. G0MRRQ, when inhibited, enables the parity generation logic. G0MWRQ, when enabled, enables the write cycle enable gate (D1TWCE) in the memory module. This signal is used to gate the data from the user device to the memory data register for storage in the addressed memory cells during the write portion of the cycle.

Fig. THEORY. 14 illustrates the data flow for the clear/write mode of memory operation. During the read (clear) portion of the memory cycle, the bit and word drive lines are enabled according to the memory address from the device. All cores in the "one" state are flipped to the "zero" state, however, the resultant signals induced in the sense windings are not gated to the memory data register. Therefore, the read (clear) portion of the memory cycle is used to place all addressed cores in the "zero" state.

The data to be stored (written) in the addressed core location is gated through the multiplexer of the memory control unit to the data register and to the parity generation logic. The parity generation logic then generates a "one", if required for odd parity, and applies this "one" to bit position 24 of the data register. The parity and data bits are then strobed in the memory data register. The contents of the memory data register are then used to control the bit drive lines during the write portion of the memory cycle. A one in the data register will enable current through the corresponding bit drive line. This current through the bit drive line, in conjunction with the current through the word drive line, will flip the core to the "one" state. A "zero" in the data register will inhibit current flow in the bit drive line. The half-select current through the word drive line will not be sufficient to flip the core and it will remain in the "zero" state. In this manner the cores of the addressed memory location will be placed in the configuration of the data bits in the memory data register.

PARITY GENERATION

Parity generation, in conjunction with parity checking, provides a means of detecting if a stored data word has gained or lost a bit. The parity generation logic ensures that an odd number of "one" bits are stored in memory during any clear/write operation. The parity checking logic ensures that an odd number of "one" bits are read back for each data word during read/restore operation. If an odd number of "one" bits are not read back, a parity error signal is provided to light the Parity Error and Alarm indicators on the computer console, enable the running program to detect the parity error (JNP), and to optional inhibit further access to memory by the Arithmetic Unit (Stop on Parity switch). If a parity error occurs when the core temperature is out of limits, memory operation is halted (G0MSTF).

As shown in Fig. THEORY. 14, data bits from the multiplexer are applied to the parity generate logic of the memory control unit during the clear/write mode of operation. The parity generate logic then enables a "one" to bit 24 of the memory data register if an even number of "one" bits are contained by the 24 data bits.

Fig. THEORY. 15 illustrates, in block diagram form, the logic used to generate the parity bit. The logic associated with parity generation is contained on sheets 17 through 21 of the memory control logic 70C180872).

The data bits to be stored in memory are gated from the user device through the multiplexer by D1MAE1, 2, 3 or 4 when memory access has been granted. These data bits are applied to the memory data register as well as the inputs of the parity generation logic (G0POG1 through G0POG8).

G0POG1 through G0POG8 each sample three data bits and if an odd number of "ones" are contained by these three data bits, a "zero" output signal is provided. If an even number of "one" data bits are contained by the three inputs, a "one" output is provided. The outputs of G0POG1 through G0POG8 are applied to G0POS1, G1POS2, and G0POS3. The output from these gates are applied to G1POFS. The output from G1POFS is then a "one" if an odd number of data bits was detected from the multiplexer or a "zero" if an even number of "one" data bits was detected from the multiplexer.

The output of G1POFS is inverted in G0POFS providing a "one" output to bit 24 of the memory data register if an even number of "one" bits is contained by the data to be stored in memory. This parity bit

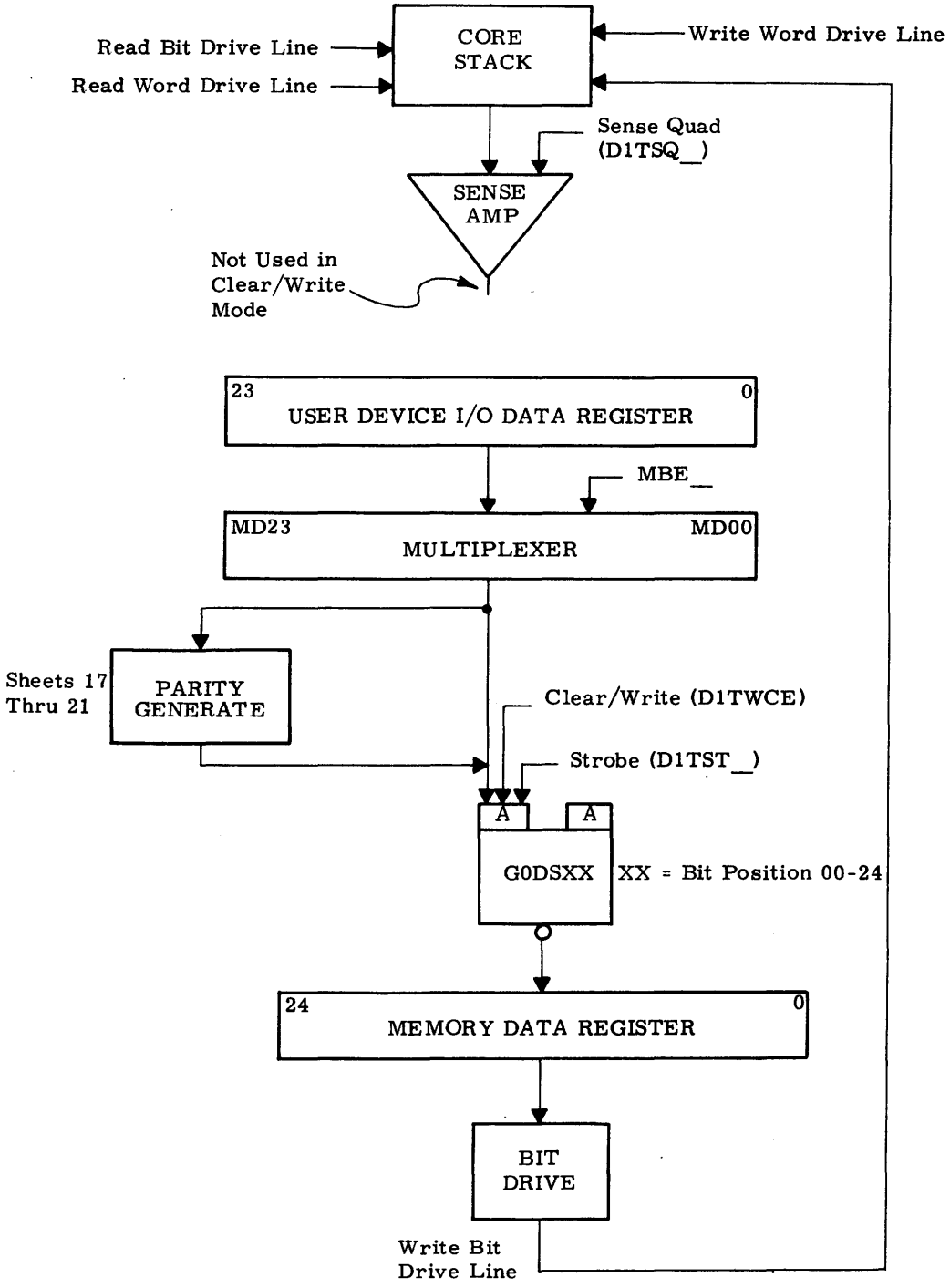


Fig. THEORY. 14 Clear/Write Block Diagram

is gated to the memory data register by the strobe at the same time the remaining data bits from the multiplexer are gated to the memory data register.

PARITY CHECKING

During a read/restore operation, the data read from the addressed core cell is checked to determine if it contains an odd number of "one" data bits. Since the parity generation logic ensured that an odd number of "one" bits was stored in the memory location, if an even number of "ones" is read back, a bit has been gained or lost and an error condition exists. When an error condition is detected, the Parity Error flip-flop (F1PPER) is set and the data word is restored in the memory location in the same form that it was read.

Setting the Parity Error flip-flop performs the following functions:

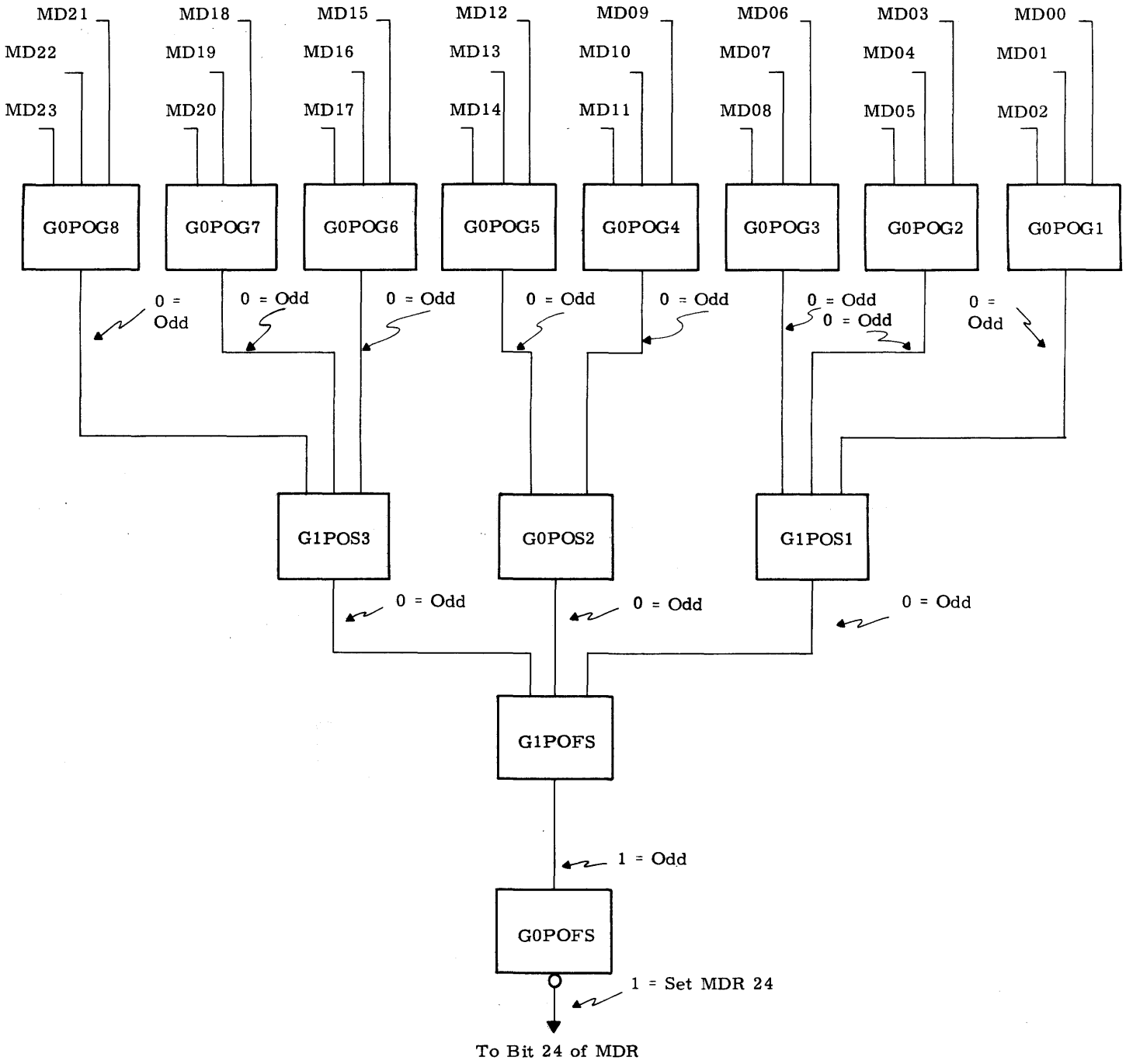


Fig. THEORY. 15 Parity Generate

1) the CORE PRTY and ALARM indicators on the Arithmetic Unit console are lighted;

2) the system program may monitor the error by executing a JNP command;

3) further accesses to memory by the AU are inhibited if the STOP ON PARITY switch is in the stop position;

4) if the system alarm option is included, an external alarm is provided; and

5) if the parity error occurred during a data transfer to a user device other than the AU, the MPLX CH1, 2, or 3 indicator on the AU console is lighted, and an indication is provided to the user (F1MEPO).

The Parity Error flip-flop may be cleared by a JNP command, the Clear Alarm switch on the Computer Console, or by Initializing the system. The only method of clearing F1PPED is by pressing the Clear Alarm switch or pressing Initialize. Therefore, if a parity error occurs, the indicators on the Computer Console will remain lighted until the Clear Alarm or Initialize switch is pressed.

As shown in Fig. THEORY. 13, the data obtained from memory during a read/restore operation is gated to the user device and to the parity check logic.

Fig. THEORY. 16 illustrates the logic associated with parity checking. This logic compares the status of the parity bit read back from memory with the status of G1COFS and G0COFS. G1COFS and G0COFS are enabled when the 24 bit data word contains an odd number of "one" bits. The parity check logic operates in the same manner as the parity generate logic previously described. G1MPBT enables the parity bit read from the addressed memory cell depending upon which stack was addressed. This gate is enabled (G0MIAD) when $>$ 24K is addressed in a 24K system forcing the parity bit, since all zeros will be read from memory as listed in the Introduction. Enabling G1MPBT in this manner simulates a parity bit, preventing a parity error indication.

The status of the parity bit is compared with the status of G1COFS and G0COES in G0PPE1 and G0PPE2 at memory control time 14_8 when in the read/restore mode. G0PPE1 is enabled if an odd number of data bits are contained by the word read from memory and the parity bit is a one. G0PPE2 is enabled if an even number of data bits are contained by the word read from memory and the parity bit is a zero.

Enabling either G0PPE1 or G0PPE2 will cause the Parity Error flip-flop (F1PPER) to set at the next clock pulse. Setting F1PPER enables the parity error functions previously described.

TRANSMISSION PARITY ERROR CHECKING

During a write into core memory cycle the data being transferred from a user device other than the AU to the core memory is checked for parity. The memory receives the 24 data bits and uses the data to generate a parity bit. This parity bit is compared with a parity bit generated by the user. If the two parity bits are not the same, a signal is applied to the user signifying a transmission parity error. The MPLX CH 1, 2, or 3 indicator on the programming and maintenance console is lighted by this error signal. The parity bit generated within the core memory is the bit stored in core.

Fig. THEORY. 17 illustrates the transmission parity error checking logic in the Memory Controller/Multiplexer. The state of the parity bit generated by the user is compared with the status of G1COFS in G0MTP1, 2, 3. As previously described in the parity checking description, G1COFS is enabled providing a "one" output when the 24 data bits contain an odd number of "one" bits. G0MTP1, 2, 3 is enabled, signifying a transmission parity error, when an odd number of "one" bits is contained by the data word and the parity bit is a "one" or when an even number of "one" bits are contained by the data word and the parity bit form the user is a "zero". The output of G0MPT1 is applied to the user. Refer to the theory of operation of the user for the effect of this parity error signal.

CORE HEATER

The core stack is an enclosed unit maintained at approximately 55°C by heating elements within the stack. Over and under thermorelays are provided within the stack to indicate when the temperature is within a range of 45°C to 60°C. If the temperature of the stack is outside these limits, the CORE TEMP and ALARM indicators on the computer console are lighted and if a parity error is detected when the temperature is out of limits, memory operation is inhibited. Two copper-constantan thermocouples are provided within each stack for monitoring of the nominal stack temperature. If the stack exceeds approximately 75°C, operating power of 28V AC is removed from the heaters.

The core heaters are designed to maintain the internal temperature of the stack at approximately 55°C over an environmental temperature range of 0° to 55°C. Environmental temperature changes of up to 15°C per hour can be compensated. The 55°C temperature is reached within 1.5 hours from an ambient temperature of 25°C. The core heaters are positioned to maintain all cores at a uniform temperature ±3°C.

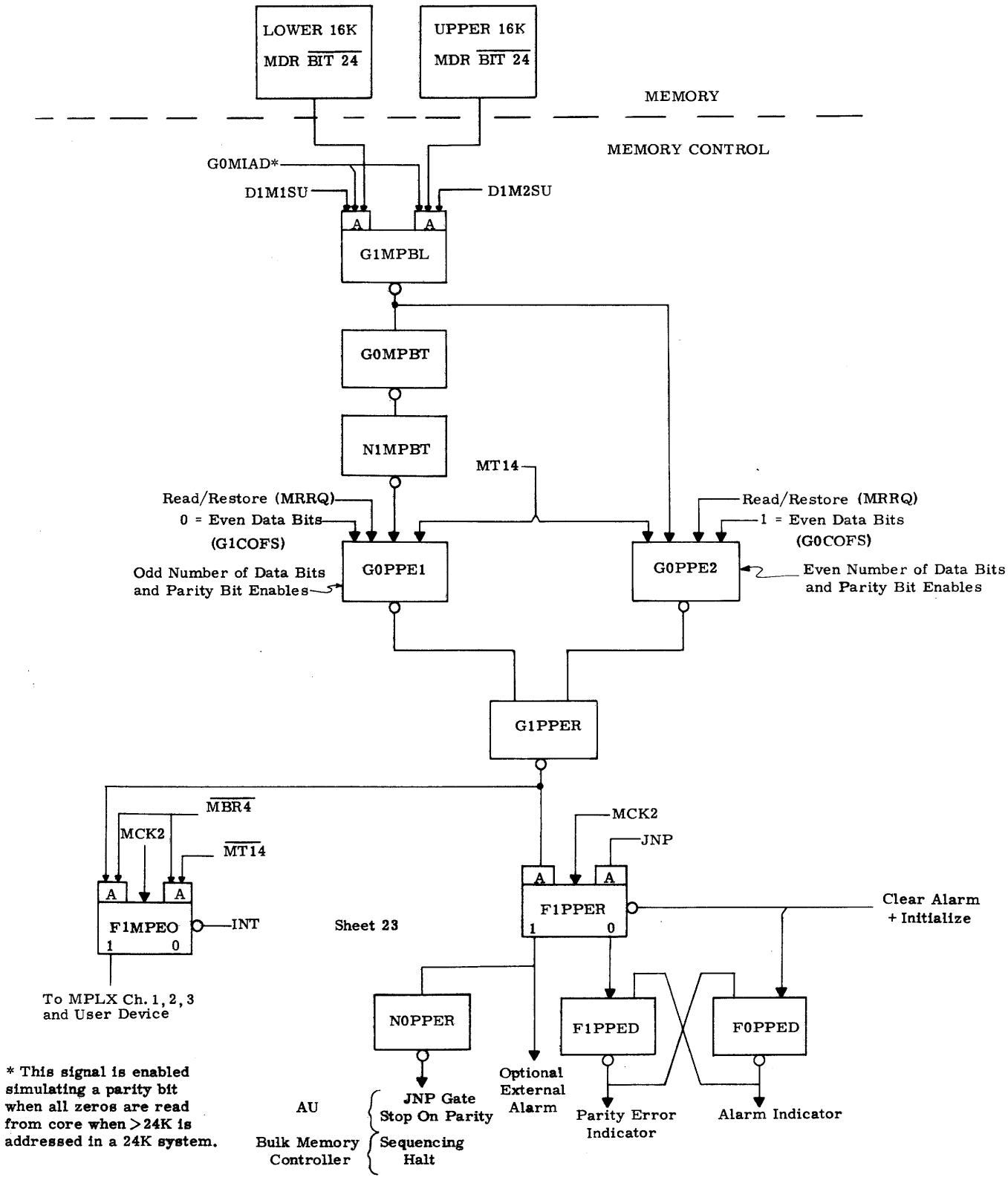


Fig. THEORY. 16 Parity Check

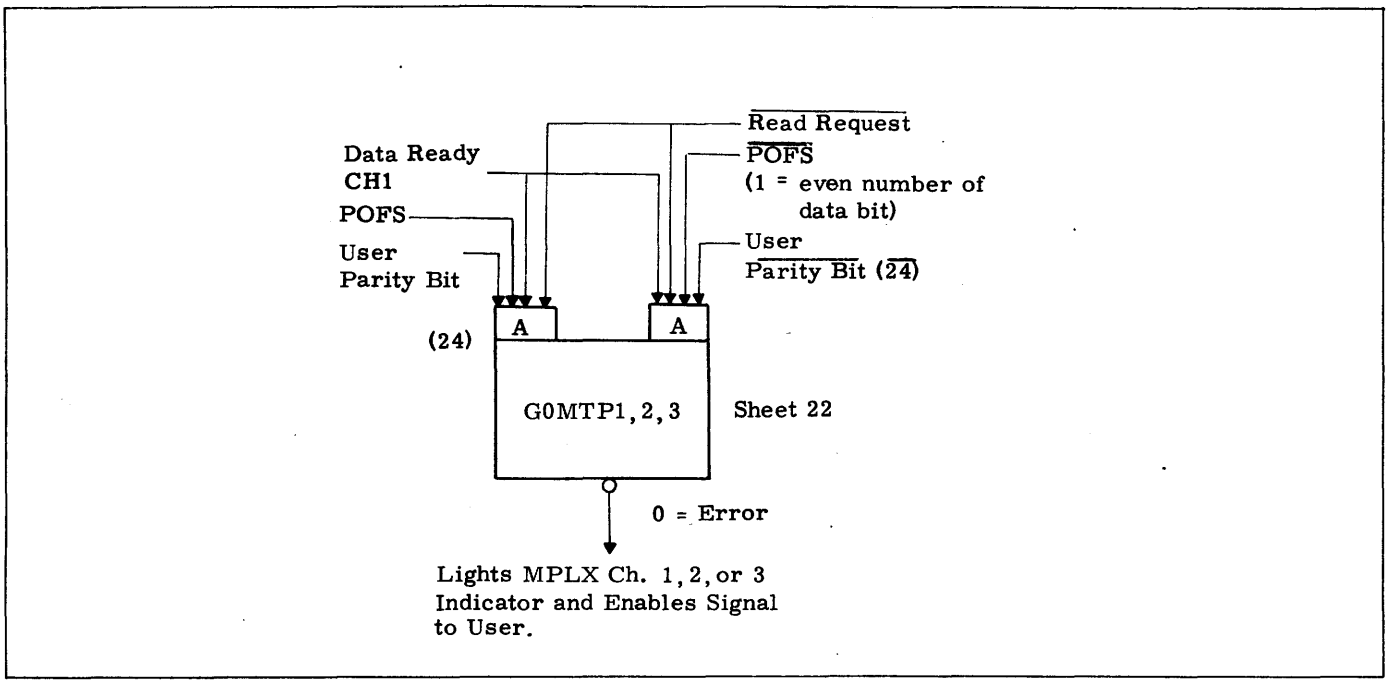


Fig. THEORY. 17 Transmission Parity Checking

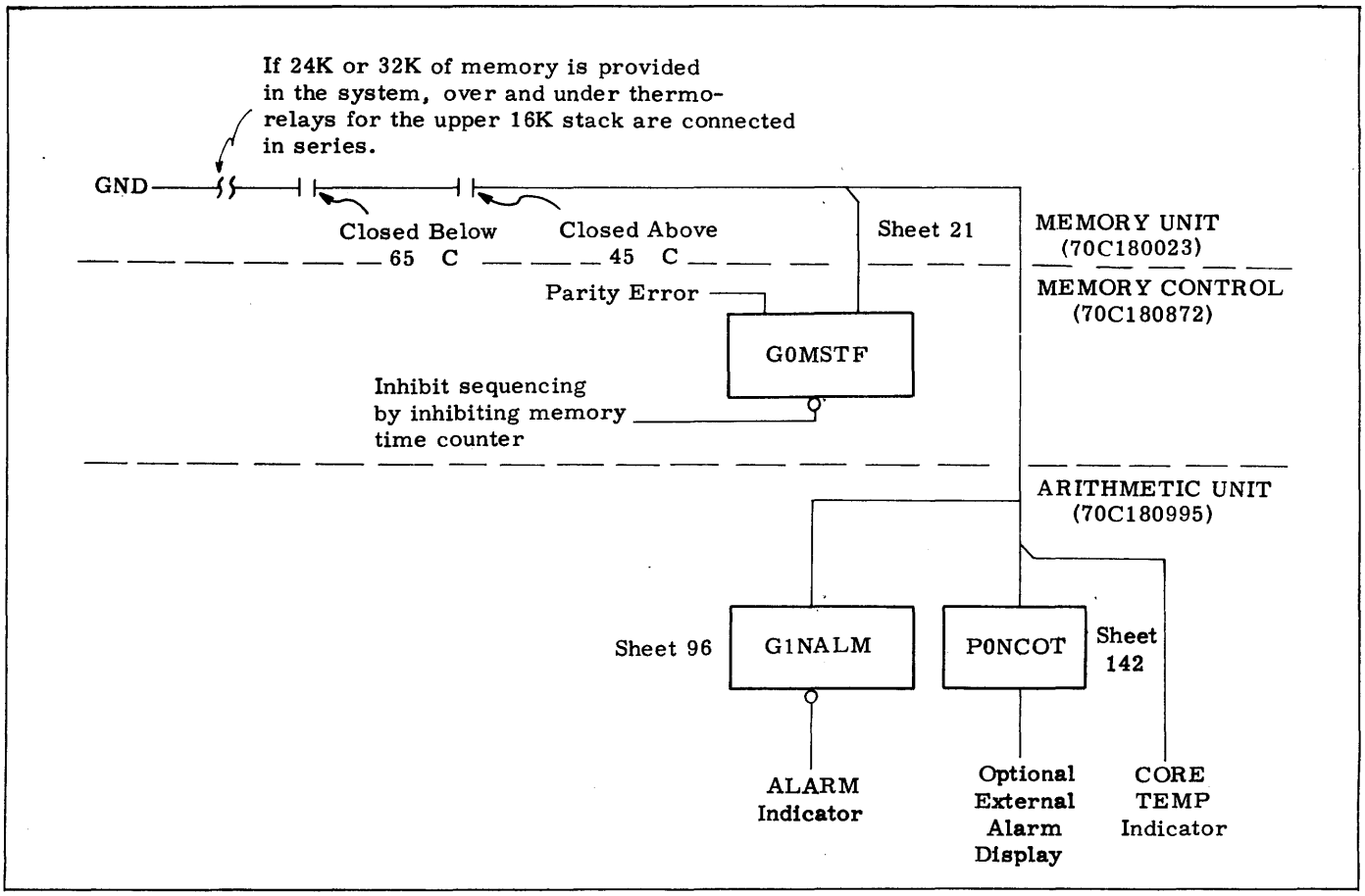


Fig. THEORY. 18 Core Heater Alarm Circuitry

The core heater control and associated circuitry is shown on sheet 21 of the memory logic (70C180023). Fig. THEORY. 18 illustrates the core heater alarm circuitry associated with over and under temperature conditions.

The heater controller (XHCB1) is used to maintain the core stack temperature at 55 ±2°C. The heater controller senses the temperature from a thermistor located internal to the stack. The change in the thermistor resistance is detected by a transistor in a bridge circuit consisting of two Zener diodes, a series of resistors for adjustment, and the sensing thermistor.

The heater controller is adjusted by shorting out the proper resistors so that it will not supply heating current when the sensing thermistor is 4100 ±2% ohms, but will supply current when the thermistor is 4139 ±2% ohms.

> NOTE
>
> A variable 2K ohm potentiometer for each of the two heaters is contained with the core stack. THESE POTENTIOMETERS SHOULD NEVER BE ADJUSTED BY FIELD PERSONNEL WHEN XHCB1 CONTROLLER BOARDS ARE USED.

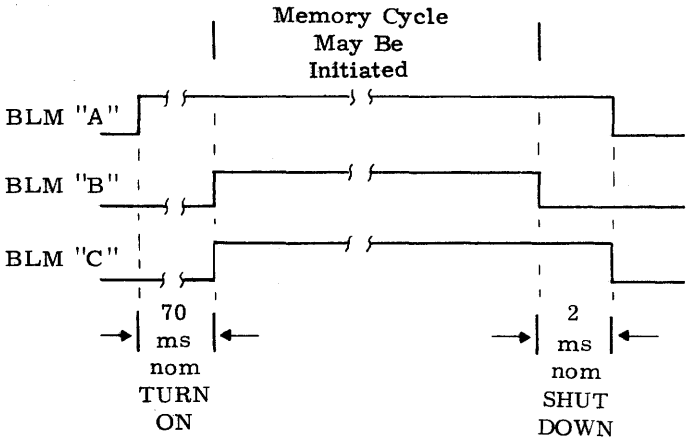
A detailed test procedure for the XHCB1 board is contained in GE Drawing 68A977764.

The heater controller checks every 8 milliseconds to determine if more heater current is required. A clocking circuit consisting of a transistor (Q1) that is controlled by a rectified 60 Hz sine wave provides this check. The output of the clock transistor is collector "ORed" with the temperature sensing circuit. The "ORed" output of the clocking and temperature sensing circuit is amplified to control a Triac which is used as a silicon controlled rectifier to control the heater current.

POWER FAILURE OR SHUTDOWN

Removal of operating power, either during normal shutdown or by a power failure, is detected by the Buss Level Monitor circuitry contained within the 4780 Power Components (GE Drawing 70C180914). The Buss Level Monitor, described in the Power Distribution section of this book set, provides three control signals to prevent destruction of data stored in core memory during power loss.

The timing of these three signals, BLM "A", BLM "B", and BLM "C" is illustrated in Fig. THEORY. 19. All of these signals must be at the +5 volt (approx.) level for a memory cycle to be initiated. Once a cycle is initiated, the cycle may be completed. BLM "C" does not go to zero volts until approximately 2 milliseconds after a shutdown has been detected so a previously initiated cycle may be completed.



NOTES:

1. BLM "A" at zero initializes memory and inhibits initiating a memory cycle.
2. BLM "B" at zero inhibits initiating a memory cycle.
3. BLM "C" at zero inhibits word read and write drive current.

Fig. THEORY. 19 Buss Level Monitor Control

The BLM "A" signal when at zero volts, clears the initiate memory cycle flip-flop (F1TIMC), clears the Memory Busy flip-flop (F1TMBU), and sets the initialize flip-flop (F1TINT) as shown on sheet 7 of 70C180023.

The BLM "B" signal when at zero volts, inhibits any initiate signal generated from a request in G0TIMS as shown on sheet 7 of 70C180023.

The BLM "C" signal when at zero volts, inhibits the word read and write drive currents by disabling D1UWEN, D1UREN, D1LWEN, and D1LREN (sheet 21 of 68C972952).

JUMPER PIN OPTIONS

Jumper pins on the RMTC1 board (panel B, slot 18) as shown in Fig. THEORY. 20, provide the following control:

1. Controls memory wrap around and parity error detection according to the size of memory (16K, 24K, or 32K) contained in the system. Logic sheet 22.1 of 70C180872 specifies the pin locations for all memory sizes.
2. Provides a disable of memory requests from users other than the AU. The jumper pin is shown on sheet 7 of 70C180872.
3. Permits the stacks in a 32K system to be interchanged with reference to the associated addresses; e.g., the stack in panel U normally associated with addresses 16K to 32K may be switched to addresses 0 to 16K by

changing jumper pins. The stack in panel V normally associated with addresses 0 to 16K would then operate with addresses of 16 to 32K. This capability is very useful for troubleshooting. Logic sheet 22.1 specifies the jumper pin locations for stack selection.

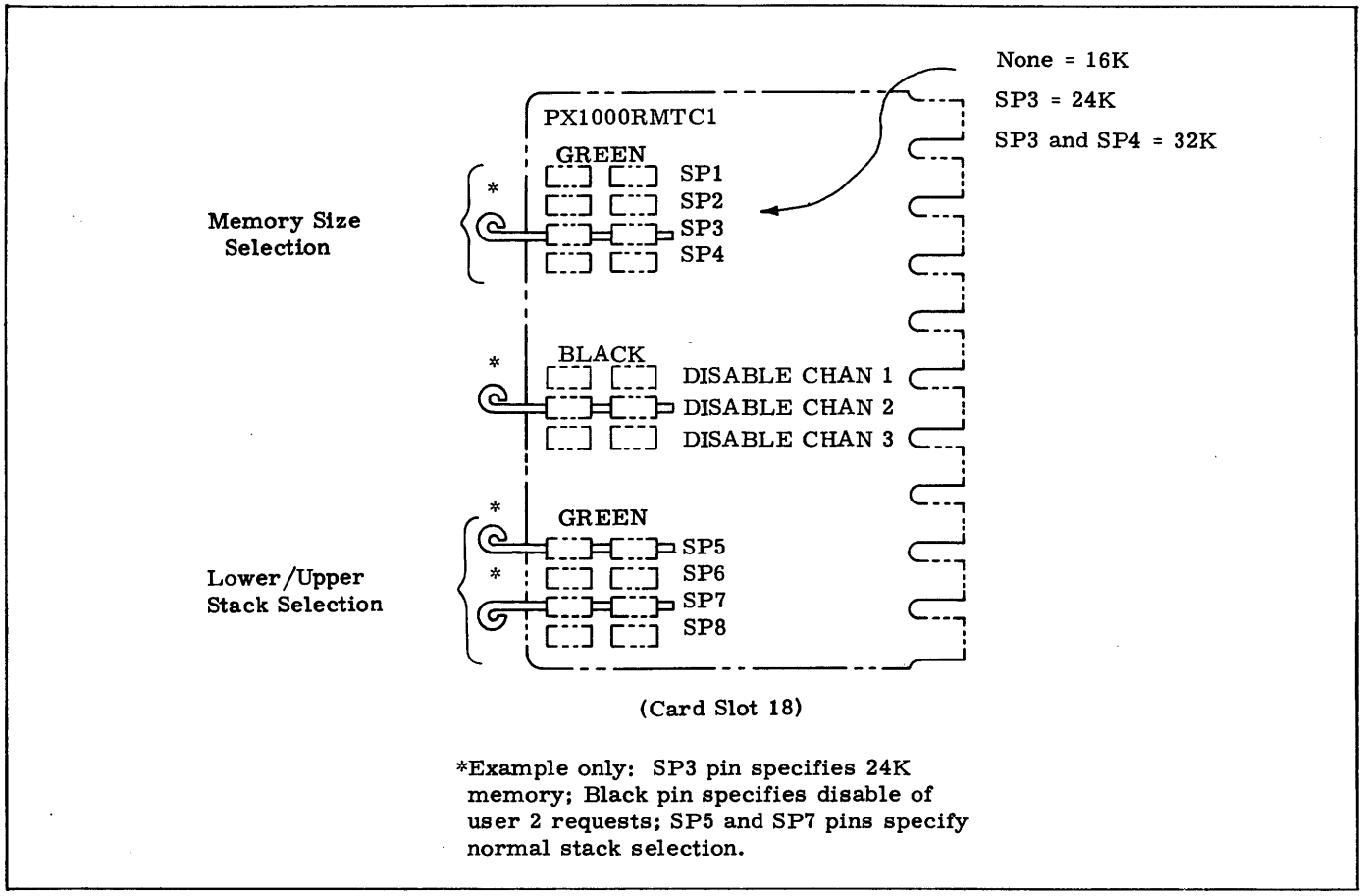


Fig. THEORY.20 Jumper Pin Locations

READER COMMENTS

The General Electric Company solicits your comments on publications covering Process Computer equipment. Please explain any "no" responses in the COMMENTS section. Your comments and suggestions become the property of the General Electric Company.

- Name of Manual: _____
- What is your computer application: _____

  _____
- How is this publication used:

  Familiarization ☐ Reference ☐

  Training ☐ Maintenance ☐

  Other (Explain) _____

	YES	NO
• Does this publication meet your requirements	☐	☐
• Is the material:		
1) Presented in clear text	☐	☐
2) Conveniently organized	☐	☐
3) Adequately detailed	☐	☐
4) Adequately illustrated	☐	☐
5) Presented at appropriate technical level	☐	☐

- Please provide specific text references (page number, line, etc.) with your comments.

NAME _____ DATE _____

TITLE _____

COMPANY NAME _____

AND ADDRESS _____

COMMENTS:

No postage necessary if mailed in the U. S. A.

Staple

Communications concerning Technical Publications should be directed to:

Manager, Technical Publications
Utility and Process Automation Systems Operation
2255 West Desert Cove Road
Phoenix, Arizona 85029

Fold Fold

FIRST CLASS
Permit No. 4091
Phoenix, Arizona

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

GENERAL ELECTRIC COMPANY
UTILITY and PROCESS AUTOMATION
SYSTEMS OPERATION
2255 West Desert Cove Road
Phoenix, Arizona 85029

Attention: Technical Publications

Fold Fold

Cut Along Line

Additional Comments:

4022D ARITHMETIC UNIT
TABLE OF CONTENTS

INTRODUCTION

This section describes the functional operation of the GE-PAC* 4022D Arithmetic Unit which is the computational and control center for the GE-PAC 4010B Process Computer System.

The 4022D Arithmetic Unit performs calculations, a wide range of logical operations, and sequences and distributes data throughout the computer system. It supplies and receives information to/from the Core Memory, Automatic Program Interrupt Module, and Input/Output Peripheral Devices as well as the Process I/O and Bulk Memory Subsystems.

The Arithmetic Unit addresses sequential programmed commands stored in Core Memory. Each command addressed is transferred from memory to the Arithmetic Unit where the command is executed. Since execution of the command may require the transfer of data to or from one of the other modules of the system, the Arithmetic Unit becomes the communications hub of the system.

The arithmetic operations performed by the Arithmetic Unit include Add, Subtract, Multiply, and Divide. Addition and Subtraction are performed in 3.2 microseconds, while Multiply requires between 8.9 and 12.1 microseconds and Divide requires 13.7 microseconds. Add and Subtract use single length 24-bit numbers. Multiply operates on two 24-bit numbers to produce a double length (46 bits plus sign) product. Divide operates on a double length dividend and a single length divisor to produce a single length (23 bits plus sign) quotient and a single length remainder.

SPECIFICATIONS

- Electrical Requirements

  Voltage: Single Phase: 115/230
  Three Phase: 208Y

  Frequency: 60 Hz/50 Hz

  Central Processor Power: 5 KW

- Environment

  Class A - General Industrial
  (Refer to 68A974933 for definitions)

- Circuit Type

  Monolithic Integrated.

- Man/Machine Communication

  Programming and Maintenance Console.

- Basic Clock Frequency

  10 MegaHertz

  Pulse Width - 15 to 35 nanoseconds

- Addressable Core Memory

  Directly 16,384, relative ± 8K and indexing to 32,768 locations.

- Operation

  Parallel and serial internal; parallel external.

- Arithmetic

  Digital, binary, fixed point, and 2's complement; floating point by subroutine.

- Word Size

  24-bits (23 through 00); bit 23 is sign and most significant bit.

- Index Words

  Total of 7 words; 5 are unrestricted and 2 (1 and 2) are restricted by Quasi and subroutine linkage.

OPTIONS

The GE-PAC 4022D Arithmetic Unit is a part of the 4DP4800BS01 Basic Central Systems Unit (CSU). In addition to describing the operation of the 4022D Arithmetic Unit, this section also describes the operation of the following Central Systems Unit options:

4DP4800AS061 - Alternate Source Timer
4DP4800AS071 - System Alarms
4DP4800AS081 - Quadritect Memory Protection/ API Watchdog
4DP4800AS091 - Adjustable Pulse Generator

The Central Systems Unit drawing, 4DP4903BSID, defines the options of the CSU for a particular system.

REFERENCE DOCUMENTS

Logic:

70C180955 - 4022D Arithmetic Unit,
4DP4800AS061 - Alternate Source Timer
4DP4800AS071 - System Alarms
4DP4800AS081 - Quadritect Memory Protection
4DP4800AS091 - Adjustable Pulse Generator

Wiring:

70A105429 - 60" Cabinet Wiring
70A105453 - 30" Cabinet Wiring
70A120521 - A-B Panel
70A120523 - A-B Panel
70A120311 - Console

Interface Module Logic:

70C180899 - I/O Expander
70C180954 - Automatic Program Interrupt
70C180872 - 4016B Core Memory Multiplexer/ Control
70C180909 - I/O Buffer
70C180914 - Power Components
70C180908 - Bulk Memory Controller
70C180023
or - Core Memory
68C972952

Maintenance:

GE-PAC 4010 Computer Maintenance Manual

*Registered Trademark of General Electric Company

LOGIC SYMBOLS AND NOMENCLATURE

The following paragraphs provide a very brief discussion of the logic symbols and nomenclature used in the GE-PAC 4010 logic drawings.

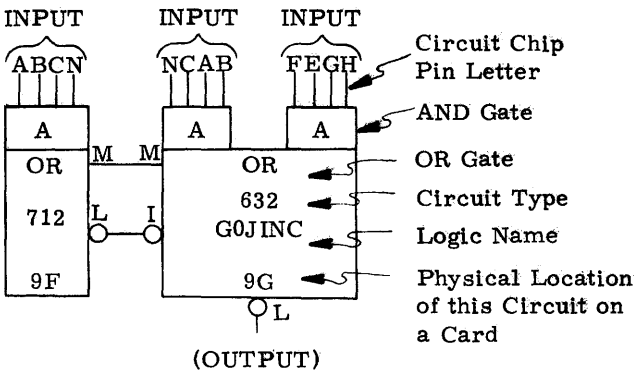
Fig. INT. 1 illustrates the most commonly used logic symbols and the logic equation associated with each symbol. Each symbol designates the integrated circuit chip pin numbers, the AND (A) and OR outputs, an identification number of the circuit chip type, the logic name of the logic element, and the physical location of the logic element on the circuit card. Each of these designations will be discussed in the following paragraphs.

Since there are many different types of integrated circuit chips, each logic symbol identifies the type. This circuit type number is derived from the drawing number assigned to the individual circuit type. For instance, the drawing number for a 632 circuit is 68A8363P2; for a 752 circuit, 68A8375P2; for a 642, 68A8364P2; etc. Since the 68A83__P_ part of the drawing number is the same for all circuits, it is deleted from the circuit type number contained in the logic symbol.

The logic name consists of eight digits as illustrated below:

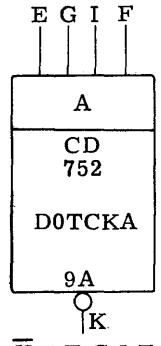
1 2 3 4 5 6 7 8

- 6 7 8 — Mnemonic signal name
- 5 — Logic area
- 4 — Polarity of signal when true (1 or 0)
- 3 — Element type:
  - D - Driver
  - F - Flip-Flop
  - G - Gate
  - J - Jack
  - M - Line Receiver
  - N - Inverter
  - P - Plug
  - R - Resistor
  - S - Switch
  - V - Voltage
  - X - Line Driver
  - Y - Special Purpose
- 1 2 — Assembly/Functional Module:
  - AI - Automatic Program Interrupt
  - AU - Arithmetic Unit
  - BC - Bulk Memory Controller
  - EX - Input/Output Expander
  - IO - Input/Output Buffer
  - MU - Core Memory Unit
  - MX - Core Memory Multiplexer
  - SQ - Sequencer



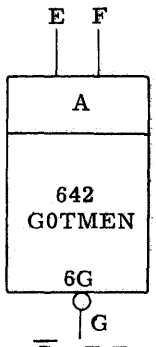
$\overline{L} = ABCN + NCAB + FEGH$

NAND Logic Symbol



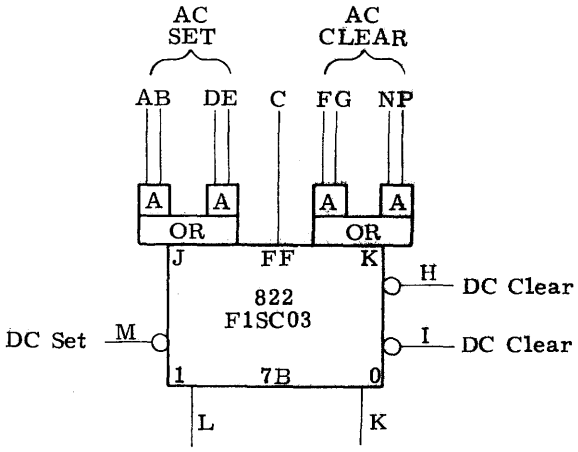
$\overline{K} = E \cdot G \cdot I \cdot F$

Clock Driver



$\overline{G} = E \cdot F$

Two-Input NAND/NOR Gate



$L = [(A\,B + D\,E)\,C] + \overline{M}$

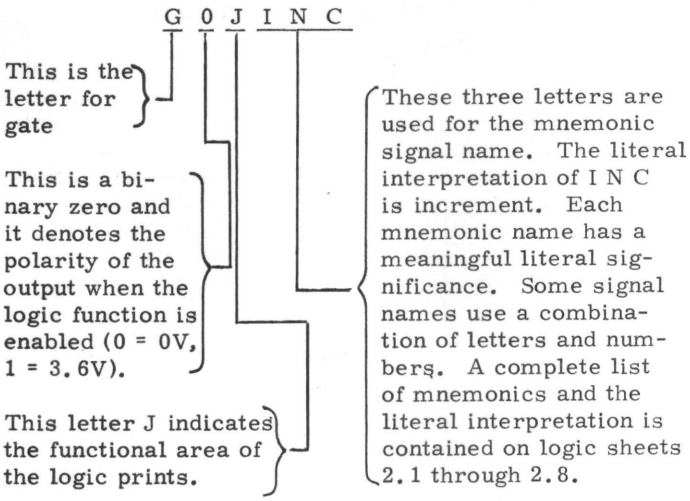
$\overline{L} = [(F\,G + N\,P)\,C] + \overline{H} + \overline{I}$

Flip-Flop (AND Inputs)

1 = +3.6V = True
0 = 0V = False

Fig. INT. 1 Logic Symbols

The logic name for the NAND symbol of Fig. INT. 1 is described for clarity.



The physical location of this circuit on the card is designated by numbered and lettered co-ordinates of the circuit position of the card. Fig. INT. 2 illustrates a circuit card and the co-ordinate designations.

The circuit chip pin letters specify the connections of the chip. Fig. INT. 2 illustrates a chip and the pin designation locations.

Fig. INT. 3 illustrates the wiring nomenclature and circuit card locations provided in the logic drawings.

The circuit card pin number defines the card connector and the pin number of that connector. As shown in Fig. INT. 2 the connector may be defined by the lettered co-ordinate. Since each connector contains 14 pins, the remaining digits specify which pin of the connector.

The card location and size is also defined on the logic drawings as shown in Fig. INT. 3. The first letter specifies the panel on which the card is located, the next two numbers specify the card slot of the panel that the card is located, and the last two letters define the card size.

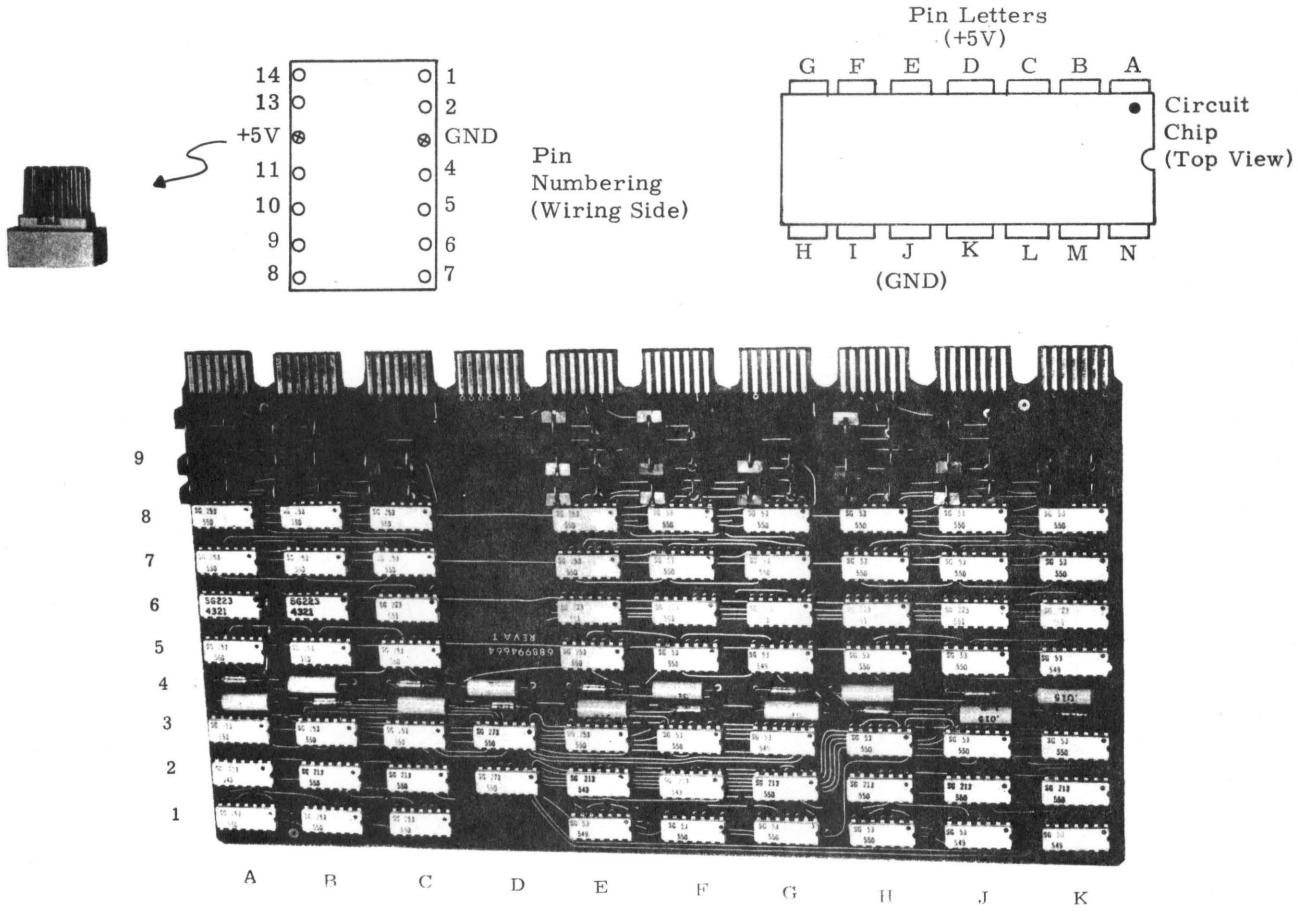


Fig. INT. 2 Circuit Card

The physical location of Panels A and B that contain the Arithmetic Unit is shown in Fig. INT. 4. The first letter then designates in which panel the card is located. The next two numbered digits specify the card slot within the panel, since 32 card slots may be provided in a panel. The last two lettered digits indicate the size of the card by indicating the connectors provided. Therefore, AK is a 10-inch card utilizing connectors A through K. The letters AE would indicate a 5-inch card using connectors A through E, and the letters FK would indicate a 5-inch card using connectors F through K.

Since wire connections must be made between the individual circuit cards, a wire list is used to specify the required connections. Fig. INT. 5 illustrates a sample logic drawing and the wire list associated with the logic drawing. The wire list gives the wire number, the name of the signal connected by the wire, where the wire originates, the route it must take, and the destination of the wire for each connection.

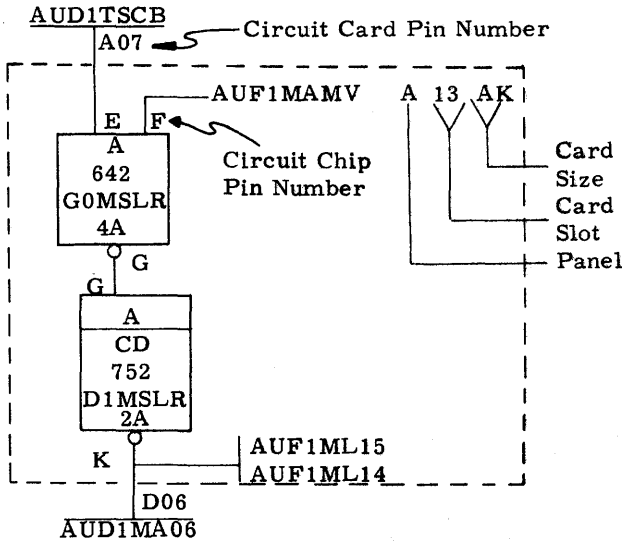


Fig. INT. 3 Logic Wiring Nomenclature

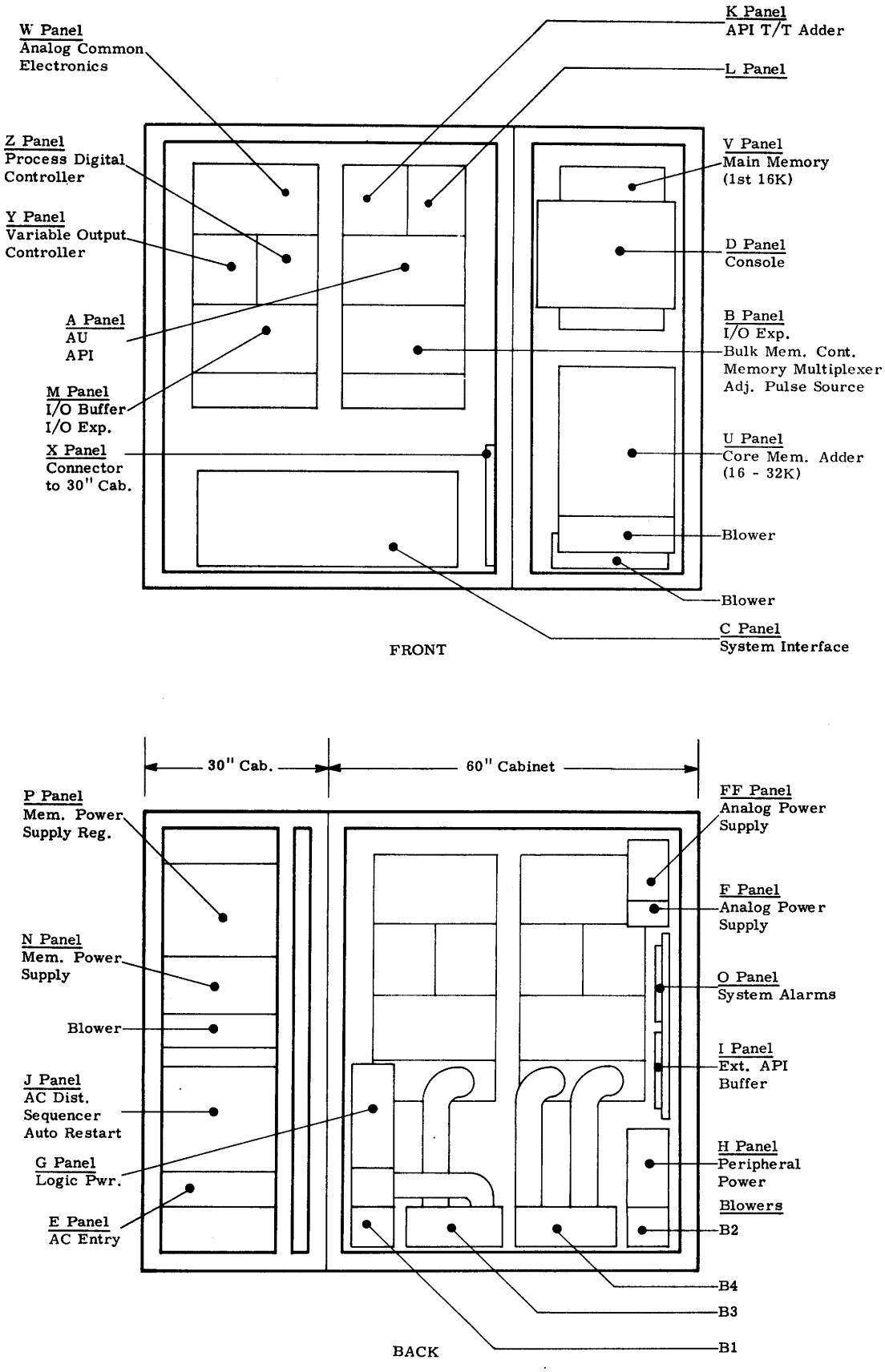
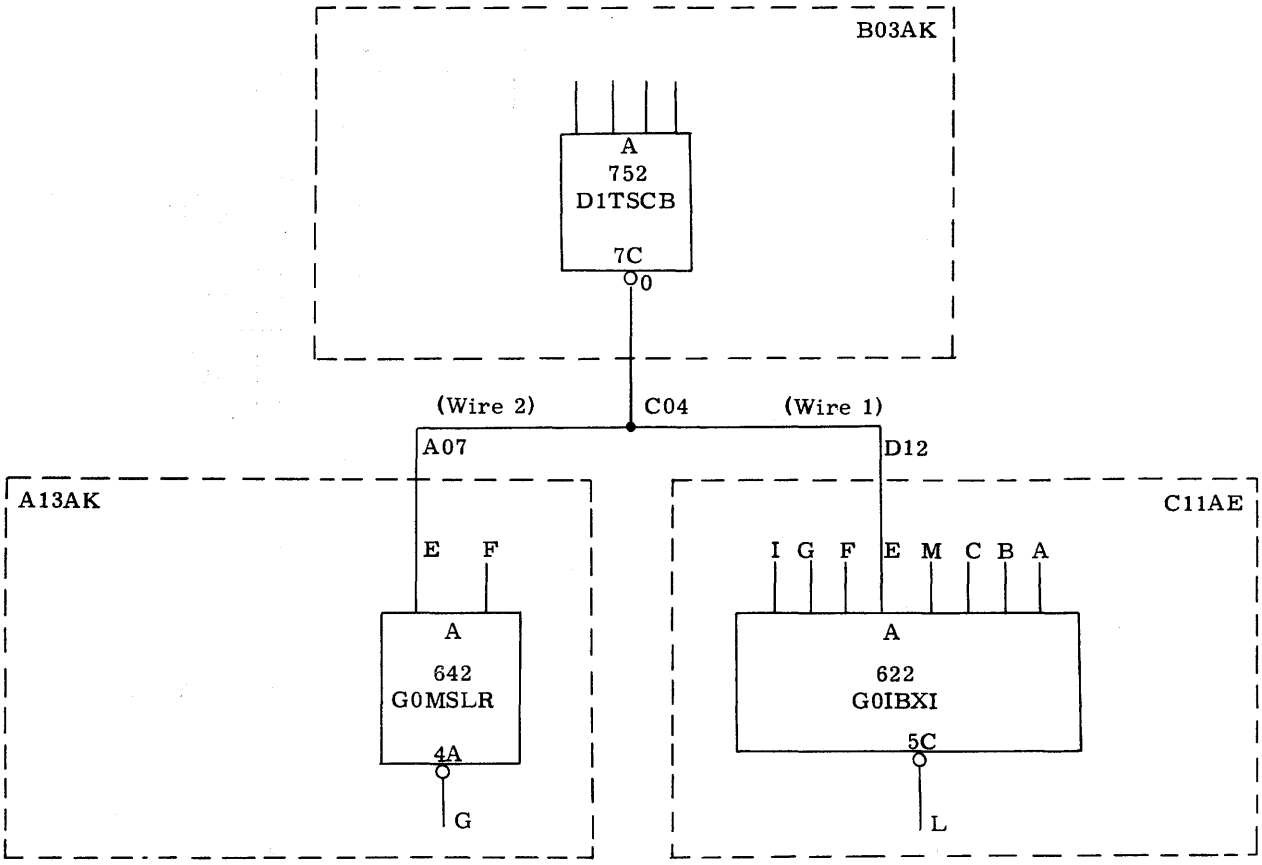
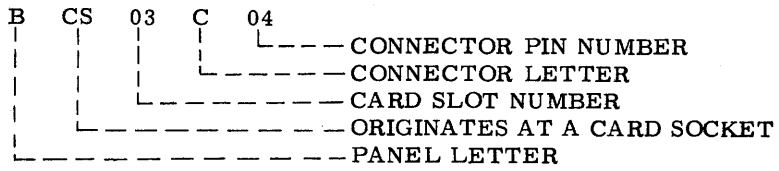


Fig. INT.4 Central Systems Unit Panel Locations



WIRE	SIGNAL	FROM	VIA	TO
1	AUD1TSCB	B CS 03 C 04	*	C CS 11 D 12
2	AUD1TSCB	B CS 03 C 04	*	A CS 13 A 07

The breakdown on the origination or "from" of the list is as follows:



* Means Point to Point.

Fig. INT. 5 Wire List

ARITHMETIC UNIT DESCRIPTION

The following discussion describes the functional block diagram of the 4022D Arithmetic Unit. Fig. DESC. 1 contains a block diagram of the Arithmetic Unit illustrating the interconnection of the various registers, parallel adder unit, serial adder, and the associated core memory module. This diagram does not attempt to define timing, sequencing, or control organization.

A Register (Accumulator)

The A Register is the accumulator for arithmetic and bit manipulation operation and temporary storage for data coming from or going to the registers of the I/O equipment. It is comprised of 24 high-speed flip-flops in a bit configuration numbered 00-23 with bit 23 being the most significant.

The A Register may be shifted right or left via the micro-coded GEN 1 command. However, the left shift (when using GEN 1) is accomplished by shifting right. A true left shift may be accomplished with a GEN 3 instruction where data from either B_{23} or B_{22} enters A_0 and is shifted left. On a typical right shift, A_0 passes through the serial full adder (where it may be altered) and back into A_{23}. Position A_{23} may also enter the serial full adder, possibly modified, and routed back to A_{23} and at the same shift A_{23} may be applied to A_{22}.

When A is shifted left, bit position 23 is lost. The contents of the A and B Registers may be shifted right where the contents of A_0 enter either B_{23} or B_{22} as specified by the shift instruction.

The contents of the A Register may be gated in parallel to the Parallel Full Adder and the A Register may receive data in parallel from the Adder. The contents of A may be selectively displayed on the programming and maintenance console.

B Register (Buffer)

The B Register is a 24-bit register used to hold all instructions and AU data going to or coming from core memory. During the fetch portion of an instruction, the 10 most significant bits of the B Register are transferred in parrallel to the I Register for decoding. The contents of the B Register are routed in parallel to the Parallel Full Adder in two fields: B_{13-0} for address modification, and B_{23-0} for full register operations. The contents of the Parallel Full Adder may be routed to the B Register in parallel.

The B Register may act as an extension to the A Register when performing right and left shifts (MPY, DVD, GEN 3, TIM/TOM). During a left shift, the contents of either B_{23} or B_{22} as specified by the instruction, enter position A_0 of the A Register. During a right shift, the contents of A_0 enter either B_{23} or B_{22} as specified by the instruction. The contents of B may be displayed on the programming and mainenance console.

P Register (Program)

The P Register is a 15-bit binary counter register whose primary function is to control the addressing of the next instruction. Depending upon the instruction being executed, the P Register may be incremented by one or two. Information from the I Register (address) is transferred to the P Register during the operation of Branch commands. The contents of P may be selectively displayed on the console.

> NOTE
>
> The P Register physically contains 16 bits, however, only 15 of these are used on systems with more than 16K of core memory and only 14 bits are used on systems with 16 K of core memory. The unused bits of the P Register are disabled by jumper pins in the Memory Multiplexer (70C180872, sheet 22.1).

J Register (Counter)

The J Register is a 5-bit binary counter used to control the length of shifts and to accumulate the count resulting from bit count instructions. J is controlled by the micro-coded GEN 1, GEN 3 instructions or Multiply, Divide, or TIM/TOM. The contents of J may be stored into any X location (LXC) for further use. The contents of J are displayed on the console.

I Register (Instruction)

The I Register is a 24-bit register used to hold the instruction while it is being decoded, interpreted, and executed. An additional bit (14) position is used in the I Register to provide for relative addressing. In addition to holding the instruction to be executed, the I Register is also used to hold the divisor during Divide, the multiplicand during Multiply, and the TIM/TOM control word during TIM/TOM operations.

The I Register may receive data in parallel from the Parallel Full Adder, B Register, and J Counter. Data is transferred from the I Register in parallel to the J Counter, P Register, Memory Address Gates, and to the Parallel Full Adder. The contents of the I Register may be selected for display on the console.

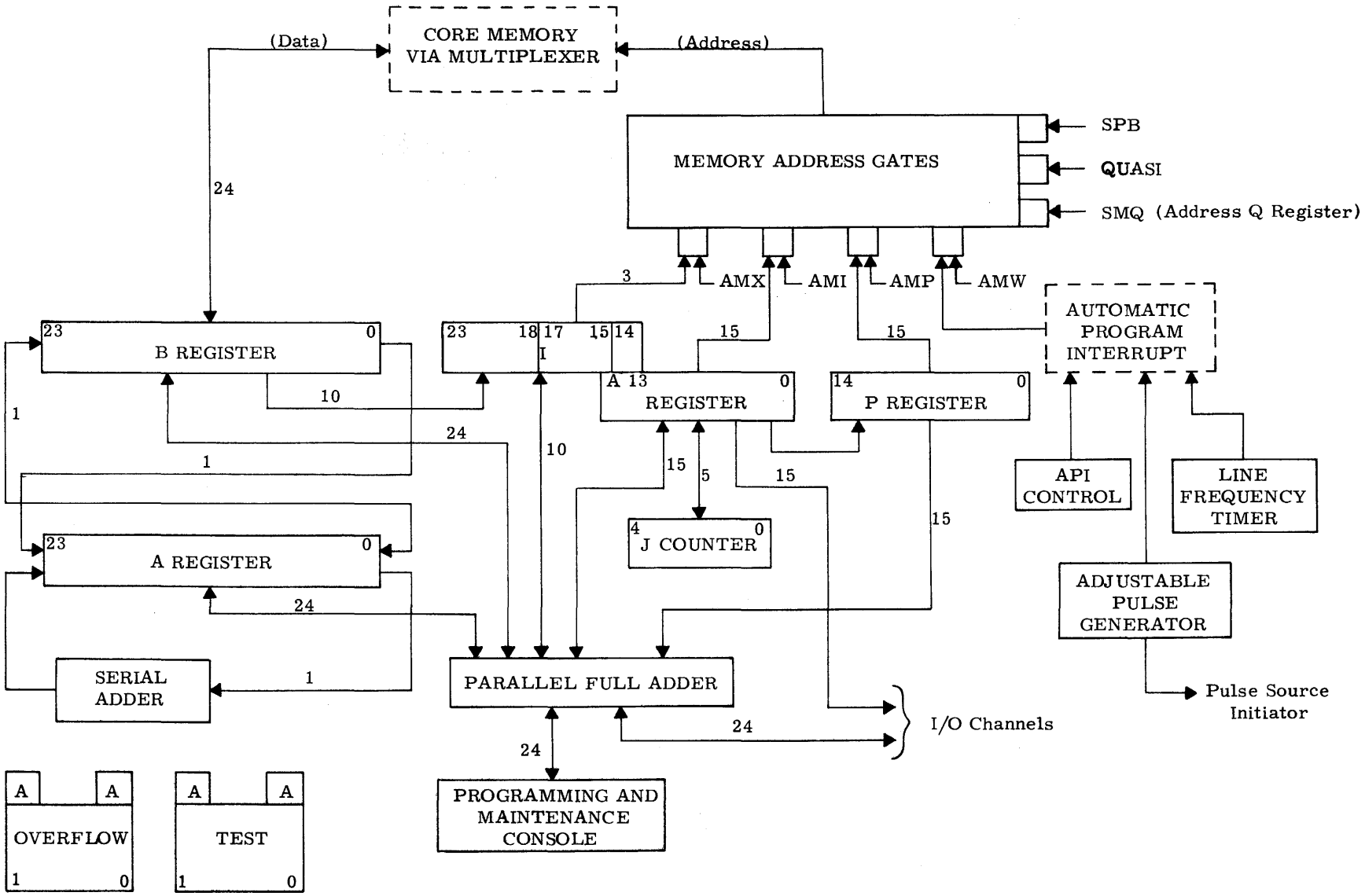


Fig. DESC. 1 GE-PAC 4022D Arithmetic Unit Block Diagram

Parallel Full Adder

The Parallel Full Adder is a 24-bit full adder. Most arithmetic operations within the arithmetic and control unit are accomplished through this adder. In addition to the normal add and subtract operations, the adder has the capability of accomplishing the following logical combinations:

Exclusive "OR" (if A = B, 0 out; if A ≠ B, 1 out).

Logical "AND" (if A = 1 and B = 1, 1 out; if A ≠ 1 or B ≠ 1, 0 out).

Logical "OR" (if A = 0 and B = 0, 0 out; if A ≠ 0 or B ≠ 0, 1 out).

The adder may receive data in parallel from A, B, I, or the P Register. The adder also acts as a buffer for Input/Output operations. Data transferred into the central processor unit from Input/Output devices is channeled through the adder for distribution within the arithmetic and control unit. Data transmitted from memory to specified Input/Output devices is routed through the B Register to the Adder for distribution outside the arithmetic and control unit. Data transmitted from the A Register is routed through the adder for distribution outside the arithmetic and control unit.

Q Register (Multiplier/Quotient)

The Q Register is a pseudo register (located in core memory address 10_8) used as an auxiliary accumulator (MPY and DVD). The contents of Q are also used to define fields of the A and/or B Register during GEN 3 commands. The Q Register is addressed and gated to the B Register during the command execution.

Memory Address Gates

The Memory Address Gates provide selection of the core memory address bits from the desired register or control circuit.

Certain memory locations are pre-assigned for specific use. These memory locations should be used only for the purpose intended unless hardware is not included to make use of the specific reserved address. The reserved memory locations and the pre-assigned uses are listed below.

Octal Address		Common Usage
000	-	Bulk Memory Pointer Word
001	-	SPB Link Storage
002	-	Quasi Operand Storage
003 – 007	-	Index Register
010	-	Q Register
011 – 012	-	Reserved
013 – 017	-	Unused
020	-	Memory Protect Branch Vector
021	-	Memory Protect/Watchdog Instruction Storage
022 – 023	-	Reserved
024	-	Watch Dog Branch Vector
025 – 037	-	Reserved
040 – 077	-	Quasi Branch Vectors
100 – 177	-	Memory Protect Status Map
200 – 377	-	Interrupt Response Locations
400	-	Quasi Package

Automatic Program Interrupt Control

The Automatic Program Interrupt control logic within the Arithmetic Unit provides timing synchronization between the AU and API modules, inhibits interrupts following the execution of specific instructions, inhibits inhibitable interrupts (response addresses 300_8 - 377_8) under program control and provides Echo interrupts to the API when certain conditions exist.

These control functions and the associated logic elements are described in the following paragraphs:

- Timing synchronization - The Enable API Gate, G1WENA (logic sheet 122) must be enabled to allow any interrupts to be serviced. This gate is only enabled during Time 3 Envelope (TSCA TSCB TSCC) of Sequence State 4 (AUA5) provided that an interrupt may occur following the execution of the instruction being executed.

  When an interrupt is generated by the API module, memory is addressed from the API module (G0SAMW) during the next Sequence State 1. This interrupt signal inhibits addressing memory from the P Register. G0SAMW enables memory address bit 7 (D1MA07) and in conjunction with the API Address Generator enables D1MA06-00 according to the interrupt level. In this manner an interrupt response address between 200_8 and 377_8 will be selected.

  A detailed description of the API module is contained in this book set.

- Inhibit Interrupts following the execution of specific instructions - Interrupts must not occur following the execution of certain commands to allow the operation performed by the command to be completed. For instance, interrupts are not allowed immediately following GEN 1 count instructions permitting the J Register count value to be stored. Interrupts are not permitted immediately following branch instructions so that the P Register can be updated.

  The following table lists those commands that do not permit automatic program interrupts following their execution and the basic logic path used to disable interrupts.

Non-Interruptable Instructions	Interrupts Disabled By:
BRU	$\overline{\text{G1WENA}}$ = $\overline{\text{AUA5}}$ (sh. 122)
BTR	" "
BTS	" "
CLO	$\overline{\text{G1WENA}}$ = DGN1·$\overline{\text{WI08}}$·WS67 (sh. 122)
CLZ	$\overline{\text{G1WENA}}$ = DGN1·$\overline{\text{WI08}}$·WS67
CMO	" "
CMZ	" "
CNTO	" "
CNTZ	" "
IAI	$\overline{\text{G1WENA}}$ = WGS3
IAI_2	AIF1ZINH (See API logic, 70C180046, sheet 10.)
JCB	$\overline{\text{G1WENA}}$ = NGS6 (sh. 122)
JDR	" "
JNO	" "
JNR	" "
LDP	$\overline{\text{G1WENA}}$ = XRMF (sh. 122)
LDX	$\overline{\text{G1WENA}}$ = WLDX (sh. 122)
LMR_1	AIG1ZNI1 (See API logic 70C180046, sheet 10.)
LMR_2	" "
LPR	$\overline{\text{G1WENA}}$ = XRMF
Quasi*	$\overline{\text{G1WENA}}$ = DQUA (sh. 122)
SPB	$\overline{\text{G1WENA}}$ = XRMF (sh. 122)
TXH	$\overline{\text{G1WENA}}$ = $\overline{\text{AUA5}}$ (sh. 122)
XEC**	$\overline{\text{G1WENA}}$ = $\overline{\text{AUA5}}$ (sh. 122)

* Quasi instructions AKA, SKA, and LDK are considered interruptable because the ADD, SUB, and LDA object instructions may be interrupted.

** If the object instruction of XEC is interruptable, XEC is considered interruptable.

The API ENBL switch on the Programming and Maintenance console in the lockout (up) position inhibits all interrupts from occuring by disabling G1LSP1 which disables G1WENA.

- Inhibit inhibitable interrupts - Inhibitable or low priority interrupts (those interrupts with response addresses of 300_8 to 377_8) may be disabled and enabled under program control.

  The Permit Automatic Interrupt flip-flop, F1WPMT (logic sheet 121), when reset inhibits all inhibitable type interrupts. F1WPMT is reset during Time 4 Envelope (TSCA·TSCC) of State 4 when there is not a Volume Difference or Memory Protect Error by an SPB, IAI, or LPR command.

  $\overline{\text{F1WPMT}}$ = TSCA·WCPM·ECLK + Initialize

  F1WPMT is set, allowing inhibitable interrupts, at Last Pulse of State 4 during the execution of a PAI or LPR command having bit 21 a "one".

  F1WPMT = WSPM·TLPE·ECLK.

Clearing F1WPMT at last pulse of State 4 inhibits inhibitable interrupts for one or more instruction following the PAI or LPR with bit 21 a "one".

- Echo interrupt signals - G1WEKO (Sheet 122) applies a signal to the API to generate an "Echo" interrupt when a TIM/TOM operation is in progress and the last character of the last word is being transferred or when a DMT command decrements the memory cell from 0 to -1. Although the Echo signal is generated whenever the DMT decrements a cell from 0 to -1, a new interrupt is generated only if the DMT resulted from a previous interrupt.

A detailed description of the automatic program interrupt module is contained following the API tab in this book set.

Overflow Flip-Flop

The Overflow flip-flop, F1UOFL (logic sheet 54), provides arithmetic overflow detection during the execution of ADD, DVD, MPY, and SUB commands. In addition, the Overflow flip-flop is set during the execution of GEN 3 commands when bit 5 of the command word is a "one" if the exclusive "OR" of A_{23} and A_{22} is a "one".

Arithmetic overflow occurs when the result of an arithmetic operation provides, or will provide, a result whose magnitude exceeds the capacity of 23 bits for single word length register (i.e., 2^{23}-1 or -2^{23} operations, or exceeds the capacity of 46 bits for double length register operations (i.e., 2^{46}-1 or -2^{46}). During GEN 3 commands, when bit 5 of the command word is a "one", overflow occurs when bit 23 (sign bit) of the A Register is changed during a left shift.

The status of the Overflow flip-flop is monitored and cleared by the JNO (GEN 2) command to establish program location control. The status of the Overflow flip-flop is stored by the SPB command; restoring of the Overflow flip-flop status may then be accomplished by the LPR command.

Test Flip-Flop

The Test flip-flop F1ETST (sheet 89) serves as a memory element to remember the status of a previous test and to retain this status until changed by the program.

The program may then use the status of the Test flip-flop to initiate a branch.

The Test flip-flop control logic provides the following functions:

- During the execution of DMT, GEN 1, and TXH commands, the Test flip-flop is used to remember the status of the condition tested.
- BTR and BTS commands will cause a branch in the program depending on the status of the Test flip-flop.
- The SPB command stores the status of the Test flip-flop; the status may be restored using the LPR command.
- The SET (GEN 1) command unconditionally sets the Test flip-flop; the RST (GEN 1) command unconditionally clears the Test flip-flop.

Programming And Maintenance Console

The Programming and Maintenance Console provides one method by which the operator may communicate with the computer in machine language. It allows the operator to load and execute programs and monitor the running program. A detailed description of the Console is contained under the mnemonic CON in this section.

Line Frequency Timer

The line frequency timer generates logic level signals from the 60Hz (or 50Hz) AC source. These signals, occurring at 1/60 (or 1/50) of a second or 1/120 (or 1/100) of a second, may be used for elapsed time counting, a digital clock, and as the 60Hz clock source for the optional Pulse Source Initiator. A jumper pin is provided (NSTC5 board in slot A21AE) to select the frequency (input power frequency or twice input power frequency) of the timing signal to the API input.

The logic level pulses are wired to the Automatic Program Interrupt module providing interrupts at the pin selected rate. The associated interrupt response address contains a Decrement Memory and Test command (DMT) to accomplish a time keeping function by decrementing a count preset in a memory location. When the count passes from zero to minus one, the DMT command causes an "echo" API signal which informs the program that a predetermined time period has elapsed. The program then takes whatever action is appropriate for this time interval and reloads the count value in the memory location decremented by the DMT command.

A schematic of the line frequency timer is contained on sheet 145 of the Arithmetic Unit logic.

STALL ALARM

The Stall Alarm is used to detect a malfunction in the computer program or hardware that causes a hang-up or stall condition in program sequencing. The Stall Alarm consists of a manually adjustable (1 to 5 second) timer, normally set at 5 seconds, that is reset by the GEN 2 SSA command. If the timer is allowed to "time out" because the program cannot get back to an SSA command to again reset it, an output signal is available to light the STALL and ALARM indicators on the computer console and to alarm equipment external to the CSU via the System Alarm option.

The Stall Alarm, Fig. DESC. 2, consists primarily of a manually variable timer and a flip-flop (F1NSTA). If the timer is allowed to "time out", the flip-flop is set providing the error signal.

Operation of the timer and flip-flop is listed below:

- Timer Circuit Enabled:
  1. The console key switch is in the disable position and the SSA command is not executed.
  2. The console key switch is in the enable position, the AUTO/MAN toggle switch is in the AUTO position, the STALL ENBL is in the enable position, and the SSA command is not executed.
- Reset the Timer Circuit:
  1. The console key switch is in the enable position and the STALL ENBL switch is in the lockout (up) position.
  2. The console key switch is in the enable position and the AUTO/MAN switch is in the MAN position.
  3. The GEN 2 SSA command is executed.
- Set the Stall Flip-Flop (F1NSTA):
  1. The timer is allowed to time out providing a 200 micro-second "0" timer output.
- Reset the Stall Flip-Flop (F1NSTA):
  1. Initialize.
  2. The console key switch is in the enable position and the STALL ENBL switch is in the up position.

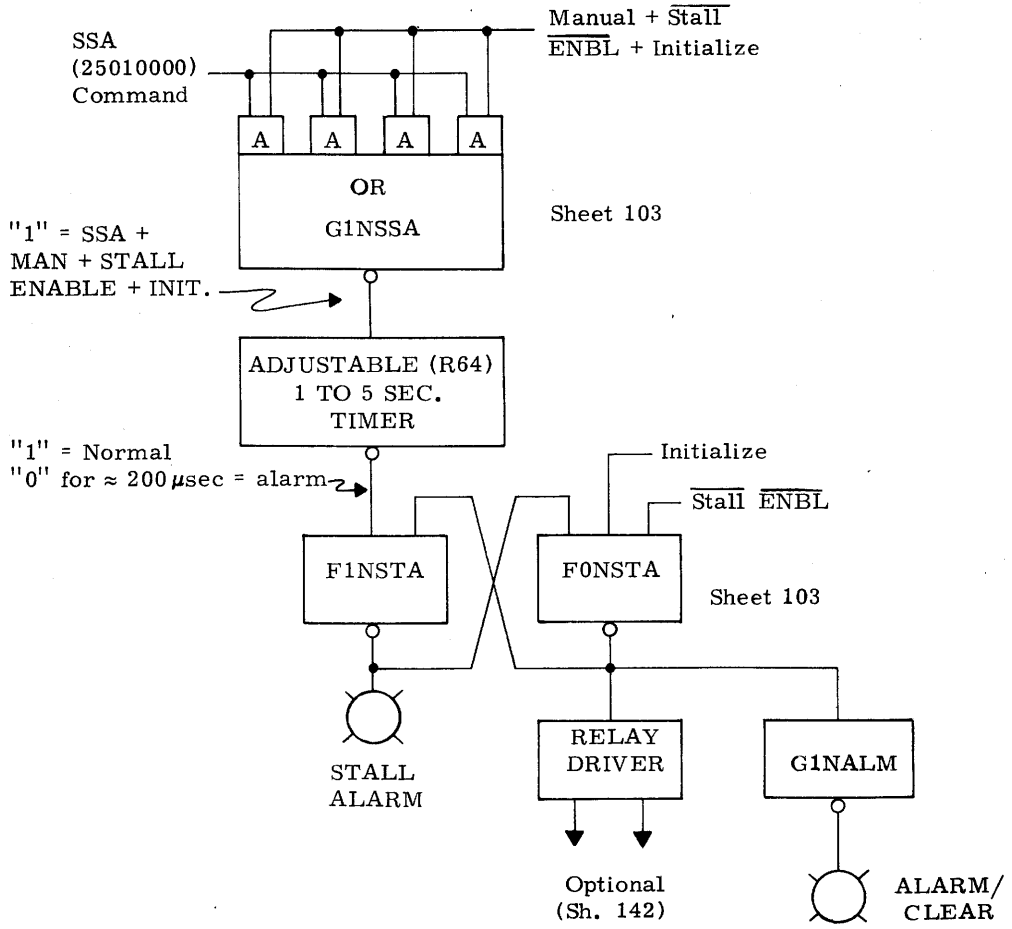


Fig. DESC. 2 Stall Alarm

CONSOLE

INTRODUCTION

The GE-PAC 4010 Console (Fig. CON. 1) is the basic man/machine communication link and control element of the Central Processor. It provides all the necessary indications and switches to apply power, initialize, load programs and monitor the operation of the machine.

This console is not normally available for process engineers. External consoles are available for communication with the computer by plant personnel.

Table CON. 1 lists the operation of the controls and indicators and the functions performed by each. The purpose of this discussion is to acquaint the reader with the operation of the Console and the function of the switches and indicators contained thereon. Basic procedures to enter data in a register, load in programs via punched cards or paper tape, store words in memory locations, display memory locations, and to store a constant throughout all core memory locations are also provided.

DESCRIPTION

Register Select—P,I,B,A

In the MANual mode, with the Console Enabled, the contents of the register selected by the Register Select-P, I, B, A switches are displayed by the Console Bit switch indicators. In this manner, the operator may observe the contents of the P, I, B, or A Registers.

Fig. CON. 2 contains a block diagram of the logic circuits involved. The contents of the particular register selected is gated to the Parallel Adder Unit during $\overline{\text{TSCA}}$ when in the MAN mode, with the console enabled, and the CLEAR switch not pressed. The contents of the Parallel Adder Unit are then displayed by the indicators. Since sequencing is held in $\overline{\text{TSCA}}$ between pressing the STEP switch (see MANual STEP Operation) a display of the selected register is provided.

Console to A, B

Using the Console Bit switches, data may be entered in the A or B Register as determined by the Register Select Switch. Fig. CON. 3 contains a block diagram of the logic utilized to implement this function. With the Console Enabled, in the MANual mode, and the A or B Register selected, pressing the Console Bit switches sets the corresponding bits in the A or B Register. The status of the pushbuttons are transferred to the A or B Register via the Parallel Adder Unit. As illustrated in the Register Select discussion above, the contents of the selected register are also gated to the Parallel Adder Unit. Therefore, if the A or B Register is not cleared prior to pressing the Console Bit switches, the Console Bit switches pressed will be ORed into the selected register. Operation to clear the A or B Register is described below.

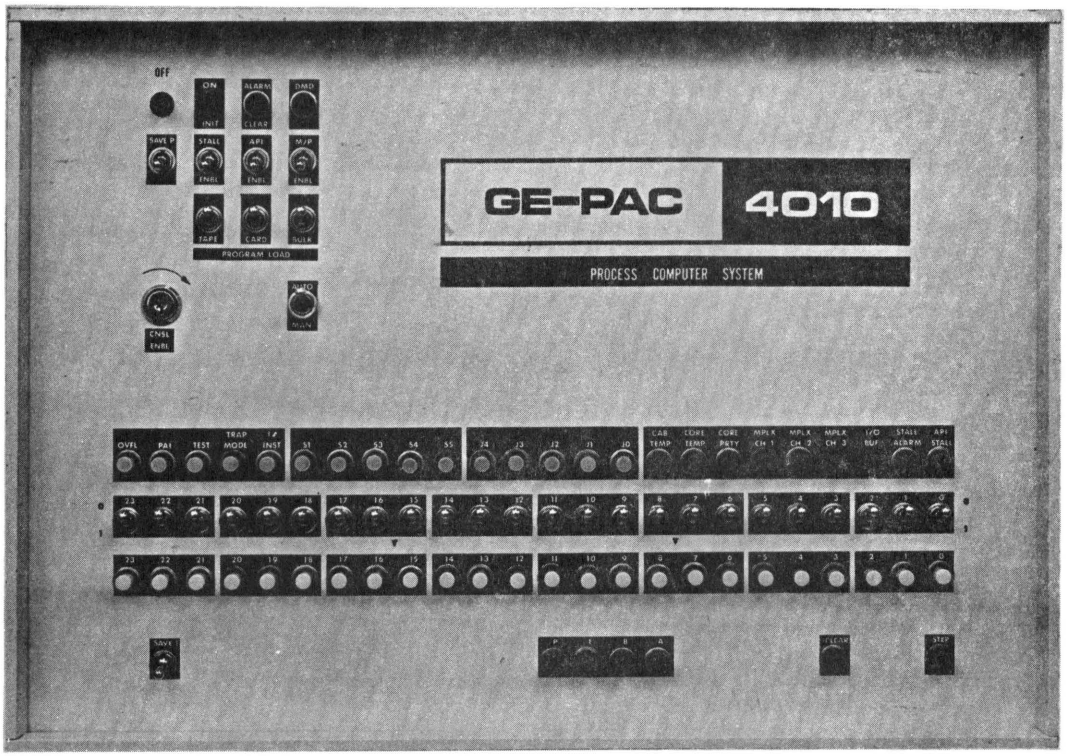


Fig. CON. 1 Operator's Console

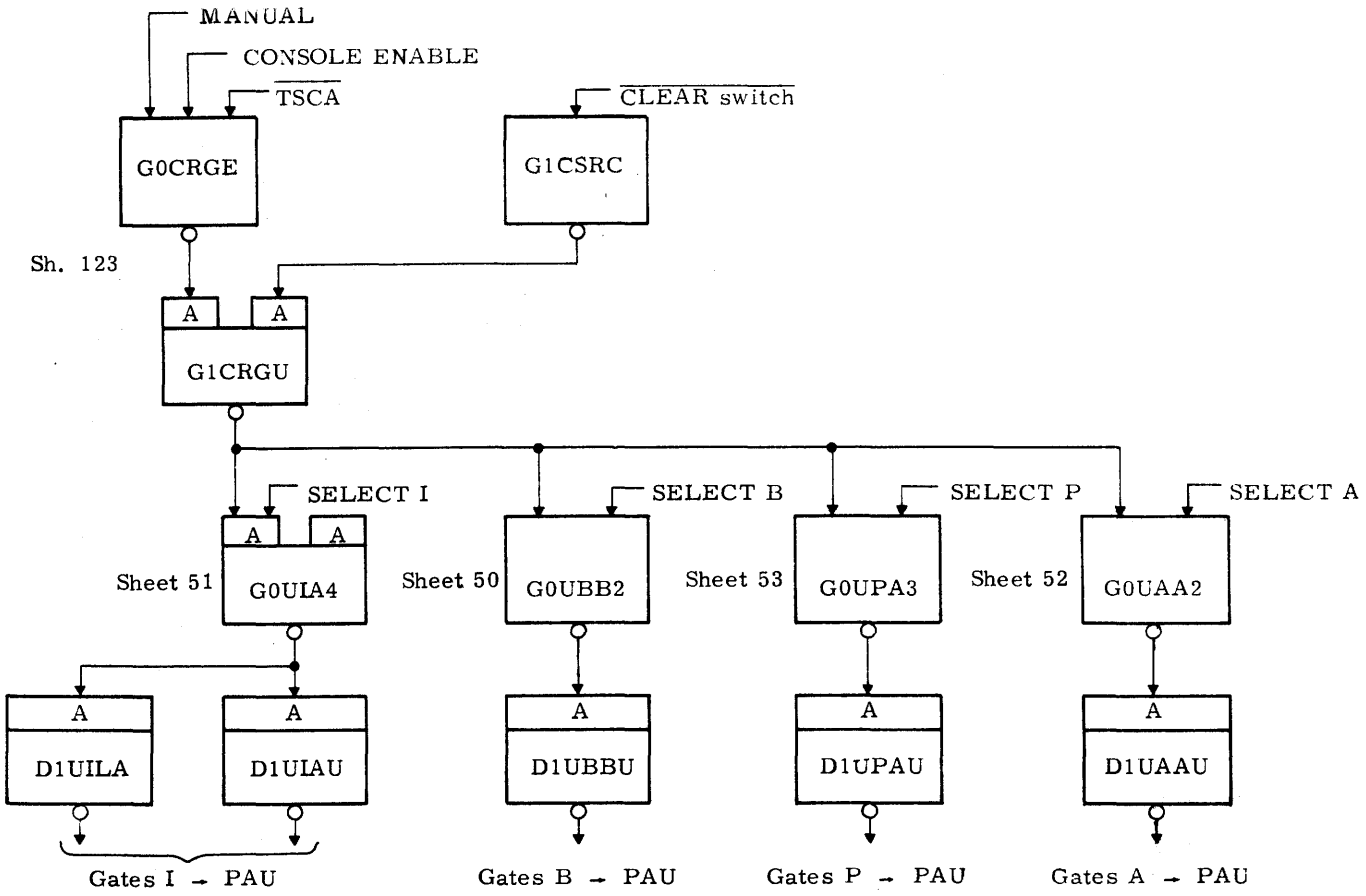


Fig. CON. 2 Register Select Block Diagram

CLEAR Register

The CLEAR Reg. switch, when pressed, clears the B or A Register depending upon which is selected by the Register Select switch. In the MANual mode with the Console Enabled, and the A or B Register Selected, the status of the Console Bit switches are gated to the selected Register via the Parallel Adder Unit. Since none of the Console Bit switches are pressed, this will clear the selected register. As shown in Fig. CON. 2, pressing the CLEAR Reg. switch, disables gating the contents of the Selected Register to the Parallel Adder Unit. In this manner, since no data is applied to the Parallel Adder Unit, the A or B Register will be cleared.

MANual STEP Operation

Pressing the STEP switch with the Console Enabled and in the MANual mode, will cause the instruction contained in the B Register to be executed. Instruction sequencing is halted after executing this instruction with the instruction just executed in the I Register and the next instruction in the B Register.

Fig. CON. 4 illustrates the basic timing, with logic equations, of the MANual STEP operation. When in the MANual mode, sequencing is halted in State 1, time 2 envelope. Pressing and releasing the STEP switch will allow the instruction contained in the B Register to be executed. Sequencing will then be halted during time 2 envelope of the following State 1. In the MANual Mode, sequencing is held in time 2 envelope by enabling G1TMEN and disabling G1CTAE during State 1. Pressing and releasing the STEP switch will enable G1CTAE and disable G1TMEN and allow sequencing to continue until the next State 1.

Program Load

The Program Load feature permits an operator to transfer a loader program to core memory from the primary bulk memory (drum or disk), the primary paper tape reader, card reader or I/O typer if it is ASR teleprinter. Only the following steps are required by the operator to bring in a loader program.

1. If a minimum loader card or tape is to be loaded, ready the reader.
2. Lower the appropriate PROG LOAD toggle switch.
3. Press the ON switch to initialize the computer.

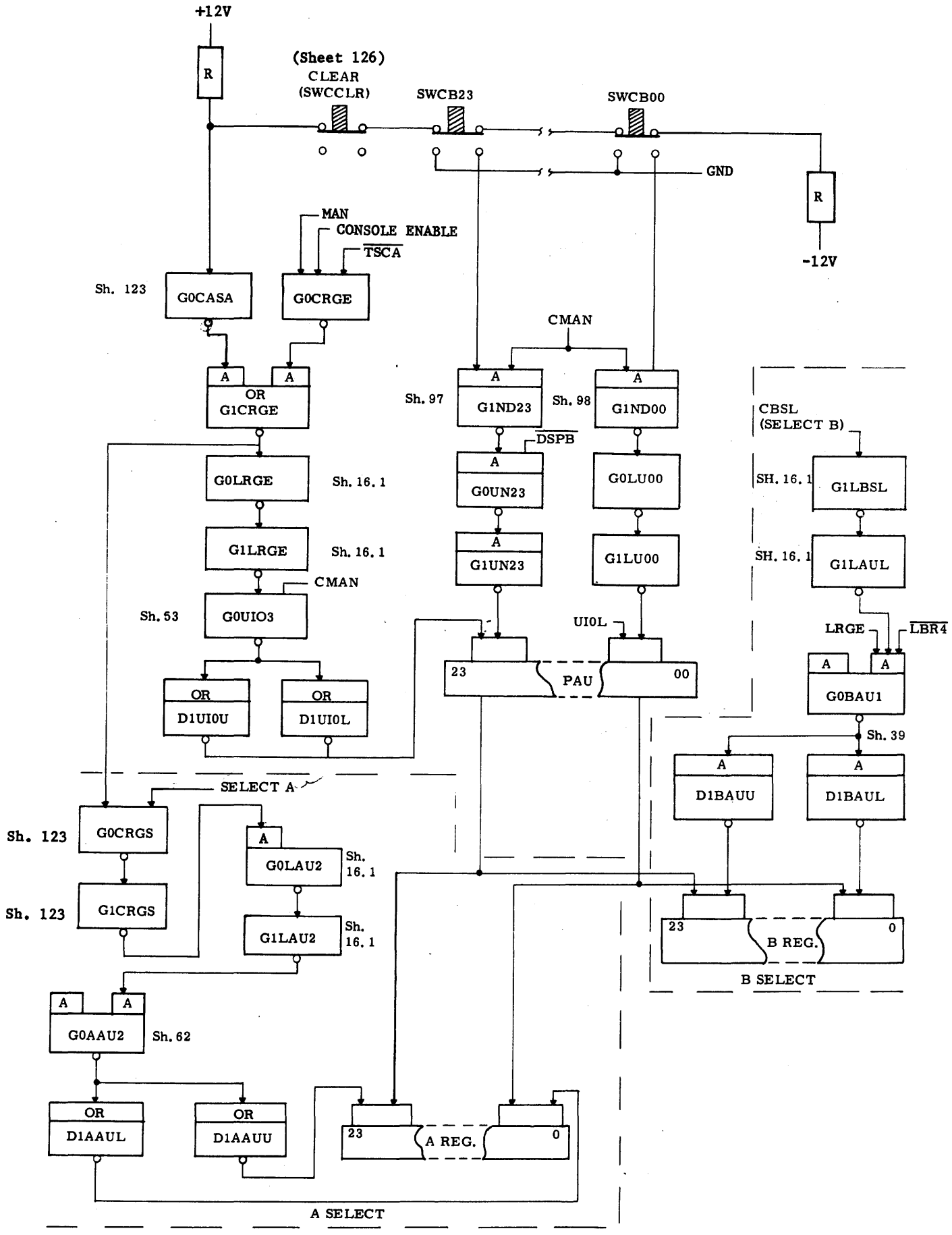
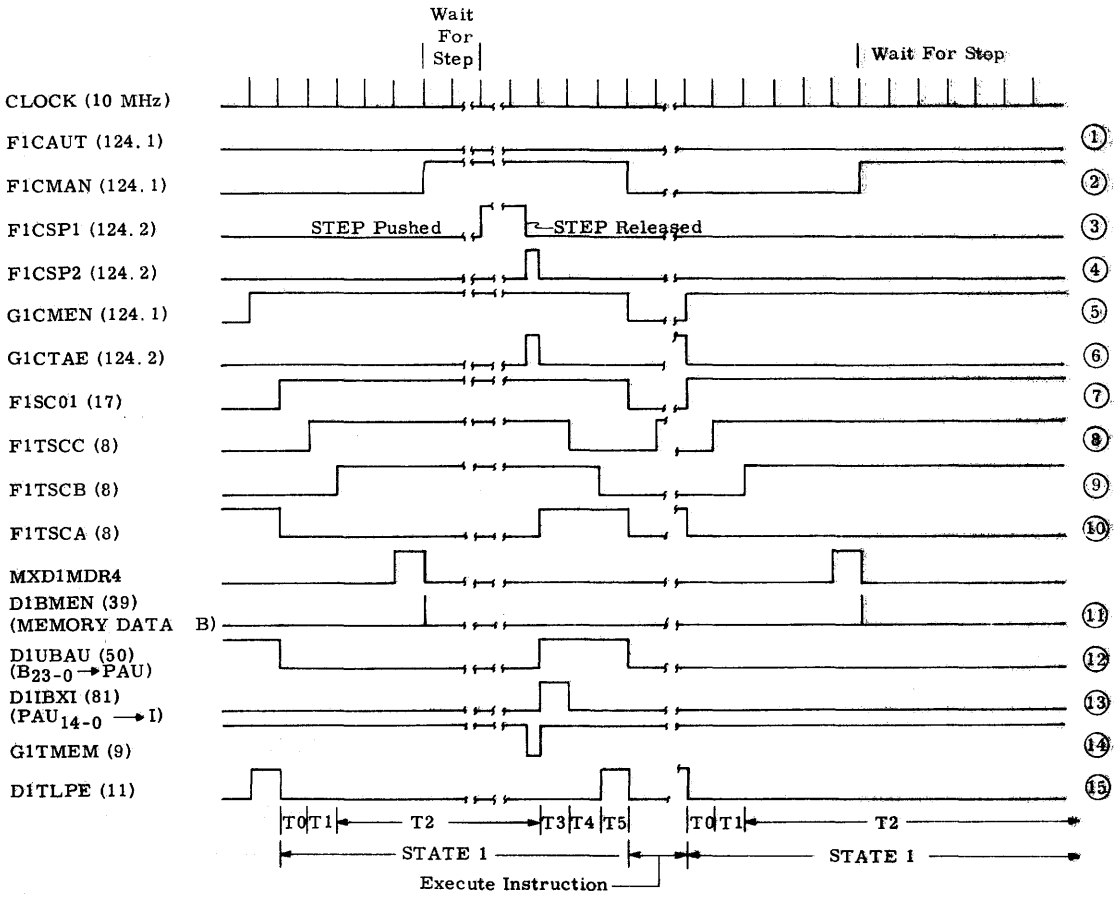


Fig. CON. 3 Console To A, B Block Diagram



(1) F1CAUT = CAUM · CCLK · ($\overline{\text{MUF0PPER}}$ + $\overline{\text{STOP ON PARITY ERROR}}$)
SWCAUM = AUTO + CONSOLE DISABLED

(2) F1CMAN = $\overline{\text{CAUT}}$ · MDR4 · CMEN · CCLK
G1CMEN = SC01 · $\overline{\text{MVDA}}$
$\overline{\text{F1CMAN}}$ = $\overline{\text{TSCB}}$ · TSCA · CCLK

(3) F1CSP1 = $\overline{\text{SSTPNO}}$ = STEP Switch Pushed
$\overline{\text{F1CSP1}}$ = SSTPNC = STEP Switch Released

(4) F1CSP2 = $\overline{\text{CSP2}}$ · Clocked by CSP1
$\overline{\text{F1CSP2}}$ = CSCA

(5) G1CMEN = SC01 · $\overline{\text{MVDA}}$

(6) G1CTAE = TSCB (CSP2 + $\overline{\text{CMEN}}$ + CAUT)

(7) F1SC01 = SSS1 · TLPE · SCLK

(8) F1TSCC = TSC2 · $\overline{\text{TSCB}}$ · TCK2
G1TSC2 = $\overline{\text{TSCA}}$
$\overline{\text{F1TSCC}}$ = TSCB · TSCA · TCK2

(9) F1TSCB = $\overline{\text{TSCA}}$ · TSCC · TCK2
$\overline{\text{F1TSCB}}$ = $\overline{\text{TSCC}}$ · TCK2

(10) F1TSCA = $\overline{\text{TMEN}}$ · TSCB · TCK2
$\overline{\text{F1TSCA}}$ = TLPE · TCK2

(11) D1BMEN = MDR4 · BMRQ · $\overline{\text{STOR}}$ · BCLK

(12) D1UBAU = UBA3 = SC01 · TSCA · $\overline{\text{UBB1}}$ · UAMV

(13) D1IBXI = $\overline{\text{ISAV}}$ · TSCB · $\overline{\text{HTTF}}$ · MSSI · SC01 · ISCA · TSCC

(14) G1TMEN = CMAN · CTAE

(15) D1TLPE = TLP4 + (TLP1, 2, 3, 4*)
TLP4 = SC01 · $\overline{\text{MBR4}}$ · TSCA · $\overline{\text{TSCB}}$

*Depends on instructions being executed.

Fig. CON. 4 Manual Mode Step Sequencing

4. If the device is the primary reader (7000_8) or if the device is the primary bulk memory (1000_8) no action is required. If the device is the I/O (ASR) typer, press register entry switches for the K0 code (3_8). These K0 bits will be ORed into the B Register forming 7103_8.

5. Place the API ENBL switch in the lockout position.

6. Place the AUTO/MAN switch in the AUTO position.

   The card minimum loader will be stored in memory locations 1 through 50_8.

   The tape minimum loader will be stored in memory locations 1 through 60_8.

   A bulk transfer of 100 octal words starting at bulk address 0000 of the first bulk device is stored in core memory beginning at location 0. This program will normally be the RTMOS initialization routine.

7. Raise the PROG LOAD switch.

   Program control goes to location 1. The minimum loader will hang up in a DEMAND loop awaiting operator communication. That is, pressing the DMD button will allow subsequent programs to be read in using the minimum loader program which has just been loaded.

Refer to the following GE drawings for further information regarding the use of the minimum loader.

68A977332	-	Instructions for Loading 4020 Test Programs
68A977090	-	Operating Instructions, Minimum Loader 4020B.
68A978788	-	Paper Tape Minimum Loader (Description and Tape)
68A978789	-	Card Minimum Loader (Description and Card)

The following conditions are required for Program Load operation.

- The paper tape reader or card reader must be connected to an automatic program interrupt with response location 201_8 or 202_8.
- The bulk memory device must use location 0 for its pointer word address.

Each of the three Program Load modes are described separately in the following text.

Tape Program Load Operation

The minimum loader program contained on paper tape may be read into memory from the first/primary reader device. The hardware specifies a device code of 7000. Since the K0 address is 3, bits 0 and 1 must be entered at the console register entry switches as mentioned in the procedure above.

Fig. CON. 5 contains a flowchart of the basic functions performed by the program load circuitry to read in a minimum loader tape. As shown in the flowchart, the following functions are performed in the listed sequence.

1. Pressing the ON switch to initialize the computer, clears the B, P, and I Registers.

2. The P Register is saved. This means that program control will be held to location 0.

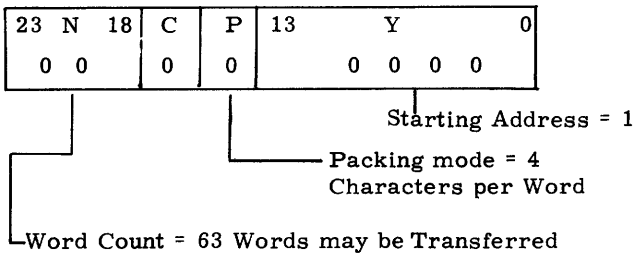
3. Interrupts to response addresses 203_8 and above are inhibited.

4. Interrupts to response addresses 200, 201, and 202_8 are enabled. Interrupt response address 200_8 is reserved for automatic restart and therefore, will not be present to interrupt the program. The paper tape reader used to load the minimum loader is connected to interrupt response address 201.

5. The A Register is set to the DC1 code (21_8) required to enable the paper tape reader with the OPR command.

6. The contents of the B Register (00000000_8) are stored in memory location 0. This LDA instruction will be the interruptable running program since the P Register is saved and contains 0000_8.

7. The TIM control word is stored in location 201_8. (The TIM control word is also stored in location 202_8, however this data is not used.) This TIM control word is all zeros specifying that 63 words with 4 characters per word may be read in and stored beginning in location 1.



8. The B Register is set to the OPR command with the 7000_8 device code. Since bits 0 and 1 were set manually into B, the command becomes (25027003_8). This OPR command will be executed when the AUTO/MAN switch

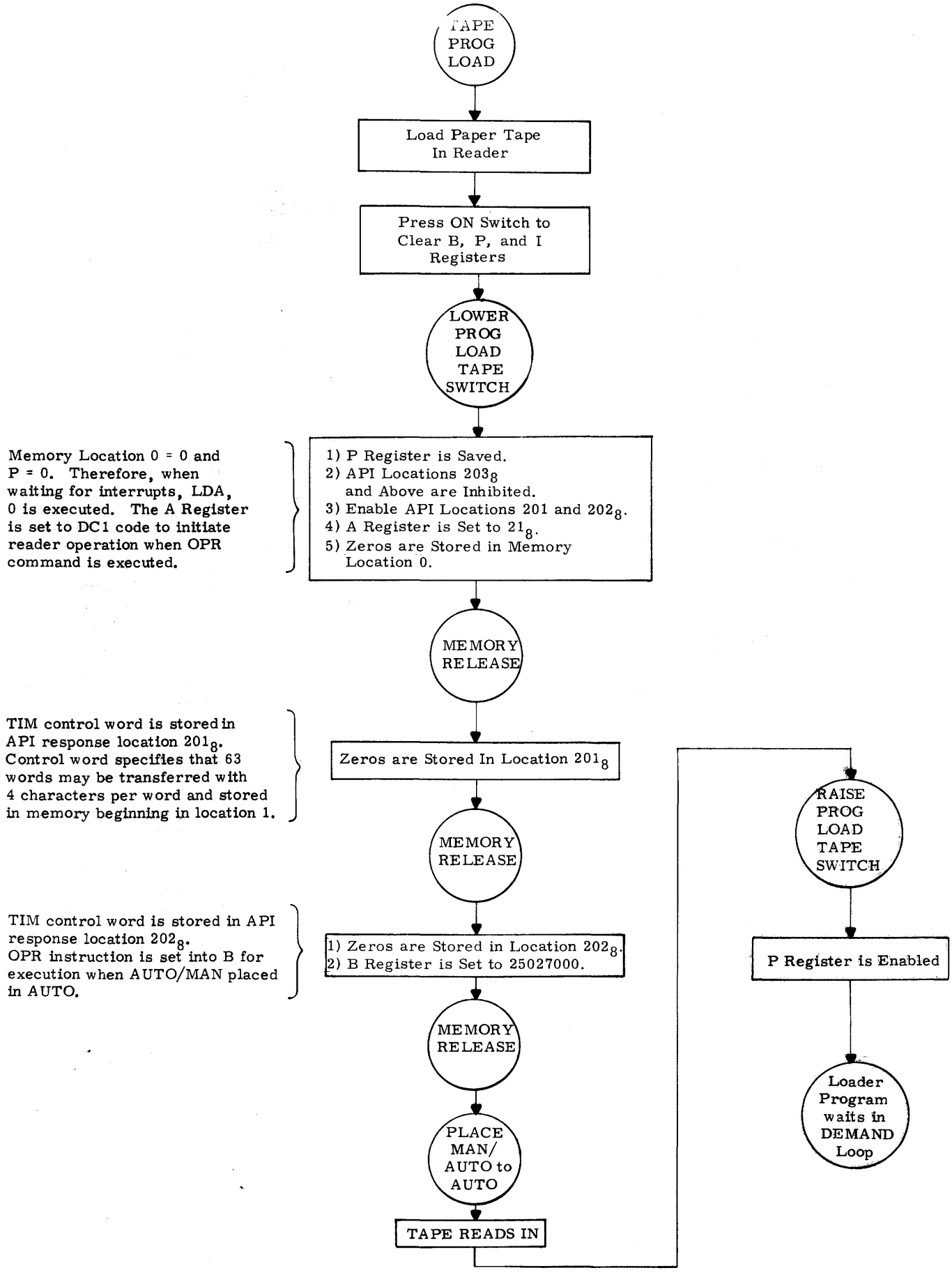


Fig. CON. 5 Program Load, Tape Flowchart

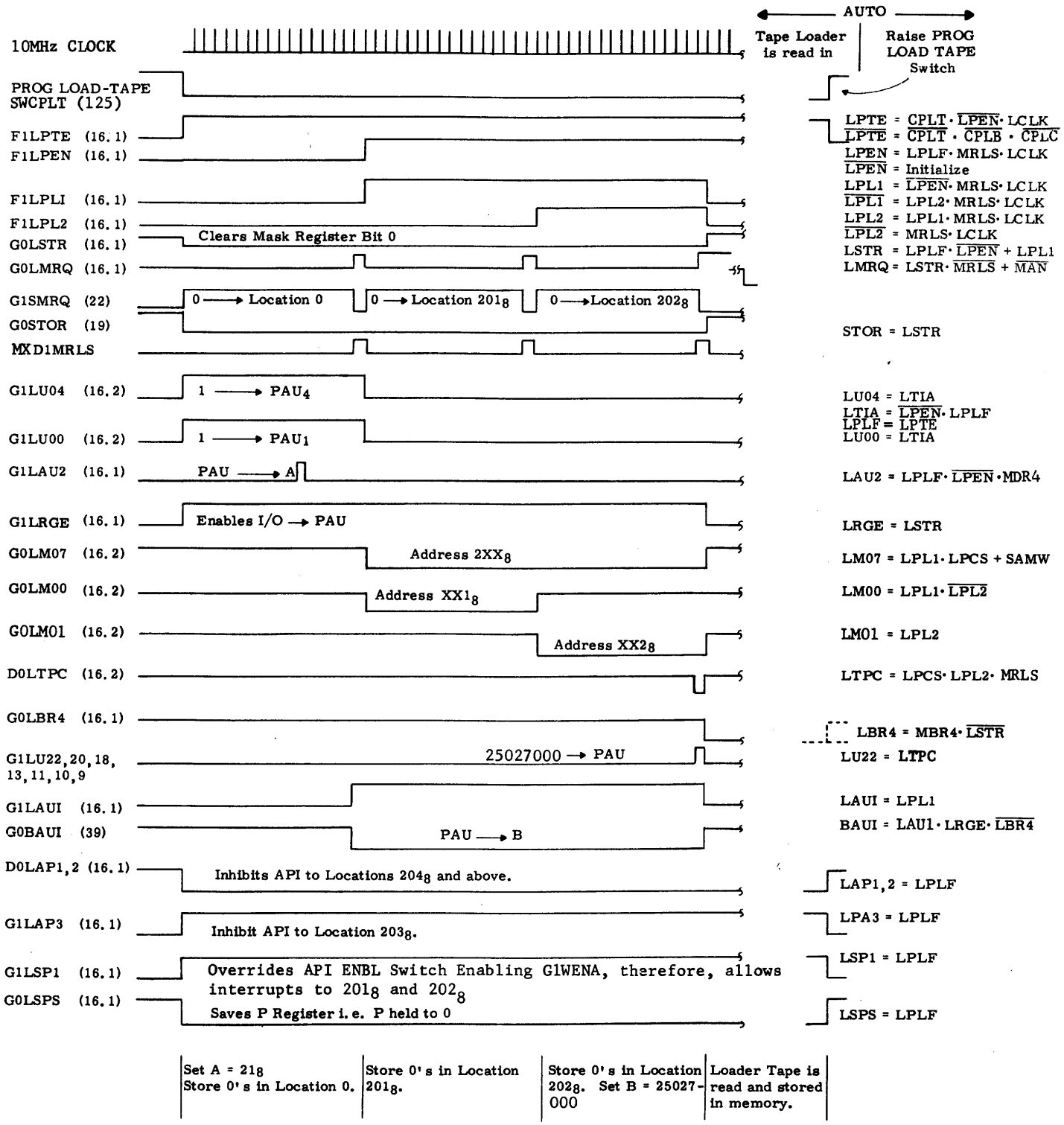


Fig. CON. 6 Tape Program Load Timing Diagram

is placed in the AUTO position. Executing the OPR command with the DC1 code in the A Register will initiate the reader operation.

9. Placing the computer in the AUTO mode will cause the reader to be initiated as described above. The interruptable running program will be the LDA command in location 0. Therefore, the minimum loader contained on paper tape will be read in and stored beginning in location 1.

10. After the tape is read in, raising the PROG LOAD TAPE switch will remove the Program Load inhibit of API locations 203_8 and above and remove the save P function. Program control goes to location 1 of the minimum loader routine.

A timing diagram with logic equations of the primary logic elements used to perform the program load function for paper tape is shown in Fig. CON. 6. Since the timing and primary function of each logic element is shown on the timing diagram, only a general discussion of these signals is provided in the following text.

F1LPTE is set when the PROG LOAD - TAPE switch is placed in the down position and remains set until the TAPE switch is raised.

G0LSTR applies a signal to the API module to clear bit 0 of the optional API Mask Register. Clearing bit 0 insures that the Mask Register will not inhibit interrupts to locations 200 through 203_8. This is necessary so that interrupts to location 201_8 from the paper tape reader may be recognized.

D0LAP1, 2 when enabled, applies a signal to the API module to inhibit interrupts to response addresses 204_8 and above. This is required so that other interrupts (e.g. real time clock) will not interrupt the program while the tape is being read.

G1LAP3 when enabled, provides an inhibit of interrupts to response address 203_8. This signal, in conjunction with D0LAP1, 2 inhibits all interrupts except 200, 201, and 202_8. Since 200_8 is used for automatic restart, only the interrupt response addresses 201 and 202_8 are enabled.

G0LSPS when enabled, performs the same function as the SAVE P switch in the save position. Therefore, since P was cleared by pressing the ON switch, P will be held equal to 0 during the load operation.

G1LSP1 when enabled, is used to enable the enable interrupt gate, G1WENA. If G1WENA is inhibited (e.g. API ENBL in the lockout position), the reader interrupts would be inhibited.

G1LU04, 0 when enabled, applies 21_8 to the A Register via the Parallel Adder Unit. This 21_8 is the DC1 code used by the OPR command to initiate the reader operation.

G1SMRQ when enabled, requests access to memory. Three requests to memory are required to store zeros in locations 0, 201 and 202_8. The zeros stored in location 0 correspond to an LDA command and will be used as the interruptable running program. The zeros stored in location 201 and 202_8 will be the TIM control word specifying that up to 63 words of 4 characters per word may be read in and stored beginning in memory location 1.

G0STOR when enabled, enables the store rather than read operation of the memory. Therefore, data is stored in locations 0, 201, and 202_8.

G0LM07, 0 when enabled, addresses memory location 201_8 to store the TIM control word.

G0LM07, 1 when enabled, addresses memory location 202_8 to store the TIM control word.

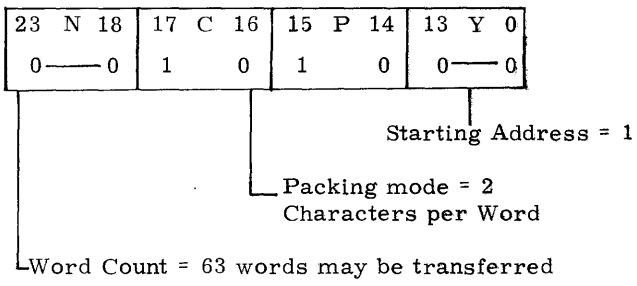
G1LU22, 20, 18, 13, 11, 10, 09 when enabled, sets the B Register (via the Parallel Adder Unit) to 25027000. This is the OPR command that will be the first instruction executed to initiate the reader operation when the computer is placed in AUTO.

I/O Teleprinter Program Load Operation

Operation is the same as that for the paper tape reader except the device code is 7000_8 and bits 0 and 1 should not be set in the B Register prior to operation.

Card Program Load Operation

The card program load operation is exactly the same as the tape program load operation except the TIM control word stored in API response addresses 201 and 202_8 specifies 2 characters per word packing instead of 4 characters per word packing. To provide control for this function, F1LPCE is set by the CARD PROG LOAD switch instead of F1LPTE. The TIM control word stored in locations 201 and 202_8 for card program load operations is in the following format:



To store this TIM control word in locations 201 and 202_8, bits 17 and 15 of the B Register are set to "1".

This is performed by enabling G1LU17 and G1LU15 during the time that these words are stored in memory (F1LPL1).

$$G1LU17,15 = G0LTCR$$

$$G0LTCR = LPL1 \cdot LCRS \cdot \overline{LRLS}$$

$$G1LCRS = LPCE$$

Although the DC1 code in the A Register is not required to initiate the card reader operation with the OPR command, 21_8 is set in the A Register during card program load operations. This DC1 code is ignored by the reader, therefore, it is not inhibited. The timing shown in Fig. CON. 6 may, therefore,

be used directly by substituting F1LPCE for F1LPTE and by realizing that G1LU17, 15 is enabled while F1LPL1 is set and memory release (MD1MRLS) is false.

The card reader used for program load is the first/primary reader. An entire minimum loader card will be read in when the computer is switched to the AUTOmatic mode. The data contained by the card will be stored in 63 consecutive locations beginning at memory location 1, as with the paper tape minimum loader. Only two characters will be stored in each word, however, from the card.

Bulk Program Load Operation

The bulk program load operation is used to transfer 100_8 (64_{10}) words beginning at location 0 of the primary bulk memory to core memory beginning at location 0. The bulk memory (drum or disk) uses core memory location 0 for the pointer word address.

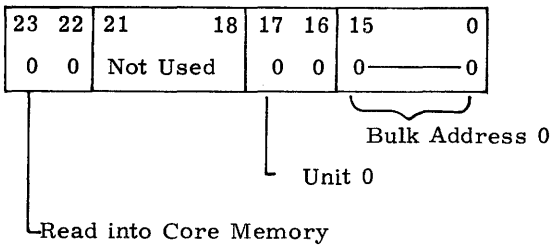
Fig. CON. 7 contains a flowchart of the basic functions performed by the program load circuitry to transfer a program from bulk memory to core memory. As shown in the flowchart, the following functions are performed in the listed sequence.

1. Pressing the ON switch to initialize the computer clears the B, P, and I Registers.

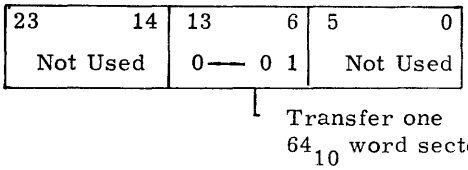
2. The P Register is saved (i.e. held to 0).

3. Interrupt response addresses 203_8 and above are inhibited.

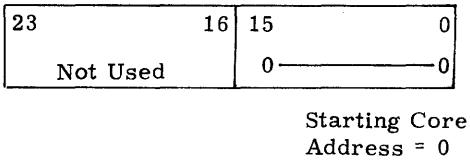
4. Zeros are stored in location 0. Since this is the memory location accessed by the bulk memory to determine the address of the first pointer word, location 0 will also be used as the first pointer word. Since location 0 contains all zeros, this first pointer word will specify a read into core operation from bulk address zero.



5. 00000100_8 is stored in memory location 1. This will be used as the second pointer word indicating that 64 words are to be transferred.



6. Zeros are stored in location 2. This will be used as the third pointer word and specify the starting core address as location 0. Therefore, the 64_{10} data words transferred from bulk unit 0 and Address 0 will be stored beginning in core memory location 0.



7. The B Register is set to 25041000 for an OUT command to the bulk memory device.

8. Placing the computer in the AUTO mode will cause the OUT command to be executed and the subsequent data transfer from bulk memory to core memory.

9. After the data transfer has occured, raising the PROG LOAD BULK switch will remove the program load inhibit of locations 203_8 and above and remove the SAVE P function. Therefore, program control goes to the program transferred from bulk memory.

A timing diagram with logic equations of the primary logic elements used to perform the program load function for bulk memory is shown in Fig. CON. 8. Since the timing and primary functions of each logic element is shown on the timing diagram, only a general discussion of these signals is provided in the following text.

F1LPDE is set when the PROG LOAD - BULK switch is placed in the down position. F1LPDE remains set until the BULK switch is raised.

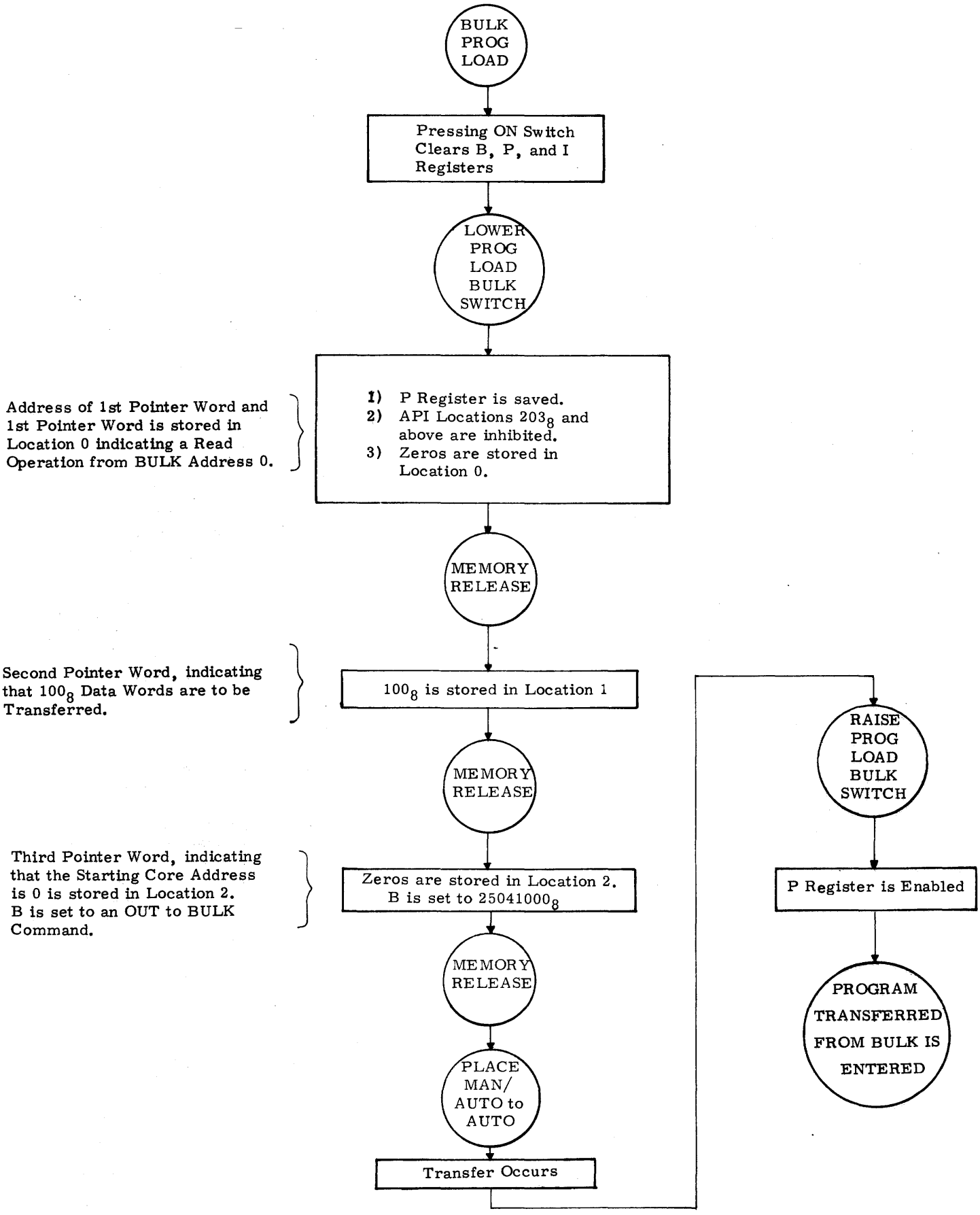


Fig. CON. 7 Program Load, Bulk, Flowchart

D0LAP1, 2 when enabled, applies a signal to the API module to inhibit interrupts to response addresses 204_8 and above. This is required so that interrupts (e.g. real time clock) will not be recognized.

G1LPA3 when enabled, inhibits interrupts to response address 203_8.

G0LSPS when enabled, performs the same function as the SAVE P switch. Therefore, since P was cleared by pressing the ON switch, P will be held equal to 0 during the Bulk Transfer operation.

G1SMRQ when enabled, requests access to memory and consequently initiates a memory operation. Three requests to memory are required to store the pointer words in locations 0, 1, and 2. Zeros are stored in location 0. Location 0 will then specify that the first pointer word is located in location 0, specify that the bulk operation is read and that the starting bulk address is 0, and provide a LDA command while the P Register is held at 0.

G0STOR when enabled, enables the store rather than read operation of the core memory. Therefore, data is stored in locations 0, 1, and 2.

G0LTDA when enabled, applies a "1" to bit 5 of the B Register via the Parallel Adder Unit. This generates 100_8 for storage in memory location 1.

G0LM00 when enabled, address memory location 1.

G0LM01 when enabled, addresses memory location 2.

G1LU22, 20, 18, 14, 9 when enabled, sets the B Register equal to 25041000 via the Parallel Adder Unit. This OUT command will be executed when the computer is placed in the AUTOmatic mode to initiate the bulk memory operation.

CONSOLE PROCEDURES

The following procedures are provided to assist the operator of the Programming and Maintenance Console. To use these procedures, the Console must be enabled by the key switch.

Enter Data in A or B Register

1. Place MAN - AUTO toggle switch to MAN.
2. Select desired Register by pressing A or B Register Select switch.
3. Press the CLEAR Reg. switch.
4. Press the Console Bit switches for the desired bit pattern.

Store a Word in Core Memory Location Y

1. Place MAN - AUTO toggle switch to MAN.
2. Press A Register Select switch.
3. Press CLEAR Reg. switch.
4. Press the Console Bit pushbuttons for desired bit pattern.
5. Press B Register Select switch.
6. Press CLEAR Reg. switch.
7. Using Console Bit switches, set a STA (320Y) in the B Register.
8. Press STEP switch.

Display the Contents of Memory Location Y

1. Place MAN - AUTO toggle switch to MAN.
2. Press B Register Select Switch.
3. Press CLEAR Reg. Switch.
4. Press the Console Bit Switches to set a LDA (000Y) into the B Register.
5. Press STEP switch.
6. Press A Register Select Switch. The contents of the addressed core cell will be displayed by the indicators.

To Store a Constant in All Core Memory Locations

1. Place MAN-AUTO toggle switch to MAN.
2. Press B Register Select switch.
3. Press CLEAR Reg. switch.
4. Press Console Bit switches for 07100001 bit pattern (LXK 1, 1).
5. Press STEP switch.
6. Press CLEAR Reg. switch.
7. Press Console Bit switches for 3210001 bit pattern (STA 1, 1).
8. Press A Register Select switch.
9. Press CLEAR Reg. switch.
10. Press Console Bit switches for the desired constant.
11. Press STEP switch.
12. Place SAVE I switch in save position.

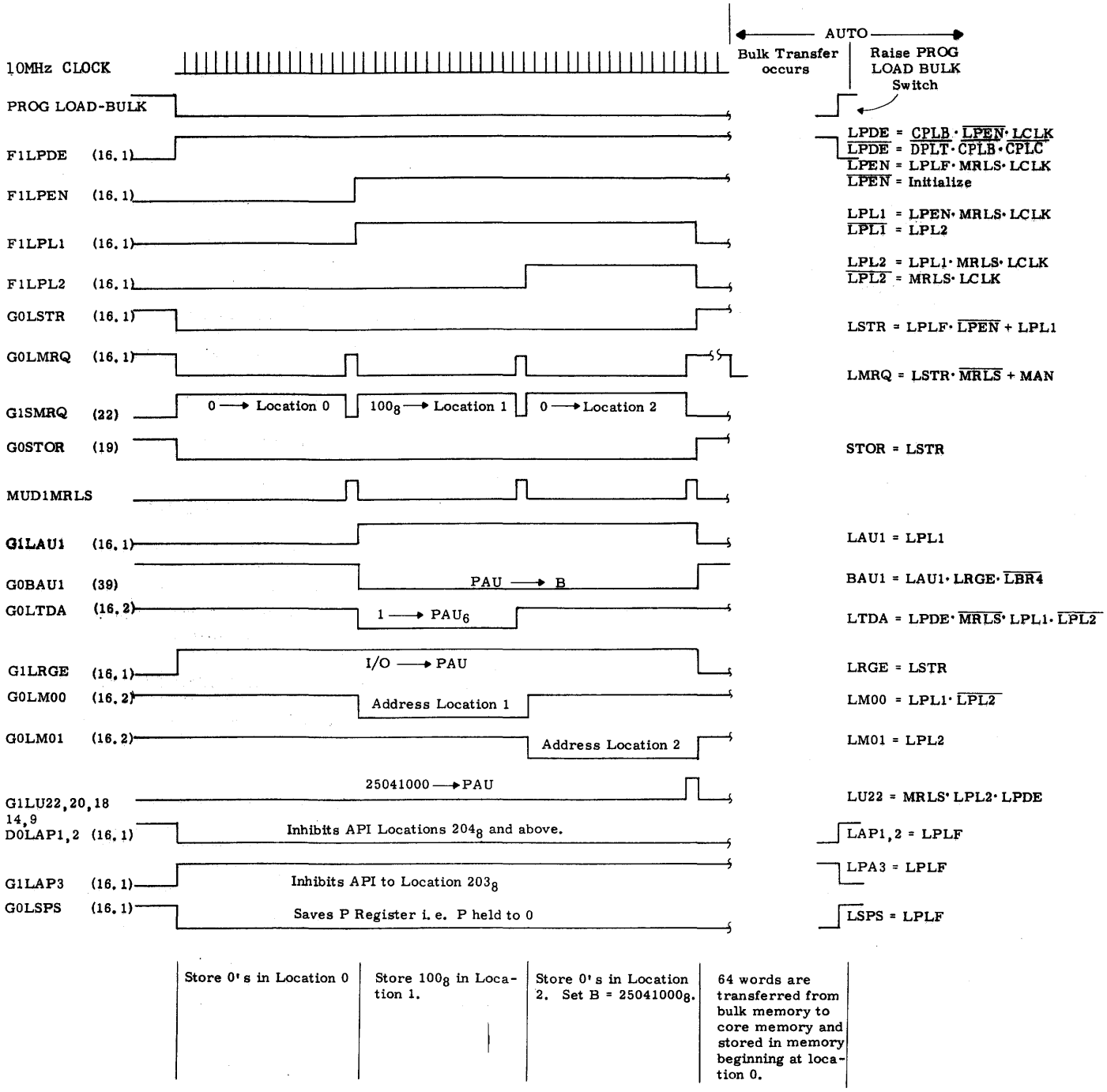


Fig. CON. 8 Bulk Program Load, Timing Diagram

13. Place MAN-AUTO toggle switch to AUTO.

14. The constant will now be stored in each memory location.

15. Place MAN-AUTO switch in MAN.

16. Lower SAVE I switch.

To Hand Load Instructions into Sequential Core Location Y, Y + 1, Y + 2, ... (Y ≥ 2).

1. Place MAN-AUTO toggle switch to MAN.

2. Press B Register Select switch.

3. Press CLEAR Reg. switch.

4. Press Console Bit switches for 0710001 (LXK 1, 1) bit pattern.

5. Press STEP switch.

6. Press CLEAR Reg. switch.

7. Press Console Bit switches for 321XXXXX (STA Y-1, 1) where XXXXX = Y - 1.

8. Press A Register Select switch.

9. Press CLEAR Reg. switch.

10. Press Console Bit switches to bit configuration of first instruction to be stored.

11. Press STEP switch.

12. Place SAVE I switch in save position.

13. For each instruction to be stored:

    a. Press CLEAR Reg. switch.

    b. Enter instruction in Console Bit Switches.

    c. Press STEP switch.

    The I Register contains the address of the last location stored.

Manually Display the Contents of Sequential Memory Locations Y, Y + 1, Y + 2, ... (Y ≥ 2).

1. Place MAN-AUTO toggle switch to MAN.

2. Press B Register Select switch.

3. Press CLEAR Reg. switch.

4. Press Console Bit switches for 07100001 (LXK 1, 1) bit pattern.

5. Press STEP switch.

6. Press CLEAR Reg. switch.

7. Press Console Bit switches for 001XXXXX (LDA Y - 1, 1) where XXXXX = Y - 1.

8. Press STEP switch.

9. Place SAVE I switch in save position.

10. Press A Register select switch. Contents of selected memory location is displayed by Register Display indicators.

11. Press STEP switch for each sequential location display. The I Register contains the address of the location being displayed.

To Load a Card or Paper Tape Minimum Loader Using Program Load.

1. Place computer in MANUAL mode.

2. Ready the reader.

    a. Paper Tape

    Place Minimum Loader tape in the reader so that leader is over the read heads.

    b. Card

    Place Minimum Loader card, either by itself or followed by files to be loaded, in the hopper and press the READY button.

3. Raise the API ENBL switch.

4. Lower PROG LOAD CARD or PROG LOAD TAPE switch.

5. Press the B-register select switch.

6. Enter the last digit of the device address in bits 0-2 if other than 0.

7. Place computer in AUTO mode. The minimum loader is read in.

8. Raise LOAD CARD or LOAD TAPE switch.

Operator Communication with Minimum Loader

Upon entry or following the detection of an end-of-file record, the Minimum Loader goes into a DEMAND loop awaiting operator communication.

When used with a reader whose device address differs from that assembled into the Minimum Loader card/tape (device address assembled is printed on the card/tape), the correct address must be entered into Location 3:

1. Place computer in MANUAL.
2. Place the SAVE P switch in the up (save) position.
3. Enter 0730XXX into B-register, where XXXX is the device address.
4. Place the SAVE P switch in the down (non-save) position.
5. Place computer in AUTO.

To Load the Program File

1. Ready the reader.
2. Enter the relocation constant into the console switches. If none is desired, raise all switches.
3. Push DEMAND.

   The program file will now load.

   Additional program files may be loaded using these same three steps.

Error Conditions and Recovery

1. In the event of a device error detectable by the JDR instruction, the Minimum Loader will hang up in a JDR loop. The I/O BUFFER ALARM light may or may not light. To recover:

   a. Correct the device problem.

   b. Backspace the medium at least one record (at least two feet, if paper tape).

   c. Manually branch to beginning of Minimum Loader +4 if cards, or beginning of Minimum Loader +6 if tape.

   d. Place computer in AUTO.

      The Minimum Loader rereads the relocation constant and continues loading.

2. In the event of a parity error (paper tape only), the I/O BUFFER ALARM light is lit and the Minimum Loader halts in a DEMAND loop.

   a. To ignore the error and continue, press DEMAND.

   b. To reread the record:

      Backspace the tape at least two feet.

      Press DEMAND.

   In both cases, the Minimum Loader rereads the relocation constant and continues loading.

Software Bootstrap

> NOTE
>
> These Software Bootstrap Procedures may be used to load a Minimum Loader if TIM or Program Load is not operating properly.

1. Ready reader.
2. Raise API ENBL switch to lock out API's.
3. Hand load the appropriate software bootstrap program.

Paper Tape Software Bootstrap

Location		
00007	320YYYYY	STA Y
00010	07400003	LXK 3,4
00011	05000000	LDZ
00012	05004062	SRC 18
00013	32000000	STA 0
00014	2506XX4X	JDR DVAD
00015	14000014	BRU /14
00016	2505XX0X	IN DVAD
00017	05045006	RBK 6
00020	21000000	ORA 0
00021	06000004	DMT 4
00022	34000012	BTS /12
00023	26700001	INX 1,7
00024	14000007	BRU /7

NOTE: Y + 1 is the desired beginning address of the Minimum Loader. ($Y \geq 24_8$).

DVAD is the Device Address.

Card Reader Software Bootstrap

Location		
00007	320YYYYY	STA Y
00010	2506XX4X	JDR DVAD
00011	14000010	BRU /10
00012	2505XXXX	IN DVAD
00013	05004054	SRC 12
00014	32000000	STA 0
00015	04000010	XEC /10
00016	14000015	BRU /15
00017	04000012	XEC /12
00020	21000000	ORA 0
00021	2670001	INX 1,7
00022	1400007	BRU /7

NOTE: Y + 1 is the desired beginning address of the Minimum Loader. ($Y \geq 22_8$).

DVAD is the Device Address

4. Push ON button to initialize computer.

5. Branch to Location 7.

   a. Enter 14000007 into B-register.

   b. Push STEP.

6. Operate Reader

   a. Enter 00000021 into A-register (not required for cards).

   b. Enter 2502XXXX into B-register where XXXX is the device address.

7. Place computer in AUTO.

   The Card Minimum Loader will be loaded into core locations Y + 1 through Y + 50_8 (the tape loader is loaded into Locations Y + 1 through 60_8).

8. Manually branch to Location Y + 1.

   a. Place computer in MANUAL.

   b. Enter BRU Y + 1 into B-register.

   c. Place computer in AUTO.

   Program control goes to Location Y + 1. The Minimum Loader will pause in a DEMAND loop waiting operator communication.

CONTROL OR INDICATOR	ACTIVE IN MODES MAN, AUTO OR CONSOLE OFF	PUSHBUTTON OR TOGGLE SWITCH	INDICATOR	DESCRIPTION
API STALL (DRCD00 - Sh. 130)	ALL MODES		X	Comes on when API Watchdog timer times out. Extinguished by SPB executed because of trap, by initialize, or API ENBL switch in lockout position. Inhibits all API's and disables API Watchdog in the lockout position. Disabling the console via the keyswitch enables API and Watchdog. Refer to the API description in this book set.
API ENBL (SWCAPI - Sh. 125)	AUTO & MAN	T		
AUTO/MAN (SWCAUM - Sh. 125)	AUTO & MAN	T		In AUTO mode, computer sequencing enabled. Computer runs automatically. In MAN mode computer sequencing occurs each time STEP switch is depressed... For one instruction. Disabling the console via the keyswitch forces AUTO mode. Refer to Manual Mode Description.
ALARM CLEAR (DRCERR, DRCPBE) - Sh. 125)	ALL MODES		X	Turned on by any of the following alarm conditions: 1. Overtemperature or blower airflow alarm 2. Watchdog trap 3. Stall alarm 4. Core parity error 5. Core temperature alarm 6. Bulk memory alarm 7. I/O alarm
ALARM CLEAR (SWCEER - Sh. 125)	MAN	PB		This switch, when pressed, clears the core parity error flip-flop, errors or alarms in the bulk memory system, and errors or alarms in the peripheral and I/O subsystems
CAB TEMP (DRCD08 - Sh. 130)	ALL MODES		X	This indicator is lighted when the temperature of the logic power supply, memory power supply, or auxiliary memory power regulator is approaching its upper limit or when the airflow from any of the blowers in the CSU has declined to a dangerous level. This indication serves only as a warning, if the temperature continues to rise to the trip level, DC power will be shutdown.
I ≠ INST (DRCD19 - Sh. 130)	ALL MODES		X	Comes on during execution of Multiply/Divide or TIM/TOM operation to indicate I Register does not contain last instruction following the execution of these commands. Cleared when next instruction is executed.
I/O BUF (DRCD02 - Sh. 130)	ALL MODES		X	Comes on when a parity or timing error occurs in an I/O subsystem. Cleared by JNE or the Alarm Clear switch.

Continued on next page.

Table CON. 1 Controls and Indicators

Table CON. 1. (Cont.)

CONTROL OR INDICATOR	ACTIVE IN MODES MAN, AUTO OR CONSOLE OFF	PUSHBUTTON OR TOGGLE SWITCH	INDICATOR	DESCRIPTION
$J_4 - J_0$ (DRCD13-09 - Sh. 130)	ALL MODES		X	Displays the contents of the J Counter. The J Counter is affected by GEN 1, GEN 3, MPY, DVD, and TIM/TOM operations.
M/P ENBL (SWCMEN - Sh. 125)	AUTO & MAN	T		Disables Memory Protect when in lockout position. When in the ENBL position Memory Protect is active, if TMFF is set.
MPLS CH. 1, (DRCD05 - Sh. 130)	ALL MODES		X	MPLX CH. 1 comes on when an error is detected by the bulk memory controller. Turned off by initialize, CLEAR ALARM, or OUT command. MPLX CH. 2, 3 are not used.
MPLS CH. 2, 3 (DRCD04, 3 - Sh. 130)			X	Comes on when an error is detected in the unassigned direct memory access user devices. Refer to the description of these devices for details.
OVFL (DRCD23 - Sh. 130)	ALL MODES		X	Comes on when arithmetic overflow occurs. Turned off by initialize, JNO or LPR commands.
OFF (SWCOFF - Sh. 125)	ALL MODES	PB		Removes DC power from CSU. Refer to Power Distribution section of this book set.
ON/INIT (SWCSON - Sh. 125)	MAN	PB		Applies DC power and initializes the system. Initialize clears B, P, I Reg., S_{2-5}, Demand, PAI, all Alarms, and priority interrupt flip-flops. Initialize sets S1 and enables B Reg. Select. Refer to the Power Distribution description of the book set.
PAI (DRCD22 - Sh. 130)	ALL MODES		X	Indicates the status of the Permit Automatic Interrupt flip-flop (F1WPMT). When lighted, F1WPMT is set permitting inhibitable interrupts. The light is extinguished when F1WPMT is reset inhibiting inhibitable interrupts. F1WPMT is controlled by PAI, IAI, SPB, LPR, and LDP commands.
PROG LOAD TAPE-CARD-BULK (SWCPLT, C, B - Sh. 125)	MAN & AUTO	T		These switches permit the operator to bring a loader program into core memory from the primary bulk memory, paper tape, or cards with a minimum of manual effort. Refer to Program Load discussion.
Reg. Select A, B, I, P (SWCSLA, B, I, P - Sh. 126)	MAN	PB	X	Each switch selects its associated register. Used in conjunction with the register entry and display switch lights. All Registers may be displayed but only A & B may receive data from the Reg. Entry switches. Activating any one of the Reg. Select Switches disables the others. Refer to Register Select discussions.
Console Bit Switches /Indicator (SWCB00-23 - Sh. 127)	MAN	PB	X	The Switches are used in conjunction with the Reg. Select Switch to enter data into A or B Register. The lights will display the contents of the register selected (I, P, B, or A) by the Register Select Switch. Refer to Register Select Discussions.

Continued on next page.

Table CON. 1 Controls and Indicators

Table CON. 1. (Cont.)

CONTROL OR INDICATOR	ACTIVE IN MODES MAN, AUTO OR CONSOLE OFF	PUSHBUTTON OR TOGGLE SWITCH	INDICATOR	DESCRIPTION
CLEAR REG. (SWCCLR - Sh. 126)	MAN	PB		Will clear the A or B Register as selected by the Reg. Select Switches. Refer to Clear Register description.
ON (DRCRDY - Sh. 125)	ALL MODES		X	Comes on when power is applied. Turns off when power is removed. Refer to the Power Distribution section of this book set for details.
CORE PRTY (DRCD06 - Sh. 130)	ALL MODES		X	Comes on when a memory parity error is detected, CLEAR ALARM resets Parity Error FF and turns off light. JNP and initialize also clears the flip-flop and turns off the light. If the STOP ON PARITY switch is in the stop position, memory operation is halted.
CORE TEMP (DRCD07 - Sh. 130)	ALL MODES		X	Comes on when core stack temperature is not within allowable range. Turns off when temperature is within allowable range. If parity error occurs during out of limits time memory sequencing is inhibited.
CONSOLE ENBL (SWCSEN - Sh. 125)	ALL MODES	KEY		When in the disable position only the DMD, RCS, and OFF switches are active. The indicators operate independent of the CONSOLE ENBL switch.
DMD (SWCDMD - Sh. 125)	ALL MODES	PB		Sets Demand Flip-Flop when depressed and released. Demand Flip-Flop is cleared and light turned off by JND or initialize. Refer to JND command description.
DMD (DRCDMD - Sh. 125)	ALL MODES		X	Lighted when demand flip-flop (F1CDMN) is set. Demand flip-flop is set by DMD switch and cleared by JND command or initialize.
Console Data Switches (Read Console Sw's) (SWCC00-23 - Sh. 129)	ALL MODES	T		Used in conjunction with the RCS command to enter the contents of these 24 switches into the A Register. The up position represents a 0, while the down position equals a 1. Refer to the RCS command descriptions.
SAVE P (SWCSSP - Sh. 125)	AUTO & MAN	T		When in the up position the P Register cannot be altered. When in the down position, P operates normally. CONSOLE ENBL disables the SAVE P function when in the disable position.
SAVE I (SWCSSI - Sh. 126)	AUTO & MAN	T		When in the up position the I Register cannot be altered during the instruction fetch cycle. This feature is used primarily to save the OP code and index portion of an instruction so that the instruction may be repeated. In the down position I operates normally. CONSOLE ENBL disables the SAVE I function when in the disable position.

Continued on next page.

Table CON. 1 Controls and Indicators

Table CON. 1. (Cont.)

CONTROL OR INDICATOR	ACTIVE IN MODES MAN, AUTO OR CONSOLE OFF	PUSHBUTTON OR TOGGLE SWITCH	INDICATOR	DESCRIPTION
STEP (SWCSTP - Sh. 126)	MAN	PB		In the MAN mode, each depression and release causes the computer to execute one instruction. Refer to the Manual Mode description.
S_1 - S_5 (DRCD18-14 - Sh. 130)	ALL MODES		X	These indicators display the status of the 5 sequence states of the AU.
STALL (DRCD01 - Sh. 130)	ALL MODES		X	Comes on when Stall Alarm timer times out. May be cleared by Initialize, or Stall ENBL Held off by the SSA instruction
STALL ENBL (SWCSTL - Sh. 125)	AUTO & MAN	T		When in the disable position, the Stall Alarm and Watchdog Timer is disabled. CONSOLE ENBL enables Stall Alarm and Watchdog Timer when in the disable position. Refer to the Stall Alarm description.
TRAP MODE (DRCD20 - Sh. 130)	ALL MODES		X	Comes on when trapping mode flip-flop is set by STMF or by a LDP or LPR with bit 19=1. The trapping mode flip-flop is reset extinquishing the indicator following a trap error or by placing the M/P Enable switch in the lockout position.
TEST (DRCD21 - Sh. 130)	ALL MODES		X	Comes on when Test flip-flop is set. The Test flip-flop can be set by: DMT, TXH, some GEN 1 commands and LPR. The Test flip-flop may be reset by DMT, TXH, some GEN 1 instructions, LPR, and Initialize. Refer to these command descriptions.

Table CON. 1 Controls and Indicators

COMMANDS

The following discussion is provided to assist the reader in a more thorough understanding of the individual command descriptions that follow.

COMMAND FORMATS

The commands, used by the GE-PAC 4022 Arithmetic Unit fall into five categories: Full Operand, GEN 1, GEN 2, GEN 3, and Quasi. The microcoded format of each of these command types is shown in Fig. CMD. 1.

Full Operand

The Full Operand commands are the most widely used commands of the Arithmetic Unit. These commands are used to perform arithmetic operations, logical operations, index control operation, and data transfers to and from memory. These commands designate the core address for information in memory as an operand address. The operand address is a full 14 bits (13-0), which may be augmented by relative addressing (14) and/or indexing (17-15).

A detailed discussion of each of these commands is provided in alphabetical order. Each command description provides the mnemonic associated with the command, the command type, format, and a description of the command usage. In addition, a specific hardware oriented discussion is provided including a block diagram of the registers used, and a timing diagram including logic equations, of the more important control signals enabled within the Arithmetic Unit.

GEN 1

GEN 1 commands are differentiated from other commands by the OP Code (bits 23 through 18) 05_8. GEN 1 commands are further subdivided into commands by the microcoding of the operation portion (bits 14 through 0) of the command word. GEN 1 commands are used primarily for bit manipulation of the A Register, but may be microcoded to affect the J Counter and Test flip-flop.

Thirty-nine different GEN 1 commands are described under the mnemonic GEN 1. Each description contains the microcoded format of the command, a description of the command usage, and a description of the hardware operation when executing the command.

GEN 2

GEN 2 commands are differentiated from other commands by the OP Code (bits 23 through 18) 25_8. GEN 2 commands are also subdivided into commands by the microcoding of the operation portion (bits 14 through 0) of the command word. These commands are employed by the GE-PAC system to: (1) select modules and devices in the input/output equipment, (2) transfer data to or from these devices, and (3) provide for program control transfers (jump) as determined by various internal and external conditions.

Twenty-three different GEN 2 commands are described under the mnemonic GEN 2. Each description contains the microcoded format, discussion of the command usage, and a detailed discussion of the hardware operation when executing the command.

GEN 3

GEN 3 commands are differentiated from other commands by the OP Code 45_8. GEN 3 commands are also subdivided into commands by the microcoding of the operation portion of the command word. These commands are used to manipulate the contents of the A and Q Registers, affect the J Counter, and optionally affect the Overflow flip-flop.

Eight individual GEN 3 commands are described in detail under the mnemonic GEN 3.

Quasi

Quasi commands provide operations not included in the hardware. These commands supply the programmer with a mnemonic which allows the running program to be linked with a subroutine. Quasi commands are identified by OP Codes 40_8 - 77_8 if the command is not an MPY, DVD, LDQ, STQ, or GEN 3.

The Quasi command word is divided into four areas: bits 23 through 18, the next command address; bits 17 through 15, index address modification; bit 14, relative addressing; and, 13 through 0, the operand.

	23 22 21 20 19 18		17 16 15	14	13 12 11 10 9 8 7 6 5 4 3 2 1 0
FULL OPERAND	OP CODE		INDEXING ADDRESS	*	OPERAND (Y)
GEN 1	0	5	INDEXING ADDRESS	C	A B T S K
GEN 2	2	5	INDEXING ADDRESS	S	K3 K2 K1 K0
GEN 3	4	5	INDEXING ADDRESS	SEE GEN 3 DESCRIPTION	K
QUASI	40_8 - 77_8 EXCEPT MPY, DVD, LDQ, STQ, OR GEN 3		INDEXING ADDRESS	*	OPERAND (Y)

*Relative Address Bit

Fig. CMD. 1 Command Formats

When a Quasi command is "fetched" from memory, indexing and/or relative address modification of the operand portion of the command is performed if specified. After modification, if required, the operand portion of the Quasi command is stored in index cell 2. The next instruction is then addressed from the OP Code portion of the Quasi command. Therefore, program control is transferred to a memory location between 40_8 and 77_8. This location will normally contain an SPB command to a subroutine to accomplish the operation. This subroutine may, of course, use the contents of index cell (operand portion of the Quasi command) as an operand address or data value.

A detailed description of the Quasi command is contained under the mnemonic QSI.

BASIC TIMING OF FULL OPERAND COMMANDS

The following paragraphs describe the basic Arithmetic Unit timing for full operand commands. The basic timing for GEN 1, GEN 2, GEN 3, TIM/TOM, and the execution states of MPY and DVD are described with the command description of these commands.

Because the basic timing is identical for all Sequence States of most commands, the command descriptions that follow this section do not include this timing except where differences exist.

The basic clock signal used throughout the 4010 system is generated from a 10 (±.01%) MHz crystal controlled oscillator (sheet 7). The output of the oscillator is applied through two variable 15 to 35 nanosecond single shot circuits. Therefore, out of the single shot, 15 to 35 nanosecond pulses occurring at a 10 MHz rate are applied through clock drivers to the system. The single shots are normally adjusted to provide a pulse width of approximately 27 nanoseconds.

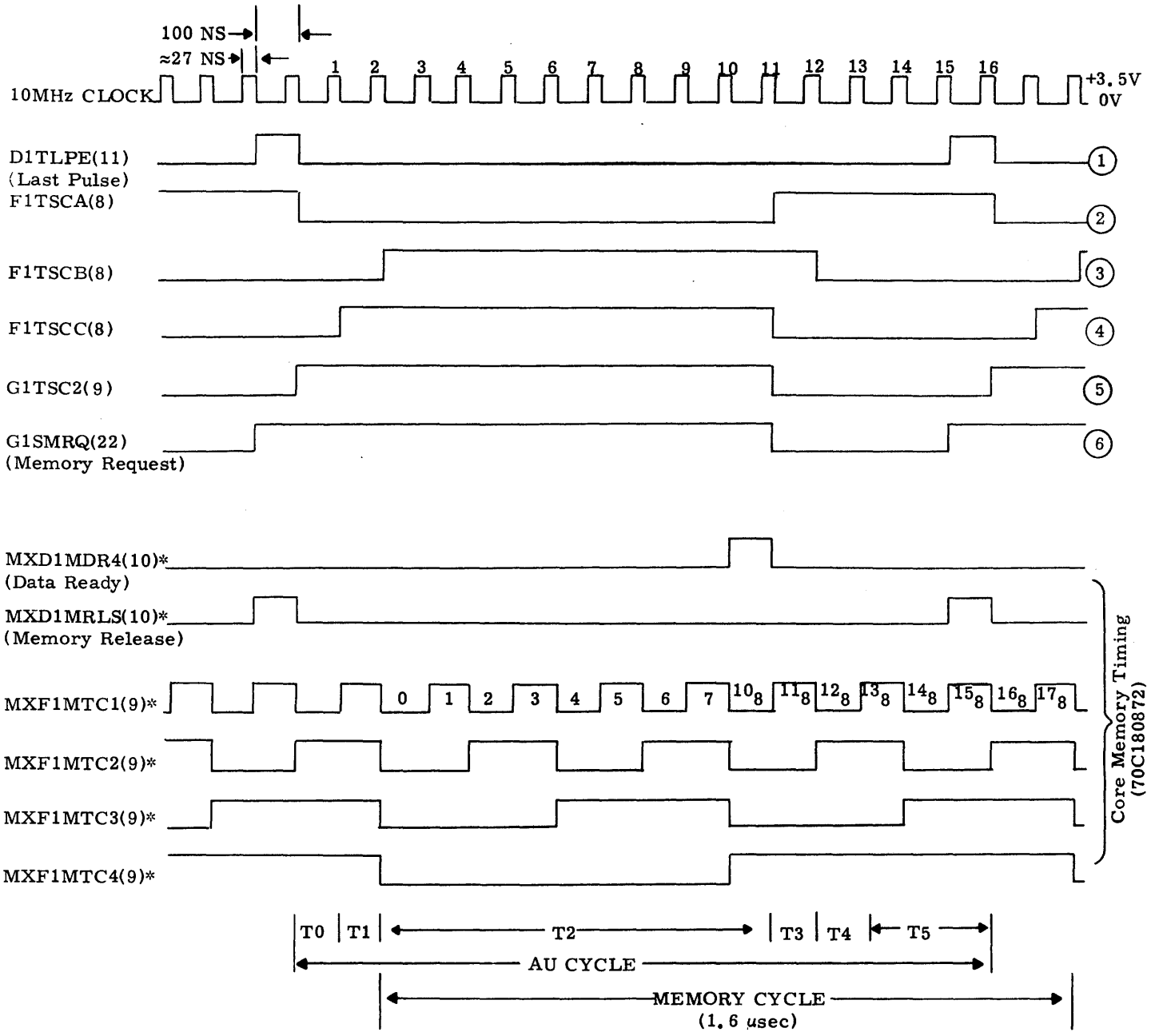
Sequential timing within the Arithmetic Unit is controlled by a 3-stage gray code counter (F1TSCA, F1TSCB, and F1TSCC). The condition of these flip-flops (set or clear) are used to specify specific time intervals to perform various functions. Six different time intervals are specified by this counter during the execution of full operand commands. For ease of description, the time intervals are called Time 0 Envelope through Time 5 Envelope. The count sequence and corresponding time envelope nomenclature are:

F1TSCA	F1TSCB	F1TSCC	
0	0	0	Time 0 Envelope
0	0	1	Time 1 Envelope
0	1	1	Time 2 Envelope
1	1	1	Time 3 Envelope
1	1	0	Time 4 Envelope
1	0	0	Time 5 Envelope
1	0	1	Time 6 Envelope - This timing envelope is entered only during execution of MPY, DVD, TIM/TOM, and GEN commands. Refer to these command descriptions for details.

Fig. CMD. 2 contains a timing diagram and logic equations of the Sequence Time Counter. This timing diagram applies to all commands except the execution states of GEN 1, GEN 2, GEN 3, MPY, DVD, and TIM/TOM operations. It is drawn to illustrate the sequencing when operating with a 1.6 microsecond memory.

Last Pulse (D1TLPE) of any Sequence State or the Initialize signal (D0TINT) clears the gray code counter forming Time 0 Envelope. Clearing F1TSCA enables G1TSC2. With TSC2 enabled, the first clock pulse sets F1TSCC, forming Time 1 Envelope. The next clock pulse sets F1TSCB, forming Time 2 Envelope. Time 2 Envelope is enabled until Data Ready (MXD1MDR4) is received from the core memory module. The clock pulse during Data Ready sets F1TSCA, forming Time 3 Envelope. The next clock pulse clears F1TSCC, forming Time 4 Envelope. The next clock pulse clears F1TSCB, forming Time 5 Envelope. Time 5 Envelope is enabled until Last Pulse is generated. Last Pulse is generated from the Memory Release signal (MXD1MRLS), applied by the Core Memory module at the end of the memory timing cycle. Last Pulse Envelope clears the Sequence Time Counter for the next timing cycle. The duration of Time 0 through Time 5 is approximately 1.6 microseconds.

As described above, the duration of Time 2 Envelope and the duration of Time 5 Envelope is dependent upon the memory cycle. The memory cycle begins with Memory Request (G1SMRQ) when access to memory is granted by the memory priority scheme. Memory Re-



*Memory Control logic - 70C180872

① TLPE = TLP1 = MRLS · $\overline{\text{CMAN}}$ · TLPE

② $\underline{\text{TSCA}}$ = MDR4 · CTAE · TCK2
$\overline{\text{TSCA}}$ = TLPE · TCK2

③ $\underline{\text{TSCB}}$ = $\overline{\text{TSCA}}$ · TSCC · TCK2
$\overline{\text{TSCB}}$ = $\overline{\text{TSCC}}$ · TCK2

④ $\underline{\text{TSCC}}$ = TSC2 · $\overline{\text{TSCB}}$ · TCK2
$\overline{\text{TSCC}}$ = TSCA · TSCB · TCK2 + TLPE · TCK2

⑤ TSC2 = $\overline{\text{TSCA}}$

⑥ SMRQ = $\underline{\text{MRLS}}$ · ($\overline{\text{DG12}}$ + $\overline{\text{BC12}}$) · LMRQ · $\overline{\text{SRQ2}}$ ·
$\overline{\text{CMAN}}$ + SRQ1 · $\overline{\text{TSCA}}$ · $\overline{\text{CMAN}}$ · LMRQ

Fig. CMD. 2 Basic Timing Diagram, Full Operand Commands

quest is normally enabled by the Memory Release (MXD1MRLS) signal generated in the memory. If, however, the previous or next cycle does not require memory access (i.e., State 4 of GEN 1 or GEN 2 command, or when State 4 is extended for the execution of GEN 3, MPY, DVD, or TIM/TOM), the Memory Request signal is inhibited until Time 0 Envelope. Fig. CMD. 3 illustrates the timing relationships for both types of memory requests.

Instruction Sequencing

All instructions performed by the 4022C Arithmetic Unit follow a definite set pattern or sequence for "fetching" the instruction from memory, performing index address modification if required, and executing the instruction. Sequence Control State flip-flops F1SC01, F1SC02, F1SC03, F1SC04, and F1SC05 are provided to control this pattern or sequence.

Fig. CMD. 4 graphically illustrates the Sequence States required to fetch, index, and execute any instruction. Briefly, the function performed during each Sequence Control State is described below.

- Sequence Control State 1 - F1SC01: During this Sequence State, all commands are "fetched" from memory. Non-indexed XEC, BRU, BTR, and BTS commands are also executed during Sequence Control State 1.
- Sequence Control State 2 - F1SC02: During Sequence State 2 index address modification occurs. Also, STX, TXH, and DMT commands use State 2 for a portion of the execution.
- Sequence Control State 3 - F1SC03: State 3 is used during the execution of MPY, DVD, STQ, STX, and TIM/TOM operations.
- Sequence Control State 4 - F1SC04: State 4 is the execution state for most commands.
- Sequence Control State 5 - F1SC05: State 5 is used during MPY, DVD, GEN 3, STQ, LDQ, and TIM/TOM instructions to complete their execution cycle.

Sequence Control State 1

Sequence Control State 1 defines the "fetch" cycle for all instructions. Because State 1 is nearly the same for all commands, it is not described in the command descriptions later in this section unless it performs unique functions (e.g., XEC, BRU, TXH, BTR, and BTS). Therefore, the following discussion describes the detailed operation of the Arithmetic Unit during State 1. This discussion applies to all commands.

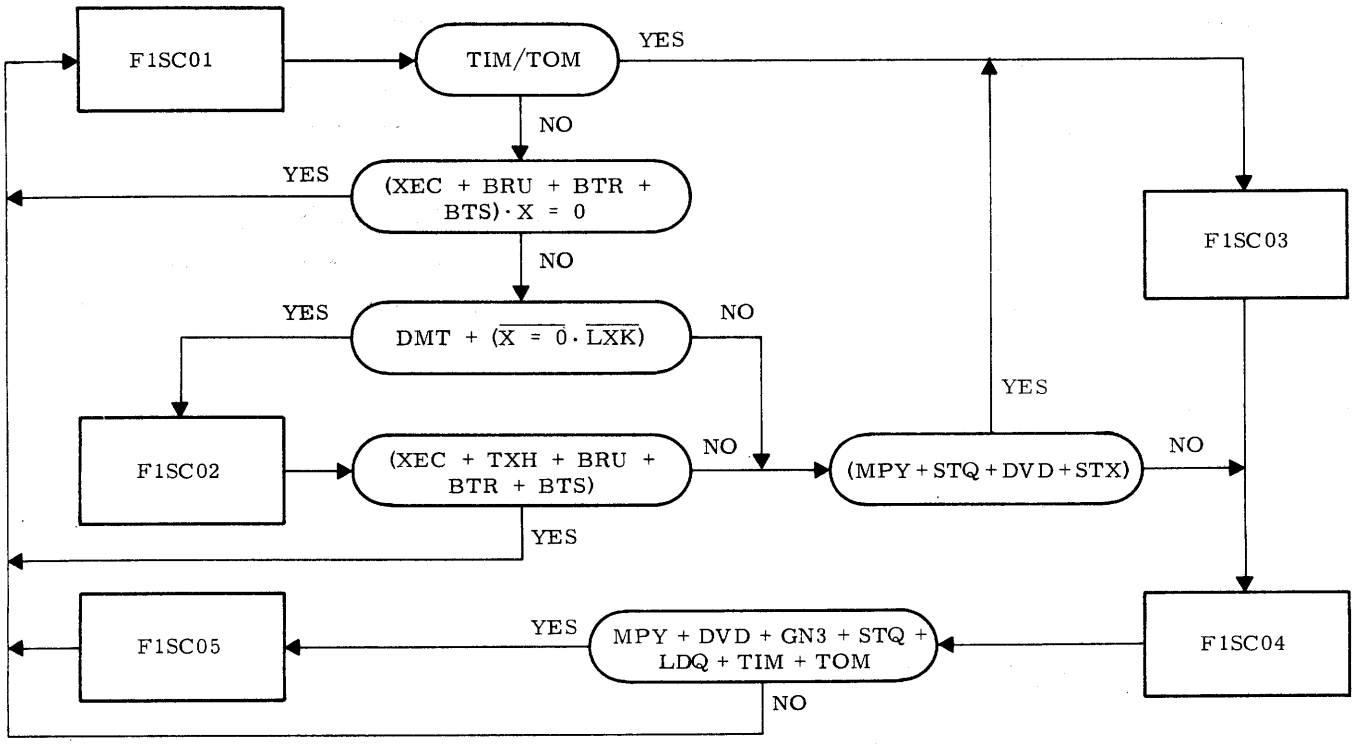
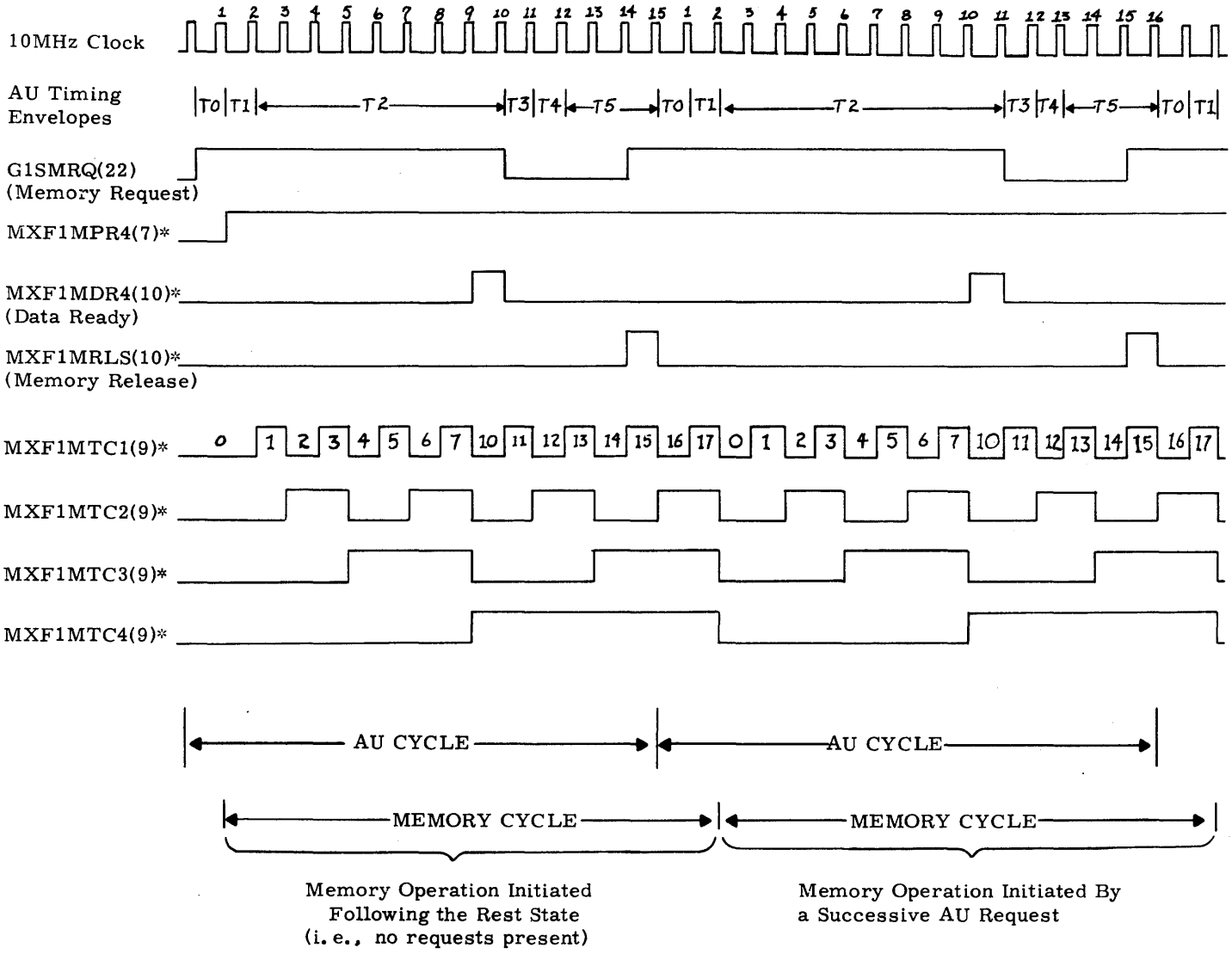


Fig. CMD. 4 Instruction Sequencing



*Memory Control Logic - 70C180872

Fig. CMD.3 AU/Memory Cycle Relationship

Fig. CMD. 5 contains a block diagram of the Arithmetic Unit operation during a normal Sequence State 1. A timing diagram and logic equations for State 1 are contained in Fig. CMD. 6. Refer to these aids during the following discussion.

During Sequence State 1, memory is always requested (G1SMRQ) to "fetch" the next command from memory. Memory is addressed from the P Register during State 1 except when following a branch instruction (SPB, BRU, BTS, BTR, LDP, or LPR), an XEC command, an Automatic Program Interrupt, a Memory Protect Error, or when a new Protect Status Word is required for the optional Memory Protect logic. Upon receipt of the Data Ready signal from the core memory module, the command located in the addressed memory location is gated to the B Register by D1BMEM. Bits 23 through 14 of the command are gated from B to the I Register (IBXI). This places the OP Code of the instruction fetched in the I Register where it is decoded and the operation to be performed is determined. The operand address portion of the command (bits 13-0) is gated to the I Register via the Adder Unit. If the command is relative addressed (bit 14 is a "one"), relative address modification of the operand address occurs in the Adder Unit prior to being transferred to the I Register. Relative Addressing is described later in this section.

At memory release, Last Pulse Envelope is enabled to end State 1. Sequencing will then continue for execution of the command as described under the mnemonic of the command.

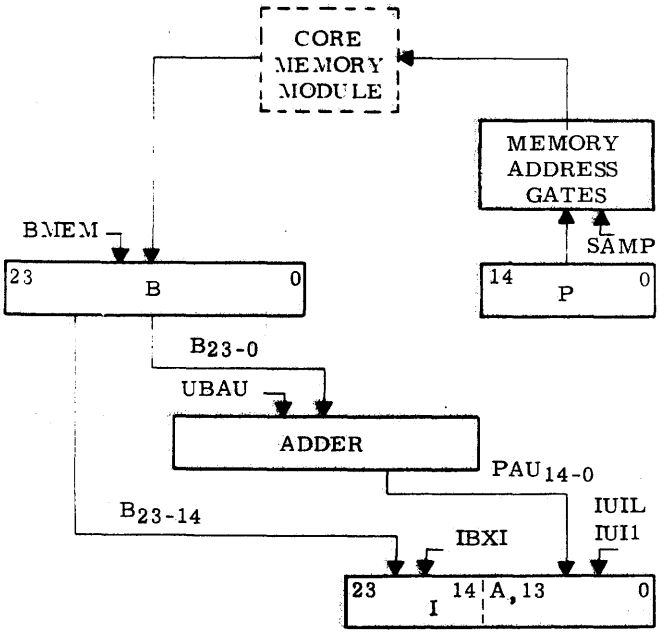
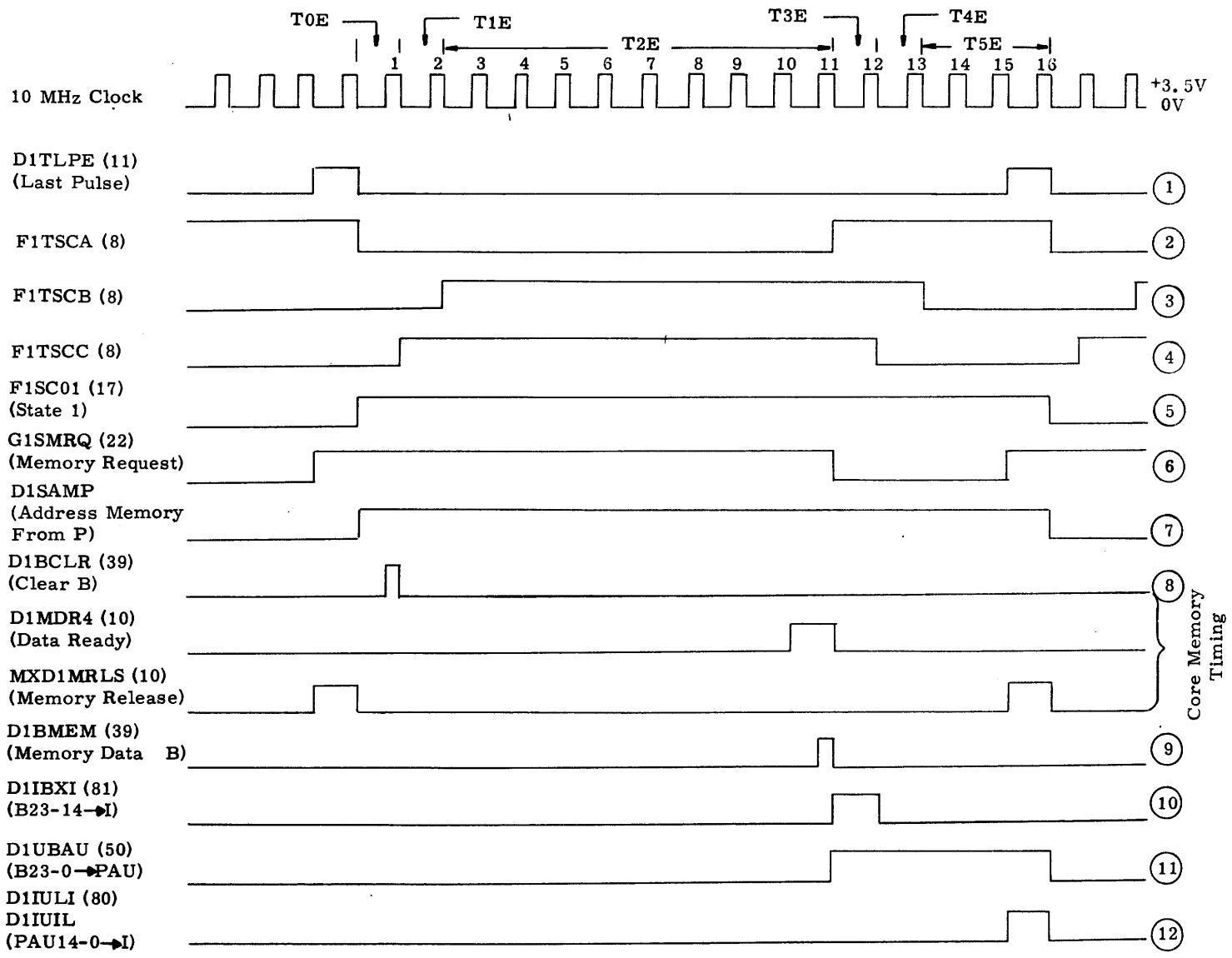


Fig. CMD. 5 State 1 Block Diagram



(1) TLPE = TLP1 = MRSL · $\overline{\text{CMAN}}$ · TLPE

(2) $\overline{\text{TSCA}}$ = MDR4 · CTAE · TCK2
$\overline{\text{TSCA}}$ = TLPE · TCK2

(3) $\overline{\text{TSCB}}$ = $\overline{\text{TSCA}}$ · TSCC · TCK2
$\overline{\text{TSCB}}$ = TSCC · TCK2

(4) $\overline{\text{TSCC}}$ = TSC2 · $\overline{\text{TSCB}}$ · TCK2
$\overline{\text{TSCC}}$ = TSCA · TSCB · TCK2 + TLPE · TCK2

(5) $\overline{\text{SC01}}$ = SSS1 · TLPE · SCLK
$\overline{\text{SC01}}$ = TLPE · (SR12 + SR14) · TSLK

(6) SMRQ = MRLS · $\overline{\text{DG12}}$ · $\overline{\text{SRQ2}}$ · $\overline{\text{CMAN}}$ · LMRQ +
SRQ1 · TSCA · $\overline{\text{CMAN}}$ · LMRQ

(7) SAMP = $\overline{\text{MAMV}}$ · $\overline{\text{MTRP}}$ · SC01 · $\overline{\text{XRMF}}$ · $\overline{\text{SEXC}}$ · $\overline{\text{SPI1}}$

(8) BCLR = BCL1 = $\overline{\text{STOR}}$ · $\overline{\text{TSCA}}$ · $\overline{\text{TSCC}}$ · BCLK · BMRQ

(9) BMEM = MDR4 · $\overline{\text{STOR}}$ · BMRQ · BCLK

(10) IBXI = $\overline{\text{ISAV}}$ · TSCB · $\overline{\text{HTTF}}$ · MSSI · SC01 · ISCA · TSCC

(11) UBAU = UBA3 = SC01 · TSCA · $\overline{\text{UBB1}}$ · $\overline{\text{UAMV}}$

(12) IULI, U = IUL1

Fig. CMD. 6 Sequence Control State 1 Timing Diagram

INDEXING

Indexing of a command involves the changing of the command operand by adding it to the contents of a designated X core cell (core cells 1 through 7). The GE-PAC command format allocates 3 bits (15, 16, and 17) to specify Index Address Modification.

When bits 15, 16, and 17 are not equal to zero, the 14 bits of the instruction operand (0 through 13) are added to the least significant 15 bits of the specified index register contents. The memory location (1 through 7) of the index register is specified by bits 15 through 17. The result of this addition allows the instruction to address one of $32,768_{10}$ memory locations.

To illustrate the effect of indexing address modification, refer to CMD. 7 and consider the following examples.

Example 1:

$P = 1000_8$

Location 1000_8 = 00500100_8

Index Register 5 = 00000040_8

Following execution of the command (LDA) in location 1000_8;

A Register = Contents of location 140_8

Location 1000_8 = 00500100_8

Index Reg. 5 = 00000040_8

Example 2:

$P = 2000_8$

Location 2000_8 = 32700100_8

Index Reg. 7 = 00040000_8

Following execution of the command (STA) in location 2000_8;

Location 40100_8 = Contents of the A Register

Location 2000_8 = 32700100_8

Index Reg. 7 = 00040000_8

Some commands use the index bits (15, 16, and 17) for instruction control rather than for index address modification. These commands are:

INX

LDX

LXC

LXK

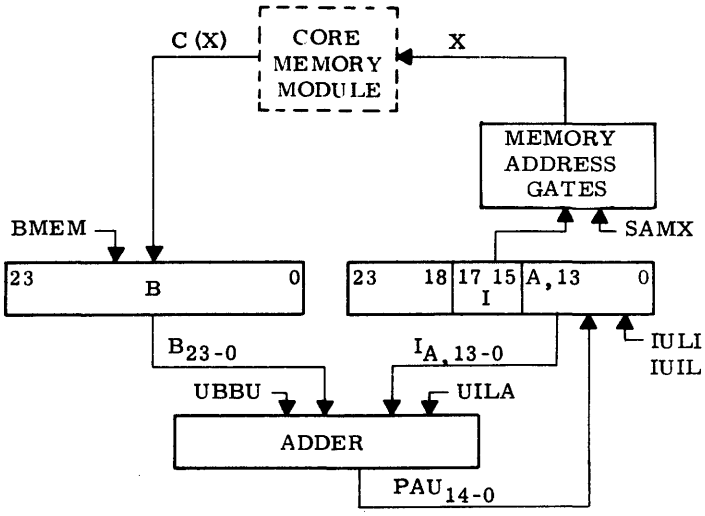
STX

TXH

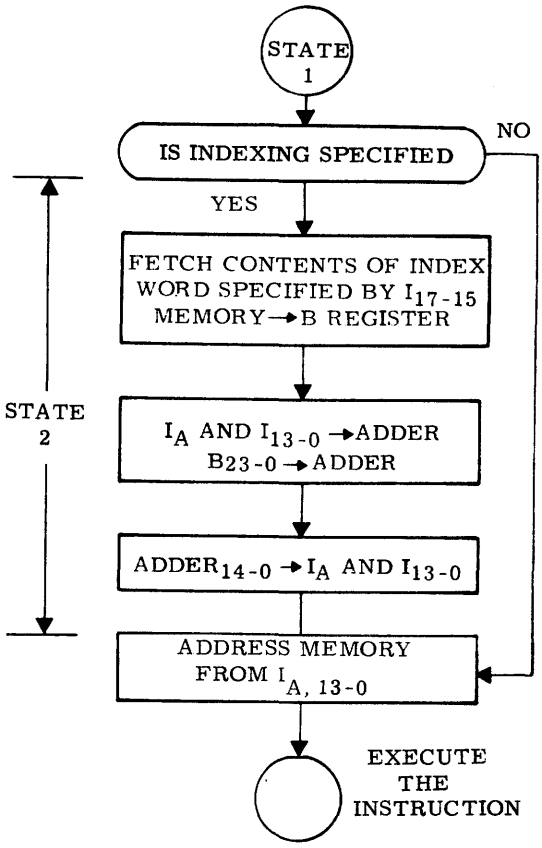
Refer to the command description for the use of the index bits for these commands.

Indexing of GEN 1, GEN 2, and GEN 3 commands may change the intelligence of the microcoding. Therefore, caution must be exercised when specifying index address modification of these commands.

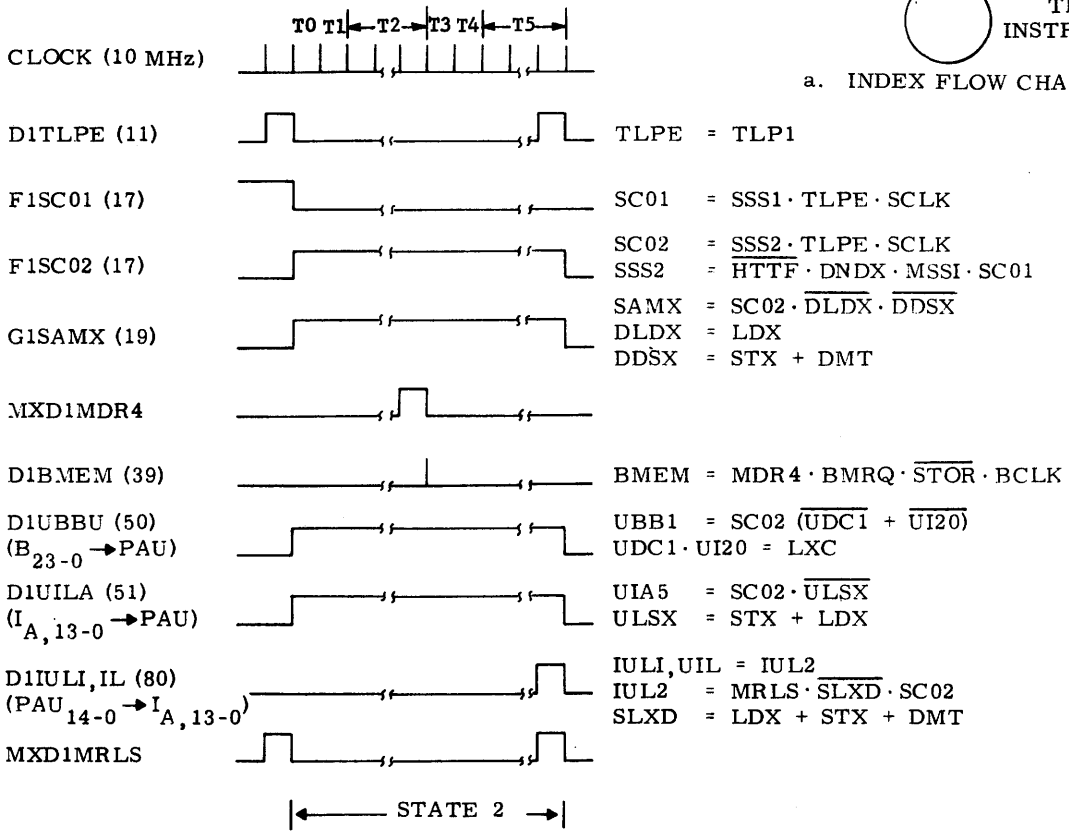
Indexing of any command requires the use of Sequence Control State 2. The indexed command is "fetched" during a normal Sequence Control State 1. Bits 15, 16, and 17 not being equal to zero causes sequencing to State 2. During State 2, memory is addressed from bits 15, 16, and 17 (G1SAMX) and the contents of the addressed index register are gated to the B Register. From B, all 24 bits of the addressed index register are gated to the Adder (D1UBBU). At the same time, $I_{A,\ 13-0}$ is gated to the Adder where summation with the index register contents occurs. The result of this summation (bits 14 through 0) is gated back to $I_{A,\ 13-0}$ (D1IULI, IL), to complete the indexing address modification operation. The next Sequence State would then be entered and memory addressed from the I Register to execute the indexed instruction.



b. INDEX BLOCK DIAGRAM



a. INDEX FLOW CHART



c. INDEX TIMING DIAGRAM

Fig. CMD. 7 Index Address Modification

RELATIVE ADDRESSING

Relative addressing modifies the operand (bits 13-0) portion of the command "fetched" from memory by adding the core cell address of the command to the operand bits. The core cell address of most commands is the bit information contained in the P Register. However, for the object instruction of XEC and Quasi (i.e., locations 40 through 77) commands, the bit information contained in $I_{A,\ 13-0}$ is the address of the command. Relative addressing of a command is specified by a "one" in bit 14 (*), of the command. All commands may be relative addressed with the exception of the following:

FIX

FLO

FMS

GEN 1

GEN 2

GEN 3

LXC

TXH

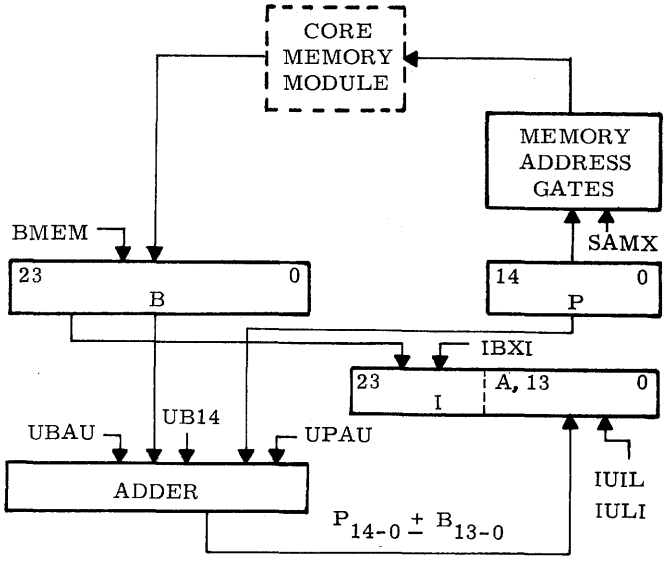
NOP

Relative addressing, as well as indexing and the normal incrementation of P, allows addressing of up to 32,768 memory locations. Relative addressing permits the program to address a memory location in the range of plus $8,191_{10}$ to minus $8,192_{10}$ with respect to the instruction address. The plus or minus direction is determined by bit position 13 of the instruction operand address. When bit 13 is a "one", the operand address is in 2's complement form and relative addressing occurs in the negative direction.

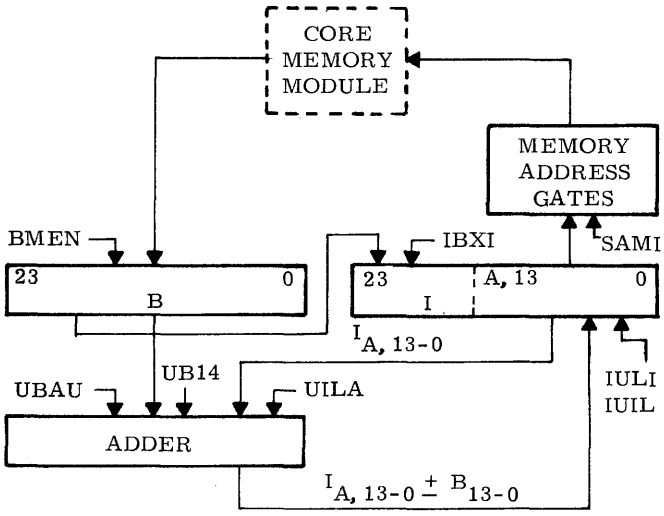
Refer to Fig. CMD. 8, 9, and 10. Relative addressing for all commands occurs during Sequence Control State 1. The relative addressed command is "fetched" from memory and gated to the B Register. The relative addressed command will normally be addressed from the P Register. However, if the relative addressed command is the object instruction of XEC or Quasi, it will be addressed from the I Register. Although BTS, BTR (with the jump condition true), and BRU commands address memory from I for the next instruction, the P Register will also contain the address of this next instruction and this value is used for relative addressing.

From the B Register, bits 23 through 14 of the relative addressed command are gated to the I Register for decoding. With bit 14 of the command a "one" and if the command is not a GEN, then IR14 will be set indicating that the command is relative addressed. B_{23-00} is gated to the Parallel Adder. Because it is relative addressed, B_{13} is also applied to bit 14 of the Adder (G1UB14). The contents of the P Register (or I Register for XEC and Quasi) are also gated to the Adder where the summation occurs. The result of this summation is then gated to $I_{A,\ 13-0}$ as the effective operand address of the command. The command is then executed in the normal manner using this effective operand address.

The examples shown in Fig. CMD. 11 illustrate the summation and the expansion of B_{13} to bit 14 of the Adder. Both positive and negative addressing is illustrated.



(1) All Relative Addressed Commands Except Quasi and XEC Object Instructions.



(2) Object Instruction of XEC or Quasi.

Fig. CMD. 8 Relative Addressing Block Diagram

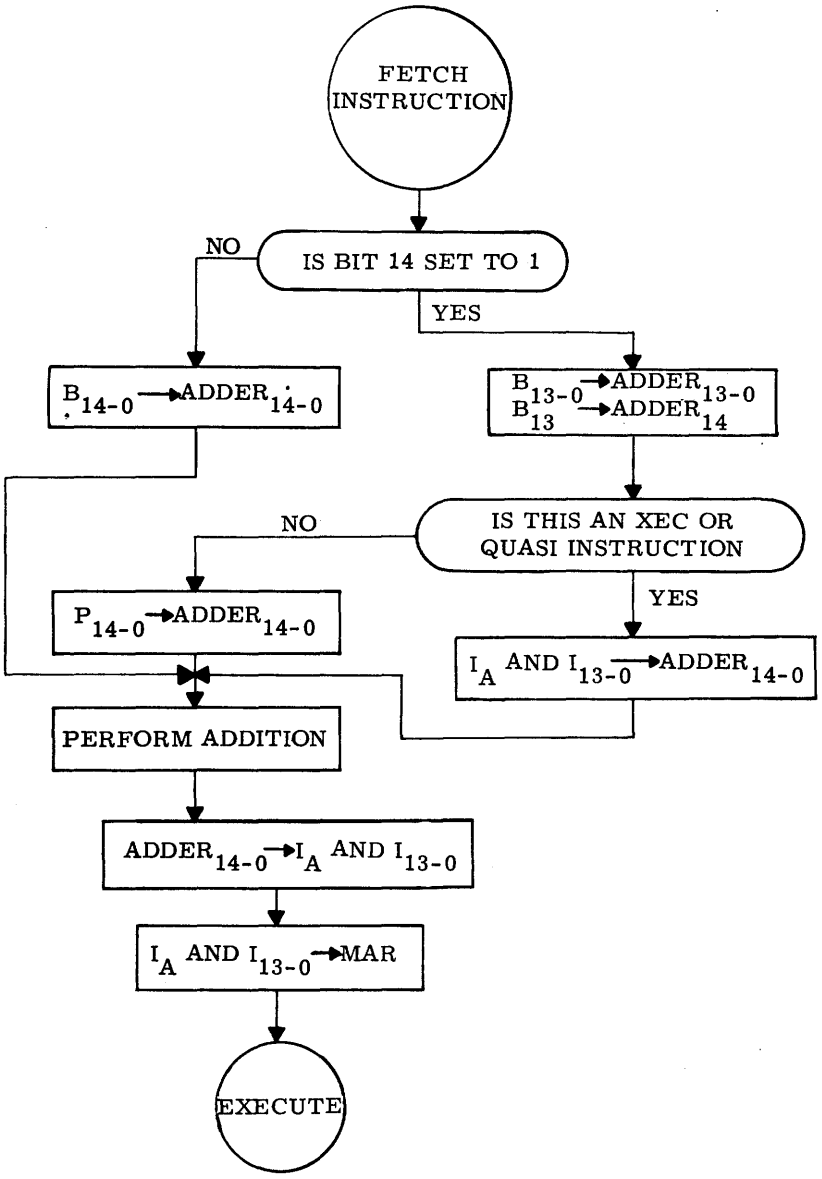


Fig. CMD. 9 Relative Addressing Flow Chart

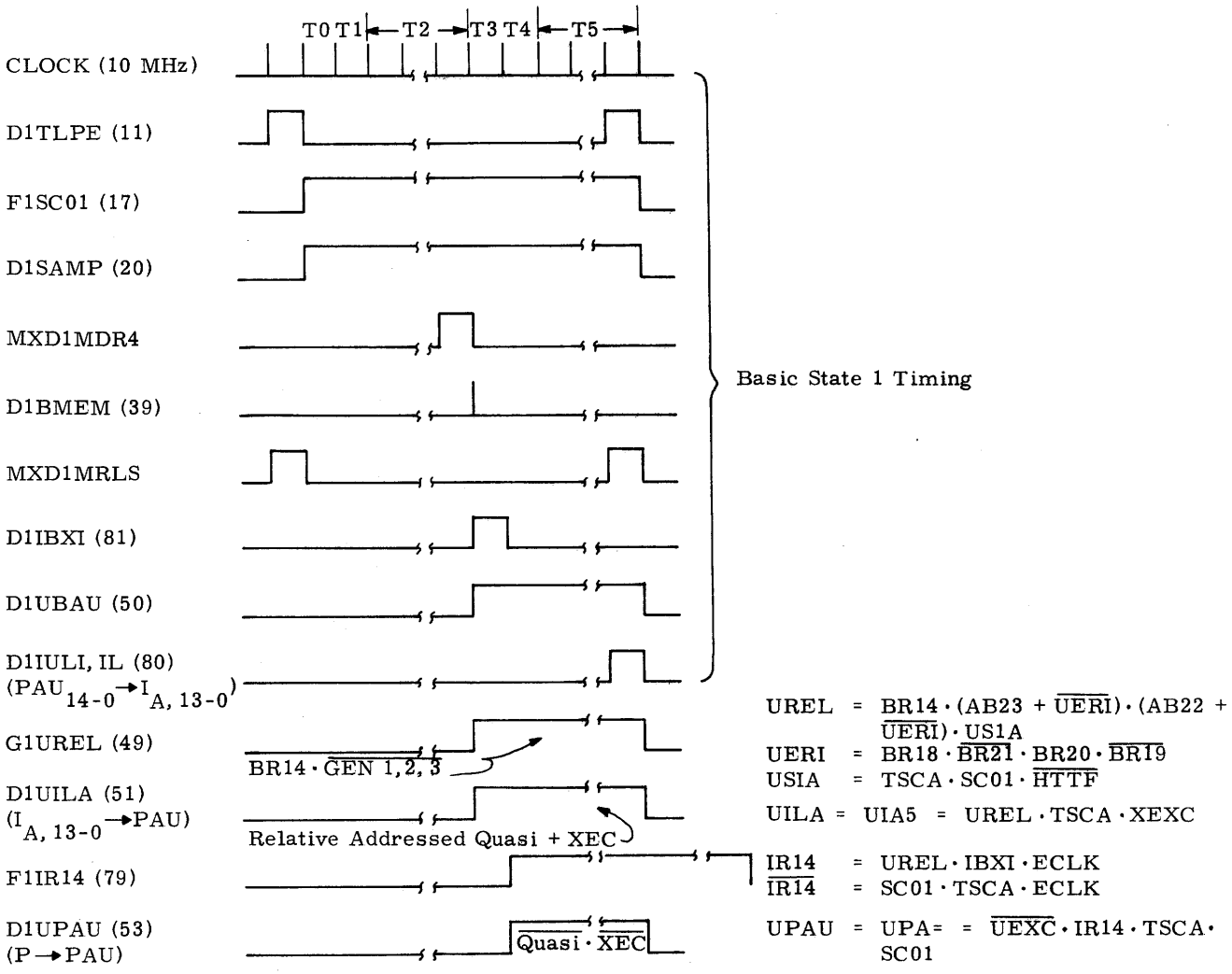
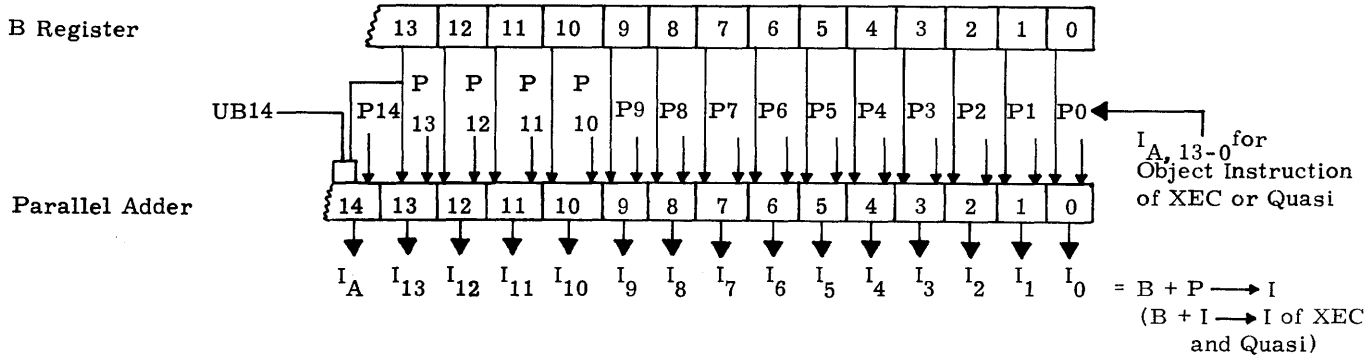


Fig. CMD. 10 Relative Addressing Timing Diagram



Example 1:

	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Command Operand in B Register	0	1	1	0	0	0	0	0	0	0	0	0	1	1	$= +6,147_{10}$
Command Address in P (or I) Register	0	1	1	1	0	0	0	0	0	0	0	0	0	0	$= 14,336_{10}$
Parallel Adder Result	1	0	1	0	0	0	0	0	0	0	0	0	1	1	$= 20,483_{10}$

Gated to $I_{A,13-0}$ as Effective Operand Address (Z)

Example 2:

	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Command Operand In B Register	1	0	0	0	0	0	0	0	0	0	0	0	0	0	$= 8,192_{10}$
Command Address in P (or I) Register	0	1	1	0	0	0	0	0	0	0	0	0	0	0	$= 12,228_{10}$
Parallel Adder Result	0	0	1	0	0	0	0	0	0	0	0	0	0	0	$= 4,096_{10}$

Gated to $I_{A,13-0}$ as Effective Operand Address (Z)

Fig. CMD. 11 Relative Addressing Examples

MEMORY WRAP AROUND

In a system that uses less than the maximum (32K) memory, instructions may be executed with an address greater than the implemented memory size.

In such cases the memory will wrap around or fetch zeros as shown in the chart below. The point at which the implemented memory size and actual 8K block addressed, cross, indicates the 8K block that will be affected.

Implemented Memory

8K Block Addressed	16K	24K	32K
0 - 8K	0 - 8	0 - 8	0 - 8
8 - 16K	8 - 16	8 - 16	8 - 16
16 - 24K	0 - 8	16 - 24	16 - 24
24 - 32K	8 - 16	*	24 - 32

* A zero word is read with no parity error. A write operation would go undetected unless Memory Protect is used.

Memory wrap around is accomplished by enabling, within the multiplexer, only those address bits required to address the maximum implemented memory location. That is, if the implemented memory is 16K words, then address bits 0 through 13 are wired to the memory, etc.

COMMAND DESCRIPTION CONVENTIONS

Each command is described individually and includes the following information:

a. Mnemonic.

b. Operation Format.

c. Effective Operand Address

d. Command Type (e.g., Full Operand, Quasi, GEN 1).

e. A chart indicating the registers, memory cells and basic flip-flops affected by the command.

f. Description of the instruction operation.

g. Block diagram of the registers effected.

h. Timing diagram of the primary control signals enabled.

i. Logic equations for enabling the primary control signals.

j. Description of the hardware operation.

Within the command descriptions, the following terms and symbols are used:

X - Specifies the index word used in the execution of a command.

Y - Represents the operand address of the command and implies the use of an operand from storage.

Z - Represents the effective operand address. It is developed as a function of X, *, and Y.

* - Indicates relative address modification (relative to the instruction's own location).

K - A constant in the address portion of certain commands.

f - Refers to the function of that which follows (e.g., Z = F (X, *, Y) should be read as Z is equal to the function of indexing, relative addressing, and the operand address). Subscripts refer to bit positions of the designated register (e.g., B_{13-0} delineates bit positions 13 through 0 of the B Register).

C () - Refers to the contents of whatever is enclosed within the parenthesis [e.g., C (A_{23-0}) is read as the contents of A Register bits 23 through 0].

All command descriptions are presented in alphabetical order except GEN 1, GEN 2, GEN 3, and Quasi commands which are contained under their group mnemonic in alphabetical order (i.e., GEN 1, GEN 2, etc.).

Table CMD. 1 contains an alphabetical listing of commands operable with the 4022D Arithmetic Unit. Table CMD. 2 lists these same commands according to their OP Code. Table CMD. 3 lists the commands along with the specific institution decode gates enabled for each command. Using these tables, the command format may be readily obtained for any command mnemonic, and the command mnemonic may be obtained for any command OP Code.

Command	Description	Type	Operation Format	Command	Description	Type	Operation Format
ABL	Append Beginning of List	Quasi	57X*Y	LMR_1	Load Mask Register (NI)	GEN 2	25000302
				LMR_2	Load Mask Register$_2$ (I)	GEN 2	25000300
ABT	Abort Device D's Operation	GEN 2	25X3D	LPR	Load Place And Restore	F.O.	35X*Y
				LXC	Load X With Count	F.O.	17X00000
ACT	Activate Device D	GEN 2	25X1D	LXK	Load X With K	F.O.	07X*K
ADD	Add	F.O.	11X*Y	MAQ	Move A to Q	GEN 3	45004330
ADO	Add One To Bit K	GEN 1	0500700K	MPY	Multiply	F.O.	55X*Y
AEL	Append End Of List	Quasi	47X*Y	NEG	Negate	GEN 1	05013000
AKA	Add K to A	Quasi	60X0K	NOP	No Operation	F.O.	26200000
ANA	Logical And To A	F.O.	20X*Y	OOM	Operate On Memory	Quasi	62X*Y
BRU	Branch Unconditionally	F.O.	14X*Y	OPR	Operate Device D	GEN 2	25X2D
BTR	Branch If Test FF Reset	F.O.	30X*Y	ORA	Logical OR To A	F.O.	21X*Y
				OUT	Out To Device D	GEN 2	25X4D
BTS	Branch If Test FF Set	F.O.	34X*Y	PAI	Permit Automatic Interrupt	GEN 2	25020000
CBK	Change Bit K	GEN 1	05X4700K				
CLO	Count Least Significant Ones	GEN 1	05004137	RALM	Reset Programmable Alarm	GEN 2	25000400
CLZ	Count Least Significant Zeros	GEN 1	05070137	RAPG	Reset Adjustable Pulse Generator	GEN 2	25000500
CMO	Count Most Significant Ones	GEN 1	05004237	RBK	Reset Bit K	GEN 1	0504500K
				RBL	Remove Beginning Item From List	Quasi	56X*Y
CMZ	Count Most Significant Zeros	GEN 1	05070237				
				RCS	Read Console Switches	GEN 2	25050000
CNTO	Count Ones GEN 1	GEN 1	05004337	REL	Remove Ending Item From List	Quasi	46X*Y
CNTZ	Count Zeros	GEN 1	05070337				
CPL	Complement A	GEN 1	0501000	REV	Reset Test FF Bit K Even	GEN 1	05X7040K
DAD	Double Length Add	Quasi	51X*Y				
DLA	Double Left Arithmetic	GEN 3	45X0644K	RNZ	Reset Test FF A Non-Zero	GEN 1	05004470
DLD	Double Length Load	Quasi	41X*Y				
DLL	Double Left Logical	GEN 3	45X0720K	ROD	Reset Test FF Bit K Odd	GEN 1	05X0440K
DMT	Decrement Memory And Test	F.O.	060*Y				
				RST	Reset Test FF	GEN 1	05004737
DRA	Double Right Arithmetic	GEN 3	45X0440K	SALM	Set Programmable Alarm	GEN 2	25000402
DRC	Double Right Circular	GEN 3	45X0530K	SAPG	Set Adjustable Pulse Generator	GEN 2	25000502
DRL	Double Right Logical	GEN 3	45X0430K				
DST	Double Length Store	Quasi	63X*Y	SBK	Set Bit K	GEN 1	05X4600K
DSU	Double Length Subtract	Quasi	61X*Y	SEL	Select Device D	GEN 2	25X0D
DVD	Divide	F.O.	65X*Y	SET	Set Test FF	GEN 1	05004637
DVDM	Divide by Magnitude	Quasi	75X*Y	SEV	Set Test FF Bit K Even	GEN 1	05X7050K
ERA	Exclusive OR To A	F.O.	10X*Y				
FAD	Floating Point Add	Quasi	70X*Y	SKA	Subtract K From A	Quasi	50X0Y
FDV	Floating Point Divide	Quasi	73X*Y	SLA	Shift Left Arithmetic	GEN 3	45X0204K
FIX	Fix-Floating Number	Quasi	74X0K	SLL	Shift Left Logical	GEN 3	45X0200K
FLO	Float-Fixed Number	Quasi	74X2K	SNZ	Set Test FF A Non-Zero	GEN 1	05004570
FMP	Floating Multiply	Quasi	72X*Y				
FMS	Floating Mode Shift	Quasi	74X14K	SOD	Set Test FF Bit K Odd	GEN 1	05X0450K
FSU	Floating Subtract	Quasi	71X*Y	SPB	Save Place And Branch	F.O.	33X*Y
IAI	Inhibit Automatic Interrupt (Inhibitable)	GEN 2	25030000	SRA	Shift Right Arithmetic	GEN 1	05X1404K
				SRC	Shift Right Circular	GEN 1	05X0404K
IAI_2	Inhibit Automatic Interrupt$_2$ (All)	GEN 2	25000304	SRL	Shift Right Logical	GEN 1	05X0004K
				SSA	Set Stall Alarm	GEN 2	25010000
IBK	Isolate Bit K	GEN 1	05X0100K	STA	Store A	F.O.	32X*Y
IN	In From Device D	GEN 2	25X5D	STI	Store A Indirect	Quasi	53X*Y
INX	Increment X	F.O.	26X*K	STMF	Set Trapping Mode	GEN 2	25000001
JCB	Jump If Channel Busy	GEN 2	25X6D2D	STQ	Store Q	F.O.	44X*Y
JDR	Jump If Data Ready	GEN 2	25X6D4D	STX	Store X	F.O.	06X*Y
JND	Jump If No Demand	GEN 2	25040000	SUB	Subtract	F.O.	31X*Y
JNE	Jump If No Error	GEN 2	25X7D	TER	Test Even Reset Bit K	GEN 1	05X4560K
JNO	Jump If No Overflow	GEN 2	25060000	TES	Test Even Set Bit K	GEN 1	05X4660K
JNP	Jump If No Parity Error	GEN 2	25070000	TEV	Test Bit K Even	GEN 1	05X7070K
JNR	Jump If Not Ready	GEN 2	25X6D	TNM	Test Not Minus One	GEN 1	05070770
LBM	Load Bit Mask	GEN 1	05X6300K	TNZ	Test A Non-Zero	GEN 1	05004770
LDA	Load A	F.O.	00X*Y	TOD	Test Bit K Odd	GEN 1	05X0470K
LDI	Load A Indirect	Quasi	52X*Y	TOR	Test Odd Reset Bit K	GEN 1	05X4570K
LDK	Load A With K	Quasi	40X*K	TOS	Test Odd Set Bit K	GEN 1	05X4670K
LDO	Load One Into Bit K	GEN 1	05X0300K	TSC	Test and Shift Circular	GEN 1	05X0464K
LDP	Load Place	F.O.	15X*Y				
LDQ	Load Q	F.O.	42X*Y	TXH	Test X High or Equal	F.O.	24X0-K
LDX	Load X	F.O.	16X*Y	TZC	Test Zero And Complement	GEN 1	05064670
LDZ	Load Zeros	GEN 1	05000000				
LMO	Load Minus One	GEN 1	05060000	TZE	Test A Zero	GEN 1	05004670
				XEC	Execute	F.O.	04X*Y

Note: Four-character Commands are not recognized by the standard assembler.

Table CMD. 1 GE-PAC 4020 Command Listing

OPERATION FORMAT	COMMAND	OPERATION FORMAT	COMMAND	OPERATION FORMAT	COMMAND
00X*Y	LDA	06X*Y	STX	32X*Y	STA
04X*Y	XEC	060*Y	DMT	33X*Y	SPB
05000000	LDZ	07X*K	LXK	34X*Y	BTS
05004137	CLO	10X*Y	ERA	35X*Y	LPR
05004237	CMO	11X*Y	ADD	40X*K	LDK
05004337	CNTO	14X*Y	BRU	41X*Y	DLD
05004470	RNZ	15X*Y	LDP	42X*Y	LDQ
05004570	SNZ	16X*Y	LDX	44X*Y	STQ
05004637	SET	17X00000	LXC	45004330	MAQ
05004670	TZE	20X*Y	ANA	45X0200K	SLL
05004737	RST	21X*Y	ORA	45X0204K	SLA
05004770	TNZ	24X0-K	TXH	45X0430K	DRL
05010000	CPL	25X0D	SEL	45X0440K	DRA
05013000	NEG	25X1D	ACT	45X0530K	DRC
05060000	LMO	25X2D	OPR	45X0644K	DLA
05064670	TZC	25X3D	ABT	45X0720K	DLL
05070137	CLZ	25X4D	OUT	46X*Y	REL
05070237	CMZ	25X5D	IN	47X*Y	AEL
05070337	CNTZ	25X6D	JNR	50X0Y	SKA
05070770	TNM	25X6D2D	JCB	51X*Y	DAD
05X0004K	SRL	25X6D4D	JDR	52X*Y	LDI
05X0100K	IBK	25X7D	JNE	53X*Y	STI
05X0300K	LDO	25000001	STMF	55X*Y	MPY
05X0404K	SRC	25000300	LMR_2	56X*Y	RBL
05X0440K	ROD	25000302	LMR	57X*Y	ABL
05X0450K	SOD	25000304	IAI_2	60X0K	AKA
05X0464K	TSC	25000400	RALM	61X*Y	DSU
05X0470K	TOD	25000402	SALM	62X*Y	OOM
05X0700K	ADO	25000500	RAPG	63X*Y	DST
05X1404K	SRA	25000502	SAPG	65X*Y	DVD
05X4500K	RBK	25010000	SSA	70X*Y	FAD
05X4560K	TER	25020000	PAI	71X*Y	FSU
05X4570K	TOR	25030000	IAI	72X*Y	FMP
05X4600K	SBK	25040000	JND	73X*Y	FDV
05X4660K	TES	25050000	RCS	74X0K	FIX
05X4670K	TOS	25060000	JNO	74X1K	FMS
05X4700K	CBK	25070000	JNP	74X2K	FLO
05X6300K	LBM	26X*K	INX	75X*Y	DVDM
05X7040K	REV	26200000	NOP	76X*Y	Optional
05X7050K	SEV	30X*Y	BTR	77X*Y	Optional
05X7070K	TEV	31X*Y	SUB		

Table CMD. 2 GE-PAC 4010 Command Operation Format Listing

Op Code Bits	23·22·21 (DCN0)	23·22·21 (DCN1)	23·22·21 (DCN2)	23·22·21 (DCN3)
20·19·18 (DLC0)	LDA 00	ERA 10	ANA 20	BTR 30
20·19·18 (DLC1)		ADD 11	ORA 21	SUB 31
20·19·18				STA 32
20·19·18				SPB 33
20·19·18 (DLC4)	XEC 04	BRU 14	TXH 24	BTS 34
20·19·18 (DLC5)	Gen I 05	LDP 15	Gen II 25	LPR 35
20·19·18 (DLC6)	STX/ 06 DMT	LDX 16	INX 26	
20·19·18 (DLC7)	LXK 07	LXC 17		

Op Code Bits	23·22·21 (DCN0)	23·22·21 (DCN1)	23·22·21 (DCN2)	23·22·21 (DCN3)
20·19·18	Quasi	Quasi	Quasi	Quasi
20·19·18	Quasi	Quasi	Quasi	Quasi
20·19·18	LDQ 42	Quasi	Quasi	Quasi
20·19·18	Quasi	Quasi	Quasi	Quasi
20·19·18 (DUC4)	STQ 44	Quasi	Quasi	Quasi
20·19·18 (DUC5)	Gen III 45	MPY 55	DVD 65	Quasi
20·19·18	Quasi	Quasi	Quasi	Quasi
20·19·18	Quasi	Quasi	Quasi	Quasi

Table CMD. 3 Operation Code Assignments

ADD - ADD Z TO A

FULL OPERAND

23 18	17 15	14	13 0
11	X	*	Y

Z = f (X, *, Y)

ADD sums the contents of core cell Z with the contents of the A Register and stores the result in A. If the result is too large to be stored in the 23 data bits of the A Register (i.e., more negative than -2^{23} or more positive than $2^{23}-1$), the Overflow flip-flop (F1UOFL) is set. Overflow may occur only when the signs of the original numbers in A and Z are alike. Overflow is then detected if a sign change occurs in the sum.

A non-indexed ADD command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The contents of memory location Z are gated to the B Register by D1BMEM during the Clock pulse of Memory Data Ready (MUD1MDRY). From B, the contents of memory location Z are gated (D1UBAU) to the Adder Unit at the same time the contents of the A Register are gated (D1UAAU) to the Adder Unit. The summation occurs in the Adder Unit and the sum is gated (D1AAUL,U) to the A Register at the Clock of Memory Release (MUD1MRLS).

If arithmetic overflow - either positive or negative - occurs during the summation, the Overflow flip-flop (F1UOFL) is set. The following examples illustrate (a) a positive overflow, and (b) a negative overflow. For simplicity, five bit (4 data and a sign) registers are illustrated. Consider the most significant bit as bit 23 and the next most significant bit as bit 22.

Consider: (A = +5) + (Z = +12) = 17. Seventeen is too large to be contained in 4 bits plus a sign bit; thus, the Overflow flip-flop is set.

Contents of Z	0 1100	(12)
Contents of A	0 0101	(5)
Sum	1 0001	Overflow Set = $23S \cdot \overline{23C} \cdot 22C$

(a) POSITIVE OVERFLOW.

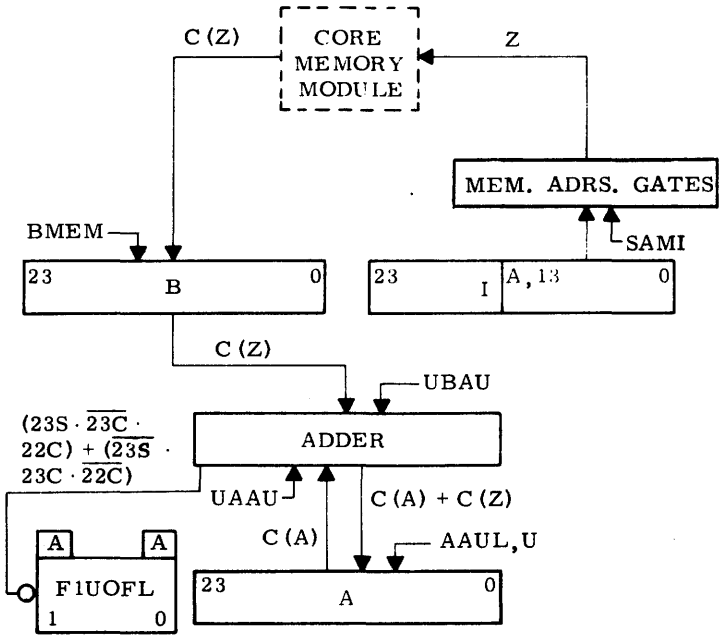
Consider: (A = -8) + (Z = -13) = -21. Minus 21 is too large to be contained in 4 bits plus a sign bit; thus, the Overflow flip-flop is set.

Contents of Z	1 0010	(-13)
Contents of A	1 1000	(-8)
Sum	0 1010	Overflow Set = $\overline{23S} \cdot 23C \cdot \overline{22C}$

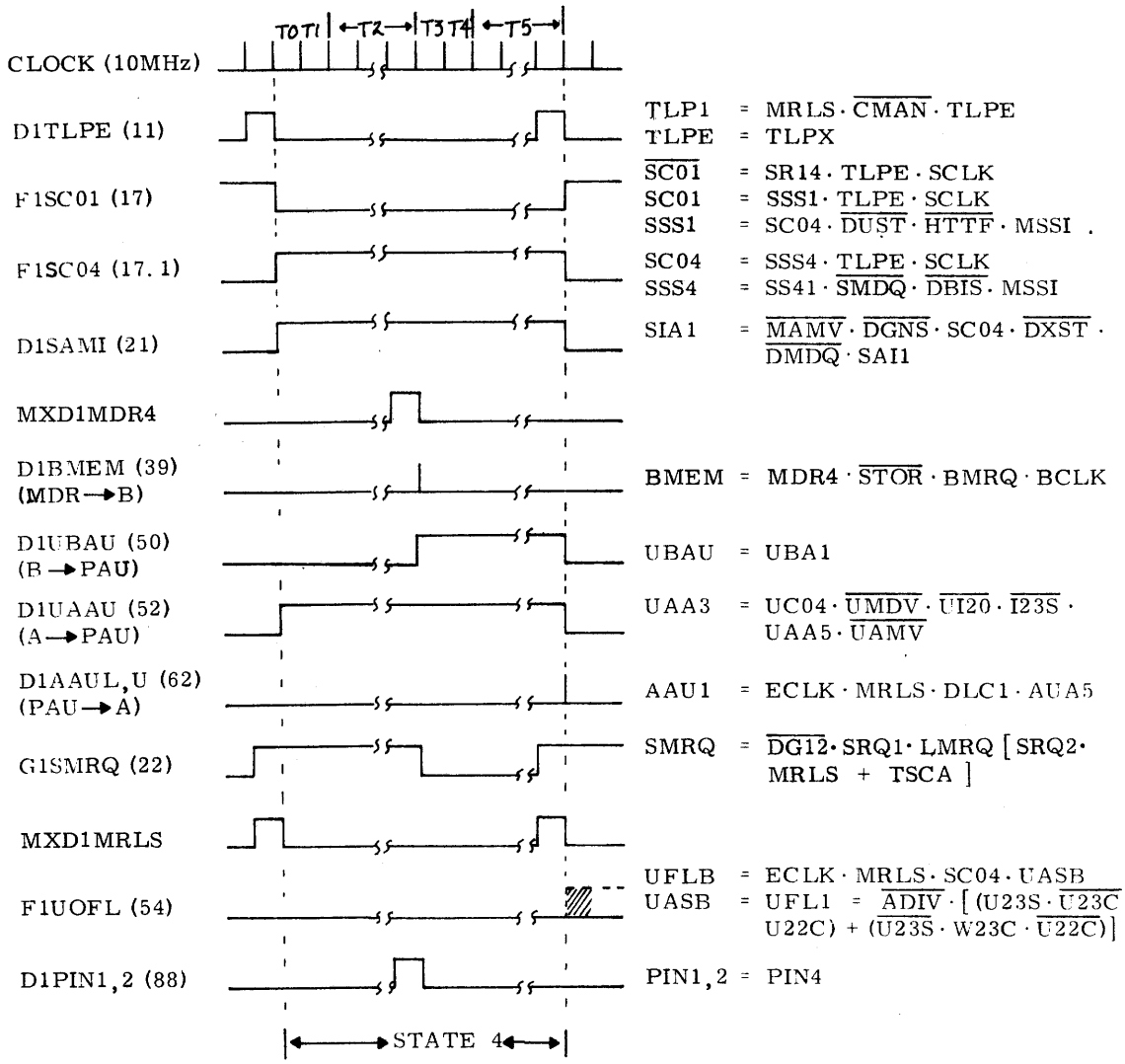
(b) NEGATIVE OVERFLOW.

Non-Indexed Word Times.	2 (S1, S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	C (A) + C (Z)
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	Set if overflow occurs
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS



ADD BLOCK DIAGRAM



ADD Timing Diagram

ANA-LOGICAL AND TO A

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
20	X	*	Y

Z = f (X, *, Y)

ANA performs the logic AND of the contents of core cell Z with the contents of A. The corresponding bits of Z and A are compared. If the corresponding bits are both "one", a "one" is placed in that position of A. If either or both of the compared positions are "zero", a "zero" is placed in that position of A.

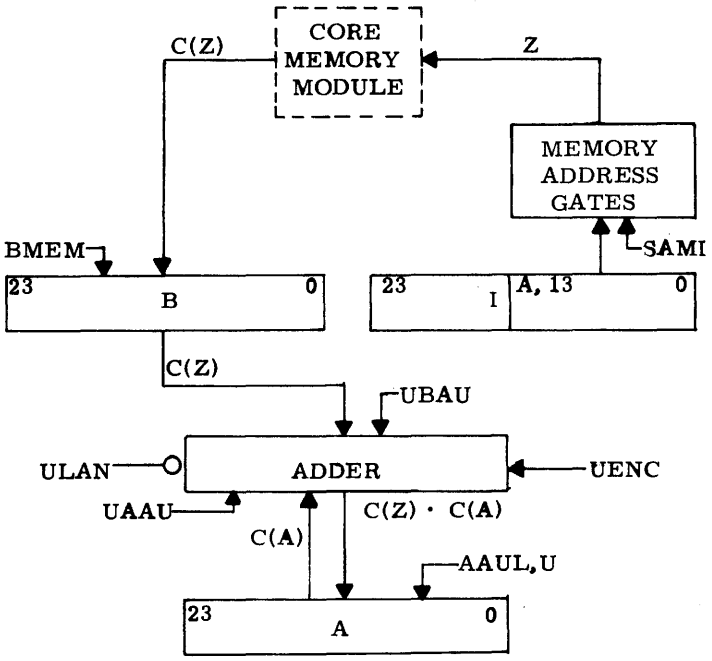
A non-indexed ANA command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,\,13-0}$ (D1SAMI) during State 4. The contents of memory cell Z are then gated to the B Register by D1BMEM during the clock pulse of Memory Data Ready (MXD1MDR4). From B, the contents of memory location Z are gated to the Adder Unit (D1UBAU). At the same time, the contents of the A Register are gated (D1UAAU) to the Adder Unit along with a control signal, D0ULAN, and the Enable Carry signal, G1UENC. D0ULAN, in conjunction with G1UENC, enables the logical AND function of the Adder Unit. The result of the logical AND is then gated back to the A Register to complete the ANA execution cycle.

To exemplify the ANA comparison, consider the following; four bits are used for simplicity.

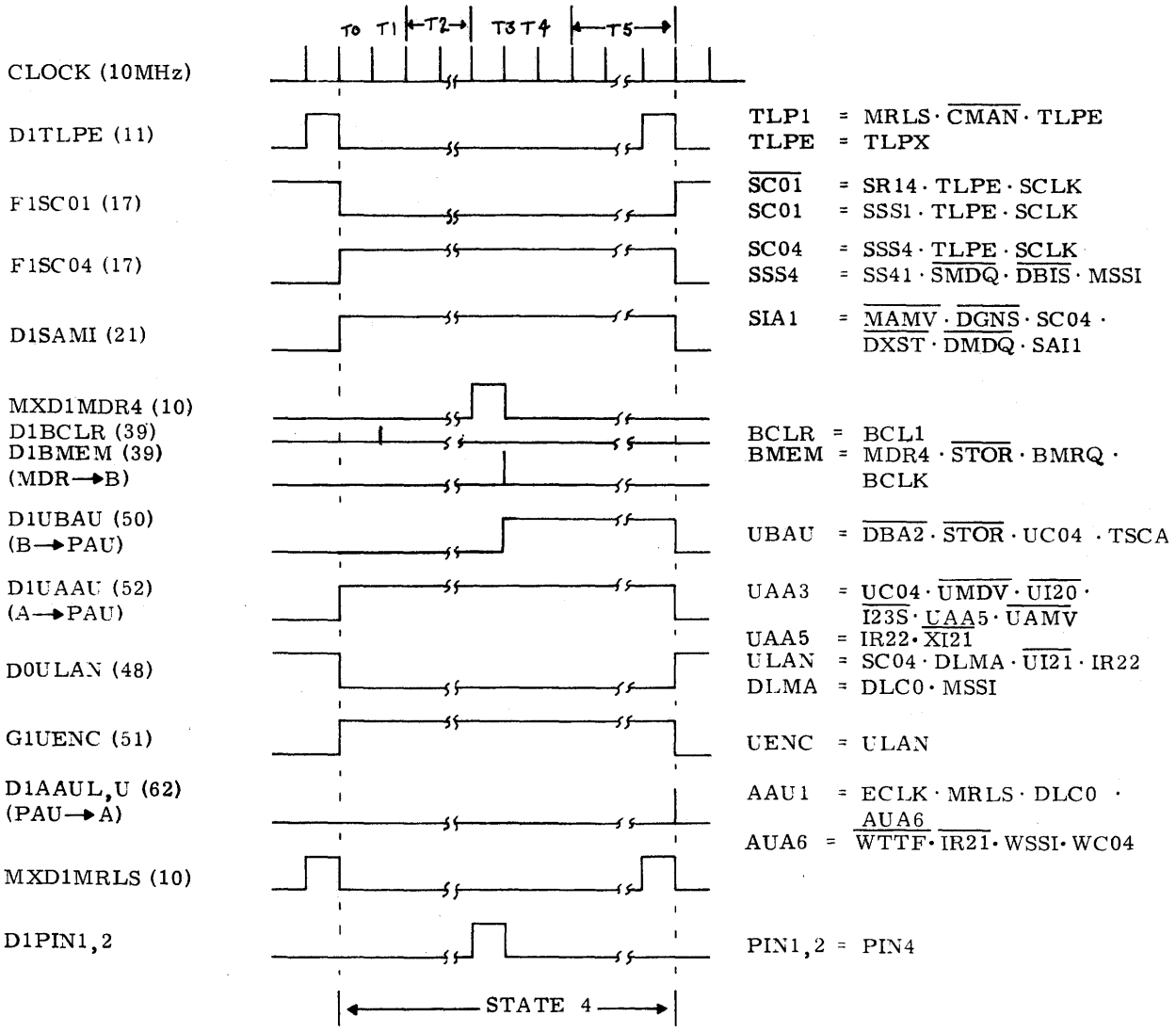
Contents of the A Register	=	0011
Contents of Cell Z	=	0101
Result Placed in A Register	=	0001

Non-Indexed Word Times	2 (S1, S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$C(Z_{23-0})$ AND with $C(A_{23-0})$
Q_{23-0}	
P_{14-0}	C(P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS



ANA BLOCK DIAGRAM



ANA TIMING DIAGRAM

BRU - BRANCH UNCONDITIONALLY

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
14	X	*	Y

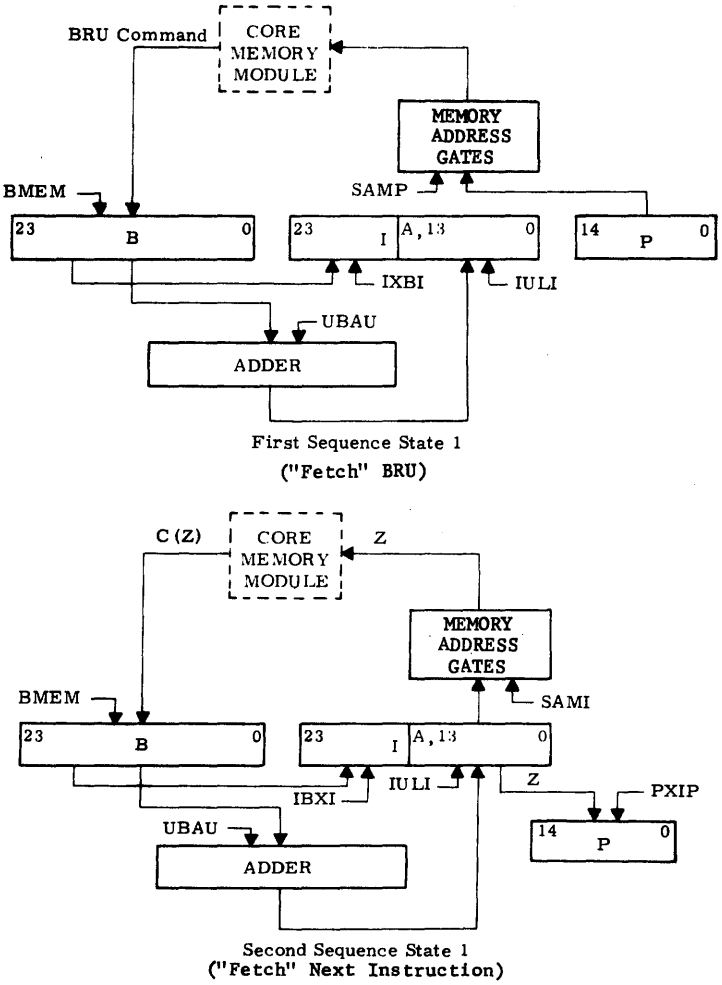
Z = f (X,*,Y)

BRU unconditionally transfers program control to the designated core address Z. This is accomplished by calculating the address Z, if it is relative addressed and/or indexed, and then transferring this address to the Program (P) Register.

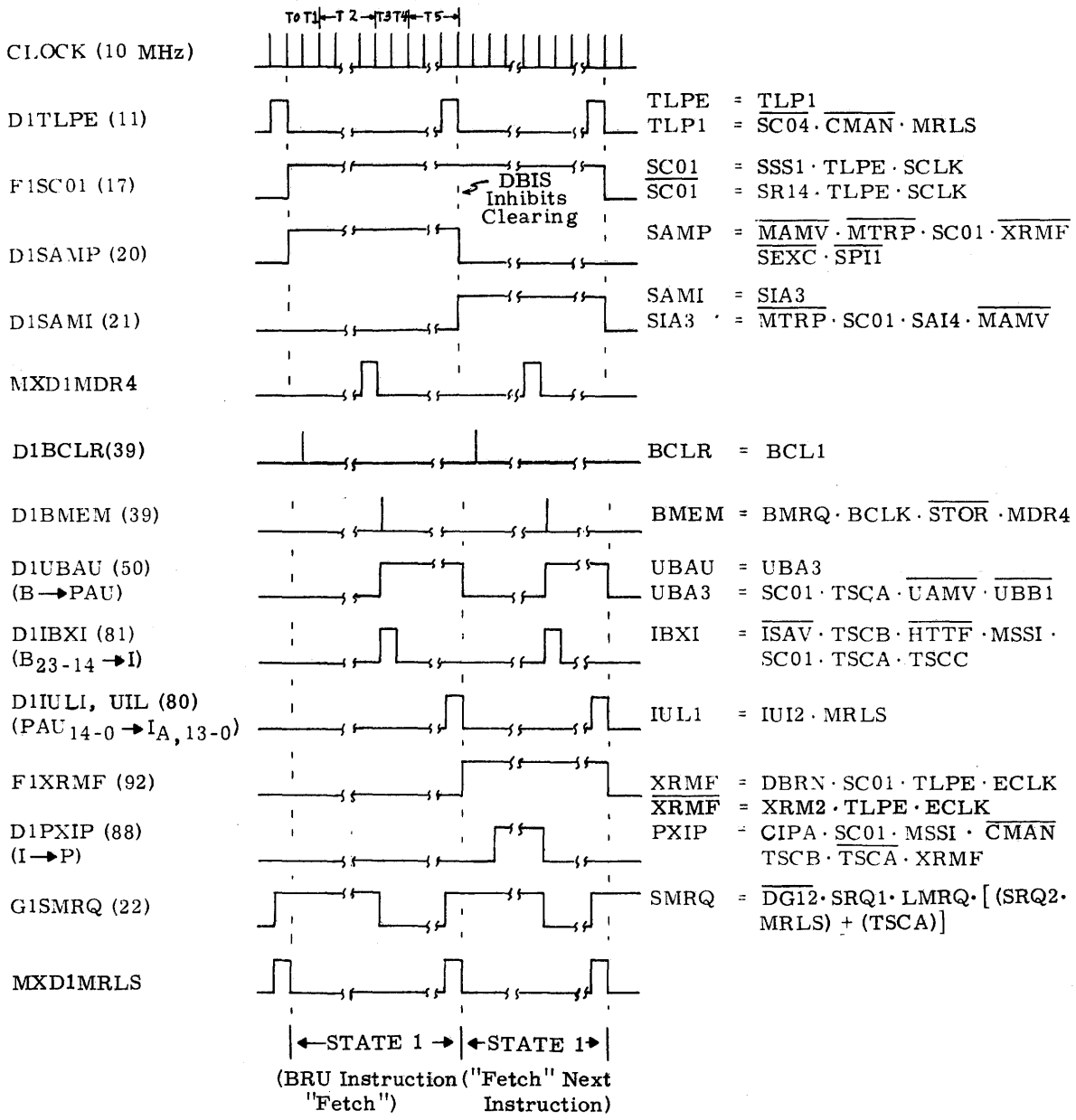
The BRU command is executed during Sequence Control State 1. For ease of understanding, two successive Sequence State 1's (i.e., 2 word times) are shown in the Block and Timing Diagrams. During the first word time of Sequence State 1 (F1SC01), the BRU command is "fetched" from memory in the normal manner except that the clock of Last Pulse Envelope (D1TLPE) sets the Remember flip-flop (F1XRMF). At the end of the first word time, State 1 is not cleared; instead, State 1 is held for another "fetch" cycle to obtain the command located in memory location Z. During this second word time, memory is addressed from $I_{A,13-0}$ and the command in location Z is "fetched" from memory in the normal manner. Also, during this second word time, the contents of $I_{A,13-0}$ are transferred to the Program Register (PXIP) to complete transfer of program control. At the end of this second word time, the Remember flip-flop is cleared.

Non-Indexed Word Times.	1 (S1)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	Z
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS



BRU BLOCK DIAGRAM



BRU TIMING DIAGRAM

BTR - BRANCH IF TEST FLIP-FLOP RESET

FULL OPERAND

23 18	17 15	14	13 0
30	X	*	Y

Z = f (X, *, Y)

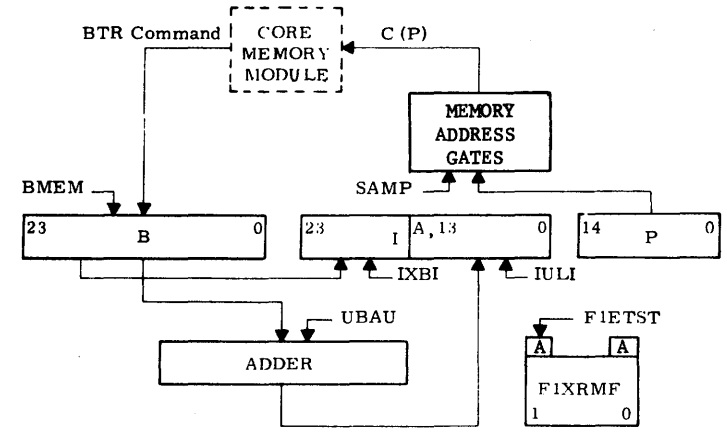
BTR transfers program control to memory location Z if the Test flip-flop (F1ETST) is reset. If the Test flip-flop is set, program control continues in sequence (i.e., C (P) +1). The status of the Test flip-flop is not changed by the BTR command.

To illustrate the operation of the BTR command, two successive word times are required (i.e., two Sequence Control State 1's). However, only one word time is used solely by the BTR command, since the second word time is used to "fetch" the next command. The Block and Timing Diagrams illustrate both word times of State 1. During the first word time, the BTR command is "fetched" from memory in the normal manner. At the clock of Last Pulse Envelope, the Remember flip-flop is set if the Test flip-flop (F1ETST) is reset. If the Test flip-flop is set, the Remember flip-flop remains reset.

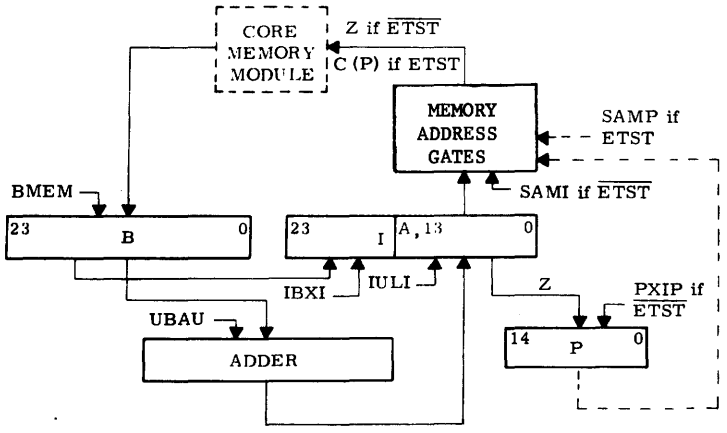
During the second word time, memory is addressed from $I_{A,13-0}$ (D1SAMI) if the Remember flip-flop is set (i.e., if the Test flip-flop is reset). In this manner, memory location Z is addressed. Also, if the Remember flip-flop is set, $I_{A,13-0}$ is transferred to the P Register (PXIP) to complete the transfer of program control. If, however, the Remember flip-flop is reset, indicating that the Test flip-flop is set, memory is addressed from the P Register and program control continues in the normal sequence.

Non-Indexed Word Times.	1 (S1)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	Z if F1ETST reset C (P) +1 if F1ETST set
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

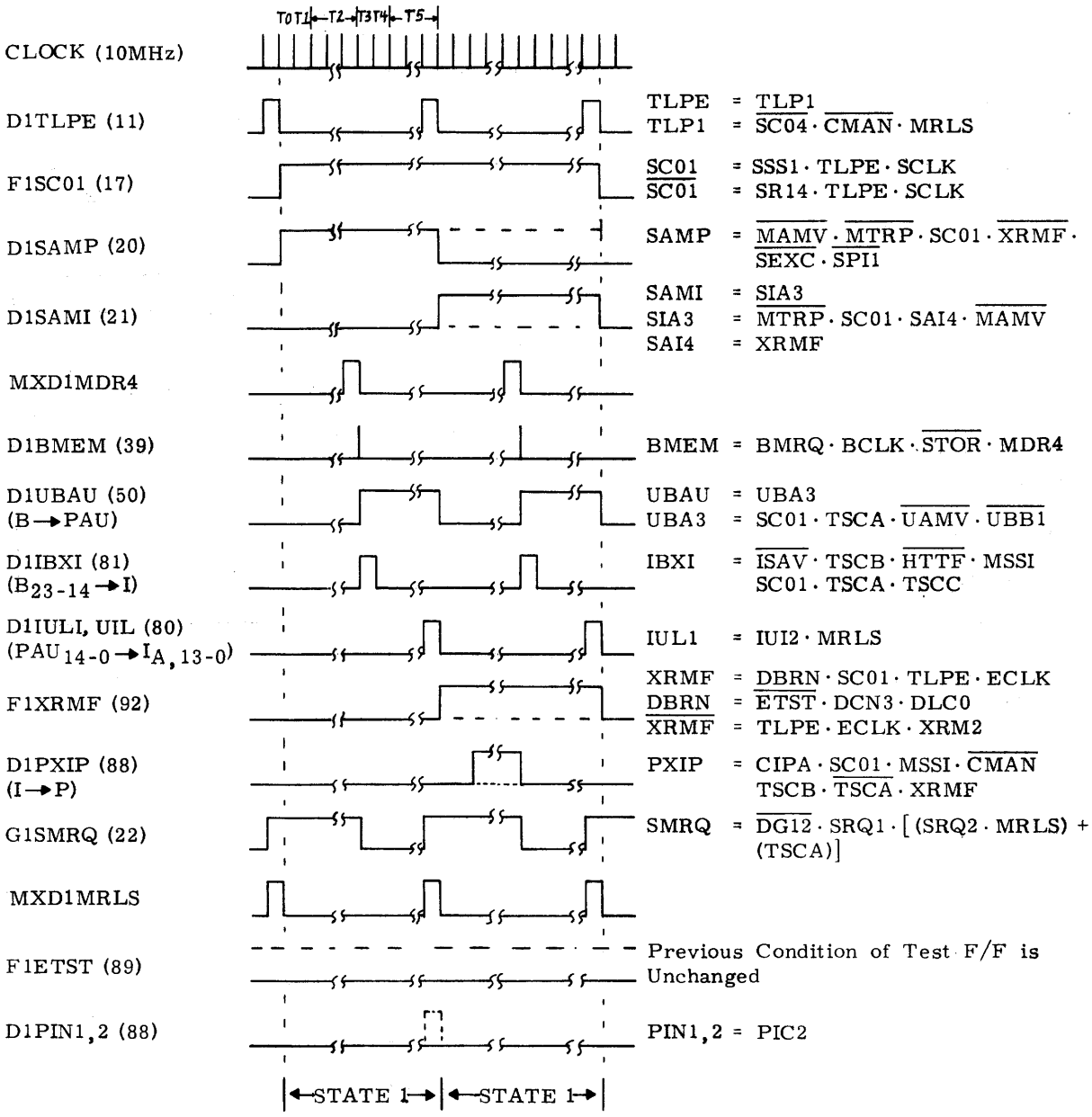


First Sequence State 1
("Fetch" BTR)



Second Sequence State 1
("Fetch" next instruction)

BTR BLOCK DIAGRAM



--- = Test flip-flop set

___ = Test flip-flop reset

BTR TIMING DIAGRAM

BTS - BRANCH IF TEST FLIP-FLOP SET

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
34	X	*	Y

Z = f (X, *, Y)

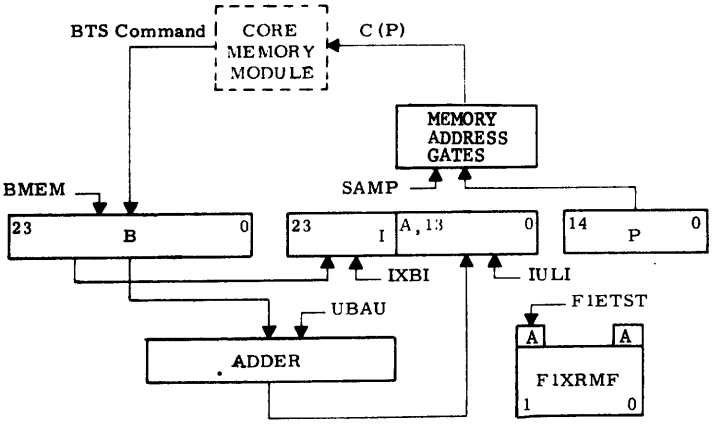
BTS transfers program control to memory location Z if the Test flip-flop (F1ETST) is set. If the Test flip-flop is reset, program control continues in sequence (i.e., C(P) +1). The status of the Test flip-flop is not changed by the BTS command.

To illustrate the operation of the BTS command, two successive word times are required (i.e., two Sequence Control State 1's). However, only one word time is used solely by the BTS command, since the second word time is used to "fetch" the next command. The Block and Timing Diagrams illustrate both word times of State 1. During the first word time, the BTS command is "fetched" from memory in the normal manner. At the clock of Last Pulse Envelope, the Remember flip-flop is set if the Test flip-flop (F1ETST) is set. If the Test flip-flop is reset, the Remember flip-flop remains reset.

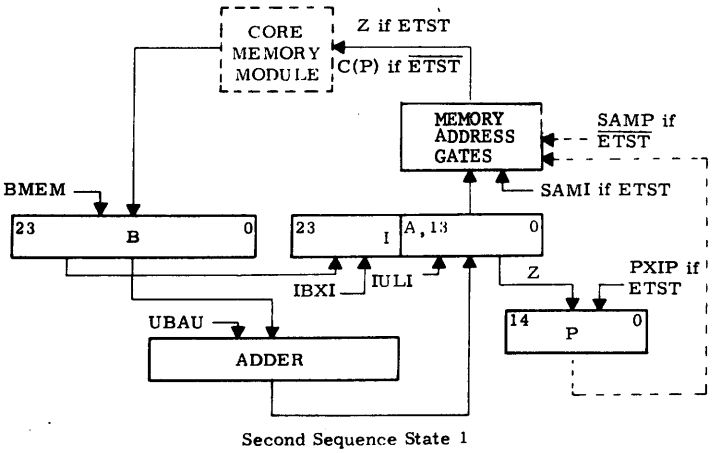
During the second word time, memory is addressed from $I_{A, 13-0}$ (D1SAMI) if the Remember flip-flop is set (i.e., if the Test flip-flop is set). In this manner, memory location Z is addressed. Also, if the Remember flip-flop is set, $I_{A, 13-0}$ is transferred to the P Register (PXIP) to complete the transfer of program control. If, however, the Remember flip-flop is reset, indicating that the Test flip-flop is reset, memory is addressed from the P Register and program control continues in the normal sequence.

Non-Indexed Word Times.	1 (S1)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	Z if F1ETST set C(P) + 1 if F1ETST reset
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

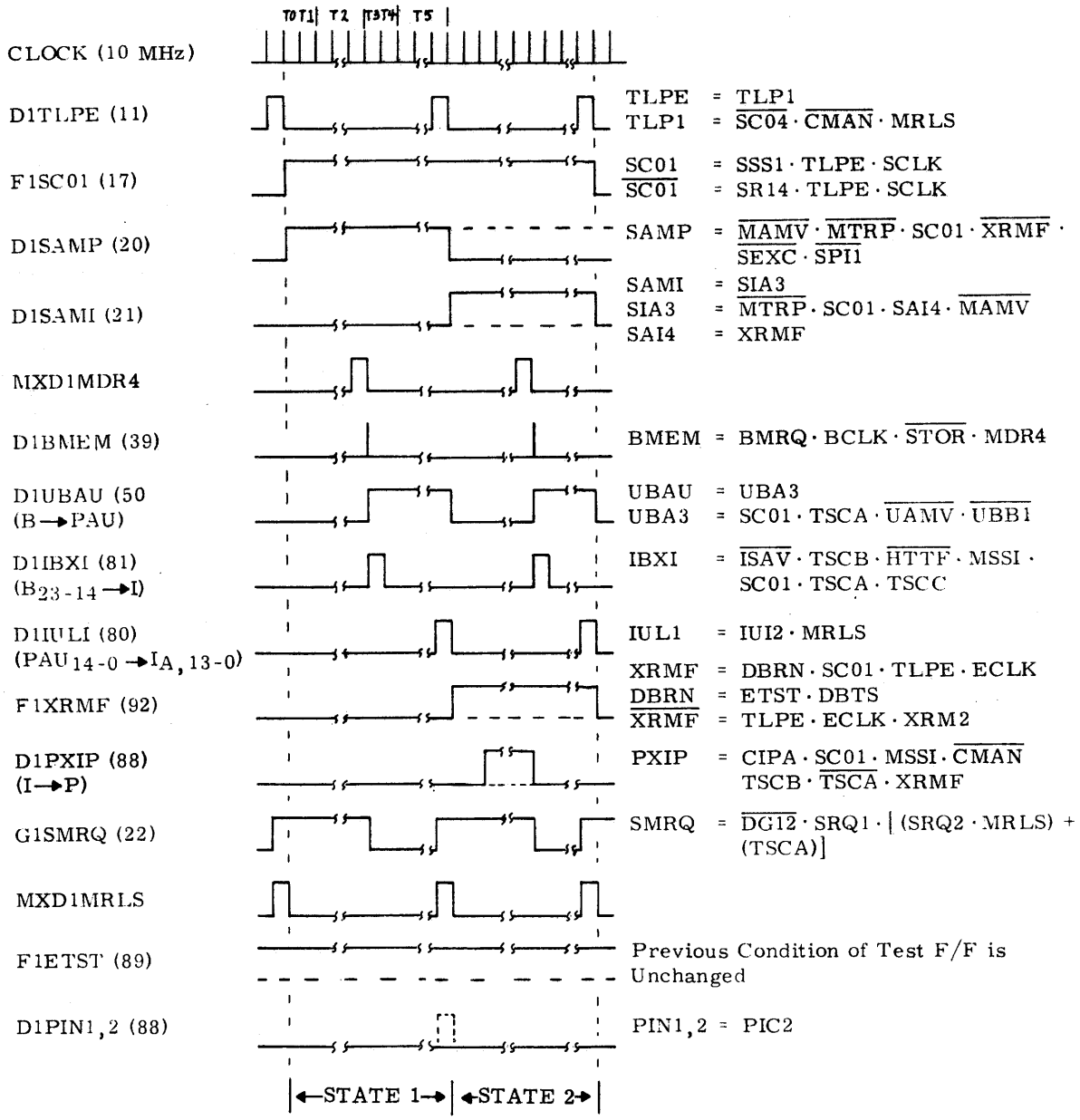


First Sequence State 1



Second Sequence State 1

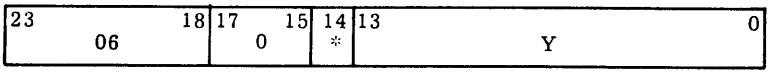
BTS BLOCK DIAGRAM



BTS TIMING DIAGRAM

DMT - DECREMENT MEMORY AND TEST

FULL OPERAND



Z = f (Y, *)

DMT subtracts 1 from the contents of memory cell Z each time it is executed. If the DMT is not the result of an API and the original contents of Z were not equal to 0, the Test flip-flop (F1ETST) is set. If the original contents of Z were equal to 0 and the DMT command is not the result of an API, the Test flip-flop is cleared. If the DMT command is the result of an API and the original contents of Z were equal to 0, a signal is applied to API for "Echo" generation. The Test flip-flop is not affected by a DMT resulting from an API. When an index address is specified (bits 17 through 15 not equal to 0) the DMT command is executed as an STX command.

The DMT command is executed during Sequence Control States 2 and 4. During State 2, memory is addressed from $I_{A,13-0}$ (D1SAMI) and the contents of memory location Z are gated to the B Register. From B, the contents of memory location Z are gated to the B input of the Adder Unit (D1UBBU). At the same time, both the true and complemented B Register contents are gated to the A input of the Adder Unit (D1UBAU, D1UBNA). Gating both the true and complemented B Register contents to the A input applies a "one" to the A input of each Adder bit position. This is the same as applying a -1 to the A inputs of the Adder. The Test flip-flop is cleared at Time 4 if the DMT command is not being executed from an interrupt response address ($\overline{\text{AIF0SPI2}}$). If a carry occurs from bit 23 of the Adder, due to the summation (contents of B plus -1), the Test flip-flop is set at the clock of Memory Release. This indicates that the original contents of Z were not equal to zero. If a carry from bit 23 of the Adder does not occur, the Test flip-flop remains in the reset state, indicating that the original contents of Z were equal to zero. If a carry from bit 23 of the Adder does not occur, G1WEKO is enabled allowing the API module to cause an interrupt at a different address, when the DMT command is executed from an interrupt response address. At the clock of Last Pulse Envelope, the Adder outputs (contents of Z -1) are gated back to the B Register.

During State 4, memory is again addressed from $I_{A,13-0}$ and the contents of B are gated back to memory location Z. In this manner, the original contents of Z -1 are stored back in memory location Z.

The following examples are used to illustrate that a carry occurs from PAU_{23} whenever the value to be decremented is not equal to 0. Only when the value being decremented goes from 0 to -1 will the absence of a carry occur from PAU_{23}. For simplicity, word lengths of 5 bits are used. Consider the most significant bit of the example to be PAU_{23}.

Word Times	3 (S1, S2, S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	* Set if C (Z) ≠ 0 Reset if C (Z) = 0
J_{4-0}	
Memory Z	C (Z) - 1

COMMAND CHARACTERISTICS

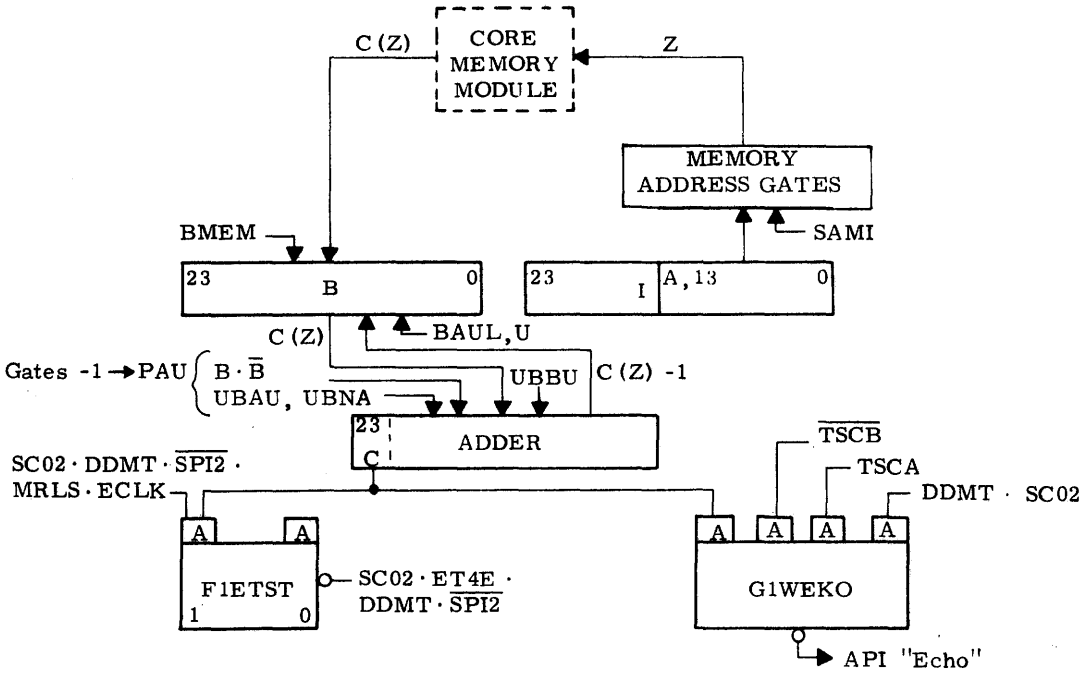
(Z = 2) 00010
(-1) 11111
00001 Plus Carry = Test F/F Set*

(Z = 1) 00001
11111
00000 Plus Carry = Test F/F Set*

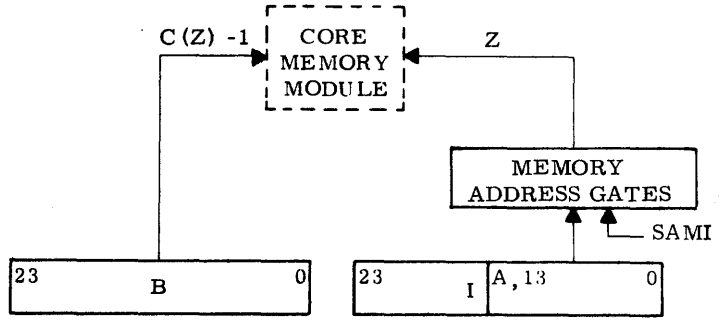
(Z = 0) 00000
11111
11111 No Carry = Test F/F Reset* or "Echo" to API

(Z = -1) 11111
11111
11110 Plus Carry = Test F/F Set*.

* The Test flip-flop is not affected if the DMT command is located in the interrupt response address and executed as the result of an Automatic Program Interrupt. In this case, The "Echo" signal applied to the API module when the original contents of Z were equal to zero will normally cause a different program interrupt.

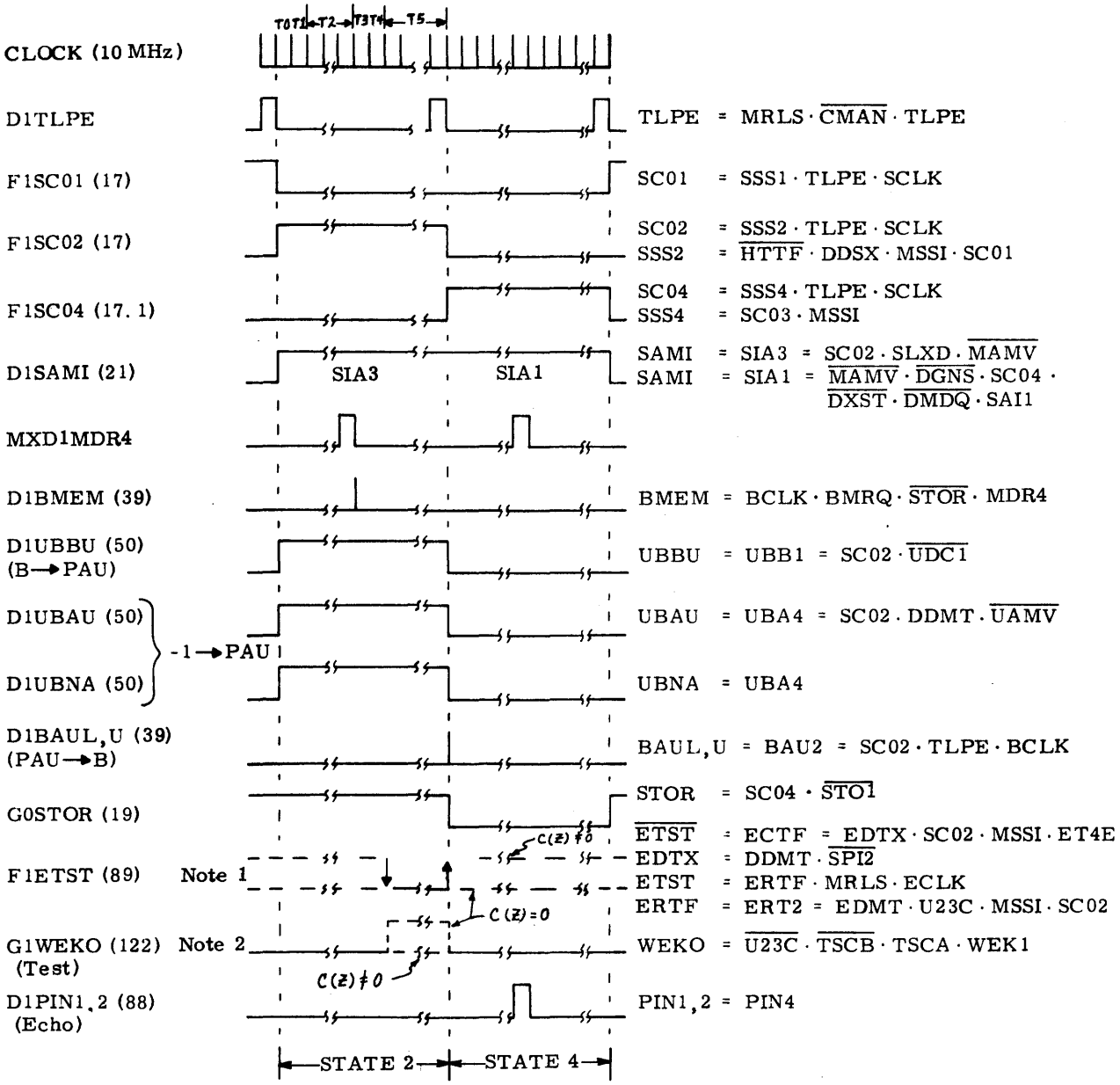


Sequence State 2



Sequence State 4

DMT BLOCK DIAGRAM



Notes:

1. The status of the Test flip-flop is not changed when the DMT is executed from an API response address as a result of an API ($\overline{SPI2}$).

2. The "Echo" signal (G1WEKO) is generated whenever the original contents of Z were zero ($\overline{U23C}$), however, this "Echo" signal is inhibited in the API module unless the DMT was executed from an API response address as the result of an API.

DMT TIMING DIAGRAM

DVD - DIVIDE

	23 — 18	17 — 15	14	13 — 0	
FULL OPERAND	65	X	*	Y	Z = f (X,*,Y)

DVD divides the 47-bit dividend, 24 bits of the A Register coupled with bits 22 through 0 of the Q Register, by the divisor contained in location Z. The quotient is placed in the Q Register and the remainder is placed in the A Register. If the quotient is too large to be contained in the Q Register, the Overflow flip-flop (F1UOFL) is set. The sign of A (A_{23}) applies to the remainder, and the sign of Q (Q_{23}) applies to the quotient.

Non-Indexed Word Times.	S1,S3,S4,S5 13.7 μs
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	Remainder
Q_{23-0}	Quotient
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	Set if arithmetic overflow
F1ETST	
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

> NOTE
>
> Either positive or negative numbers may be divided, however, the remainder always has the sign of the divisor, which may yield an unexpected result. The difference between the divisor and the remainder obtained yields the expected divisor (e.g., (+4) ÷ (-2) = -3, remainder -2.
>
> In actual mathematical operations within the computer, however, this is of little consequence since the quotient formed by any division is only an approximation (i.e., the remainder must be considered). Only when small integer values are divided is the quotient obtained of any significant difference from the expected quotient.

BINARY DIVISION

Division of positive numbers is the process of counting how many times one number (the divisor) can be subtracted from another number (the dividend) while still leaving a positive result. The number of times the divisor can be subtracted from the dividend is the result, or quotient. The value remaining after the repeated subtractions becomes the remainder.

The simplest form of binary division is accomplished by first subtracting the divisor from the most significant portion of the dividend. If the subtraction is valid (i.e., a positive result is obtained), a "one" is placed in that position of the quotient. The dividend is then shifted one place to the left and the divisor again subtracted from the dividend. If the subtraction is not valid (i.e., a negative result is obtained), a "zero" is placed in that position of the quotient and the divisor is added to the negative result to restore the divident. The dividend is then shifted one place to the left and the divisor again subtracted from the dividend. The remainder, after the division, is that portion of the dividend remaining after the last valid subtraction. This method of forming the quotient is illustrated in the following example. The most significant bit of the divisor and the dividend are sign bits.

Decimal Problem:

```
7 ÷ 3 = 2, remainder 1.
↑   ↑   ↑
|   |   └─Quotient
|   └─Divisor
└─Dividend
```

Binary Equivalent:

$$\frac{0\ 000111}{0\ 011} = 0\ 010, \text{ remainder } 0\ 001.$$

Binary Method:

```
                  0010
0   011)  0     000111
          1     101       - 1
   [0]<-- 1     101       - 2
          0     011       - 3
          0     000       - 4
                1101      - 5
      [0]<----  1110      - 6
                0011      - 7
                0001      - 8
                 1101     - 9
       [1]<---   0000     -10
                  1101    -11
        [0]<--    1110    -12
                  0011    -13
                  0001    -14
```

1. Subtract divisor.
2. Negative result, therefore, 0 → quotient (Sign).
3. Add divisor to restore dividend.
4. Restored dividend.
5. Shift left dividend and subtract divisor.
6. Negative result, therefore, 0 → quotient.
7. Add divisor to restore dividend.
8. Restored dividend.
9. Shift left dividend and subtract divisor.
10. Positive result, therefore, 1 → quotient.
11. Shift left dividend and subtract divisor.
12. Negative result, therefore, 0 → quotient.
13. Add divisor for remainder "fix up".
14. Remainder.

Example 1

The first subtraction in Example 1 is described by:

$D - 2^n (d) \rightarrow D$

where: D = dividend

d = divisor

n = register length minus 1.

The multiplication of the divisor (d) by 2^n merely aligns the divisor with the correct portion of the dividend and is accomplished by shifting left the divisor n places. For each successive subtraction, the divisor is moved one place to the right:

1st Sub:	-2^n (d)	=	1 101000
2nd Sub:	-2^{n-1} (d)	=	1 10100
3rd Sub:	-2^{n-2} (d)	=	1 1010
Final Sub:	-2^{n-n} (d)	=	1 101.

Notice in the previous example that, if the subtraction is invalid (i. e., quotient bit set to "zero"), it is necessary to add back the divisor to restore the dividend to its correct form. That is, the number 2^n (d) is first subtracted from the dividend, then added back. In the 4022 Arithmetic Unit, this function is combined with the next subtraction, which takes 2^{n-1} (d) from the dividend:

$$+2^n (d) - 2^{n-1} (d) = 2^{n-1} (2d - d)$$
$$= 2^{n-1} (d).$$

This number is generated by adding the divisor to the dividend at the 2^{n-1} position.

Example 2 more closely resembles the mechanics of the division process within the Arithmetic Unit. For comparison, the same values (7 ÷ 3) used in Example 1 are used in Example 2.

```
                         0010
                 ________________
  0     011) 0         000111
             1         101           - 1
     [0] <-- 1         101           - 2
                       0011          - 3
      [0] <----------  1110          - 4
                        0011         - 5
       [1] <---------   0000         - 6
                         1101        - 7
        [0] <--------    1110        - 8
                         0011        - 9
                         0001        -10
```

1. Subtract divisor.
2. Negative result, therefore, 0 → quotient (Sign).
3. Shift left dividend and add divisor.
4. Negative result, therefore, 0 → quotient.
5. Shift left dividend and add divisor.
6. Positive result, therefore, 1 → quotient.
7. Shift left dividend and subtract divisor.
8. Negative result, therefore, 0 → quotient.
9. Add divisor for remainder "fix-up".
10. Remainder.

Example 2

By comparing Example 2 with Example 1, notice that 4 less steps are required to perform the division in Example 2. Steps 3, 4, and 5 of Example 1 are performed in step 3 of Example 2, providing the same result. That is, when a negative result from a subtraction was obtained in Example 1, the divisor was added back to restore the dividend, then the dividend was shifted and the divisor subtracted. In Example 2, when a negative result from a subtraction is obtained, this negative result is shifted and the divisor added to it. The final result of each operation is the same (i. e., step 6 of Example 1 has the same result as step 4 in Example 2). In this manner, a much faster divide cycle is obtained.

In both Examples 1 and 2, notice that the first subtraction is used to determine the sign bit of the quotient. The divide cycle time in the 4022 Arithmetic Unit is further decreased by eliminating this step. Within the Arithmetic Unit, the sign of the quotient is determined by comparing the sign of the divisor with the sign of the dividend. If the signs are alike, the sign bit is "zero" (positive); if the signs are unlike, the sign bit is "one" (negative). Therefore, the first subtraction is not required to determine the sign of the quotient.

Example 3 illustrates the mechanics of division by comparing the signs of the divisor and dividend to determine the sign of the quotient. The same values (7 ÷ 3) used in Examples 1 and 2 are used for comparison.

```
                                0010
                     ___________________
[0] <- 0      011)  0         000111      - 1
                              1101        - 2
              [0] <------     1110        - 3
                               0011       - 4
                [1] <-----     0000       - 5
                                1101      - 6
                 [0] <-----     1110      - 7
                                0011      - 8
                                0001      - 9
```

1. Sign of divisor and dividend alike ∴ 0 → quotient.
2. Shift left dividend and subtract divisor.
3. Negative result, therefore, 0 → quotient.
4. Shift left dividend and add divisor.
5. Positive result, therefore, 1 → quotient.
6. Shift left dividend and subtract divisor.
7. Negative result, therefore, 0 → quotient.
8. Add divisor for remainder "fix-up".
9. Remainder.

Example 3

COMMAND DESCRIPTION

Fig. DVD. 1 contains a basic flow chart of the Sequence Control States required to execute the DVD command and lists the basic functions performed within each Sequence State. Block diagrams of States 3, 4, and 5 are contained in Fig. DVD. 2.

Sequence State 1

The DVD command is "fetched" from memory during a normal Sequence State 1. At Last Pulse of State 1, the

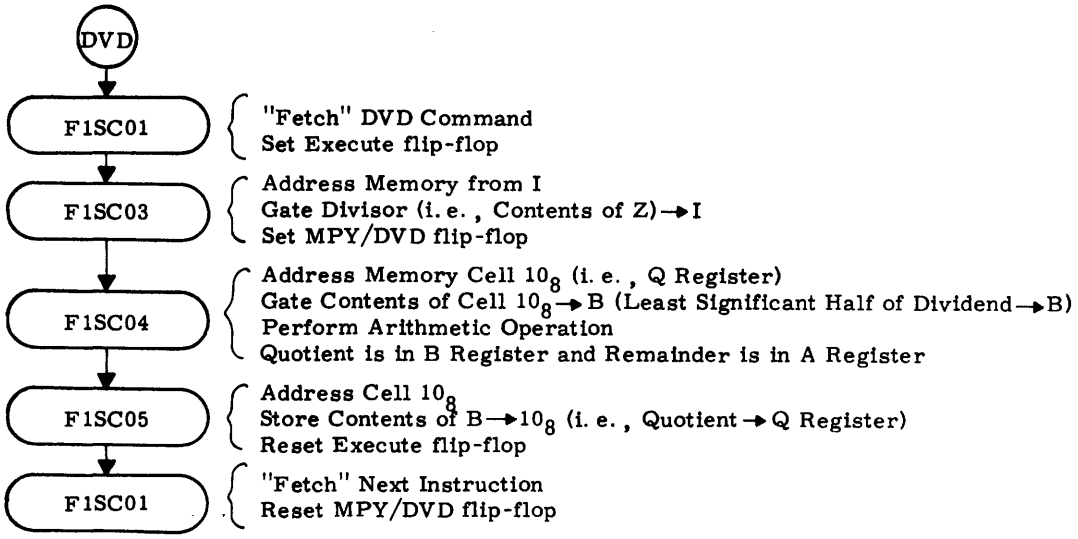


Fig. DVD. 1 DVD Basic Flow Chart

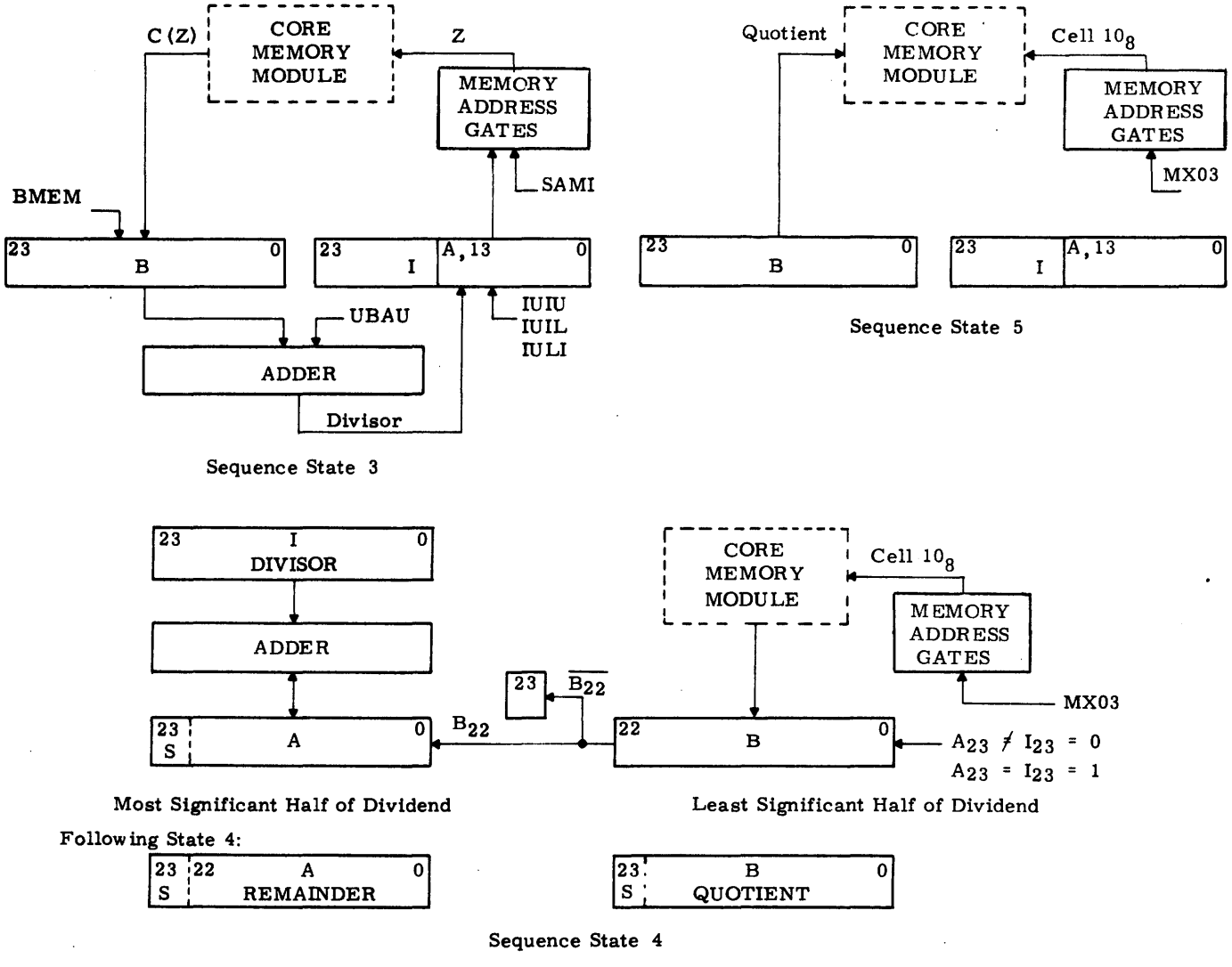


Fig. DVD. 2 DVD Block Diagram

Execute flip-flop (F1XEXC) is set. Therefore, at the end of State 1, the DVD command is contained by the I Register and the Execute flip-flop is set. Following State 1, a non-indexed DVD command enters Sequence State 3.

Sequence State 3

A timing diagram, including logic equations, for State 3 is contained in Fig. DVD. 3. During State 3, memory is addressed from $I_{A,13-0}$ (D1SAMI) and the contents of memory location Z (divisor) are gated to the B Register. From the B Register, the divisor is gated to the Adder (D1UBAU) and from the Adder to the I Register (D1IUIU, D1IUIL, D1IULI). At Time 3, the Execute MPY/DVD flip-flop (F1XMDV) is set to indicate that the I Register does not contain an instruction and to provide control for the DVD operation. The DVD command is differentiated from the MPY command by having both F1XMDV and F1XEXC (State 1) set. Following State 3, State 4 is entered.

Sequence State 4

Sequence State 4 is used to bring the least significant half of the dividend (Q_{22-0}) to the B Register and to perform the actual arithmetic operation. To provide these functions, the duration of State 4 is extended by entering Time 6 Envelope.

For ease of understanding, a detailed flow chart of State 4 is contained in Fig. DVD. 4. The logic elements and associated logic equation used to perform the individual steps of the flow chart are contained in Table DVD. 1.

An example of the arithmetic operation performed in State 4 is provided in Table DVD. 2. For simplicity, 9-bit registers are illustrated. The timing diagram of State 4, contained in Fig. DVD. 5, illustrates the timing associated with this example. Using these aids, little difficulty should be encountered in determining the operation of the Arithmetic Unit for any dividend and divisor values.

During the first portion of State 4, memory cell 10_8 is addressed (G0MX03) and the contents of the Q Register are gated to the B Register. Bits 22-0 of the Q Register contain the least significant half of the dividend.

The J Counter is initially cleared and then preset to 7 (G1JP07). The J Counter is then incremented at each shift of the A and B Registers. The count value of the J Counter is then used to determine when the divide cycle is completed ($J = 37_8$). The Delay Time Counter is initially preset to 30_8 and provides timing control for determining when to add (or subtract) and when to shift for each quotient bit generated.

During Time 3 Envelope, the signs of the divisor (I_{23}) and dividend (A_{23}) are compared to determine the sign bit of the quotient and to provide control for the Overflow check. If the signs are alike, the Strings flip-flop (F1XSTG) is reset; if the signs are unlike, the Strings

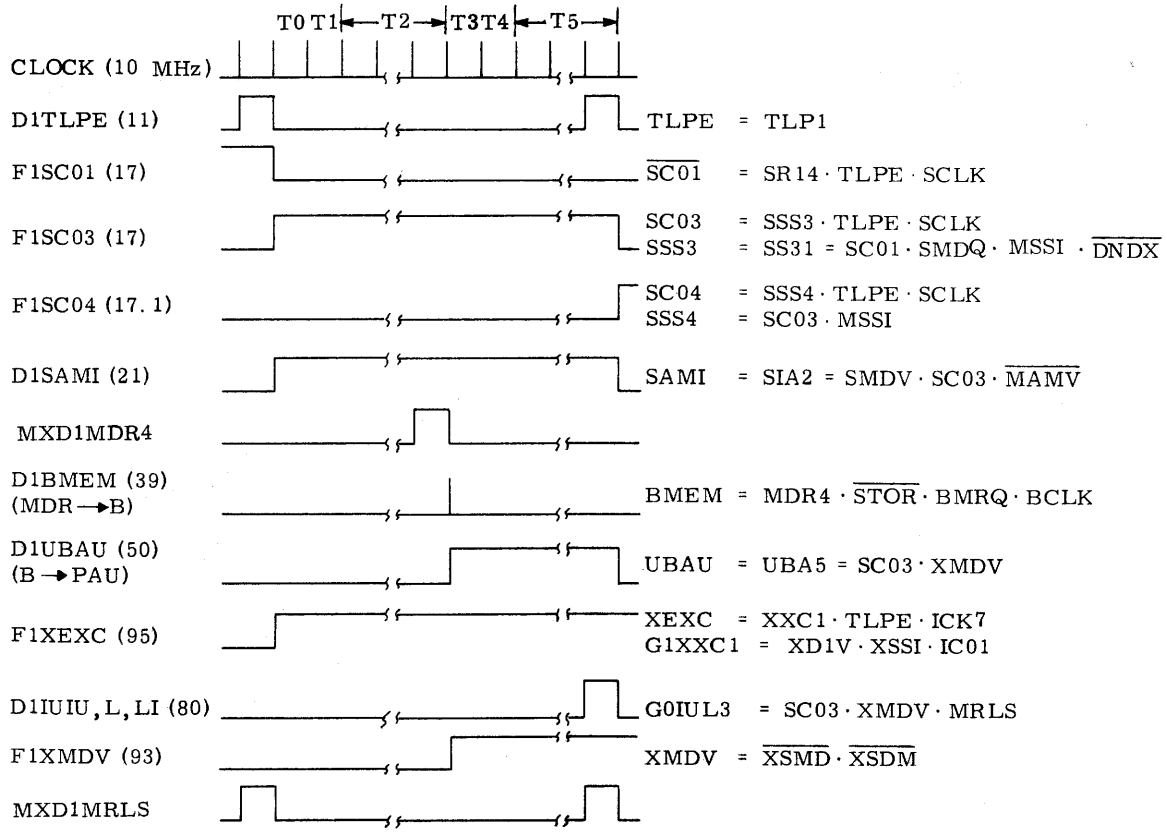


Fig. DVD. 3 DVD Sequence State 3 Timing

flip-flop is set. Also, if the signs are alike, B_0 is armed to set at the first shift. This "one" will represent a positive quotient, since the complement of this bit will be shifted to B_{23} at the end of the DVD operation. If the signs are unlike, B_0 will be armed to reset at the first shift, specifying a negative quotient.

The Overflow check is used to determine if the value of the quotient will be too large to be contained in a 24-bit register. If the magnitude of the A Register is greater than the contents of the I Register, or the contents of the A and I are equal and A is greater than or equal to zero the quotient will exceed the capacity of 24 bits and the Overflow flip-flop (F1UOFL) is set. The Overflow test is made by performing either an addition or subtraction of the A and I Registers. If the signs of A and I are alike (Strings flip-flop reset), I is subtracted from A. If the sign of the result in the Adder does not change, then A is greater than I, or A and I are equal and A is greater than or equal to zero and the Overflow flip-flop is set. If I and A have unlike signs (Strings flip-flop set), then I and A are added. If the sign of the result in the Adder is equal to the sign of A, A is greater than or equal to I and the Overflow flip-flop is set. If, however, following the addition or subtraction the sign of the result is not the same as the sign of A, the quotient may be contained in 24 bits and a valid division may take place.

After the test for Overflow, the A and B Registers are shifted left with the complement of the sign bit, as determined by like or unlike signs, gated to B_0. At the shift, the complement of B_{22} is gated to B_{23} and the true output of B_{22} is gated to A_0. The J Counter is incremented to indicate that one of the 24 quotient bits (sign) has been determined.

Time 6 Envelope is then entered. The Delay Time Counter is preset to 26_8 and the first addition or subtraction is performed to determine the first data quotient bit and to generate the partial remainder. If the most significant bit of the A Register and the sign of the I Register were alike prior to the last shift (Strings flip-flop reset), a subtraction is performed. If prior to the last shift, the most significant bit of A and the sign of I were unlike (Strings flip-flop set), an addition is performed. The result of this addition or subtraction is gated to the A Register and the most significant bits of A and I are again compared. If they are alike, a valid subtraction (in effect) was performed and the quotient bit (B_0) is armed to set at the shift. If the most significant bits are unlike, an invalid subtraction (in effect) was performed and the quotient bit (B_0) is armed to reset. The Strings flip-flop is also armed to set (if most significant bits are unlike) or reset (if most significant bits are alike) at the next shift to provide control for the next subtraction.

The A and B Registers are then shifted left with the quotient bit determined above gated to B_0. The complement of B_{22} is gated to B_{23} and the true output of B_{22} is gated to A_0. At the shift, the J Counter is incremented to indicate that a quotient bit has been determined.

Each data bit of the quotient is determined in this manner until the last data bit has been determined ($J = 36_8$). When the last data bit has been determined, only the B Register is shifted left. This gates the final data bit to B_0, the sign bit to B_{23}, and preserves the partial remainder in the A Register. At this shift, the J Counter is incremented to 37_8, indicating that the required number of shifts (24) have occurred.

Following the 24 shifts, the quotient is contained in the B Register, and the "fix-up" of the remainder, if required, is performed. If, prior to the last shift, the sign of the divisor (I_{23}) and the sign of the remainder (A_{23}) were unlike, the Strings flip-flop was set at the shift of B. If the Strings flip-flop is set, then one too many subtractions were performed and I is added to A to provide the correct remainder. If the Strings flip-flop is reset, the contents of A are gated to the Adder and back to A unchanged.

Following State 4, State 5 is entered to store the quotient contained in the B Register, in cell 10_8 (Q Register).

Sequence State 5

A timing diagram and logic equations for State 5 are contained in Fig. DVD. 6.

Memory cell 10_8 is addressed (G0MX03) and the quotient is gated from the B Register to the Q Register. At Last Pulse of State 5, the Execute flip-flop (F1XEXC) is reset.

During State 1 of the command following DVD, the Execute MPY/DVD flip-flop (F1XMDV) is reset.

NOTE

The previous description applies to both positive and negative values. However, when the divisor is negative, the quotient and remainder must be modified to obtain the expected result. If the divisor is negative, a positive one must be added to the quotient and the divisor must be subtracted from the remainder to obtain the expected result. The following example (using decimal numbers) illustrates the quotient and remainder obtained, proves that it is correct, and how it may be modified to obtain the expected result.

Decimal Problem: $\frac{+4}{-2}$

Result Obtained:
Quotient = -3, Remainder = -2

Prove Result is Correct:
Divisor x Quotient + Remainder = Divident
(-2) x (-3) + (-2) = +4

Result Modification:
Add +1 to Quotient; -3 + (+1) = -2
Subtract Divisor from Remainder:
(-2) - (-2) = 0.

As previously described, this is of little consequence in actual mathematical operations within the computer and normally the quotient is used without any "fix-up".

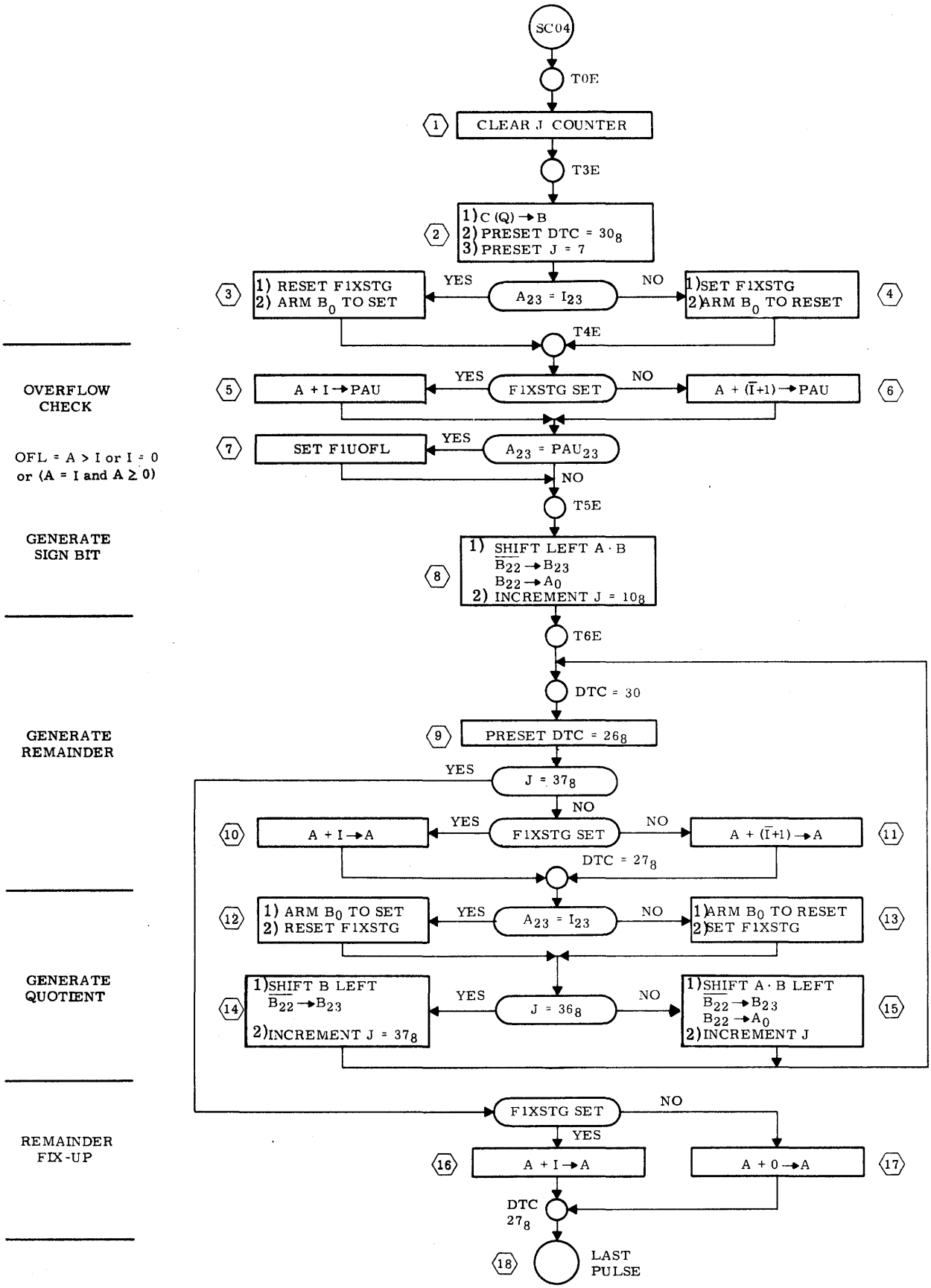


Fig. DVD. 4 State 4 Flow Chart

(1) Clear J Counter:

D0JJE0 (70) = JCK1· $\overline{\text{TSCA}}$· $\overline{\text{TSCB}}$· JS4G
G1JS4G (70) = SC04· DMD1
G0DMD1 = XMDV

(2) Preset Delay Time Counter to 30_8:

G1TPAF (12) = TP30 + TD30
G0TP30 (12) = TSCB· TSCA· TSCC· DMGT

Preset J Counter to 7:

G1JP07 (70) = TP30· XMDV

(3) Reset Strings flip-flop:

$\overline{\text{F1XSTG}}$ (94) = $\overline{\text{G0XCA1}}$· XDVT· ICK7
$\overline{\text{G0XCA1}}$ (94) = (XA23· $\overline{\text{XI23}}$) + (AR23· IR23)
G1XDVT (94) = UDVI· XT3E

Arm B_0 to set at first shift ((8)):

G1BSL2 (38) = DDIV· [(BI23· BA23) + ($\overline{\text{AR23}}$· $\overline{\text{IR23}}$)]

(4) Set Strings flip-flop:

F1XSTG (94) = XCAI· XDVT· ICK7
G1XCA1 (94) = (XA23· $\overline{\text{IR23}}$) + (XI23· $\overline{\text{AR23}}$)

Arm B_0 to reset at first shift ((8)):

G1BLS1 (38) = DDIV· [(BI23· $\overline{\text{AR23}}$) + (BA23· $\overline{\text{IR23}}$)]

(5) Add A and I:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA4 = UC04· UDIV· XSTG

(6) Subtract I from A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UINA (51) = UIN1 = UC04· UDIV· $\overline{\text{XSTG}}$· $\overline{\text{JE37}}$
D1UENC (51) = UIN1

(7) Set Overflow flip-flop at shift ((8)):

F1UOFL (54) = UFL5· UFL6· ECLK
G1UFL5 (54) = (UA23· U23S) + ($\overline{\text{AR23}}$· U23S)
G1UFL6 (54) = ABSL = DDIV· JC04· TT5E· MRLS

(8) Shift left A and B Registers one place:

D1BSLL, U, 1 (40) = ABSL
G0ABSL (63) = DDIV· JC04· TT5E· MRLS
D1ASLL, U (63.1) = ABSL

Gate $\overline{B_{22}}$ to B_{23} at shift:

G1B3LD (38) = BR22· BDIV
G1B2LD (38) = $\overline{\text{BR22}}$· BDIV

Gate B_{22} to A_0 at shift:

G1AMLN (64) = AML1· $\overline{\text{BR22}}$
G1AMLD (64) = AML1· AB22

Increment J Counter:

N1JINC (71) = G0JINC = XMDV· TT5E· MRLS· JC04

(9) Preset the Delay Time Counter to 26_8:

G1TP26 (12) = TD30· $\overline{\text{JE07}}$· XMDV· DDIV

(10) Add I to A and gate result to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA4 = UC04· UDIV· XSTG
D1AAUU, L (62) = AAU2 = ECLK· TE26· AMDV· JE37

(11) Subtract I from A and gate result to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UINA (51) = UIN1 = UC04· UDIV· $\overline{\text{XSTG}}$· $\overline{\text{JE37}}$
G1UENC (51) = UIN1

(12) Arm B_0 to set at shift:

G1BLS2 (38 = BDIV· [(BI23· BA23) + ($\overline{\text{AR23}}$· $\overline{\text{IR23}}$)]

Reset Strings flip-flop at shift:

$\overline{\text{F1XSTG}}$ (94) = XDVT· $\overline{\text{XCA1}}$· ICK7
G1XDVT (94) = UDIV· ASLB
$\overline{\text{G1XCA1}}$ (94) = (AR23· IR23) + ($\overline{\text{AR23}}$· $\overline{\text{IR23}}$)

(13) Arm B_0 to reset at shift:

G1BLS1 (38) = BDIV· [(BI23· $\overline{\text{AR23}}$) + (BA23· $\overline{\text{IR23}}$)]

Set Strings flip-flop at shift:

F1XSTG (94) = XCA1· XDVT· ICK7
G1XCA1 (94) = (AR23· $\overline{\text{IR23}}$) + ($\overline{\text{AR23}}$· IR23)

(14) Shift B left one place:

D1BSLL, U, 1 (40) = ASLB
G0ASLB (63) = DDIV· TE27· TT6E· $\overline{\text{JE37}}$

Gate $\overline{B_{22}}$ to B_{23} at shift:

G1B3LD (64) = BR22· BDIV
G1B2LD (64) = $\overline{\text{BR22}}$· BDIV

Increment the J Counter to 37_8:

N1JINC (71) = G0JINC = $\overline{\text{JE37}}$· TT6E· XMDV· TEFF

(15) Shift left A and B one place:

D1BSLL, U, 1 (40) = ASLB
G0ASLB (63) = DDIV· TE27· TT6E· $\overline{\text{JE37}}$
D1ASLL, U (63.1) = ASL1
G0ASL1 (63) = DDIV· $\overline{\text{J367}}$· TT6E· TE27

Gate the complement of B_{22} to B_{23}:

G1B3LD (38) = BR22· BDIV
G1B2LD (38) = $\overline{\text{BR22}}$· BDIV

Gate B_{22} to A_0:

G1AMLN (64) = AML1· $\overline{\text{BR22}}$
G1AMLD (64) = AML1· AB22

Increment the J Counter:

N1JINC (71) = G0JINC = $\overline{\text{JE37}}$· TT6E· XMDV· TEFF

(16) Add A to I and gate result to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA4 = UC04· UDIV· XSTG
D1AAUU, L (62) = ECLK· TE26· AMDV· JE37

(17) Gate A to Adder and back to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1AAUU, L (62) = ECLK· TE26· AMDV· JE37

(18) Generate Last Pulse:

D1TLPE (11) = TLP3 = JE37· AMDV· TE27· TT6E

Table DVD.1 State 4 Flow Chart Equations

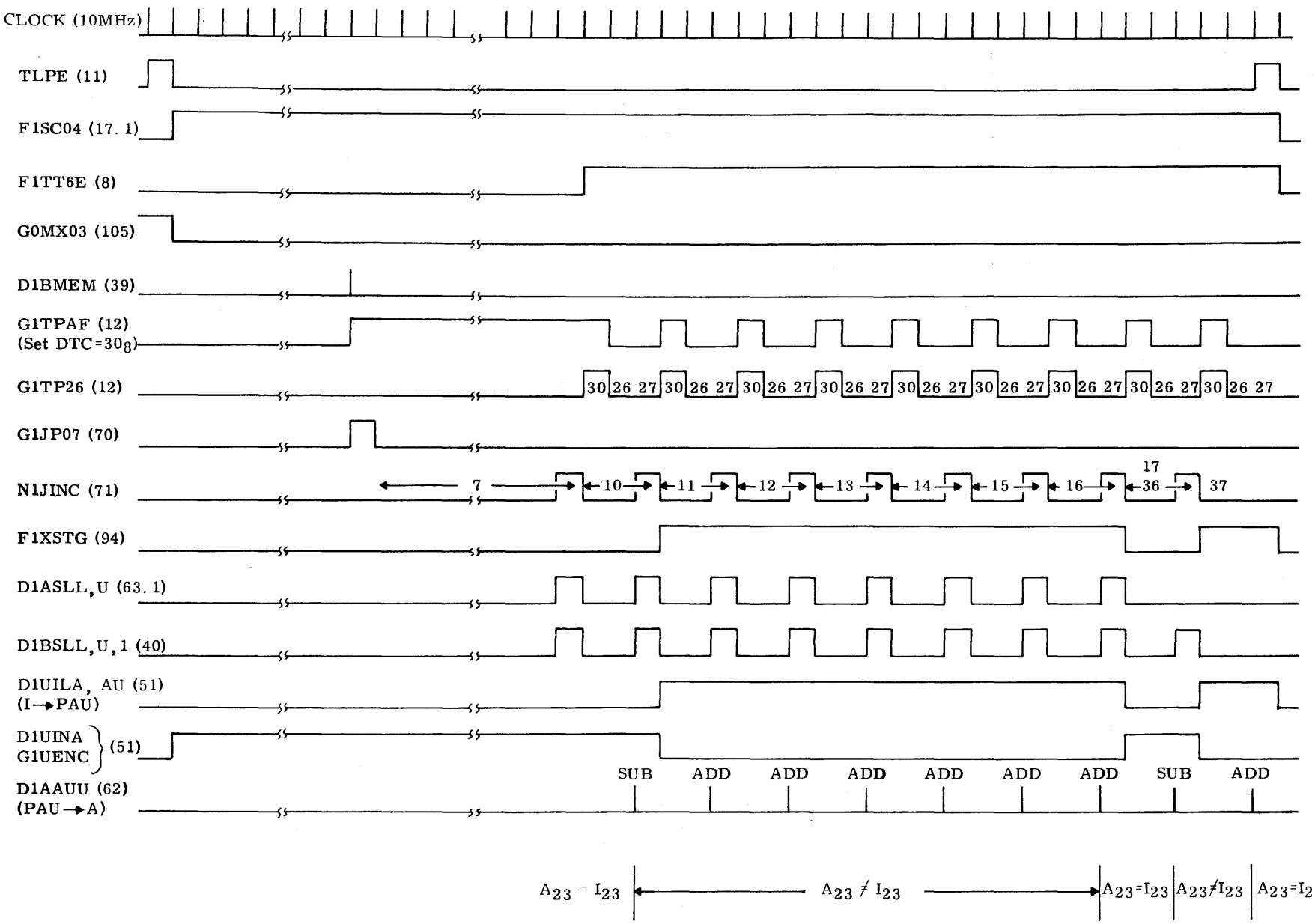


Fig. DVD. 5 DVD Sequence State 4 Example (+27 ÷ +13)

	S										
I =	0	0	0	0	0	1	1	0	1	=	+13
A =	0	0	0	0	0	0	0	0	0	=	0
B =	0	0	0	0	1	1	0	1	1	=	+27

$$+13\overline{)+27}\quad 2,\ 26,\ 1$$

J COUNT	DTC	XSTG	A REGISTER S 7 6 5 4 3 2 1 0	B REGISTER S 7 6 5 4 3 2 1 0	B_0 INPUT	ACTION
7	30	0	0 0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 1 1	1	T3E - $A_{23} = I_{23}$
7	30		0 0 0 0 0 0 0 0 0	0 0 0 0 1 1 0 1 1	1	T4E - Overflow Test
10	30		0 0 0 0 0 0 0 0 0	1 0 0 1 1 0 1 1 1		T5E - Shift - $\overline{\text{Sign}}$
10	26		1 1 1 1 1 0 0 1 1		0	$\overline{\text{XSTG}}$ = Subtract
11	27	1	1 1 1 1 0 0 1 1 0	1 0 1 1 0 1 1 1 0		Shift
	30					
	26		1 1 1 1 1 0 0 1 1		0	XSTG = Add
12	27	1	1 1 1 1 0 0 1 1 0	1 1 1 0 1 1 1 0 0		Shift
	30					
	26		1 1 1 1 1 0 0 1 1		0	XSTG = Add
13	27	1	1 1 1 1 0 0 1 1 1	0 1 0 1 1 1 0 0 0		Shift
	30					
	26		1 1 1 1 1 0 1 0 0		0	XSTG = Add
14	27	1	1 1 1 1 0 1 0 0 1	0 0 1 1 1 0 0 0 0		Shift
	30					
	26		1 1 1 1 1 0 1 1 0		0	XSTG = Add
15	27	1	1 1 1 1 0 1 1 0 0	1 1 1 1 0 0 0 0 0		Shift
	30					
	26		1 1 1 1 1 1 0 0 1		0	XSTG = Add
16	27	1	1 1 1 1 1 0 0 1 1	0 1 1 0 0 0 0 0 0		Shift
	30					
	26		0 0 0 0 0 0 0 0 0		1	XSTG = Add
17=36	27	0	0 0 0 0 0 0 0 0 1	0 1 0 0 0 0 0 0 1		Shift
	30					
	26		1 1 1 1 1 0 1 0 0		0	$\overline{\text{XSTG}}$ = Subtract
37	27	1		0 0 0 0 0 0 0 1 0		Shift B Only
	30					
	26		0 0 0 0 0 0 0 0 1			XSTG = Add = Remainder Fix-Up
	27					Last Pulse

Remainder = 1 (A register, bit 0)

Quotient = 2 (B register, bits 1 0)

Table DVD. 2 Divide Example (27 ÷ 13)

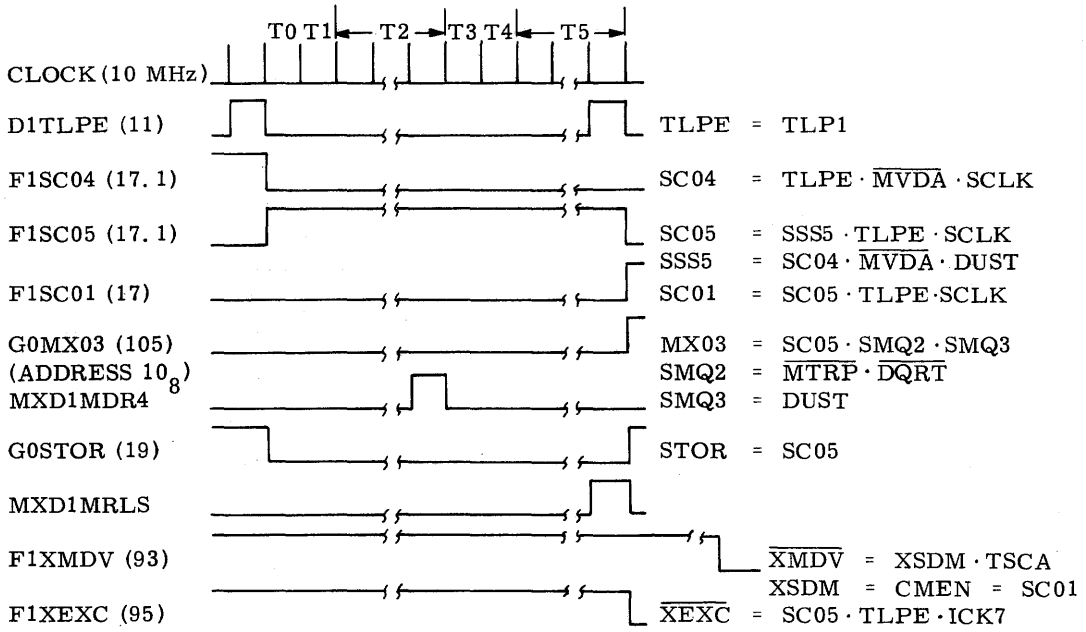


Fig. DVD.6 Sequence State 5 Timing

ERA - EXCLUSIVE OR TO A

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
10	X	*	Y

Z = f (X, *, Y)

ERA compares the corresponding bits of A with those of core cell Z. If the corresponding bits of both A and Z are alike, a "zero" is placed in that position of A. If the corresponding bits of A and Z are not alike, a "one" is placed in that position of A.

A non-indexed ERA command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The contents of memory location Z are then gated to the B Register by D1BMEN during the clock pulse of Memory Data Ready (MXD1MDR4). From B, the contents of memory location Z are gated to the Adder Unit (D1UBAU). At the same time, the contents of the A Register are gated to the Adder Unit (D1UAAU). The Exclusive OR control signal, D0ULXR, is also applied to the Adder Unit to enable the Exclusive OR function. In effect, the contents of memory cell Z and the A Register are summed in the Adder with carry generation inhibited by D0ULXR to provide the Exclusive OR result. The result is gated back to the A Register (D1AAUL, U) to complete the ERA execution cycle.

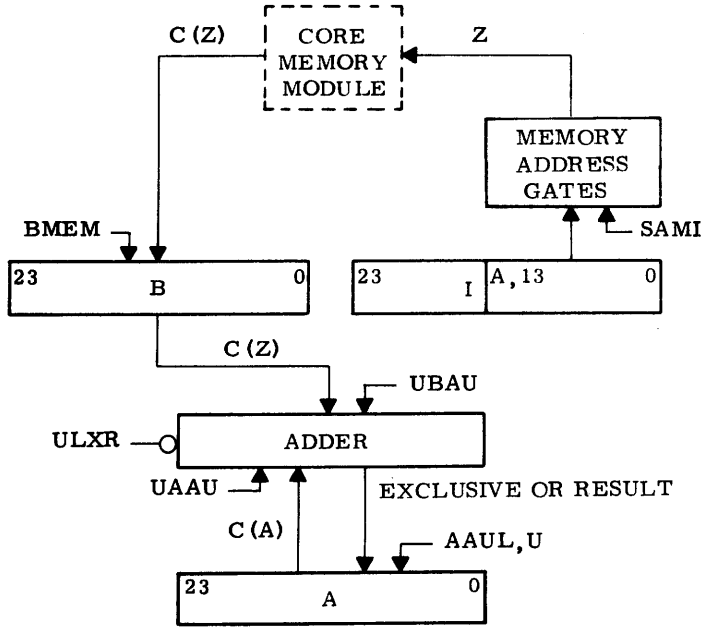
To exemplify the ERA comparison, consider the following; 4 bits are used for simplicity.

Contents of A Register = 0011
Contents of cell Z = 0101
Result Placed in A Register = 0110.

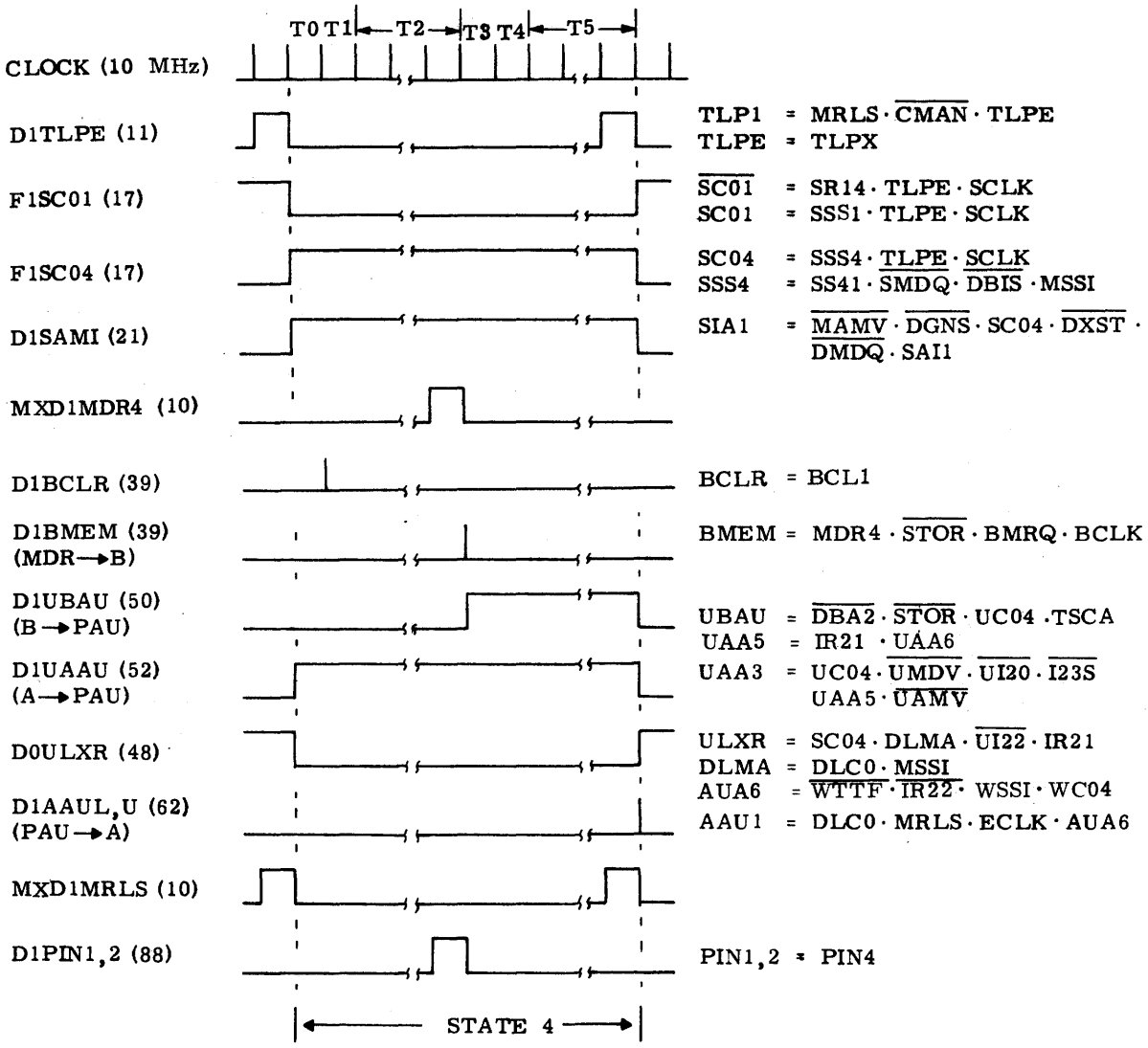
Non-Indexed Word Times.	2 (S1, S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$C(Z_{23-0}) \oplus C(A_{23-0})$
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

⊕ = Exclusive OR

COMMAND CHARACTERISTICS



ERA BLOCK DIAGRAM



ERA TIMING DIAGRAM

GEN 1 COMMANDS

GEN 1 commands are used for bit manipulation of the A Register. By controlling the operation of the serial Full Adder (G1AFNS), individual bits of the A Register may be shifted in position, masked by ones or zeros, tested for polarity, or counted for the number of ones or zeros contained therein, etc. Microcoding of the instruction may be manipulated to affect the J Counter and Test flip-flop as well as manipulate the A Register.

GEN 1 commands are identified by the operation code 05 (bits 23-18). They may be indexed, if desired, but cannot be relative addressed. Indexing of GEN 1 must be handled with care since it may change the microcoded action bits, thus changing the command functions. The microcoded action bits of the command format (bits 14-0) provide the logic signals necessary to perform the individual commands. Fig. GN1.1 illustrates the use of the microcoded bits and the control each provides. The instruction format is divided into microcoded categories as defined in the following:

05 = GEN 1 Instruction Octal Code.

X = Index Word Indicator.

C = Carry Control - Controls the Full Adder Carry flip-flop (F1AFNP) and, therefore, the Carry input of the Full Adder.

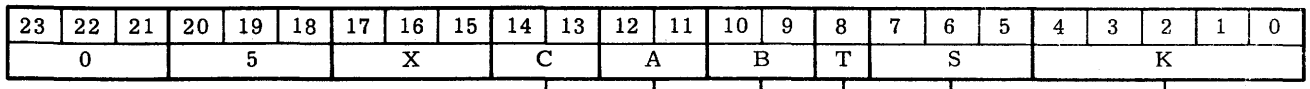
A = A Control - Controls the A input to the Full Adder.

B = B Control - Controls the B input to the Full Adder.

T = Test Control - 1 indicates Test flip-flop (F1ETST) operation.

- 0 indicates no Test flip-flop operation.

S = Shift Control.



FULL ADDER CARRY CONTROL		
14	13	FUNCTION
0	0	NORMAL CARRY
0	1	INITIAL SET ALLOW CARRY
1	0	HOLD CLEAR
1	1	HOLD SET

K BITS, THE COMPLEMENT OF WHICH ARE SET IN THE J COUNTER

FULL ADDER A INPUT CONTROL		
12	11	FUNCTION
0	0	NO INPUT
0	1	A0
1	0	$\overline{A0}$
1	1	A_{23}

8	7	6	5	FUNCTION
0	1			COUNT MOST SIGNIFICANT BITS
0		1		COUNT LEAST SIGNIFICANT BITS
0			1	STOP SHIFTING AT J = 37
1	1			INITIALLY SET TSTF = $\overline{\text{Bit 6}}$
1	0			NO INITIAL CHANGE OF TSTF
1		1		SET TSTF IF FULL ADDER-A INPUT = 1 AT TIME SPECIFIED BY BIT 5
1		0		CLEAR TSTF IF FULL ADDER-A INPUT = 1 AT TIME SPECIFIED BY BIT 5
1			1	PERFORM TEST WHEN J ≠ 37 AND STOP SHIFTING WHEN J = 37
1			0	PERFORM TEST WHEN J = 37; ALLOW 24-BIT SHIFT

1 = TSTF OPERATION
0 = NO TSTF OPERATION

FULL ADDER B INPUT CONTROL		
10	9	FUNCTION
0	0	NO INPUT
0	1	A0 WHEN J = 37
1	0	$\overline{A0}$ WHEN J = 37
1	1	1 WHEN J = 37

K = Length of shift or bit position of A Register designator.

There are 1024 unique GEN 1 instruction octal words. Thirty-nine of these have been determined to be useful enough to justify assigning them mnemonics and are listed in Table GN1.1. There are three categories into which these commands fall: bit manipulation, Test flip-flop operation, and shift A right. By associating the control bit configuration of Table GN1.1 to the control bit action of Fig. GN1.1, it may be seen how the various commands are implemented.

Basic timing for all GEN 1 commands is described below. Timing and Block Diagrams of the GEN 1 commands in Table GN1.1 denoted by shading, are provided to illustrate the operation of GEN 1 commands. These commands were selected as being representative of all GEN 1 commands. A general description of all GEN 1 commands is presented. Because of the similarity of commands within each group, little difficulty should be encountered in determining how any GEN 1 command is implemented. Fig. GN1.2 contains a flow chart of the functions performed during the execution of any GEN 1 command.

BASIC TIMING

All GEN 1 commands are "fetched" during a normal Sequence Control State 1. All Gen 1 commands are executed during Sequence Control State 4. Since the execution of GEN 1 commands do not require the use of memory, Memory Request (G1SMRQ) is inhibited during State 4. The basic timing of State 4 for all GEN 1 commands is the same. These basic timing signals are shown in Fig. GN1.3.

The Grey Code Sequence Time Counter (F1TSCA, B, C) is incremented by the first six clock pulses of State 4. That is, since memory is not requested, the Sequence Time Counter cannot and does not await Data Ready and Memory Release. Because GEN 1 commands require serial shifting of the $\overline{A}$ Register, State 4 is extended by using the Delay Time Counter (F1TAFF - F1TEFF) to allow sufficient time to shift the A Register and to determine when all 24 shifts have occurred.

Time 6 envelope (F1TT6E) is entered after the first six clock pulses of State 4. Each clock pulse of Time 6 envelope increments the Delay Time Counter until it contains 30_8. Allowing the Delay Time Counter to increment to 30_8 defines 24_{10} clock pulses that may be used to shift the A Register. When the Delay Time Counter has been incremented to 30_8, Last Pulse (D1TLPE) is enabled to end the execution of the GEN 1 command.

The timing diagram also illustrates that the J Counter is always cleared (D0JJE0) and the complement of bits 4 through 0 (K bits) of the I Register are always transferred to the J Counter. Operation of the J Counter and the function of the count value set in the J Counter depends on the microcoding of the GEN 1 instruction to be executed.

MNEMONIC	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIT MANIPULATION															
ADO	0	0	0	1	1	1	0	0	0	0	←	K			→
CBK	1	0	0	1	1	1	0	0	0	0	←	K			→
CLO	0	0	0	1	0	0	0	0	1	0	1	1	1	1	1
CLZ	1	1	1	0	0	0	0	0	1	0	1	1	1	1	1
CMO	0	0	0	1	0	0	0	1	0	0	1	1	1	1	1
CMZ	1	1	1	0	0	0	0	1	0	0	1	1	1	1	1
CNTO*	0	0	0	1	0	0	0	1	1	0	1	1	1	1	1
CNTZ*	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1
CPL	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
IBK	0	0	0	0	0	1	0	0	0	0	←	K			→
LBM	1	1	0	0	1	1	0	0	0	0	←	K			→
LDO	0	0	0	0	1	1	0	0	0	0	←	K			→
LDZ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LMO	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
NEG	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
RBK	1	0	0	1	0	1	0	0	0	0	←	K			→
SBK	1	0	0	1	1	0	0	0	0	0	←	K			→
TEST FLIP-FLOP OPERATION															
REV	1	1	1	0	0	0	1	0	0	0	←	K			→
RNZ	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0
ROD	0	0	0	1	0	0	1	0	0	0	←	K			→
RST	0	0	0	1	0	0	1	1	1	0	1	1	1	1	1
SET	0	0	0	1	0	0	1	1	0	0	1	1	1	1	1
SEV	1	1	1	0	0	0	1	0	1	0	←	K			→
SNZ	0	0	0	1	0	0	1	0	1	1	1	1	0	0	0
SOD	0	0	0	1	0	0	1	0	1	0	←	K			→
TER	1	0	0	1	0	1	1	1	0	0	←	K			→
TES	1	0	0	1	1	0	1	1	0	0	←	K			→
TEV	1	1	1	0	0	0	1	1	1	0	←	K			→
TNM	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0
TNZ	0	0	0	1	0	0	1	1	1	1	1	1	0	0	0
TOD	0	0	0	1	0	0	1	1	1	0	←	K			→
TOR	1	0	0	1	0	1	1	1	1	0	←	K			→
TOS	1	0	0	1	1	0	1	1	1	0	←	K			→
TSC	0	0	0	1	0	0	1	1	0	1	←	K			→
TZC	1	1	0	1	0	0	1	1	0	1	1	1	0	0	0
TZE	0	0	0	1	0	0	1	1	0	1	1	1	0	0	0
SHIFT A RIGHT															
SRA	0	0	1	1	0	0	0	0	0	1	←	K			→
SRC	0	0	0	1	0	0	0	0	0	1	←	K			→
SRL	0	0	0	0	0	0	0	0	0	1	←	K			→

Timing and Block diagrams of the GEN 1 commands denoted by shading are provided in the following text and are representative of the different types of GEN 1 commands.

* These commands are not recognized by the standard assembler.

Table GN1.1 Microcoded Bit Configuration of GEN 1 Commands

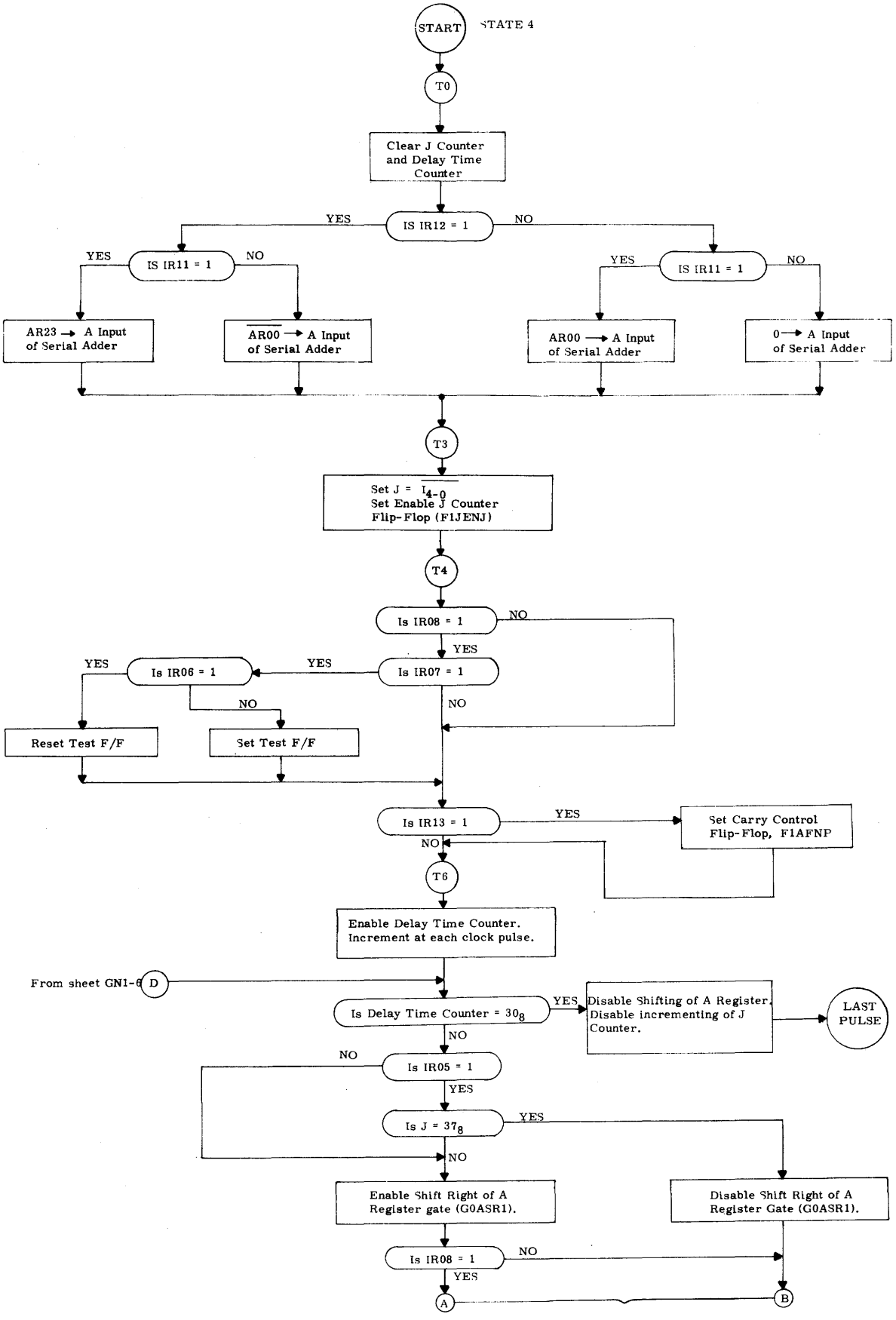


Fig. GN1.2 GEN1 Flow Chart

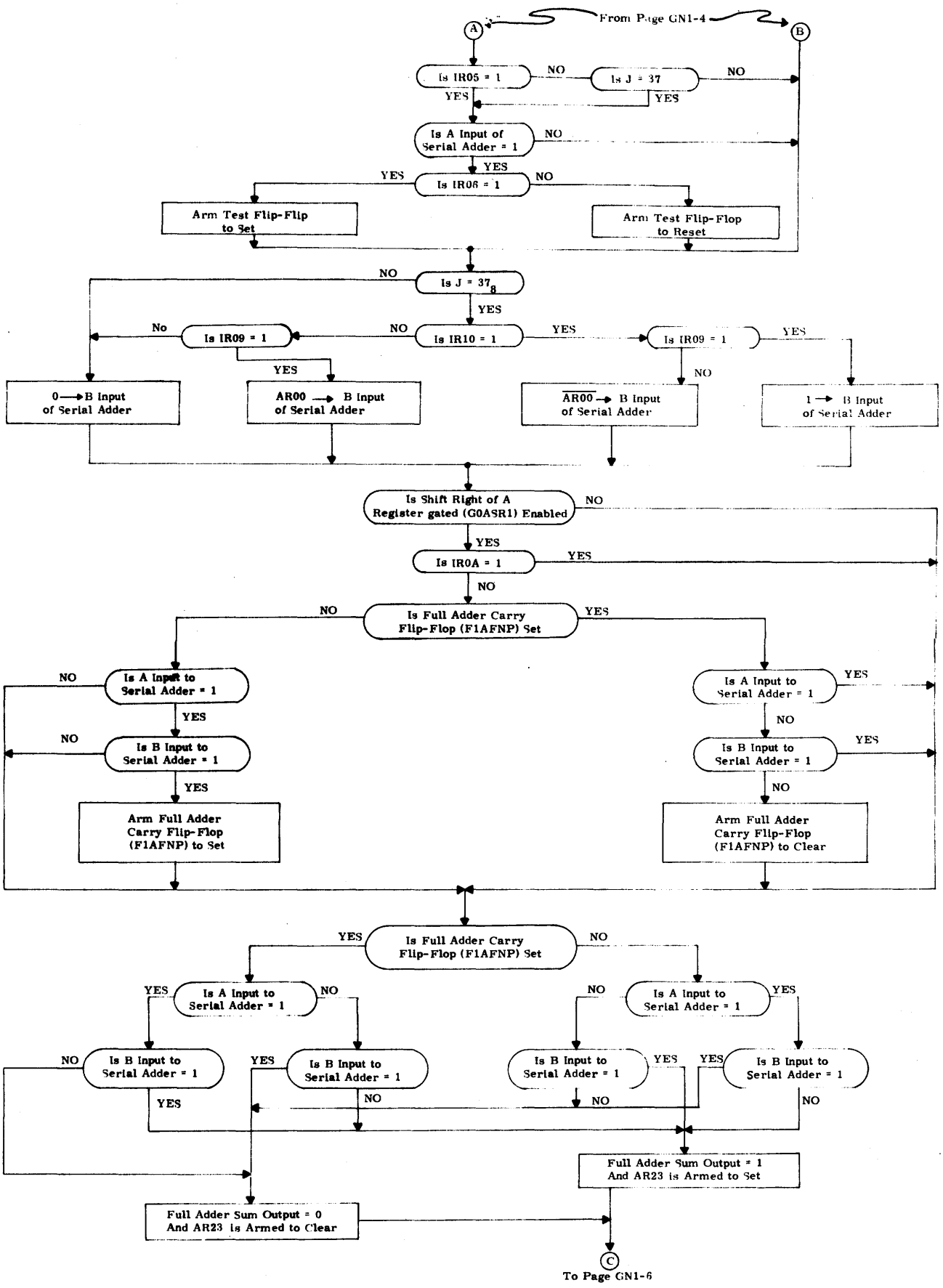
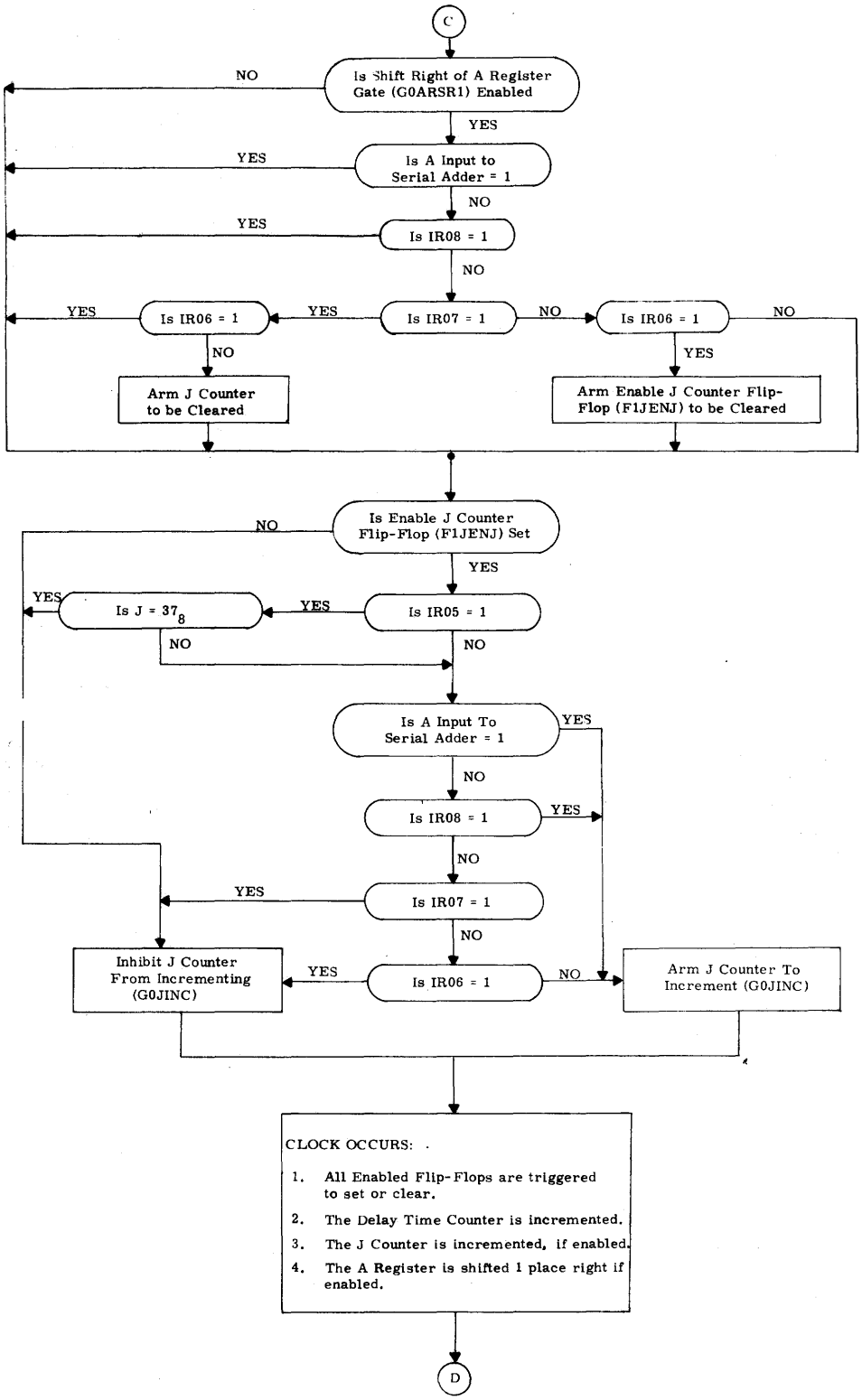


Fig. GN1.2 GEN1 Flow Chart

From Page GN1-5



To sheet GN1-4

Fig. GN1.2 GEN1 Flow Chart

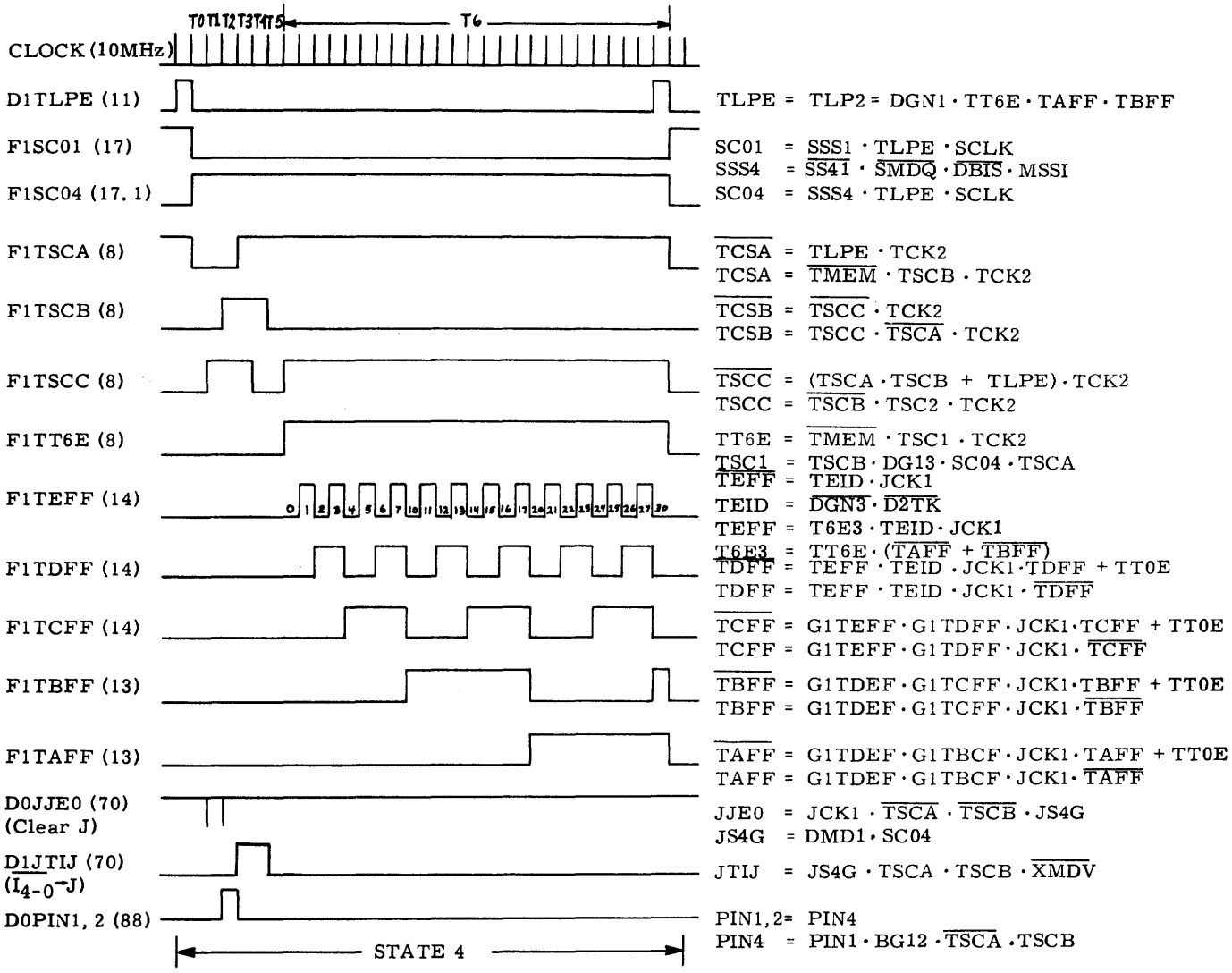


Fig. GN1.3 GEN 1 Basic Timing Diagram

ADO-ADD ONE TO BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	070	0	K

$Z = f(X, K)$
$Z \leq 37_8$

ADO adds a "one" bit to bit position Z in the A Register. Carries out of A_{23}, resulting from the summation, are lost but the Overflow flip-flop (F1UOFL) is not affected.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular thru the serial adder, with the J Counter incremented at each shift. When the J Counter is incremented to 37_8, bit position Z will be shifted from A_0 and a "one" is added to it in the serial adder. The result is shifted to A_{23}. The Carry flip-flop (F1AFNP) provides normal carry propagation if a carry resulted from the summation. The A Register continues to shift until 24 shifts have occurred, as determined by the Delay Time Counter.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$C(A) + 2^Z$
Q_{23-0}	
P_{15-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	JE37 · AGN1 · (IR10 · $\overline{AR00}$ + IR09 · AR00)	Enabled when J = 37_8 } Adds 1 to bit Z of A
G1AFNB	67	AI10 · AI09 · JE37 · DGN1	Enabled when J = 37_8 } Adds 1 to bit Z of A
G0AFNA	65	AR00 · ANA2	Gates A_0 to Adder
F1AFNP	66	$\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK AFNP = AFL3 · $\overline{AFNC}$ · BCLK	Normal Carry propagation
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. 24 times
N1J1NC	71	JIN2 · JIN4	Increment J 24 times

CBK-CHANGE BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	470	0	K

$Z = f(X, K)$
$Z \leq 37_8$

CBK complements bit Z of the A Register. All other bit positions of A remain unchanged. Z may be specified from 0 to 30 (decimal), however, if Z exceeds 23, A will be unchanged.

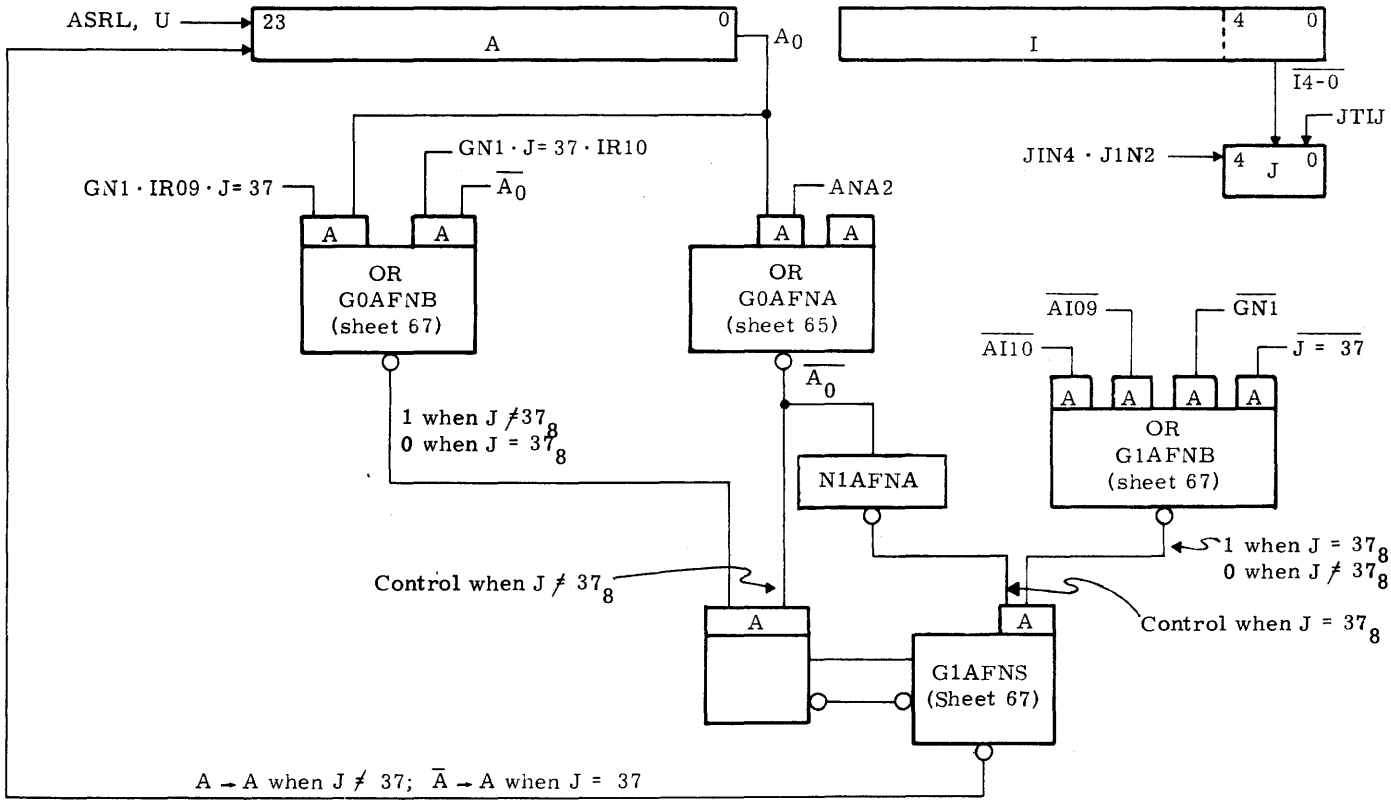
Timing and block diagrams of the CBK command are contained in Fig. GN1. 4, 5.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular thru the serial adder, with the J Counter incremented at each shift. When the J Counter is incremented to 37_8, bit Z will be shifted from A_0 and a "one" is added to it in the serial adder. The result is shifted to A_{23}. Any carry resulting from the summation is lost. In this manner, only bit Z is changed. The A Register continues to be shifted until 24 shifts have occurred, as determined by the Delay Time Counter.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$a_{23} \cdots \overline{a_Z} \cdots\cdots a_0$
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	JE37 · AGN1 · (IR10 · $\overline{AR00}$ + IR09 · AR00)	Enabled when J = 37_8
G1AFNB	67	AI10 · AI09 · JE37 · DGN1	Enabled when J = 37_8
G0AFNA	65	AR00 · ANA2	Gates A_0 to Adder
F1AFNP	66	$\overline{AFNP}$ = SC01 · BCLK	Inhibit carry propagation by IR0A
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. 24 places
N1J1NC	71	JIN2 · JIN4	Increment J 24 times



Condition	G0AFNB Output	G1AFNB Output	G0AFNA Output	F1AFNP Output	G1AFNS Output
A_0=1·J≠37	1	0	0	0	1
A_0=0·J≠37	1	0	1	0	0
A_0=1·J=37	0	1	0	0	0
A_0=0·J=37	0	1	1	0	1

Fig. GN1.4 CBK Block Diagram

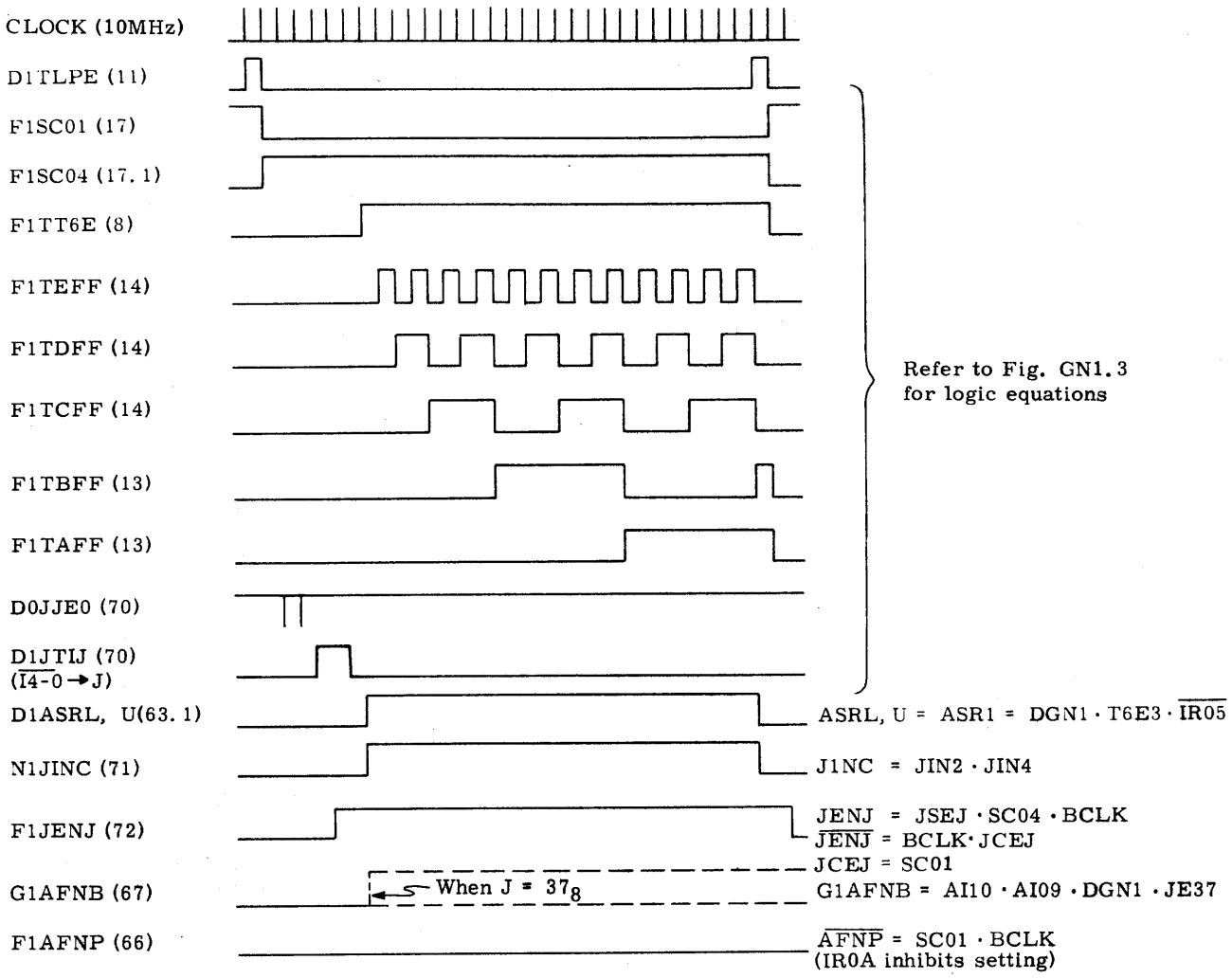


Fig. GN1.5 CBK Timing Diagram

CLO-COUNT LEAST SIGNIFICANT ONES

GEN 1

23							0
0	5	0	0	4	1	3	7

CLO counts the number of "one" bits to the right of the rightmost "zero" bit in the A Register. The count value is placed in the J Counter. If A equals 77777777_8, the count in J is 24_{10}.

> **NOTE**
>
> An LXC command must follow a CLO command before another GEN 1 command is executed. This is required to save the J Counter value.

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular thru the serial adder. The B inputs (G0AFNB, G1AFNB) are disabled during the CLO command. As the A Register is shifted right, each "one" bit increments the J Counter until the first "zero" bit is shifted from A_0. The first "zero" bit clears the Enable J flip-flop (F1JENJ) which inhibits further incrementation of the J Counter. Therefore, at the end of 24 shifts, the J Counter contains the number of least significant "one" bits in the A Register.

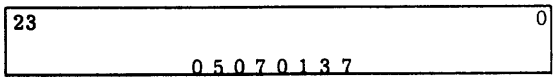
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number of rightmost "ones" in the A Register
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{G0AFNB} = (\overline{IR10} \cdot \overline{IR09}) + \overline{JE37}$	Disabled
G1AFNB	67	$\overline{G1AFNB} = (\overline{IR10} \cdot \overline{AR00}) + (\overline{IR09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JINC = JIN2 \cdot AFNA \cdot \overline{IR05}$ $JIN2 = F1JENJ \cdot TT6E \cdot (\overline{TAFF} + \overline{TBFF})$ $JENJ = JDEJ \cdot ASR1 \cdot AFNA$	Counts least significant "ones" from A_0

CLZ-COUNT LEAST SIGNIFICANT ZEROS

GEN 1



CLZ counts the number of "zero" bits to the right of the rightmost "one" bit in the A Register. The count value is placed in the J Counter. If A equals 00000000_8 the count in J is 24_{10}.

> **NOTE**
>
> An LXC command must follow a CLZ command before another GEN 1 command is excuted. This is required to save the J Counter value.

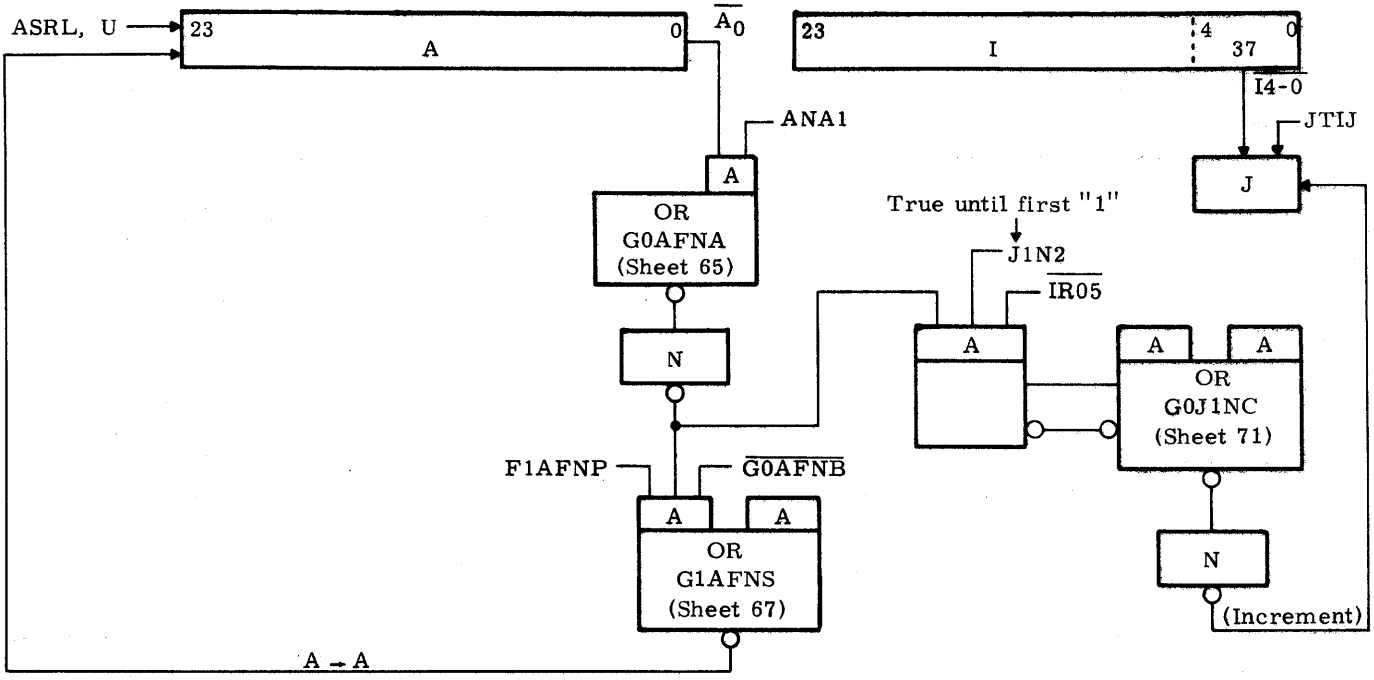
Timing and block diagrams of the CLZ command are contained in Fig. GN1. 6, 7.

Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number of rightmost "zeros" in the A Register
Memory Z	

COMMAND CHARACTERISTICS

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular with the complement of A_0 applied to the Adder. The Carry flip-flop (F1AFNP) is held set applying a continuous "one" to the Adder. In this manner, the Adder output is equal to the true output of A_0. The Adder output is shifted to A_{23} leaving the A Register unchanged after 24 shifts. As the A Register is shifted, the J Counter is incremented by each "zero" from the A Register until the first "one" is detected. The first "one" from A_0 clears the Enable J flip-flop (F1JENJ) which inhibits further incrementation of the J Counter. Therefore, at the end of 24 shifts, the J Counter will contain the number of least significant "zero" bits in the A Register.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$AFNB = (\overline{IR10} \cdot \overline{IR09}) + \overline{JE37}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{IR10} \cdot \overline{AR00}) + (\overline{IR09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00} \cdot ANA1$	Gates $\overline{A_0}$ to Adder
F1AFNP	66	AFL2	Held set, applies "1" to Adder
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shift A Reg. 24 places
N1J1NC	71	$JINC = JIN2 \cdot AFNA \cdot \overline{IR05}$ $JIN2 = F1JENJ \cdot TT6E \cdot (\overline{TAFF} + \overline{TBFF})$ $JENJ = JDEJ \cdot ASR1 \cdot AFNA$	Count least significant "zeros" from A_0



Condition	G0AFNB Output	G1AFNB Output	G0AFNA Output	F1AFNP Output	G1AFNS Output	F1JENJ Output
$A_0 = 1$	1	0	1	1	1	0
$A_0 = 0$	1	0	0	1	0	1

Fig. GN1.6 CLZ Block Diagram

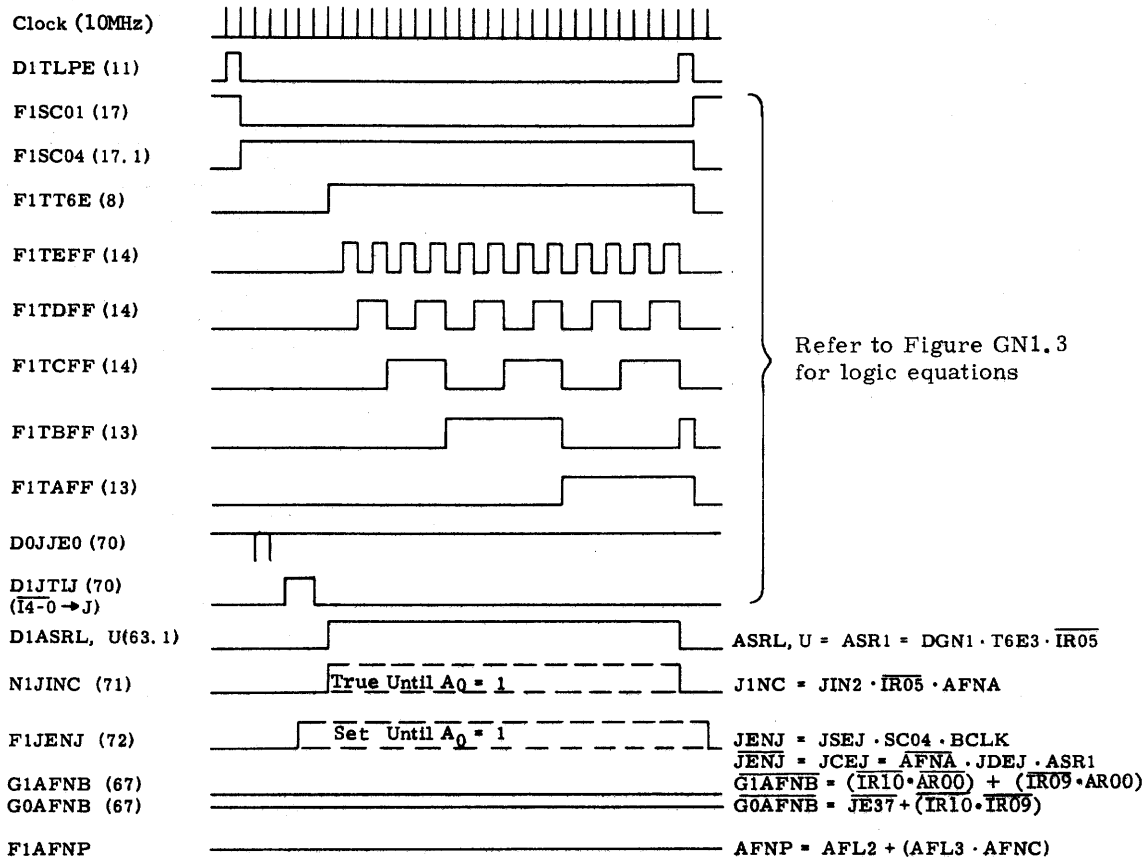


Fig. GN1.7 CLZ Timing Diagram

CMO-COUNT MOST SIGNIFICANT ONES

GEN 1

23		0
	0 5 0 0 4 2 3 7	

CMO counts the number of "one" bits to the left of the leftmost "zero" bit in the A Register. The count value is placed in the J Counter. If A equals 77777777_8, the count in J is 24_{10}.

> NOTE
>
> An LXC command must follow a CMO command before another GEN 1 command is executed. This is required to save the J Counter value.

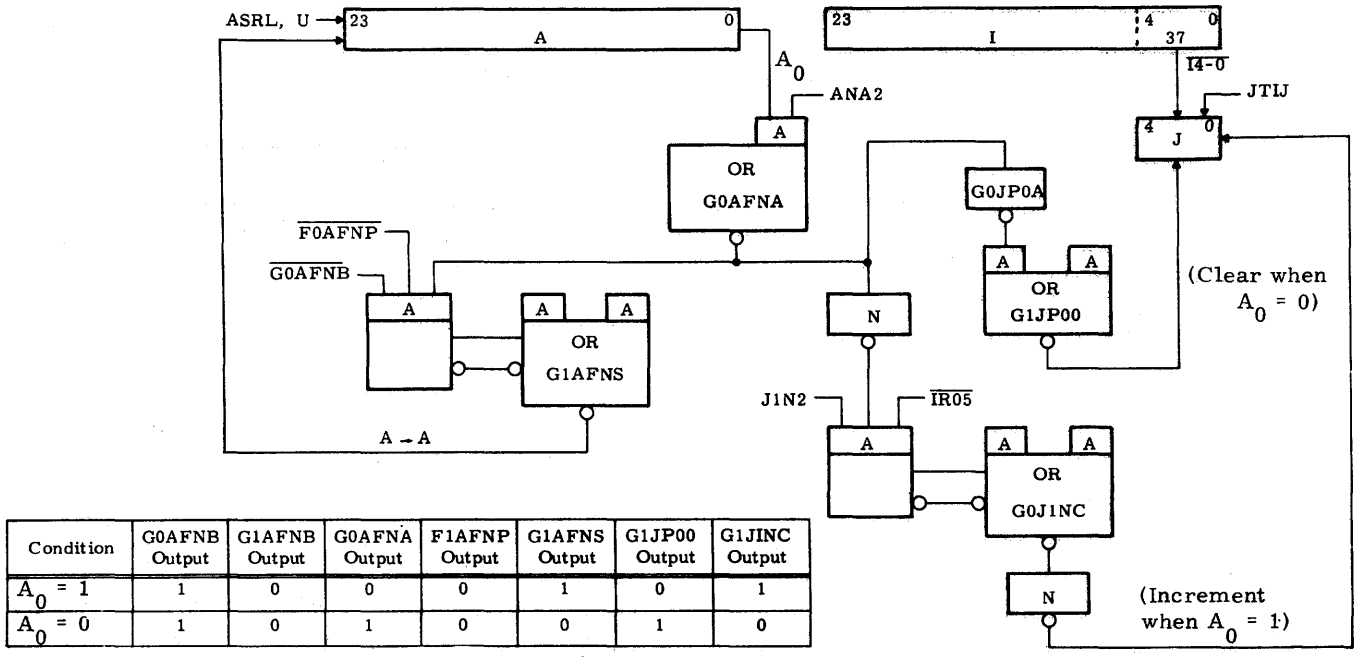
Timing and block diagrams of the CMO command are contained in Fig. GN1. 8, 9.

Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number of left most "ones" in the A Register
Memory Z	

COMMAND CHARACTERISTICS

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular with the true output of A_0 shifted thru the Adder. Each "zero" bit shifted from A_0 causes the J Counter to be reset. Each "one" bit shifted from A_0 causes the J Counter to be incremented. In this manner, after the 24 shifts of the A Register, the count in the J Counter corresponds to the number of most significant "one" bits.

Logic Element	Logic Sheet	Logic Equation	Function
G1AFNB	67	$\overline{G1AFNB} = (\overline{IR10} \cdot \overline{IR09}) + \overline{JE37}$	Disabled
G0AFNB	67	$\overline{G0AFNB} = (\overline{IR10} \cdot \overline{AR00}) + (\overline{IR09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63. 1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot AFNA \cdot \overline{IR05}$	Each "one" increments the J Counter
G1JP00	70	$ASR1 \cdot JP0A$ $JP0A = DGN1 \cdot \overline{IR06} \cdot IR07 \cdot \overline{IR08} \cdot \overline{AFNA}$	Each "zero" clears the J Counter



Condition	G0AFNB Output	G1AFNB Output	G0AFNA Output	F1AFNP Output	G1AFNS Output	G1JP00 Output	G1JINC Output
$A_0 = 1$	1	0	0	0	1	0	1
$A_0 = 0$	1	0	1	0	0	1	0

Fig. GN1.8 CMO Block Diagram

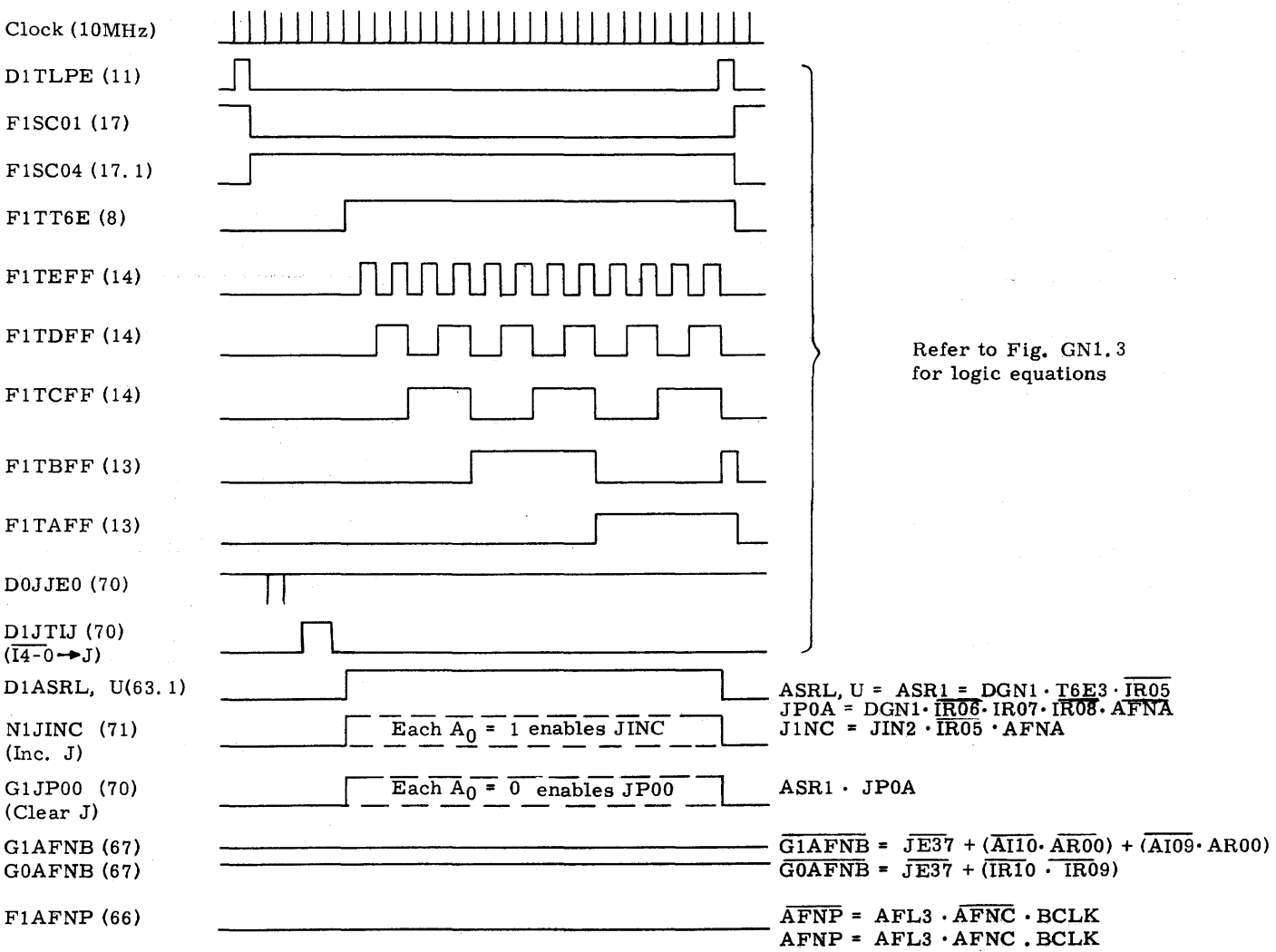


Fig. GN1.9 CMO Timing Diagram

CMZ-COUNT MOST SIGNIFICANT ZEROS

GEN 1

23								0
0	5	0	7	0	2	3	7	

CMZ counts the number of "zero" bits to the left of the leftmost "one" bit in the A Register. The count value is placed in the J Counter. If A equals 00000000_8, the count in J is 24_{10}.

> **NOTE**
>
> An LXC command must follow a CMZ command before another GEN 1 command is executed. This is required to save the J Counter value.

Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number of left most "zeros" contained in the A Register
Memory Z	

COMMAND CHARACTERISTICS

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular with the complement of A_0 applied to the Adder. The Carry flip-flop (F1AFNP) is held set applying a continuous "one" to the Adder. In this manner, the Adder output is equal to the true output of A_0. This Adder output is shifted to A_{23} leaving the A Register unchanged after 24 shifts. As the A Register is shifted, each "zero" bit increments the J Counter. Each "one" bit clears the J Counter. In this manner, after 24 shifts, the J Counter will contain the number of most significant "zero" bits contained by the A Register.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AFNB = $\overline{JE37}$ + ($\overline{IR10}$ · $\overline{IR09}$)	Disabled
G1AFNB	67	AFNB = $\overline{JE37}$ + $\overline{AI10}$ · $\overline{AR00}$) + ($\overline{AI09}$ · AR00)	Disabled
G0AFNA	65	$\overline{AR00}$ · ANA1	Gates $\overline{A_0}$ to Adder
F1AFNP	66	AFL2 + (AFL3 · AFNC · BCLK)	Held set applies a "1" to Adder
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Register right 24 places
N1J1NC	71	JIN2 · AFNA · $\overline{IR05}$	Each "zero" increments the J Counter
G1JP00	70	ASR1 · JP0A JP0A = DGN1 · $\overline{IR06}$ · IR07 · $\overline{IR08}$ · $\overline{AFNA}$	Each "one" clears the J Counter

CNTO-COUNT ONES

GEN 1

23																								0
0	5	0	0	4	3	3	7																	

CNTO counts the number of "one" bits in the A Register. The count value is placed in the J Counter. If A equals 77777777_8, the count in J is 24_{10}.

> NOTE
>
> An LXC command must follow CNTO command before another GEN 1 command is executed. This is required to save the J Counter value.

Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number "ones" in the A Register
Memory Z	

COMMAND CHARACTERISTICS

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular with the true output of A_0 shifted thru the Adder. Each "one" bit shifted from A_0 causes the J Counter to be incremented. In this manner, after the 24 shifts of the A Register, the count in the J Counter corresponds to the number of "one" bits in the A Register.

Logic Element	Logic Sheet	Logic Equation	Function
G1AFNB	67	$\overline{G1AFNB} = (\overline{IR10} \cdot \overline{IR09}) + \overline{JE37}$	Disabled
G0AFNB	67	$\overline{G0AFNB} = (\overline{IR10} \cdot \overline{AR00}) + (\overline{IR09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot AFNA \cdot \overline{IR05}$	Each "one" increments the J Counter

CNTZ-COUNT ZEROS

GEN 1

23								0
0	5	0	7	0	3	3	7	

CNTZ counts the number of "zero" bits in the A Register. The count value is placed in the J Counter. If A equals 00000000_8, the count in J is 24_{10}.

NOTE
An LXC command must follow a CNTZ command before another GEN 1 command is executed. This is required to save the J Counter value.

Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	Number of "zeros" contained in the A Register
Memory Z	

COMMAND CHARACTERISTICS

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "ones", this clears the J Counter. The A Register is then shifted right circular with the complement of A_0 applied to the Adder. The Carry flip-flop (F1AFNP) is held set applying a continuous "one" to the Adder. In this manner, the Adder output is equal to the true output of A_0. This Adder output is shifted to A_{23} leaving the A Register unchanged after 24 shifts. As the A Register is shifted, each "zero" bit increments the J Counter. In this manner, after 24 shifts, the J Counter will contain the number "zero" bits contained by the A Register.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AFNB = $\overline{JE37}$ + ($\overline{IR10}$ · $\overline{IR09}$)	Disabled
G1AFNB	67	AFNB = $\overline{JE37}$ + ($\overline{AI10}$ · $\overline{AR00}$) + ($\overline{AI09}$ · AR00)	Disabled
G0AFNA	65	$\overline{AR00}$ · ANA1	Gates $\overline{A_0}$ to Adder
F1AFNP	66	AFL2 + (AFL3 · AFNC · BCLK)	Held set applies a "1" to Adder
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Register right 24 places
N1J1NC	71	JIN2 · AFNA · $\overline{IR05}$	Each "zero" increments the J Counter

CPL-COMPLEMENT A

GEN 1

23								0
0	5	0	1	0	0	0	0	

CPL inverts each bit in the A Register; that is, each "one" is replaced by a "zero" and each "zero" is replaced by a "one". In this manner, CPL replaces the contents of the A Register with its ones complement.

During State 4, the A Register is shifted right 24 places with the complement of A_0 shifted through the Serial Full Adder and back to A_{23}. The B input of the Adder is inhibited. Since no summation is performed, no carries result. Therefore, at the end of 24 shifts, the ones complement of the original contents of A is contained in the A Register.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	C ($\overline{A_{23-0}}$)
Q_{23-0}	
P_{14-0}	C(P+1)
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{JE37} + (\overline{IR10} \cdot \overline{IR09})$	Disabled
G1AFNB	67	$\overline{AFNB} = \overline{JE37} + (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00} \cdot ANA1$	Gates $\overline{A_0}$ to Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ARS1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN4 \cdot JIN2$	Allows incrementation of J Counter

IBK-ISOLATE BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	010	0	K

$Z = f(X, K)$
$Z \leq 37_8$

IBK leaves bit Z of the A Register unchanged and clears all other bits in A. If Z is greater than 23_{10}, A will contain all "zeros".

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular. At each shift of the A Register, the J Counter is incremented. Until the J Counter is incremented to 37_8, the serial adder inputs are disabled, and "zeros" are shifted to A_{23}. When J is equal to 37_8, the bit specified by Z is shifted from A_0 and applied to the B inputs of the serial adder. The output of the Serial Adder is then equal to A_0 and is shifted to A_{23}. Therefore, this bit is unchanged. The inputs of the Serial Adder are enabled only when the J Counter is equal to 37_8. The J Counter will continue to be incremented at each shift of A until all 24 shifts have occurred. Therefore, following State 4, the A Register will contain all "zeros" except the bit specified by Z which will be unchanged.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$23_0 \ . \ . \ . \ C(A_Z) \ . \ . \ . \ 0^0$
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · IR09 · JE37 · AR00	Enables A_0 to Adder when J = 37_8
G1AFNB	67	AI09 · $\overline{AR00}$ · DGN1 · JE37	
G0AFNA	65	$\overline{AFNA}$ = $\overline{ANA3}$ · $\overline{DGN3}$ · $\overline{ANA1}$ · $\overline{ANA2}$	Disabled
F1AFNP	66	$\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK AFNP = AFL3 · AFNC · BCLK	Remains cleared because no summation is performed
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter

LBM-LOAD BIT MASK

GEN 1

23 18	17 15	14 6	5	4 0
05	X	630	0	K

$Z = f(X, K)$
$Z \leq 37_8$

LBM places "zero" in bit Z of the A Register and sets all other bits in A to "one". If Z is greater than 23_{10}, A will contain all "ones".

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with the J Counter incremented at each shift. The Carry flip-flop (F1AFNP) is held set throughout the shift of A. All other Adder inputs are disabled until $J = 37_8$. Therefore, the Carry flip-flop forces a "one" to the Adder and the "one" result is shifted to A_{23}. When J is equal to 37_8, a "one" is also applied to the B input. This provides a "zero" Adder output which is shifted to A_{23}. Therefore, when J is equal to 37_8, a "zero" is set in that position of the A Register. This corresponds to bit position Z. Since the J Counter is incremented at each shift of the A Register a "zero" adder output will only occur when bit position Z is shifted from A_0. At all other shifts, a "one" Adder output will be generated from the forced carry.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$23_1 \ldots Z_0 \ldots 1^0$
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · JE37 (IR09 · AR00 + IR10 · $\overline{AR00}$)	Applies a "1" to Adder when $J = 37_8$
G1AFNB	67	AI10 · AI09 · DGN1 · JE37	
G0AFNA	65	$\overline{ARNA}$ = $\overline{ANA3}$ · $\overline{DGN3}$ · $\overline{ANA1}$ · $\overline{ANA2}$	Disabled
F1AFNP	66	AFNP = AFL2	Held set, applies "1" to Adder
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

LDO-LOAD ONE INTO BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	030	0	K

$Z = f(X, K)$
$Z \leq 37_8$

LDO places a "one" in bit Z of the A Register. All other bits of A are cleared (zero). If Z is greater than 23, A will contain all "zeros".

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with the J Counter being incremented at each shift. All inputs to the Serial Adder are disabled, providing a "zero" to A_{23} at each shift except when J is equal to 37_8. When J is equal to 37_8, a "one" is applied to the B input of the Serial Adder providing a "one" to A_{23}. The J Counter is equal to 37_8 when the A Register bit specified by Z is shifted from A_0. Since the J Counter is incremented at each shift of A, only bit Z will be a "one". All other bits of A will be "zero".

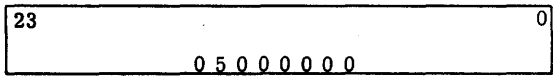
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$23_0 \ldots Z_1 \ldots 0^0$
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · JE37 · (IR09 · AR00 + IR10 · $\overline{AR00}$)	Applies a "1" to Adder when J = 37_8
G1AFNB	67	AI10 · AI09 · DGN1 · JE37	
G0AFNA	65	$\overline{AFNA}$ = $\overline{ANA3}$ · $\overline{DGN3}$ · $\overline{ANA1}$ · $\overline{ANA2}$	Disabled
F1AFNP	66	AFNP = AFL3 · AFNC · BCLK $\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK	Remains cleared because no summation is performed
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

LDZ-LOAD ZEROS INTO A

GEN 1



LDZ replaces the contents of the A Register with "zeros".

During State 4, the A Register is shifted right 24 places, however, the Serial Adder inputs are disabled providing a "zero" output to A_{23}. Therefore, after 24 shifts of A have occurred, all bits in the A Register are cleared.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 "Zeros" 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB}$ = $\overline{IR09}$ · $\overline{IR10}$	Disabled
G1AFNB	67	$\overline{AFNB}$ = ($\overline{AI09}$ · AR00) + ($\overline{AI10}$ · $\overline{AR00}$)	Disabled
G0AFNA	65	$\overline{AFNA}$ = $\overline{ANA3}$ · $\overline{DGN3}$ · $\overline{ANA1}$ · $\overline{ANA2}$	Disabled
F1AFNP	66	AFNP = AFL3 · AFNC · BCLK $\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK	Remains cleared because no summation is performed
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

LMO-LOAD MINUS ONE INTO A

GEN 1

23		0
0 5 0 6 0 0 0 0		

LMO places a "one" in each bit of the A Register (equals minus 1).

During State 4, the A Register is shifted 24 places to the right. During the shifts, the Carry flip-flop (F1AFNP) is held set. All other Serial Adder inputs are inhibited. In this manner, a "one" is applied to A_{23} at each shift. Therefore, at the end of 24 shifts, the A Register will contain all "ones" which is equal to a minus one in two's complement integer representation.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	22 "Ones" 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR09} \cdot \overline{IR10}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI09} \cdot AR00) + (\overline{AI10} \cdot \overline{AR00})$	Disabled
G0AFNA	65	$\overline{AFNA} = \overline{ANA3} \cdot \overline{DGN3} \cdot \overline{ANA1} \cdot \overline{ANA2}$	Disabled
F1AFNP	66	AFL2	Held set. Applies a "1" to Adder
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

NEG-NEGATE

GEN 1

23		0
0 5 0 1 3 0 0 0		

NEG replaces the contents of the A Register with its 2's complement value.

A timing and block diagram of the NEG command is contained in Fig. GN1.10, 11.

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{4-0} are all "zeros," this sets the J Counter equal to 37_8. The A Register is then shifted right circular with the complement of A_0 applied to the A input of the Serial Full Adder. Because the J Counter is equal to 37_8 at the first shift, a "one" is also applied to the B input of the Adder. The result of the summation of the complement of A_0 and one is gated to A_{23}. Any carry resulting from the summation is propagated by the Carry flip-flop (F1AFNP). After the first shift of A, the J Counter is no longer equal to 37_8 and and the B input of the Adder is inhibited. Therefore, after 24 shifts of the A Register, the A Register contains the 2's complement of the original contents of A.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	— C(A)
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	JE37·AGN1·(AR00·IR09 + $\overline{AR00}$·IR10)	Applies "1" to Adder (B input) at first shift because J = 37_8
G1AFNB	67	AI10·AI09·DGN1·JE37	
G0AFNA	65	$\overline{AR00}$·ANA1	Gates $\overline{A_0}$ to Adder (A input)
F1AFNP	66	$\overline{AFNP}$ = AFL3·$\overline{AFNC}$·BCLK AFNP = AFL3·AFNC·BCLK	Allows normal carry
D1ASRL, U	63.1	ASR1 = DGN1·T6E3·$\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4·JIN2	Enables incrementation of J Counter

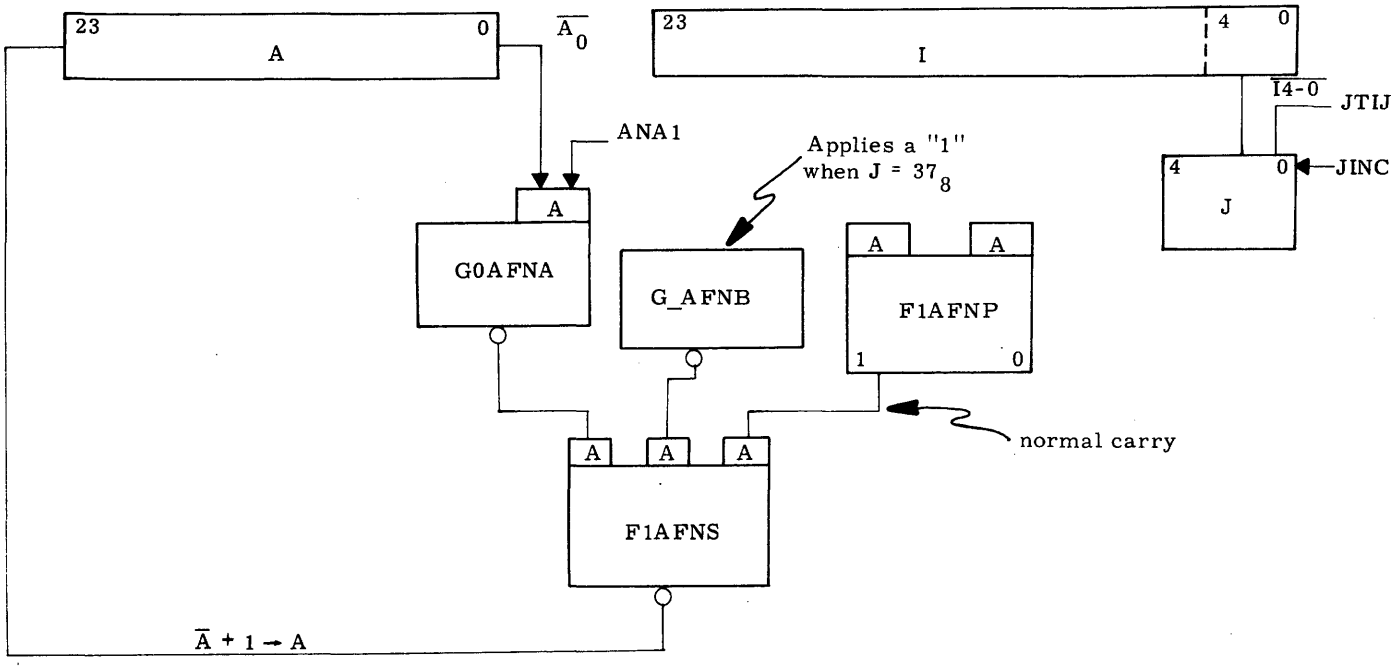


Fig. GN1.10 NEG Block Diagram

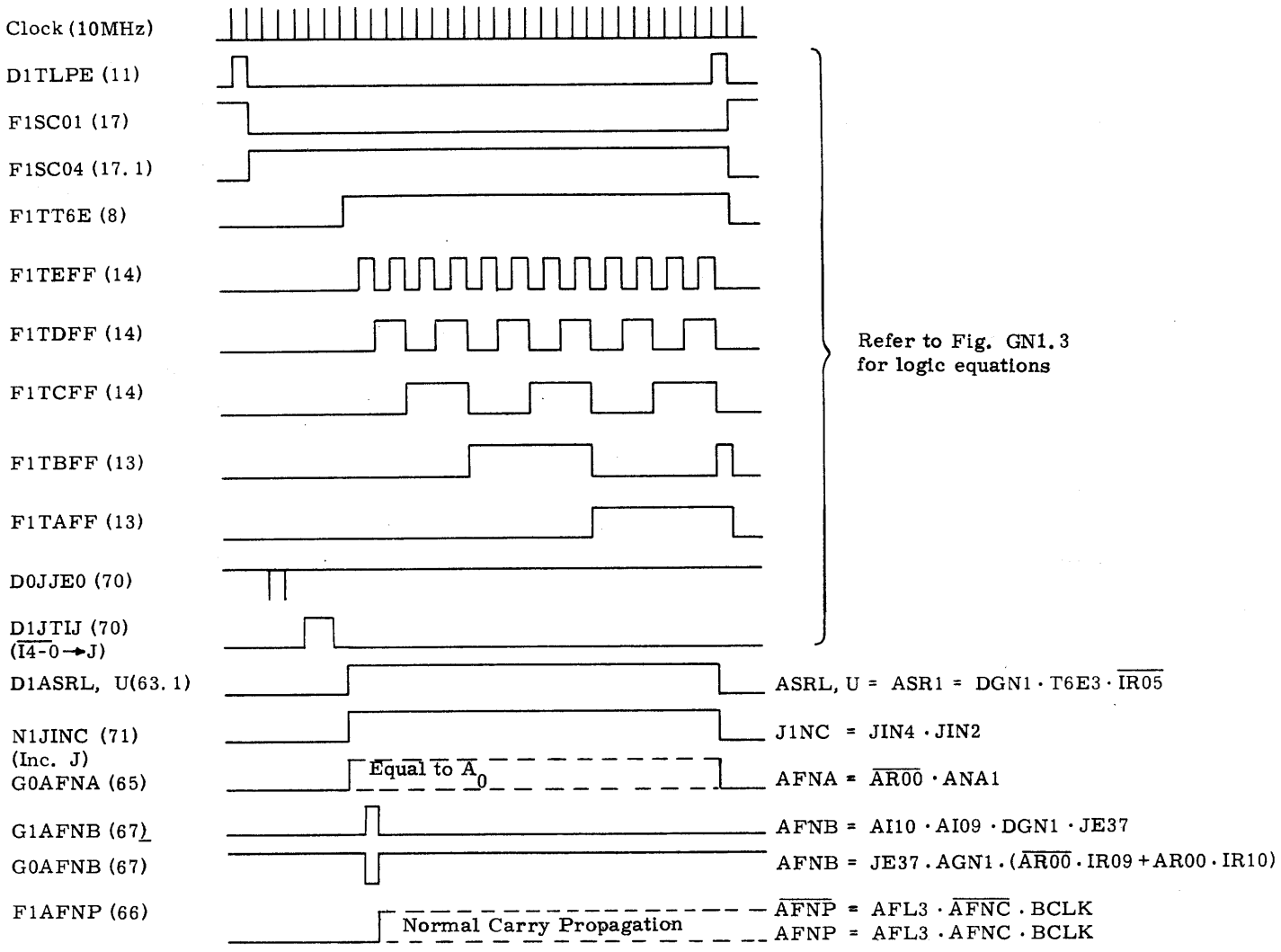


Fig. GN1.11 NEG Timing Diagram

RBK-RESET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	450	0	K

$Z = f(X, K)$
$Z \leq 37_8$

RBK places a "zero" into bit Z of the A Register. All other bits in A remain unchanged. If Z is greater than 23_{10}, A will be unchanged.

During State 4, the complement of I_{4-0} is gated to the J Counter. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. At each shift of A, the J Counter is incremented. When J is equal to 37_8, the bit specified by Z is shifted from A_0 and A_0 is also applied to the B input of the Adder. When A_0 is applied to both the A and B inputs of the Adder, the sum output will always be "zero". This "zero" is gated to A_{23} at the shift. Since J is equal to 37_8 only when bit Z is shifted from A_0, this bit will always be a "zero". The remaining bits of the A Register will be unchanged because A_0 is only applied to the Adder A input during the remaining shifts.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 Z_0 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · IR09 · JE37 · AR00	Enables A_0 to B input of Adder when $J = 37_8$
G1AFNB	67	AR00 · AI09 · DGN1 · JE37	
G0AFNA	65	AR00 · ANA2	Enables $A_0 \rightarrow$ A input of Adder
F1AFNP	66	$\overline{AFNP} = \overline{AFL3} + \overline{AFL2}$	Held clear
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

REV-RESET TEST FLIP-FLOP IF BIT K IS EVEN

GEN 1

23 18	17 15	14 6	5	4 0
05	X	704	0	K

Z = f(X, K)
$Z \leq 37_8$

REV clears the Test flip-flop (F1ETST) if bit Z in the A Register is a "zero". If bit Z is a "one", the status of the Test flip-flop is unchanged. If Z is greater than 23_{10}, the Test flip-flop is unchanged. The original contents of A are unchanged by the REV command.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with the complement of A_0 applied to the A input of the Adder. The Carry flip-flop (F1AFNP) is held set throughout the shift of A applying a "one" to the B input of the Adder. The summation of 1 plus the complement of A_0 provides a Sum output of the Adder equal to the true A_0. This output is gated to A_{23}, leaving the A Register unchanged. The J Counter is incremented at each shift of A. When the J Counter is equal to 37_8, bit Z is shifted from the A Register. If this bit is a "zero", the Test flip-flop (F1ETST) is cleared. If bit Z is a "one", the status of the Test flip-flop is unchanged.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Reset if $A_Z = 0$
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00}$ · ANA1	$\overline{A_0}$ to Adder A input
F1AFNP	66	AFL2· $\overline{AFL3}$	Held set
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST}$ = $\overline{IR06}$· ESTF· ECLK ESTF = JE37· EGIT· AFNA· ASR1	Reset Test flip-flop if $A_Z = 0$

RNZ-RESET TEST FLIP-FLOP IF A IS NON-ZERO

GEN 1

23								0
0	5	0	0	4	4	7	0	

RNZ clears the Test flip-flop (F1ETST) if any bit in the A Register is a "one". If all bits in A are "zero", the status of the Test flip-flop is unchanged. The original contents of A are unchanged by the RNZ command.

During State 4, the complement of I_{4-0} is gated to the J Counter. Since I_{2-0} are "zeros" from the RNZ command, this presets the J Counter equal to 7. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Adder. All other Adder inputs are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23} leaving the A Register unchanged. Any "one" shifted from A_0 will cause the Test flip-flop (F1ETST) to be cleared. The J Counter is incremented at each shift of the A Register. Since the J Counter was preset to 7_8, after 24 shifts of A the J Counter will equal 37_8. When J equals 37_8 further shifting of A is inhibited. Therefore, the A Register is shifted 24 places and only 24 places.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Reset if C(A) $\neq$ 0
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains clear because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$JIN2 \cdot JIN1 \cdot \overline{JE37}$	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST} = ESTF \cdot \overline{IR06} \cdot ECLK$ $ESTF = IR05 \cdot EG1T \cdot AFNA \cdot ASR1$	Any "1" from A_0 resets the Test flip-flop

ROD-RESET TEST FLIP-FLOP IF K IS ODD

GEN 1

23			18	17		15	14			6	5	4		0
0	5			X			0	4	4		0	K		

$Z = f(X, K)$
$Z \leq 37_8$

ROD clears the Test flip-flop (F1ETST) if bit Z in the A Register is a "one". If bit Z is a "zero", the status of the Test flip-flop is unchanged. The original contents of A are unchanged by the ROD command. If Z is greater than 23_{10}, the status of the Test flip-flop is unchanged.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs of the Adder are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged. The J Counter is incremented at each shift of the A Register. When J is equal to 37_8, bit Z is shifted from A_0. If bit Z is a "one", the Test flip-flop (F1ETST) is cleared. If bit Z is a "zero", the status of the Test flip-flop is unchanged.

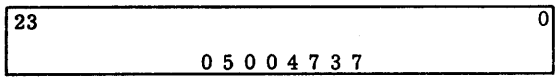
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Reset if A_Z = 1
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts the A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST} = \overline{IR06} \cdot ESTF \cdot ECLK$ $ESTF = JE37 \cdot EG1T \cdot AFNA \cdot ASR1$	Resets Test flip-flop if A_Z = "1"

RST-RESET THE TEST FLIP-FLOP

GEN 1



RST unconditionally clears the Test flip-flop (F1ETST). The contents of A are unchanged by RST.

During State 4, the Test flip-flop (F1ETST) is reset. The A Register is circular shifted right with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Reset
J_{4-0}	30_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts the A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST} = \overline{ECTF} = IR07 \cdot EI06 \cdot EG1T \cdot \overline{ET4E}$	Reset unconditionally

SBK-SET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	460	0	K

$Z = f(X, K)$
$Z \leq 37_8$

SBK sets bit Z in the A Register to "one". All other bits in the A Register are unchanged. If Z is greater than 23_{10}, the contents of A will be unchanged.

During State 4, the complement of I_{4-0} is gated to the J Counter. The A Register is then shifted right circular with A_0 applied through the Serial Full Adder to A_{23}. At each shift of A, the J Counter is incremented. When J is equal to 37_8, bit Z is shifted from A_0. At this time, the complement of A_0 is applied to the B input of the Adder. The summation of $\overline{A_0}$ plus A_0 will always provide a "one" output to A_{23}. Therefore bit Z of the A Register is set to a "one". Since the J Counter is equal to 37_8 only when bit Z is shifted from A_0, all other bits will be unchanged.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 Z_1 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · JE37 · IR10 · $\overline{AR00}$	Gates $\overline{A_0}$ to Adder B input when J = 37_8
G1AFNB	67	AI10 · $\overline{AR00}$ · JE37 · DGN1	
G0AFNA	65	AR00 · ANA2	Gates A_0 to Adder A input
F1AFNP	66	$\overline{AFNP}$ = AFL2	Held clear by $\overline{AFL3}$
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN4 · JIN2	Enables incrementation of J Counter

SET-SET TEST FLIP-FLOP

GEN 1

23 0
0 5 0 0 4 6 3 7

SET unconditionally sets the Test flip-flop (F1ETST). The original contents of the A Register are unchanged.

During State 4, the Test flip-flop (F1ETST) is set. The A Register is circular shifted right with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set
J_{4-0}	30_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$G0ESTF = \overline{IR06} \cdot IR07 \cdot EG1T \cdot ET4E$	Set unconditionally

SEV-SET TEST FLIP-FLOP IF BIT K IS EVEN

GEN 1

23 18	17 15	14 6	5	4 0
05	X	705	0	K

$Z = f(X, K)$
$Z \leq 37_8$

SEV sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "zero". If bit Z in A is a "one", the status of the Test flip-flop is unchanged. The original contents of the A Register are unchanged.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with the complement of A_0 applied to the A input of the Adder. The Carry flip-flop (F1AFNP) is held set applying a "one" to the B input of the Adder. The summation of 1 plus the complement of A_0 provides a Sum output of the Adder equal to the true A_0 output. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts. The J Counter is incremented at each shift of A. When J is equal to 37_8, bit Z of the A Register is shifted from A_0. If this bit is a "zero", the Test flip-flop (F1ETST) is set. If this bit is a "one", the Test flip-flop status is unchanged.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if $A_Z = 0$
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00} \cdot ANA1$	$\overline{A_0}$ to A input of the Adder
F1AFNP	66	$AFNP = AFL2$	Held set by $\overline{AFL3}$
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$ETST = ESTF \cdot EI06 \cdot ECLK$ $ESTF = JE37 \cdot EG1T \cdot AFNA \cdot ASR1$	Set Test flip-flop if $A_Z = 0$

SNZ-SET TEST FLIP-FLOP IF A IS NON-ZERO

GEN 1

23																							0
0	5	0	0	4	5	7	0																

SNZ sets the Test flip-flop (F1ETST) if any bit in the A Register is a "one". If all bits in A are "zero", the status of the Test flip-flop is unchanged. The original contents of the A Register are unchanged.

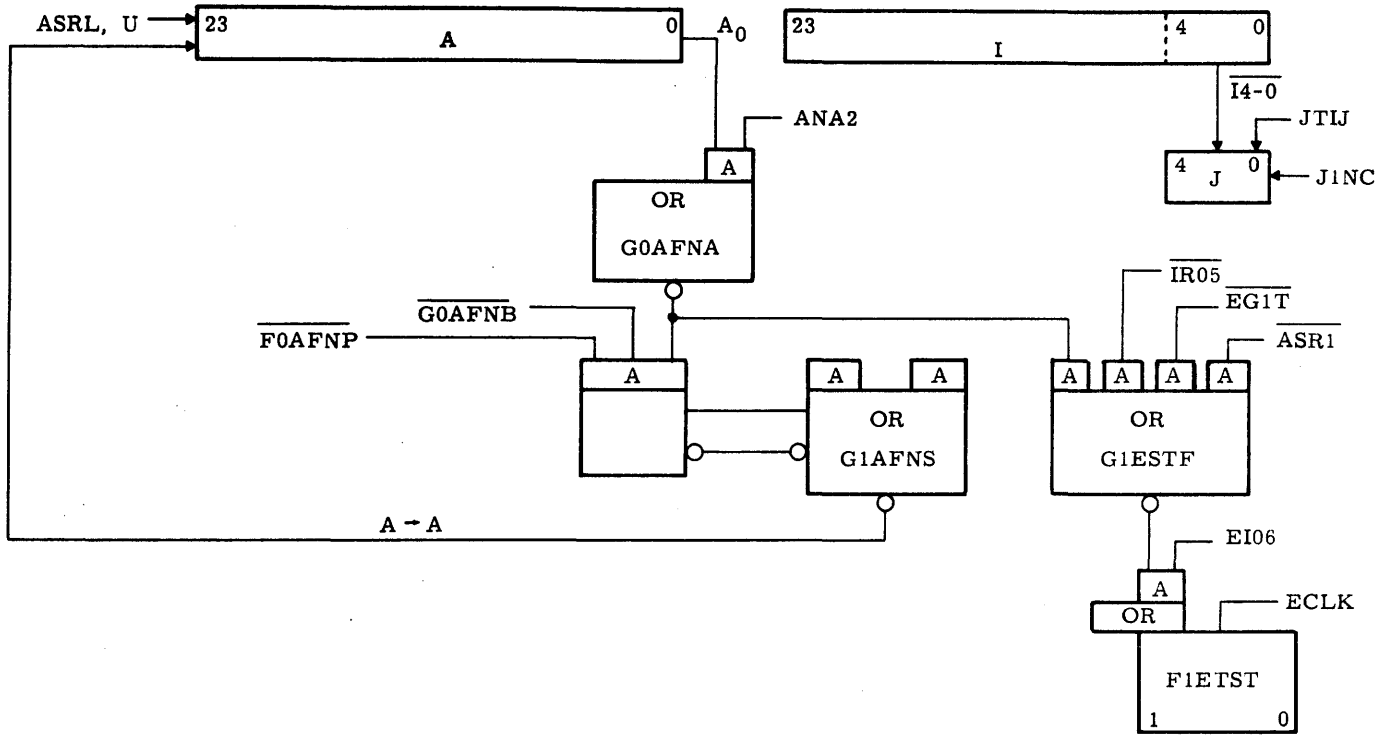
A timing and block diagram of the SNZ command is contained in Fig. GN1.12, 13.

During State 4, the complement of I_{4-0} is gated to the J Counter. Since bits 2 through 0 of the SNZ command are "zeros", this presets the J Counter equal to 7. The A Register is then shifted right with A_0 applied to the A input of the Serial Full Adder. All other Adder inputs are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts. If at any shift of A, A_0 is a "one", the Test flip-flop (F1ETST) is set. If all bits of A are "zero", the status of the Test flip-flop is unchanged. The J Counter is incremented at each shift of the A Register and when equal to 37_8 inhibits further shifting of A. Since the J Counter was preset to 7, when J equals 37_8, 24 shifts of A have occurred.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if C(A) $\neq$ 0
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$JIN2 \cdot JIN1 \cdot \overline{JE37}$	Enables incrementation of J Counter until J = 37_8
F1ETST	89	$ETST = G1ESTF \cdot EI06 \cdot ECLK$ $ESTF = IR05 \cdot EG1T \cdot AFNA \cdot ASR1$	Any "1" from A_0 sets the Test flip-flop



Condition	G0AFNB Output	G1AFNB Output	G0AFNA Output	F1AFNP Output	G1AFNS Output	F1ETST
$A_0 = 1$	1	0	0	0	1	1
$A_0 = 0$	1	0	1	0	0	-

Fig. GN1.12 SNZ Block Diagram

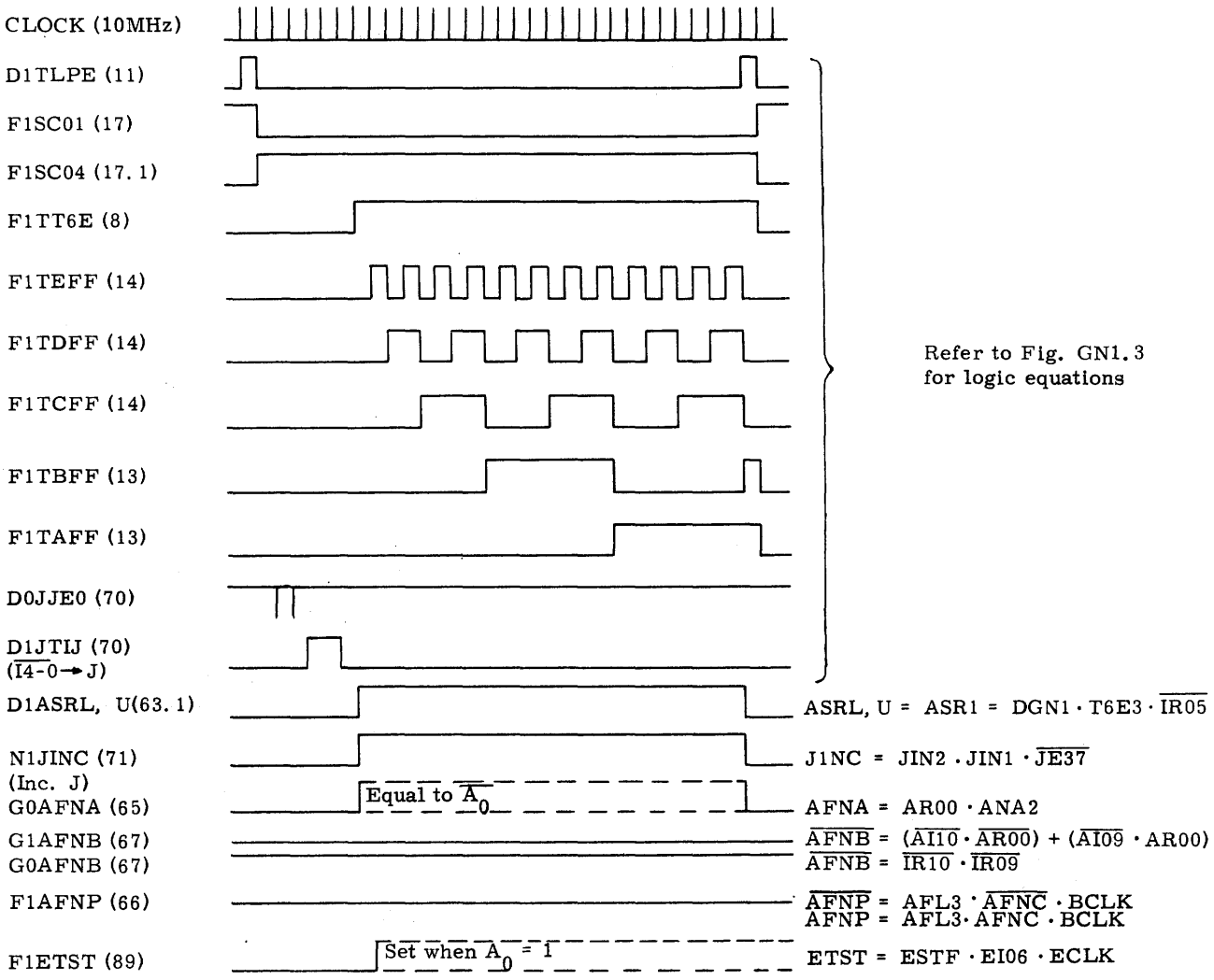


Fig. GN1.13 SNZ Timing Diagram

SOD-SET TEST FLIP-FLOP IF BIT K IS ODD

GEN 1

23 18	17 15	14 6	5	4 0
05	X	045	0	K

Z = f(X, K)
$Z \leq 37_8$

SOD sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "one". If bit Z in A is a "zero", the status of the Test flip-flop is unchanged. The original contents of the A Register are unchanged. If Z is greater than 23_{10}, the status of the Test flip-flop is unchanged.

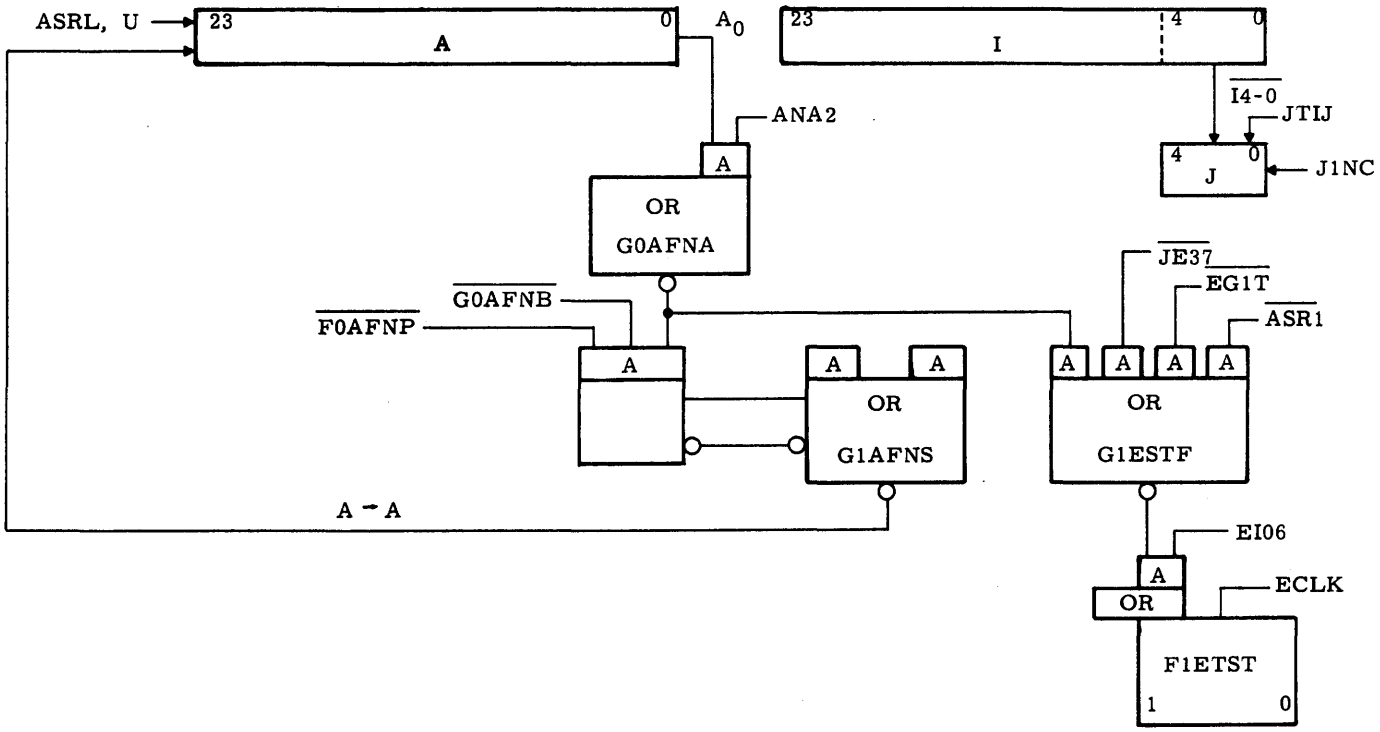
A timing and block diagram of the SOD command are contained in Fig. GN1.14, 15.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs of the Adder are disabled. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts. The J Counter is incremented at each shift of A. When J is equal to 37_8, bit Z is shifted from A_0 and if a "one", the Test flip-flop is set.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if $C(A_Z) = 1$
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$ETST = G1ESTF \cdot EI06 \cdot ECLK$ $G1ESTF = JE37 \cdot EG1T \cdot AFNA \cdot ASR1$	Set Test flip-flop if $A_Z = 1$ when $J = 37_8$



Condition	G0AFNB Output	G1AFNB Output	G0AFNA Output	F1AFNP Output	G1AFNS Output
$A_0 = 1$	1	0	0	0	1
$A_0 = 0$	1	0	1	0	0

Fig. GN1.14 SOD Block Diagram

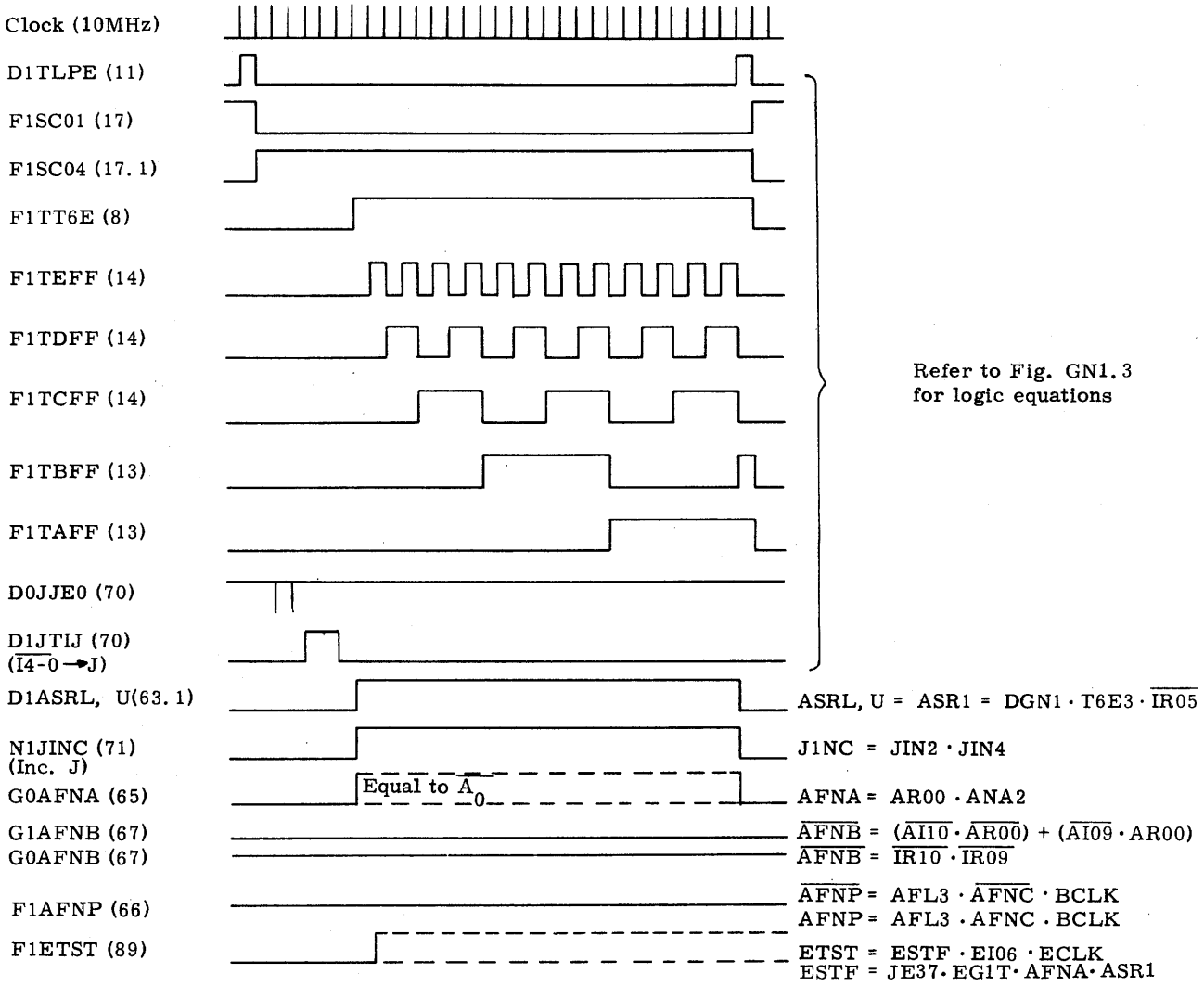


Fig. GN1.15 SOD Timing Diagram

SRA-SHIFT A RIGHT ARITHMETIC

GEN 1

23 18	17 15	14 6	5	4 0
05	X	140	1	K

$Z = f(X, K)$
$Z \leq 37_8$

SRA shifts the contents of the A Register Z places to the right. The sign bit (23) of A is unchanged. Bits shifted out of A_0 are lost. Bits shifted into A_{22} are the same as the sign bit. The maximum number of shifts is 24 places.

A timing and block diagram of the SRA command is contained in Fig. GN1.16, 17.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right with A_{23} shifted to A_{22} and with A_{23} also shifted through the Serial Full Adder back to A_{23}, leaving A_{23} unchanged. At each shift of the A Register, the J Counter is incremented. When J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. Therefore, the A Register is shifted right the number of places specified by Z.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	See Text
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	ANA3 · AR23 ANA3 = IR11 · IR12 · AGN1	Gates A_{23} to A input of Adder
F1AFNP	66	$\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK AFNP = AFL3 · AFNC · BCLK	Remains cleared because no summation is performed
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$\overline{JE37}$ · JIN1 · JIN2	Increments J Counter until J = 37_8

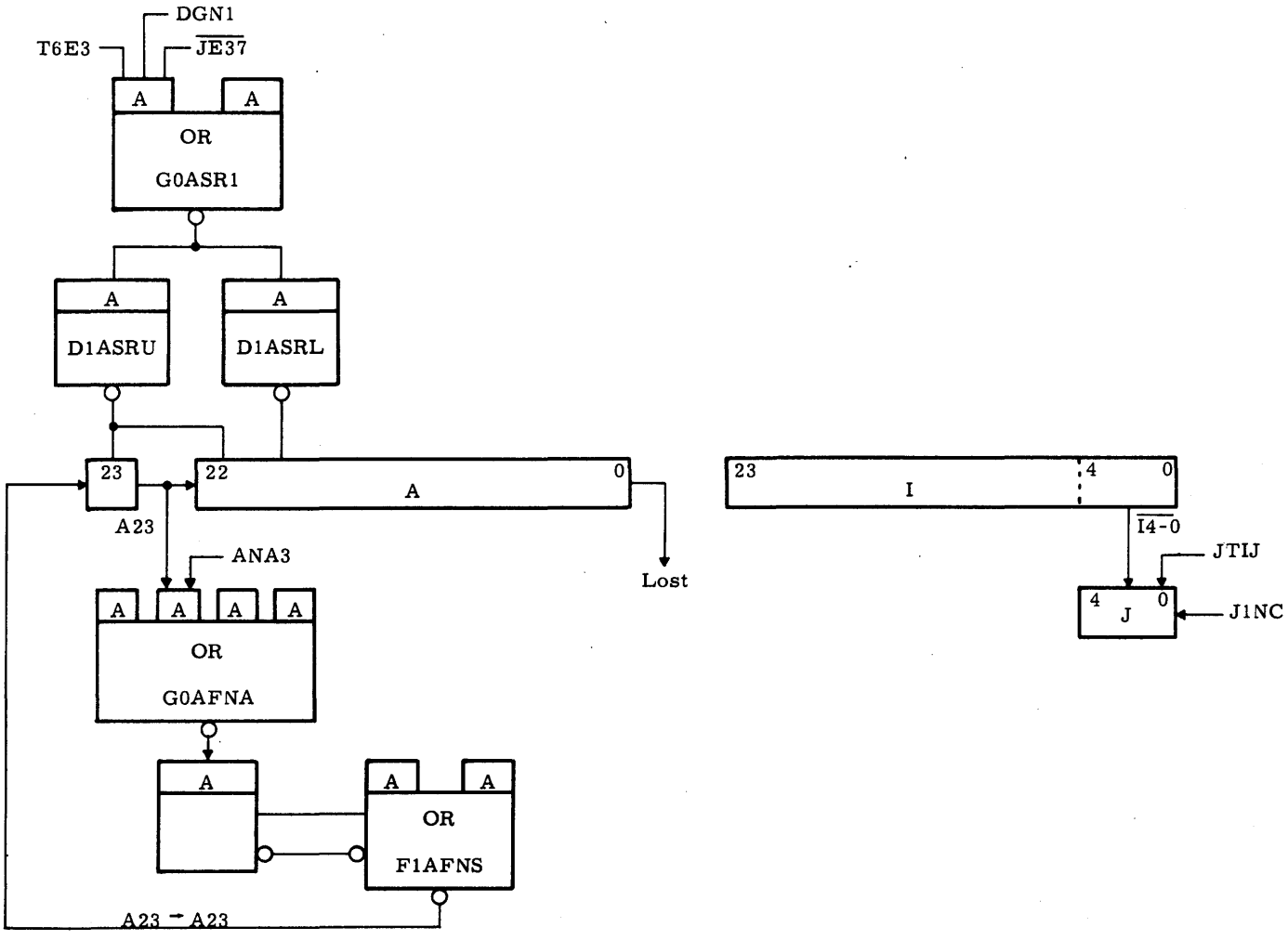


Fig. GN1.16 SRA Block Diagram

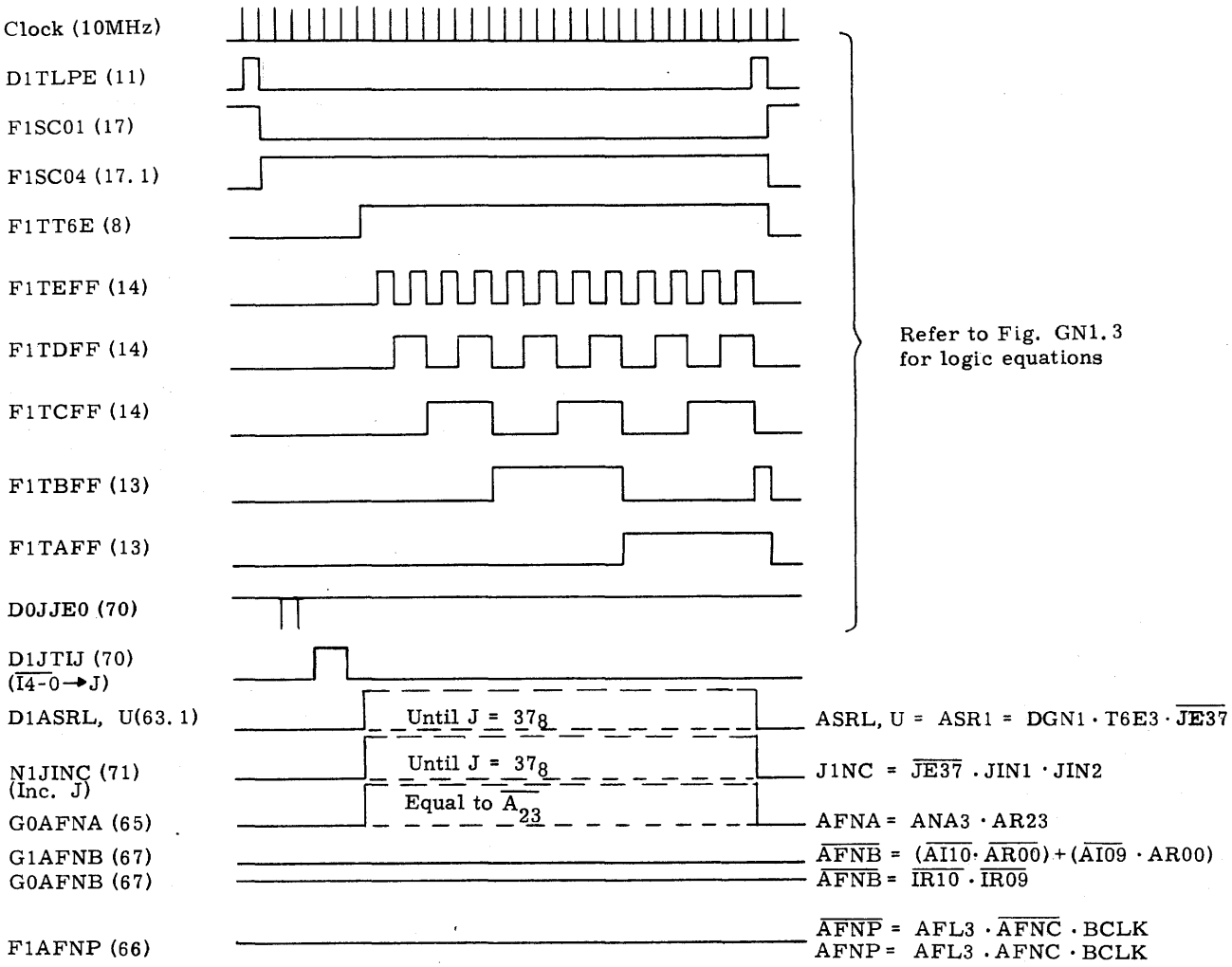


Fig. GN1.17 SRA Timing Diagram

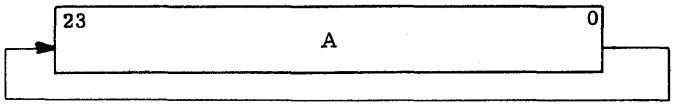
SRC-SHIFT RIGHT CIRCULAR

GEN 1

23 18	17 15	14 6	5	4 0
05	X	040	1	K

Z = f(X, K)
Z ≤ 37_8

SRC shifts the contents of the A Register Z places to the right in a circular fashion; that is, the bit shifted from A_0 is inserted in A_{23}, replacing the bit shifted out of A_{23}. The maximum number of shifts is 24.



During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right circular with A_0 applied through the A input of the Serial Adder back to A_{23}. At each shift of the A Register, the J Counter is incremented. When J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting of A is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	See Text
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until $J = 37_8$
N1J1NC	71	JE37 · JIN1 · JIN2	Increments J Counter until $J = 37_8$

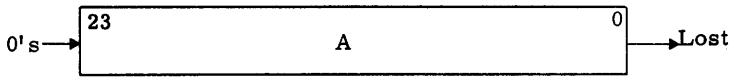
SRL-SHIFT RIGHT LOGICAL

GEN 1

23 18	17 15	14 6	5	4 0
05	X	000	1	K

$Z = f(X, K)$
$Z \leq 37_8$

SRL shifts all twenty-four bits of the A Register to the right Z places. Zeros are shifted in through A_{23}. Bits shifted out of A_0 are lost. The maximum number of shifts is 24 places.



During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted right with the J Counter being incremented at each shift. When J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting of A is inhibited. Bits shifted from A_0 are lost. "Zeros" are shifted into A_{23} at each shift.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	See Text
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AFNA} = \overline{ANA1} \cdot \overline{ANA2} \cdot \overline{ANA3}$	Disabled
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no inputs to Adder
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until $J = 37_8$
N1J1NC	71	$\overline{JE37} \cdot JIN1 \cdot JIN2$	Enabled J incrementation until $J = 37_8$

TER-TEST EVEN AND RESET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	456	0	K

$Z = f(X, K)$
$Z \leq 37_8$

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 A_Z 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if A_Z = 0; Reset if A_Z = 1
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

TER sets the Test flip-flop (F1ETST) if Bit Z in the A Register is "zero", and places a "zero" back in bit Z. If bit Z is "one" the Test flip-flop is cleared and a "zero" is placed in bit Z. The Test flip-flop will be set if Z is greater than 23_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is unconditionally set. The A Register is shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled until the J Counter is equal to 37_8. In this manner, the Sum output of the Adder is equal to A_0 until J is equal to 37_8 and this Sum output is applied to A_{23} leaving these bits of A unchanged. The J Counter is incremented at each shift of A. When J is equal to 37_8, A_0 is also applied to the B input of the Adder. This always results in a Sum output of "zero". This "zero" is gated to A_{23}, placing a "zero" in position Z of the A Register. Also, when J is equal to 37_8, if the bit shifted from A (A_Z) is a "one", the Test flip-flop is cleared. If this bit is a "zero", the Test flip-flop remains in the set state.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · IR09 · JE37 · AR00	Gates A_0 to B input of Adder when J = 37_8
G1AFNB	67	AR00 · AR09 · DGN1 · JE37	
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP}$ = (AFL1 · BCLK) + $\overline{AFL2}$ + $\overline{AFL3}$	Held Clear
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter
F1ETST	89	ETST = G0ESTF = $\overline{IR06}$ · IR07 · ET4E · $\overline{EGIT}$ $\overline{ETST}$ = G1ESTF · $\overline{IR06}$ · ECLK G1ESTF = JE37 · EGIT · AFNA · ASR1	Set Test F/F; Clear Test if A_Z = 1

TES-TEST EVEN AND SET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	466	0	K

$Z = f(X, K)$
$Z \leq 37_8$

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 Z_1 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if A_Z = 0 Reset if A_Z = 1
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

TES sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "zero", and replaces bit Z with a "one". If bit Z is a "one", the Test flip-flop is cleared and a "one" is placed back in bit Z. The Test flip-flop will be set if Z is greater than 23_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is unconditionally set. The A Register is shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled until the J Counter is equal to 37_8. In this manner, the Sum output of the Adder is equal to A_0, unless J is equal to 37_8, and this Sum output is gated to A_{23} leaving these bits of A unchanged. The J

Counter is incremented at each shift of A. When J is equal to 37_8, the complement of A_0 is applied to the B input of the Adder. The summation of A_0 and $\overline{A_0}$ always provides a Sum output of "one". This "one" is applied to A_{23}, thereby setting bit Z to a "one". Also, when J is equal to 37_8, if the bit shifted from A_0 (A_Z) is a "one", the Test flip-flop is cleared. If this bit is a "zero", the Test flip-flop remains in the set state.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · JE37 · IR10 · $\overline{AR00}$	Gates $\overline{A_0}$ to B input of Adder when J = 37_8
G1AFNB	67	AI10 · $\overline{AR00}$ · DGN1 · JE37	
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP}$ = (AFL1 · BCLK) + $\overline{AFL2}$ + $\overline{AFL3}$	Held clear
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST}$ = G0ESTF = $\overline{IR06}$ · IR07 · ET4E · EG1T $\overline{ETST}$ = G1ESTF • $\overline{IR06}$ · ECLK G1ESTF = JE37 · EGIT · AFNA · ASR1	Set Test F/F; Clear Test F/F if A_Z = 1

TEV-TEST BIT K EVEN

GEN 1

23 18	17 15	14 6	5	4 0
05	X	707	0	K

Z = f(X, K)
$Z \leq 37_8$

TEV sets the Test flip-flop (F1ETST) if bit Z in the A Register is "zero". If bit Z in the A is "one", the Test flip-flop is cleared. The Test flip-flop is cleared if Z is greater than 23_{10}. The original contents of the A Register are unchanged by the TEV command.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop is unconditionally cleared. The Carry flip-flop (F1AFNP) is held set applying a "one" to the B input of the Serial Adder. The A Register is then shifted right with the complement of A_0 applied to the A input of the Serial Adder. The summation of A_0 and the forced "one" from F1AFNP provides a Sum output of the Adder equal to the true A_0. This sum output is applied to A_{23}, leaving the A Register unchanged. At each shift of A, the J Counter is incremented. When J is equal to 37_8, bit Z is shifted from A_0. If A_0 is a "zero", the Test flip-flop is set. If A_0 is a "one" when J equals 37_8, the Test flip-flop remains cleared.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if A_Z = 0; Reset if A_Z = 1
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00} \cdot ANA2$	Gates $\overline{A_0}$ to A input of Adder
F1AFNP	66	$AFL2 + \overline{AFL3}$	Held set
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	ETST = EI06· G1ESTF· ECLK G1ESTF = JE37· EG1T· AFNA· ASR1 ETST = ECTF = IR07· EI06· EG1T· ET4E	Clear Test F/F; Set Test F/F if $A_Z = 0$

TNM-TEST NOT MINUS ONE

GEN 1

23							0
0	5	0	7	0	7	7	0

TNM sets the Test flip-flop (F1ETST) if any bit in the A Register is a "zero". When all bits of the A Register are "ones" (minus 1) the Test flip-flop is cleared. The original contents of the A Register are unchanged by the TNM command.

During State 4, the complement of I_{4-0} is gated to the J Counter. This presets the J Counter equal to 7, since bits 2 through 0 of the TNM command are "zeros". The Test flip-flop (F1ETST) is reset. The Carry flip-flop (F1AFNP) is held set applying a "one" to the B input of the Serial Full Adder. The A Register is then shifted to the right with the complement of A_0 applied to the Serial Full Adder. The Sum output of the Adder is equal to the true A_0. This Sum output is applied to A_{23}, leaving A unchanged after 24 shifts. If at any shift of A, A_0 is a "zero", the Test flip-flop is set. If all bits of A are "ones", the Test flip-flop remains in the reset state. At each shift of the A Register, the J Counter is incremented. When the J Counter is equal to 37_8, 24 shifts of the A Register have occurred and further shifting is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if any bit in A = 0; Reset if all bits in A = 1
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$\overline{AR00} \cdot ANA1$	Gates $\overline{A_0}$ to A input of Adder
F1AFNP	66	$AFL2 + \overline{AFL3}$	Held set
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until J = 37
N1J1NC	71	$\overline{JE37} \cdot JIN1 \cdot JIN2$	Enables incrementation of J Counter until $J = 37_8$
F1ETST	89	$\overline{ETST} = \overline{ECTF} = IR07 \cdot EI06 \cdot EG1T \cdot ET4E$ $\overline{ETST} = G1ESTF \cdot EI06 \cdot ECLK$	Clear Test F/F; Set Test F/F if any bit of A = 0

TNZ-TEST A NON-ZERO

GEN 1

23								0
0	5	0	0	4	7	7	0	

TNZ sets the Test flip-flop (F1ETST) if any bit in the A Register is a "one". If all bits in the A Register are "zero", the Test flip-flop is cleared. The original contents of the A Register are unchanged by the TNZ command

During State 4, the complement of I_{4-0} is gated to the J Counter. This presets J equal to 7, since bits 2 through 0 of the TNZ command are "zeros". The Test flip-flop (F1ETST) is cleared. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled providing a Sum output equal to A_0. This Sum output is applied to A_{23}, leaving A unchanged after 24 shifts. If at any shift of A, A_0 is equal to a "one", the Test flip-flop is set. If all bits of A are "zero", the Test flip-flop remains in the reset state. The J Counter is incremented at each shift of A. When the J Counter is equal to 37_8, 24 shifts of A have occurred and further shifting of A is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if C(A) ≠ 0; Reset if C(A) = 0
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (AI09 \cdot AR00)$	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$\overline{JE37} \cdot JIN1 \cdot JIN2$	Enables J incrementation until J = 37_8
F1ETST	89	$\overline{ETST} = ECTF = IR07 \cdot EI06 \cdot EG1T \cdot ET4E$ $ETST = EI06 \cdot G1ESTF \cdot ECLK$	Clear Test F/F; Set Test F/F if any bit of A = 1

TOD-TEST BIT K ODD

GEN 1

23	18	17	15	14	6	5	4	0
05		X		047		0	K	

$Z = f(X, K)$
$Z \leq 37_8$

TOD sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "one". If bit Z in A is "zero", the Test flip-flop is cleared. The original contents of the A Register are unchanged by the TOD command. The Test flip-flop is cleared if Z is greater than 23_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is unconditionally reset. The A Register is shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are inhibited. Therefore, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts. At each shift of A, the J Counter is incremented. When J is equal to 37_8, bit Z is shifted from A and if bit Z is a "one", the Test flip-flop is set. If bit Z is a "zero", the Test flip-flop remains in the reset state.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if A_Z = 1 Reset if A_Z = 0
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains cleared because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	$JIN2 \cdot JIN4$	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST} = ECTF = IR07 \cdot EI06 \cdot EG1T \cdot ET4E$ $ETST = EI06 \cdot G1ESTF \cdot ECLK$	Clear Test F/F; Set Test F/F if $A_Z = 1$

TOR-TEST ODD AND RESET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	457	0	K

$Z = f(X, K)$
$Z \leq 37_8$

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 Z 0
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if $A_Z = 1$ Reset if $A_Z = 0$
J_{4-0}	$27_8 - Z$
Memory Z	

COMMAND CHARACTERISTICS

TOR sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "one" and replaces bit Z with a "zero". If bit Z in A is a "zero", the Test flip-flop is cleared and bit Z remains "zero". The Test flip-flop is cleared if Z is greater than 23_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is reset. The A Register is shifted right with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled, except when J is equal to 37_8. When J is not equal to 37_8, the Sum output of the Adder is equal to A_0. This Sum output is applied to A_{23}, leaving these bits of A unchanged. The J Counter is incremented at each shift of A and when equal to 37_8, bit Z is shifted from A_0. If bit Z is a "one", the Test flip-flop is set. Also, when J is equal to 37_8, A_0 is applied to the B input of the Adder. The Sum of A_0 and A_0 will always result in a "zero" Sum output of the Adder. The Sum output is applied to A_{23}, providing a "zero" in position Z of the A Register after 24 shifts have occurred.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · IR09 · JE37 · AR00	Gates A_0 to B input of Adder When J = 37_8
G1AFNB	67	AR00 · AI09 · DGN1 · JE37	
G0AFNA	65	AR00 . ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFL2}$ + $\overline{AFL3}$ + (AFL1 · ECLK)	**Held reset**
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST}$ = ECTF = IR07 · EI06 · EG1T · ET4E ETST = G1ESTF · EI06 · ECLK	Clear Test F/F; Set Test F/F if A_Z = 1

TOS-TEST ODD AND SET BIT K

GEN 1

23 18	17 15	14 6	5	4 0
05	X	467	0	K

Z = f(X, K)
Z ≤ 37_8

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	23 Z 0 1
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if A_Z = 1; Reset if A_Z = 0
J_{4-0}	27_8 - Z
Memory Z	

COMMAND CHARACTERISTICS

TOS sets the Test flip-flop (F1ETST) if bit Z in the A Register is a "one" and bit Z remains a "one". If bit Z in A is a "zero", the Test flip-flop is cleared and bit Z is replaced with a "one". The Test flip-flop is cleared if Z is greater than 23_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is unconditionally reset. The A Register is shifted right with A_0 applied to the A input of the Serial Full Adder. All other inputs to the Adder are disabled, except when the J Counter is equal to 37_8. Therefore, the Sum output of the Adder is equal to A_0, unless J is equal to 37_8. This Sum output is applied to A_{23}, leaving these bits of A unchanged. The J Counter is incremented at each shift of A. When J is equal to 37_8, bit Z is shifted from the A Register. The complement of bit Z is applied to the B input of the Adder. The result of the summation of A_0 and $\overline{A_0}$ results in a "one" Sum output of the Adder. This "one" is applied to A_{23}, making bit Z a "one". Also, when J is equal to 37, if A_0 is a "one", the Test flip-flop is set.

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	AGN1 · JE37 · IR10 · $\overline{AR00}$	Gates $\overline{A_0}$ to B input of Adder when J = 37_8
G1AFNB	67	AI10 · $\overline{AR00}$ · DGN1 · JE37	
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFL2}$ + $\overline{AFL3}$ + (AFL1 · ECLK)	Held reset
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{IR05}$	Shifts A Reg. right 24 places
N1J1NC	71	JIN2 · JIN4	Enables incrementation of J Counter
F1ETST	89	$\overline{ETST}$ = ECTF = IR07 · EI06 · EG1T · ET4E ETST = G1ESTF · EI06 · ECLK	Clear Test F/F; Set Test F/F if A_Z = 1

TSC-TEST AND SHIFT CIRCULAR

GEN 1

23 18	17 15	14 6	5	4 0
05	X	046	1	K

Z = f(X, K)
Z ≤ 37_8

TSC shifts the contents of the A Register right circular Z places. If all bits shifted out of A_0 are "zero", the Test flip-flop (F1ETST) is set, otherwise it is cleared. The maximum number of shifts is 24_{10}.

During State 4, the complement of I_{4-0} (Z) is gated to the J Counter. The Test flip-flop (F1ETST) is set. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other Adder inputs are disabled providing a Sum output of the Adder equal to A_0. This Sum output is applied to A_{23}. If any bit shifted from A_0 is a "one", the Test flip-flop is cleared. The J Counter is incremented at each shift of A. When J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	C(A) shifted right circular Z places
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set / Reset } See Text
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB}$ = $\overline{IR10}$ · $\overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB}$ = ($\overline{AI10}$ · $\overline{AR00}$) + ($\overline{AI09}$ · AR00)	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP}$ = AFL3 · $\overline{AFNC}$ · BCLK AFNP = AFL3 · AFNC · BCLK	Remains reset because no summation is performed
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$\overline{JE37}$ · JIN1 · JIN2	Enables incrementation of J Counter until J = 37_8
F1ETST	89	ETST = G0ESTF = $\overline{IR06}$ · IR07 · ET4E · EG1T $\overline{ETST}$ = G1ESTF · $\overline{IR06}$ · ECLK G1ESTF = IR05 · EGIT · AFNA · ASR1	Set Test Flip-flop; Clear if any bit shifted from A_0 = 1

TZC-TEST ZERO AND COMPLEMENT

GEN 1

23							0
0	5	0	6	4	6	7	0

TZC sets the Test flip-flop (F1ETST) if all bits in the Register are "zero". If any bit in the A Register is a "one", the Test flip-flop is cleared. TZC also replaces the contents of the A Register with its 1's complement (i.e. "ones" are changed to "zeros" and "zeros" are changed to "ones").

During State 4, the complement of I_{4-0} is gated to the J Counter. This presets the J Counter to 7 because bits 2 through 0 of the TZC command are "zeros". The Test flip-flop (F1ETST) is set. The A Register is then shifted right with A_0 applied to the A input of the Serial Full Adder. The Carry flip-flop (F1AFNP) is held set throughout the shift of A applying a "one" to the P input of the Adder. The Sum output of the Adder is equal to the 1's complement A_0 (i.e. A_0 plus 1 = complement of A_0). This sum output is applied to A_{23} placing the 1's complement of A back in A. If at any shift of A, A_0 is a "one", the Test flip-flop is cleared. The J Counter is incremented at each shift of A. When J is equal to 37_8, 24 shifts of A have occurred and further shifting is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$C(\overline{A_{23-0}})$
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if C(A) = 0; Reset if C(A) ≠ 0
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	AR00 · ANA2	Gates A_0 to A input of Adder
F1AFNP	66	AFL2 + $\overline{AFL3}$	Held set
D1ASRL, U	63.1	ASR1 = DGN1 · T6E3 · $\overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$\overline{JE37}$ · JIN1 · JIN2	Enables incrementation of J Counter until J = 37_8
F1ETST	89	ETST=G0ESTF=$\overline{IR06}$· IR07· ET4E· EG1T $\overline{ETST}$ = G1ESTF · $\overline{IR06}$ · ECLK G1ESTF = IR05· EGIT· AFNA· ASR1	Set Test flip-flop; Clear if C(A) ≠ 0

TZE-TEST A ZERO

GEN 1

23							0
0	5	0	0	4	6	7	0

TZE sets the Test flip-flop (F1ETST) if all bits in the A Register are "zero". If any bit in the A Register is a "one", the Test flip-flop is cleared. The original contents of A are unchanged.

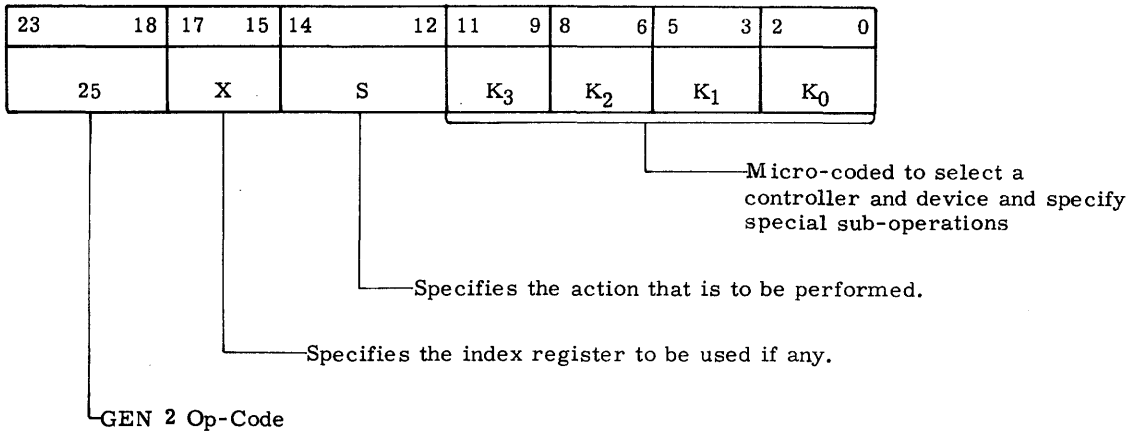
During State 4, the complement of I_{4-0} is gated to the J Counter. This presets the J Counter equal to 7 because bits 2 through 0 of the TZE command are "zeros". The Test flip-flop is set. The A Register is then shifted right circular with A_0 applied to the A input of the Serial Full Adder. All other inputs of the Adder are inhibited providing a Sum output of the Adder equal to A_0. This Sum output is applied to A_{23}, leaving the A Register unchanged after 24 shifts. If at any shift of A, A_0 is a "one", the Test flip-flop is cleared. The J Counter is incremented at each shift of A When J is equal to 37_8, 24 shifts of A have occurred and further shifting is inhibited.

Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C(P) + 1
F1WPMT	
F1UOFL	
F1ETST	Set if C(A) = 0; Reset if C(A) ≠ 0
J_{4-0}	37_8
Memory Z	

COMMAND CHARACTERISTICS

Logic Element	Logic Sheet	Logic Equation	Function
G0AFNB	67	$\overline{AFNB} = \overline{IR10} \cdot \overline{IR09}$	Disabled
G1AFNB	67	$\overline{AFNB} = (\overline{AI10} \cdot \overline{AR00}) + (\overline{AI09} \cdot AR00)$	Disabled
G0AFNA	65	$AR00 \cdot ANA2$	Gates A_0 to A input of Adder
F1AFNP	66	$\overline{AFNP} = AFL3 \cdot \overline{AFNC} \cdot BCLK$ $AFNP = AFL3 \cdot AFNC \cdot BCLK$	Remains reset because no summation is performed
D1ASRL, U	63.1	$ASR1 = DGN1 \cdot T6E3 \cdot \overline{JE37}$	Shifts A Reg. right until J = 37_8
N1J1NC	71	$\overline{JE37} \cdot JIN1 \cdot JIN2$	Enables incrementation of J Counter until J = 37_8
F1ETST	89	$ETST=G0ESTF=\overline{IR06} \cdot IR07 \cdot ET4E \cdot EG1T$ $\overline{ETST} = G1ESTF \cdot \overline{IR06} \cdot ECLK$	Set Test flip-flop; Clear if any bit shifted from $A_0 = 1$

GEN 2 INPUT/OUTPUT COMMANDS



GEN 2 commands are micro-coded and used primarily to control input/output subsystems. GEN 2 commands, however, are also used to perform some operations internal to the Central Systems Unit. To further illustrate the microcoding of the GEN 2 format, Fig. GN2.1, 2, 3, 4, 5 and 6 are provided. These figures illustrate the specific microcoded functions of GEN 2 commands when operating various groups of controllers. While data may be exchanged between the A Register and input/output modules by means of GEN 2 commands, in normal system operation the TIM/TOM hardware is used to provide faster and more direct data transfers.

The GEN 2 command is identified by the operation code 25_8. GEN 2 commands are divided into two categories, internal and external, according to the K3 bits (11 thru 9). If the K3 bits are equal to 0, the command is internal. If any of the K3 bits is a "one", the command is external.

External commands are Input/Output commands that select modules and devices for inputting and outputting data. A single GEN 2 command will select and address a specific device and specify the action that is to be performed. The selection of a device and the transfer of data is accomplished with a single GEN 2 command.

Internal GEN 2 commands are used to control the API module, test for Parity, Overflow, or Demand indications, read the console switches, reset the Stall Alarm, turn off or on alarm contacts, turn on and off the adjustable pulse generator and set the Trapping Mode.

GEN 2 commands cannot be relative addressed. They may be interrupted upon completion except for IAI, IAI_2, LMR_1, LMR_2, JNR, JDR, JCB and JNO. An interrupt may not occur immediately following a PAI command that is executed after an IAI_2 command has been executed.

Since indexing may alter bit positions 14-0 of the GEN 2 command, caution must be exercised when specifing an index register.

The following table lists the GEN 2 commands as defined by the S (14-12) and K3 (11-9) bits.

S Bit	Internal (K3=0)	External (K3≠0)
0	STMF, LMR_1, LMR_2, IAI_2, RALM, SALM, RAPG, SAPG	SEL
1	SSA	ACT
2	PAI	OPR
3	IAI	ABT
4	JND	OUT
5	RCS	IN
6	JNO	JNR, JDR, JCB
7	JNP	JNE

BASIC TIMING

The basic timing of GEN 2 commands is divided into three groups; internal (K3=0), high-speed external (K3=4 or 7), and low-speed external (K3=1 or 2). The basic timing of each of these groups is described in the following paragraphs. Following this discussion of basic timing, each GEN 2 command is described individually.

Internal GEN 2 Commands

GEN 2 commands with K3=0 perform various functions within the Central Processor and are executed at very high speeds (i.e. 2.2 μs.).

Fig. GN2.7 contains a timing diagram and logic equations for GEN 2 commands with K3=0. These commands are executed during Sequence State 4 which has a duration of 0.6 microseconds.

Since memory is not required to execute GEN 2 commands, memory is not requested and the Sequence Time Counter (F1TSCA-C) is not held in Time 2 ($\overline{SCA}\cdot SCB\cdot SCC$) awaiting Data Ready. Instead, it is allowed to increment to Time 5 ($SCA\cdot\overline{SCB}\cdot\overline{SCC}$) which generates Last Pulse Envelope (D1TLPE), ending the execution cycle.

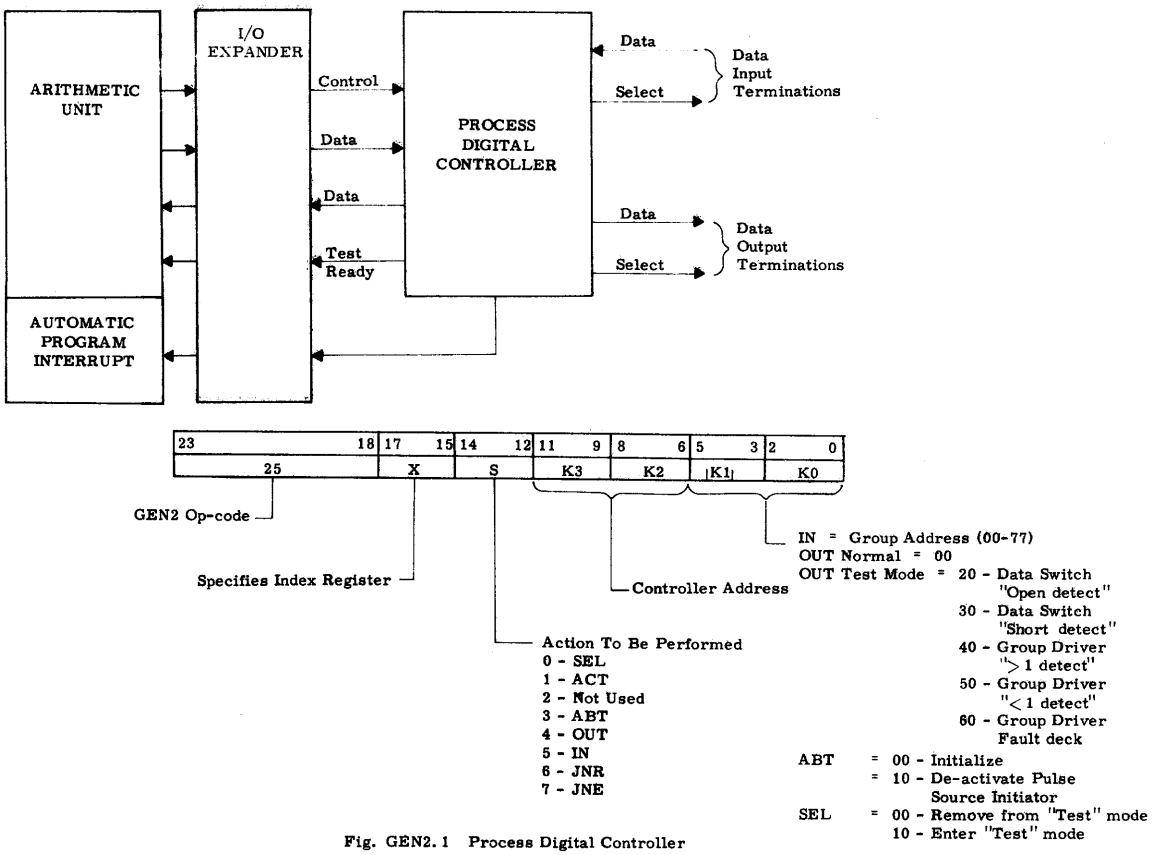


Fig. GEN2.1 Process Digital Controller

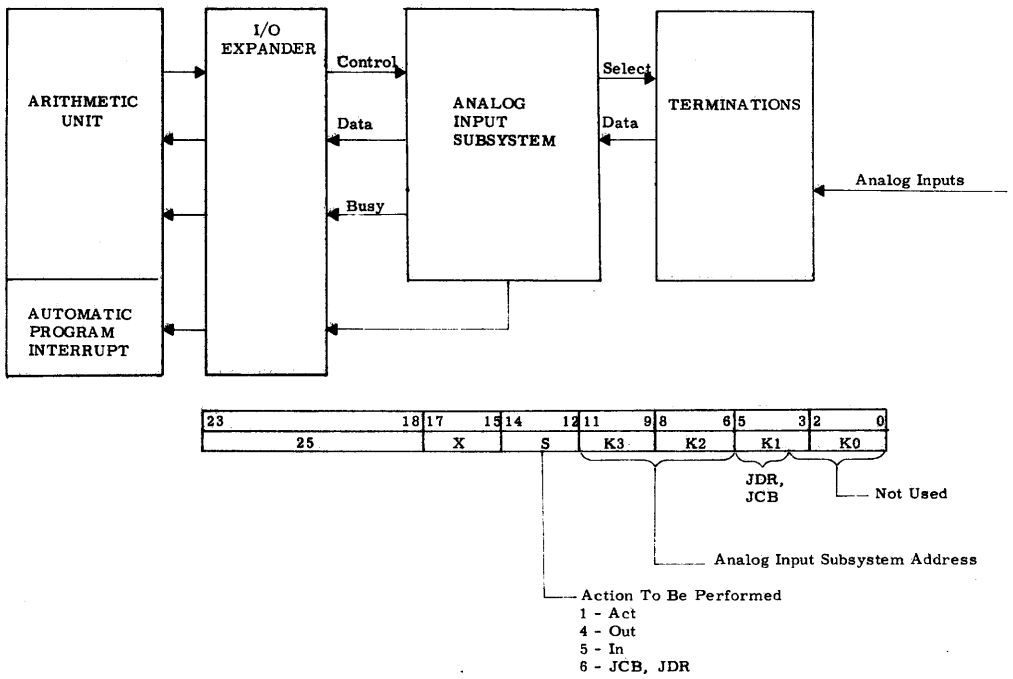
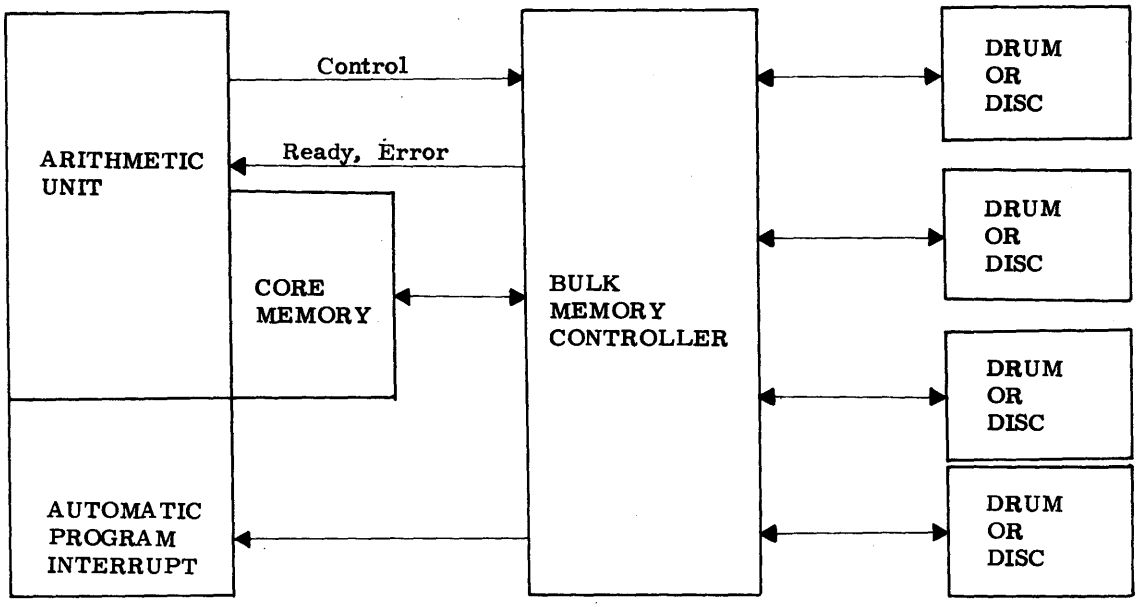
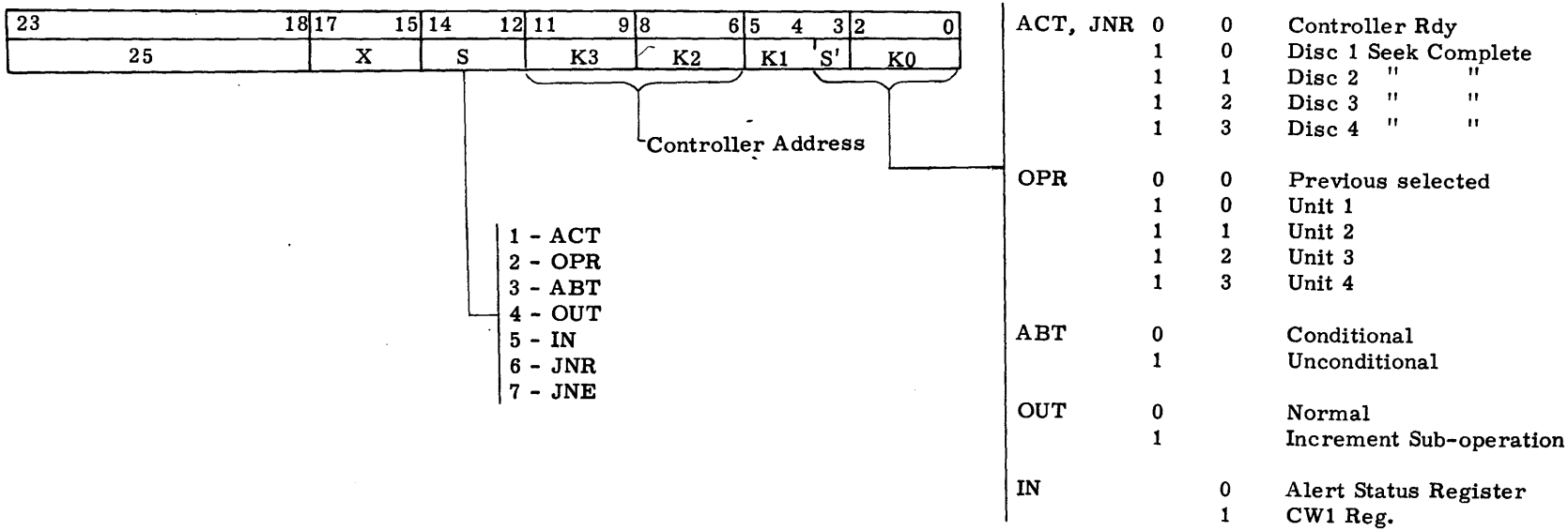


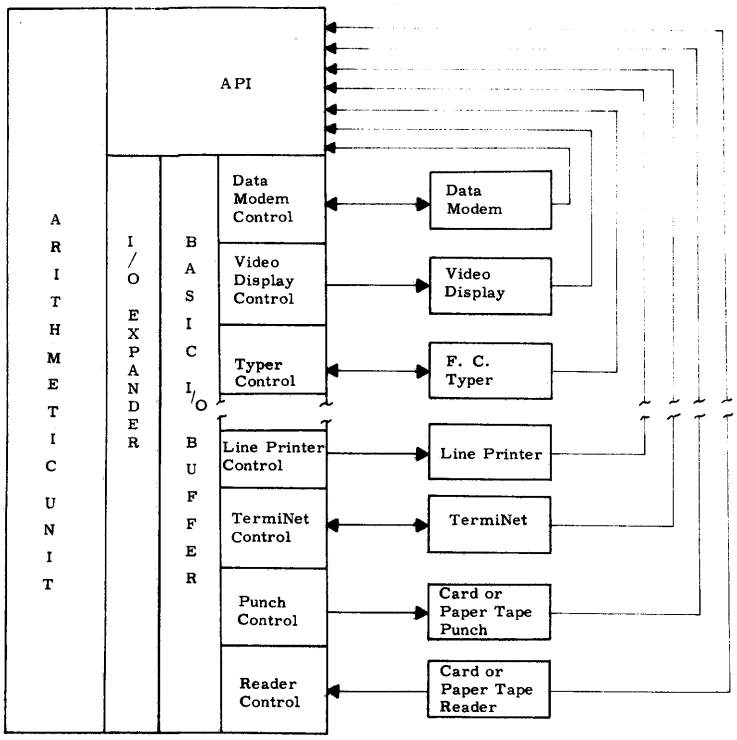
Fig. GEN2.2 Analog Input Subsystem





	S^1	$K0_8$	
ACT, JNR	0	0	Controller Rdy
	1	0	Disc 1 Seek Complete
	1	1	Disc 2 " "
	1	2	Disc 3 " "
	1	3	Disc 4 " "
OPR	0	0	Previous selected
	1	0	Unit 1
	1	1	Unit 2
	1	2	Unit 3
	1	3	Unit 4
ABT	0		Conditional
	1		Unconditional
OUT	0		Normal
	1		Increment Sub-operation
IN		0	Alert Status Register
		1	CW1 Reg.
		2	CW2 Reg.
		3	CW3 Reg.
		4	Data Buffer Reg.
		5	Accumulator Reg.
		6	Cont. F/F L Counter Diff. Reg.
		7	Check Sum Accumulator

Fig. GEN2.3 Bulk Memory Controller



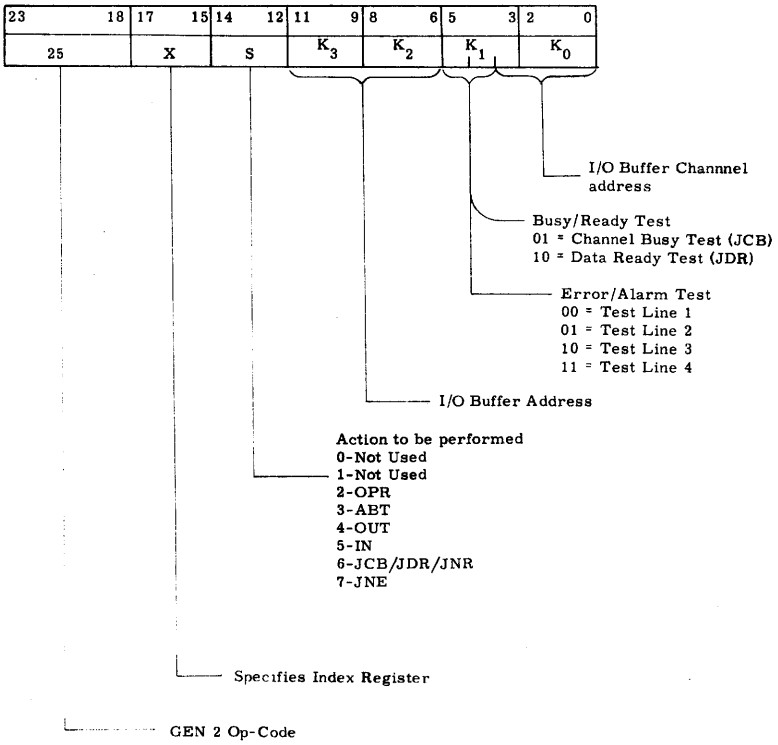
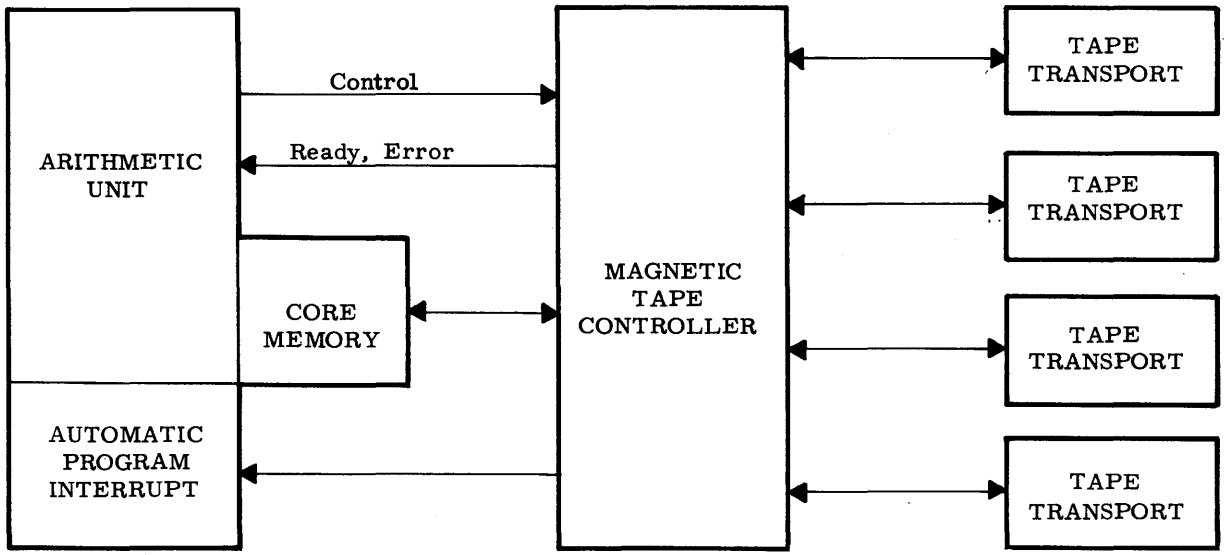


Fig. GN2.4 I/O Buffer Application



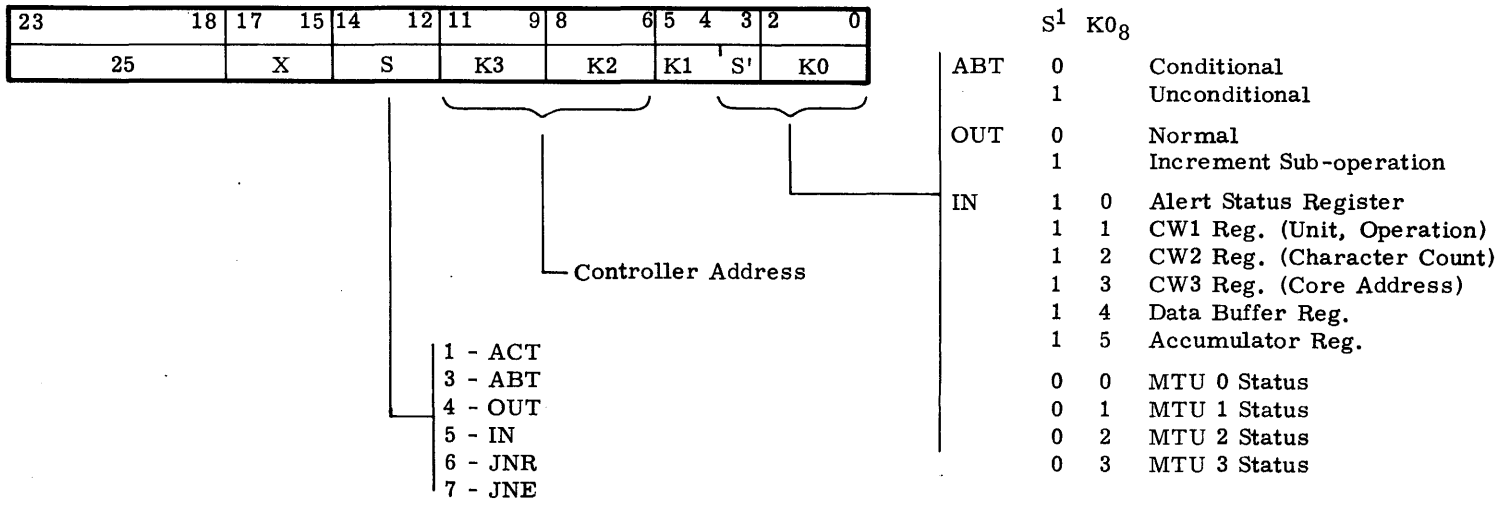


Fig. GEN2.5 Magnetic Tape Controller

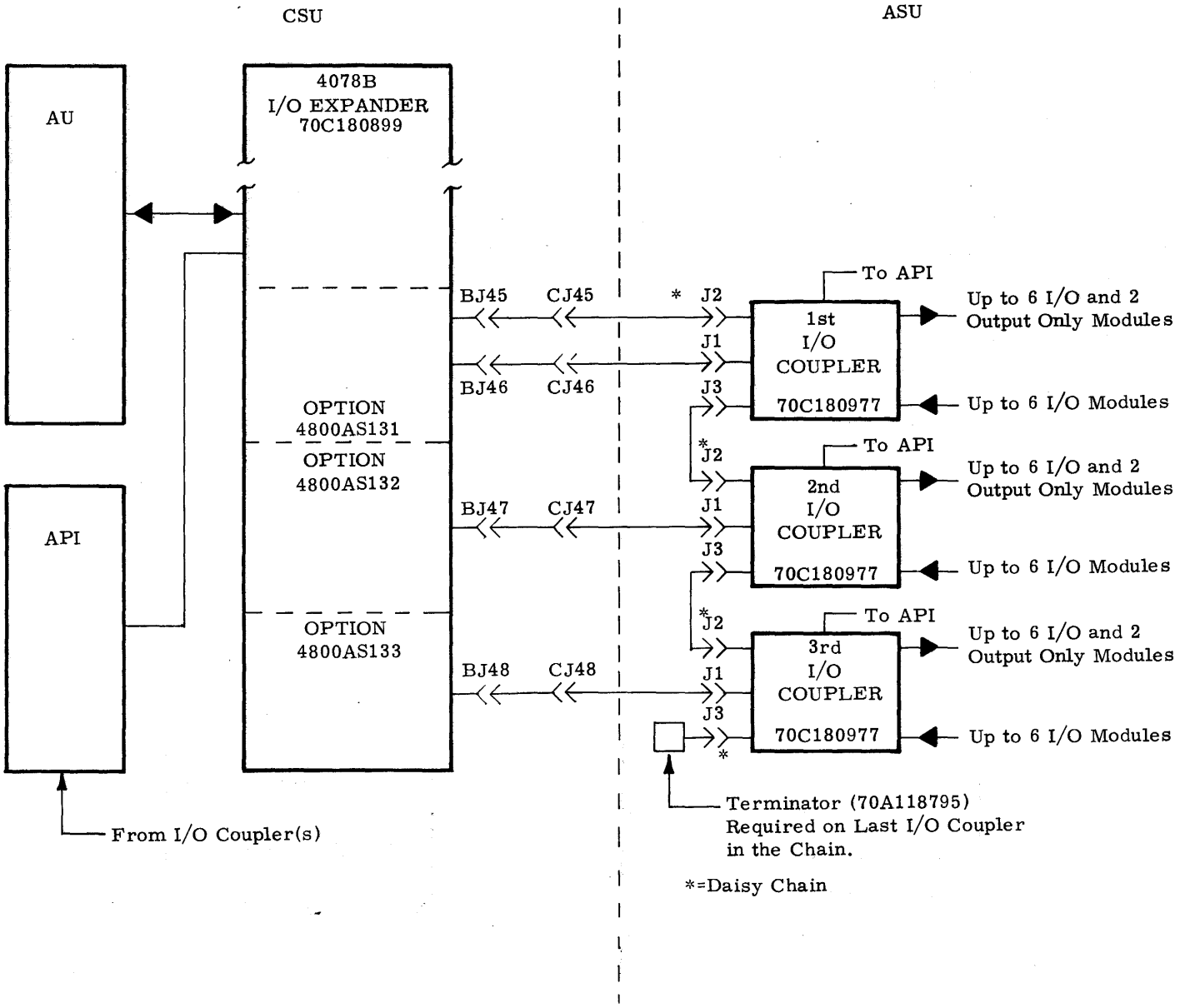


Fig. GN2.6 Optional I/O Coupler Applications

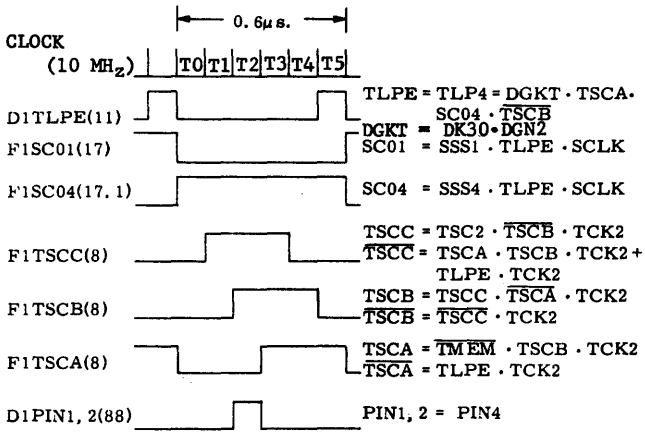


Fig. GN2.7 Internal GEN 2 Basic Timing

High-Speed External GEN 2 Commands

High-speed GEN 2 commands are those GEN 2 commands having the K3 bits equal to 4 or 7. These commands are used with I/O Devices that operate at high-speeds. High-speed GEN 2 commands are executed in 8.5 microseconds.

Fig. GN2.8 illustrates the basic timing signals and the associated logic equations for the execution state (State 4) of GEN 2 commands with K3 equal to 4 or 7. Sequence Control State 4 has a duration of 6.9 microseconds for these commands. The duration of State 4 is controlled primarily by Time 6 Envelope (F1TT6E), the Delay Time (binary) Counter (F1TAFF-F1TEFF), and the Extended Time Counter flip-flops (F1TEC1, 2).

The Sequence Time Counter (F1TSCA, B, C) is incremented by the first six Clock pulses of State 4 to Time 6 Envelope (F1TT6E). At Time 3 (SCA·SCB·SCC) of the Sequence Time Counter, the Delay Time Counter is preset to a count of 30_8 by G1TPAF, and the Extended Time Counter flip-flops (F1TEC1, 2) are set. At Time 4 (SCA·SCB·$\overline{SCC}$) of the Sequence Time Counter, the Delay Time Counter is preset to a count of 4 by D0TP04. During Time 6 Envelope, the Delay Time Counter is incremented by each Clock pulse until it is equal to 30_8. When the Delay Time Counter is equal to 30_8, F1TEC2 is reset and the Delay Time Counter is again preset to 4 by D0TP04. The Delay Time Counter is again incremented until it is equal to 30_8. At this time, F1TEC1 is reset and the Delay Time Counter is again preset to 4 by D0TP04. F1TEC2 is set if the command is not a JNE, JNR, JDR or JCB command with the jump condition true (i.e. if $\overline{G1TP1X}$). The Delay Time Counter is again incremented by each Clock pulse until it is equal to 30_8. At this time, with the Delay Time Counter equal to 30_8 and the Extended Time Counter flip-flops (F1TEC1, 2) reset, Last Pulse Envelope (D1TLPE) is enabled to end the execution cycle.

Phase A and Phase B timing signals with a duration of 2.1 microseconds are applied to the I/0 Subsystem by D0NPHA and D0NPHB. Notice that D0NPHB is not enabled for external jump commands (JNE, JNR, JCB, JDR) when the jump condition is true.

Low-Speed External GEN 2 Commands

External GEN 2 commands have the K3 bits equal to 1 or 2 are executed in 26.5 microseconds. Most external GEN 2 commands are in this category.

Fig. GN2.9 illustrates the basic timing signals and the associated logic equations for the execution state (State 4) of these external GEN 2 commands. Sequence Control State 4 has a time duration of 24.9 microseconds. This time duration is controlled in much the same manner as the High-Speed GEN 2 commands described above, except that the Delay Time Counter is incremented by every fourth Clock pulse instead of at every Clock pulse. The Frequency Divider (F1TED1, 2) is used to provide the Delay Time Counter triggers at every 4th Clock pulse of these external GEN 2 commands.

The Sequence Time Counter (F1TSCA, B, C,) is incremented by the first six Clock pulse of State 4 to Time 6 Envelope (F1TT6E). At Time 3 (SCA·SCB·SCC) of the Sequence Time Counter, the Delay Time Counter (F1TAFF-F1TEFF) is preset to 30_8 by G1TPAF and the Extended Time Counter flip-flops (F1TEC1, 2) are set. At Time 4 (SCA·SCB·$\overline{SCC}$) of the Sequence Time Counter, the Delay Time Counter is preset to 4 by D0TP04. Each Clock pulse of Time 6 Envelope increments the Frequency Divider flip-flops (F1TFD1, 2), when the Delay Time Counter is not equal to 30_8. F1TFD1 and F1TFD2 act as a binary counter and are, therefore, both set at every fourth Clock pulse. When both F1TFD1 and F1TFD2 are set, G0TDFD is enabled. When enabled, G0TDFD enables G1TEID and allows incrementation of the binary Delay Time Counter (F1TAFF-F1TEFF). Therefore, every fourth Clock pulse increments the Delay Time Counter until it is equal to 30_8. When the Delay Time Counter is equal to 30_8, the Extended Time Counter flip-flop, F1TEC2, is cleared and the Delay Time Counter is preset to 4 by D0TP04. The Delay Time Counter is again incremented at every fourth Clock pulse until it is again incremented to 30_8. At this time the Extended Time Counter flip-flop, F1TEC1, is reset and the Delay Time Counter is again preset to 4 by D0TP04. The Delay Time Counter is again incremented by every fourth Clock pulse until equal to 30_8. At this time, with F1TEC1 and F1TEC2 reset and the Delay Time Counter equal to 30_8, Last Pulse Envelope (D1TLPE) is enabled ending the execution cycle.

Phase A and Phase B timing signals with a duration of 8.1 microseconds are applied to the I/0 Subsystem by D0NPHA and D0NPHB. Notice that D0NPHB is inhibited during external jump commands (JNE, JNR, JCB, JDR) when the jump condition is true.

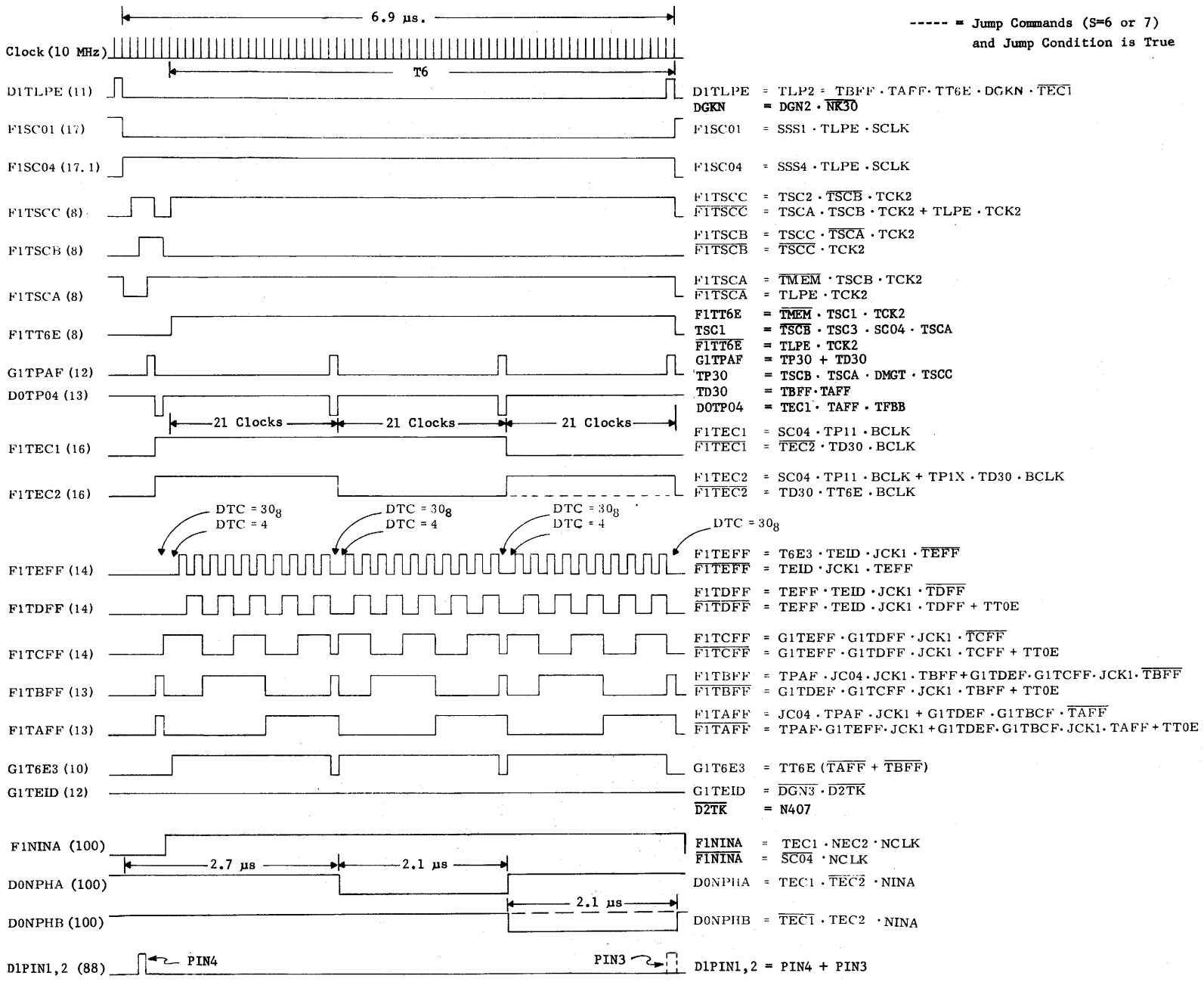


Fig. GN2.8 High-Speed External GEN 2 Basic Timing

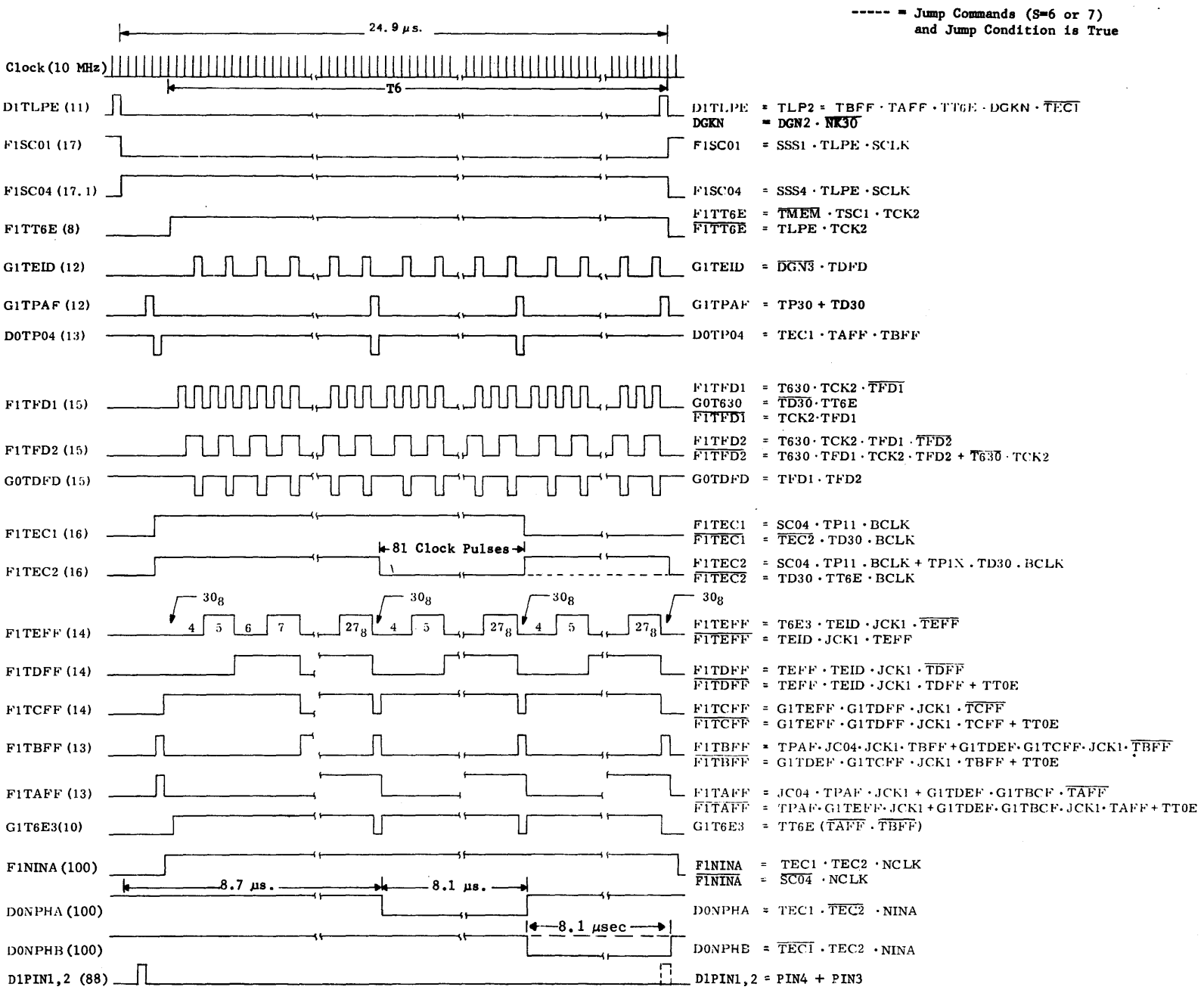


Fig. GN2.9 External GEN 2 (K3=1 or 2) Basic Timing

ABT-ABORT DEVICE D's OPERATION

GEN 2 (S= 3, K3≠ 0)

23 18	17 15	14 12	11 0
25	X	3	D

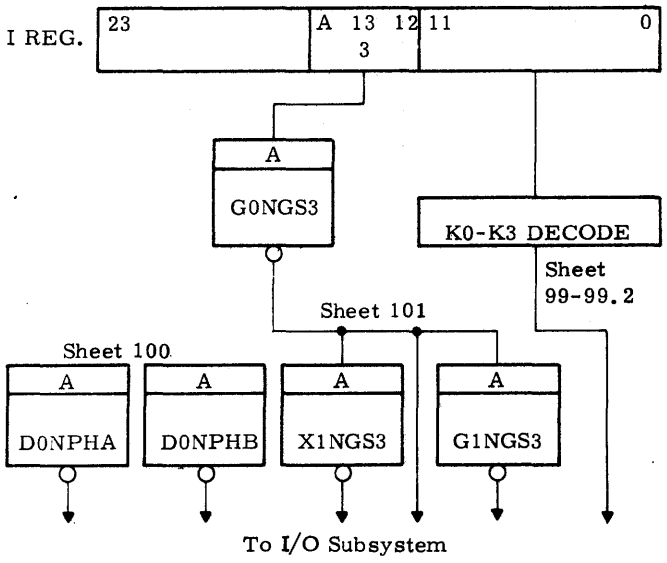
Z= f(X, D)

ABT terminates the operation of the addressed device and initializes the addressed channel. The operation may or may not have been completed, thus, ABT provides an interrupt to indicate that the ABT command has been executed. Some device operations cannot be terminated instantly, (Teletypewriter, Magnetic Disc seek) therefore, the program must account for this delay to complete termination of the operation. For example, the Teletypewriter requires 100 ms to transmit one character. If ABT is executed before the full character has been transmitted, the channel will be initialized but the Teletypewriter unit must complete its cycle. Therefore, if another command is issued to the Teletypewriter before it completes that cycle, a synchronization problem occurs.

> NOTE
>
> The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.

Since ABT is an external GEN 2 command (K3≠0), the duration of Sequence Control State 4 is 6.9 microseconds if K3 is equal to 4 or 7, or 24.9 micro-seconds if K3 is equal to 1 or 2. The timing diagram contained in Fig. GN2.8 or GN2.9 applies to the ABT command.

Because the S bits (14-12) of the ABT command are equal to 3, the NGS3 gates are enabled by the corresponding bits of the I Register. Bits 11 through 0 of the ABT command are decoded from the I Register, enabling the corresponding K decode gates. These signals, along with the Phase A and Phase B timing signals, are applied to the I/O Subsystem.



ABT Block Diagram

ACT-ACTIVATE DEVICE D's INTERRUPT

GEN 2 (S= 1, K3≠ 0)

23 18	17 15	14 12	11 0
25	X	1	D

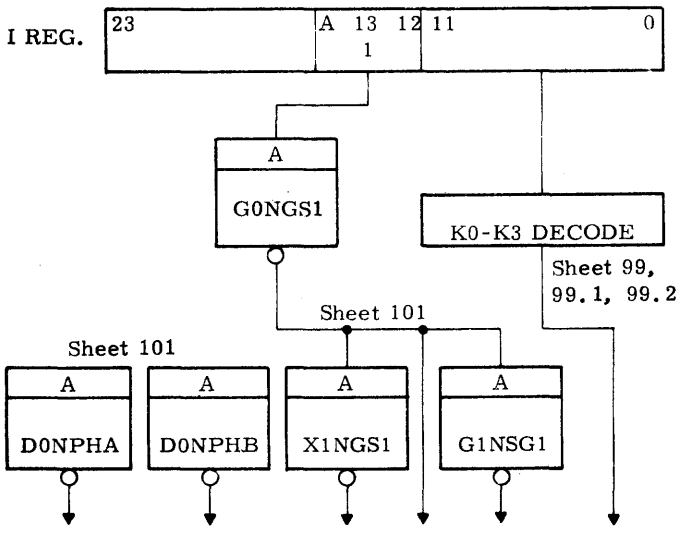
Z= f(X, D)

ACT indirectly initiates programmed operation of device Z by initiating an Automatic Program Interrupt from the device. If device Z is not in operation, ACT simulates the completion of an operation to provide the Automatic Program Interrupt (i.e. the ready signal is cycled to not ready and back to ready). If device Z is in operation, the ACT command is ignored. The ACT command may be used to initiate subroutines for devices connected to modules such as the Analog Input Subsystem Controller, Process Digital and Bulk Memory Controller.

> NOTE
>
> The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.

Since the ACT command is an external command (K3≠0), the duration of Sequence Control State 4 will be 6.9 microseconds if K3 is equal to 4 or 7 or 24.9 microseconds if K3 is equal to 1 or 2. The timing diagram shown in Fig. GN2.8 or GN2.9 applies directly to the ACT command.

The S (bits 14-12) and K (bits 11-0) bits of the ACT command are decoded from the I Register. Since S is equal to 1, the NGS1 gates are enabled. This signal along with the K decode gates enabled and the Phase A and Phase B timing signals are applied to the addressed I/O Subsystem.



ACT Block Diagram

IAI INHIBIT AUTOMATIC INTERRUPT

GEN 2 (S= 3, K3= 0)

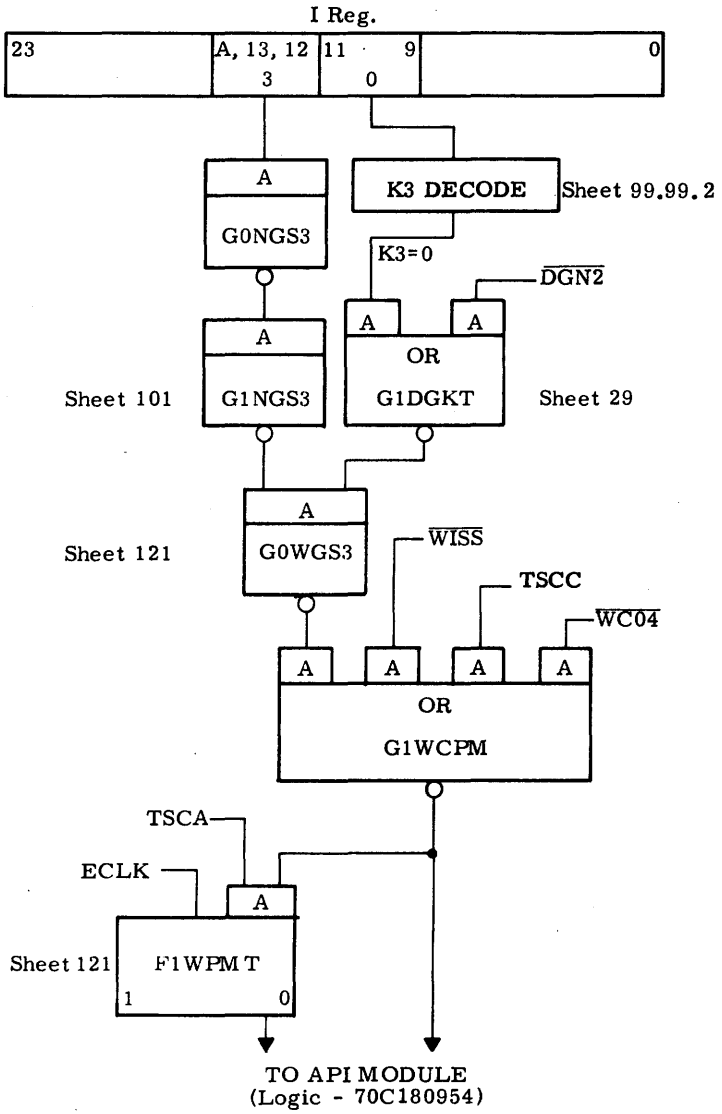
23 0
2 5 0 3 0 0 0 0

IAI clears the Permit Automatic Interrupt flip-flop

(F1WPMT) to inhibit inhibitable program interrupts. Non-inhibitable interrupts are not affected by the Permit Automatic Interrupt flip-flop.

Since the IAI command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the IAI command.

The S bits of the IAI command are decoded from $I_{A,\ 13,\ 12}$ to enable the NGS3 gates. The K3 bits are decoded from I_{11-9} and enable the NK30 gates. At time 4 (TSCA·$\overline{TSCC}$) the Permit Automatic Interrupt flip-flop (F1WPMT) is reset. F1WPMT when reset, applies a signal to the Automatic Program Interrupt module inhibiting inhibitable type interrupts.



IAI Block Diagram

IAI_2-INHIBIT AUTOMATIC INTERRUPT

GEN 2

(S=0, K3=0)

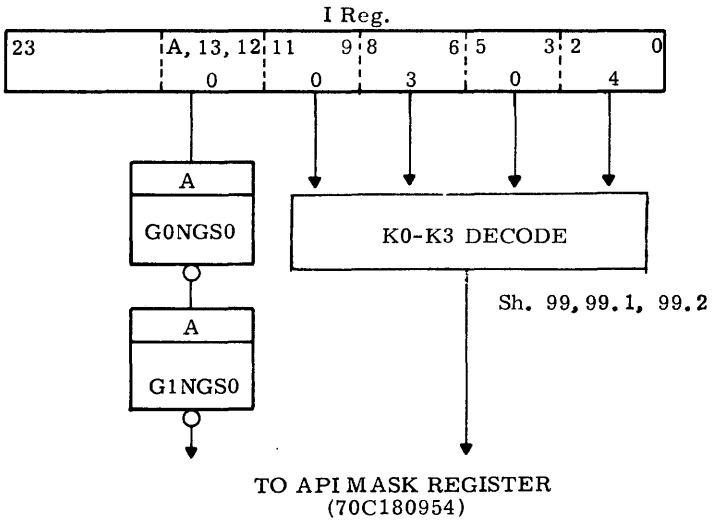
23	0
2 5 0 0 0 3 0 4	

IAI_2 is used in conjunction with the Automatic Interrupt Mask Register. IAI_2 inhibits all interrupts, both inhibitable and non-inhibitable provided the Permit Automatic Interrupt flip-flop has been previously cleared.

Interrupts may subsequently be permitted by executing a PAI instruction or on LPR with bit 21 of the information word set.

Since the IAI_2 command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the IAI_2 command.

The S and K bits of the IAI_2 command are decoded from the I Register. Decoding these bits apply the S=0 and corresponding K decode signals to the optional API Mask Register logic. From these signals, the Mask Register logic inhibits all interrupts by clearing the Time Counter, clearing the Priority Interrupt flip-flops and disabling Echo generation.



IAI_2 Block Diagram

IN-INPUT FROM DEVICE D

GEN 2

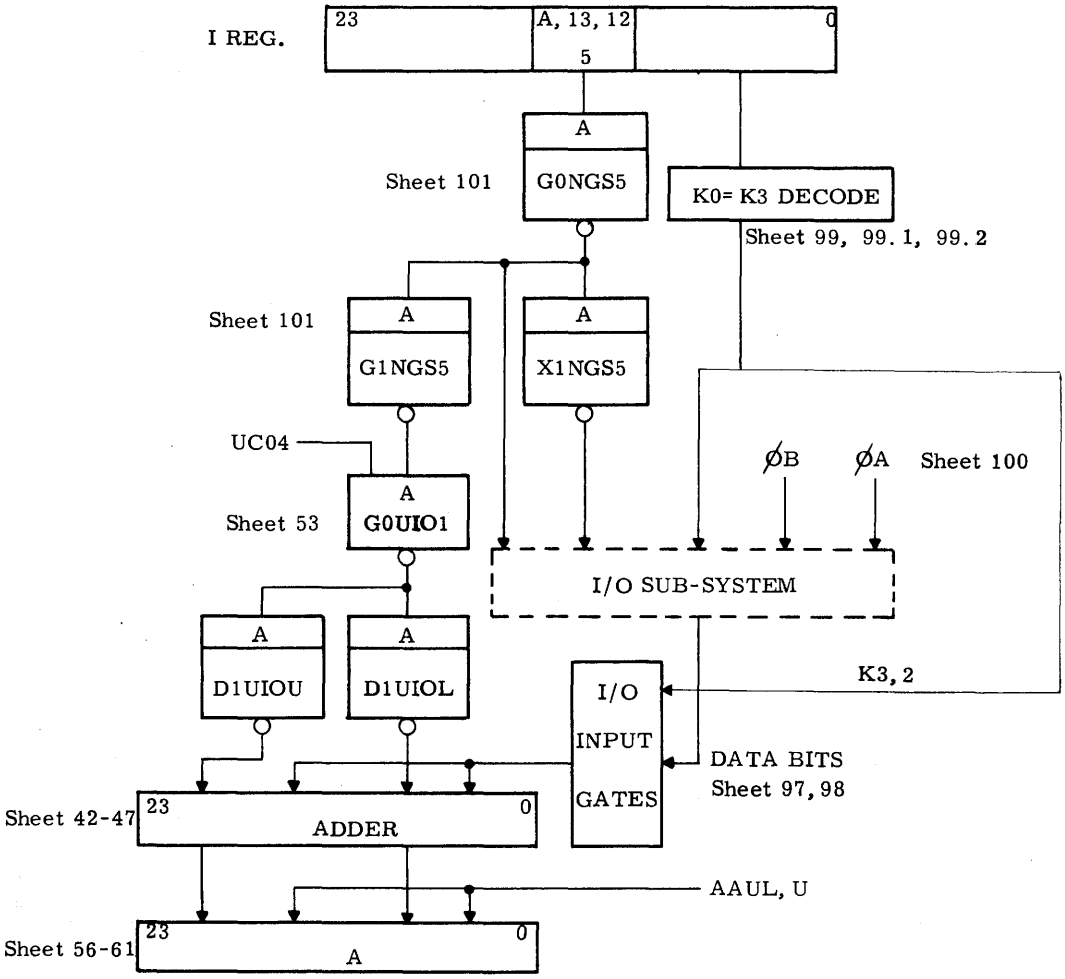
(S=5, K3≠0)

23 18	17 15	14 12	11 0
25	X	5	D

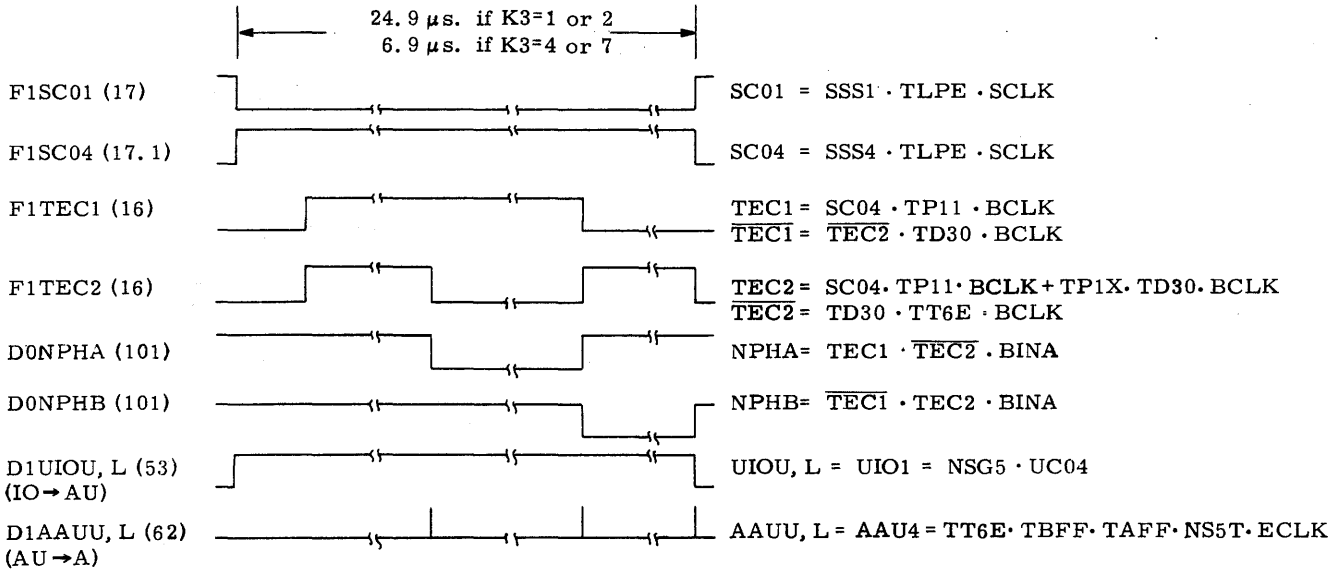
Z= f(X, D)

IN transfers data from the addressed device to the A Register and initiates the next operation of the device.

> NOTE
>
> The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.



IN Block Diagram



IN Timing Diagram

The duration of State 4 is 24.9 microseconds if the K3 bits are equal to 1 or 2, or the duration is 6.9 microseconds if the K3 bits are equal to 4 or 7. The basic timing diagram in Fig. GN2.8 or GN2.9 applies to the IN command.

Since the S bits of the IN command are equal to 5, X1NGS5 is enabled and the resultant signal is applied to the I/O Sub-system along with the Phase A and Phase B timing signals and the decoded address signals (K0 - K3). Data from the addressed subsystem is applied through the I/O Input Gates (logic sheets 97 and 98) and gated to the Adder Unit by D1UIOU, L. From the Adder Unit, the data bits are gated to the A Register by D1AAUU, L.

JCB-JUMP IF CHANNEL BUSY

GEN 2 (S = 6, K3 ≠ 0)

23 18	17 15	14 12	11 6	5	4	3 0
25	X	6	D	0	1	D

JCB transfers program control to the second sequential location (P + 2) if the addressed channel is busy. If the addressed channel is ready (not busy), program control is transferred to the first sequential location (P + 1).

Hardware operation of the JCB command within the Arithmetic Unit is identical to that of the JDR and JNR commands. Refer to the description of the JNR command for further details.

JDR-JUMP IF DATA READY

GEN 2 (S = 6, K3 ≠ 0)

23 18	17 15	14 12	11 6	5	4	3 0
25	X	6	D	1	0	D

JDR transfers program control to the second sequential location (P + 2) if the addressed input channel is busy and its data ready indicator is set. If the indicator of the addressed channel is reset, program control is transferred to the first sequential location (P + 1). When the addressed channel is not busy, program control is transferred to the first sequential location (P + 1).

Hardware operation of the JDR command within the Arithmetic Unit is identical to that of the JCB and JNR commands. Refer to the description of the JNR command for further details.

JND-JUMP IF NO DEMAND

GEN 2 (S=4, K3=0)

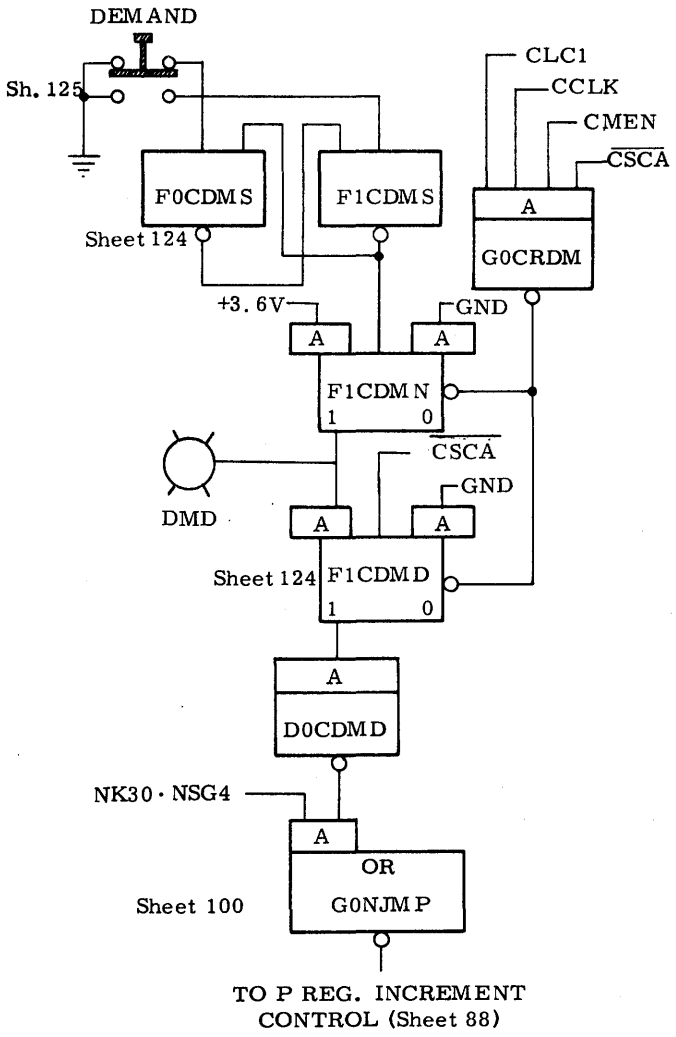
23 0
2 5 0 4 0 0 0 0

JND transfers program control to the second sequential location (P + 2) if the Demand flip-flop (F1CDMD) is reset. If the Demand flip-flop is set, JND clears it and advances program control to the first sequential location (P + 1). The Demand flip-flop is set by pressing and then releasing the DEMAND switch on the computer console.

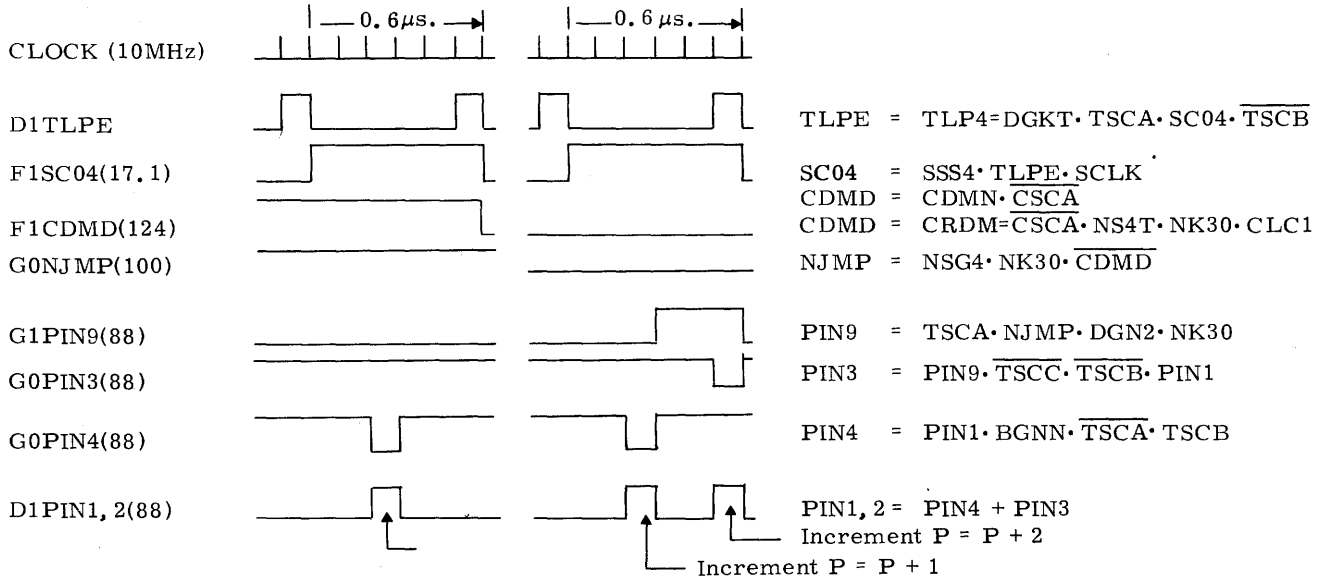
Since the JND command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the JND command.

The P Register is always incremented during Time 2 ($\overline{TSCA}$· TSCB) of the JND command. If the Demand flip-flop (F1CDMD) is in the reset state, the P Register is again incremented at Time 5 (TSCA· $\overline{TSCB}$· $\overline{TSCC}$). If the Demand flip-flop is set, the JND command resets it during time 0 of the following State 1.

The Demand flip-flop is set by pressing and then releasing the DEMAND switch on the computer console. Pressing the DEMAND switch applies a ground to F1CDMS providing a "one" output to arm F1CDMD. Releasing the



JND Block Diagram



JND Timing

DEMAND switch removes the ground from F1CDMS and applies it to F0CDMS. This provides a "one" output from F0CDMS which is applied to F1CDMS causing its output to fall to "zero" thereby setting F1CDMD. F1CDMD is cleared by the decoded JND command (NS4T· NK30) at the beginning of the following State 1.

JNE-JUMP IF DEVICE D NOT IN ERROR

GEN 2

(S=7, K3≠0)

23 18	17 15	14 12	11 0
25	X	7	D

Z=f(X, D)

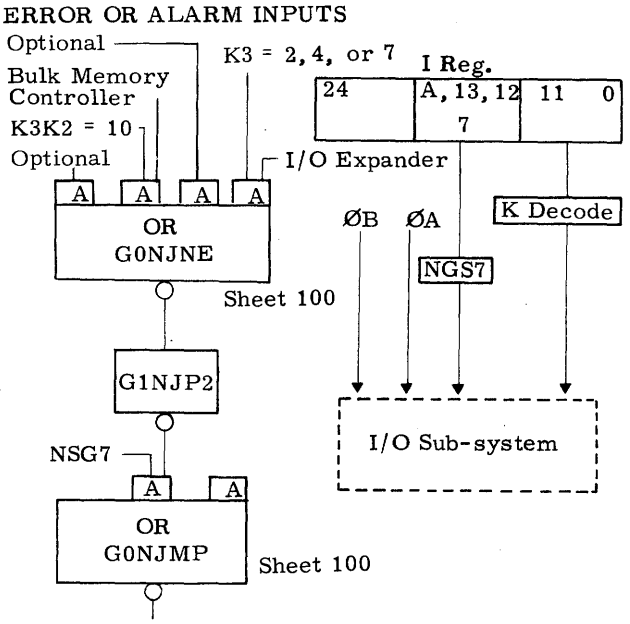
JNE transfers program control to the second sequential location (P + 2) if no error or alarm exists in the addressed module. If an error or alarm exists, program control is transferred to the first sequential location (P + 1). Depending on the module addressed, the JNE command may also clear the error or alarm indicator in the module.

> NOTE
>
> The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.

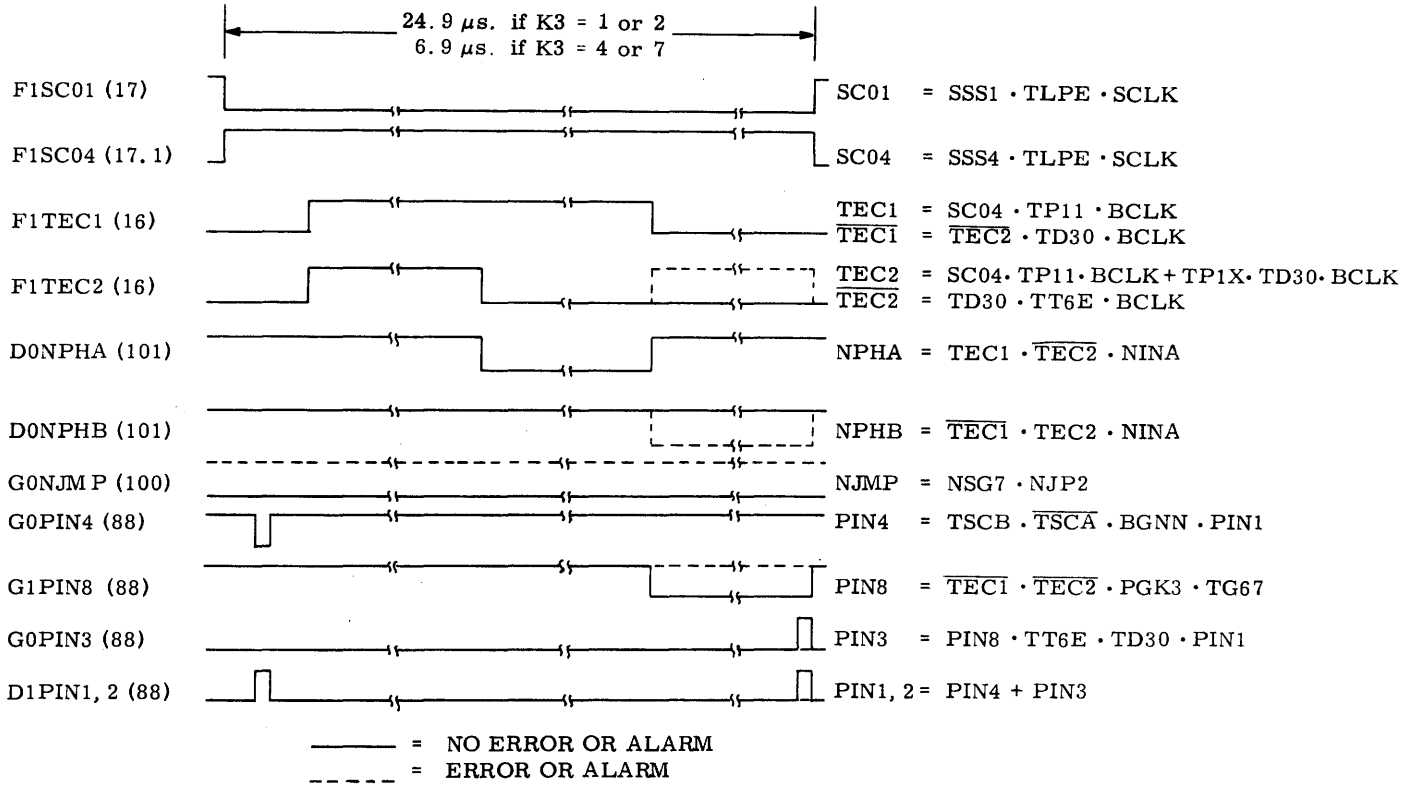
The duration of Sequence Control State 4 is 24.9 microseconds if K3 (bits 11 - 9 of the JNE command) is equal to 1 or 2 or the duration is 6.9 microseconds if K3 of the JNE command is equal to 4 or 7. Refer to the basic timing diagram contained in Fig. GN2.8 or GN2.9.

The S bits of the JNE command are decoded from $I_{A, 13, 12}$ to enable NGS7. This signal, along with the decoded K3-0 signals and Phase A and Phase B, are applied to the I/O Subsystem. The P Register is incremented (G0PIN4) at Time 2 of State 4. The error or alarm signal from the addressed module is applied through G0NJNE. If an error or alarm exists, G0NJMP is disabled. When disabled, G0NJMP enables setting of F1TEC2 when the Delay Time Counter is equal to 30_8.

Setting F1TEC2 enables generation of the Phase B signal to the addressed module and inhibits incrementing of the P Register for the second time. If, howerver no error or alarm exists, G0NJMP is enabled. When enabled, setting of F1TEC2 is inhibited when the Delay Time Counter is equal to 30_8. Inhibiting F1TEC2, inhibits generation of Phase B and enables G1PIN8. G1PIN8 enables G0PIN3 causing the P Register to be incremented a second time (P + 2).



JNE Block Diagram

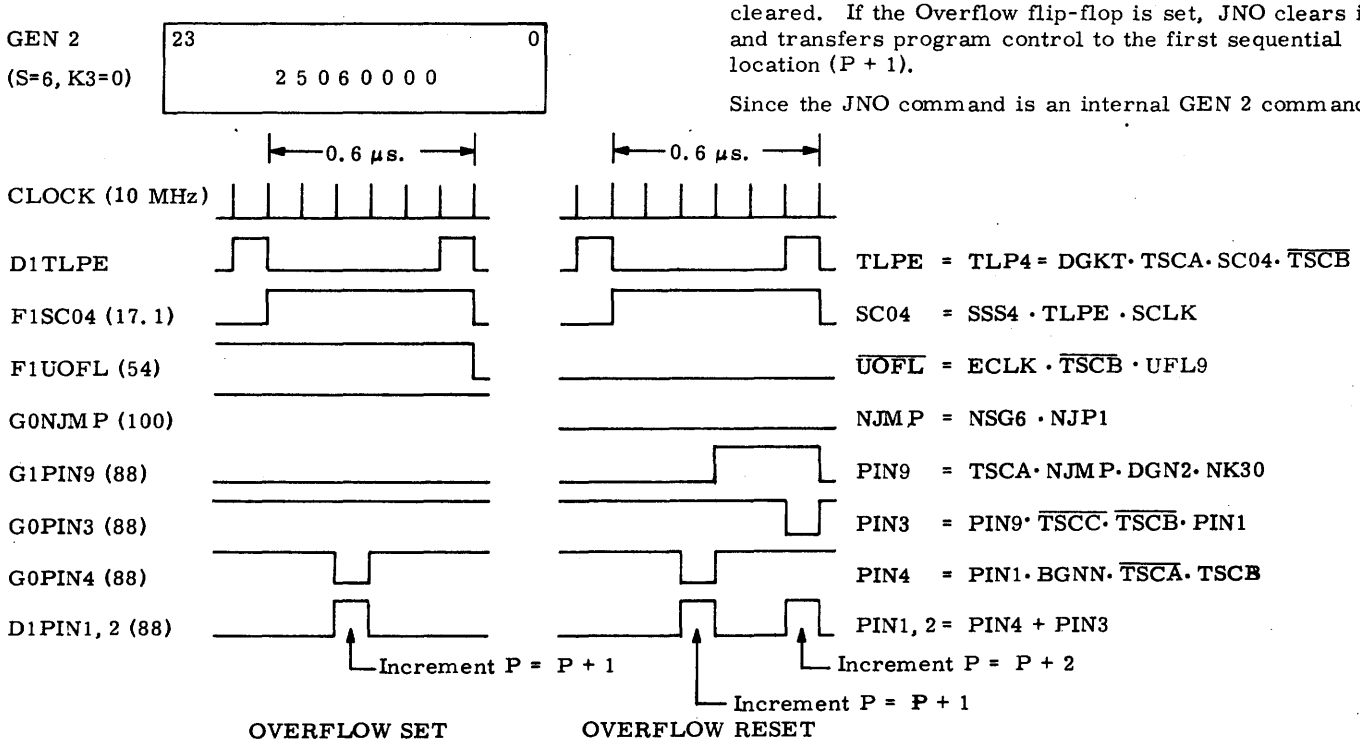


JNE Timing Diagram

JNO-JUMP IF NO OVERFLOW

JNO transfers program control to the second sequential location (P + 2) if the Overflow flip-flop (F1UOFL) is cleared. If the Overflow flip-flop is set, JNO clears it and transfers program control to the first sequential location (P + 1).

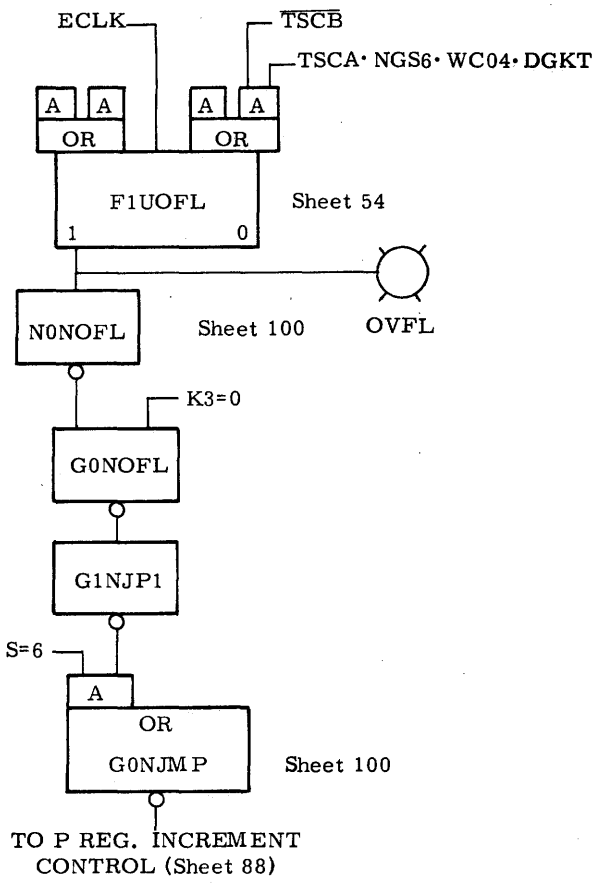
Since the JNO command is an internal GEN 2 command,



JNO Timing

Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the JNO command.

The P Register is always incremented during Time 2 ($\overline{TSCA}$ · TSCB) of the JNO command. If the Overflow flip-flop (F1UOFL) is in the reset state, the P Register is again incremented at Time 5 (TSCA · $\overline{TSCB}$ · $\overline{TSCC}$). If the Overflow flip-flop is set, the JNO command resets it during Time 5 (G1UFL9).



JNO Block Diagram

JNP-JUMP IF NO PARITY ERROR

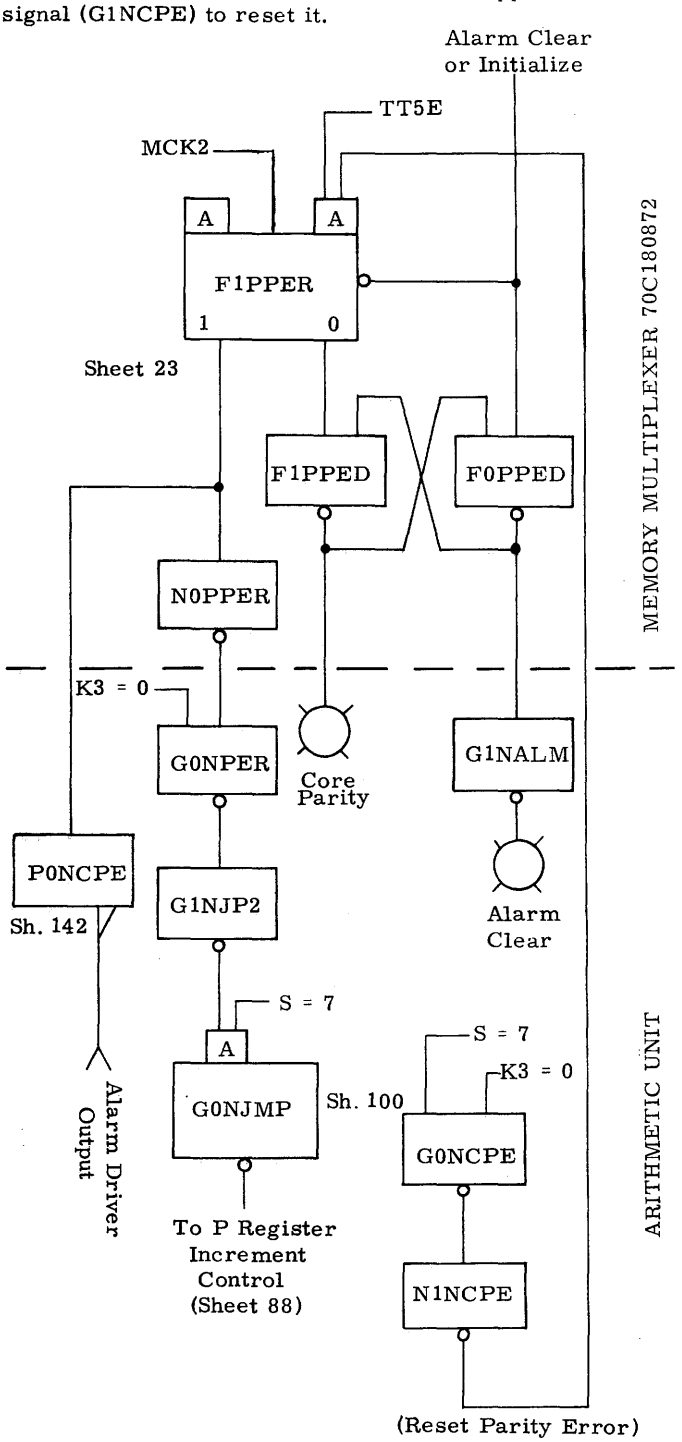
GEN 2
(S=7, K3=0)

23	0
2 5 0 7 0 0 0 0	

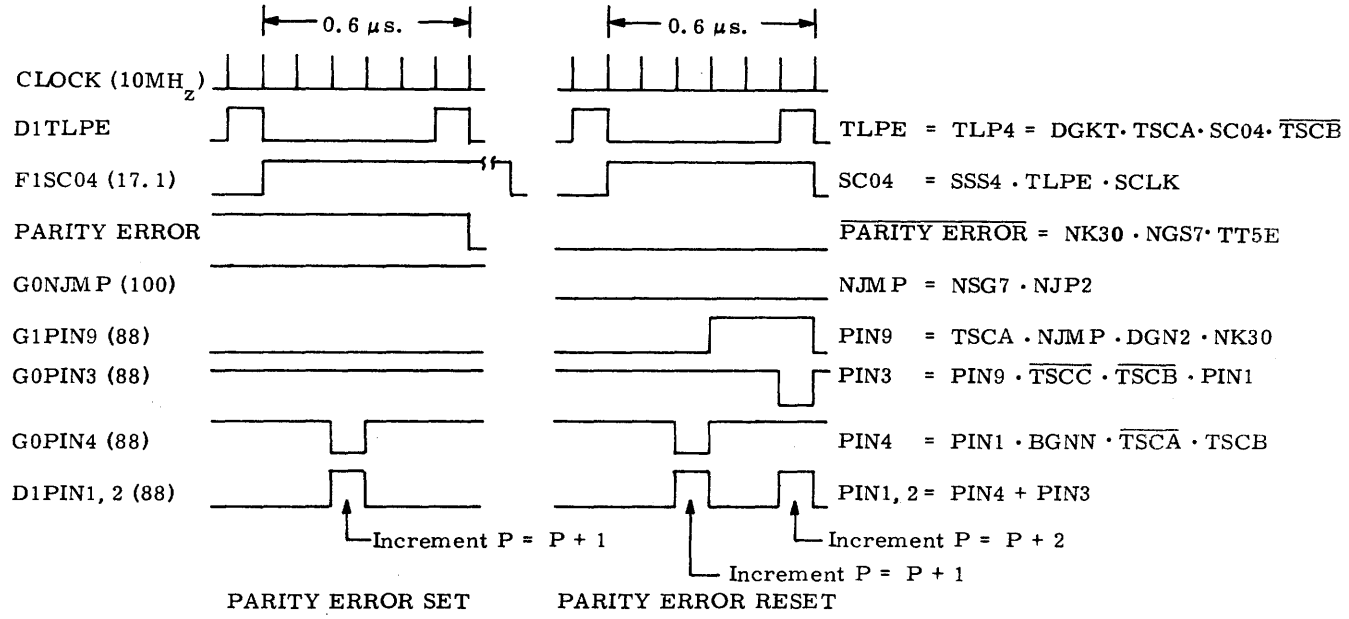
JNP transfers program control to the second sequential location (P + 2) if a core memory parity error does not exist (i.e. the Core Parity Error flip-flop is clear). If a core parity error exists, JNP transfers program control to the next sequential location (P + 1) and clears the Core Parity Error flip-flop.

Since the JNP command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the JNP command.

The P Register is always incremented during Time 2 ($\overline{TSCA}$· TSCB) of the JNP command. If the Parity Error flip-flop in the Core Memory Module is reset, the P Register is incremented for the second time (P + 2) at Time 5 (TSCA · $\overline{TSCB}$ · $\overline{TSCC}$). If the Parity Error flip-flop is set, the JNP command applies a signal (G1NCPE) to reset it.



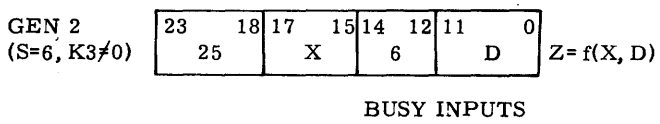
JNP Block Diagram



JNP Timing

JNR-JUMP IF DEVICE D NOT READY

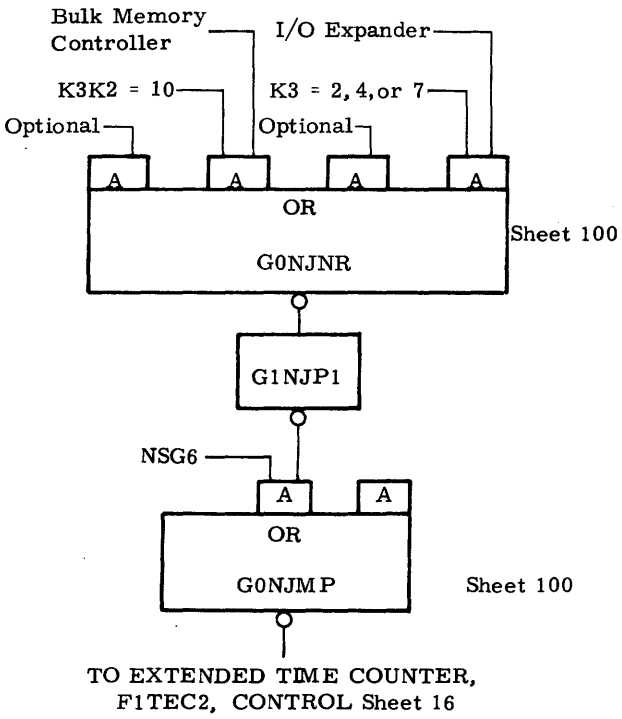
JNR transfers program control to the second sequential location (P + 2) if the addressed device (Z) is not ready. If the addressed device is "ready", JNR transfers program control to the first sequential location (P + 1).

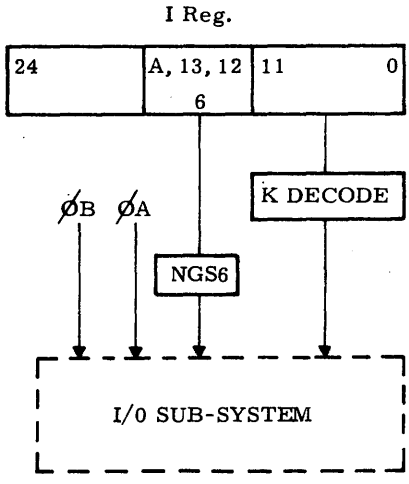


BUSY INPUTS

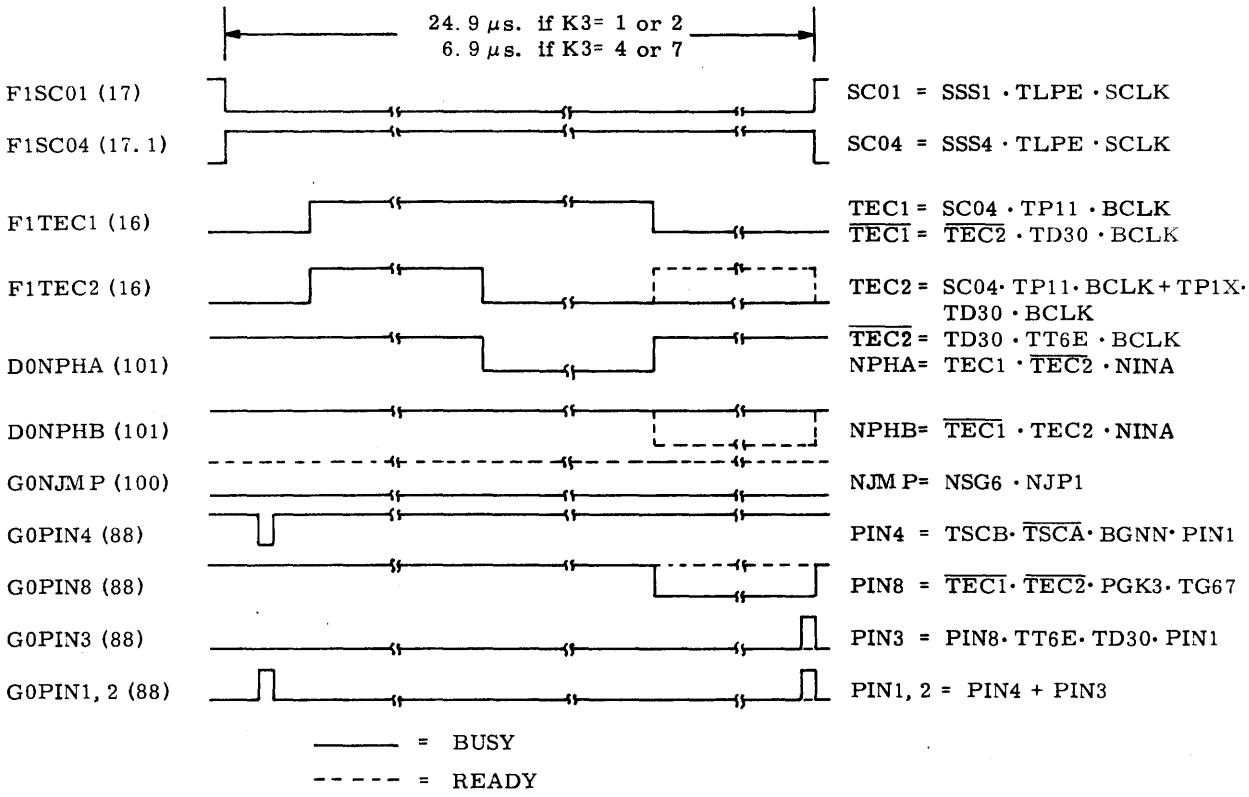
NOTE

The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.





JNR Block Diagram

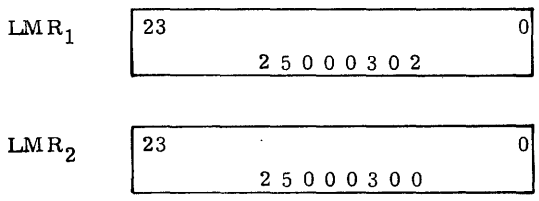


JNR Timing Diagram

The duration of Sequence Control State 4 is 24.9 microseconds if K3 (bits 11-9 of the NJR command) is equal to 1 or 2 or the duration is 6.9 microseconds if K3 is equal to 4 or 7. Refer to the appropriate basic timing diagram contained in Fig. GN2.8 or GN2.9.

The S bits of the JNR command are decoded from $I_{A,13,12}$ to enable NGS6. This signal, along with the decoded K3-0 signals and Phase A and Phase B, are applied to the I/O Sub-system. The P Register is incremented (G0PIN4) at Time 2 of State 4. The ready signal from the addressed device is applied through G0NJNR. If the addressed device is busy (not ready), G0NJNR is enabled, enabling G0NJMP. When enabled, G0NJMP inhibits setting of F1TEC2 when the Delay Time Counter is equal to 30_8. Inhibiting F1TEC2, inhibits generation of Phase B (D0NPHB) and enables G1PIN8. G1PIN8 enables G1PIN3 causing the P Register to be incremented a second time (P + 2). If, however, the device is ready (not busy), G0NJMP is inhibited. Inhibiting G0NJMP enables setting F1TEC2 when the Delay Time Counter is equal to 30_8. Setting F1TEC2 enables generation of Phase B and inhibits incrementation of the P Register for the second time.

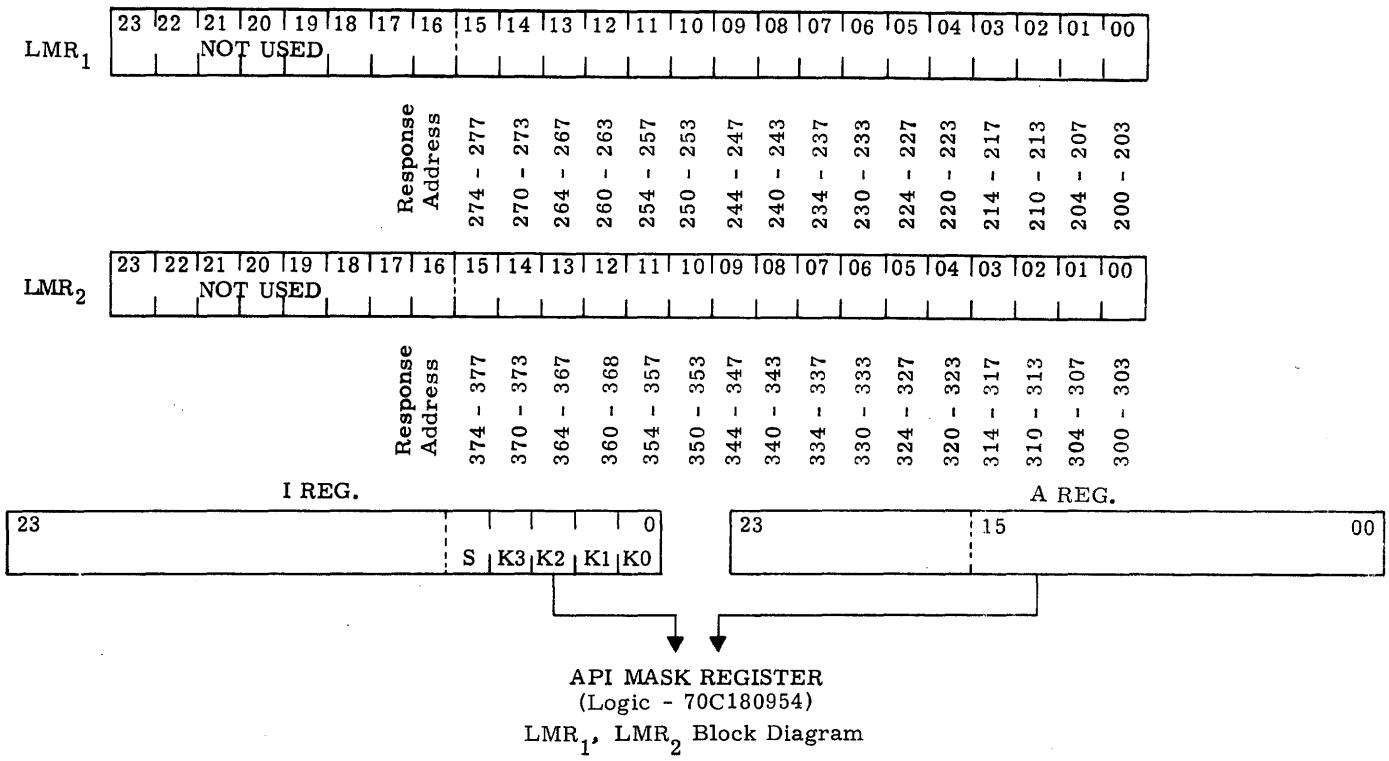
LMR_1, LMR_2 -LOAD MASK REGISTER



LMR_1 transfers the contents of the A Register bits 15-0 to the first 16 bits of the API Mask Register controlling non-inhibitable interrupts. LMR_2 transfer the contents of the A Register bits 15-0 to the second 16 bits of the API Mask Register controlling inhibitable interrupts.

Since LMR and LMR_2 are considered internal GEN 2 commands, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.5 apply to the LMR_1 and LMR_2 command.

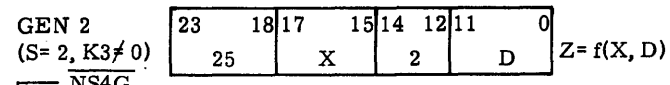
Each bit of the A Register will specify if a specific group

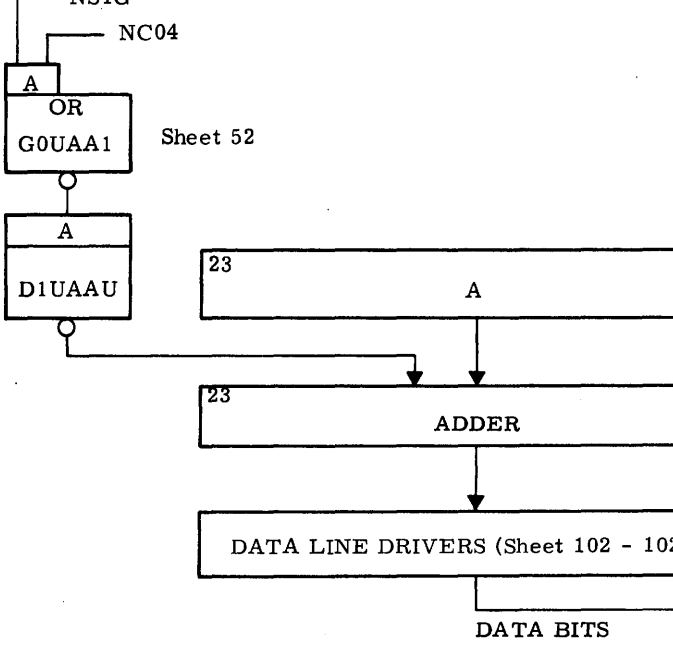


LMR_1, LMR_2 Block Diagram

of 4 interrupts are inhibited. If the A Register bit is a "one", the 4 associated interrupts are inhibited. If the A Register bit is a "zero", the 4 corresponding interrupts are allowed. The A Register bits and the interrupts affected are shown above.

OPR-OPERATE





OPR Block Diagram

OPR initiates one operation of device Z.

> NOTE
>
> The specific function performed by this instruction may vary with each I/0 Device. Refer to the description of the device or I/0 subsystem for operating details.

Refer to the command description for the OUT command.

for details of operation of the OPR command within the Arithmetic Unit. The description and timing diagram of the OUT command applies to the OPR command except that the A Register is gated to the Adder Unit by enabling G0UAA1 by $\overline{\text{G0NS4G}}$ · D1UC04 for the OPR command.

OUT-OUTPUT TO DEVICE D

GEN 2
(S=4, K3≠0)

23 18	17 15	14 12	11 0
25	X	4	D

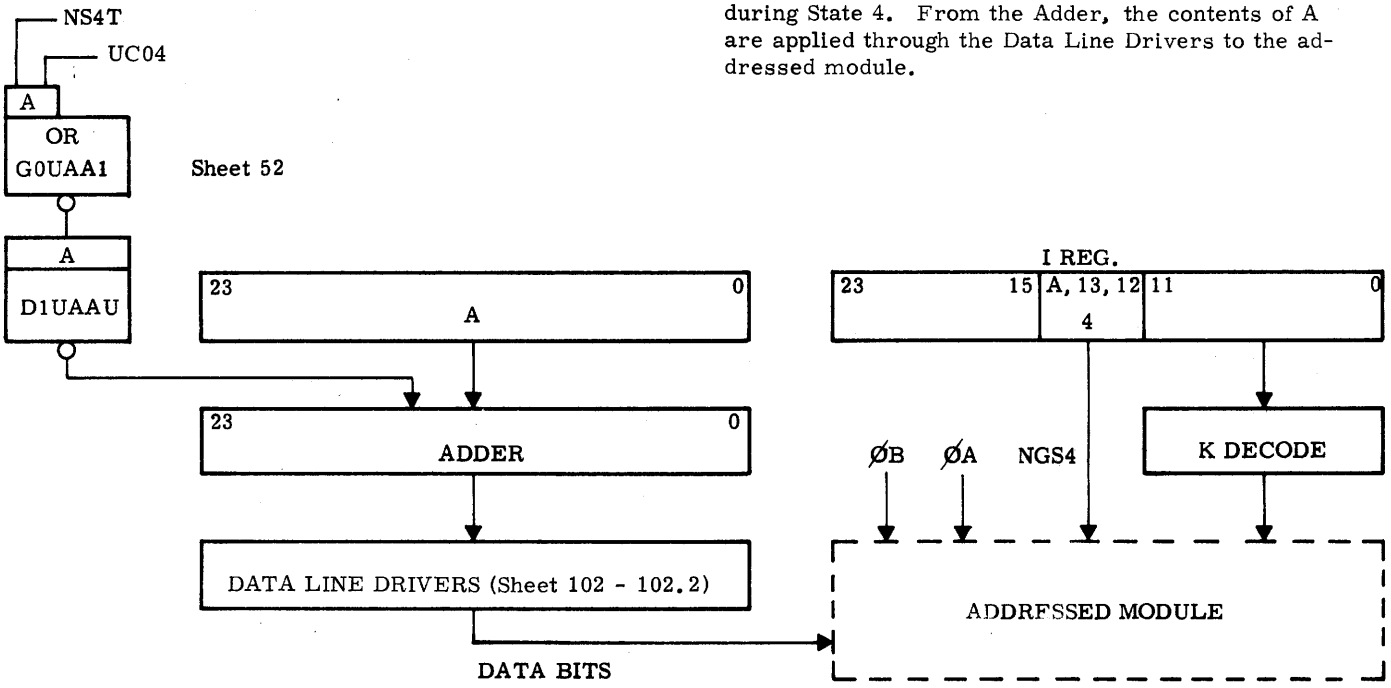
Z= f(X, D)

OUT transfers data from the A Register to the addressed device and initiates one operation of the device OUT.

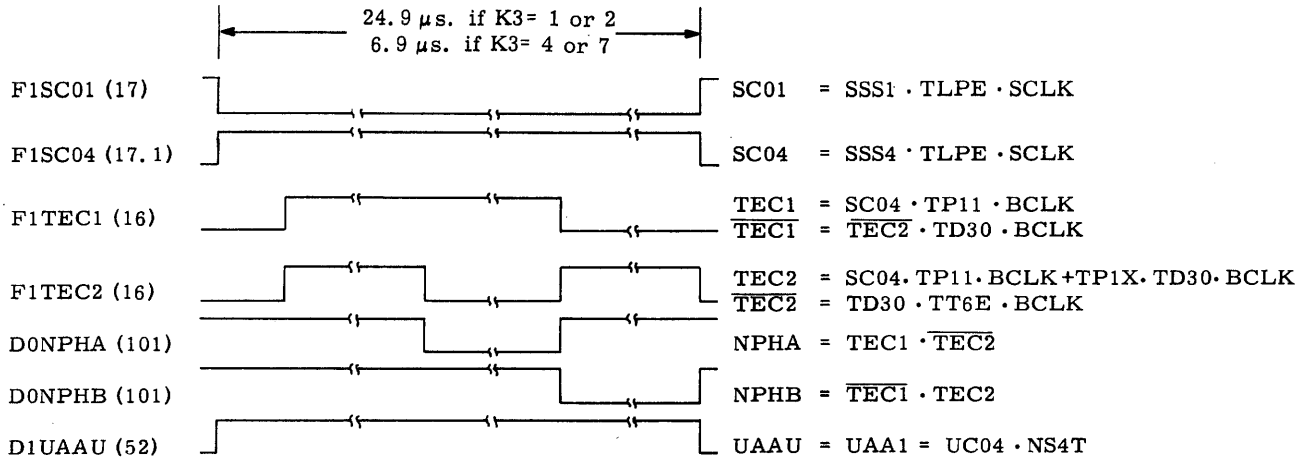
> NOTE
>
> The specific function performed by this instruction may vary with each I/O Device. Refer to the description of the device or I/O subsystem for operating details.

Since the OUT command is an external GEN 2 command, the duration of Sequence Control State 4 is 24.9 microseconds if the K3 bits are equal to 1 or 2 or the duration is 6.9 microseconds if the K3 bits are equal to 4 or 7. The basic timing diagram contained in Fig. GN2.8 or GN2.9 applies to the OUT command.

The S and K bits of the command are decoded from the I Register and applied to the addressed module along with the Phase A and Phase B timing signals. The contents of the A Register are applied to the Adder unit during State 4. From the Adder, the contents of A are applied through the Data Line Drivers to the addressed module.



OUT Block Diagram



OUT Timing Diagram

PAI-PERMIT AUTOMATIC INTERRUPT

GEN 2
(S=2, K3=0)

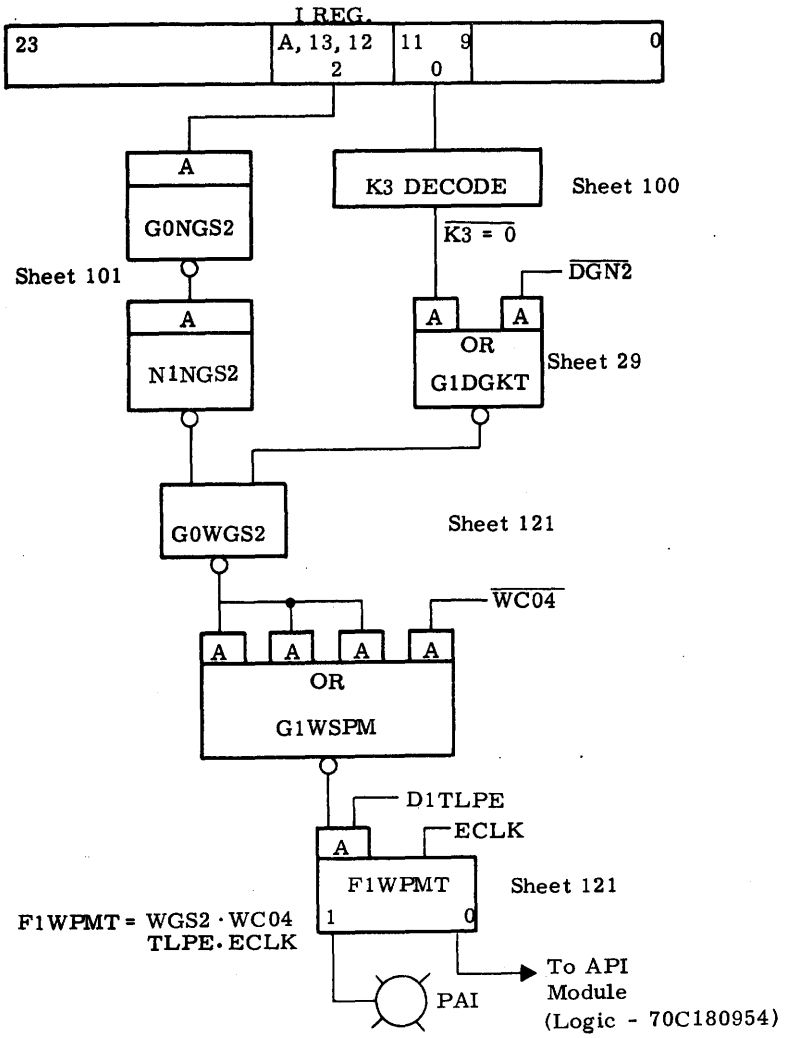
23		0
	2 5 0 2 0 0 0 0	

PAI sets the Permit Automatic Interrupt flip-flop (F1WPMT) to allow any program interrupts following the next interruptable command. Setting the Permit Automatic Interrupt flip-flop, clears out an inhibit of all interrupts from a previously executed IAI_2 command permitting interrupts following the next interruptable command.

Non-inhibitable interrupts may be serviced immediately following PAI. Inhibitable interrupts must wait until the next interruptable command.

Since the PAI command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the PAI command.

The S bits of the PAI command are decoded from $I_{A,13,12}$ to enable the NGS2 gates. The K3 bits of the PAI command are decoded from I_{11-9} and enable the NK30 gates. At the Clock of Last Pulse of State 4, the Permit Automatic Interrupt flip-flop (F1WPMT) is set. When set, F1WPMT applies a signal to the Automatic Program Interrupt module to enable inhibitable type interrupts.



PAI Block Diagram

RALM-RESET PROGRAMMABLE ALARM

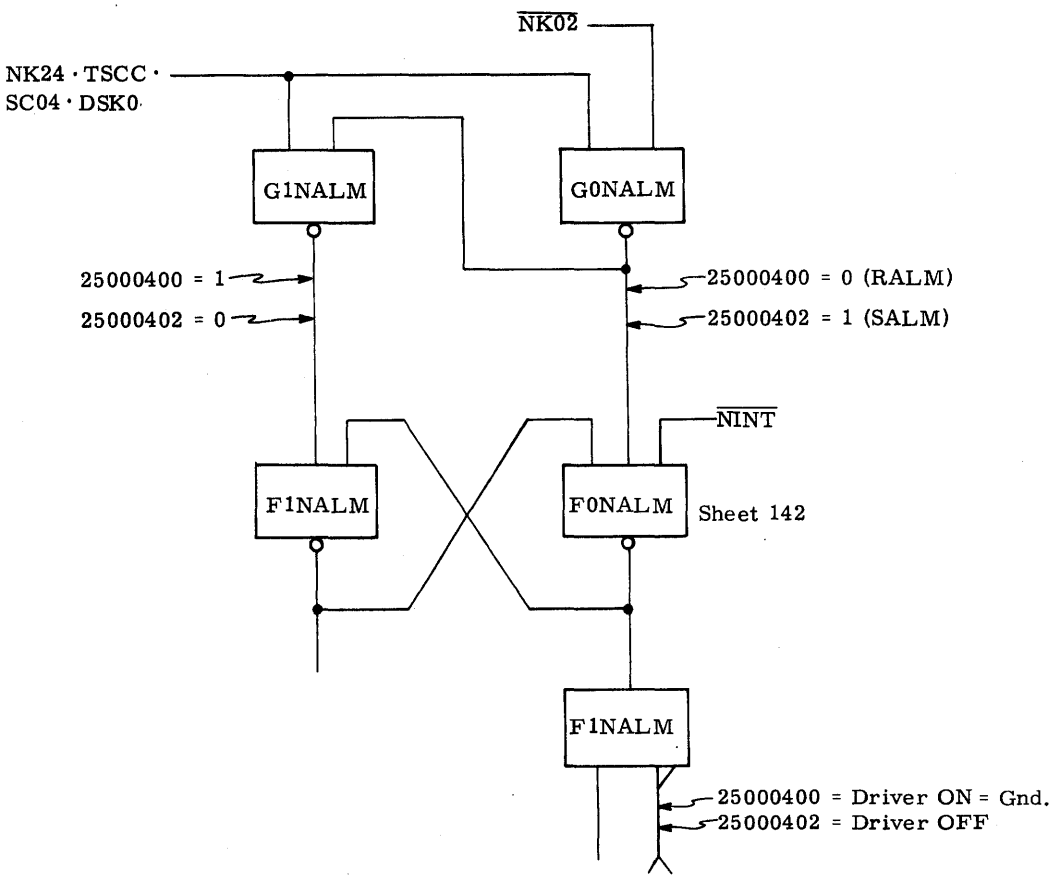
GEN 2
(S = 0, K3 = 0)

2 5 0 0 0 4 0 0

The programmable alarm internal GEN 2 commands provide the capability of enabling a relay driver, under program control, for optional use by the system. The relay driver is turned on by executing 25000400 (RALM) and turned off by executing SALM.

Since the K3 bits of the RALM command are equal to 0, it is executed in 0.6 microseconds. The basic timing diagram contained in Fig. GN2.7 applies to this command. The logic associated with RALM is shown on sheet 142 of the logic.

When the SALM command is executed, F1NALM is set disabling the relay driver P1NALM. F1NALM remains in the set state until the RALM command is executed. When executed, the RALM command resets F1NALM enabling the relay driver.



RALM and SALM Block Diagram

RAPG-RESET ADJUSTABLE PULSE GENERATOR

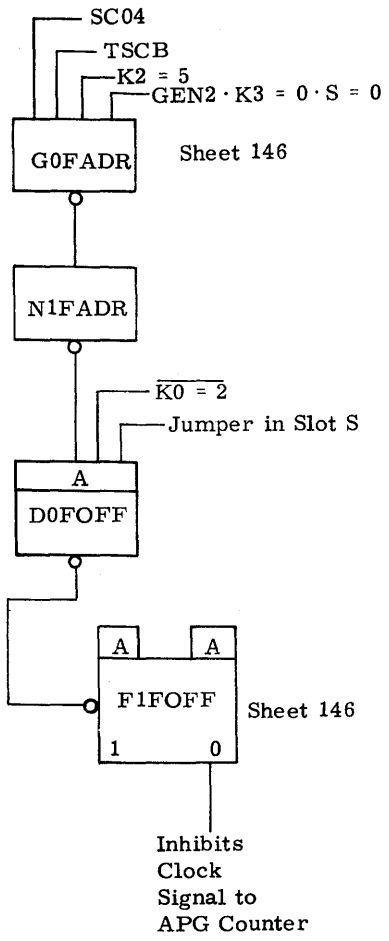
GEN 2
(S = 0, K3 = 0)

2	5	0	0	0	5	0	0

RAPG inhibits the generation of square wave pulses from the optional Adjustable Pulse generator. The adjustable pulse generator, as described in the OPT section, provides logic level square wave pulses for application to external process control stations via the optional Pulse Source Initiator or provides a high resolution clock API input. The pulses may be started (SAPG) and stopped (RAPG) under program control when the jumper pin on the SPGA5 board in slot B16FK is in the S position. When the jumper pin is in the I position, the pulse output is continuous.

Since the RAPG command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the RAPG command.

The S and K bits of the RAPG command are decoded from $I_{A,13-0}$ to enable G1DSK0 (sheet 29) and N1NK25 (sheet 99.1). At TSCB time of State 4, D0FOFF (sheet 146) is enabled provided the jumper pin of the SCGA5 board is in the S position. Enabling D0FOFF sets F1FOFF inhibiting the further generation of pulse outputs from the adjustable pulse generator.



RAPG Block Diagram

RCS-READ CONSOLE SWITCHES

GEN 2

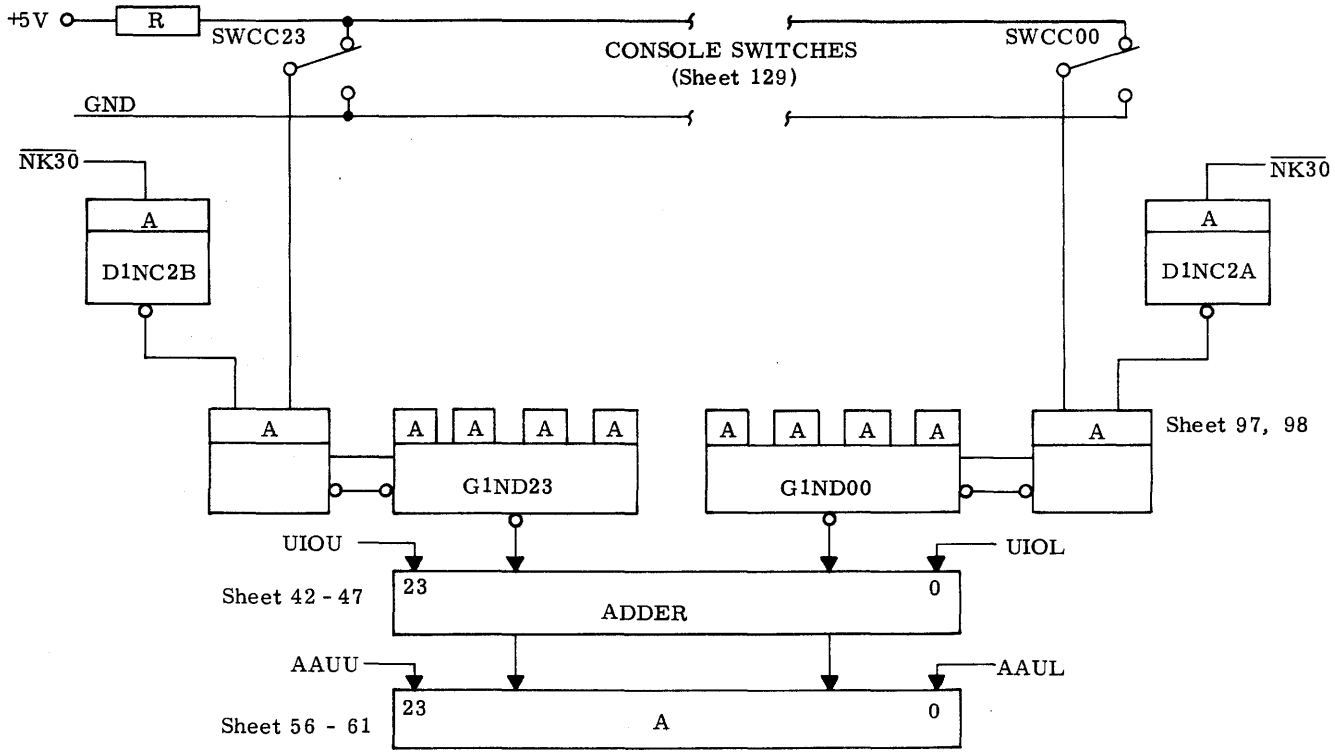
(S = 5, K3 = 0)

23							0
2	5	0	5	0	0	0	0

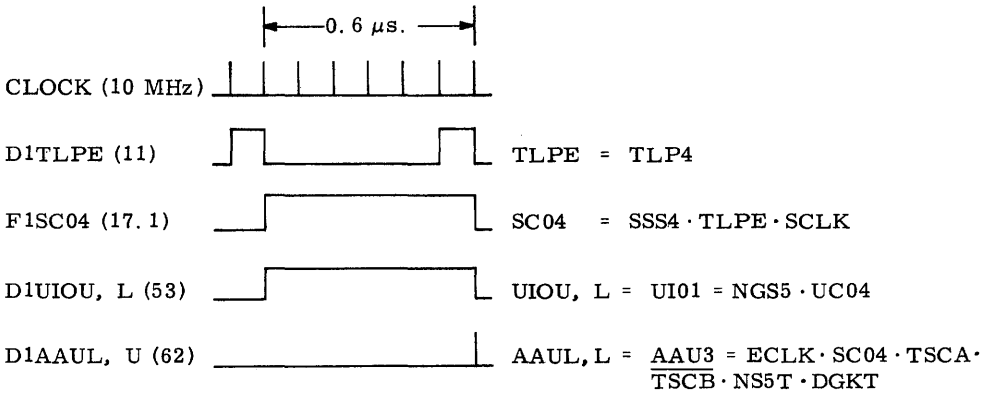
RCS places the contents of the console toggle switches in the A Register. A console toggle switch in the down position generates a "one"; a toggle switch in the up position generates a "zero".

Since the RCS command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the RCS command.

The S bits of the RCS command are decoded from $I_{A,13,12}$ to enable the NGS5 gates. The K3 bits of the RCS command are decoded from I_{11-9} and enable the NK30 gates. The status of the Console Switches (SWCC23 - SWCC00) are applied through the I/O Input gates. From the I/O Input gates, the status of the Console Switches are gated to the Parallel Adder (D1UIOU, L) and at the Clock of Time 5 the contents of the Parallel Adder are gated to the A Register (D1AAUU, L).



RCS Block Diagram



RCS Timing Diagram

SALM-SET PROGRAMMABLE ALARM

GEN 2

(S = 0, K3 = 0)

2	5	0	0	0	4	0	2

The programmable alarm internal GEN 2 commands provide the capability of enabling a relay driver, under program control, for optional use by the system. The relay driver is turned on by executing 25000400 (RALM) and turned off by executing 25000402 (SALM).

Since the K3 bits of the SALM command are equal to 0, it is executed in 0.6 microseconds. The basic timing diagram contained in Fig. GN2.7 applies to this command. The logic associated with SALM is shown on sheet 142 of the logic.

When the SALM command is executed, F1NALM is set disabling the relay driver, P1NALM. F1NALM remains in the set state until the RALM command is executed. When executed, the RALM command resets F1NALM enabling the relay driver.

SAPG-SET ADJUSTABLE PULSE GENERATOR

GEN 2

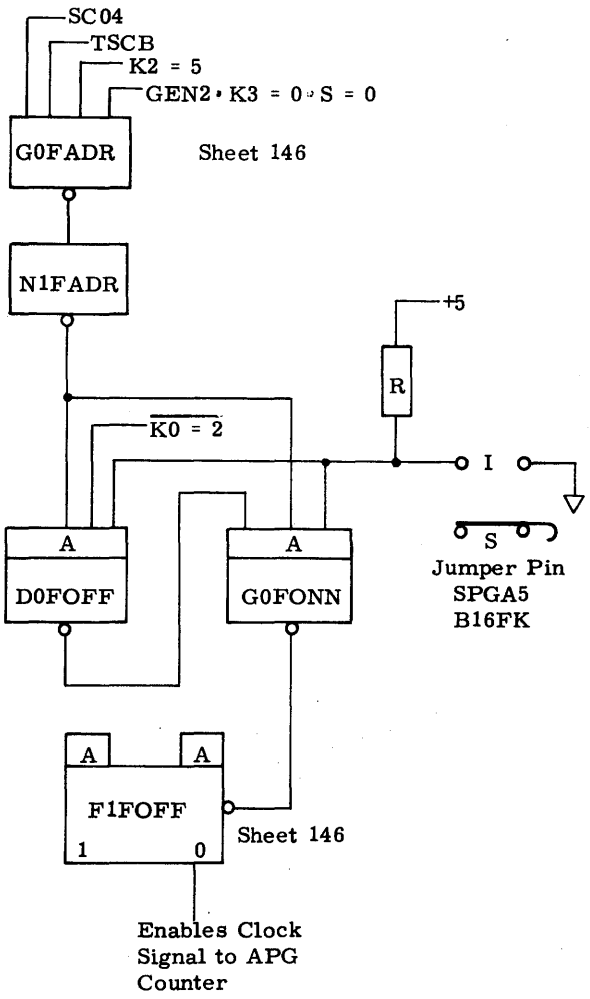
(S = 0, K3 = 0)

2	5	0	0	0	5	0	2

SAPG enables the generation of square wave pulses from the optional Adjustable Pulse Generator. The adjustable pulse generator, as described in the OPT section, provides logic level square wave pulses for application to external process control stations via the optional Pulse Source Initiator or provides a high resolution clock API input. The pulses may be started (SAPG) and stopped (RAPG) under program control when the jumper pin on the SPGA5 board in slot B16FK is in the S position. When the jumper pin is in the I position, the pulse output is continuous.

Since the SAPG command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the SAPG command.

The S and K bits of the SAPG command are decoded from $I_{A,\,13-0}$ to enable G1DSK0 (sheet 29), N0NK02 (sheet 99), and N1NK25 (sheet 99.1). At TSCB time of State 4, N1FADR (sheet 146) is enabled. Since N0NK02 is enabled, D0FOFF is disabled and G0FONN is enabled. Enabling G0FONN resets F1FOFF enabling the pulse outputs from the adjustable pulse generator.



SAPG Block Diagram

SEL-SELECT DEVICE D

GEN 2

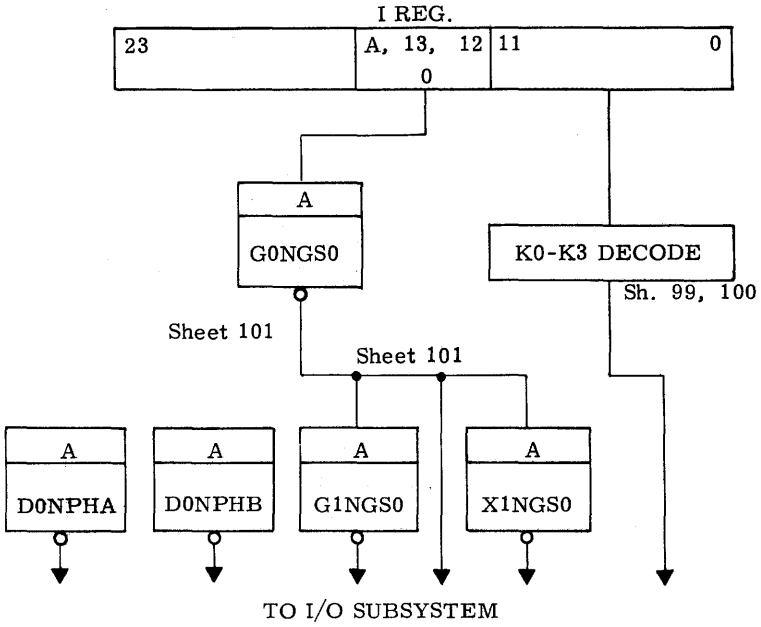
(S = 0, K3 ≠ 0)

23 18	17 15	14 12	11 0
25	X	0	D

Z = f(X, D)

SEL enables the phase A, phase B, S = 0, and decoded K bits to the addressed I/O subsystem. The function performed within individual subsystems differs. For example, SEL is used to enter and exit the "test mode" of the Process Digital Controller.

Since SEL is an external GEN 2 command (K3 ≠ 0), the duration of Sequence Control State 4 is 6.9 microseconds if K3 is equal to 4 or 7 or 24.9 microseconds if K3 is equal to 1 or 2. The timing diagram and logic equations contained in Fig. GN2.8 or Fig. GN2.9 apply to the SEL command. The S bits of the SEL command are decoded from $I_{A, 13, 12}$ to enable the NGS0 gates. I_{11-0} is decoded to enable the corresponding K decode gates. These signals along with Phase A and Phase B are applied to the I/O Subsystem.



SEL Block Diagram

SSA-SET STALL ALARM

GEN 2

(S = 1, K3 = 0)

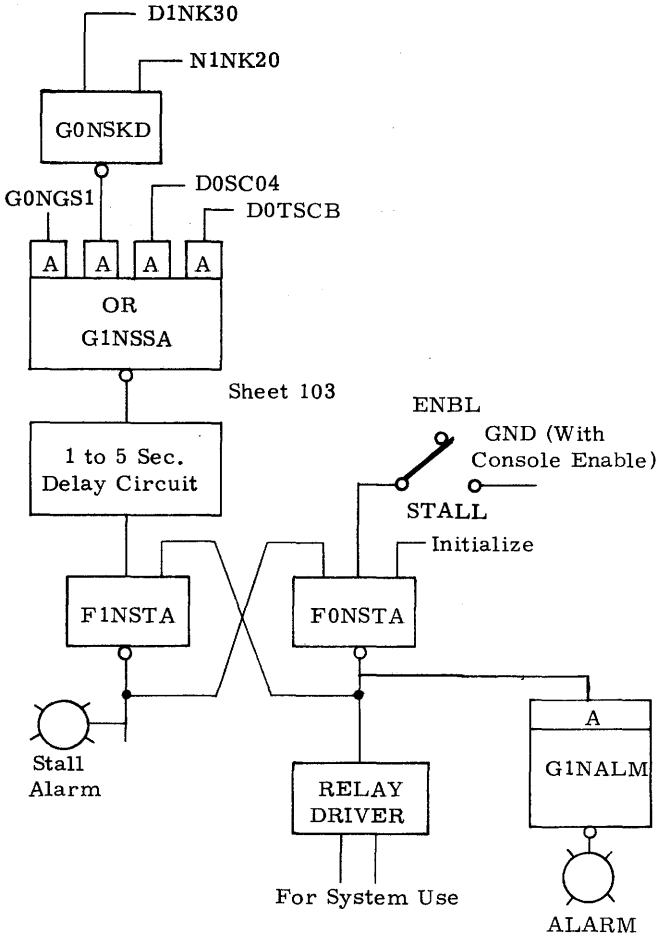
23 0
2 5 0 1 0 0 0 0

SSA resets a manually adjustable (1 to 5 second, which is normally set to 5 seconds) timer in the Stall Alarm circuitry. If the timer is allowed to "time out", because the program does not execute another SSA command to again reset the timer, an output signal lights the STALL ALARM and ALARM indicators on the console and enables a relay driver for systems use.

The Stall Alarm, as described in the DESC section, is used to detect a malfunction in the computer program or system that causes a hang-up or stall condition in program sequencing. The stall alarm circuitry is inhibited in the manual mode of operation or when the Stall Enable switch is in the lockout position. Refer to the Description (DESC) section for a detailed discussion of the Stall Alarm circuitry.

Since the SSA command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the SSA command.

The S and K bits of the SSA command are decoded enabling G1NSAA at TSCB time of State 4. Enabling G1NSSA initiates a new time period in the manually variable 1 to 5 second timer. If the timer is allowed to "time out", F1NSTA is set providing the error signal. Once set, F1NSTA may be cleared by initializing the system or by placing the STALL ENBL switch in the up (lockout) position when the console enabled.



SSA Block Diagram

STMF-SET TRAPPING MODE

GEN 2

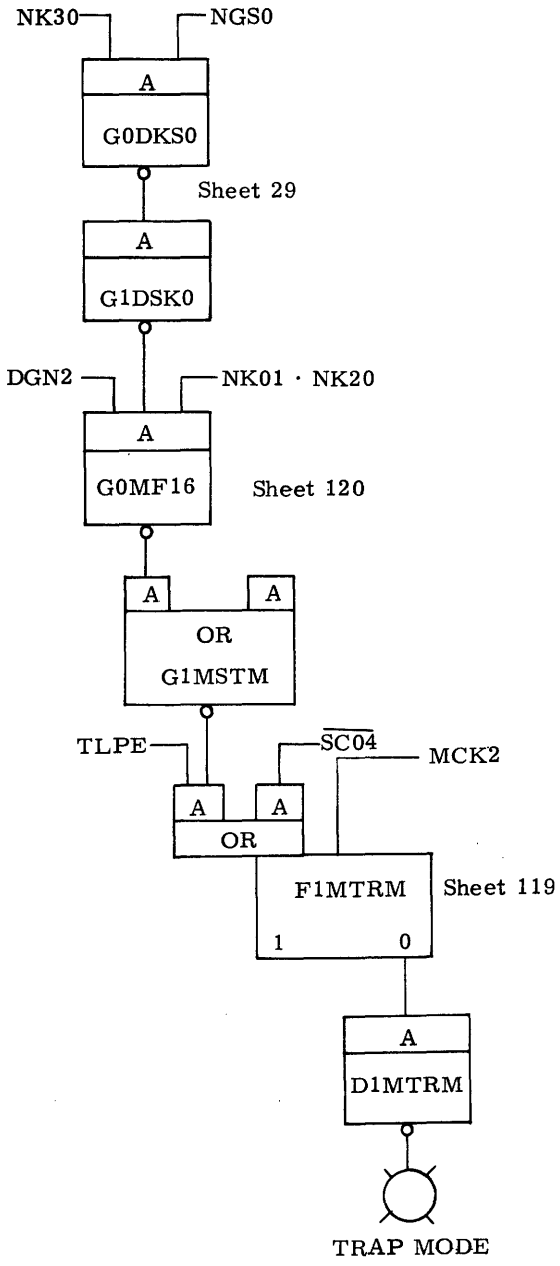
(S=0, K3=0)

23	0
2 5 0 0 0 0 0 1	

STMF sets the Trapping Mode flip-flop (F1MRTM) to enable the protection functions of the optional Memory Protect (Quadritect) capability. Refer to the Memory Protect discussion contained in the Options Description portion of this section for details.

Since the STMF command is an internal GEN 2 command, Sequence Control State 4 has a duration of 0.6 microseconds. The timing diagram and logic equations shown in Fig. GN2.7 apply to the STM command.

The microcoding of the STMF command is decoded from $I_{A,\ 13-0}$ to enable NGS0, NK30, and NK20. These signals enable setting of the Trapping Mode flip-flop F1MTRM at the Clock of Last Pulse of State 4.



STMF Block Diagram

GEN 3 COMMANDS

GEN 3 commands are microcoded instructions that provide serial shifting of the A and Q Registers, both left and right, and optionally affect the Overflow flip-flop. Actual shifting of the Q Register (memory cell 10_8) occurs in the B Register.

GEN 3 commands are identified by the OP code 45_8 (bits 23-18). Fig. GN3.1 illustrates the GEN 3 format and explains the microcoded bit control.

GEN 3 commands "fetched" during Sequence Control State 1 are executed during Sequence Control States 4 and 5. The duration of State 4 is extended and controlled by the J Counter. The J Counter is preset from the shift constant (bits 4 - 0) of the GEN 3 command and then incremented at each shift of the B and/or A Register until it is equal to 37_8. When J is equal to 37_8, Last Pulse of State 4 is enabled. In this manner, the execution time of GEN 3 commands varies from 5.0 to 8.1 microseconds depending upon the value of the shift constant. GEN 3 commands may be interrupted following execution.

Due to the microcoding of GEN 3 commands, there are hundreds of unique operations that may be performed by GEN 3 commands. At this time, 8 of these operations are considered to be used frequently and mnemonics have been assigned. These 8 commands and the microcoding are listed in Table GN3.1. Following the basic timing and sequence of events for GEN 3 commands, each of the 8 commands having assigned mnemonics is described.

BASIC TIMING

Fig. GN3.2 illustrates the basic sequencing of GEN 3 commands and Fig. GN3.3 illustrates the affect of microcoding on the operation of GEN 3 commands. Fig. GN3.4 contains a timing diagram with logic equations that applies to all GEN 3 commands. Refer to these aids during the following discussion.

Like all other commands, GEN 3 commands are "fetched" from memory during Sequence Control State 1. Non-indexed GEN 3 commands are then executed in Sequence State 4. During State 4, core memory cell 10_8 is addressed (G0MX03). As previously described, memory cell 10_8 is the Q Register. The contents of memory cell 10_8 are gated to the B Register. Also, during this time, the J Counter is cleared (D0JJE0) and the complement of I_{4-0} (K) is gated to J.

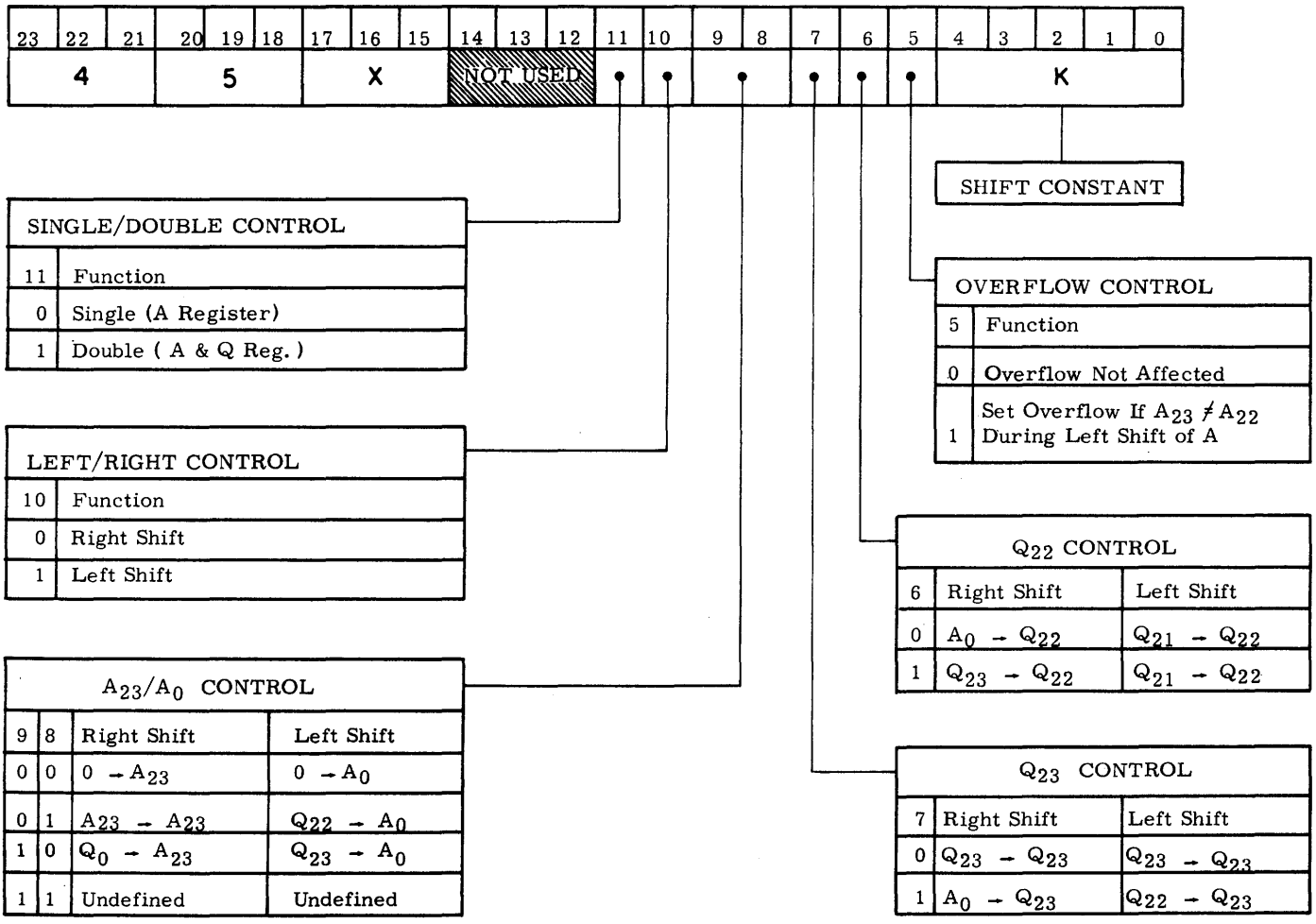


Fig. GN3.1 Action Control Bits of GEN 3 Commands

MNEMONIC	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLA	0	0	0	1	1	0	1	0	0	1	←		K		→
DLL	0	0	0	1	1	1	0	1	0	0	←		K		→
DRA	0	0	0	1	0	0	1	0	0	0	←		K		→
DRC	0	0	0	1	0	1	0	1	1	0	←		K		→
DRL	0	0	0	1	0	0	0	1	1	0	←		K		→
MAQ	0	0	0	1	0	0	0	1	1	0	1	1	0	0	0
SLA	0	0	0	0	1	0	0	0	0	1	←		K		→
SLL	0	0	0	0	1	0	0	0	0	0	←		K		→

Table GN3.1 Microcoding of Defined GEN 3 Commands

During this portion of State 4, the Sequence Time Counter (F1TSCA - C) is incremented in the same manner as full operand commands. That is, the Sequence Time Counter is incremented and held in Time 2 ($\overline{SCA} \cdot SCB \cdot SCC$) until Data Ready (MXD1MDR4) is received from memory. The Sequence Time Counter is then incremented until Time 5 ($SCA \cdot \overline{SCB}\ \overline{SCC}$), where it is held until Memory Release (MXD1MRLS) is received from memory. After Memory Release is received, the Sequence Time Counter is incremented to Time 6 (F1TT6E).

During Time 6, the B and/or A Registers are shifted according to the microcoding of the GEN 3 command. These shifts occur at each clock pulse of Time 6. At each shift, the J Counter is incremented. When the J Counter is equal to 37_8, the number of shifts specified by the K bits (4-0) of the GEN 3 command have occurred, further shifting is inhibited and Last Pulse is enabled to end State 4.

Following State 4, State 5 is entered. During State 5, memory cell 10_8 is again addressed (G0MX03) and the contents of B are stored back in the Q Register (cell 10_8). The basic timing of State 5 is the same as for all full operand store commands. State 5 completes the execution of the GEN 3 command and State 1 is entered to "fetch" the next instruction.

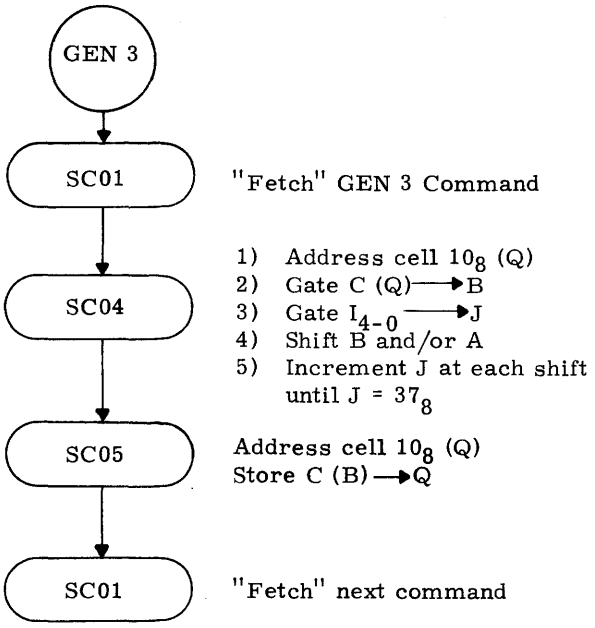


Fig. GN3.2 Basic GEN 3 Sequencing

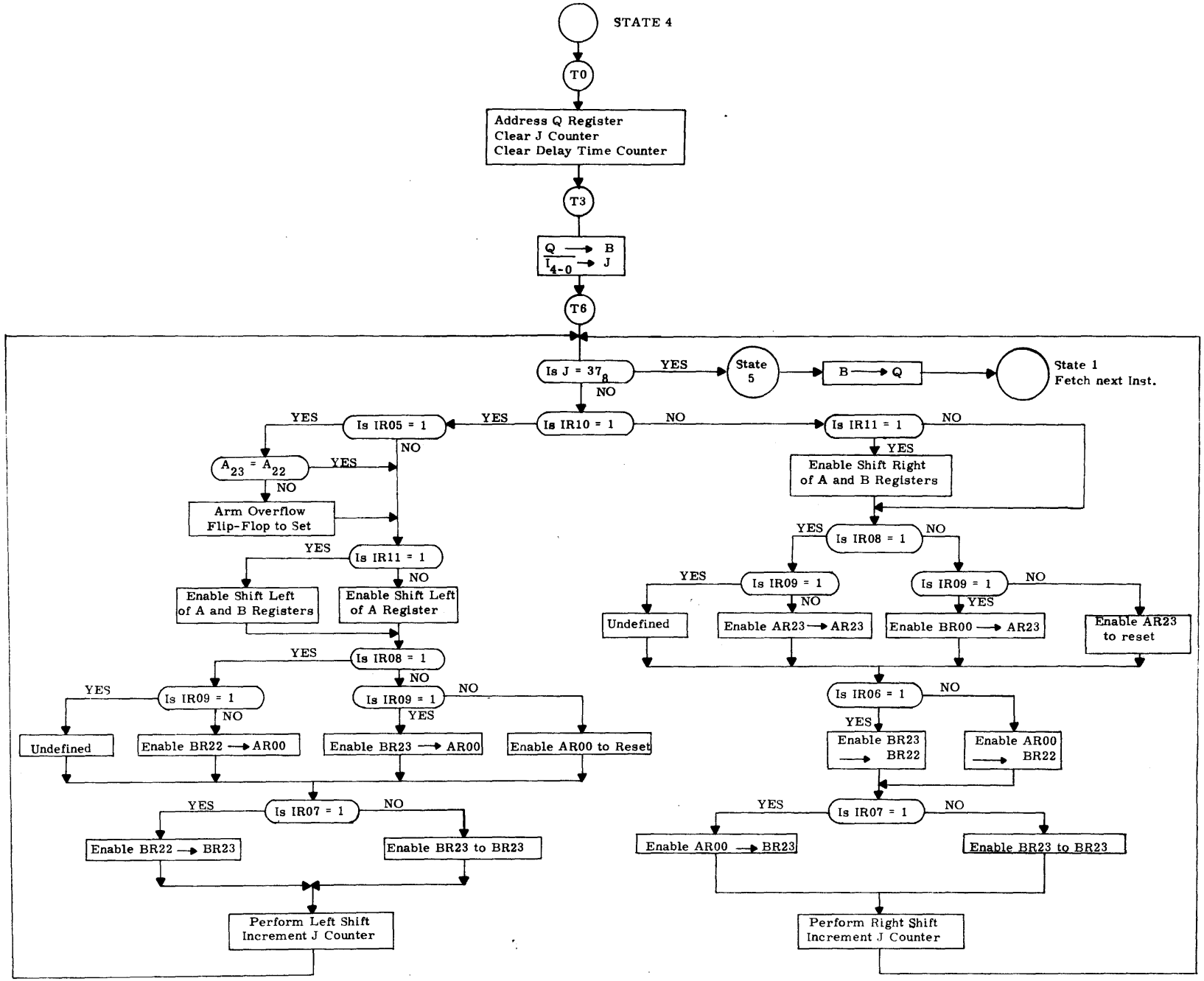


Fig. GN3.3 GEN 3 Flowchart

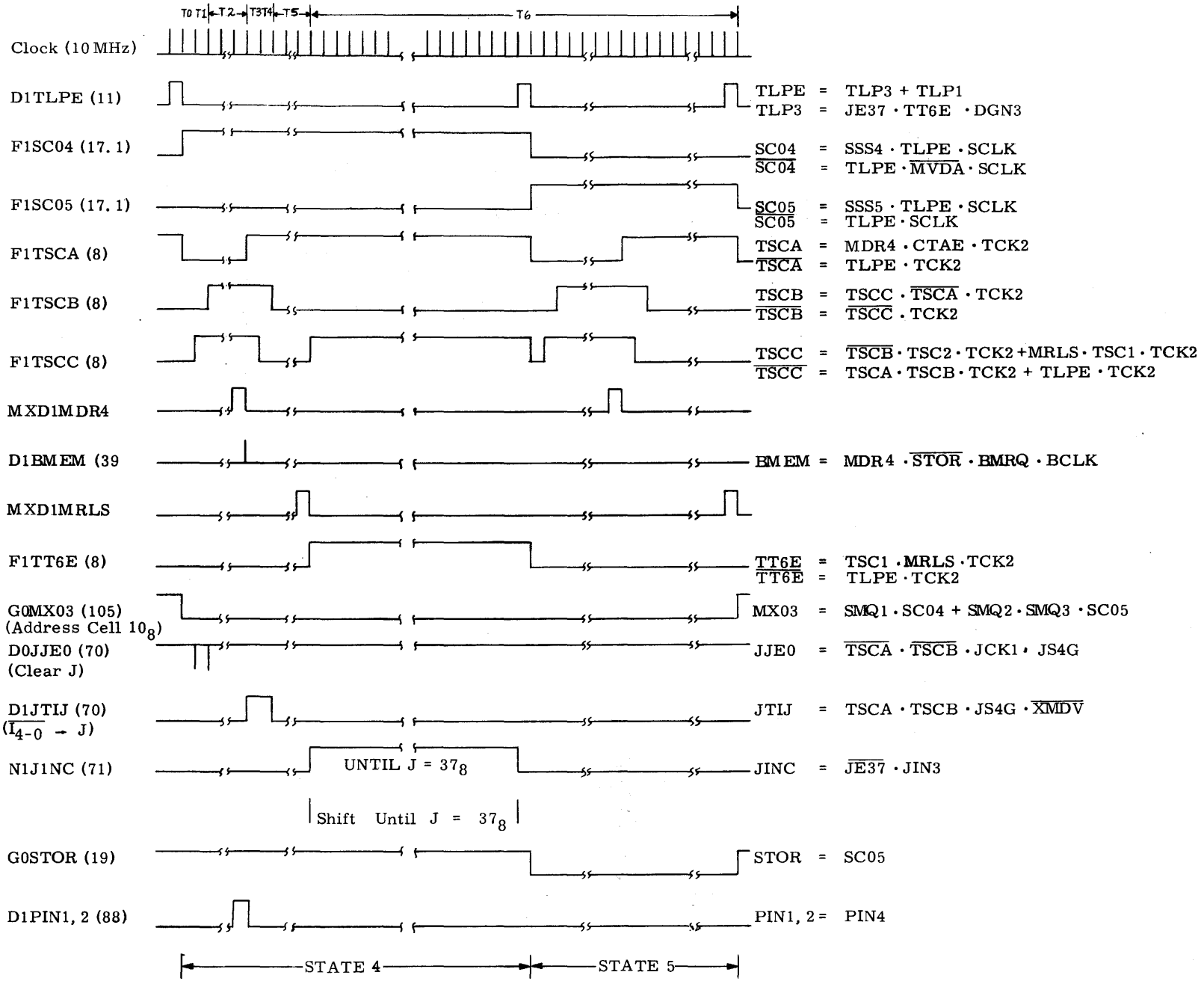


Fig. GN3.4 GEN 3 Basic Timing

DLA-(SHIFT) DOUBLE LEFT ARITHMETIC

GEN 3

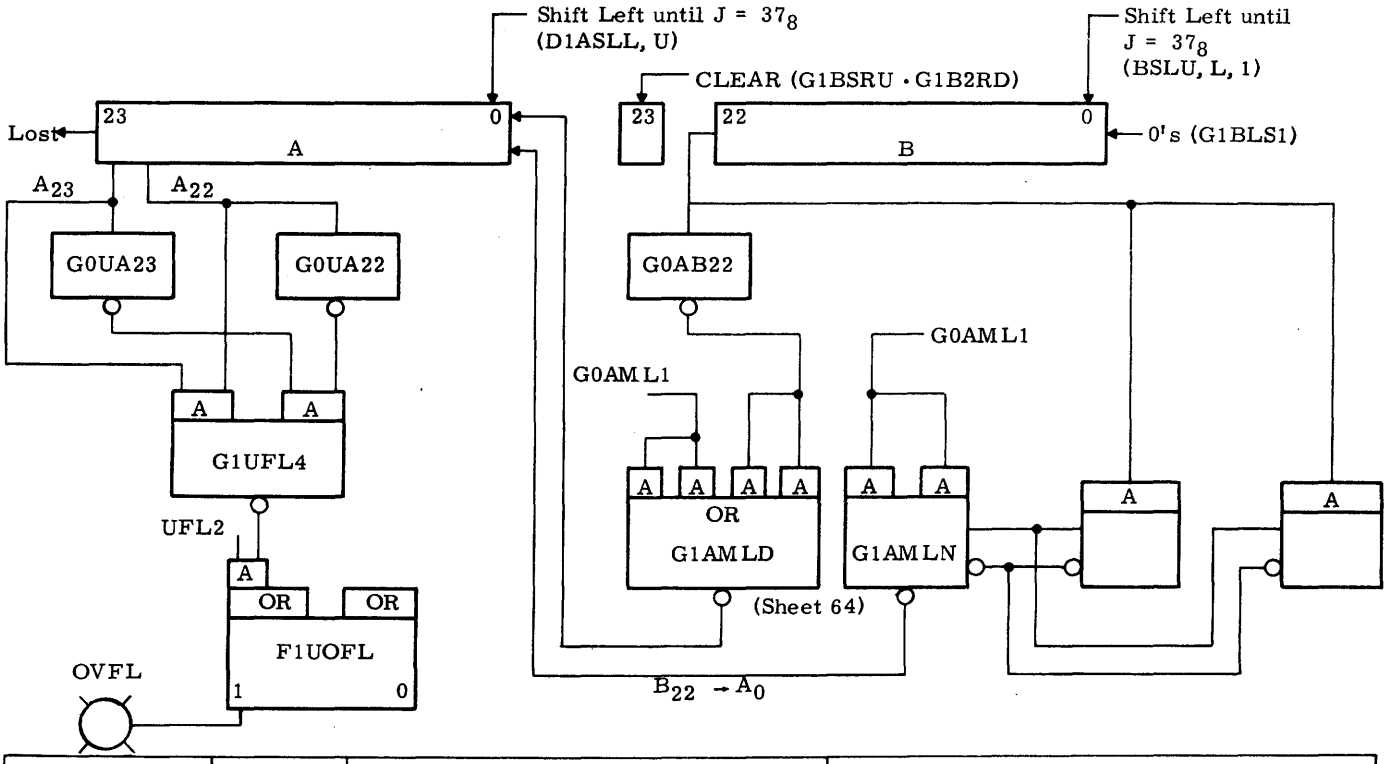
23 18	17 15	14 6	5	4 0
45	X	064	1	K

Z= f(X, K)

DLA shifts the contents of the A and Q Registers, Z places to the left. Bits shifted out of Q_{22} enter A_0. Bits shifted out of A_{23} are lost. Bit Q_{23} is cleared. If any of the bits shifted into A_{23} are unlike A_{23}'s original contents, the Overflow flip-flop (F1UOFL) is set.

During State 4, the contents of memory cell 10_8 are gated to the Arithmetic Unit B Register. The complement of I_{4-0} (Z) is gated to the J Counter. Both the A and B Registers are shifted left with bit 22 of the B Register applied to A_0 via G1AMLD and G1AMLN. "Zeros" are shifted into B_0 at each shift (G1BSL1). Bits shifted out of A_{23} are lost. If B_{23} is set, it is cleared by G1BSRU and G1B2RD. If at any shift of the A Register, bit 22 is not equal to bit 23, the Overflow flip-flop is set. This indicates that the sign bit of the A Register has been changed by the shift.

The J Counter is incremented at each shift of A and B and when J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. State 5 is then entered and the contents of B are stored in cell 10_8 (Q).



Logic Element	Logic Sheet	Logic Equation	Function
D1ASLL, U	63.1	G0ASL1 = AABL · TT6E · $\overline{JE37}$	Shift Left A_{23-0}
D1BSLL, 1, U	40	G0ASLB = TT6E · IR11 · $\overline{JE37}$ · AABL	Shift Left B_{23-0}
G1AMLD G1AMLN	64 64	AB22 · AML1 AML1 · $\overline{BR22}$	$B_{22} \rightarrow A_0$
G1BSRU G1B2RD	40 41	B3RD · BB23 B3RD · BB23	Clear B_{23}
G1BLS1	38	$\overline{DDIV}$ · BGN3	$0 \rightarrow B_0$
G1UFL4 G1UFL2	54 54	UA22 · $\overline{AR23}$ + UA23 · $\overline{AR22}$ ASL1 · IR05 · UGN3	Set Overflow

DLA Block Diagram

DLL-(SHIFT) DOUBLE LEFT LOGICAL

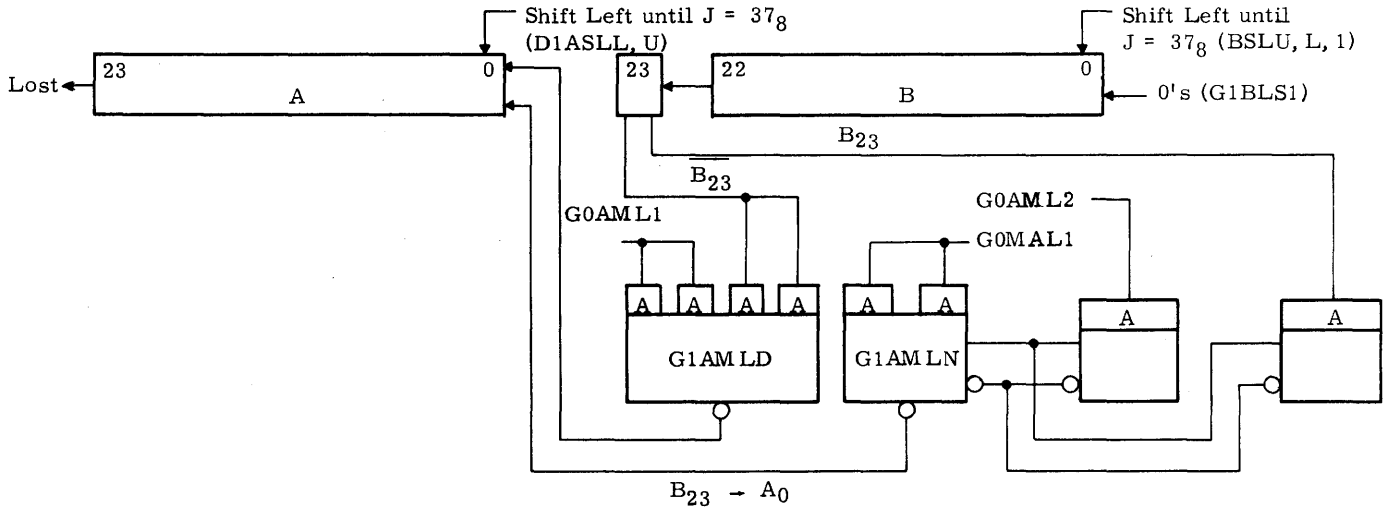
GEN 3

23 18	17 15	14 6	5	4 0
45	X	072	0	K

Z= f(X, K)

DLL shifts the contents of the A and Q Registers Z places to the left. Bits shifted out of Q_{23} enter A_0. Bits shifted out of A_{23} are lost. Zeros are shifted into Q_0.

During State 4, the contents of memory cell 10_8 are gated to the Arithmetic Unit B Register. The complement of I_{4-0} (Z) is gated to the J Counter. Both the A and B Registers are then shifted left. Bits shifted from B_{23} are applied to A_0 by G1AMLD and G1MAMN. "Zeros" are shifted into B_0 at each shift (G1BSL1). Bits shifted out of A_{23} are lost. The J Counter is incremented at each shift of A and B and when J is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. State 5 is then entered and the contents of B are stored in memory cell 10_8 (Q).



Logic Element	Logic Sheet	Logic Equation	Function
D1ASLL, U	63.1	G0ASL1 = AABL · TT6E · $\overline{JE37}$	Shift Left A_{23-0}
D1BSLL, 1, U	40	G0ASLB = TT6E · IR11 · $\overline{JE37}$ · AABL	Shift Left B_{23-0}
G1BLS1	38	$\overline{DDIV}$ · BGN3	0 → B_0
G1AMLD G1AMLN	64 64	AB23 · AML2 $\overline{BR23}$ · AML2	B_{23} → A_0
G1B2LD G1B3LD	38 38	B22R · BB22 BGN3 · $\overline{BR22}$	B_{22} → B_{23}

DLL Block Diagram

DRA-(SHIFT)DOUBLE RIGHT ARITHMETIC

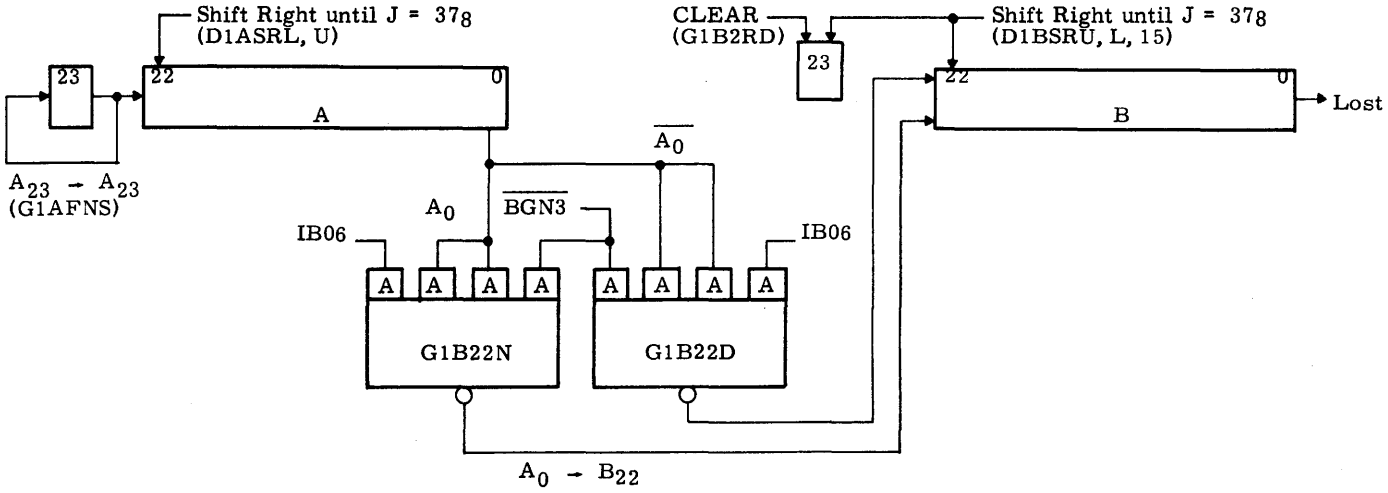
GEN 3

23 18	17 15	14 6	5	4 0
45	X	044	0	K

Z= f(X, K)

DRA Shifts the A and Q Registers Z places to the right. Bits shifted out of A_0 are shifted into Q_{22}. Bits shifted out of Q_0 are lost. Bits shifted into A_{22} are the same as A_{23}. A_{23} remains unchanged and Q_{23} is cleared.

During State 4, the contents of memory cell 10_8 are gated to the Arithmetic Unit B Register. The complement of I_{4-0} (Z) is gated to the J Counter. Both the A and B Registers are then shifted right. Bits shifted from A_0 are shifted into B_{22} via G1B22N and G1B22D. A_{23} is shifted to A_{22} and through the Serial Full Adder back to A_{23} leaving A_{23} unchanged. B_{23} is cleared by G1B2RD. Bits shifted from B_0 are lost. The J Counter is incremented at each shift of A and B. When the J Counter is incremented to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. State 5 is then entered and the contents of B are stored in cell 10_8 (Q) to complete execution of the DRA command.



Logic Element	Logic Sheet	Logic Equation	Function
D1ASRL, U	63.1	G0ASR2 = AABR · T6E3 · $\overline{JE37}$	Shift Right A_{23-0}
D1BSRU, L, 15	40	G0ASRB = AABR · $\overline{JE37}$ · IR11 · TT6E	Shift Right B_{23-0}
G1B22D	38	$\overline{BI06}$ · AR00 · BGN3	$A_0 \rightarrow B_{22}$
G1B22N	38	$\overline{BI06}$ · BGN3 · $\overline{BA00}$	
G1AFNS	67	G0AFNA = ANA3 · AR23	$A_{23} \rightarrow A_{23}$
$\overline{N0AFNS}$	67	$\overline{G1AFNS}$	
G1B2RD	41	B3RD · BB23	Clear B_{23}

DRA Block Diagram

DRC-(SHIFT) DOUBLE RIGHT CIRCULAR

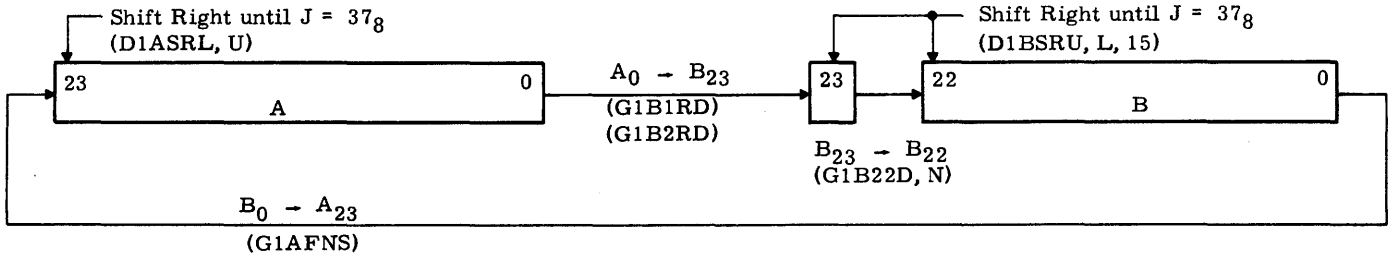
GEN 3

23 18	17 15	14 6	5	4 0
45	X	053	0	K

Z= f(X, K)

DRC shifts all 48 bits of the A and Q Registers Z places in a right circular fashion with Q_0 shifted into A_{23} and A_0 shifted to Q_{23}.

During State 4, the contents of memory cell 10_8 (Q) are gated to the B Register. The complement of I_{4-0} (Z) is gated to the J Counter. The A and B Registers are then shifted right with B_0 applied through the Serial Full Adder to A_{23}. A_0 is shifted through G1B1RD and G1B2RD to B_{23}. The J Counter is incremented at each shift of A and B. When the J Counter is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. State 5 is then entered and the contents of the B Register are stored in memory location 10_8 (Q) to complete execution of the DRC command.



DRC Block Diagram

Logic Element	Logic Sheet	Logic Equation	Function
D1ASRL, U	63.1	G0ASR2 = AABR · T6E3 · $\overline{JE37}$	Shift Right A_{23-0}
D1BSRU, L, 15	40	G0ASRB = AABR · $\overline{JE37}$ · IR11 · TT6E	Shift Right B_{23-0}
G1B1RD G1B2RD	41 41	AR00 · B22R B22R · $\overline{BA00}$	$A_0 \rightarrow B_{23}$
G1B22D G1B22N	38 38	$\overline{BB23}$ · $\overline{DMPY}$ · BGN3 · IR06 $\overline{BR23}$ · BGN3 · $\overline{DMPY}$ · IR06	$B_{23} \rightarrow B_{22}$
G1AFNS $\overline{N0AFNS}$	67 67	G0AFNA = DGN3 · BR00 · IR09 $\overline{G1AFNS}$	$B_0 \rightarrow A_{23}$

DRC Block Diagram

DRL-(SHIFT) DOUBLE RIGHT LOGICAL

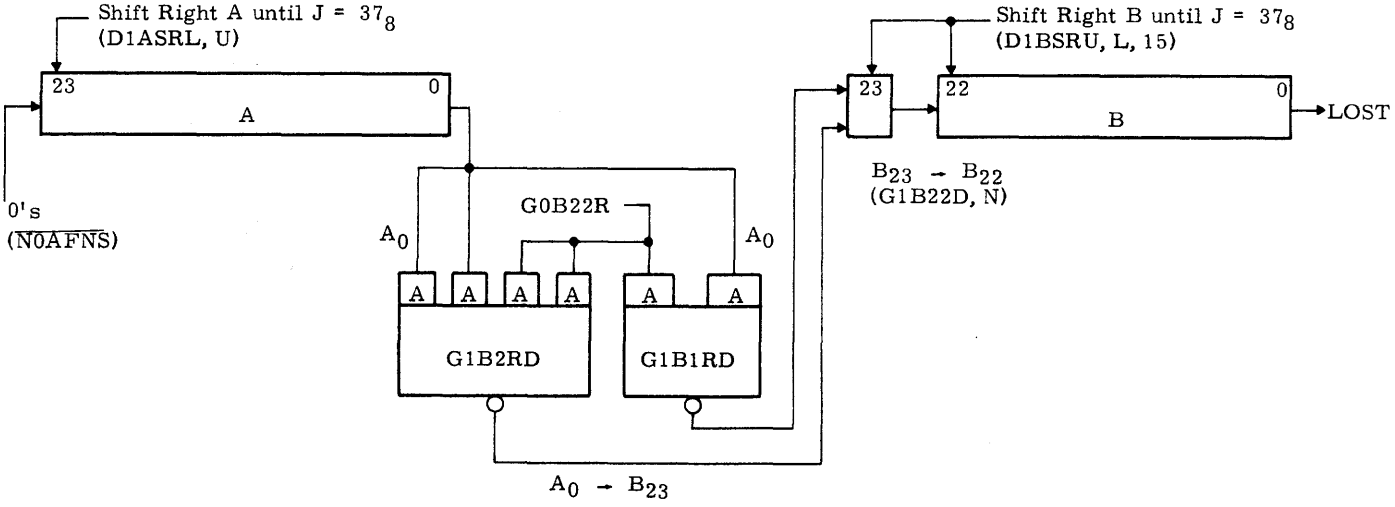
GEN 3

23 18	17 15	14 6	5	4 0
45	X	043	0	K

Z= f(X, K)

DRL shifts all 48 bits of the A and Q Registers Z places to the right. A_0 is shifted into Q_{23}. Bits shifted out of Q_0 are lost. Zeros are shifted into A_{23}.

During State 4, the contents of memory cell 10_8 (Q) are gated to the B Register. The complement of I_{4-0} (Z) is gated to the J Counter. The contents of both the A and B Registers are then shifted right with the bits shifted from A_0 applied to B_{23} via G1B1RD and G1B2RD. "Zeros" are shifted into A_{23} by inhibiting the Serial Full Adder. Bits shifted from B_0 are lost. The J Counter is incremented at each shift of the A and B Registers. When the J Counter is equal to 37_8, the number of shifts specified by Z have occurred and further shifting is inhibited. Sequence State 5 is then entered and the contents of the B Register are stored in memory cell 10_8 (Q) to complete the execution of the DRL command.



Logic Element	Logic Sheet	Logic Equation	Function
D1ASRL, U	63.1	G0ASR2 = AABR · T6E3 · $\overline{JE37}$	Shift Right A_{23-0}
D1BSRU, L, 15	40	G0ASRB = AABR · $\overline{JE37}$ · IR11 · TT6E	Shift Right B_{23-0}
G1B1RD G1B2RD	41 41	AR00 · B22R B22R · $\overline{BA00}$	$A_0 \rightarrow B_{23}$
G1B22D G1B22N	38 38	$\overline{BB23}$ · $\overline{DMPY}$ · BGN3 · IR06 $\overline{BR23}$ · BGN3 · $\overline{DMPY}$ · IR06	$B_{23} \rightarrow B_{22}$

DRL Block Diagram

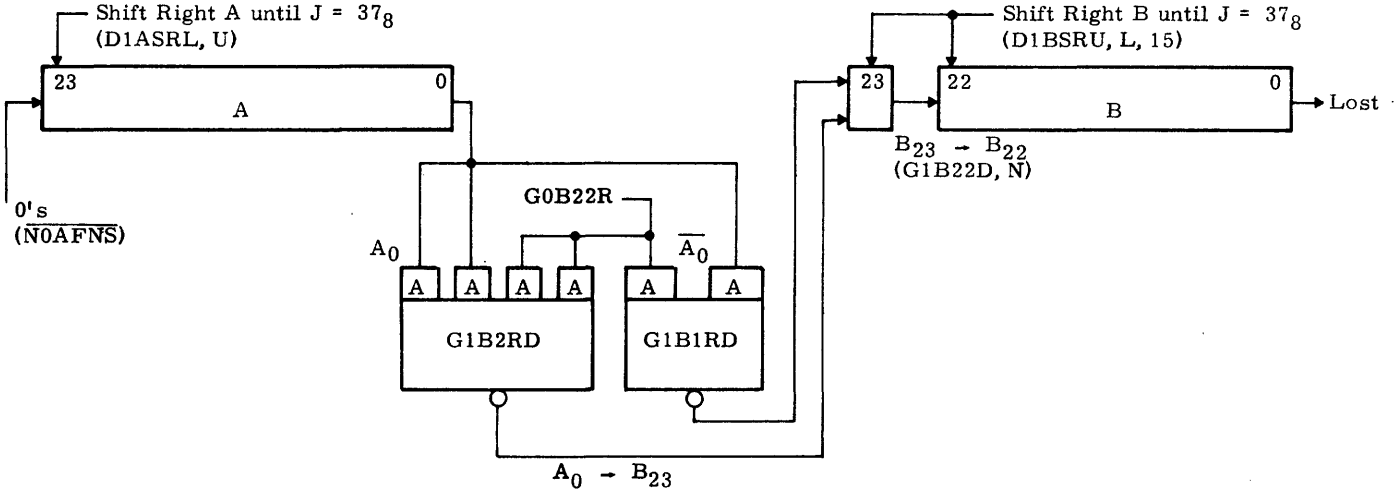
MAQ-MOVE A TO Q

GEN 3

23							0
4	5	0	0	4	3	3	0

MAQ places the contents of the A Register into the Q Register. The original contents of Q are lost. Zeros are placed in the A Register.

During State 4, the contents of memory location 10_8 (Q) are gated to the B Register. The complement of I_{4-0} (bits 4 through 0 of the MAQ command) are gated to the J Counter. Since bits 2 through 0 of the MAQ command are zero, this presets the J Counter equal to 7. The A and B Registers are then shifted right with the J Counter incremented at each shift. Bits shifted from A_0 are gated through G1B1RD and G1B2RD to B_{23}. Bits shifted from B_0 are lost. "Zeros". "Zeros" are shifted to A_{23} from the disabled Serial Full Adder. Since the J Counter was preset to 7, 24 shifts will have occurred when the J Counter is equal to 37_8 and further shifting is inhibited. After 24 shifts, the contents of A have been shifted to B, "zeros" have been shifted to all 24 bits of A, and the original contents of B (Q) are lost. State 5 is then entered and the contents of B are stored in cell 10_8 (Q) to complete the MAQ command.



Logic Element	Logic Sheet	Logic Equation	Function
D1ASRL, U	63.1	G0ASR2 = AABR · T6E3 · $\overline{JE37}$	Shift Right A_{23-0}
D1BSRU, L, 15	40	G0ASRB = AABR · $\overline{JE37}$ · IR11 · TT6E	Shift Right B_{23-0}
G1B1RD G1B2RD	41 41	AR00 · B22R B22R · $\overline{BA00}$	$A_0 \rightarrow B_{23}$
G1B22D G1B22N	38 38	$\overline{BB23}$ · $\overline{DMPY}$ · BGN3 · IR06 $\overline{BR23}$ · BGN3 · $\overline{DMPY}$ · IR06	$B_{23} \rightarrow B_{22}$

MAQ Block Diagram

SLA-SHIFT LEFT ARITHMETIC

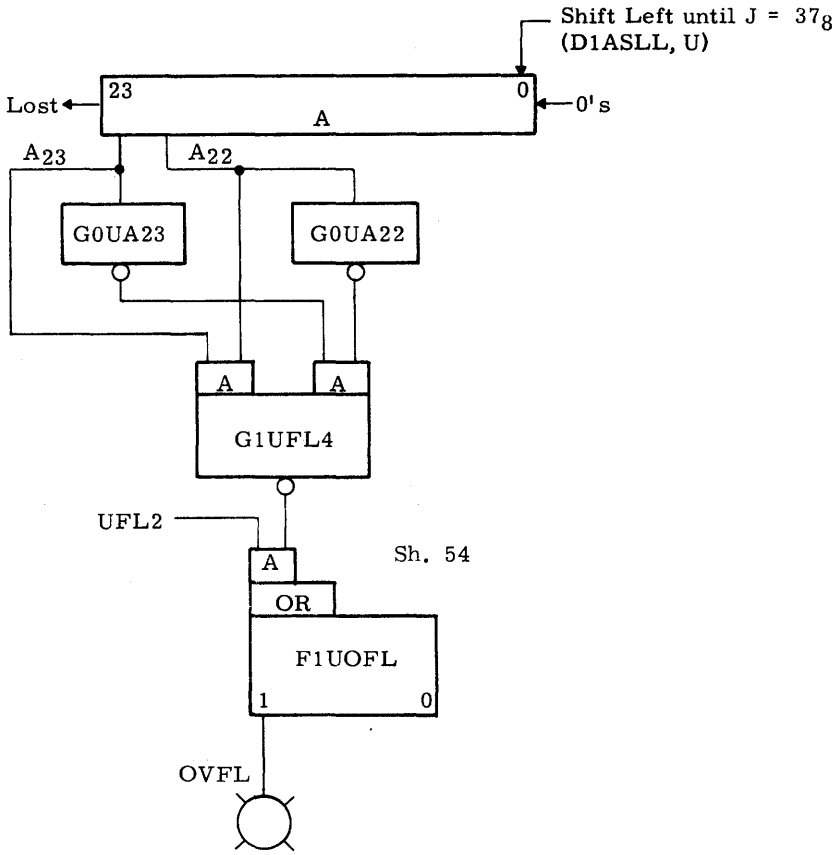
GEN 3

23 18	17 15	14 6	5	4 0
45	X	020	1	K

Z= f(X, K)

SLA shifts the contents of the A Register Z places to the left. Bits shifted out of A_{23} are lost. Zeros are shifted into A_0. The Overflow flip-flop (F1UOFL) is set if any bit shifted into A_{23} is unlike the original content of A_{23}.

During State 4, the contents of memory location 10_8 (Q) are gated to the B Register. The complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted left with "zeros" shifted into A_0 (G1AMLN). Bits shifted out of A_{23} are lost. If at any shift of A, bit 23 changes ($A_{23} \neq A_{22}$) the Overflow flip-flop is set. At each shift of A, the J Counter is incremented. When the J Counter is equal to 37_8, further shifting of A is inhibited. State 5 is then entered and the contents of B (original contents of Q) are gated back to cell 10_8 unchanged. This completes execution of the SLA command.



Logic Element	Logic Sheet	Logic Equation	Function
D1ASLL, U	63.1	G0ASL1 = AABL · TT6E · $\overline{JE37}$	Shift Left A_{23-0}
G1AMLN	64	G0AML3 = DGN3 · $\overline{WI08}$ · $\overline{AIR9}$	0 → A_0
G1UFL4 G1UFL2	54 54	UA22 · $\overline{AR23}$ + UA23 · $\overline{AR22}$ ASL1 · IR05 · UGN3	Set Overflow

SLA Block Diagram

SLL-SHIFT LEFT LOGICAL

GEN 3

23 18	17 15	14 6	5	4 0
45	X	020	0	K

Z= f(X, K)

SLL shifts the contents of the A Register Z places to the left. Bits shifted out of A_{23} are lost. Zeros are shifted into A_0.

During State 4, the contents of memory location 10_8 (Q) are gated to the B Register although this data is not changed or used during the SLL command. The complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted left. "Zeros" are shifted into A_0 from G1MALN. Bits shifted out of A_{23} are lost. The J Counter is incremented at each shift of A. When the J Counter is equal to 37_8, the A Register has been shifted left the number of places specified by Z and further shifting is inhibited. State 5 is then entered and the contents of B are stored back in cell 10_8 (Q). The contents of B (Q) are not changed by the SLL command.

Logic Element	Logic Sheet	Logic Equation	Function
D1ASLL, U	63.1	G0ASL1 = AABL · TT6E · $\overline{JE37}$	Shift Left A_{23-0}
G1AMLN	64	G0AML3 = DGN3 · $\overline{WI08}$ · $\overline{AIR9}$	0 → A_0

SLL Block Diagram

SLL-SHIFT LEFT LOGICAL

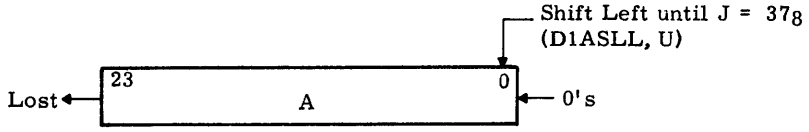
GEN 3

23 18	17 15	14 6	5	4 0
45	X	020	0	K

Z= f(X, K)

SLL shifts the contents of the A Register Z places to the left. Bits shifted out of A_{23} are lost. Zeros are shifted into A_0.

During State 4, the contents of memory location 10_8 (Q) are gated to the B Register although this data is not changed or used during the SLL command. The complement of I_{4-0} (Z) is gated to the J Counter. The A Register is then shifted left. "Zeros" are shifted into A_0 from G1MALN. Bits shifted out of A_{23} are lost. The J Counter is incremented at each shift of A. When the J Counter is equal to 37_8, the A Register has been shifted left the number of places specified by Z and further shifting is inhibited. State 5 is then entered and the contents of B are stored back in cell 10_8 (Q). The contents of B (Q) are not changed by the SLL command.



Logic Element	Logic Sheet	Logic Equation	Function
D1ASLL, U	63.1	G0ASL1 = AABL · TT6E · $\overline{JE37}$	Shift Left A_{23-0}
G1AMLN	64	G0AML3 = DGN3 · $\overline{WI08}$ · $\overline{AIR9}$	0 → A_0

SLL Block Diagram

INX - INCREMENT X

FULL OPERAND

23 18	17 15	14	13 0
26	X	*	K

Z = f (K, *)

INX adds the numeric value of Z to the contents of the X core cell specified by bits 17 through 15 of the command. The value of K must have a value within the range of +8,191 to -8,192. As the INX instruction is performed, K_{13} is extended into bit 14 (i.e., if $K_{13} = 0$, bit 14 is set to 0 and represents a positive incrementation; if $K_{13} = 1$, bit 14 is set to 1 and represents a negative incrementation). Then, K_{14-0} is added to the contents of X_{23-0} and stored back in X. If the INX command is relative addressed, K_{14-0} plus P_{14-0} are added to X_{23-0} and stored back in X.

NOTE

An INX command specifying to increment X cell 2 by zero (26200000) is NOP - No Operation. If the index field of the INX command (17 through 15) is equal to zero, the instruction is undefined.

Non-Indexed Word Times.	3 (S1, S2, S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory X	C (X_{23-0}) + Z

COMMAND CHARACTERISTICS

Sequence Control State 1 operates as a normal "fetch" cycle for the INX command except that bit 13 of the INX command is applied to both bits 13 and 14 of the Adder Unit and the result transferred to $I_{A,13-0}$. In this manner, a negative value of K in 2's complement form will be extended from 14 bits (13 through 0) to 15 bits in the I Register ($I_{A,13-0}$).

Following State 1, State 2 is entered. Memory is addressed from the X bits of the I Register (17 through 15) and the contents of the addressed X cell are gated to the B Register. From B, the contents of the addressed X cell are gated to the Adder Unit (D1UBBU). At the same time, the contents of $I_{A,13-0}$ (Z) are gated to the Adder Unit (D1UILA). The sum of these two adder inputs is then gated back to the B Register at the end of State 2.

Following State 2, State 4 is entered. During State 4, memory is again addressed from the X bits of the I Register and the contents of the B Register (original contents of the X cell plus Z) are gated back to the X cell.

The following examples illustrate the operation of the INX command.

Example 1:

K = -1

23 14	13 0
	1 ——— 1

X = 0

23 0
0 ——— 0

After Execution,

X = -1 in bits 14-0

23 15	14 0
0 ——— 0	1 ——— 1

Example 2:

K = +8191

23 14	13 12 0
	0 1 ——— 1

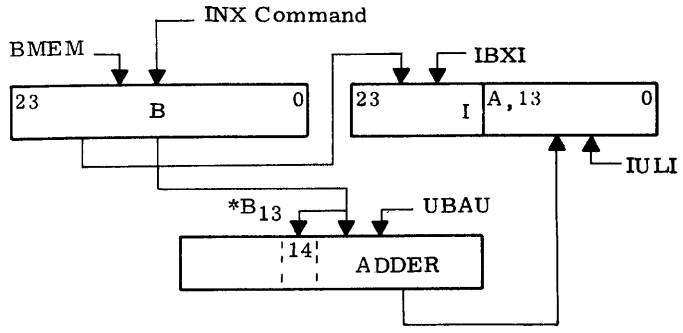
X = + 16384

23 15	14	13	0
0 ——— 0	1	0	——— 0

After Execution

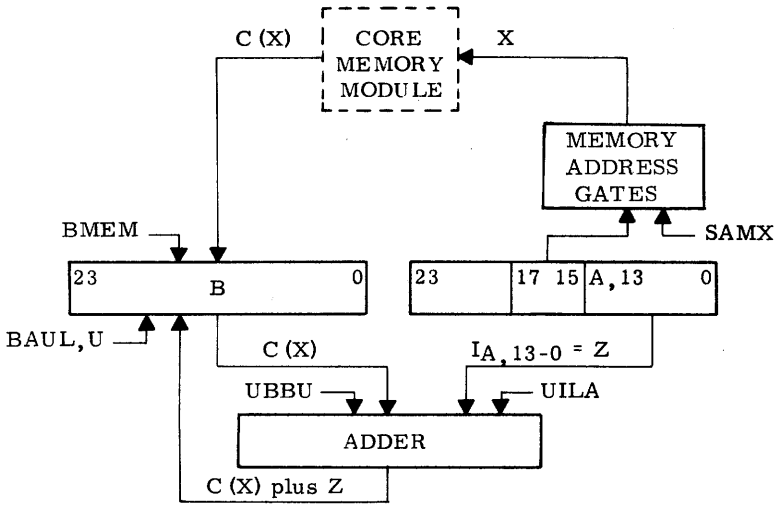
+24575

20 15	14	13	12	0
0 ——— 0	1	0	1	——— 1

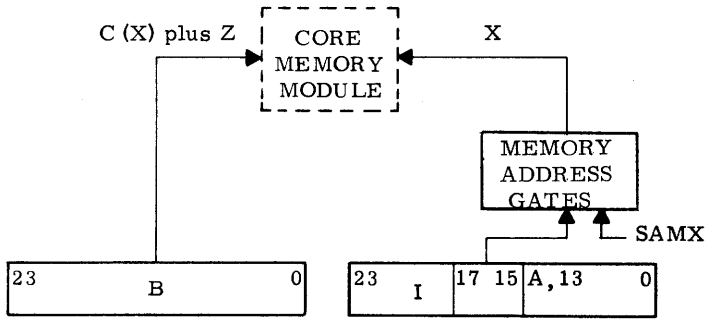


* G1UB14 (49) = UINX · USIA · BR13

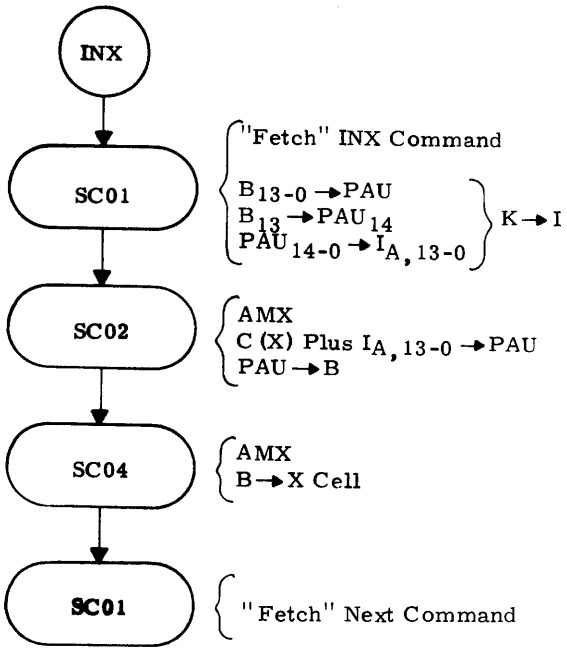
Sequence State 1



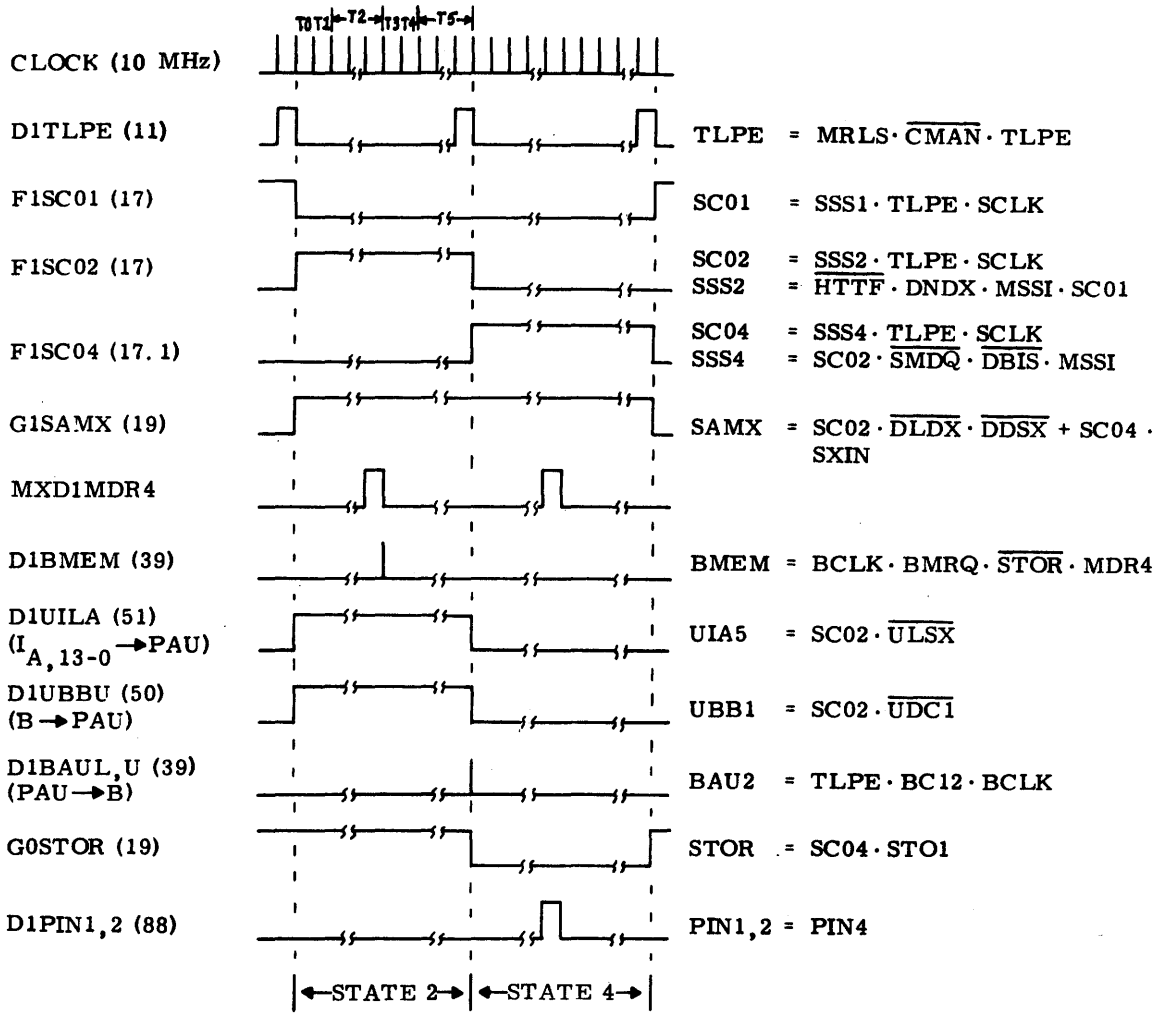
Sequence State 2



Sequence State 4



INX BLOCK DIAGRAM



INX TIMING DIAGRAM

LDA - LOAD THE A REGISTER

FULL OPERAND

23 18	17 15	14	13 0
00	X	*	Y

Z = f (X,*,Y)

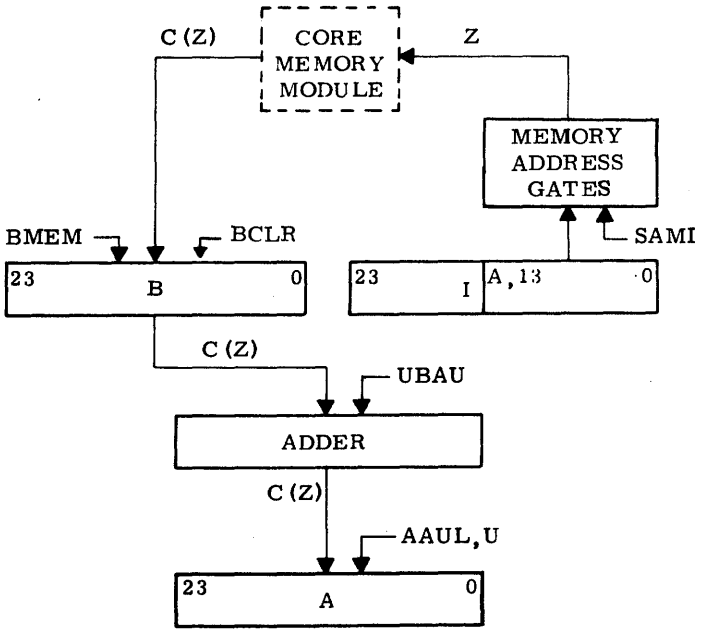
LDA places the contents of memory location Z into the A Register. The contents of memory location Z are unchanged.

A non-indexed LDA command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The contents of memory location Z are gated to the B Register by D1BMEM during the Clock Pulse of Memory Data Ready (MXD1MDR4).

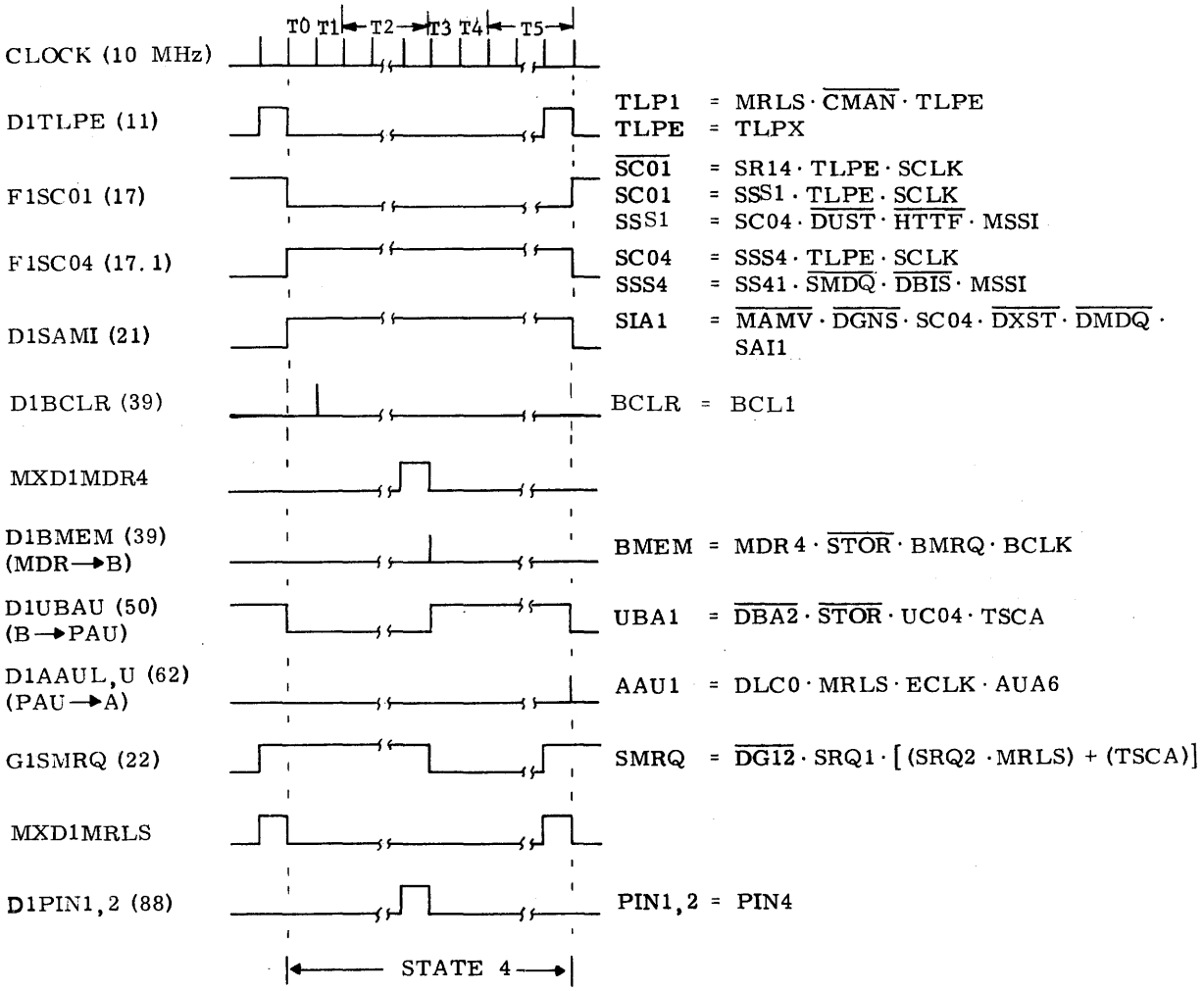
From B, the contents of memory location Z are gated to the Adder Unit (D1UBAU). At the Clock of Last Pulse, the contents of the Adder Unit are gated to the A Register (D1AAUL,U), completing execution of the LDA command. Sequence Control State 1 is then entered to "fetch" the next command.

Non-Indexed Word Times.	2 (S1,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	C (Z)
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS



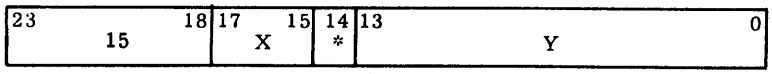
LDA BLOCK DIAGRAM



LDA TIMING DIAGRAM

LDP - LOAD PLACE

FULL OPERAND



Z = f (X, *, Y)

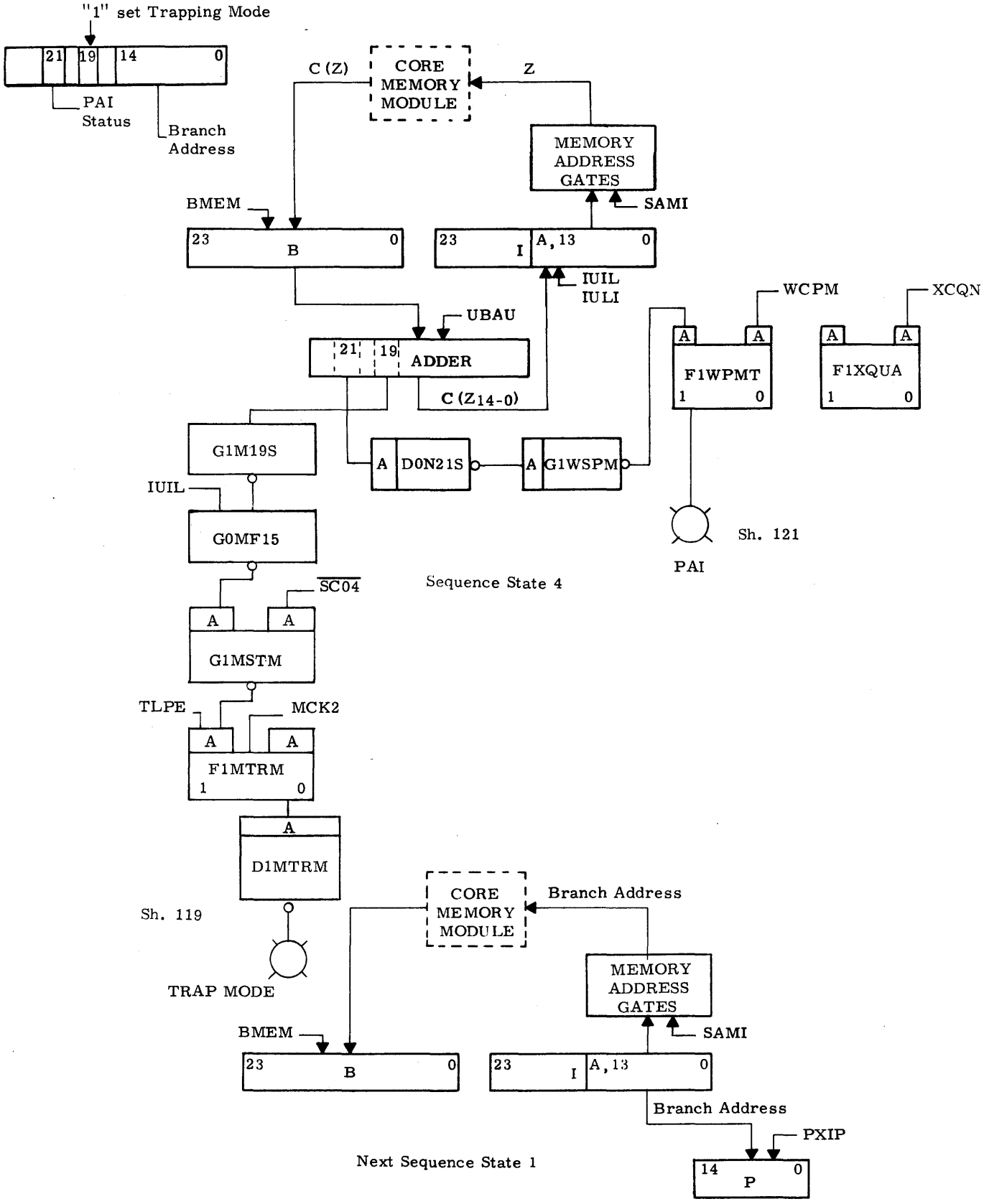
LDP transfers program control to the location specified by the contents of Z_{14-0}. Bit 21 of Z controls the status of the Permit Automatic Interrupt flip-flop (F1WPMT); if bit 21 is a "one", F1WPMT is set; if bit 21 is a "zero", F1WPMT is cleared. Bit 19 of Z, if a "one", sets the Trapping Mode flip-flop, F1MTRM. The LDP command clears the Quasi flip-flop, F1XQUA, if previously set.

The LDP command is "fetched" during a normal Sequence Control State 1. At the last pulse of this State 1, the Remember flip-flop (F1XRMF) is set. Following this State 1, State 4 is entered. Memory is addressed from $I_{A,\ 13-0}$ and the contents of core cell Z are gated to the B Register. From B, the contents of core cell Z are gated to the Adder Unit (D1UBAU). Bits 14-0 are then gated to $I_{A,\ 13-0}$ at memory release (D1LULI, UIL). At Time 4, the Permit Automatic Interrupt flip-flop is cleared. At the Clock of Last Pulse Envelope it is set if bit 21 in the Adder (i.e., bit 21 of Z) is a "one". If Bit 19 of Z is a "one", F1MTRM is also set at last pulse. Also, at Last Pulse Envelope the Quasi flip-flop, F1XQUA, is reset if previously set.

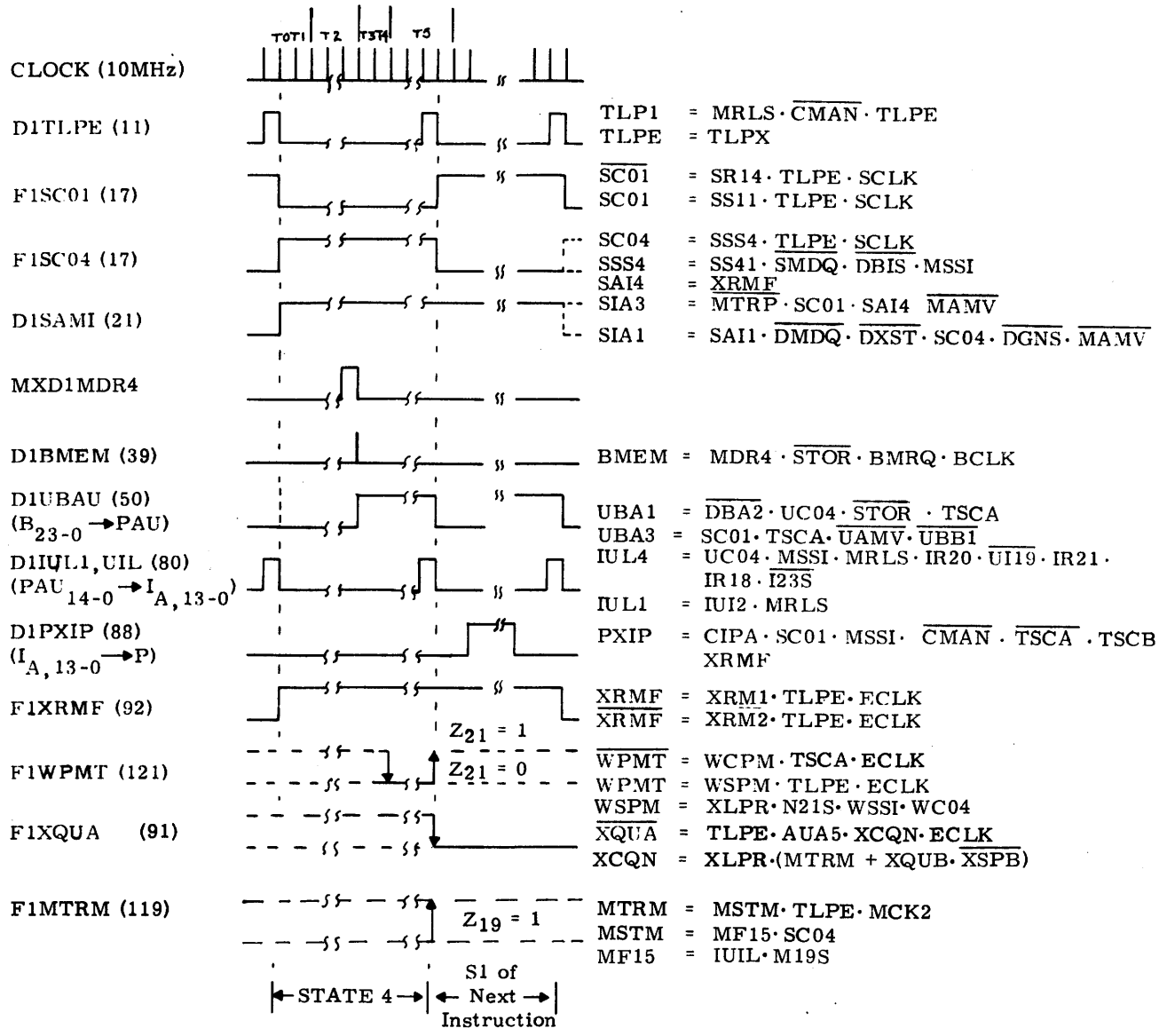
Following this State 4, State 1 is entered to "fetch" the next command. Memory is addressed from $I_{A,13-0}$ and the contents of $I_{A,13-0}$ are transferred to the P Register to complete the transfer of program control. This State 1 then operates in the normal manner except that at the Clock of Last Pulse Envelope, the Remember flip-flop is cleared. Sequencing then continues to execute this command.

Non-Indexed Word Times.	2 (S1, S4)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (Z_{14-0})
F1WPMT	Set if $Z_{21} = 1$ Reset if $Z_{21} = 0$
F1UOFL	
F1ETST	
F1MTRM	Set if $Z_{19} = 1$
Memory Z	
F1XQUA	Reset

COMMAND CHARACTERISTICS



LDP BLOCK DIAGRAM



LDP TIMING DIAGRAM

LDQ - LOAD THE Q REGISTER

FULL OPERAND

23 18	17 15	14	13 0
42	X	*	Y

Z = f (X, *, Y)

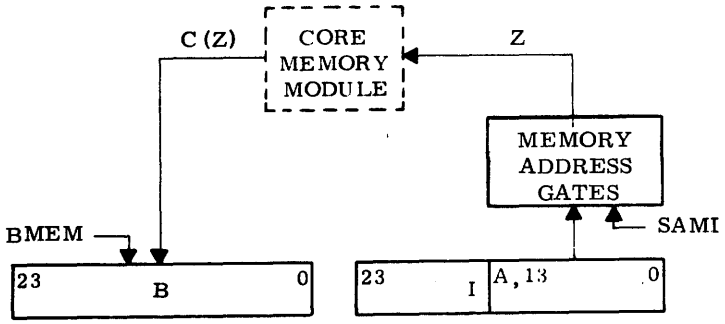
LDQ places the contents of memory location Z into the Q Register (memory location 10_8). The contents of memory location Z are unchanged by the LDQ command.

The LDQ command is executed during Sequence Control States 4 and 5. During State 4, memory is addressed from $I_{A,13-0}$ (SAMI) and the contents of memory location Z are gated to the B Register.

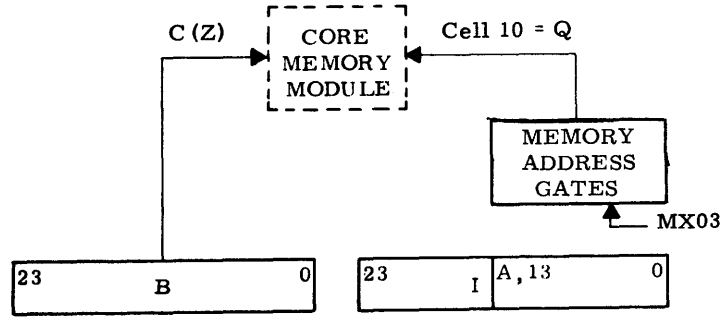
Following State 4, State 5 is initiated. During State 5, memory location 10_8 is addressed by enabling G0MX03, and the contents of the B Register are stored in cell 10_8. In this manner, the contents of memory location Z are stored in the Q Register (memory cell 10_8).

Non-Indexed Word Times.	3 (S1, S4, S5)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	C (Z)
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

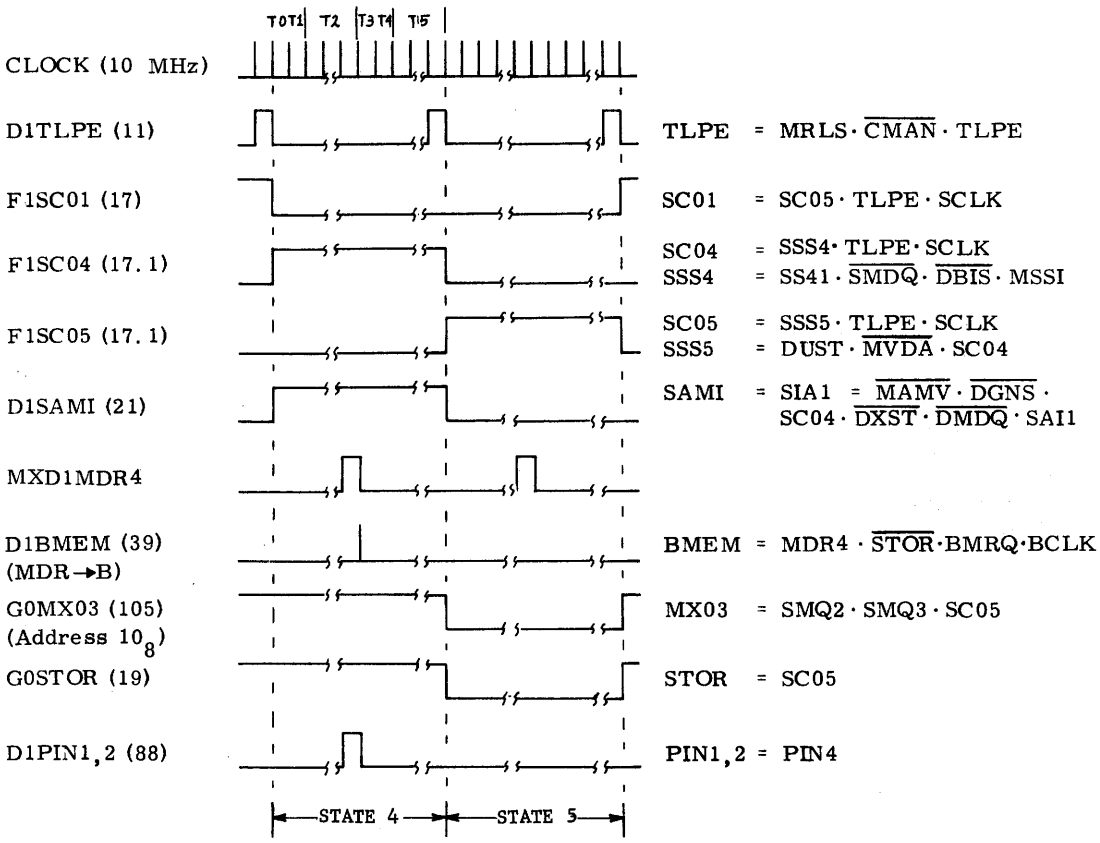


Sequence State 4



Sequence State 5

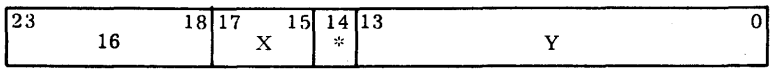
LDQ BLOCK DIAGRAM



LDQ TIMING DIAGRAM

LDX - LOAD X LOCATION FROM Z

FULL OPERAND



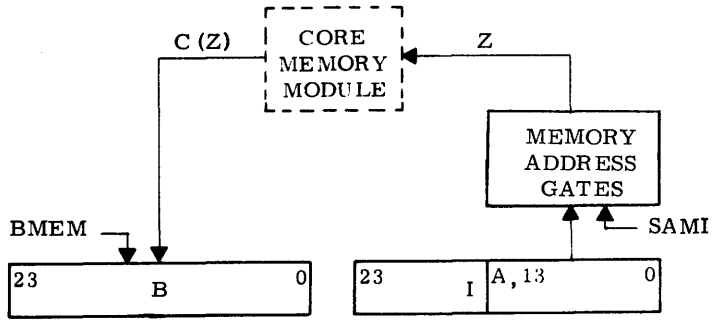
Z = f (Y,*)

LDX places the contents of memory location Z into the specified (bits 17 through 15) index core cell. If the index bits (bits 17 through 15) of the LDX command are zero, the command is undefined.

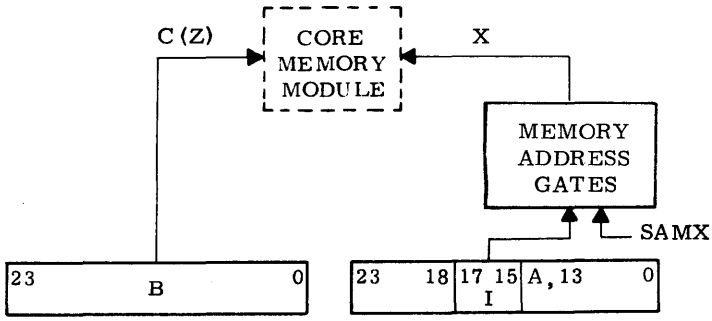
The LDX command is executed during Sequence Control States 2 and 4. During State 2, memory is addressed from $I_{A,13-0}$ (D1SAMI) and the contents of memory location Z are gated to the B Register. Following State 2, State 4 is entered. During State 4, memory is addressed from I_{17-15} (G1SAMX) and the contents of the B Register are stored in the index core cell specified by bits 17 through 15 of the LDX command.

Word Times.	3 (S1,S2,S4)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory X	C (Z)

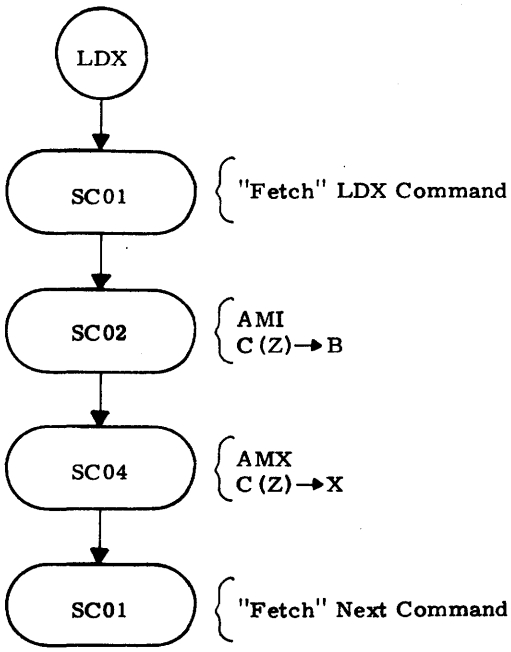
COMMAND CHARACTERISTICS



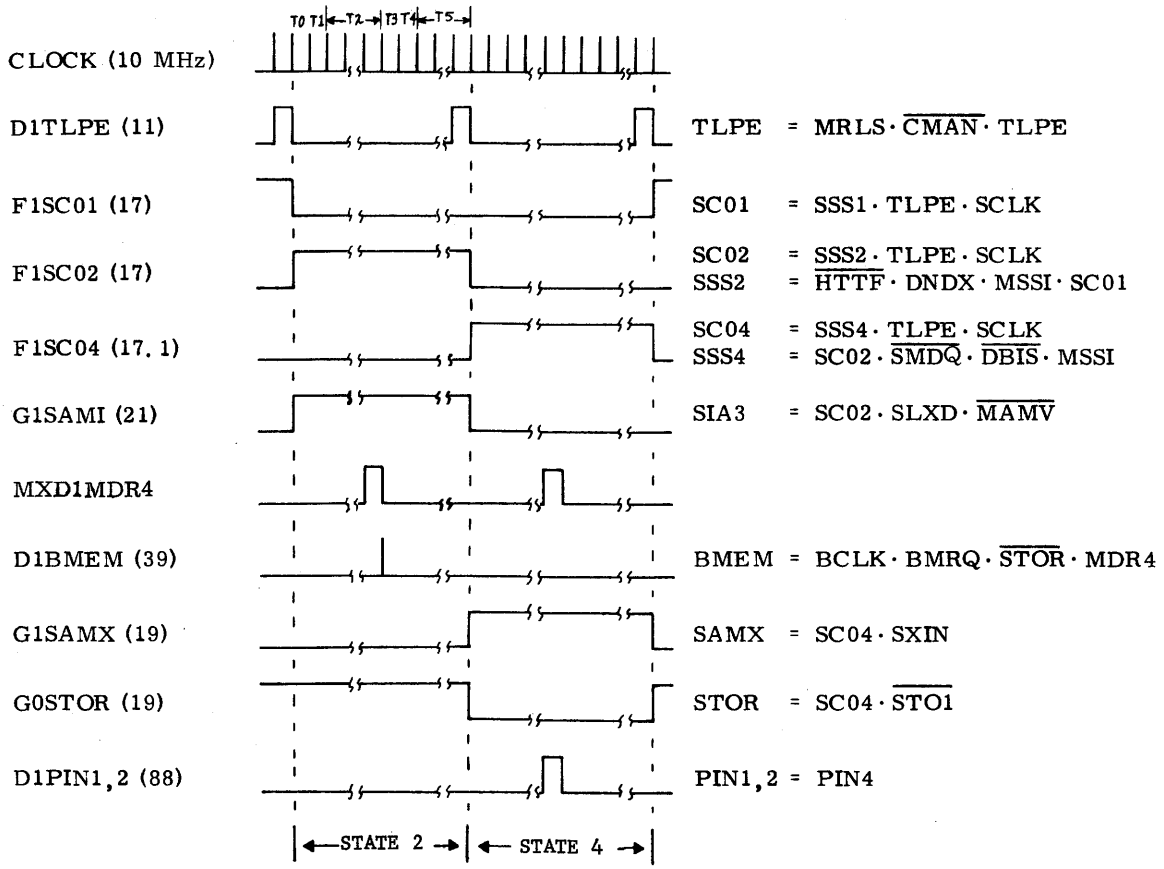
Sequence State 2



Sequence State 4



LDX BLOCK DIAGRAM



LDX TIMING DIAGRAM

LPR - LOAD PLACE AND RESTORE

FULL OPERAND

23 18	17 15	14	13 0
35	X	*	Y

Z = f (X,*,Y)

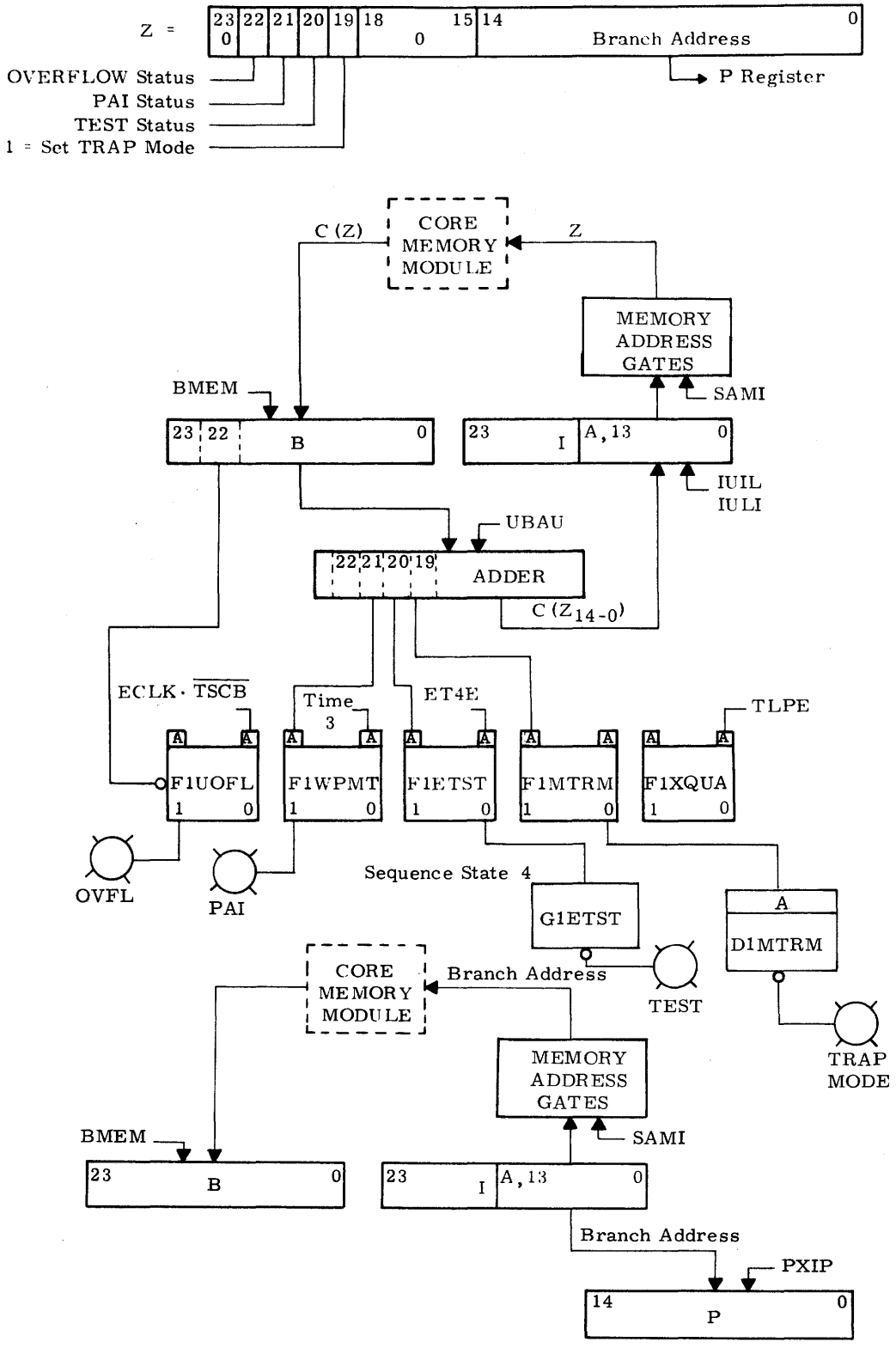
LPR restores the status of the Overflow flip-flop (F1UOFL), Permit Automatic Interrupt flip-flop (F1WPMT), and Test flip-flop (F1ETST) from the contents of bits 22, 21, and 20, respectively, of core location Z. The contents of Z_{14-0} are placed in the P Register to transfer program control. If bit 19 of Z is a "one", the Trapping Mode flip-flop (F1MTRM) is set. If bit 19 of Z is a "zero", the status of the Trapping Mode flip-flop is unchanged. The LPR command resets the Quasi flip-flop (F1XQUA) when in the Trapping Mode.

The LPR command is "fetched" during a normal Sequence Control State 1 (SC01). At Last Pulse Envelope of State 1, the Remember flip-flop (F1XRMF) is set. Following State 1, State 4 is entered and memory is addressed from $I_{A,13-0}$. The contents of memory location Z are gated from memory to B and from B to the Adder Unit. The Overflow, Permit Automatic Interrupt, Test, and Quasi flip-flops are cleared during State 4. The Overflow flip-flop is then set, if bit 22 of the Adder is a "one"; the Permit Automatic Interrupt flip-flop is set if Adder bit 21 is a "one"; the Test flip-flop is set if Adder bit 20 is a "one"; and the Trapping Mode flip-flop is set if Adder bit 19 is a "one". The contents of Adder bits 14-0 are transferred to $I_{A,13-0}$.

Following State 4, State 1 is entered to "fetch" the next instruction as specified by the contents of Z_{14-0}. Memory is addressed from $I_{A,13-0}$ to obtain this instruction. The contents of $I_{A,13-0}$ are then transferred to the P Register during this State 1, thereby transferring program control to the address specified by the contents of Z_{14-0}. At Last Pulse Envelope of State 1, the Remember flip-flop is cleared. Sequencing then continues to execute the new command (i.e., the command located in the address specified by the contents of Z_{14-0}).

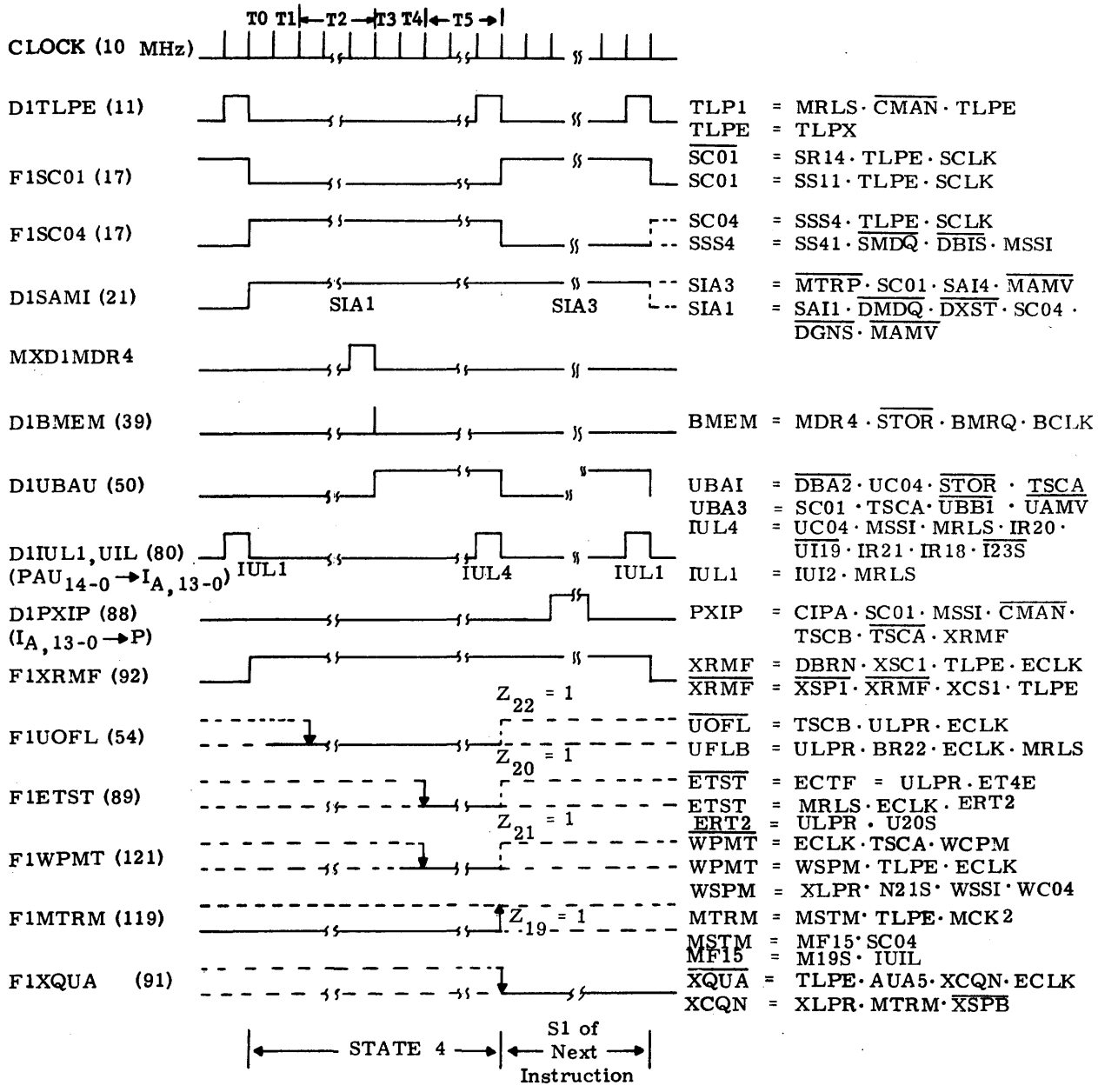
Non-indexed Word Times.	2 (S1,S4)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	$C(Z_{14-0})$
F1WPMT	Set if $C(Z_{21}) = 1$ Reset if $C(Z_{21}) = 0$
F1UOFL	Set if $C(Z_{22}) = 1$ Reset if $C(Z_{22}) = 0$
F1ETST	Set if $C(Z_{20}) = 1$ Reset if $C(Z_{20}) = 0$
F1MTRM	Set if $C(Z_{19}) = 1$
F1XQUA	Reset

COMMAND CHARACTERISTICS



Next Sequence State 1

LPR BLOCK DIAGRAM



LPR TIMING DIAGRAM

LXC - LOAD X WITH COUNT

FULL OPERAND

23 18	17 15	14	13 0
17	X	0	SEE TEXT

The function performed by the LXC command depends upon the configuration of bits 14 through 0. If bits 14 through 0 of the LXC command are "zero", the contents of the J Counter are stored in bits 4 through 0 of the addressed X cell. Bits 23 through 5 of the addressed X cell are replaced by "zeros". If bits 14 through 0 are not "zeros", the LXC command causes the following to occur:

1. Bits 23 through 14 of the indicated X cell are cleared.
2. Bits 13 through 5 of the indicated X cell are replaced by bits 13 through 5 of the instruction operand. If the LXC command is relative addressed (bit 14 is a "one"), bits 15 through 5 are replaced by the sum of bits $I_{A,13-5}$ of the instruction operand plus the core address (P) of the LXC command.
3. Bits 4 through 0 of the indicated X cell are replaced by the logical "OR" of J_{4-0} and I_{4-0}.

The LXC command is executed during Sequence Control States 2 and 4. During State 2, the contents of the J Counter are transferred (single-ended) to bits 4 through 0 of the I Register. Then, $I_{A,13-0}$ is gated to the Adder Unit (D1UILA). The contents of the Adder (24 bits) are then transferred to the B Register.

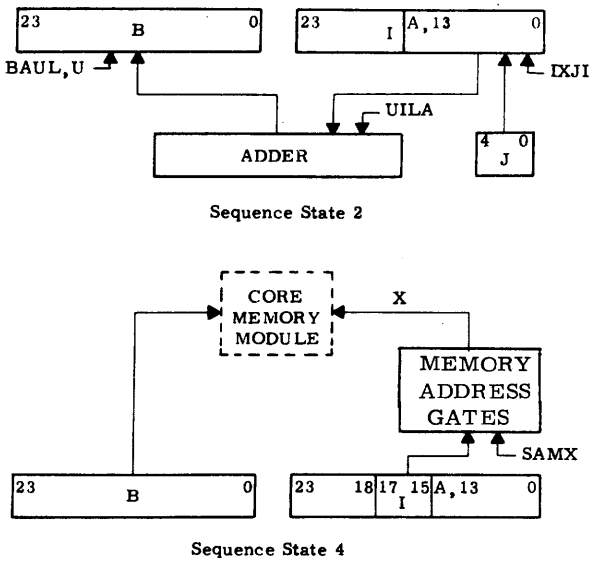
> **NOTE**
>
> Although memory is requested during State 2 (G1SMRQ), operation of the LXC command is not affected. The contents of core cell 0 will be transferred to the B Register but this data will be destroyed by the double-ended transfer of the Adder to the B Register. This operation is implemented for memory protect error detection.

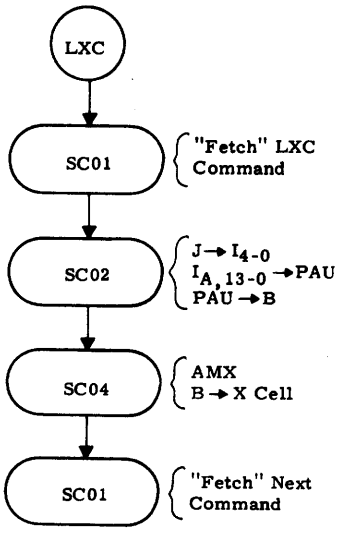
Following State 2, State 4 is entered. Memory is addressed from the X bits of the I Register (17 through 15). The contents of the B Register are then stored in the addressed X cell to complete the LXC operation.

Word Times.	3 (S1,S2,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory X	23 Zero 5 | 4 C (J_{4-0})* 0

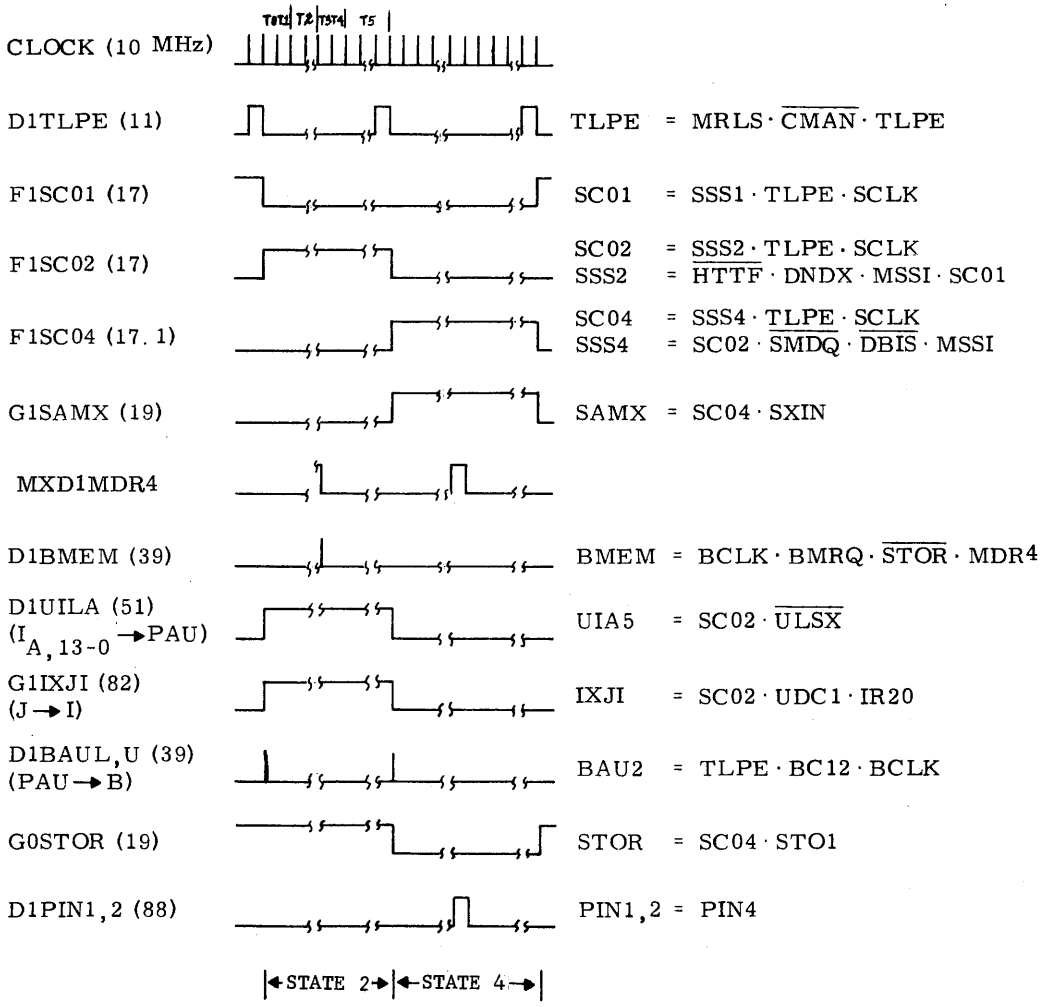
COMMAND CHARACTERISTICS

* If LXC = 17X00000. See text for contents of X following other LXC commands.





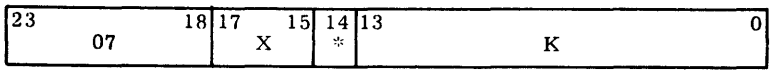
LXC BLOCK DIAGRAM



LXC TIMING DIAGRAM

LXK - LOAD X WITH K

FULL OPERAND



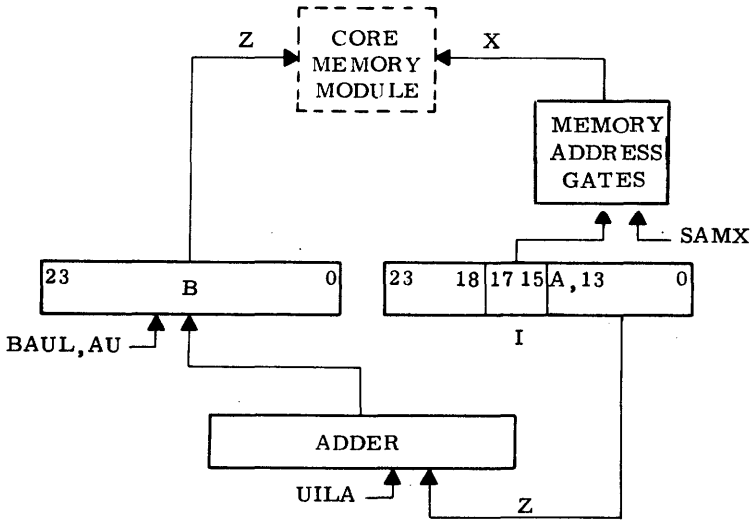
Z = f (K,*)

LXK stores the value Z into the addressed X core cell. Leading bits of the addressed X cell are set to "zero". The range of K, when not relative addressed, may vary from 0 to +16,383. The range of K when relative addressed may vary from -8,192 to +8,191 since bit 13 represents the sign of bits 12-0 when relative addressed. If bits 17, 16, and 15 of the LXK command are zero, the command is undefined (i.e., an X cell address must be specified).

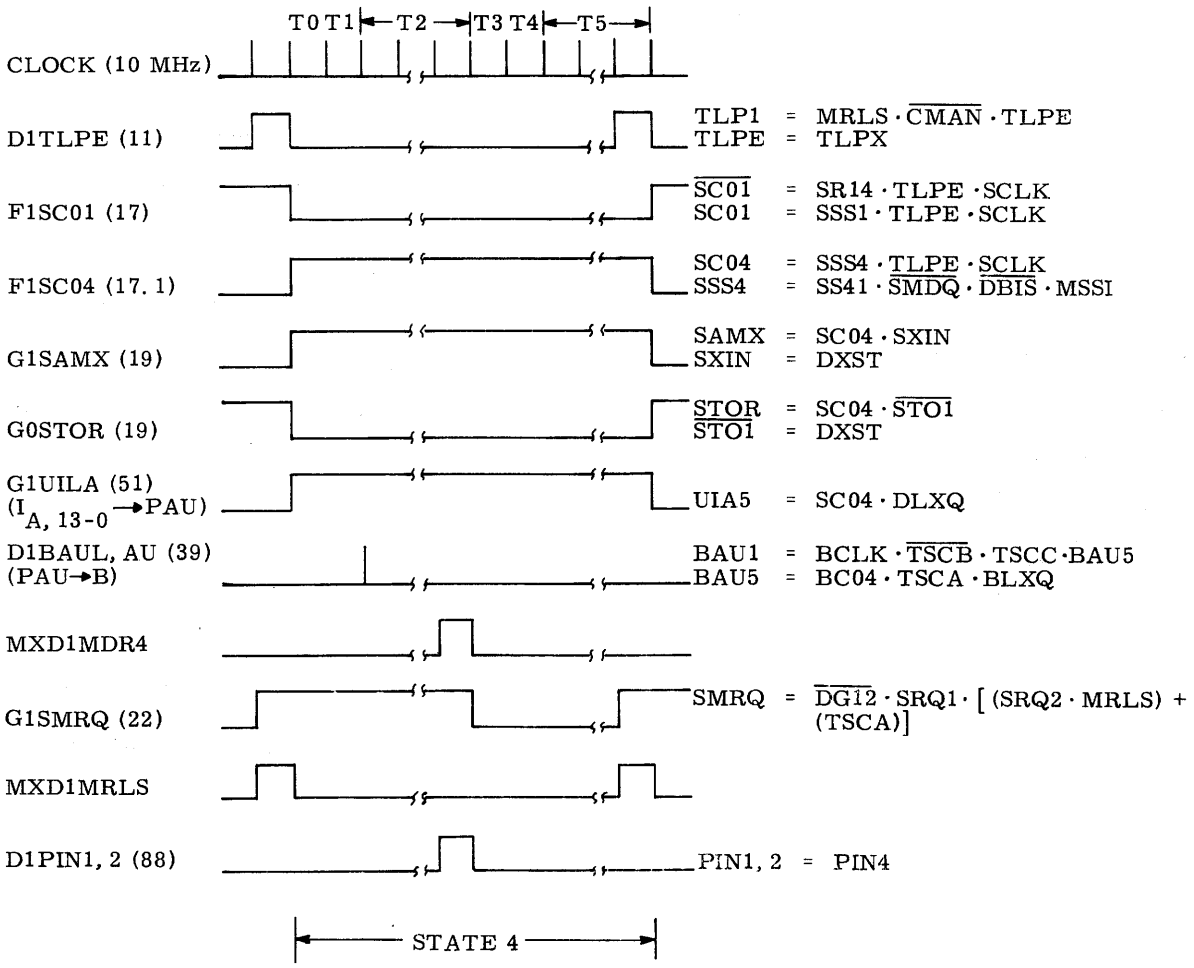
The LXK command is executed during Sequence Control State 4 (SC04). During State 4, memory is addressed from I_{17-15} (G1SAMX). The contents of $I_{A,13-0}$ are gated, via the Adder Unit (G1UILA), to the B Register (D1BAUL,AU). From B, this value is stored in the addressed X cell to complete the LXK command.

Word Times.	2 (S1,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory X	23 Zeros 15 | 14 Z 0

COMMAND CHARACTERISTICS



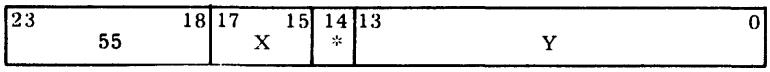
LXK BLOCK DIAGRAM



LXK TIMING DIAGRAM

MPY - MULTIPLY

FULL OPERAND



Z = f (X,*,Y)

MPY forms the product of the contents of core cell Z (multiplicand) and the contents of the Q Register (multiplier). The contents of the A Register are added algebraically to the least significant half of the product. Thus, with proper scaling, it is possible to form the value QZ+A. The result is stored in A_{23-0} and Q_{22-0}, with the most significant half in A. Bit 23 of Q is set to "zero" and is not a part of the product. The sign of A (A_{23}) applies to the entire product. Either positive or negative (2's complement) values may be multiplied, with the correct product, positive or negative (2's complement), contained in A and Q.

BINARY MULTIPLICATION

For a better understanding of the following description, two important steps of multiplication must be recalled: two numbers are multiplied by a series of (1) additions, and (2) shifts. The following examples illustrate that the mechanics employed for decimal numbers is true also for binary numbers.

Decimal		Binary		
3		(multiplicand)	00011 = 3	
x15		(multiplier)	01111 = 15	
3	Add		00011	Add & Shift
3	Add		00011	Add & Shift
3	Add		00011	Add & Shift
3	Add		00011	Add & Shift
3	Add & Shift		00000	Shift
3	Add		0000101101 = 45	
45				

As illustrated in the example of binary multiplication, the following rules are obeyed:

(1) If a multiplier digit is a "one", the multiplicand is added to obtain a partial product. Then, a shift one place to the left occurs for the next partial product.

(2) If a multiplier digit is a "zero" , the multiplicand is not added but a shift one place to the left occurs for the next partial product.

(3) The partial products are then added to obtain the result.

Multiplication within the 4022 Arithmetic Unit utilizes the "string" concept of multiplication to reduce the number of additions required, thereby increasing the speed of the multiply cycle. Using the "string" concept, it is not necessary to form the partial product for each bit when the multiplier contains two or more successive "one" bits.

Non-Indexed Word Times.	S1,S3,S4,S5 8.9 - 12.1 μs
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	Most Significant Part of Product
Q_{23-0}	23 0 22 Least Significant Part of Product 0
P_{14-0}	C(P) +1
F1WPMT	
F1UOFL	Set if Overflow Occurs*
F1ETST	
J_{4-0}	37_8
Memory Z	

*The Overflow flip-flop is set only if the multiplier and multiplicand are 40000000_8 and the contents of the A Register are not negative.

COMMAND CHARACTERISTICS

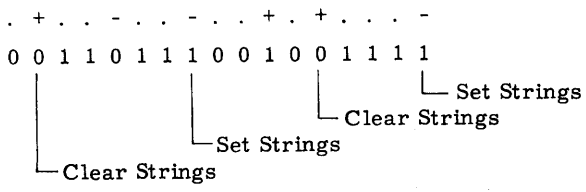
The following observations will be of assistance in understanding the "string" concept of multiplication. First, a binary number such as, 0100001000 may be written as $2^8 + 2^3$. Observe also, that a binary number such as 000111111 may be written $2^6 - 2^0$ (i.e., 001000000 minus 1 = 000111111). Thus, when a binary multiplier has a number of successive "one" bits, it is not necessary to form the partial product for each "one" bit, but, as in the case above, merely subtract 2^0 from 2^6. Now consider a binary number such as 0011011, which can be written $2^5 - 2^2 - 2^0$, since it is the same as 0011111 reduced by 2^2. A number such as this contains a "string" of "ones" with an included "zero". Note that the effect of an included "zero" is a subtraction of the corresponding power of two. This effect holds for more than one included "zero". Using this method, the least number of powers of 2 representing a binary number may be determined. For example:

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	0	1	1	1	0	0	1	0	0	1	1	1	1

$= 2^{15} - 2^{12} - 2^9 + 2^6 + 2^4 - 2^0.$

The following example is provided as an aid in determining when to add the multiplicand to the partial product and shift, when to subtract the multiplicand from the partial product and shift, and when to just shift the partial product. This example illustrates a multiplier with

various combinations of "ones" and "zeros". A minus sign (-) above a digit indicates that a subtraction should be performed; a plus sign (+) above a digit indicates that an addition should be performed; and a dot (.) above a digit indicates that only a shift is required.



As the multiplier is shifted right from least toward more significant bits, each bit must be examined and the decision made to add, subtract, or shift.

These observations can best be summarized in terms of the concept of a string of "ones". Two or more adjacent "ones" constitute a string of "ones", and as noted above, the least significant "one" of a string is interpreted as $\bar{1}$, while the zero to the left of the most significant "one" is interpreted as $\overset{+}{0}$, for example, as in $0\dot{0}1\dot{1}1\dot{1}1\dot{1}$. A string may include isolated "zeros", i.e., "zeros" each of which is flanked by "ones", such "zeros" are interpreted as $\bar{0}$, for example, as in $0\dot{0}1\dot{1}0\dot{1}0\dot{1}1\dot{1}$. Isolated "ones", i.e., "ones" each of which is flanked by "zeros" and is not a member of a string, are interpreted as $\overset{+}{1}$, for example, as in $0\dot{0}\overset{+}{1}0\dot{1}0\dot{0}$. Thus, working from right to left, as a string of "ones" is met, a subtraction must take place, and only subtraction can take place within a string; in exiting from a string, an addition takes place and only additions can take place outside a string. Finally, note that the above discussion shows that, when working from right to left, a string is not initiated until at least two adjacent "ones" occur and does not end until at least two adjacent "zeros" occur.

COMMAND DESCRIPTION

Fig. MPY. 1 contains a basic flow chart of the Sequence Control States required to execute the MPY command and lists the basic functions performed within each Sequence State. Block diagrams of States 3, 4, and 5 are contained in Fig. MPY. 2.

Sequence State 1

The MPY command is "fetched" from memory during a normal State 1. At the end of State 1, the MPY command is contained in the I Register. Following State 1, a non-indexed MPY command enters Sequence State 3.

Sequence State 3

A timing diagram with logic equations for State 3 is contained in Fig. MPY. 3. During State 3, memory is addressed from $I_{A,13-0}$ (D1SAMI) and the contents of memory location Z (multiplicand) are gated to the B Register. From the B Register, the multiplicand is gated to the Adder (D1UBAU) and from the Adder to the I Register (D1IUIU, D1IUIL, D1IULI). At Time 3 of State 3, the Execute MPY or DVD flip-flop (F1XMDV) is set to indicate that the I Register does not contain the instruction, and to provide control for the MPY operation. The MPY command leaves the Execute flip-flop (F1XEXC) reset and the DVD command sets F1XEXC. Therefore, when F1XMDV is set and F1XEXC is reset, the MPY command is specified. Following State 3, State 4 is entered.

Sequence State 4

Sequence State 4 is used to bring the multiplier from the Q Register (memory cell 10_8) to the B Register and to perform the actual arithmetic operation. To provide these functions, the duration of State 4 is extended by entering Time 6 Envelope.

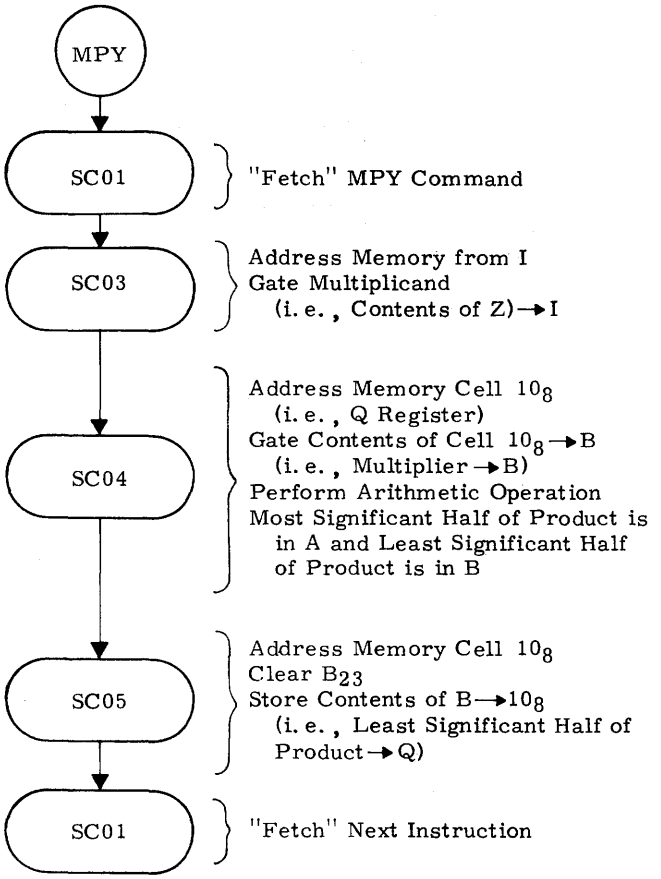


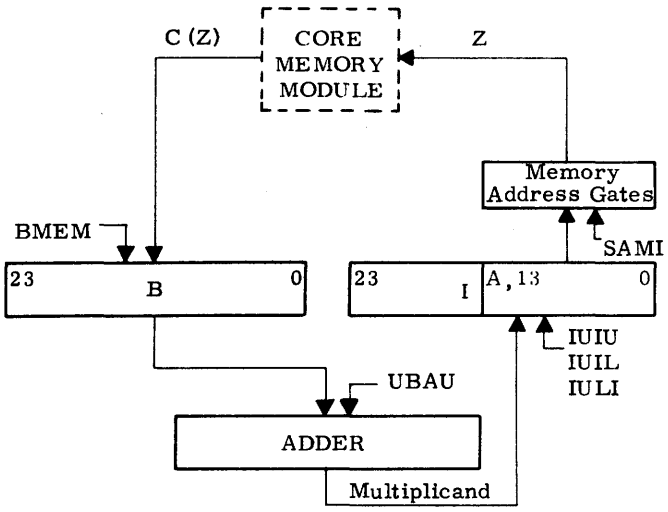
Fig. MPY. 1 MPY Flow Chart

For ease of understanding, a detailed flow chart of State 4 is contained in Fig. MPY. 4. The logic elements performing the functions listed in the flow chart are provided, along with the logic equations and the logic sheet numbers on which these elements may be found, in Table MPY. 1.

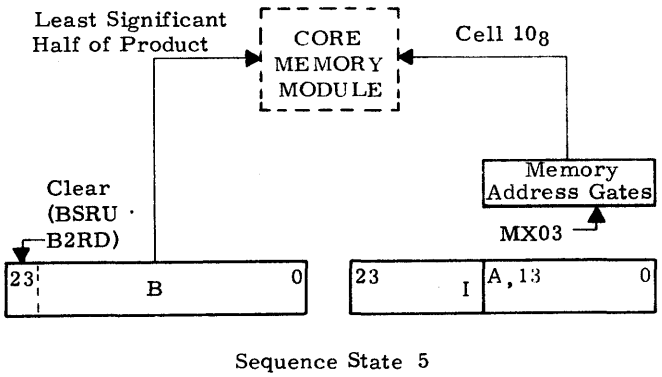
An example of the arithmetic operation is provided in Table MPY. 2. For simplicity, nine-bit registers are used. The timing diagram of State 4 contained in Fig. MPY. 5, illustrates the timing associated with this example. Using these aids, little difficulty should be encountered in determining the operation of the Arithmetic Unit for any multiplier and multiplicand values.

During the first 1. 6 microseconds of State 4, memory cell 10_8 is addressed (G0MX03) and the multiplier is gated from the Q Register (cell 10_8) to the B Register.

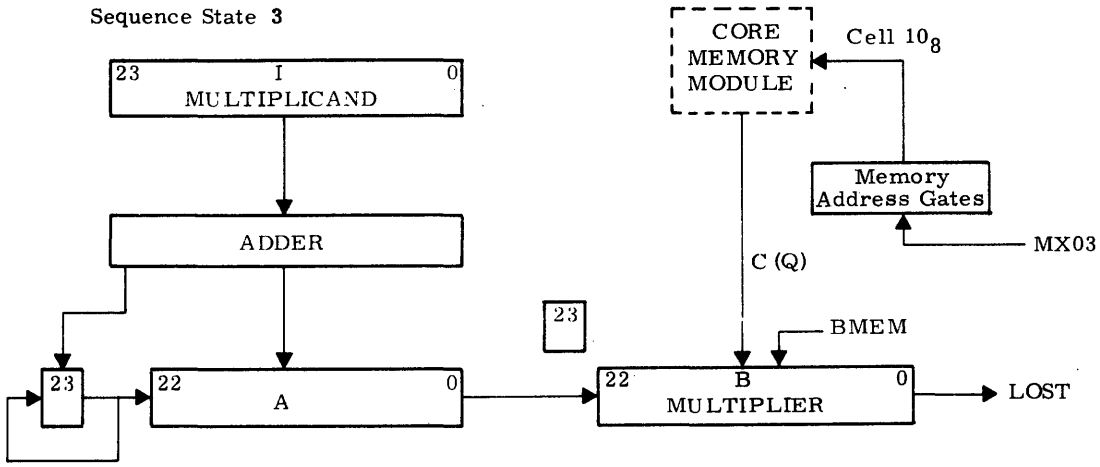
At Time 3, the Delay Time Counter is preset to 30_8 and the J Counter is preset to 7_8. The J Counter is incremented as each bit of the multiplier is considered to determine the final product. When J is equal to 37_8, all 23 bits of the multiplier have been considered and the fix-up cycle, if required, is entered to end the MPY cycle. The Delay Time Counter is used to provide control for the addition or subtraction and shift required with each multiplier bit.



Sequence State 3



Sequence State 5



Sequence State 4

Fig. MPY. 2 MPY Block Diagram

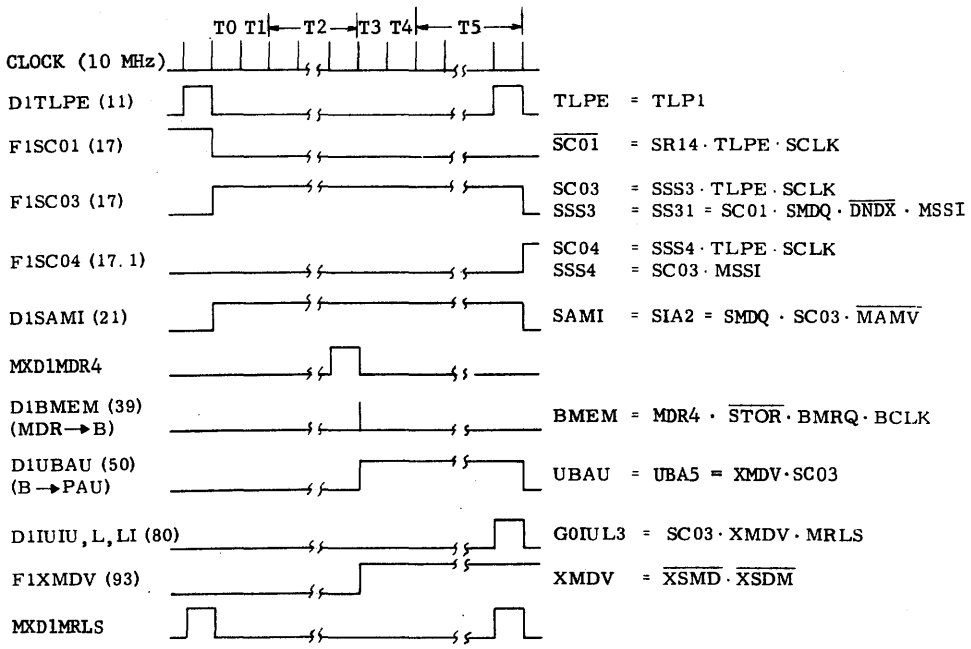


Fig. MPY. 3 MPY Sequence State 3 Timing

At Time 5 (Memory Release), the J Counter is allowed to increment for the first time. Time 6 is then entered and the actual computation is performed.

As described in the basic multiply discussion, the two least significant bits of the multiplier (B_1 and B_0) and the status of the Strings flip-flop (F1XSTG) are used to determine if the multiplicand is added or subtracted from the partial product in the A Register and then the partial product is shifted or if only a shift of the partial product is required.

After the J Counter is equal to 37_8, the status of the Strings flip-flop and the sign bit of the multiplier (B_{23}) is examined to determine if an addition or subtraction is required to "fix-up" the product. If B_{23} is a "zero" and the Strings flip-flop is set, then an addition must be performed to end the within strings cycle. If B_{23} is a "one" and the Strings flip-flop is reset, then the multiplier was negative and an additional subtraction is required. Following this "fix-up", if required, State 4 is ended. Therefore, at the end of State 4, the most significant half of the product is in the A Register and the least significant half of the product is in the B Register.

Sequence State 5

During State 5, the least significant half of the product contained in the B Register is stored in the Q Register (cell 10_8). A timing diagram and logic equations for State 5 are contained in Fig. MPY. 6.

At the first clock pulse of State 5, B_{23}, if a "one", is reset to "zero". Memory cell 10_8 is addressed (G0MX03) and the contents of the B Register are stored in cell 10_8.

During State 1 of the command following MPY, the Multiply/Divide flip-flop (F1XMDV) is reset.

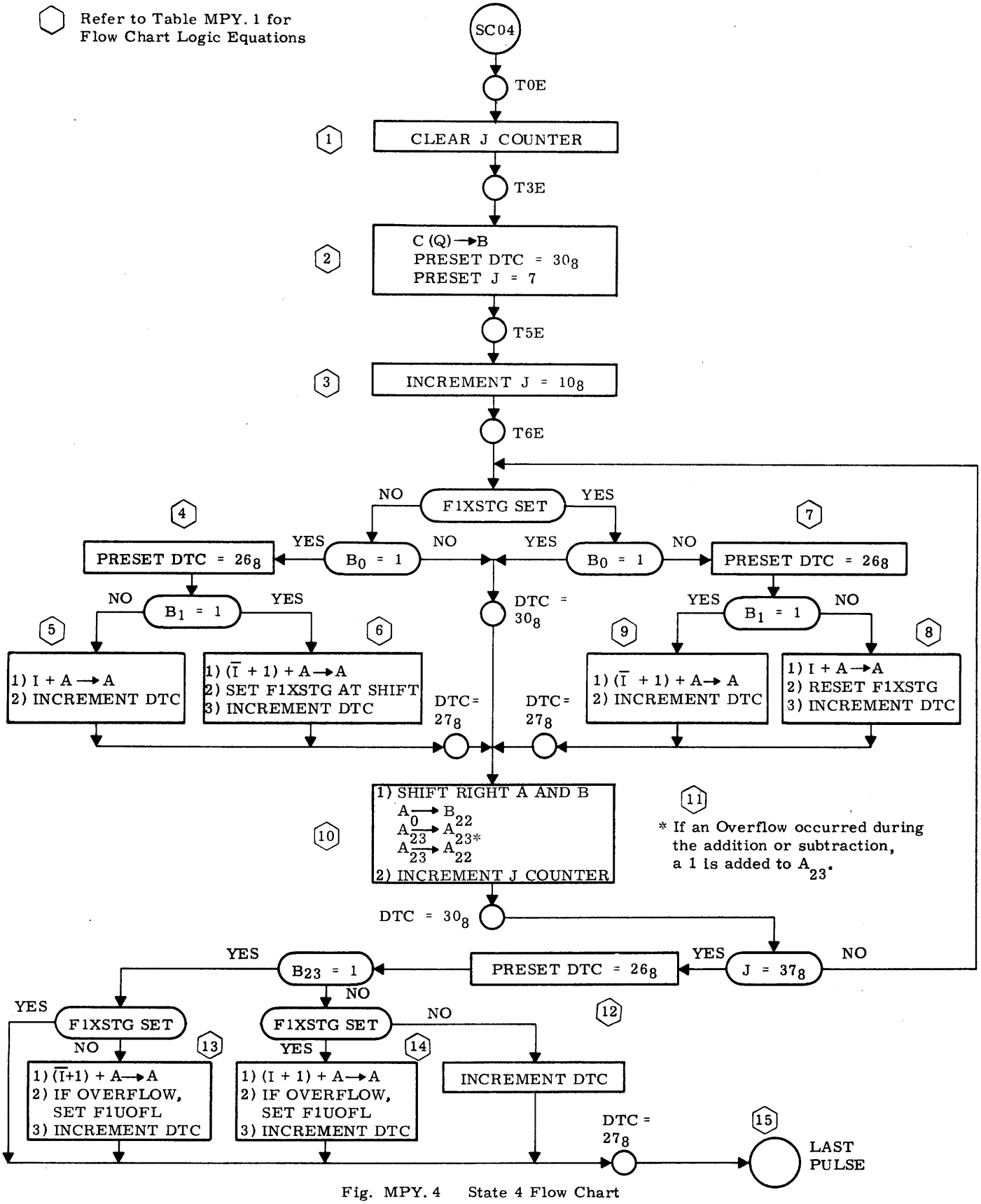


Fig. MPY. 4 State 4 Flow Chart

(1) Clear J Counter:

D0JJE0 (70) = JCK1· $\overline{\text{TSCA}}$· $\overline{\text{TSCB}}$· JS4G
G1JS4G (70) = SC04· DMD1

(2) Preset Delay Time Counter to 30_8:

G1TPAF (12) = TP30 + TD30
G0TP30 (12) = TSCB· TSCA· DMGT· TSCC

Preset J Counter to 7:

G1JP07 (70) = TP30· XMDV

(3) Increment J Counter:

N1JINC (71) = G0JINC = XMDV· TT5E· MRLS· JC04

(4) Preset Delay Time Counter to 26_8:

G1TP26 (12) = TD30· $\overline{\text{JE07}}$· XMDV· G0TP26
G0TP26 (12) = $\overline{\text{XSTG}}$· BR00· DMPY

(5) Add I to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA3 = $\overline{\text{JE37}}$· UIAA· $\overline{\text{XSTG}}$· BR01· BR00
D1AAUU, L (62) = AAU2 = $\overline{\text{ECLK}}$· TE26· AMDV· JE37

(6) Subtract I from A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UINA (51) = UIN2 = UC04· $\overline{\text{XMDV}}$· $\overline{\text{JE37}}$· BR01· BR00· $\overline{\text{XSTG}}$· $\overline{\text{XEXC}}$
G1UENC (51) = UIN2
D1AAUU, L (62) = AAU2 = $\overline{\text{ECLK}}$· TE26· AMDV· $\overline{\text{JE37}}$

Set Strings Flip-Flop at Shift of A and B:

F1XSTG (94) = XDST· XBSR· ICK7
G1XDST (94) = BR01· BR00

(7) Preset Delay Time Counter to 26_8:

G1TP26 (12) = TD30· $\overline{\text{JE07}}$· XMDV· G0TP26
G0TP26 (12) = $\overline{\text{BR00}}$· XSTG· DMPY

(8) Add I to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA2 = $\overline{\text{BR00}}$· $\overline{\text{BR01}}$· XSTG· JE37· UIAA
D1AAUU, L (62) = AAU2 = $\overline{\text{ECLK}}$· TE26· AMDV· JE37

Reset Strings Flip-Flop at Shift of A and B:

$\overline{\text{F1XSTG}}$ (94) = $\overline{\text{XNST}}$· $\overline{\text{XBSR}}$· ICK7
G1XNST (94) = BR01· BR00

(9) Subtract I from A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UINA (51) = UIN3 = $\overline{\text{UC04}}$· XMDV· $\overline{\text{JE37}}$· BR01· BR00· XSTG· $\overline{\text{XEXC}}$
G1UENC (51) = UIN3
D1AAUU, L (62) = AAU2 = $\overline{\text{ECLK}}$· TE26· AMDV· JE37

(10) Shift Right A and B:

G0ABSR (63) = $\overline{\text{JE37}}$· DMPY· TE27· TT6E + TT6E· $\overline{\text{JE37}}$· TE30· ABSG
ABSG = ($\overline{\text{XSTG}}$· BR00) + (XSTG· BR00)

Gate A_0 to B_{22}:

G1B22N (38) = $\overline{\text{BA00}}$· BMPY
G1B22D (38) = BMPY· AR00

Gate A_{23} to A_{23}:

G1AFNS (67) = AFNA = ANA3· AR23
N0AFNS (67) = $\overline{\text{G1AFNS}}$

Increment J Counter:

N1JINC (71) = G0JINC = $\overline{\text{JE37}}$· ABSG· TE30· TT6E + JE37· TT6E· XMDV· TEFF

(11) Add 1 to A_{23} if Overflow Occurred During (5), (6), (8), or (9):

F1AFNP (66) = TE26· UFL1· BCLK
G1UFL1 (48) = W23C· U22C· $\overline{\text{U23S}}$· ADIV + U23S· U22C· $\overline{\text{U23C}}$· ADIV

(12) Preset the Delay Time Counter to 26_8:

G1TP26 (12) = TD30· $\overline{\text{JE07}}$· XMDV· JE37

(13) Subtract I from A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UINA (51) = UIN4 = JE37· XSTG· UIAA· BR23
G1UENC (51) = UIN4
D1AAUU, L (62) = ECLK· TE26· AMDV· $\overline{\text{JE37}}$ + JE37· TE27· AMDV· ECLK

If Overflow Occurs During Subtraction, Set F1UOFL:

F1UOFL (54) = UFLB
G0UFLB (48) = JE37· TE27 ECLK· UFL1
G1UFL1 (48) = W23C· U22C· $\overline{\text{U23S}}$· ADIV + U23S· U22C· $\overline{\text{U23C}}$· $\overline{\text{ADIV}}$

(14) Add I to A:

D1UAAU (52) = UAA4 = XMDV· UC04
D1UILA, AU (51) = UIA4 = $\overline{\text{UIAA}}$· JE37· XSTG· BR23
D1AAUU, L (62) = $\overline{\text{AAU2}}$ = ECLK· TE26· AMDV· JE37 + JE37· TE27· AMDV· ECLK

If Overflow Occurs During Addition, Set F1UOFL:

F1UOFL (54) = UFLB
G0UFLB (48) = JE37· TE27· ECLK· UFL1
G1UFL1 (48) = W23C· U22C· $\overline{\text{U23S}}$· $\overline{\text{ADIV}}$ + U23S· U22C· $\overline{\text{U23C}}$· $\overline{\text{ADIV}}$

(15) Enable Last Pulse:

D1TLPE (11) = TLP3 = JE37· AMDV· TE27

Table MPY. 1 State 4 Flow Chart Logic Equations

	S										
I =	0	0	0	0	0	1	1	1	1	= +15 =	MULTIPLICAND
B =	0	0	0	0	0	1	1	0	0	= +12 =	MULTIPLIER
A =	0	0	0	0	0	0	0	0	0	= 0	

J COUNT	DTC	A REGISTER S 7 6 5 4 3 2 1 0	B REGISTER S 7 6 5 4 3 2 1 0	I REGISTER S 7 6 5 4 3 2 1 0	XSTG	ACTION
10	30	0 0 0 0 0 0 0 0 0	0 0 0 0 0 1 1 0 0	0 0 0 0 0 1 1 1 1	0	T5E of S4
11	30	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 1 0		0	Shift
12	30	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1		0	Shift
12	26	1 1 1 1 1 0 0 0 1			0	SUB
13	27	1 1 1 1 1 1 0 0 0	0 1 0 0 0 0 0 0 1		1	Set XSTG Shift
14	30	1 1 1 1 1 1 1 0 0	0 0 1 0 0 0 0 0 0		1	Shift
14	26	0 0 0 0 0 1 0 1 1			1	ADD
15	27	0 0 0 0 0 0 1 0 1	0 1 0 1 0 0 0 0 0		0	Clear XSTG Shift
16	30	0 0 0 0 0 0 0 1 0	0 1 1 0 1 0 0 0 0		0	Shift
17	30	0 0 0 0 0 0 0 0 1	0 0 1 1 0 1 0 0 0		0	Shift
20	30	0 0 0 0 0 0 0 0 0	0 1 0 1 1 0 1 0 0		0	Shift

↑ Equivalent to J = 37_8 When Using 24-Bit Registers

↑ = 180_{10}

Table MPY. 2 MPY Example (+15 x +12)

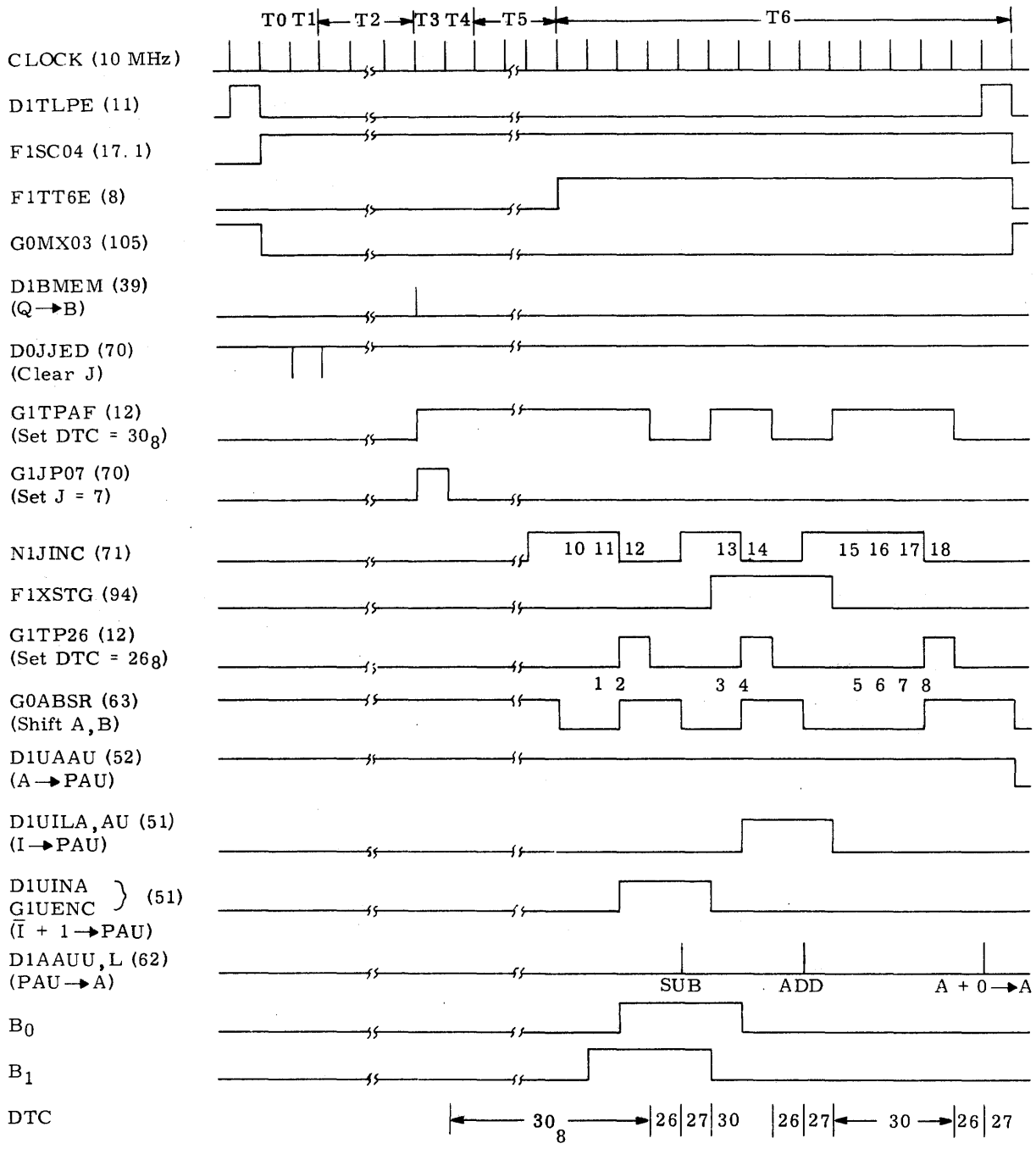


Fig. MPY.5 MPY Sequence State 4 Timing Example (+15 x +12)

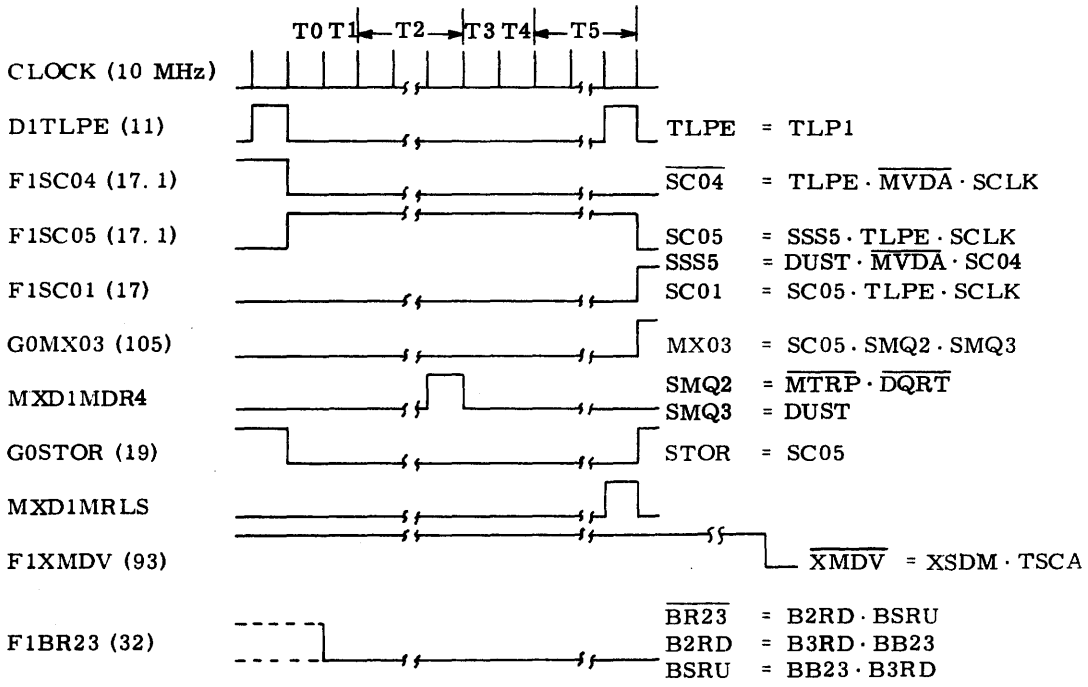


Fig. MPY. 6 MPY Sequence State 5 Timing

NOP - NO OPERATION

FULL OPERAND	23								0
	2	6	2	0	0	0	0	0	

NOP transfers program control to the next sequential location (P+1). No operation is performed. NOP is useful as a programming tool to replace deleted commands or to provide room for the insertion of new commands in the program.

NOP is an INX (Increment X) command that specifies incrementing of index cell 2 by 0. Sequence States 1, 2, and 4 are required to "fetch" and "execute" the NOP command. Refer to the INX command description for details of the operation of the NOP command.

Word Times.	3 (S1,S2,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory X	

COMMAND CHARACTERISTICS

ORA - LOGICAL OR TO A

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
21	X	*	Y

Z = f (X, *, Y)

ORA performs the logical OR of the contents of core cell Z with the contents of the A Register. Each bit of Z is compared with the corresponding bit of A. When either or both is a "one", a "one" is placed in that position of A. When both bits are "zero", that position of A is not changed.

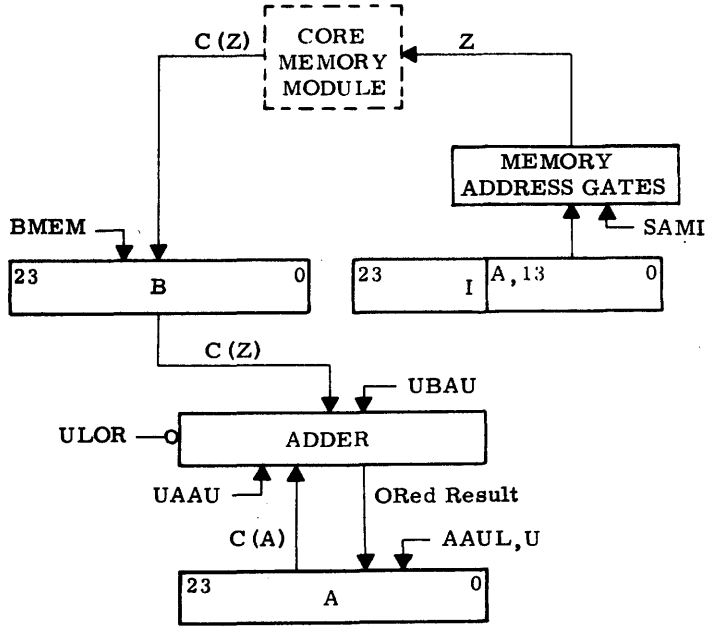
A non-indexed ORA command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The contents of memory location Z are gated to the B Register by D1BMEM during the Clock pulse of Memory Data Ready (MXD1MDR4). From B, the contents of memory location Z are gated to the Adder Unit (D1UBAU). At the same time, the contents of the A Register are gated to the Adder Unit. Also, the control signal (D0ULOR) to enable the Logical OR function of the Adder Unit is applied to the Adder Unit. The result is then gated back to the A Register (D1AAUL,U) to complete the ORA execution.

To exemplify the ORA comparison, consider the following; 4 bits are used for simplicity:

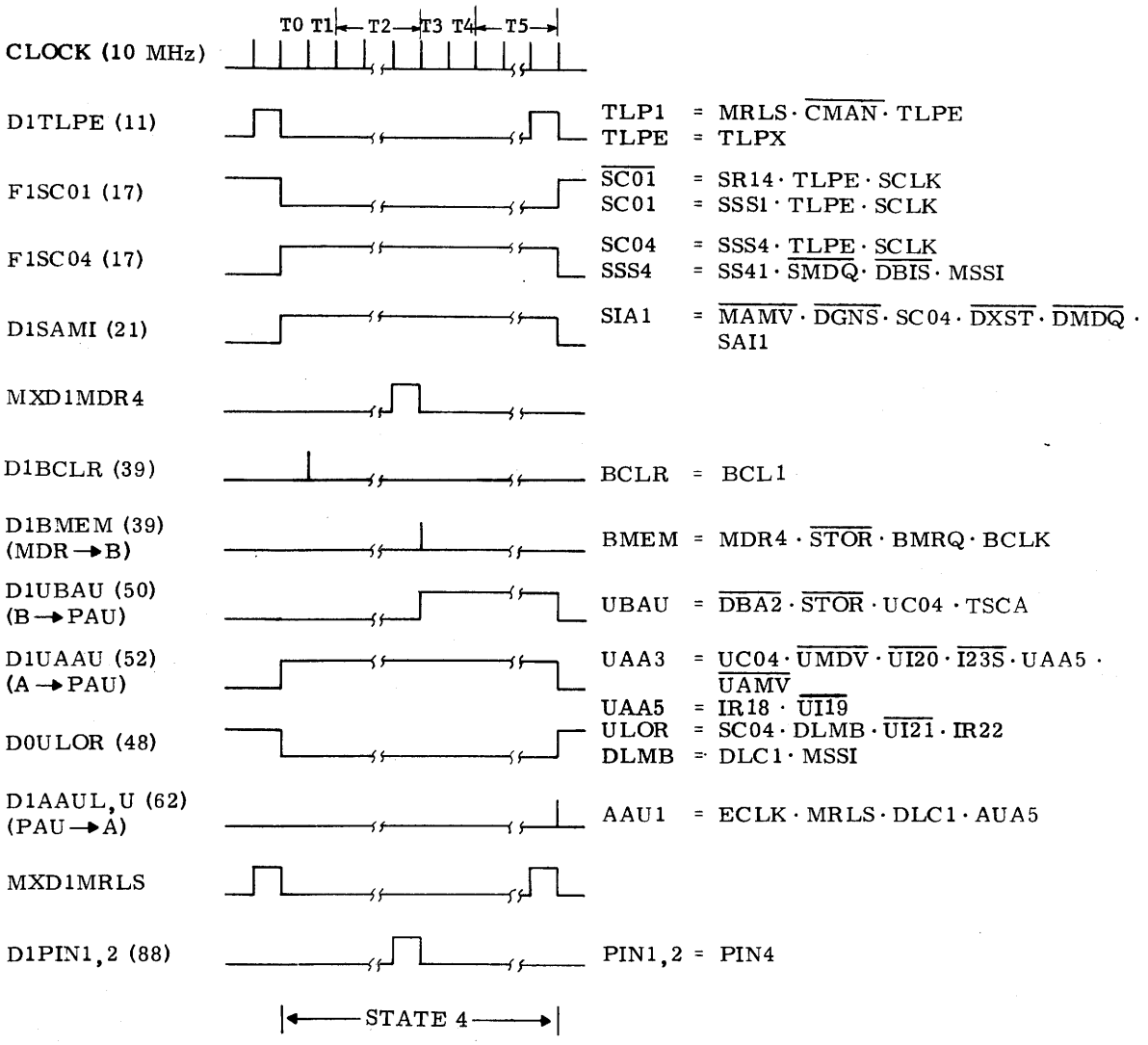
Contents of A Register	0011
Contents of B Register (Z)	0101
Result Placed in A Register	0111.

Non-Indexed Word Times.	2 (S1,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	$C(Z_{23-0})$ ORed with $C(A_{23-0})$
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

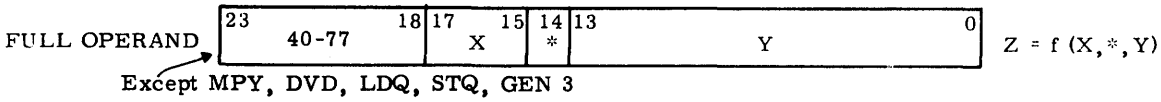


ORA BLOCK DIAGRAM



ORA TIMING DIAGRAM

QSI - QUASI COMMANDS



Quasi commands provide operations not included in the hardware. These commands supply the programmer with a mnemonic which allows the running program to be linked to a subroutine. Quasi commands are identified by operation codes 40_8 through 77_8 if the command is not an MPY, DVD, LDQ, STQ, or GEN 3. Quasi commands store the operand portion of the command (Z) in memory location 2 and the next instruction is "fetched" from the memory location specified by the operation code (bits 23 through 18) of the Quasi command. The command located at the specified operation code address is normally an SPB which branches to a software subroutine associated with the Quasi command. Some Quasi commands, however, contain an instruction other than SPB in the location specified by the operation code. These instructions include AKA, LDK, and SKA. These instructions will contain an ADD, LDA, and SUB command with an operand address equal to 2, respectively, in the location specified by the operation code. The combination of the Quasi command storing the value Z in cell 2 and then executing the instruction in the operation code address completes the operation to be performed for these commands.

Although the function to be performed by the Quasi command depends on the instruction located in the operation code address, certain operation codes are normally reserved to perform specific Quasi functions, thereby, providing program compatibility with other GE-PAC computers. The following is a list of these commands:

Quasi Operation Code	Quasi Command Mnemonic
40	LDK
41	DLD
46	REL
47	AEL
50	SKA
51	DAD
52	LDI
53	STI
56	RBL
57	ABL
60	AKA
61	DSU
62	OOM
63	DST
70	FAD
71	FSU
72	FMP
73	FDV
74	FIX/FLO/FMS
75	DVDM

Except for AKA, LDK, and SKA, Quasi command cannot be interrupted following execution. That is, the command located in the operation code address must be executed before an interrupt can occur. Commands within the Quasi subroutine, however, may be interrupted following execution provided the command is normally interruptable following execution. If the optional Memory Protect function is enabled relative addressed instructions within the Quasi subroutine are not subject to the protect criteria. Commands that are located within the Quasi subroutine and not relative addressed are subject to the established protect criteria.

To better understand the operation of Quasi commands, consider the following examples.

Example 1 - FAD 70X*Y

The Floating Add command (FAD) adds the single length floating point number contained in memory location Z to the floating point number contained in the A Register.

The FAD command is "fetched" from memory and since the Op Code is 70_8, it is decoded as a Quasi command. If the FAD command is relative addressed and/or indexed, the computation is performed and the result (Z) is gated to $I_{A,13-0}$. State 4 is then entered and index cell 2 is addressed. The contents of $I_{A,13-0}$ (Z) are then stored in index cell 2. The Op Code portion of the FAD command (70_8) is gated from I_{23-18} to I_{5-0}. The P Register is not incremented during this command execution. State 1 is then entered and memory is addressed from the I Register and the command located in cell 70_8 is "fetched". Since the FAD function cannot be completed in one instruction, this will be an SPB command. During execution of the SPB command, the P Register will be incremented by 1. This provides a value of P equal to the location of the FAD command plus 1. This value is stored in index cell 1 and will be used to return to the running program following the completion of the subroutine required to perform the FAD operation. After storing the contents of the P Register in cell 1, the SPB command loads the P Register with the branch address specified by bits 13-0 of the SPB command. This will transfer program control to the subroutine that will perform the FAD operation.

The first instruction of this subroutine will contain an STX command to save the location of the running program that was stored in cell 1. The rest of the subroutine will perform the operations required to obtain the result of floating point addition. This routine will use the contents of index cell 2 to obtain the address (Z) of one of the operands.

The last instruction of the FAD subroutine will be an LDP or LPR command. This LDP or LPR command will restore the P Register with the location of the running program (i.e., with the value stored in cell 1 by the SPB command and saved in another location by the

STX command. This LDP or LPR command will also clear the Quasi flip-flop (F1XQUA) if the Memory Protect Mode is enabled. F1XQUA is used by the optional memory protect circuitry to allow instructions within the Quasi subroutine to be fetched and to allow relative addressed instructions to be executed without trapping.

Example 2 AKA - 60X0K

During State 1, the AKA command is "fetched" from memory. Since the Op Code of the AKA command is 60_8, it is decoded as a Quasi command. If the AKA command is indexed, indexing occurs and the result (Z) is gated to $I_{A,13-0}$. State 4 is then entered and index cell 2 is addressed. The contents of $I_{A,13-0}$ are stored in cell 2. The Op Code portion of the AKA command (60_8) is gated from I_{23-18} to I_{5-0}. The P Register is not incremented during this command. State 1 is then entered. Memory is addressed from the I Register and the command in location 60_8 is "fetched". This command will normally be an ADD (11000002). This ADD command will sum the contents of the A Register with the contents of index cell 2 (value Z). During execution of this ADD command, the P Register will be incremented and the Quasi flip-flops (F1XQUA,B) will be cleared. Program control will then return to the location following the AKA command.

In this example, an SPB command was not located in the Quasi Op Code address because only one instruction was required, in addition to the AKA command, to complete the AKA operation.

HARDWARE OPERATION

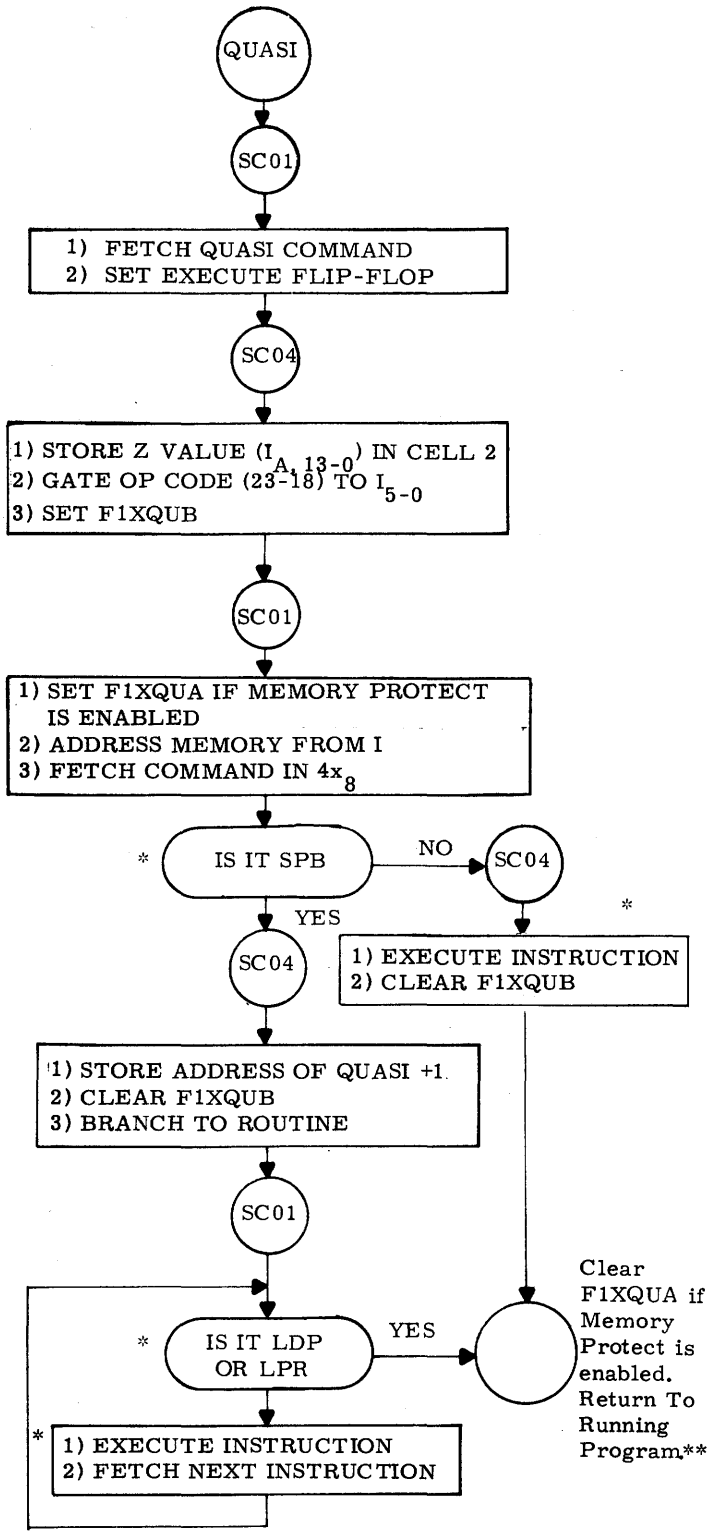
Although there are many Quasi commands that may be executed, the hardware operation for these commands is the same. Therefore, this discussion will describe only these functions.

Fig. QSI. 1 contains a flow chart of the basic hardware functions performed. Note that the only concern about the subroutine addressed by the Quasi command is to determine when the Quasi flip-flop F1XQUA is reset if Memory Protect is enabled. This flip-flop is used by the optional Memory Protect logic, when enabled, to determine if a violation exists. Quasi commands within the subroutine are not subject to trapping if the command is relative addressed. A more detailed discussion of Memory Portect is contained in the Options (OPT) portion of this Arithmetic Unit Description.

Fig. QSI. 2 contains a block diagram of the functions performed by the Sequence States of the Quasi command. A timing diagram, including logic equations, is contained in Fig. QSI. 3.

During State 1, the Quasi command is "fetched" from memory. At Last Pulse of State 1, the Execute flip-flop is set. Following State 1, State 4 is entered.

During State 4, index cell 2 is addressed (G0MX01). $I_{A,13-0}$ (Z) is gated to the Adder and from the Adder to the B Register and stored in location 2. I_{23-18} (Quasi Op Code) is gated to I_{5-0}. The Quasi flip-flop F1XQUB, is set. Following State 4, the second State 1 is entered.



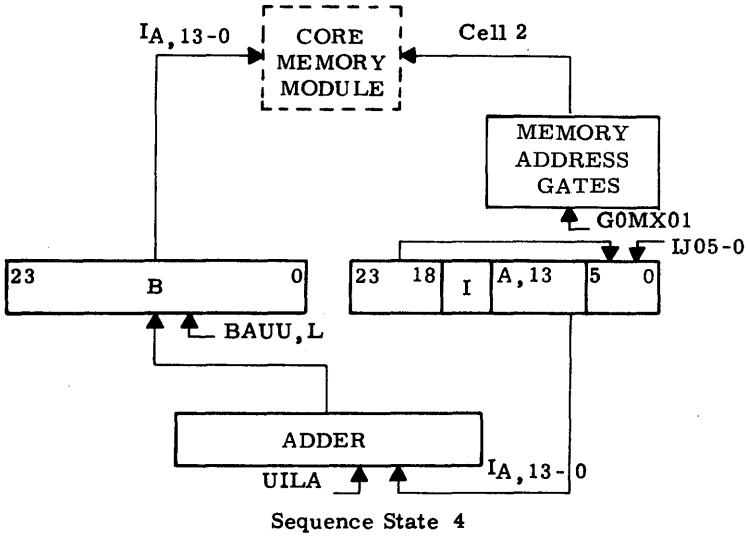
* All commands within the Quasi subroutine that are not relative addressed, are subject to memory protect trapping if enabled.

** Running program starts at the location of the Quasi command plus 1.

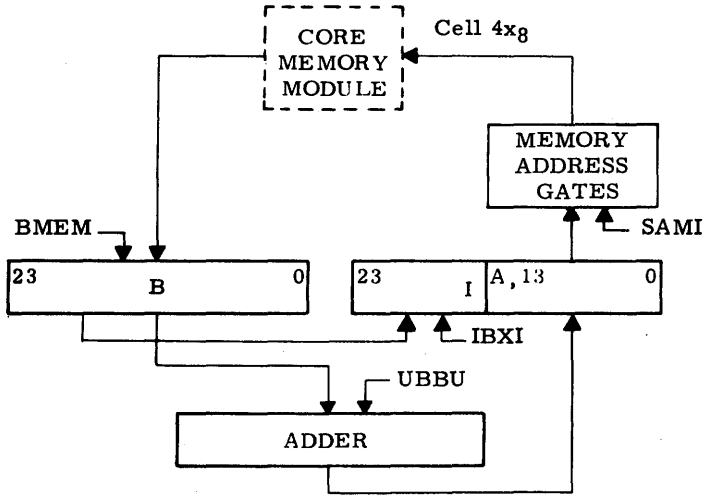
Fig. QSI. 1 Quasi Flow Chart

During this State 1, F1XQUA is set if Memory Protect is enabled. Memory is addressed from the I Register and the instruction located in the memory cell specified by the Quasi Op Code is gated to memory. This command is then executed. Normal incrementation of the P Register occurs.

If the command is not an SPB, both Quasi flip-flops (F1XQUA and F1XQUB) are cleared at Last Pulse of State 4. If the command is an SPB, F1XQUB will be cleared at Last Pulse of State 4 of the SPB command but, F1XQUA will not be cleared until an LDP or LPR command is executed.

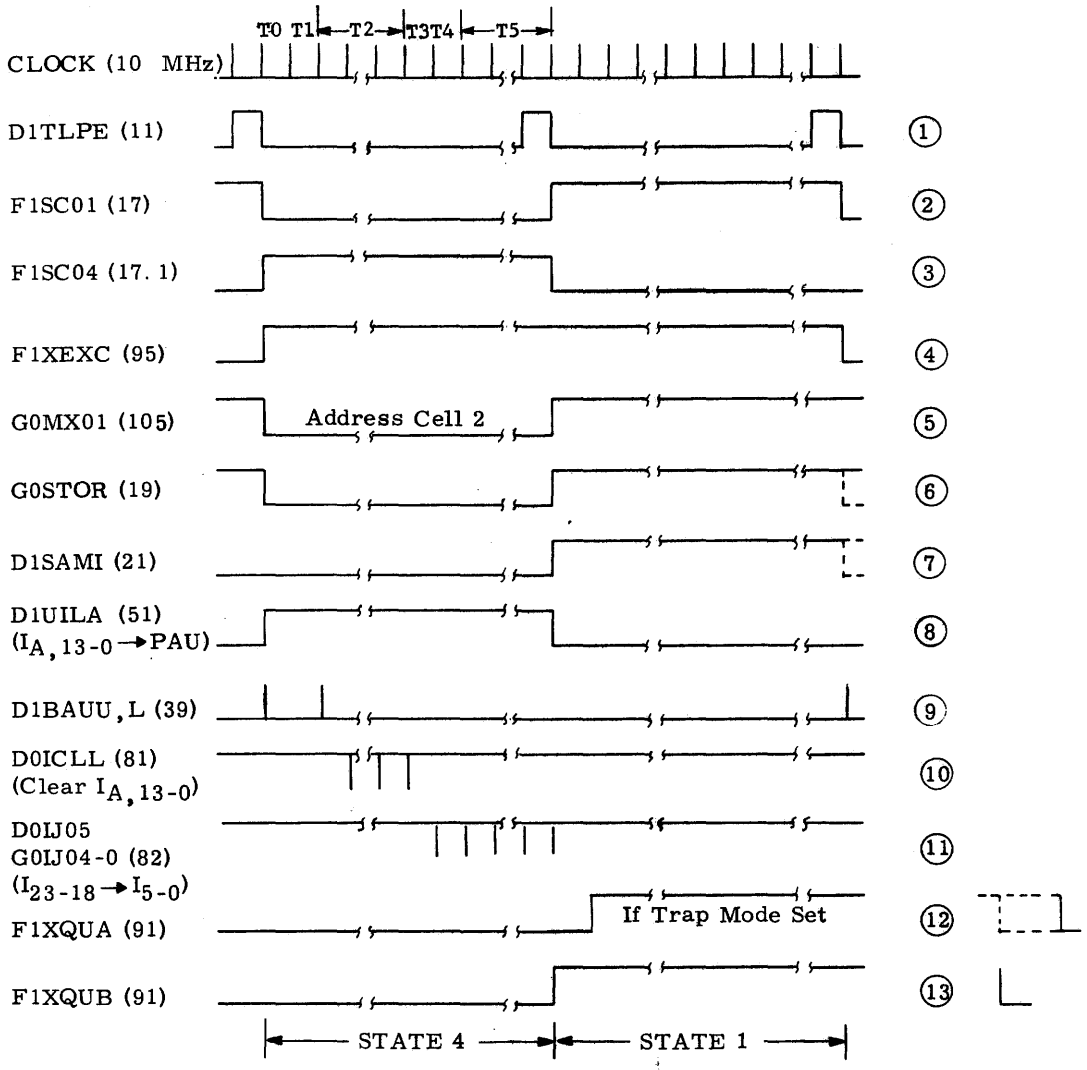


Sequence State 4



Next Sequence State 1

Fig. QSI. 2 Quasi Block Diagram



(1) D1TLPE = TLP1 = TLPE · MSSI · SC04

(2) F1SC01 = SSS1 · TLPE · SCLK

(3) F1SC04 = SSS4 · TLPE · SCLK

(4) F1XEXC = XXC1 · TLPE · ICK7

$\overline{\text{F1XEXC}}$ = XXC2 · TLPE · ICK7

(5) G0MX01 = SC04 · DQUA

(6) G0STOR = SC04 · $\overline{\text{STO1}}$

(7) D1SAMI = SIA3

(8) D1UILA = UIA5 = SC04 · DLXQ

(9) D1BAUU, L = BAU1 = BCLK · $\overline{\text{TSCB}}$ · TSCC · BAU5 + BAU2

G1BAU5 = BLXQ · $\overline{\text{TSCA}}$ · BC04

(10) D0ICLL = $\overline{\text{ISCA}}$ · ICK7 · TSCB · $\overline{\text{ICL1}}$

(11) D0LJ05 = IQAA · ISCA · ICK7 · IR23

G0LJ04-0 = IQAA · ISCA · ICK2 · IR22-18

(12) F1XQUA = DQUA · XQUB · XTRM · ECLK

$\overline{\text{F1XQUA}}$ = AUA5 · TLPE · ECLK · [(XQUB · $\overline{\text{XSPB}}$ · MTRM) + (LXPR · $\overline{\text{XSPB}}$ · MTRM)]

(13) F1XQUB = DQUA · SC04 · TLPE · ECLK

$\overline{\text{F1XQUB}}$ = AUA5 · TLPE · ECLK

Fig. QSI. 3 Quasi Timing Diagram

SPB - SAVE PLACE AND BRANCH

FULL OPERAND

23 18	17 15	14	13 0
33	X	*	Y

Z = f (X, *, Y)

SPB places the status of the Overflow (F1UOFL), Permit Automatic Interrupt (F1WPMT), Test (F1ETST), and Trapping Mode (F1MTRM) flip-flops in bit position 22 through 19, respectively, of index cell 1. Bit 18 is set if the SPB command is executed (location 24_8) due to a Watchdog Stall caused by PAI flip-flop being reset too long, thereby, inhibiting inhibitable interrupts. The contents of the P Register, plus 1*, are stored in bits 14 through 0 of index cell 1. The SPB command resets the Permit Automatic Interrupt flip-flop, inhibiting inhibitable interrupts. The Trapping Mode flip-flop is cleared if the SPB command is executed due to a trap error or if the SPB command is executed as a result of an Automatic Program Interrupt.

> * NOTE
>
> The address that is stored by the SPB command is normally the address of the SPB command plus 1. However, there are exceptions to this. If an SPB command is performed due to the use of XEC or Quasi commands, the address of the XEC or Quasi plus 1 is stored. If SPB is executed immediately following an acknowledged Automatic Program Interrupt or memory protect trap, P contains the address of current program control. In these cases only, the value saved in index word 1 is P, the first unexecuted instruction in the interrupted program (i.e., the address to which program control will return when the interrupt has been serviced).

The SPB command is executed during Sequence Control State 4 (SC04). During the next Sequence Control State 1 (SC01) memory is addressed from I and the contents of $I_{A,13-0}$ are gated to the P Register to complete the transfer of program control.

During State 4, memory cell 1 is addressed (G0MX00). The contents of the P Register (D1UPAU) and the status of the aforementioned flip-flops are gated (D1UIOU) to the Parallel Adder Unit. From the Adder, this data is gated to the B Register (D1BAUL) and from B to memory cell 1. At Time 4 of State 4, the Permit Automatic Interrupt flip-flop (F1WPMT) is cleared.

Non-indexed Word Times.	2 (S1, S4)
Interruptable Following Execution?	No

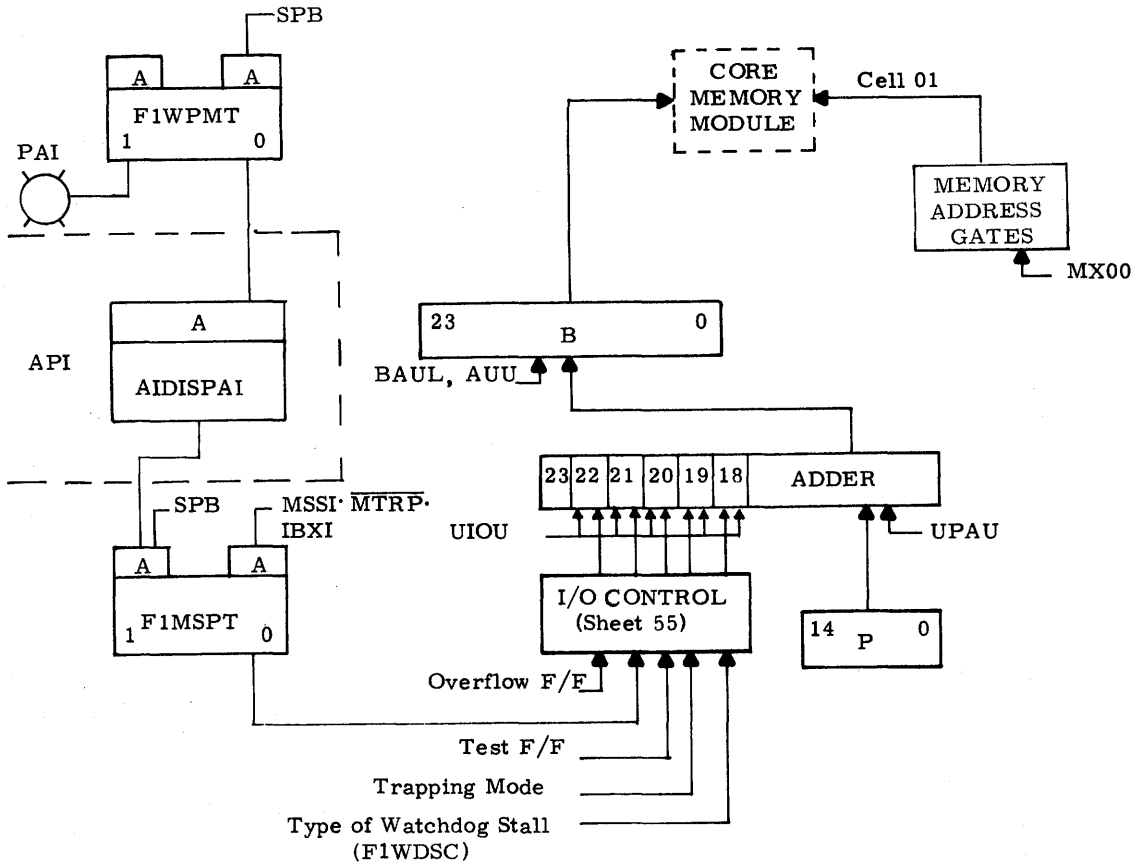
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	Z
F1WPMT	Reset
F1UOFL	
F1ETST	
F1MTRM	Reset if SPB is Executed Because of Trap Error or API
J_{4-0}	
Memory Cell 1	23 0 | 22 | 21 | 20 | 19 | 18 | 17 15 0 | 14 0 C (P) +1 *

Overflow F/F
PAI F/F
Test F/F
Type of Watchdog Trap
Trapping Mode F/F

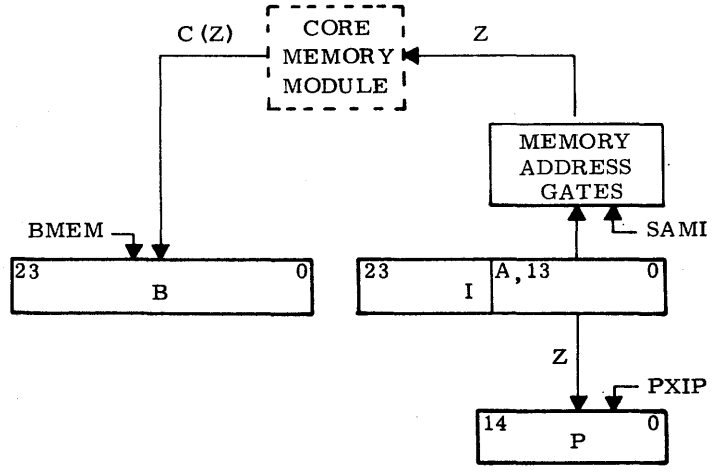
COMMAND CHARACTERISTICS

The Trapping Mode flip-flop (F1MTRM) and Watchdog error type flip-flop (F1WDSC) are cleared at Last Pulse Envelope if the SPB command was executed as the result of a trapping mode error, or if the SPB command was executed as the result of an Automatic Program Interrupt.

Following State 4, State 1 is entered to "fetch" the command located in memory cell Z. Memory is addressed from $I_{A,13-0}$ (D1SAMI) and the contents of $I_{A,13-0}$ are transferred to the P Register. In this manner, core cell Z is addressed and Z is transferred to the P Register for subsequent program control.

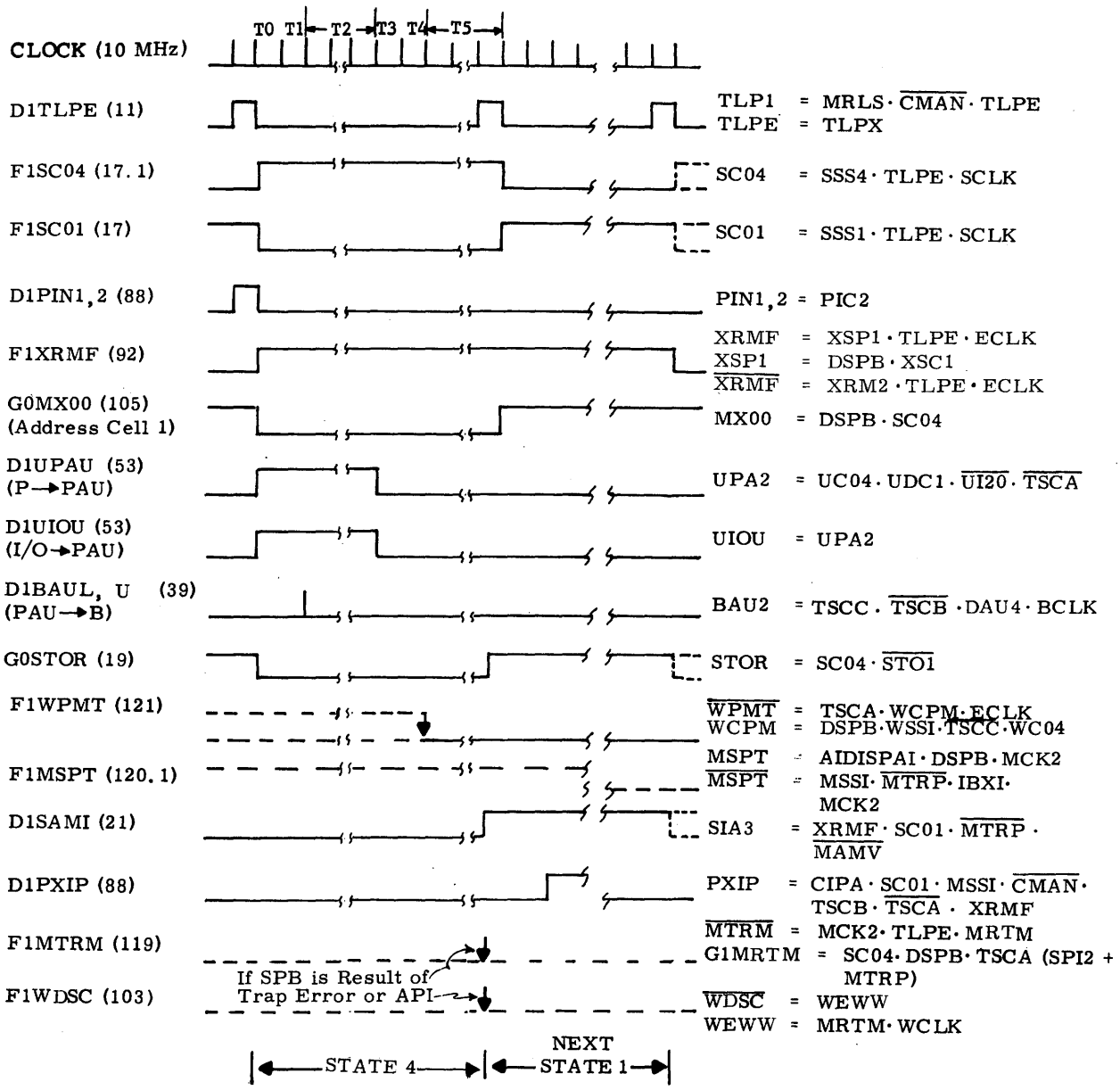


Sequence State 4



Next Sequence State 1

SPB BLOCK DIAGRAM



SPB TIMING DIAGRAM

STA - STORE CONTENTS OF A

FULL OPERAND

23 18	17 15	14	13 0
32	X	*	Y

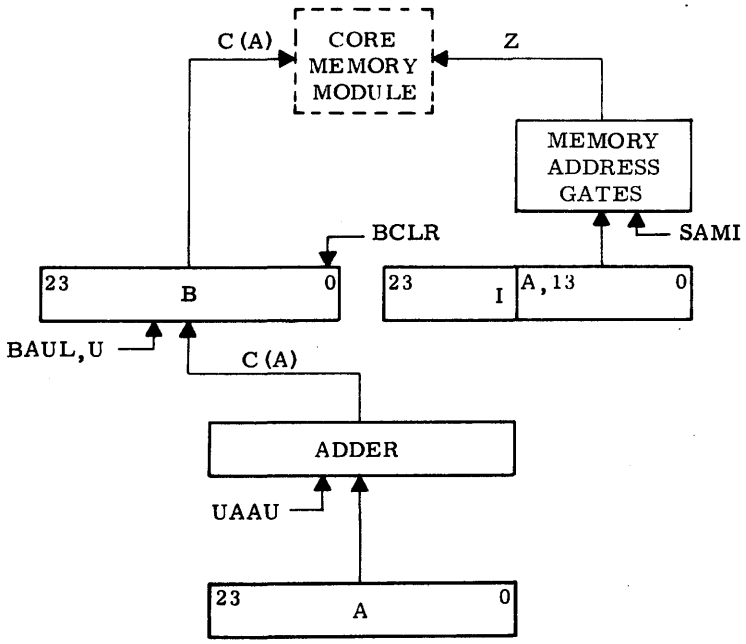
Z = f (X,*,Y)

STA places the contents of the A Register into memory location Z. The contents of the A Register are unchanged.

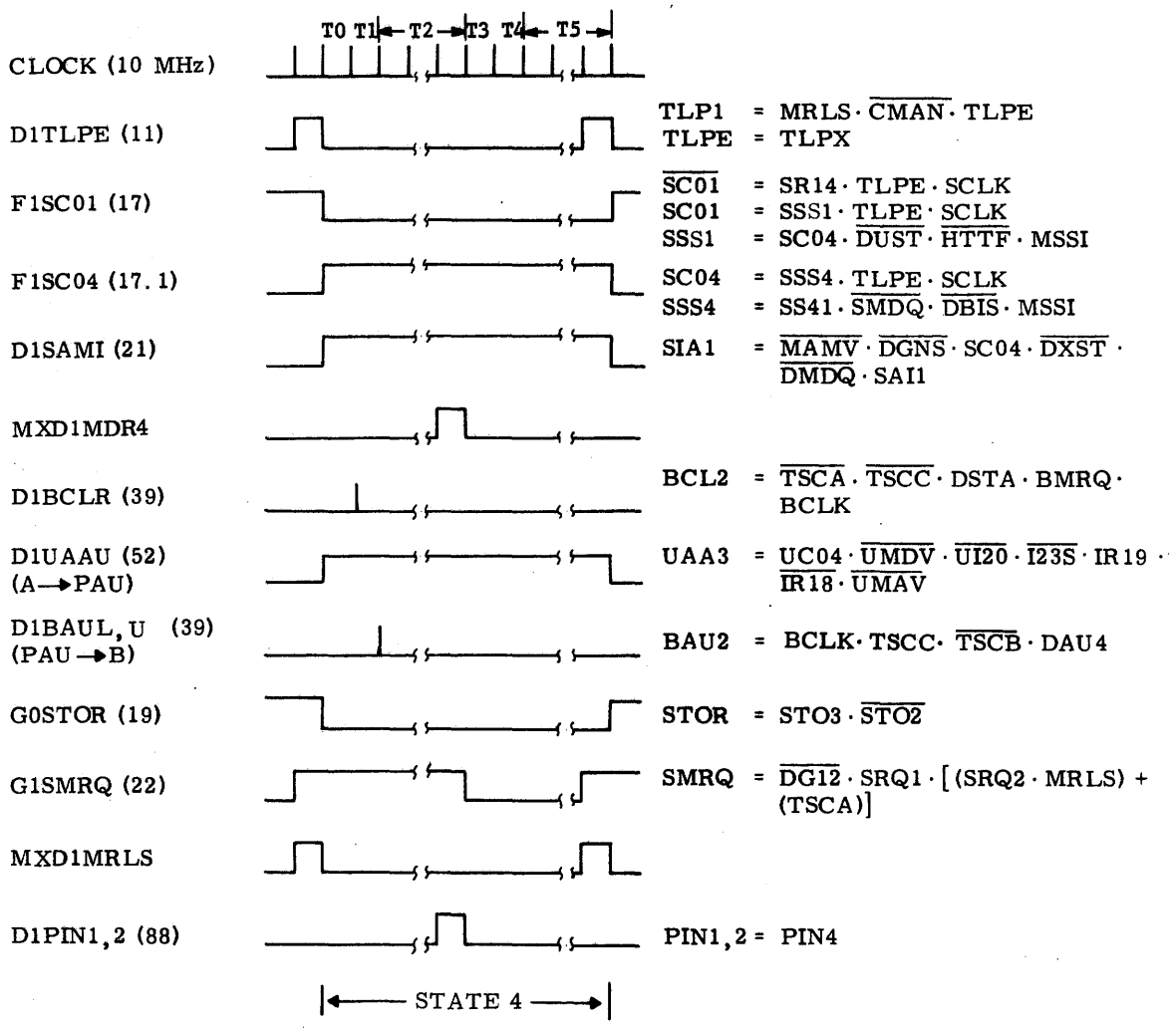
A non-indexed STA command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The B Register is cleared (BCLR) and then the contents of the A Register are gated (UAAU), via the Adder Unit, to the B Register (BAUL,U). From the B Register, the data from the A Register is transferred to the memory module where it is stored in location Z. Following Sequence Control State 4, Sequence Control State 1 is entered to "fetch" the next command.

Non-Indexed Word Times.	2 (S1,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	C (A)

COMMAND CHARACTERISTICS



STA BLOCK DIAGRAM



STA TIMING DIAGRAM

STQ - STORE CONTENTS OF Q

FULL OPERAND

23 — 18	17 — 15	14	13 — 0
44	X	*	Y

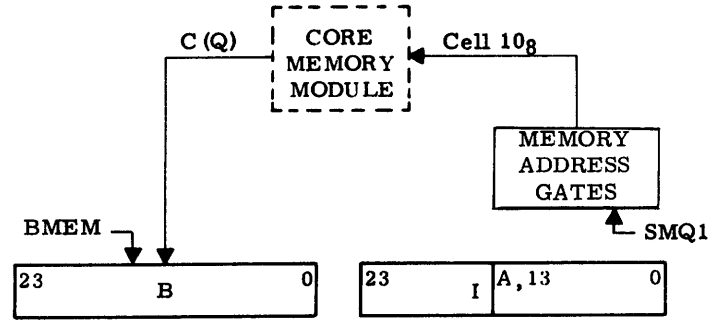
Z = f (X, *, Y)

STQ places the contents of the Q Register (memory location 10_8) into memory location Z. The contents of the Q Register are unchanged by the STQ command.

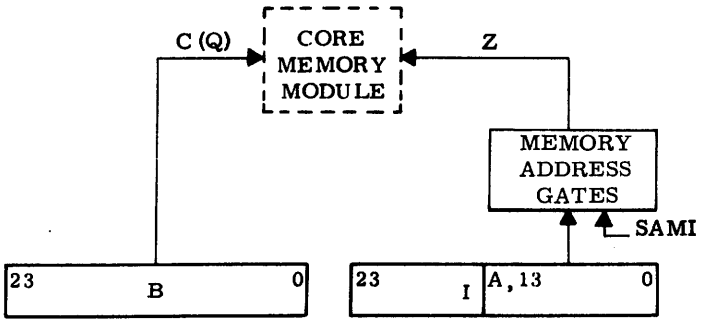
Sequence Control States 3, 4, and 5 are required to execute the STQ command. During State 3, memory is addressed from $I_{A,13-0}$ and the contents of memory location Z are gated to the B Register, but this is only used to detect a Memory Protect Error. Following State 3, State 4 is entered. During State 4, memory location 10_8 is addressed by enabling G0MX03. The contents of cell 10_8 (Q Register) are gated to the B Register. Following State 4, State 5 is entered. During State 5, memory is addressed from $I_{A,13-0}$ (SAMI) and the contents of the B Register are stored in memory location Z. In this manner, the contents of the Q Register are stored in memory location Z.

Non-Indexed Word Times.	4 (S1, S3, S4, S5)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	C (Q) i.e., memory cell 10_8

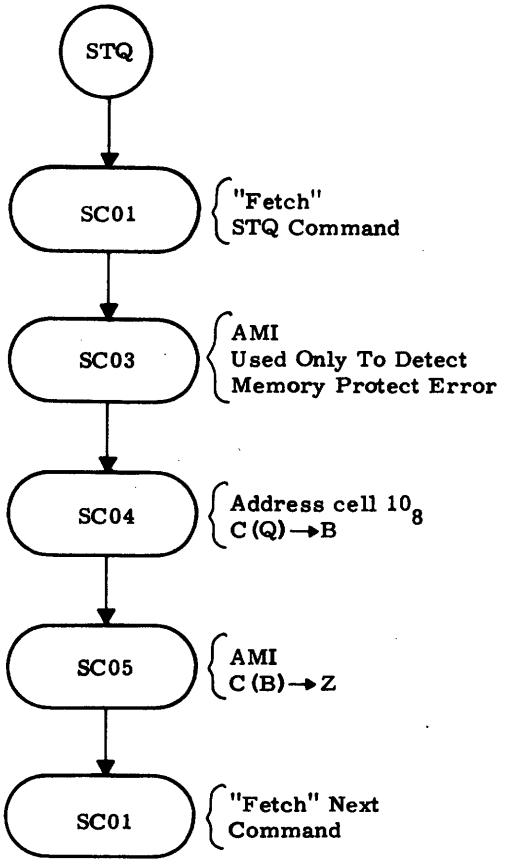
COMMAND CHARACTERISTICS



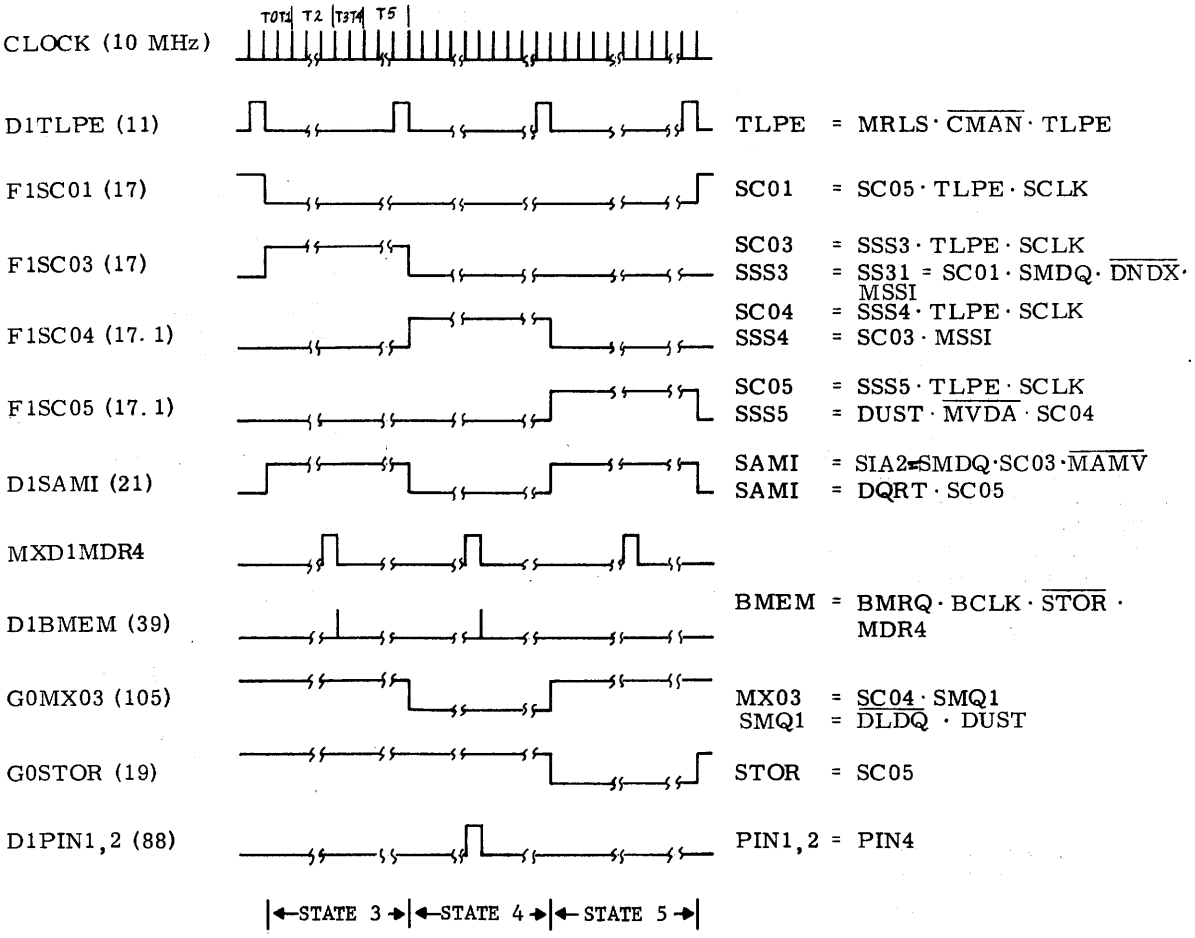
Sequence State 4



Sequence State 5



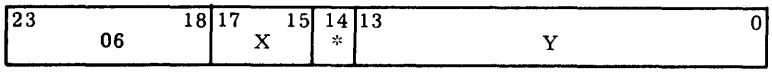
STQ BLOCK DIAGRAM



STQ TIMING DIAGRAM

STX - STORE X LOCATION INTO Z

FULL OPERAND



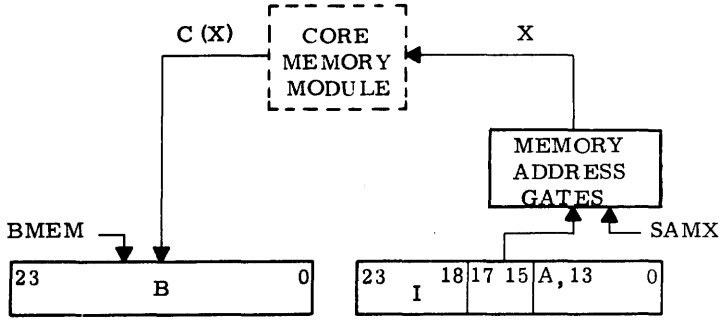
Z = f (Y,*)

STX stores the contents of the indicated X core cell into memory location Z. The STX command may not be automatically modified since bits 15, 16, and 17 are used to specify the X cell to be stored. If no index address is indicated, the STX command is executed as a DMT command.

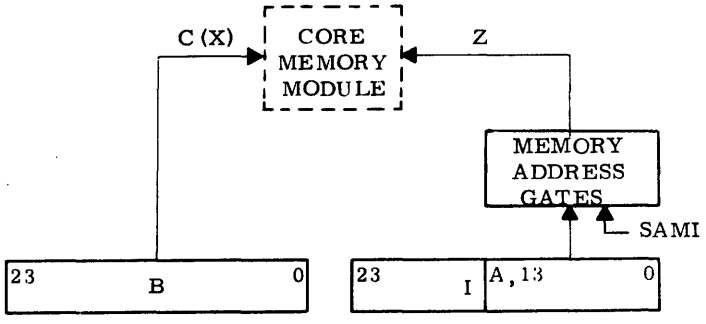
The STX command is executed during Sequence Control States 2, 3, and 4. During State 2 memory is addressed from $I_{A,13-0}$ for detection of a Memory Protect (optional) violation. Following State 2, State 3 is entered. During State 3, memory is addressed from I_{17-15} (G1SAMX) and the contents of the addressed X cell are gated to the B Register. Following State 3, State 4 is entered. During State 4, memory is addressed from $I_{A,13-0}$ and the contents of the B Register are stored in memory location Z. In this manner, the contents of the addressed X cell are stored in location Z.

Word Times.	4 (S1,S2,S3,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	C (X)

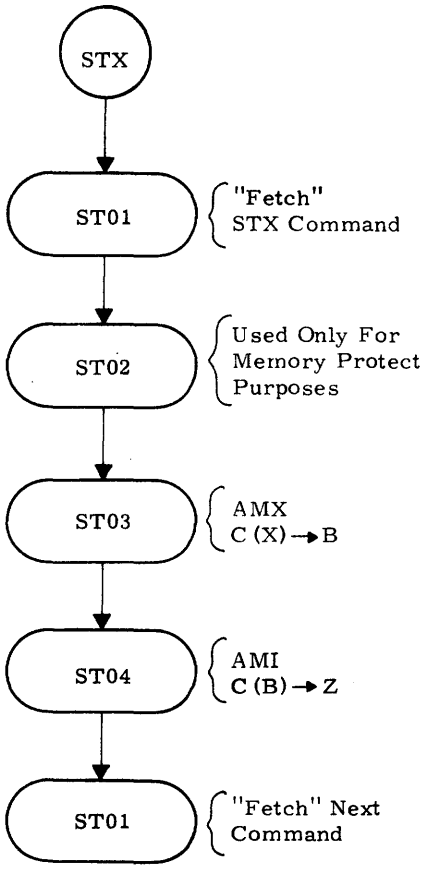
COMMAND CHARACTERISTICS



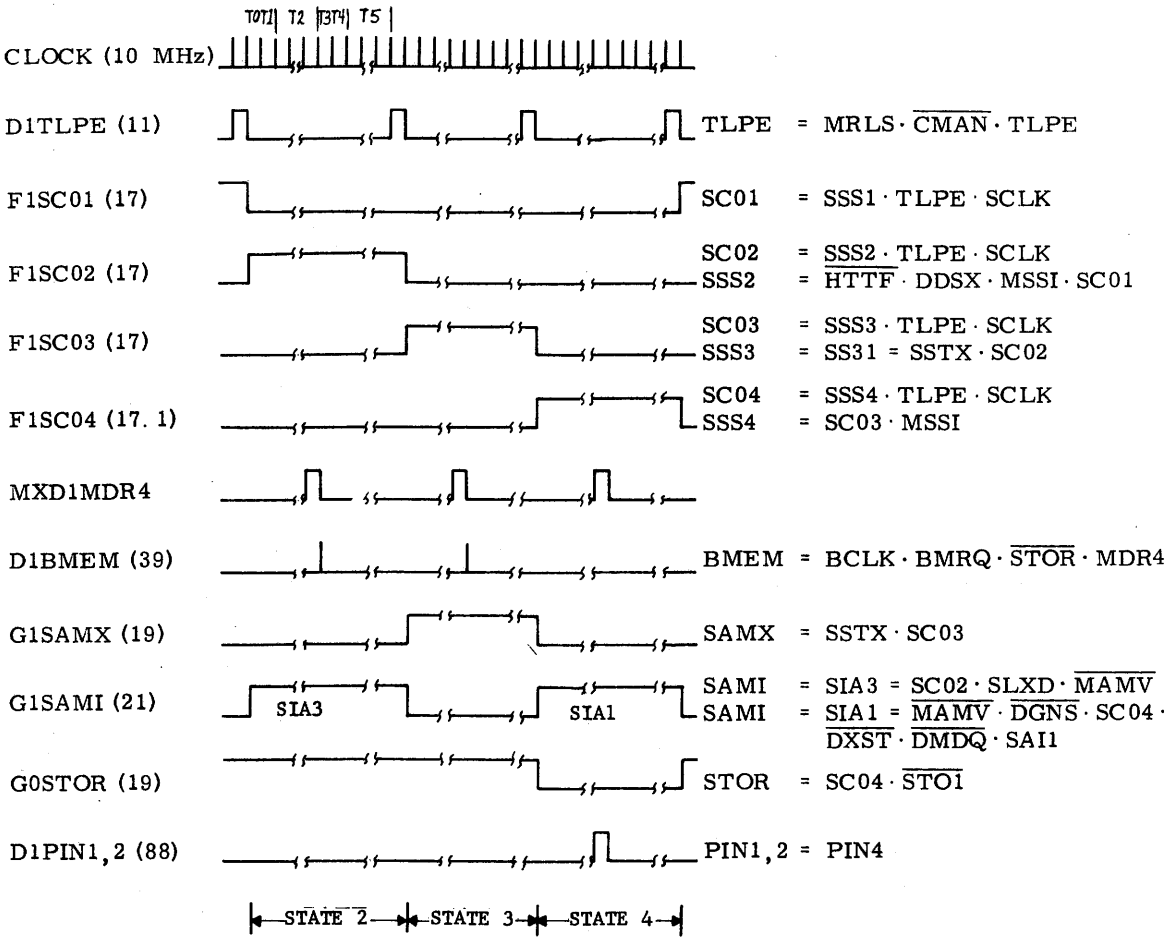
Sequence State 3



Sequence State 4



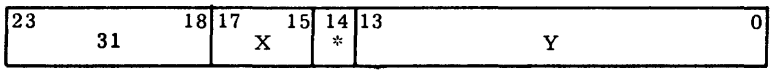
STX BLOCK DIAGRAM



STX TIMING DIAGRAM

SUB - SUBTRACT Z FROM A

FULL OPERAND



Z = f (X,*,Y)

SUB performs algebraic subtraction of the contents of core cell Z from the contents of the A Register. The result of the subtraction is stored in the A Register. If the result is too large to be stored in the 23 bits of A (i.e., more negative than -2^{23} or more positive than $2^{23}-1$), the Overflow flip-flop (F1UOFL) is set.

A non-indexed SUB command is executed during Sequence Control State 4 (SC04). Memory location Z is addressed from $I_{A,13-0}$ (D1SAMI) during State 4. The contents of memory location Z are gated to the B Register by D1BMEM during the Clock pulse of Memory Data Ready (MXD1MDR4). From B, the complement of the contents of memory location Z and the Enable Carry signal (G1UENC) are gated to the Adder Unit. The Enable Carry signal adds 1 to the complemented contents of core memory cell Z forming the 2's complement of Z. The contents of the A Register are also gated to the Adder Unit where the summation of A and the 2's complement of Z occurs. The result is gated (D1AAUL,U) back to the A Register. This is the difference between the contents of the A Register and core cell Z.

If arithmetic overflow - either positive or negative occurs during the summation, the Overflow flip-flop (F1UOFL) is set. The following examples illustrate (a) a positive overflow, and (b) a negative overflow. For simplicity, five bit registers are illustrated. Consider the most significant bit as bit 23, and the next most significant bit as bit 22.

Non-Indexed Word Times.	2 (S1,S4)
Interruptable Following Execution?	Yes
CHANGES FOLLOWING EXECUTION	
A_{23-0}	C (A) - C (Z)
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	Set if Overflow Occurs
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

Consider:

(A = +8) - (Z = -11) = +19. Nineteen is too large to be contained in 4 bits plus a sign bit; thus, the Overflow flip-flop is set.

Contents of Z	1 0101	(-11)
1's Complement of Z	0 1010	
+ Enable Carry	1	(+1)
2's Complement of Z	0 1011	
+ Contents of A	0 1000	(+8)
Total	1 0011	Overflow Set = $23S \cdot \overline{23C} \cdot 22C$.

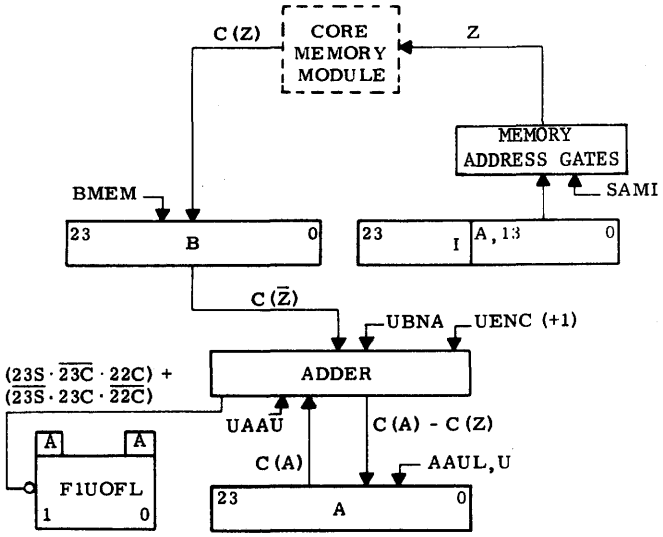
(a) Positive Overflow.

Consider:

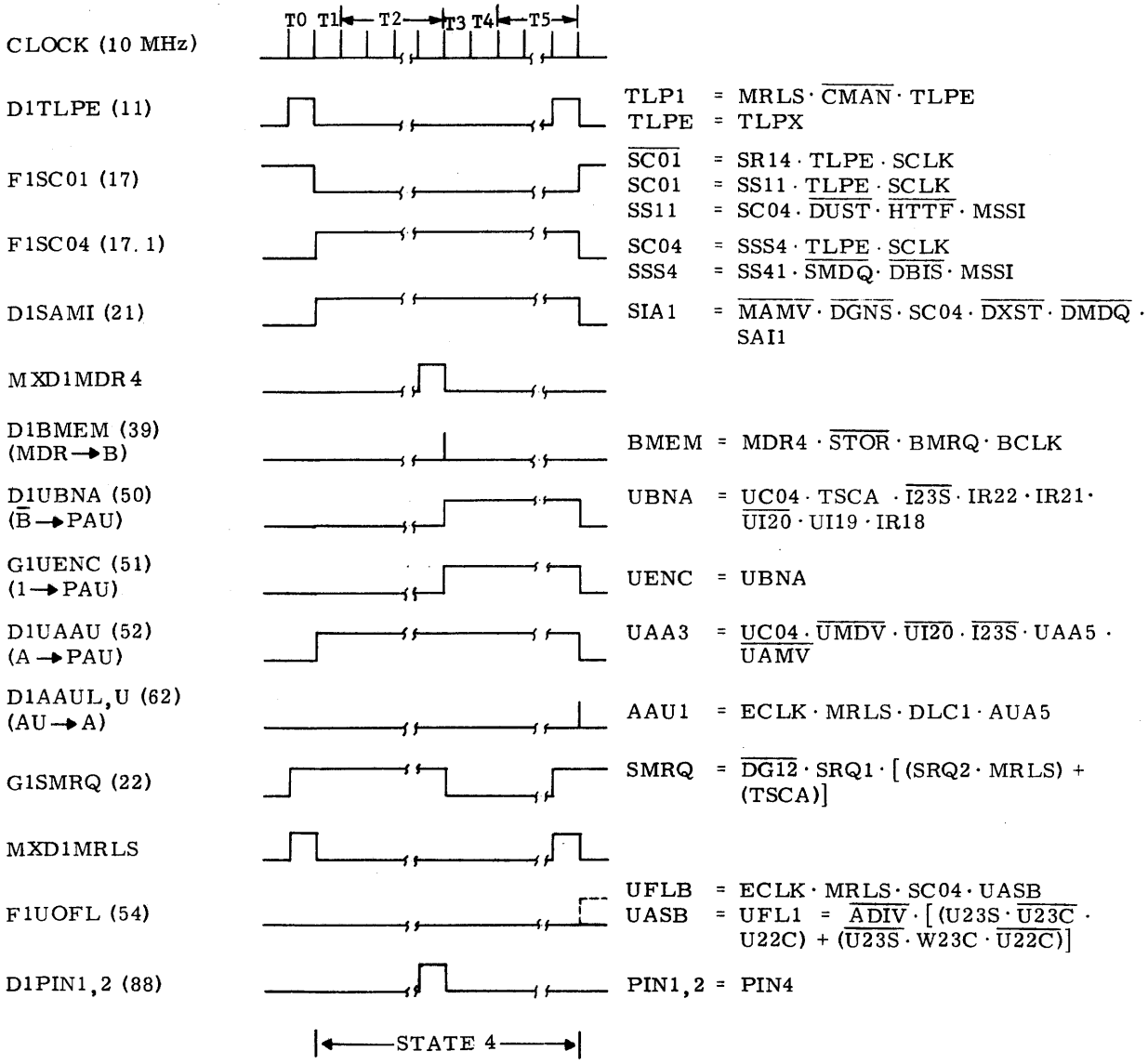
(A = -7) - (Z = +13) = -20. Negative twenty is too large to be contained in 4 bits plus a sign bit; thus, the Overflow flip-flop is set.

Contents of Z	0 1101	(+13)
1's Complement of Z	1 0010	
+ Enable Carry	1	(+1)
2's Complement of Z	1 0011	
+ Contents of A	1 1001	(-7)
Total	0 1100	Overflow Set = $\overline{23S} \cdot 23C \cdot \overline{22C}$.

(b) Negative Overflow.



SUB BLOCK DIAGRAM



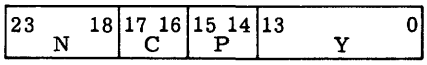
SUB TIMING DIAGRAM

TIM/TOM - TABLE INPUT TO MEMORY/TABLE OUTPUT FROM MEMORY

The Table Input to Memory (TIM) and Table Output from Memory (TOM) capability provides a faster I/O rate and eliminates the normal software "housekeeping" chores required when using GEN 2 commands. This is a semi-direct to memory I/O scheme where data flow is from the I/O Device, to the B Register, to memory, or from memory, to the B Register to the I/O Device. The TIM/TOM method of input/output does not disturb the A, P, Q, J, or Index Registers.

TIM/TOM is not an instruction in the true sense of the word (i.e., such as GEN 2). It is a hardware sequence that is activated by an interrupt from the I/O Device. Each TIM or TOM channel is operated in conjunction with a unique interrupt that is activated by the device. Each of these interrupts is wired to a matrix board which determines the device code and the operation, TIM or TOM, for that device.

When an interrupt from an I/O Device using the TIM/TOM function occurs, the word stored in the interrupt response address is "fetched" from memory. The word stored in this address will be a TIM/TOM Control Word. This Control Word is used to define how the data is to be organized (i.e., how many characters or bytes are contained in each 24 bit word), where in the first 16K of core memory the table of words is located or to be located, and how many words are to be transferred. The format of the TIM/TOM control word is shown below:



N Field - Word Count:

This field contains the current number, in 1's complement form, of words remaining to be read (TOM) or written (TIM). The maximum number of words is 63.

C Field - Character Count:

This field specifies at any given point in time, the number of characters still to be packed (TIM) or unpacked (TOM) in the word being operated on. Initially, C is normally set equal to P.

P Field - Packing Mode:

This field specifies the number, in 2's complement form, of characters to be packed or unpacked in each 24 bit word.

00 = 4 characters

01 = 3 characters

10 = 2 characters

11 = 1 character

Y Field - Starting Address:

This field initially specifies the starting address (-1) of the data table. This field is then incremented to address the word being operated on at any given point in time.

> NOTE
>
> Because the N field may specify up to 63 words and 1, 2, 3, or 4 characters may be packed or unpacked in each word, the number of characters that may be transferred in a block is 252 six-bit characters, 189 eight-bit characters, 126 twelve-bit characters, or 63 twenty-four-bit characters.

As many as forty-eight different TIM/TOM channels may be included in the 4022D System. Each channel has a defined interrupt response address. Associated with each interrupt response address, is a defined device code and input or output enable. The device code and the input or output enable is obtained from a wired matrix board. Therefore, individual interrupts are required for each TIM and each TOM channel.

Each TIM/TOM function requires 12.7 or 30.7 μs to execute depending on whether a high-speed (K3 = 4 or 7) or low-speed channel (K3 = 1 or 2) is used. Therefore, burst mode transfer rates of up to 62,000 words or characters per second is possible.

BASIC OPERATION

Although the TIM and TOM methods of inputting and outputting data are very similar, the general operation of TIM and TOM are listed separately to provide a clearer understanding.

TIM

1. The program loads the assigned API Response Address with the Control Word.

2. When an interrupt occurs from the input device, a matrix board generates the associated device code and enables the input function.

3. The Control Word is "fetched" from the interrupt response address. The C field (bits 16 and 17) of the Control Word is compared with the P field, and if they are equal, the Y address field is incremented by one. The C field is always incremented by one to indicate that one character will be transferred during this cycle. If a carry occurs from the C field (C incremented to 00) it is applied to the N field. A carry will occur from the C field during the cycle that the last character of a word is read in from the device. In this manner, the word count is incremented when the last character of a word is received. When the last character of the last word is received, the N field is incremented to 77_8 and an "Echo" interrupt is generated. When C is equal to 00, C is set equal to the P field to prepare for the next word.

4. The updated Control Word is stored back in the API response address. This updated Control

Word will be addressed when the next interrupt occurs.

5. The contents of the Y Address specified by the Control Word is gated from memory to the B Register. If this is the first character of this word to be obtained, the B Register is cleared. If more than one character is to be stored in the word, the B Register is then shifted left in a circular fashion. That is, if 2 characters are to be stored in this word, B is shifted left circular 12 places; if 3 characters are to be stored in this word, B is shifted left circular 8 places, etc.

6. The contents of the B Register are gated to the Parallel Adder where the character from the device is ORed into the least significant bits. The result is then gated back to the B Register. This input operation occurs similiar to the GEN 2 IN command.

7. The contents of B are then stored back into the memory location specified by the Y field of the Control Word.

8. The above sequence will occur each time the interrupt occurs until the number of words specified by the Control Word have been read. When the specified number of words have been read, N will equal 77_8 and C will equal 0, an "Echo" interrupt is generated, as described in Step 3. After the "Echo" interrupt is generated, the program will normally load the interrupt response address with another TIM control word in preparation for the next input function.

TOM

1. The program loads the API response address with the Control Word.

2. When an interrupt occurs from the output device, a matrix board generates the associated device code and enables the output function.

3. The Control Word is "fetched" from the interrupt response address. The C field of the control word is compared with the P field and if they are equal, the Y address is incremented. The C field is always incremented by one to indicate that one character will be transferred during this cycle. If a carry occurs from the C field (i. e. , C is incremented to 00), it is applied to the N field. A carry will occur from the C field during the cycle that the last character of a word is transferred to the device. In this manner, the word count is incremented when the last character of a word is transferred to the device. When the last character of the last word is transferred, the N field will be incremented to 77_8 and an "Echo" signal is generated. Whenever the C field is equal to 00, the C field is set equal to the P field, to prepare for the next word.

4. The updated Control Word is stored back in the API response address. This updated Control Word will be addressed when the next interrupt occurs.

5. The contents of the Y Address specified by the Control Word is gated from memory to the B Register. If more than one character is contained in a word, the B Register is then shifted left circular. That is, if 2 characters are contained in a word, the B Register is shifted left circular 12 places; if 3 characters are contained in the word, B is shifted left circular 8 places, etc.

6. The contents of the B Register are gated to the Adder where the least significant bits (1 character) is transferred to the addressed output device. As previously mentioned, the device address (code) is generated by a matrix board from the interrupt input. The output operation is similiar to the GEN 2 OUT command.

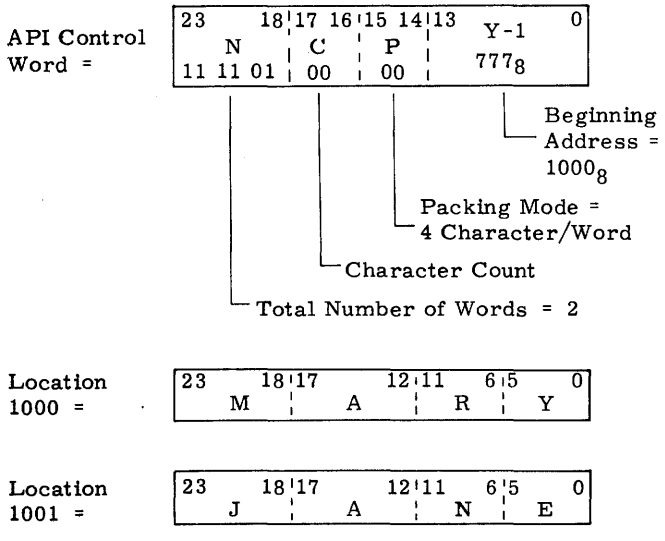
7. The contents of B are then stored back in the memory location specified by the Y Address of the Control Word.

8. The above sequence will occur each time the interrupt occurs, until an "Echo" interrupt is generated, as described in step 3. The "Echo" signal is generated when N is equal to 77_8 and C is equal to 0 . After the "Echo" interrupt is generated, the program will normally load the interrupt response address with another TOM control word in preparation for the next function.

To further illustrate the operation of the TOM function, the following example is provided. This example illustrates the organization of data to be transferred, how the Control Word is updated, and the general sequence of events during a TOM operation.

TOM Example:

This example illustrates the basic operation using the TOM function, to type out the words, MARY JANE coded and previously stored in memory beginning in location 1000_8.



1st INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 01	01	00	1000_8

Store Control Word in API Location

Contents of $1000_8 \rightarrow$B

23 18	17 12	11 6	5 0
A	R	Y	M

Shift B Left Circular 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1000_8

2nd INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 01	10	00	1000_8

Store Control Word in API Location

Contents of $1000_8 \rightarrow$B

23 18	17 12	11 6	5 0
R	Y	M	A

Shift B Left Circular 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1000_8

3rd INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 01	11	00	1000_8

Store Control Word in API Location

Contents of $1000_8 \rightarrow$B

23 18	17 12	11 6	5 0
Y	M	A	R

Shift B Left Circular 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1000_8

4th INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 10	00	00	1000_8

Store Control Word in API Location

$Bit_{14,15} \rightarrow Bit_{16,17}$

Contents of $1000_8 \rightarrow$B

23 18	17 12	11 6	5 0
M	A	R	Y

Shift B Left Circular 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1000_8

5th INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 10	01	00	1001_8

Store Control Word in API Location

Contents of $1001_8 \rightarrow$B

23 18	17 12	11 6	5 0
A	N	E	J

Shift B Left Circular 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1001_8

6th INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 10	10	00	1001_8

Store Control Word in API Location

Contents of $1001_8 \rightarrow$B

23 18	17 12	11 6	5 0
N	E	J	A

Shift B Left 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1001_8

7th INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 10	11	00	1001_8

Store Control Word in API Location

Contents of $1001_8 \rightarrow$B

23 18	17 12	11 6	5 0
E	J	A	N

Shift B Left 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1001_8

8th INTERRUPT:

Update Control Word

23 18	17 16	15 14	13 0
11 11 11	00	00	1001_8

Store Control Word in API Location

$Bit_{14,15} \rightarrow Bit_{16,17}$ GENERATE ECHO

Contents of $1001_8 \rightarrow$B

23 18	17 12	11 6	5 0
J	A	N	E

Shift B Left 6 Places

Simulate OUT → To Typer

Store Shifted Word in 1001_8

The ECHO signal indicates this is the last character in the table.

THEORY OF OPERATION

The TIM/TOM function is initiated upon receipt of an interrupt at an interrupt channel wired for the TIM/TOM function. This interrupt is wired to a matrix board to define the operation of TIM or TOM (S bit = 5 or 4) and to define the device code (K3, K2, K1, and K0) address of the device causing the interrupt. The TIM or TOM function is then executed in Sequence Control States 1, 3, 4, and 5. Fig. TIM/TOM. 1 illustrates the basic functions performed in each of these Sequence States. A detailed flow chart of the TIM/TOM function is contained in Fig. TIM/TOM. 2. Refer to these aids during the following discussion.

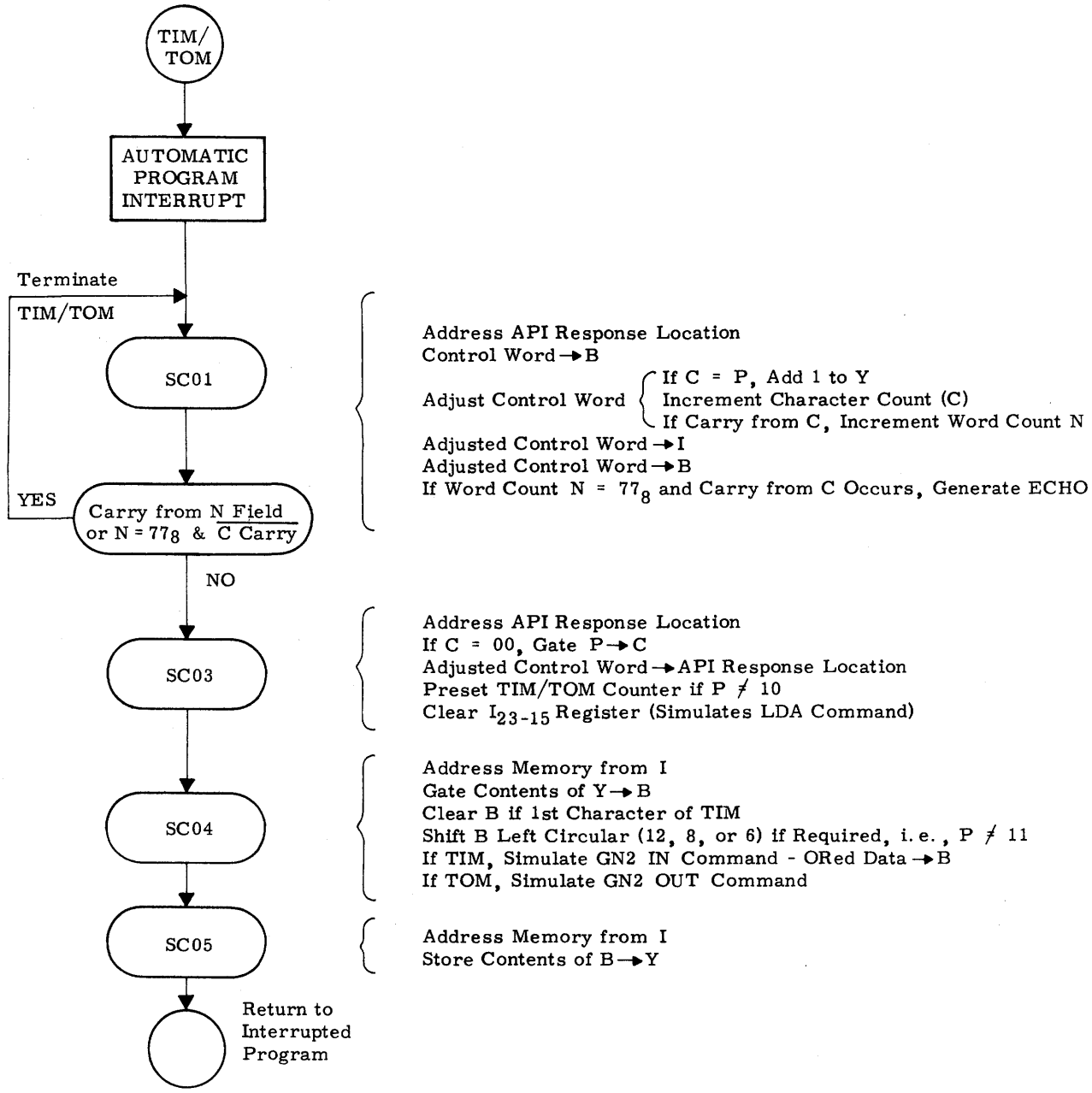


Fig. TIM/TOM. 1 Sequence State Flow Chart

Device Code Matrix

A diode matrix board (RTTF1 and RTTD1) is used to generate the device and operation code from an interrupt input defined for use as a TIM or TOM function. This matrix board is shown on sheet 133 through 133.3 of the Arithmetic Unit logic.

Each TIM and TOM interrupt is wired to one of the input drivers, D0HL01 through D0HL48. From each of these drivers, diodes are inserted to enable only the inverters (N1H(G)T00 through N1H(G)T11) corresponding to the K bits of the device code. Also, all input drivers used for TIM functions will have a diode inserted to enable N1H(G)T12. The following list of inverters illustrates the K bit decode pattern. This K bit decode corresponds to the device code specified by the GEN 2 commands.

N1H(G)T00 N1H(G)T01 N1H(G)T02	K0
N1H(G)T03 N1H(G)T04 N1H(G)T05	K1

N1H(G)T06 N1H(G)T07 N1H(G)T08	K2
N1H(G)T09 N1H(G)T10 N1H(G)T11	K3
N1H(G)T12	S

Any TIM operation enables N1H(G)T12. N1H(G)T12 is then used in S bit decoding to enable an S = 5 (IN) function. Whenever N1H(G)T12 is disabled, during a TIM/TOM function, an S = 4 (OUT) function is enabled indicating a TOM operation.

The diodes are organized on the RTTF1 and RTTD1 boards and the logic of the boards shown on sheet 133.1 and 133.3 are drawn to permit the device code and S bit for any TIM/TOM channel to be easily determined by observing the diodes inserted. For example, the T/T#1 channel (API 201) has diodes inserted to enable S = 5 (N1H(G)T12) and to provide a device code (K3, K2, K1, K0) of 7003_8 which is the first/primary card or paper tape reader.

Sequence State 1

Block and timing diagrams of State 1 operation are contained in Figs. TIM/TOM. 3 and 4. During State 1, memory is addressed from the API (D1SAMW) and the TIM/TOM Control Word from the interrupt response address is gated to the B Register. The I Register is cleared (D0ICU1,CUL). If C (bits 16 and 17) is equal to P (bits 14 and 15), a "one" is set in I_0. A "one" is unconditionally set into I_{16}. The contents of I (I_{16} = 1 and I_0 = 1 if C = P) are gated to the Adder (U1AU,LA). The contents of B (Control Word) are also gated to the Adder (D1UBBU) where it is summed with the contents of I. The result of this summation will then add 1 to C, add 1 to N if a carry results from C, and add 1 to Y if C was equal to P. The result of this summation is then gated to both the I and B Registers.

If during the summation, a carry resulted from C (PAU_{17}) causing N to be incremented to 77_8, an Echo signal (G1WEKO) is generated to indicate that this is the last character of the last word to be transferred. This Echo signal is then used to generate an API which is used by the program to determine that the transfer is complete and that a new Control Word must be stored in the Interrupt response address if further data transfers are desired.

If during the summation, a carry resulted from PAU_{23}, indicating that N was incremented beyond 77_8, or if N is equal to 77_8 and no carry results from C (i.e., PAU_{17}) the operation is terminated (G1HSTP). This indicates that an interrupt occurred after the last word specified by the Control Word was transferred.

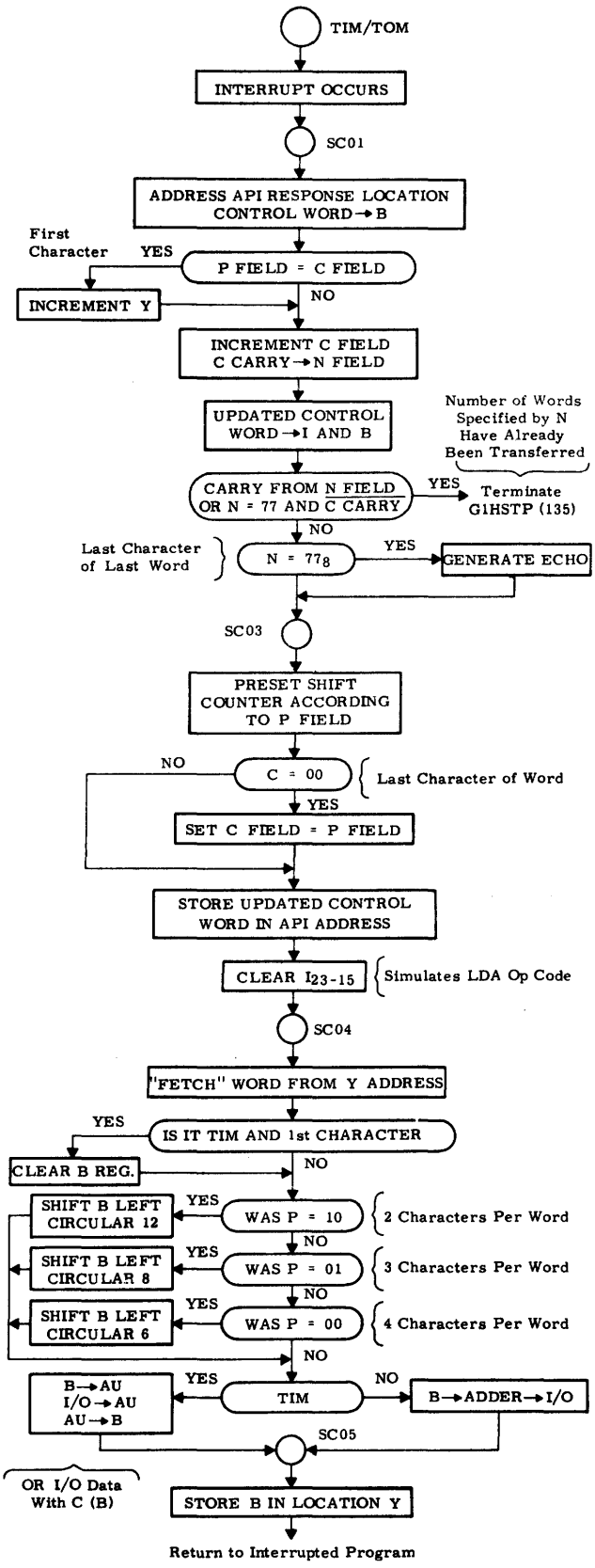


Fig. TIM/TOM. 2 TIM/TOM Flow Chart

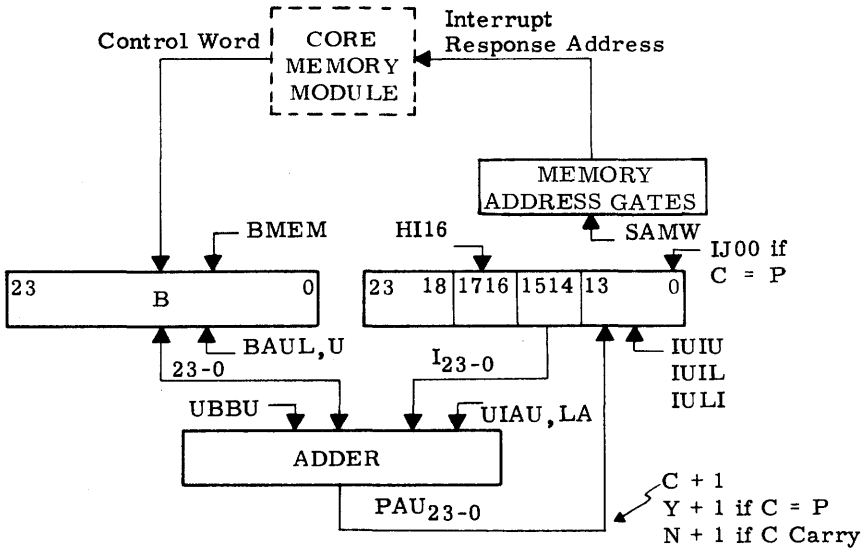


Fig. TIM/TOM. 3 State 1 Block Diagram

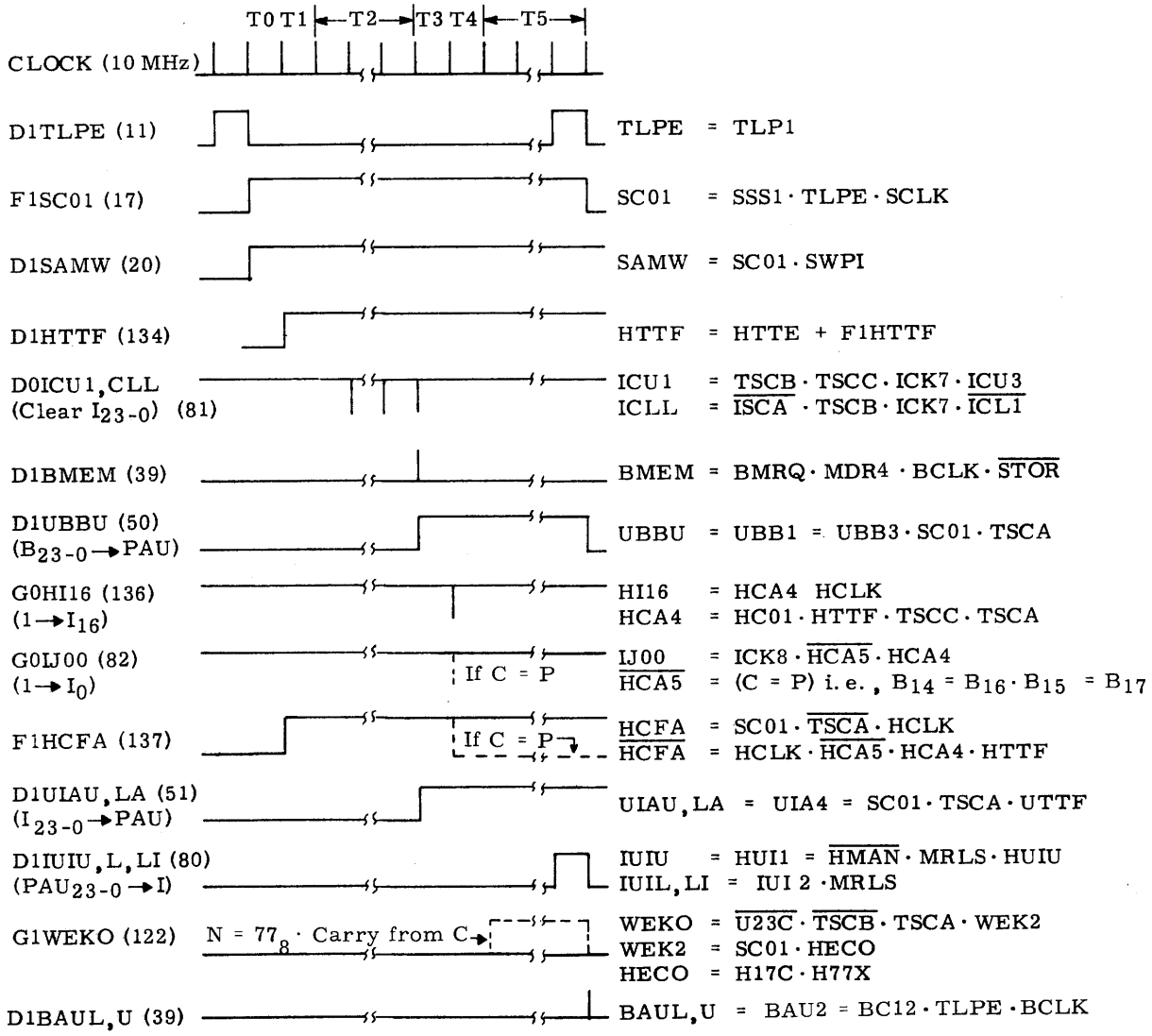


Fig. TIM/TOM. 4 State 1 Timing Diagram

Termination is implemented by holding sequencing in State 1 and allowing the interrupted program to continue.

Following State 1, State 3 is entered.

Sequence State 3

Timing and block diagrams of State 3 operation are contained in Figs. TIM/TOM. 5 and 6. During Sequence State 3, the C field ($I_{17,16}$) is examined and if it is equal to 00, the C field is set (G0HI17,16) equal to the P field ($I_{15,A}$). This indicates that this is the last character of the present word and presets the C field for the next word. The contents of I (updated Control Word) are then gated to the B Register via the Adder. Memory is addressed from the API Module (D1SAMW) and the updated Control Word is stored back in the API Response Address. The next time an interrupt occurs for this address, this Control Word will again be used.

$I_{23-15,A}$ is cleared (D0ICU1). Clearing $I_{23-15,A}$ simulates an LDA command for use during the next Sequence State.

Also during State 3, the TIM/TOM Counter (F1HCFA-E) is preset, if required, to control the number of shifts required to position the data word. A discussion of this control is provided later in this section.

Following State 3, State 4 is entered.

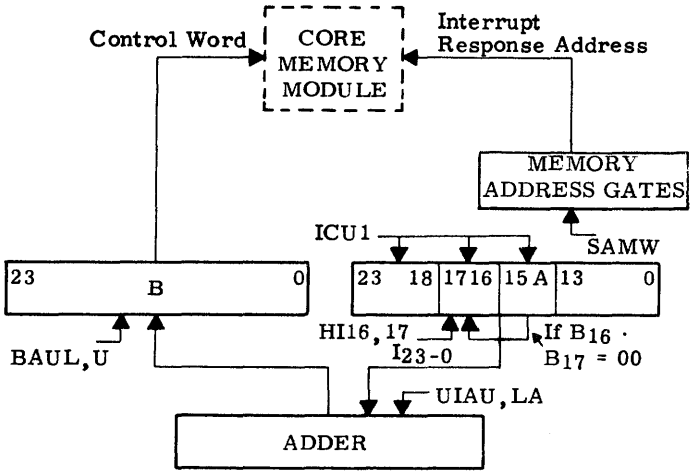


Fig. TIM/TOM. 5 State 3 Block Diagram

Sequence State 4

Block and timing diagrams of State 4 operation are contained in Figs. TIM/TOM. 7 and 8. During State 4, the actual data transfer occurs. To accomplish this function, the duration of State 4 is extended in the same manner as are GEN 2 commands.

During the first portion of State 4, an LDA command is simulated, except that the contents of the AU are not gated to the A Register. This is accomplished because the Op Code portion of the I Register was cleared during State 3. Memory is addressed from Y and the contents of the memory cell are gated to the B Register. This is the contents of the memory cell that data is to be transferred from (TOM) or that data is to be transferred to (TIM).

If this is the first character of a TIM word, the B Register is cleared (D1BCLR). Clearing of B is necessary since the data bits received from the input device are ORed with B in the Adder.

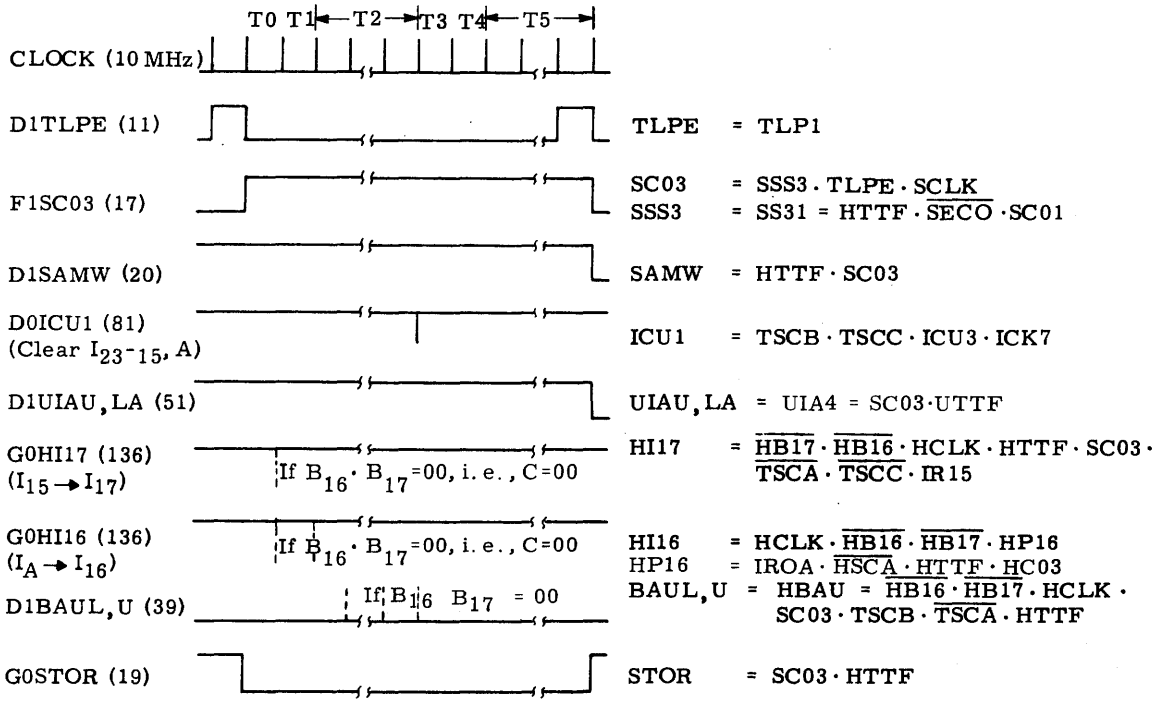


Fig. TIM/TOM. 6 State 3 Timing Diagram

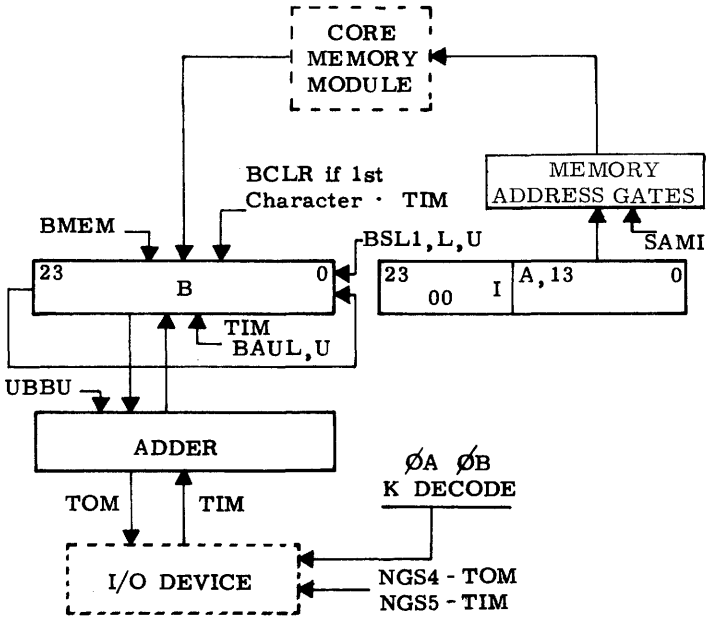


Fig. TIM/TOM. 7 State 4 Block Diagram

Time 6 Envelope is then entered. The basic timing of Time 6 is the same as that for an IN (TIM) or OUT (TOM) command. First, the data word must be positioned so that the character to be transferred is either in the least significant bits of the B Register or that space is reserved in the least significant bits of the B Register for the character to be received. This positioning of the B Register is accomplished by shifting the B Register left in a circular fashion, the number of bit positions that a character requires. This operation is described below.

After the shifting is complete, data is transferred from B to the output device via the Adder (TOM) or the data received from the input device is gated to the Adder where it is ORed with the least significant bits of the B Register. In either case, the contents of the Adder are gated back to the B Register.

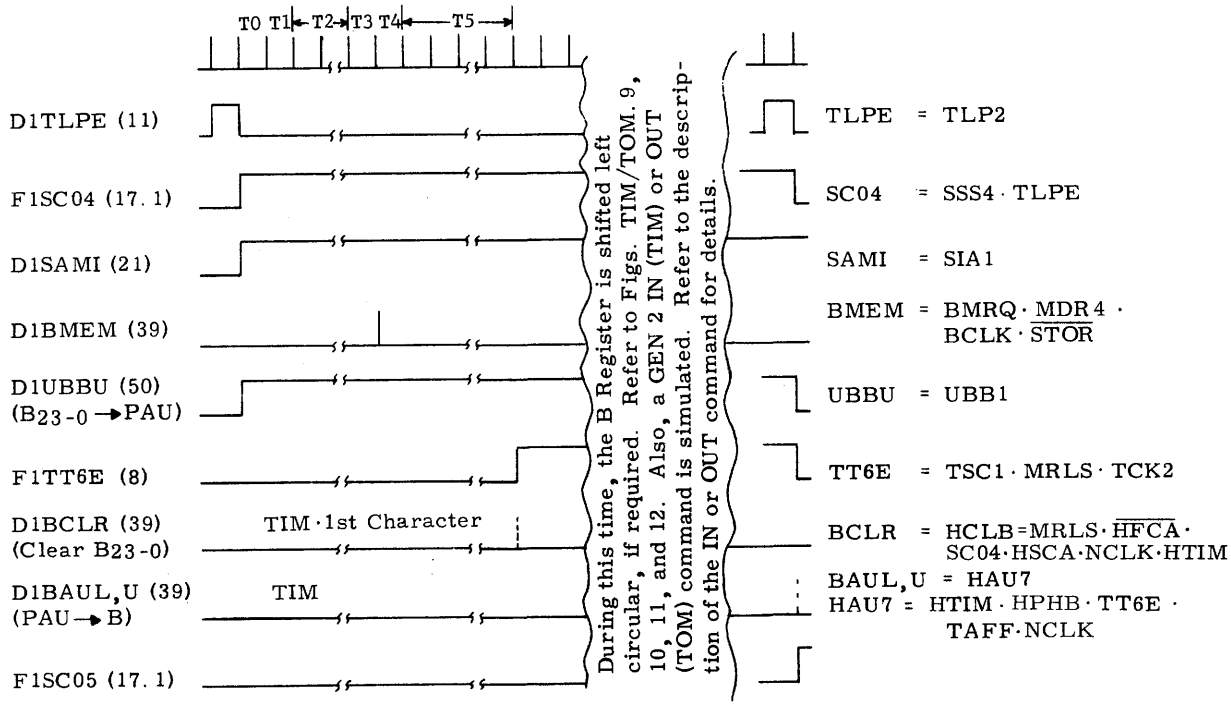


Fig. TIM/TOM. 8 State 4 Timing Diagram

Following State 4, State 5 is entered to store the data word back in memory location Y. Prior to discussing State 5, however, a discussion of positioning the character in the B Register is provided.

> NOTE
>
> When a TOM function is executed specifying 4 characters per word (i.e., P = 00), the complement of bit 5 of the data word is transferred to bit 6 of the output device by X1ND06 (sheet 101). This feature is primarily used by the paper tape punch. Using this technique, ASCII coded tape may be punched from six bit characters. Examination of the ASCII character code reveals that the seventh bit is always the complement of the sixth bit. This gating structure, therefore, conserves memory space by generating a seven bit ASCII code from six bits stored in memory. For example:
>
> 6-bit code to be punched = 0 1 0 1 0 1
>
> 7-bit ASCII code punched = 1 0 1 0 1 0 1
>
> X1ND06 (102) = N6N5
> G1N6N5 = N1N6N5·N1N05S
>
> If a 7-bit ASCII punched tape is read-in using a TOM function that specifies 4 characters per word, all 7-bits are read-in to the I/O Buffer where the character is checked for parity. The seventh bit is then discarded and 6-bits are transferred to the Arithmetic Unit B Register.

Character Positioning

Since more than one character may be contained by a word, the B Register must be positioned so that the character being operated on will be in the least significant bits. This is accomplished by shifting the B Register left circular the number of placed that the character contains, except when one character is equal to a word. The P field of the Control Word specifies the number of characters in a word. This value is used to determine the number of shifts required.

P = 11 - 1 character per word, no shifting required.

P = 10 - 2 characters per word, 12 shifts required.

P = 01 - 3 characters per word, 8 shifts required.

P = 00 - 4 characters per word, 6 shifts required.

The TIM/TOM Counter (F1HCFA-E) provides control of the number shifts. During State 3, the Counter is preset, if necessary. During the first clock pulses of Time 6 Envelope of State 4, the B Register is shifted, if required. At each shift of B, the TIM/TOM Counter is incremented. When the required number of shifts have occurred, further shifting is inhibited by the TIM/TOM Counter.

Figs. TIM/TOM. 9, 10, 11, and 12 contain timing diagrams and logic equations of the shift control logic.

Sequence State 5

During Sequence State 5, the contents of B (word being operated on) is stored back in memory location Y, as specified by the Control Word. Figs. TIM/TOM. 13 and 14 contain block and timing diagrams of the operation of State 5.

During State 5, memory is addressed from I (D1SAMI), and the contents of the B Register are stored in memory location Y. The TIM/TOM flip-flop (F1HTFF) is cleared to end the TIM/TOM operation.

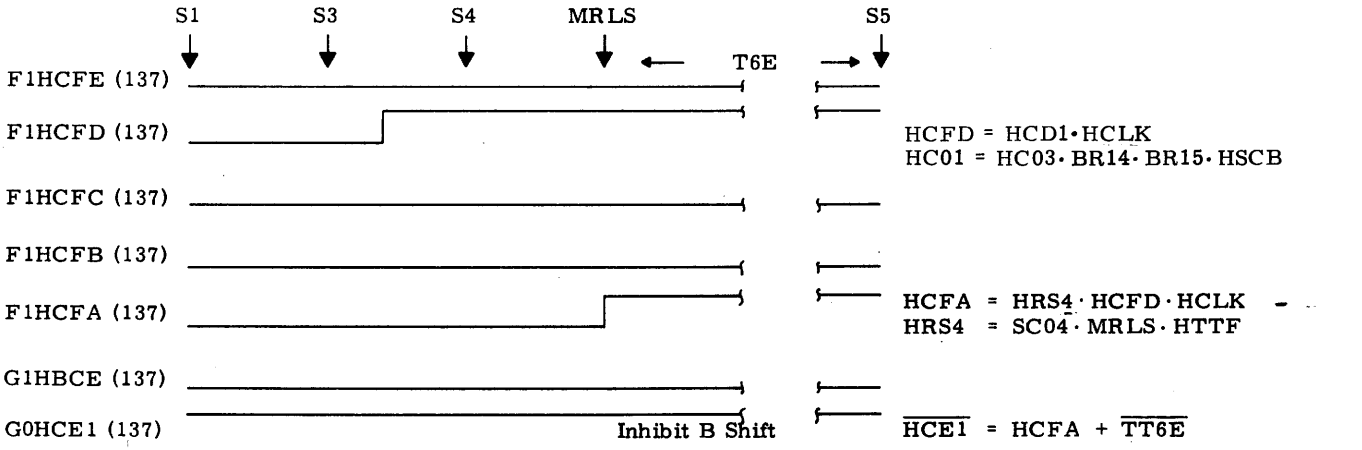
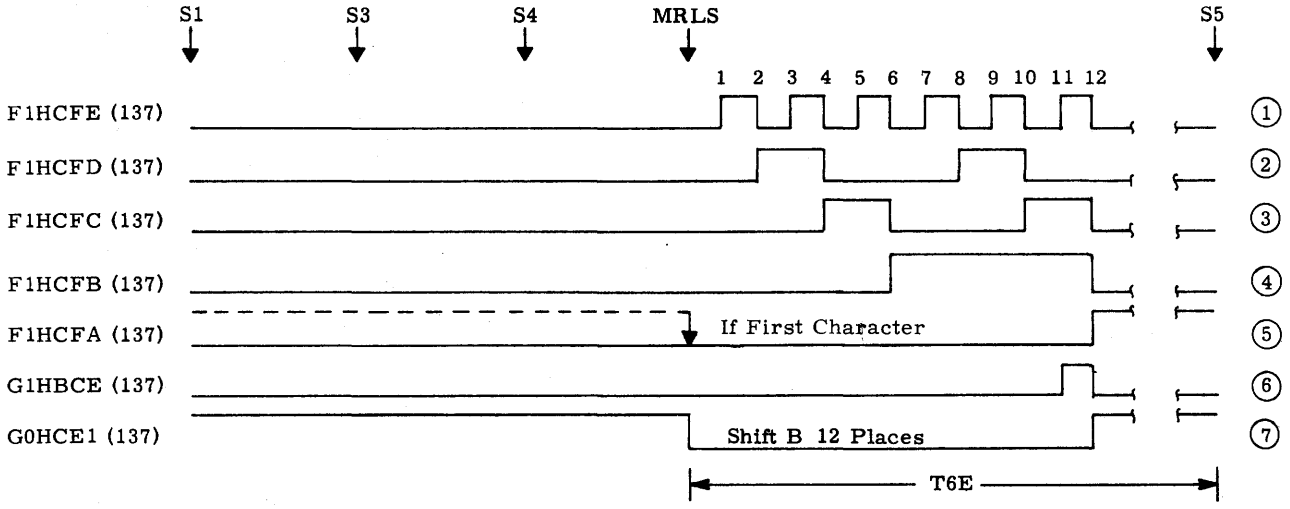


Fig. TIM/TOM. 9 P = 11 - 1 Character/Word Timing Diagram



① $\underline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot \overline{HCFE}$
$\overline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot HCFE + SC01 \cdot HCLK$

② $\underline{HCFD} = HCFE \cdot \overline{HCFC} \cdot HCLK$
$\overline{HCFD} = HCFE \cdot HCLK + SC01 \cdot HCLK$

③ $\underline{HCFC} = HCFE \cdot \underline{HCFD} \cdot HCLK$
$\overline{HCFC} = HCFE \cdot \overline{HCFD} \cdot HCLK + SC01 \cdot HCLK$

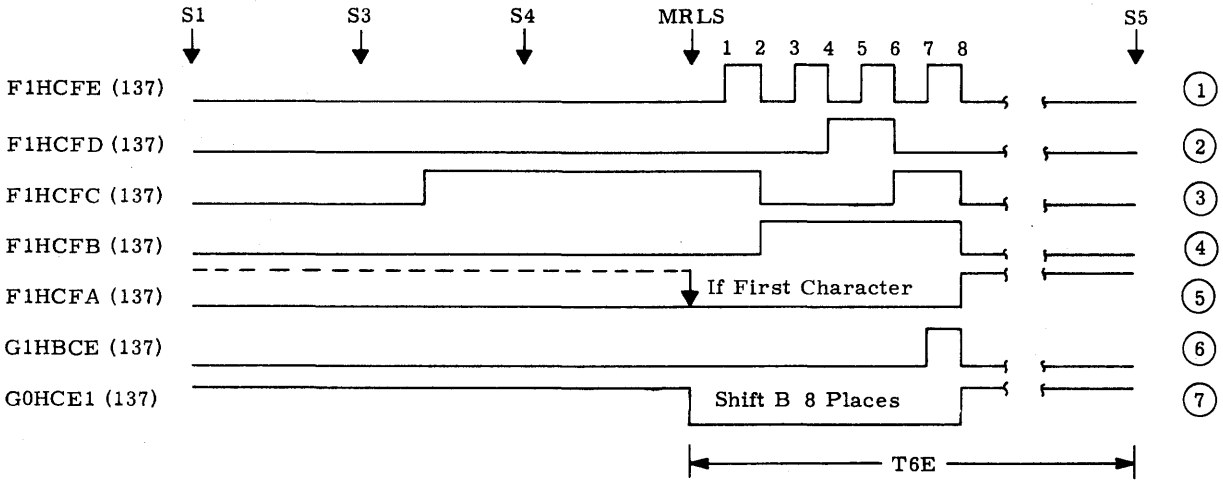
④ $\underline{HCFB} = HCFC \cdot HCFE \cdot HCLK$
$\overline{HCFB} = HCFE \cdot HCFC \cdot HCLK + SC01 \cdot HCLK$

⑤ $HCFA = HBCE \cdot HCLK$

⑥ $HBCE = HCFB \cdot HCFC \cdot HCFE$

⑦ $HCE1 = \overline{HCFA} \cdot TT6E$

Fig. TIM/TOM. 10 P = 10 - 2 Character/Word Timing Diagram



① $\underline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot \overline{HCFE}$
$\overline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot HCFE + SC01 \cdot HCLK$

② $\underline{HCFD} = HCFE \cdot \overline{HCFC} \cdot HCLK$
$\overline{HCFD} = HCFE \cdot HCLK + SC01 \cdot HCLK$

③ $\underline{HCFC} = \overline{BR15} \cdot \underline{HB14} \cdot SC03 \cdot HSCB + HCFE \cdot HCFD \cdot HCLK$
$\overline{HCFC} = HCFE \cdot \overline{HCFD} \cdot HCLK + SC01 \cdot HCLK$

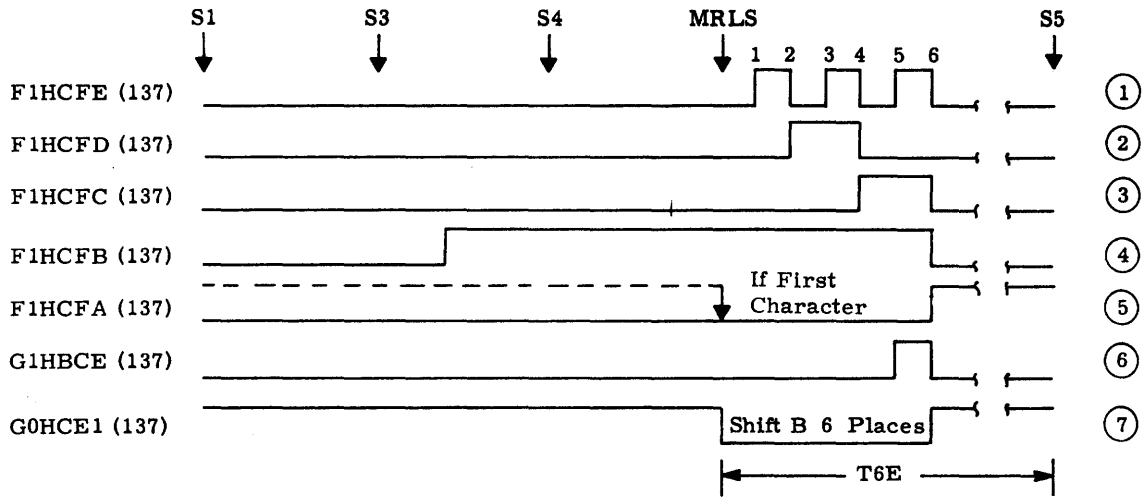
④ $\underline{HCFB} = HCFC \cdot HCFE \cdot HCLK$
$\overline{HCFB} = HCFE \cdot HCFC \cdot HCLK + SC01 \cdot HCLK$

⑤ $HCFA = HBCE \cdot HCLK$

⑥ $HBCE = HCFB \cdot HCFC \cdot HCFE$

⑦ $HCE1 = \overline{HCFA} \cdot TT6E$

Fig. TIM/TOM. 11 P = 01 - 3 Character/Word Timing Diagram



(1) $\underline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot \overline{HCFE}$
$\overline{HCFE} = \overline{HCFA} \cdot TT6E \cdot HCLK \cdot HCFE + SC01 \cdot HCLK$

(2) $\underline{HCFD} = HCFE \cdot \overline{HCFC} \cdot HCLK$
$\overline{HCFD} = HCFE \cdot HCLK + SC01 \cdot HCLK$

(3) $\underline{HCFC} = HCFE \cdot \underline{HCFD} \cdot HCLK$
$\overline{HCFC} = HCFE \cdot \overline{HCFD} \cdot HCLK + SC01 \cdot HCLK$

(4) $\underline{HCFB} = \overline{BR15} \cdot \overline{BR14} \cdot HSCB \cdot SC03 + HCFC \cdot HCFE \cdot HCLK$
$\overline{HCFB} = HCFE \cdot HCFC \cdot HCLK + SC01 \cdot HCLK$

(5) $HCFA = HBCE \cdot HCLK$

(6) $HBCE = HCFB \cdot HCFC \cdot HCFE$

(7) $HCE1 = \overline{HCFA} \cdot TT6E$

Fig. TIM/TOM. 12 P = 00 - 4 Character/Word Timing Diagram

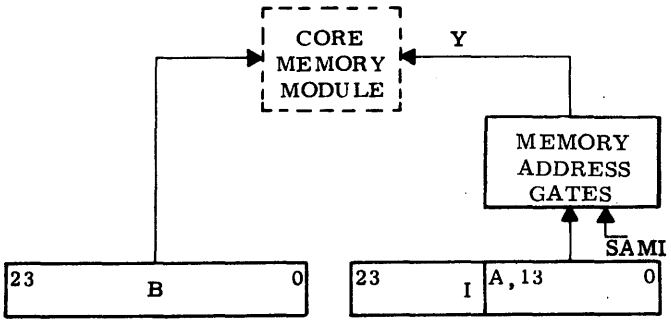


Fig. TIM/TOM. 13 State 5 Block Diagram

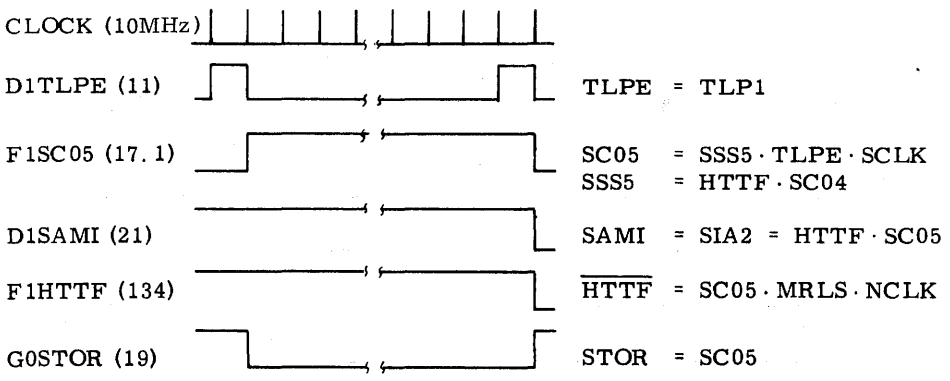


Fig. TIM/TOM. 14 State 5 Timing Diagram

TXH - TEST X HIGH OR EQUAL

FULL OPERAND

23 18	17 15	14	13 0
24	X	0	-K

TXH sets the Test flip-flop (F1ETST) if the contents of the specified X cell (bits 14-0) are greater than or equal to the value K. If the contents of X_{14-0} are less than K, the Test flip-flop is cleared. Bits 15, 16, and 17 of the TXH command specify the address of the X cell to be compared. The K value of the TXH command must be specified in 2's complement form. The assembly program forms the 2's complement of the K value for the programmer. The value of K may range between 16,383 and 1. The value of the index cell contents may vary between 0 and 32,767. The contents of the addressed X cell are not changed by the TXH command. If the value of K is zero, the Test flip-flop cannot be set. If bits 15, 16, and 17 are "zero", the command is undefined, i.e., an X cell address must be specified.

During Sequence Control State 1, the TXH command is "fetched" from memory in the normal manner. However, as bits 13-0 of the command are gated from B to the Adder Unit, a "one" is forced to Adder Unit bit 14 by G1UB14. The result is then transferred from the Adder Unit to $I_{A,13-0}$. In this manner, bit I_A is forced to a "one".

During Sequence Control State 2, memory is addressed from I_{17-15} (G1SAMX) and the contents of the addressed X cell are gated to the B Register (D1BMEM). The contents of the X cell are then gated from the B Register to the Adder Unit (UBBU). At the same time, the contents of $I_{A,13-0}$ are gated to the Adder Unit (UILA). The Test flip-flop is unconditionally cleared by the TXH command during Time 4 Envelope. At the Clock pulse of Memory Release (MXD1MRLS), the Test flip-flop is set if the summation of B (contents of X) and $I_{A,13-0}$ (-K) results in a carry output from bit 14 of the Adder Unit.

The following examples are provided to illustrate that only when the contents of the addressed X cell are equal to or greater than the value K, is the carry output of Adder bit 14 true. For simplicity, a 5 bit K value and a 6 bit X value is used. Assume the most significant bit of the summed values to be PAU_{14}.

Word Times.	2 (S1,S2)
Interruptable Following Execution?	No
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	C (P) +1
F1WPMT	
F1UOFL	
F1ETST	Set if K ≤ C (X) Reset if K > C (X)
J_{4-0}	
Memory X	

COMMAND CHARACTERISTICS

(a) X = K = 5:

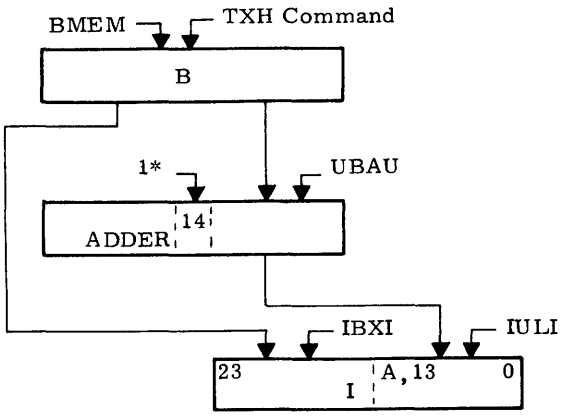
2's complement of K = 1011
Forced "one" bit = 1
C (X) = 00101
00000

PAU_{14} Carry = Test flip-flop set.

(b) K = 5:
X = 4:

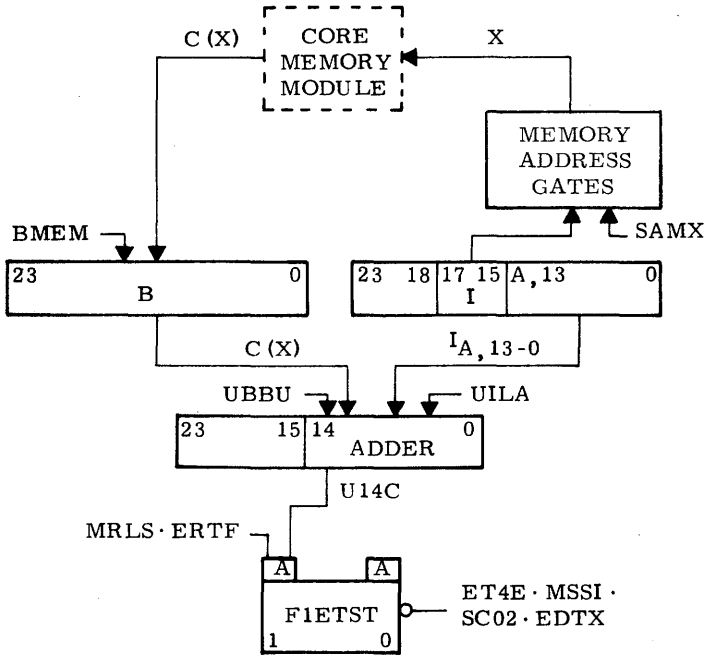
2's complement of K = 1011
Forced "one" bit = 1
C (X) = 00100
11111

PAU_{14} $\overline{\text{Carry}}$ = Test flip-flop remains cleared.



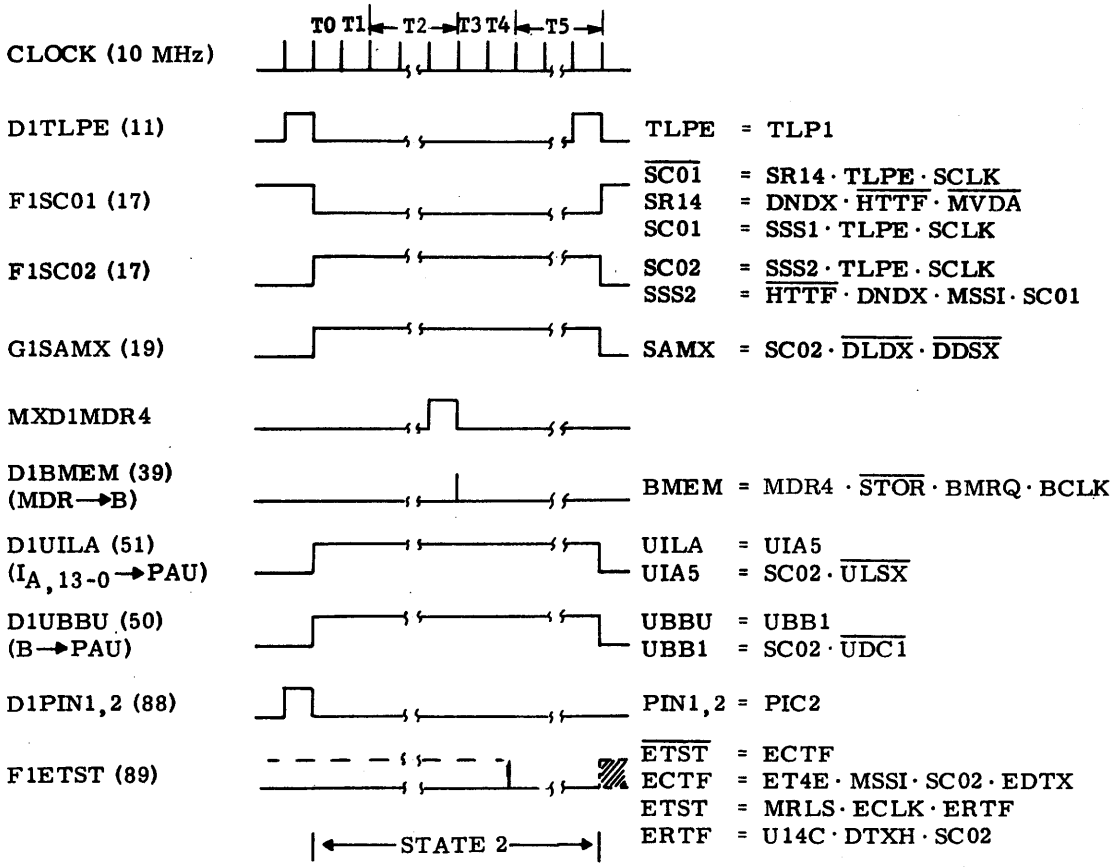
* G1UB14 (49) = ETXN · $\overline{\text{UIXR}}$ · USIA

Sequence State 1



Sequence State 2

TXH BLOCK DIAGRAM



TXH TIMING DIAGRAM

XEC - EXECUTE

FULL OPERAND

23 18	17 15	14	13 0
04	X	*	Y

Z = f (X, *, Y)

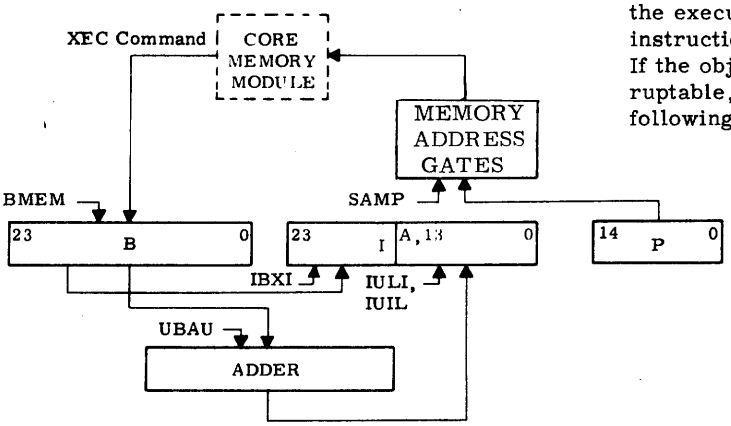
XEC indicates the address Z of the next instruction to be executed. Program control does not change, that is, the P Register is not incremented and the program continues in sequence after executing the instruction located at the effective operand address. All instructions including XEC, may be executed. If the object instruction (contents of cell Z) is relative addressed, the effective operand of the object instruction is computed from the location of the object instruction rather than from the contents of the P Register.

Two successive Sequence Control State 1's are required to illustrate the operation of the XEC command. During the first State 1, a normal "fetch" cycle occurs, except that at Last Pulse Envelope the Execute flip-flop (F1XEXC) is set. During the next State 1, memory is addressed from $I_{A,13-0}$ (D1SAMI) and the normal "fetch" cycle for the object instruction occurs except that at Last Pulse the Execute flip-flop is cleared. Sequencing then continues to execute the object instruction.

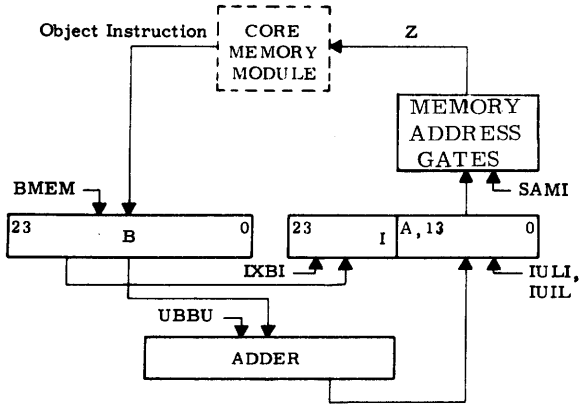
Non-Indexed Word Times.	1 (S1)
Interruptable Following Execution?	No *
CHANGES FOLLOWING EXECUTION	
A_{23-0}	
Q_{23-0}	
P_{14-0}	
F1WPMT	
F1UOFL	
F1ETST	
J_{4-0}	
Memory Z	

COMMAND CHARACTERISTICS

* No interrupts may occur between the execution of XEC and its object instruction located in location Z. If the object instruction is interruptable, an interrupt may occur following its execution.

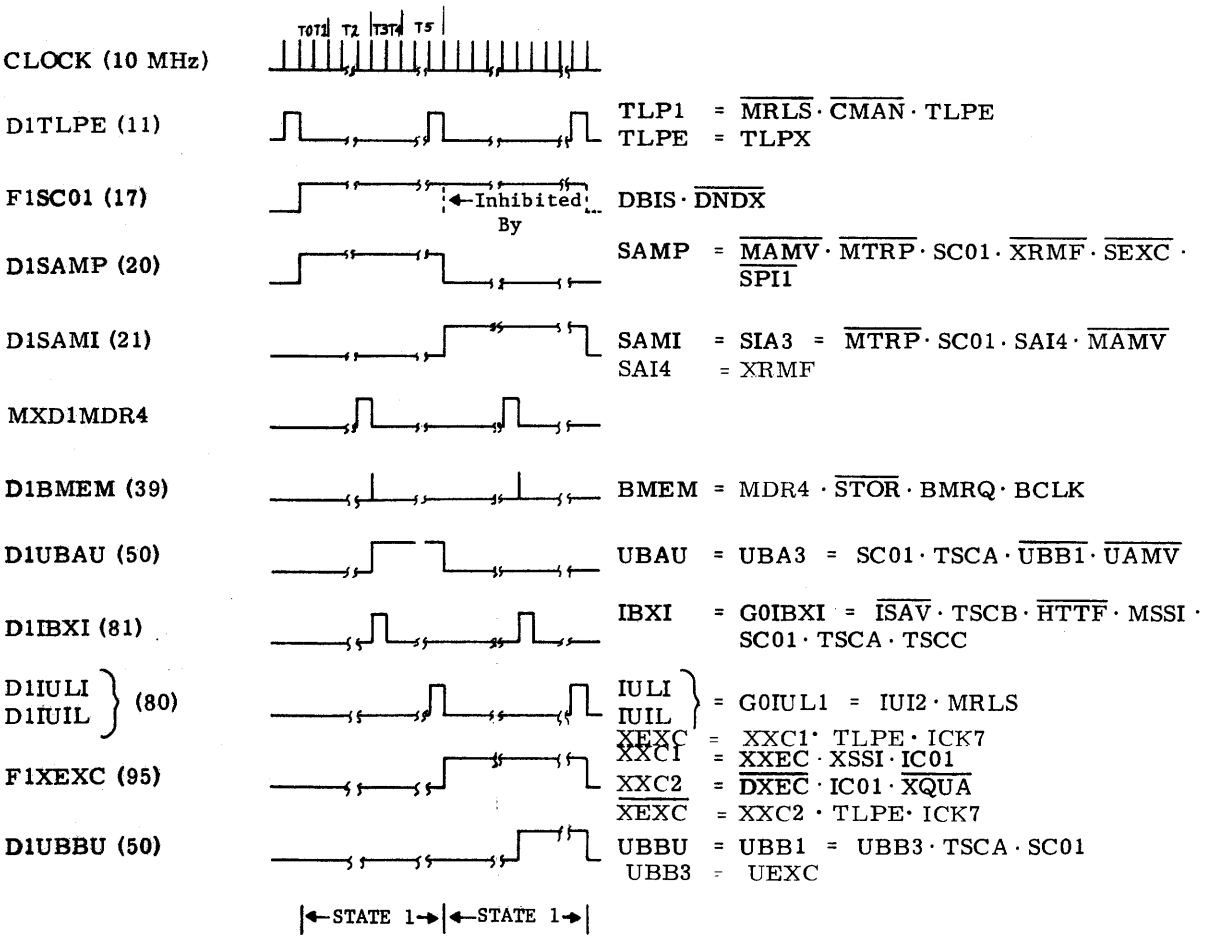


First Sequence State 1



Second Sequence State 1

XEC BLOCK DIAGRAM



XEC TIMING DIAGRAM

OPTIONS

The Central Systems Unit options contained by a particular system are specified on the "Central Systems Unit" drawing, 4DP4903BSID.

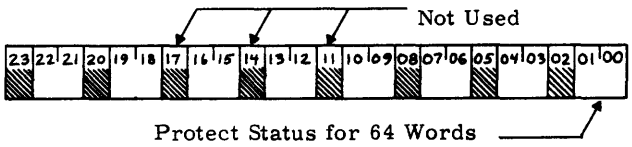
QUADRITECT MEMORY PROTECTION

The optional (4DP4800AS08) GE-PAC 4010 method of memory protection is a combination hardware and software technique where memory is protected in 64-word blocks. The programmer determines which of four possible classes of protection is desired for each 64-word block. The programmer then prepares a table of words to define the class of protection for each 64-word block and the hardware checks the instructions in the program against the table of status words for any illegal operations. If an illegal operation is detected, the contents of the I Register are stored in core memory location 21_8 and program control is transferred to memory location 20_8. Using the Memory Protect option, on-line debugging of programs may be performed without destroying information outside its own area or disturbing the rest of the system.

The Memory Protect mode is active only when the Trapping Mode flip-flop (F1MTRM) is set. F1MTRM is set by the GEN 2 instruction STMF, (25000001_8), or by an LDP or LPR command with bit 19 = 1. The Trapping Mode flip-flop is reset by an SPB command executed as the result of a trap error, by an SPB command in an interrupt response address, or by the Memory Protect (M/P ENBL) switch.

Protect Status

The programmer sets up a table of "Protect Status Words". Memory locations 100 through 177_8 have been allocated for these words. Each Protect Status Word in the table controls the protect status of eight consecutive 64-word memory blocks or 512 words. Each Protect Status Word is divided into 8 octal fields. The two least significant bits of each field contain the Protect Status information for one 64-word memory block. The most significant bit of each octal field is unused, as shown below.



Protect Status Word

The Protect Status information (2 bits) defines one of four types of protection for the associated 64-word block.

11 The 11 code indicates that the associated 64-word memory block is part of the running program. The running program may read, write, or branch into memory blocks that are coded 11. Only GEN 2 instructions are trapped.

01 The 01 code indicates that the associated 64-word memory block is a "Read Only" data area. The program may branch to or read from memory block so coded but will trap when a write or store operation is attempted in the block. Data may however, be stored in these areas by a relative addressed instruction within a quasi subroutine and data may always be stored in location $0\text{-}17_8$. An instruction addressed by an Automatic Program Interrupt may be fetched and executed without being trapped. All GEN 2 instructions within 01 blocks are trapped.

10 The 10 code indicates that the associated 64-word memory block is a read or write data area. The program may read or write in memory blocks so coded. Instructions may not be fetched and executed from this area except:

1. Fetching the object instruction of XEC is permitted.
2. Fetching the instruction specified by the OP code of a Quasi instruction being executed is permitted.
3. No trap occurs on any instruction addressed (200-377) by an Automatic Program Interrupt.

00 The 00 code indicates that the associated 64-word memory block is inaccessible to the running program. Any attempts by the running program to use this area will cause a trap error except:

1. Fetching of any instruction within a Quasi subroutine is permitted.
2. Relative addressed instructions within a Quasi subroutine may access this area during execution.
3. Fetching the object instruction of XEC is permitted.
4. No trap occurs when addressing memory locations $0\text{-}17_8$ during instruction execution.
5. No trap occurs on fetch or execute of any instruction addressed (200-377) by an Automatic Program Interrupt.

Table OPT. 1 lists (a.) the trapping conditions during the fetch of an instruction and (b.) the trapping conditions during execute of each instruction.

As previously mentioned, the Memory Protect Status Words (MPSW) are stored in memory locations 100_8 through 177_8. These 100_8 locations are sufficient to provide Memory Protect Status Words for up to $32,767_{10}$ memory locations. The following table indicates the MPSW location and the field bits associated with the 64-word memory blocks.

Instruction	From Memory Locations Assigned Status Bits	Trapped/ Not Trapped
API Response Instruction	00 01 10 11	Not Trapped Not Trapped Not Trapped Not Trapped
XEC Object Instruction	00 01 10 11	Not Trapped Not Trapped Not Trapped Not Trapped
Quasi Subroutine Instructions	00 01 10 11	Not Trapped Not Trapped Not Trapped Not Trapped
GEN 2	00 01 10 11	Trapped Trapped Trapped Trapped
Trap Error Instruction (20_8)	00 01 10 11	Not Trapped Not Trapped Not Trapped Not Trapped
All Others	00 01 10 11	Trapped Not Trapped Trapped Not Trapped

a. Instruction Fetch Trapping

Instruction	Status Bits of Effective Operand Address Location	Trapped/ Not Trapped
STX, DMT, STQ, STA	00 01 10 11	Trapped* Trapped* Not Trapped Not Trapped
LDX, DVD, MPY, LDA, ERA, ADD, LDP, ANA, ORA, LDX, SUB, LPR, LOQ	00 01 10 11	Trapped* Not Trapped Not Trapped Not Trapped
GEN1, GEN3 LXK, LXC, TXH, INX, Quasi, Relative Addressed Instructions within Quasi subroutine	00 01 10 11	Not Trapped Not Trapped Not Trapped Not Trapped
Instructions within Quasi subroutine that are not relatiave addressed	00 01 10	Instructions are subject to trapping during execution as listed above.

*If the effective address of the store or load function is within 00 thru 17_8, the instruction is not trapped.

NOTE
Since the execution of BRU, BTR, BTS, XEC and SPB is fetching the next instruction, refer to Table A for trapping conditions.

b. Instruction Execute Trapping

Table OPT 1. Trapping Conditions

MPSW Location	MPSW Field Bits	Core Block Protected (Octal)
100_8	1- 0	0- 77_8 *
100_8	4- 3	100- 177
100_8	7- 6	200- 277
100_8	10- 9	300- 377
100_8	13-12	400- 477
100_8	16-15	500- 577
100_8	19-18	600- 677
100_8	22-21	700- 777_8
101_8	1- 0	1000-1077_8
101_8	4- 3	1100-1177
101_8	7- 6	1200-1277
101_8	10- 9	1300-1377
101_8	13-12	1400-1477
101_8	16-15	1500-1577
101_8	19-18	1600-1677
101_8	22-21 etc.	1700-1777_8

*0-17_8 will be treated as a 11 area when the protect status word is 01 or 11. When the protect status word is 00 or 10, locations 0-17_8 will be treated as a 10 area.

Quadritect Rules

The following rules and general functions apply to the Quadritect Memory Protection feature of the 4010 system.

- GEN 2 instructions are always trapped when memory protect is active.
- The object instruction of XEC may be fetched but is subject to trapping before execution.
- Any API Response instruction may be fetched and executed without being trapped. If the instruction is an SPB, the Trapping Mode flip-flop is cleared, disabling the Memory Protect function.
- The memory protect trap location, cell 20_8, must contain an SPB command to an error routine. Location 1 will then contain the address of the offending instruction.
- When a trap occurs, the contents of the I Register are stored in location 21_8.

  If the trap occurs because of an illegal fetch that does not result from a branch instruction, the I Register will contain the last executed instruction. If that instruction is a branch, I will contain the branch instruction but the branch instruction will not have been fully executed (i.e. the P Register will not have changed).

  If the trap is caused by an illegal execution, the I Register will contain the illegal instruction.
- Any instruction can be fetched from any location when the instruction is within a Quasi subroutine. Relative addressed instructions within the Quasi subroutine may be executed regardless of the protect status.
- Memory locations 00 thru 17_8 include the index registers and the Q Register and are not subject to trapping during instruction execution.

Theory of Operation

A flow chart of the basic operations performed in the Quadritect Memory Protect Mode is contained in Fig. OPT. 1. Refer to the flow chart during the following discussion.

The Memory Protect Mode is active only when the Trapping Mode flip-flop (F1MTRM) is set. F1MTRM is set by the GEN 2 instruction STMF (25000001_8) or by an LDP or LPR command with bit 19 set. F1MTRM is cleared by an SPB executed from cell 20_8 as a result of a trap error, by an SPB in an interrupt response address, by an SPB as the result of a watchdog trap, or by the M/P ENBL switch in the up position. The SPB command stores the status of F1MTRM in bit 19 of cell 1.

F1MTRM (119) = MSTM · TLPE · MCK2

G1MSTM (120) = (MF16 + MF15) · SC04

G0MF16 (120) = STMF

G0MF15 (120) = SPB · Bit 19

$\overline{\text{F1MTRM}}$ (119) = MMPL + MRTM · MCK2

G1MRTM (120) = SC04 · TSCA · DSPB · (MTRP + SP12)

G0MMPL (119) = M/P ENBL switch in disable position

Two registers are used by the memory protect. One register is used to hold the Memory Protect Status Word (L Register) and the other register (V Register) is used to determine which Memory Protect Status Word is currently in the L Register.

The V (Volume) Register is six bits in length. The contents of the V Register correspond to the most significant 6 bits of the address of the memory block for which the Protect Status Word is contained in the L Register. These 6 bits define one 512 word block of memory. As long as memory locations within this block are addressed, the Memory Protect Status Word contained in the L Register is applicable. Each time memory is addressed from the P or I Registers, the contents of the six most significant bits (14-9) of the Memory Address Gates are compared with the contents of the V Register. If these bits differ, a "volume difference" exists, and the contents of Memory Address Gates D1MA14-9, are transferred to the V Register. The V Register then addresses memory via D1MA05-00 with D1MA06 enabled. In this manner, the Protect Status Word for the current 512 word memory block is addressed (i.e., a location between 100 and 177_8 is addressed).

The following examples are provided to help clarify the function of the V Register.

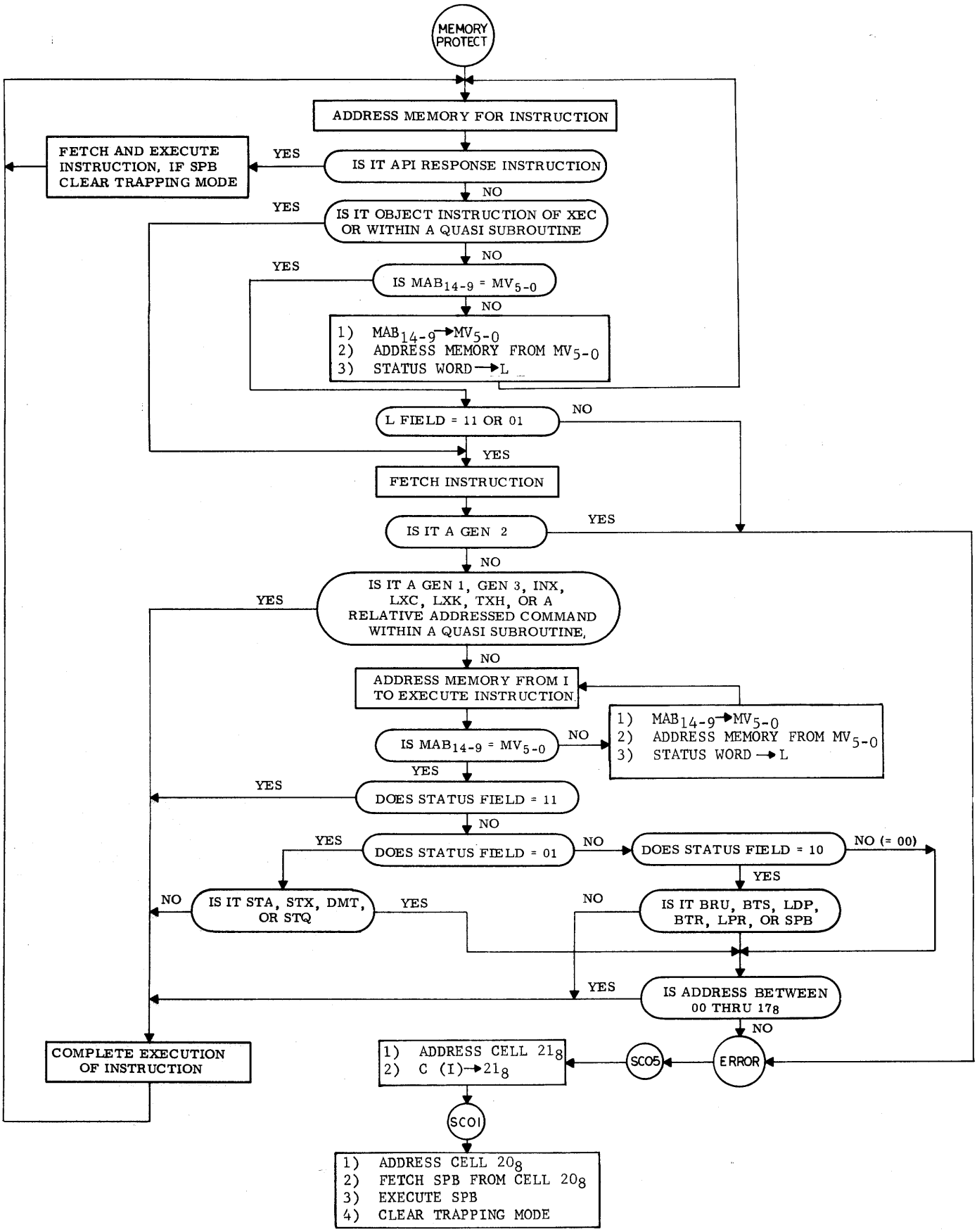


Fig. OPT. 1 Quadritect Flow Chart

Consider:

D1MA14-00 =

14 13 12 11 10 09 08 07 06 05 04 03 02 01 00

0 0 0 0 0 0 1 1 1 1 1 1 1 1 1

7 7 7

V Reg. = 05 04 03 02 01 00 | 0 0 0 0 0 0 | = Defines first 512 word block.

L Reg. = 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00 | 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 |

These 2 bits are used to determine the protect status of the memory address above

This corresponds to the bits used for the MPSW in cell 100_8. This MPSW applies to address $0\text{-}777_8$.

D1MA14-09 are equal to the contents of the V Register. Therefore, the Memory Protect Status Word contained in the A Register is correct for this address.

Now, consider that the next memory cell is addressed.

D1MA14-00 =

14 13 12 11 10 09 08 07 06 05 04 03 02 01 00

0 0 0 0 0 1 0 0 0 0 0 0 0 0 0

1 0 0 0

V Reg. = 05 04 03 02 01 00 | 0 0 0 0 0 0 |

Now, D1MA14-09 = V_{5-0}. Therefore, a Volume Difference exists and D1MA14-09 is gated to V_{5-0}.

V Reg. = 05 04 03 02 01 00 | 0 0 0 0 0 1 |

V_{5-0} is gated to D1MA05-00 and D1MA06 is enabled.

D1MA14-00 =

14 13 12 11 10 09 08 07 06 05 04 03 02 01 00

0 0 0 0 0 0 0 0 1 0 0 0 0 0 1

1 0 1

Memory cell 101_8 is addressed and the Memory Protect Status Word contained in cell 101_8 is transferred to the L Register.

Figs. OPT. 2 and OPT. 3 contain a block and timing diagram of the operation performed to fetch a new Memory Protect Status Word and gate it to the L Register.

The contents of the Volume Register (F1MV05-00) are compared with bits 9 through 14 of the Memory Address Gates in G1MCM1, 2, 3. If any of the bits differ, G1MCM1, 2, or 3 is disabled. With G1MCM1, 2, or 3 disabled, G1MVDF is enabled if: (1) memory is being

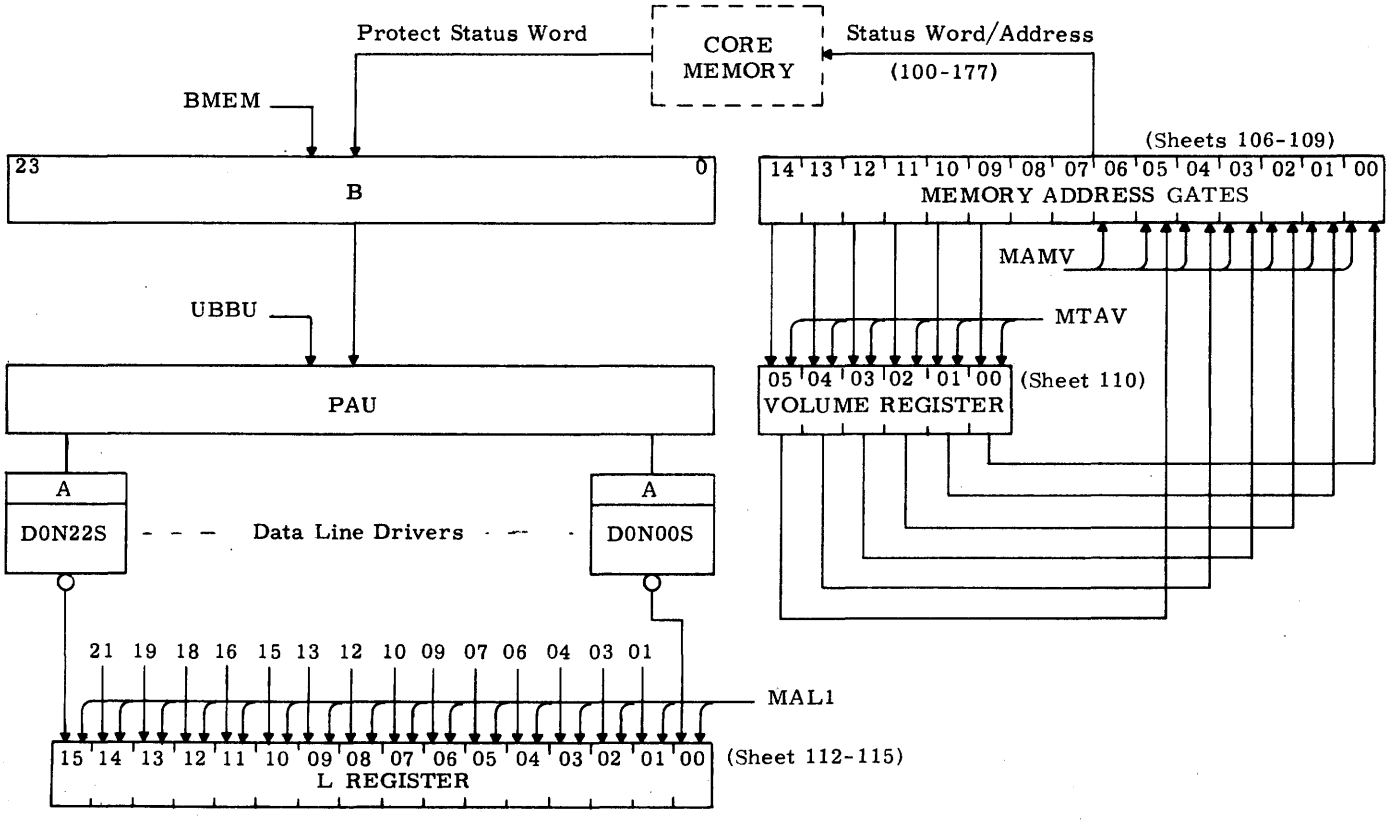


Fig. OPT. 2 Fetch Protect Status Word Block Diagram

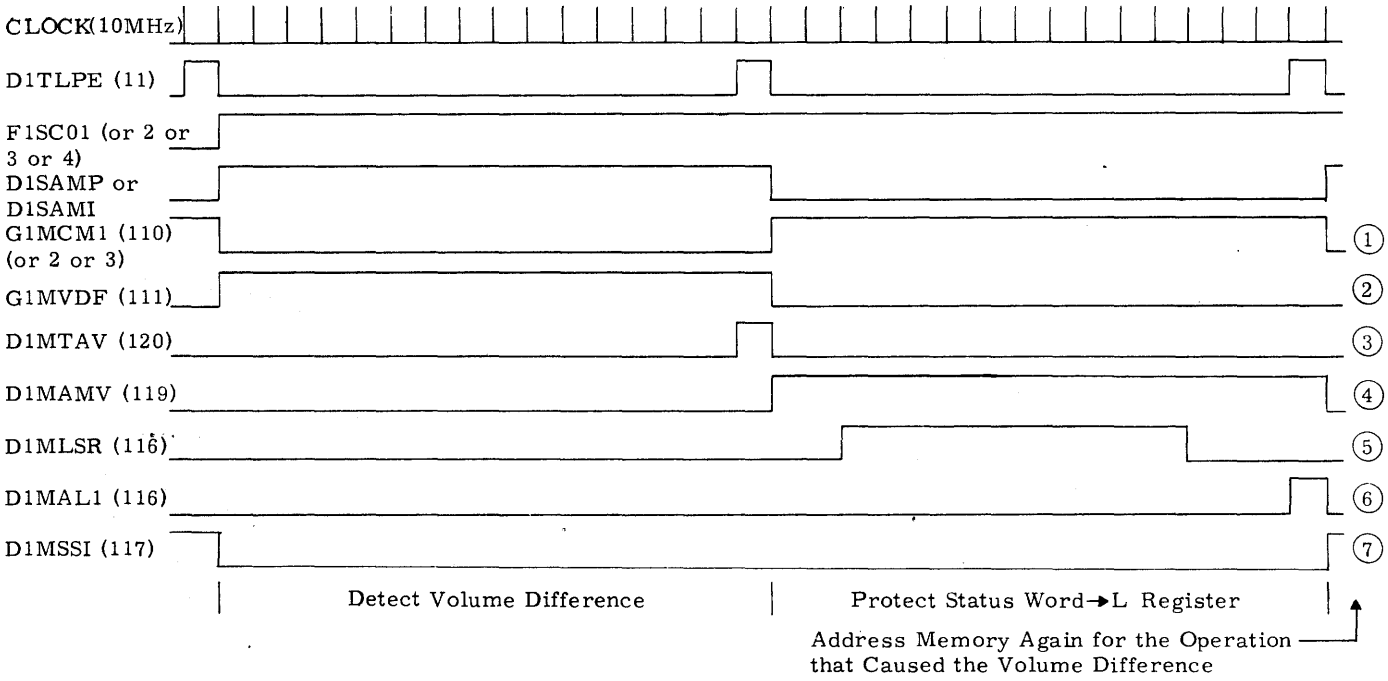
addressed during State 1 and is not the result of an API, Trap Error, Volume Difference, the object instruction of XEC, or within a Quasi subroutine or, (2) memory is addressed from the I Register and it is not State 1, the command is not relative addressed within a Quasi subroutine, and it is not the result of an API or Trap Error. When G1MVDF is enabled, a Volume Difference exists indicating that a new Memory Protect Status Word must be gated to the L Register to specify the protect status of the memory location being addressed.

G1MVDF is also enabled immediately following the setting of the Trapping Mode flip-flop (F1MTRM).

$$G1MVDF = \overline{MB09} \cdot MV00$$
$$D0MB09 = MTMC$$
$$F1MV00 = \overline{MTRM}$$
$$F1MTMC = MSTM \cdot \overline{MTRM} \cdot TLPE \cdot MCKL$$

G1MVDF, when enabled, inhibits AU operation and holds sequencing in the present Sequence State (D0MSSI and D1MSSI). At Memory Release, Memory Address Gate bits 14 through 9 are gated to the Volume Register (D1MTAV). Memory is then addressed from the Volume Register (F1MAMV) with Memory Address Gate bit 6 enabled. This addresses one of the locations between 100_8 and 177_8 that contains the Memory Protect Status Word for the memory location desired. The Memory Protect Status Word is "fetched" from memory, gated to the B Register, and then to the L Register via the Parallel Adder and Data Line Drivers (D1MAL1).

After the proper Memory Protect Status Word is contained in the L Register, sequencing is held for one more word time, during which the operation that caused the Volume Difference is again attempted. This time, the operation is allowed to complete provided a fence violation (error) does not occur. This detection is described in the following paragraphs.



① $\overline{G1MCM1} = (MA14 \cdot \overline{MV05}) + (MV05 \cdot \overline{MA14}) + (MA13 \cdot \overline{MV04}) + (MV04 \cdot \overline{MA13})$
$\overline{G1MCM2} = (MA12 \cdot \overline{MV03}) + (MV03 \cdot \overline{MA12}) + (MA11 \cdot \overline{MV02}) + (MV02 \cdot \overline{MA11})$
$\overline{G1MCM3} = (MA10 \cdot \overline{MV01}) + (MV01 \cdot \overline{MA10}) + (MA09 \cdot \overline{MV00}) \; (MV00 \cdot \overline{MB09})$

② $G1MVDF = (SC01 + MF14)\,(MF14 + \overline{MXAQ})\,(\overline{MCM1} + \overline{MCM2} + \overline{MCM3})\,(MF13)$
$G0MF14 = SAMI \cdot MF17 \cdot \overline{SC01} \cdot \overline{SPI2}$
$G0MF13 = \overline{MTRP} \cdot \overline{MAMV} \cdot MTRM \cdot \overline{SPI1}$

③ $D1MTAV = MVDF \cdot MRLS$

④ $D1MAMV = F1MAMV$
$\overline{F1MAMV} = MVDF \cdot TLPE \cdot MCK2$
$\overline{F1MAMV} = MCK2 \cdot TLPE + \overline{MTRM}$

⑤ $D1MLSR = MAMV \cdot TSCB$

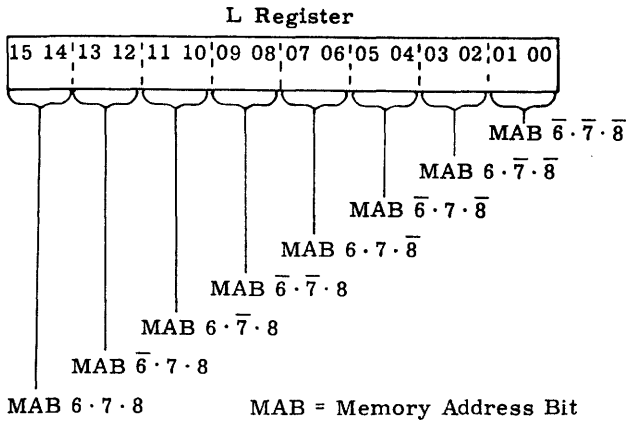
⑥ $D1MAL1 = MAMV \cdot MRLS$

⑦ $\overline{D1MSSI} = MVDF + MAMV$

Fig. OPT. 3 Fetching Protect Status Word Timing

After the proper Memory Protect Status Word is contained in the L Register, it must be determined which of the field bits specify the protect status for the memory location currently being addressed. Memory Address bits 6, 7, and 8 are used to determine which group of 64 words is currently being addressed within the 512 word block for which the Memory Protect Status Word is contained in the L Register.

Selection of the proper L Register field bits and decoding the value of these bits occurs in G0M0X1, G0M01X through G0M3X1, G0M31X and G1ML00, G1ML01, G0ML11, and G0ML1X.



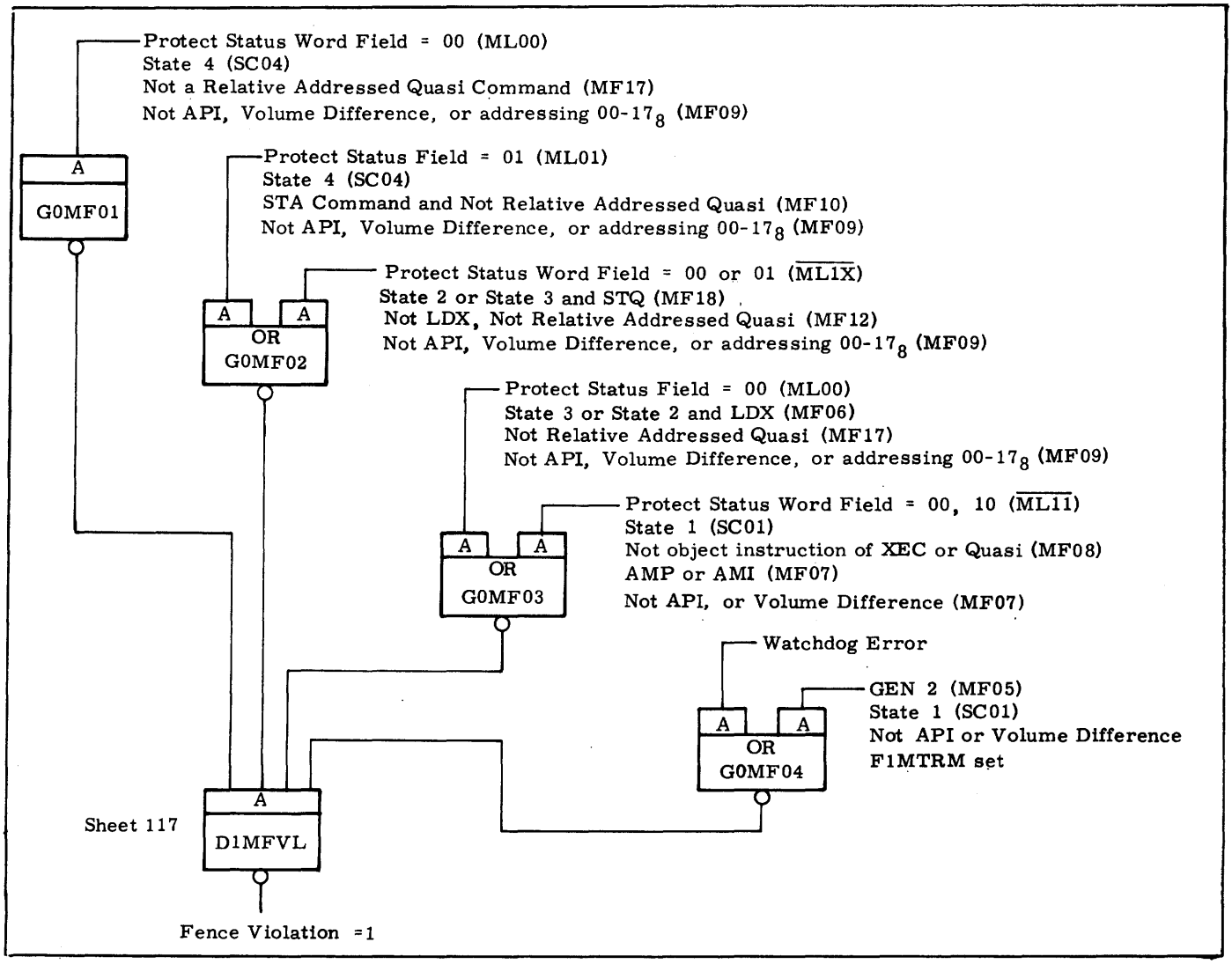
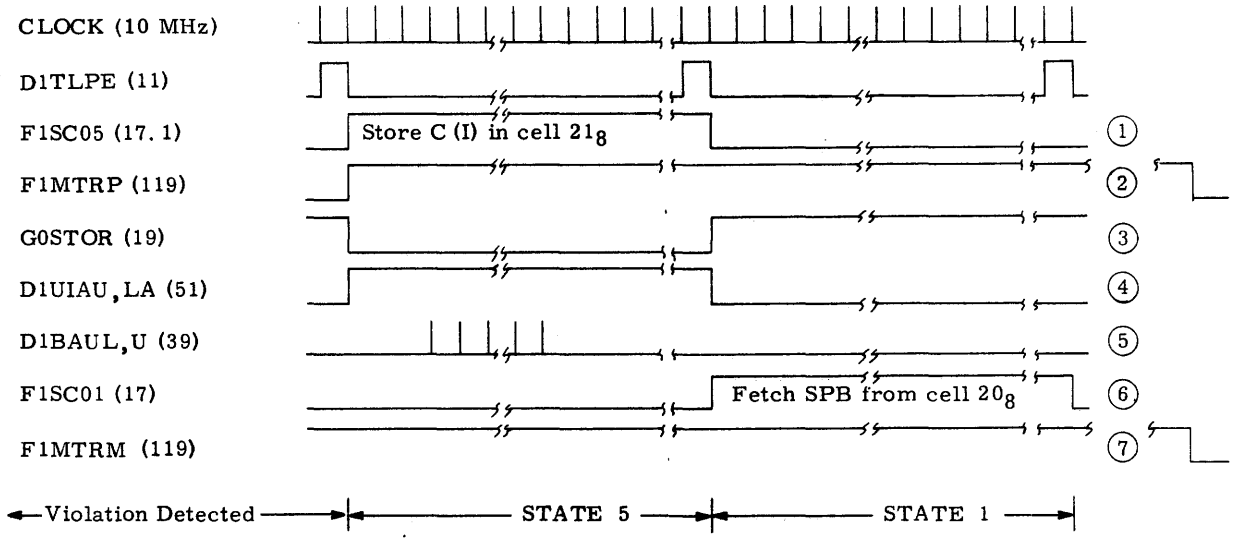


Fig. OPT. 4 Fence Violation Block Diagram

Selected Protect Status Code	Gates Enabled (Sheet 116)
11	G0ML11, G0ML1X
10	G0ML1X
01	G1ML01, G0ML11
00	G1ML00

Using these decoded Status Code signals, a Fence Violation is checked in G0MF01, 02, 03, and 04. Fig. OPT. 4 contains a block diagram of the detection logic and indicates the function of each gate and the respective input control signals. Using this block diagram, little difficulty should be encountered in determining which gate is used to detect a Fence Violation for any particular command, status code, or Sequence State. Enabling any of these gates (G0MF01, 02, 03, or 04), indicates a Fence Violation (D1MFVL).

A timing diagram, with logic equations of the operation following a Fence Violation, is contained in Fig. OPT 5. When a Fence Violation is detected, operation of the AU is inhibited during the current Sequence State (D0MSSI) as shown in Fig. OPT 6. Store operations (G0STOR) are inhibited if there is a fence violation (D0MFVL). To prevent any timing problem and allow a store to occur during a relative addressed command with a quasi subroutine, G1MF17 enables G0STO2. When the current memory cycle is completed, Sequence State 5 is enabled. Memory location 21_8 (G0MX04, G0M00) is addressed and the contents of the I Register are stored in this location. This operation places the instruction that caused the Fence Violation in cell 21_8 where the error routine may examine the command. Following State 5, State 1 is entered and memory location 20_8 (G0MX04) is addressed and the SPB command for the error routine is gated to memory. This SPB command stores the contents of P in location 1, clears the Trapping Mode flip-flop and enables the error routine to be executed.



① F1SC05 = MFVL · TLPE · SCLK

② $\overline{\text{F1MTRP}}$ = MFVL · TLPE · MCK2
F1MTRP = MRTM · TLPE · MCK2

③ G0STOR = SC05

④ D1UIAU,LA = UIA4 = SC05 · MTRP

⑤ D1BAUL,U = BAU1 = BAU3 · BCLK · $\overline{\text{TSCA}}$ · TSCB

⑥ F1SC01 = SC05 · TLPE · SCLK

⑦ $\overline{\text{F1MTRM}}$ = MRTM · MCK2

Fig. OPT. 5 Fence Violation Timing

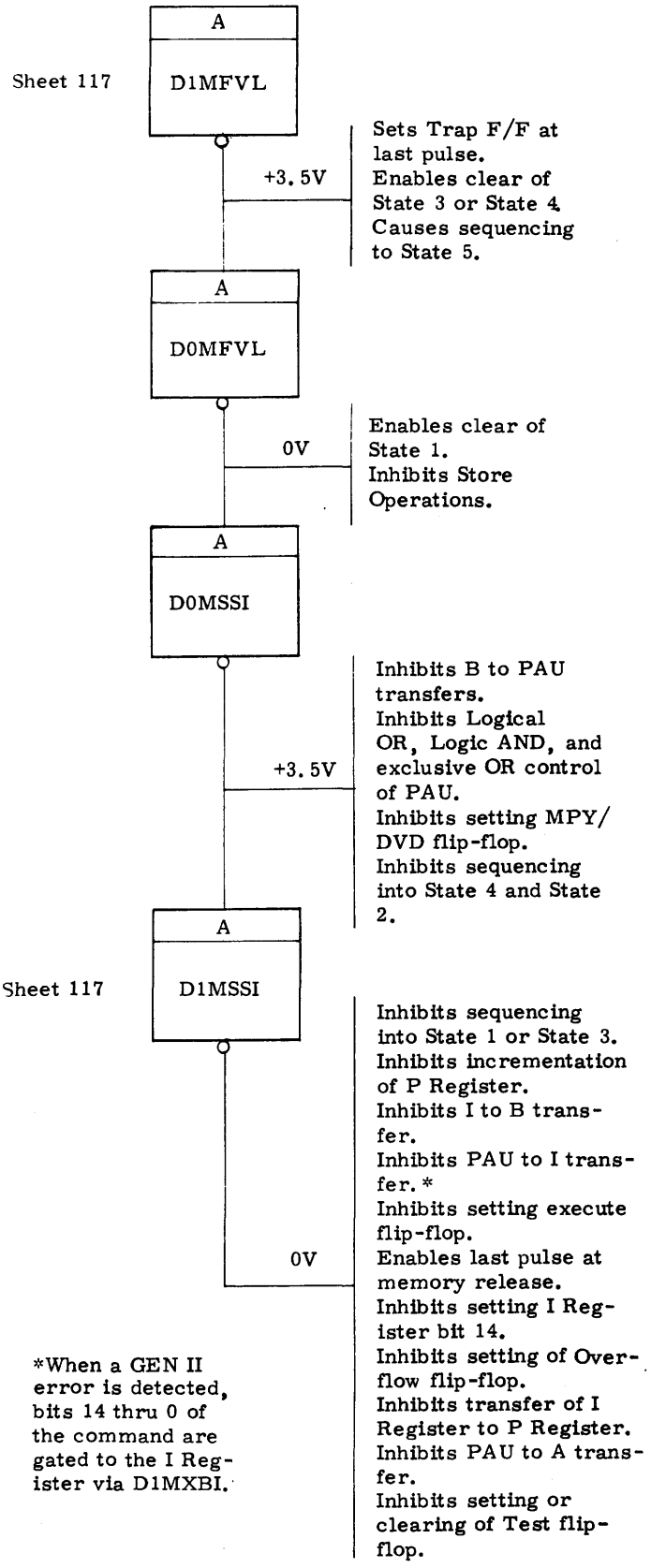


Fig. OPT 6 Fence Violation, Arithmetic Unit Control

API WATCHDOG

API watchdog timer is included in the Quadritect (4DP4800AS08) option. It provides hardware protection against the possibility of programming errors which inhibit the servicing of interrupts. Examples of programming errors detected by the Watchdog Timer are: (1) a program loop of non-interruptable instructions such as branch to self, (2) a program that inhibits all interrupts by the IAI_2 instruction for a lengthy period of time or (3) a program that resets the PAI flip-flop and/or (4) continuous interrupts that are not serviced within an adjustable time period. If an error is detected by the Watchdog Timer, the contents of the I Register are stored in location 21_8 and program control is transferred to the instruction in location 24_8 only. The API STALL, and ALARM CLEAR indicators on the console are lighted.

Following a trap error, memory location 21_8 contains the following data:

Bits	
23 - 14	Op code of the next instruction that would have been executed if a trap error had not occurred.
13 - 0	The address portion I_{13-0} of the I Register as used by the last instruction executed prior to the trap error.

Location 24_8 must contain an SPB instruction to an error subroutine. Identification of the type of error detected is stored in bit 18 of location 1 by the SPB instruction.

Bit 18 = 1	Permit Automatic Interrupt flip-flop reset, inhibiting inhibitable interrupts for too long a period of time.
Bit 18 = 0	All interrupts are locked out for too long a period of time by the IAI_2 command or by a long sequence of non-interruptable commands (e.g. BRU*)

The Watchdog alarm condition is cleared by the SPB instruction in location 24_8, by the API ENBL switch in the lockout position, or by pressing the ON switch to initialize.

The API Stall Alarm does not provide hardware protection against program misuse of the API Mask Register delaying servicing of API's for extended periods of time. The very nature of the application and purpose of the register makes it impossible to define hardware protection against misuse. Programs which control the Mask Register must be known to be operational. Any instruction which attempts to alter the Mask Register will be trapped by Memory Protect.

The three conditions that are checked by the Watchdog circuitry are described below:

- Checks program misuse of the capability to temporarily lockout all automatic program interrupts by chaining together long sequences of non-interruptable instructions (e.g. a BRU* sequence). This check is performed only when all interrupts are not inhibited by the IAI_2 command.

  A timer circuit that is optionally wired to time out at approximately 300 microseconds is started when any interrupt request that is not masked out or inhibited by IAI is present. The timer is disabled and reset when the Arithmetic Unit services this interrupt. An error, therefore, exists when no interrupts are serviced within the 300 microsecond time period.

- Checks program misuse of the IAI_2 instruction (25000304_8). An error occurs if this instruction is not followed by a PAI instruction (25020000) within the 300 microsecond time period.

  Executing the IAI_2 instructions starts a timer. The timer continues to run until a PAI instruction is executed. The PAI instruction disables the timer and resets it. If the timer is allowed to time out, an error is detected.

- Checks for program misuse of the capability to inhibit inhibitable interrupts for too long a period of time. Inhibitable interrupts are inhibited by clearing the Permit Automatic Interrupt flip-flop using the IAI, SPB, LDP, or LPR instruction.

A timer, set according to system requirements between 25 and 250 milliseconds (normally set to 100 milliseconds), is started whenever the PAI flip-flop is reset or when any interrupt that is not masked out is waiting to be serviced. The timer is disabled and reset, whenever the PAI flip-flop is set and no interrupts (that are not masked out) are waiting to be serviced.

If the timer is allowed to time out, an error is detected. In addition to transferring program control to location 24_8, the SPB command in location 24_8 will store a "1" in bit 18 location 1 indicating this type of error.

Operation of the Watchdog Timer is inhibited when the console is enabled and either the STALL ENBL or API ENBL switch is in the lockout position.

> NOTE
>
> Prior to switching the CONSOLE ENABLE switch to the disable position, perform the following procedure:
>
> 1. Place the AUTO/MAN switch to AUTO.
> 2. Place the API ENBL switch to the enable position.
> 3. Place the STALL ENBL switch to the enable position.
> 4. Place the CONSOLE ENABLE switch to the disable position.

Non-Interruptable Instruction Sequence

Fig. OPT 7 contains a block diagram and Fig. OPT 8 contains a timing diagram of the Watchdog circuits used in detecting an error caused by a long non-interruptable sequence of instructions. Such a sequence of instructions disable G1WENA, thereby, inhibiting the servicing of any interrupt requests. Examples of instructions that inhibit G1WENA are BRU, BTS, BTR, TXH, CLO, CLZ, CMZ, CMO, LPR, LDP, SPB, JNO, JNR, JDR, JCB, LDX, IAI, and Quasi. The time duration that G1WENA may be inhibited is wired at approximately 300 microseconds. This test is valid, only if all interrupts are not inhibited by the IAI_2 instruction or by the mask register.

As previously mentioned, the 300 microsecond timer circuit is started when any interrupt request has been recognized by the API module. The timer circuit is in effect reset and disabled when the interrupt has been serviced. Therefore, if any interrupt is serviced within the preset time period, no error is detected.

As shown in Fig. 7 and 8, the output of D0SNIP is at 3.5 volts when any interrupt that is not inhibited by the mask register or IAI_2 is present and has not yet been serviced. This signal in conjunction with the API and STALL ENBL switches in the ENBL position and the ON switch not pressed to initialize, enables G0WIMP. The output of G0WIMP is inverted in G1WIMP applying a "1" input to Delay Circuit 2. Delay Circuit 2 provides a "0" output with a duration of approximately 3 microseconds whenever the circuit input remains at the "1" level for a 300 microsecond time duration. Therefore, if an interrupt is present and no interrupts are serviced within the preset time interval, a "0" output from the delay circuit will be present. This "0" output is inverted and used to set F1WDO1 and F1WDO2.

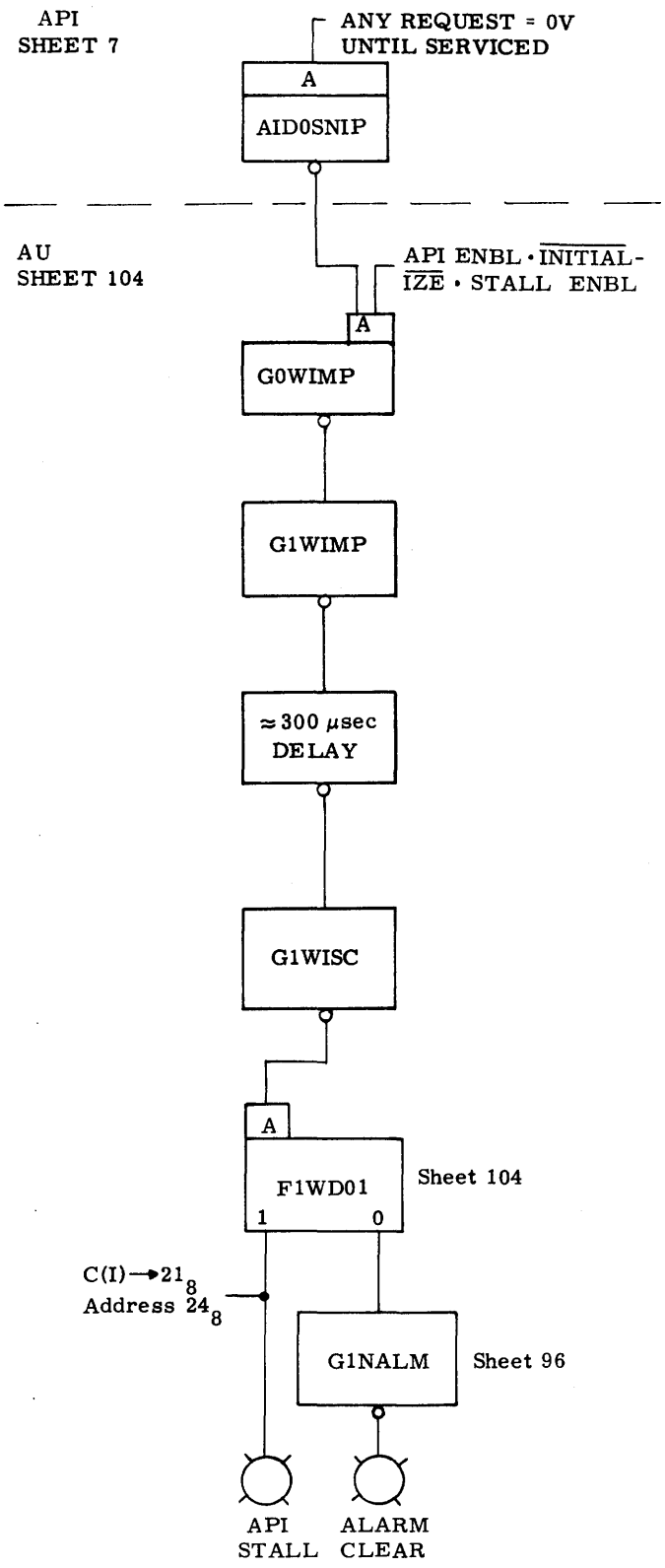


Fig. OPT. 7 Non-Interruptable Instruction, Watchdog Block Diagram

Setting F1WDO1 lights the API STALL, and ALARM indicators and simulates a Memory Protect error to store the contents of the I Register into location 21_8 and transfers program control to location 24_8.

The Memory Protect error is simulated by enabling G0MFO4 during Time 5 of the next State 1. Enabling G0MFO4 enables D1MFVL which forces AU sequencing into State 5. During this State 5, the Trap flip-flop is set and the contents of the I Register are stored in location 21_8. This is accomplished in the same manner as when Memory Protect error is detected.

Following State 5, State 1 is entered. During State 1, memory location 24_8 is addressed. Location 24_8 is addressed in the same manner as location 20_8 is addr ssed following a Memory Protect error except G0MXO2 is also enabled. The SPB to the error routine is fetched from location 24_8. During the execution of this SPB, F1MTRP and F1WDO2 are cleared to complete the cycle.

IAI$_2$ Error Detection

Fig. OPT. 9 contains a block diagram and Fig. OPT. 10 contains a timing diagram of the IAI$_2$ error detection circuits. These circuits detect an error caused by executing an IAI$_2$ instruction and not following this instruction with a PAI instruction within a 300 microsecond time period.

As shown in Fig. OPT. 9, F1ZINH is set when the Permit flip-flop is reset and the IAI$_2$ instruction is executed to inhibit all interrupts. F1ZINH is reset when the Permit flip-flop is set (PAI). Setting F1ZINH enables G1ZIN1 which enables G0WIMP if the STALL and API ENBL switches are in the ENBL position. The output of G0WIMP is inverted in G1WIMP applying a "1" to the input of the 300 microsecond delay circuit. If this "1" input is applied to the delay circuit for the delay period, a "0" output with a duration of approximately 3 microseconds is generated. This "0" output is inverted and used to set F1WDO1 and F1WDO2. Setting F1WDO1 causes the contents of the I Register to be stored in location 21_8 and causes the next instruction to be fetched from location 24_8 as previously described.

Although G1ZIN1 is enabled during the execution of LMR and LMR$_2$ commands, the duration of these signals is of little consequence to the watchdog circuitry. These signals are used for control within the API module and are not associated with Watchdog.

The input to the delay circuit must remain in the "1" state for the entire set time period to provide a "0" output. If at any time the input goes to "0" the delay circuit is disabled and must then receive a "1" input for the entire 300 microseconds time period to provide a "0" output.

Permit F/F Reset

Fig. OPT. 11 contains a block diagram and Fig. OPT. 12 contains a timing diagram of the circuits used to detect an error caused by Permit Automatic Interrupt flip-flop being reset for a system selected time duration of 25 to 250 milliseconds.

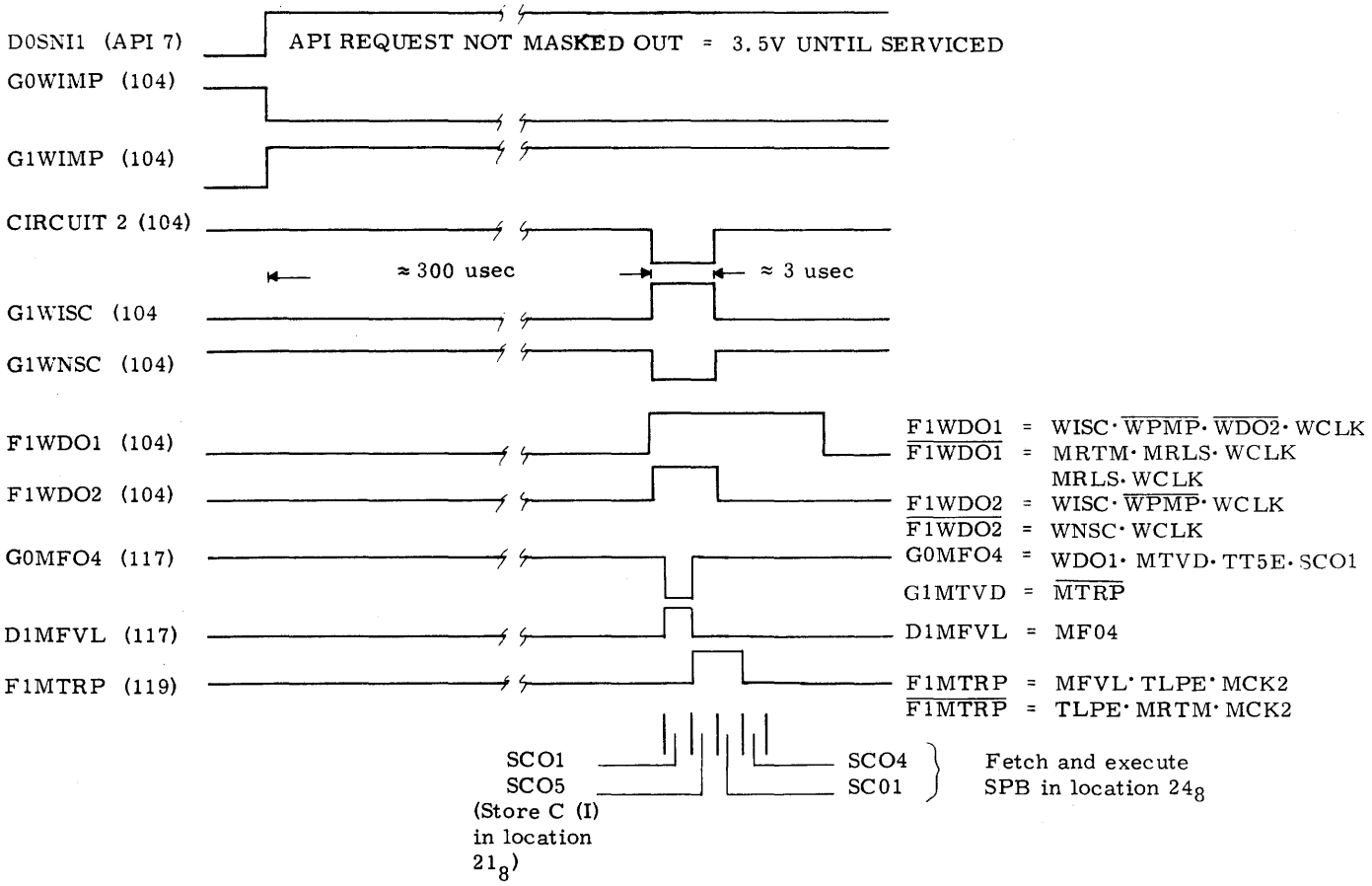


Fig. OPT. 8 Non-Interruptable Instruction, Watchdog Timing

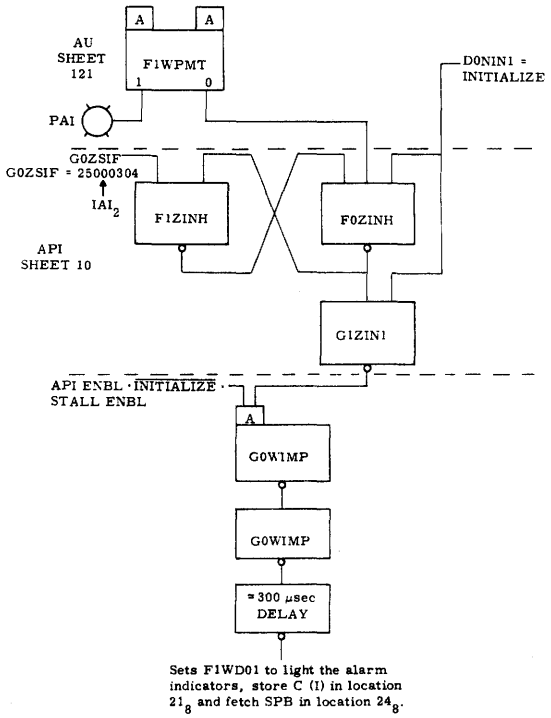


Fig. OPT. 9 IAI_2 Error Detection, Block Diagram

The timing delay circuit is started when the Permit Automatic Interrupt flip-flop is reset. The Permit Automatic Interrupt flip-flop may be reset for too long a period of time inhibiting inhibitable interrupts by misuse of the IAI, SPB, LDP, or LPR instructions. The time delay circuit is disabled and reset when the Permit Automatic Interrupt flip-flop is set and no interrupts that have not been masked out are waiting to be serviced.

> **NOTE**
> Although the timer circuit is enabled whenever an interrupt is waiting to be serviced, it is disabled whenever no interrupts are waiting service if the Permit flip-flop is set. Due to the time duration of the delay circuit (25 to 250 milliseconds) this is of little consequence unless the Permit Automatic Interrupt had been reset for some period of time.

If this type of error is detected, in addition to lighting the alarm indicators, storing the contents of the I Register in location 21_8, and fetching the SPB instruction located in location 24_8, bit 18 of location 1 will be set when the SPB command is executed. Software checks this bit to determine the type of Watchdog error detected. Normally, recovery from this type of Watchdog alarm is accomplished by a software routine.

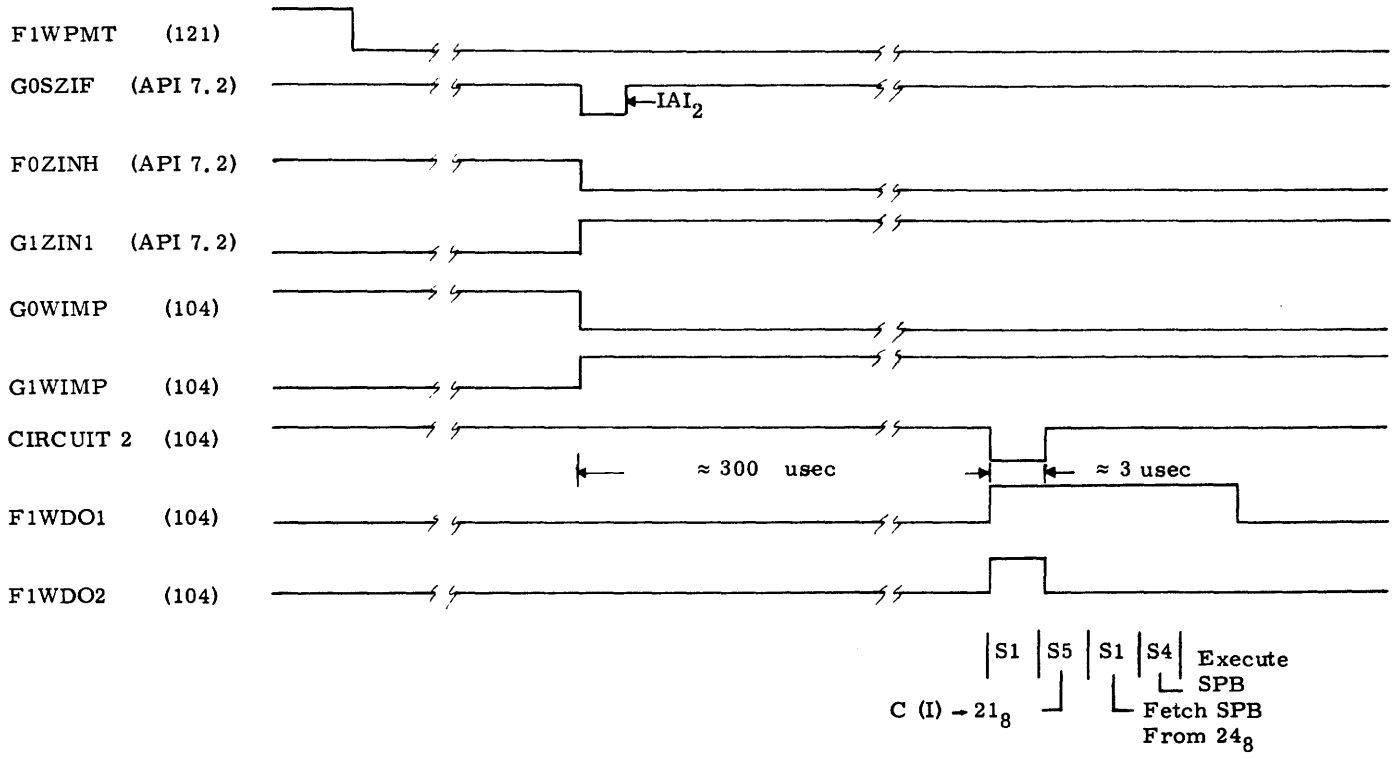


Fig. OPT. 10 IAI_2 Error Detection, Timing Diagram

As shown in Fig. OPT. 11 and 12, G1WPAI is enabled whenever the Permit Automatic Interrupt flip-flop (F1WPMT) is reset or whenever a request for access is present from the API module. The output of G1WPAI is applied through a 1 microsecond filter to filter out changes in the request for access signal from the API module as each waiting interrupt is recognized for service. This filter requires that all waiting interrupts be serviced before a "1" output will go to "0". The "1" output from the filter along with API and STALL ENBL switches in the ENBL position and the ON switch not pressed to initialize will enable G0WPMP. If the output of G0WPMP remains a "0" for the set time duration (between 25 and 250 milliseconds) of the delay circuit an output of "0" for approximately 20 microseconds is provided. This "0" output in conjunction with G1MRTM sets F1WDSC. G1MRTM is used to reset F1WDSC when executing the SPB located in location 24_8.

Setting F1WDSC causes a "1" to be stored in bit 18 of location 1 when the SPB in location 24_8 is executed. In addition, F1WDSC enables G0WDSC when interrupts are not inhibited by IAI_2 and $\overline{G1WENA}$. IAI_2 and G1WENA errors are checked in the circuitry previously described.

Enabling G0WDSC causes F1WDO1 to be set. Setting F1WDO1 causes the contents of I to be stored in location 21_8 and transfers program control to location 24_8 as previously described.

ALTERNATE SOURCE TIMER

The optional (4DP4800AS06) alternate source timer permits a more stable 60 Hz or 50 Hz signal (normally from the power company) to be used for elapsed time counting than the computer power source (e.g., an MG set, static inverter, motor/motor generator, engine/motor/generator). Logic level timing pulses are generated from this source and applied to the Line Frequency Timer for application to the API module for elapsed time counting and/or the clock source of optional pulse source initiator.

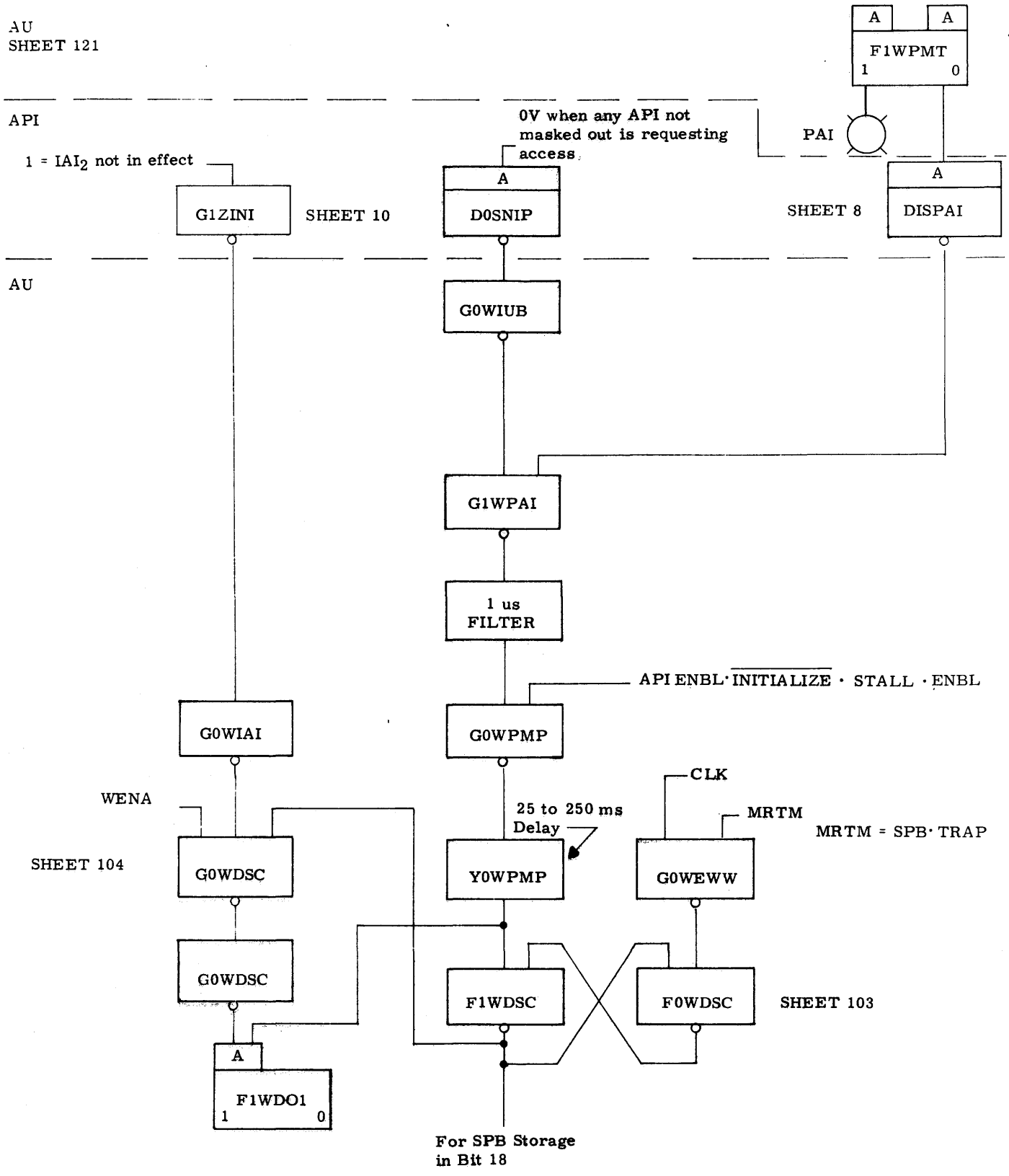


Fig. OPT. 11 Permit F/F Reset, Error Detection Block Diagram

Logic for the alternate source timer is shown on sheet 144 of the arithmetic unit logic. The more stable 115VAC 60 Hz or 50 Hz signal is connected to TB1 pins 2 and 3. The computer power is connected to TB1 pins 4 and 5 from the AC Distribution Panel. Whenever computer power is on, relay K1 is energized applying the stable source to the diode rectifier circuit and to transformer T1. Whenever the stable power source is present, relays K2 and K3 are held energized. Relay K2, when energized, applies the stable source pulse from T1 to the line frequency timer shown on sheet 145 of the arithmetic unit logic. This stable power source would then be used for lapsed time counting.

If, however, this stable power source should fail, relays K3 and K2 would de-energize. In the de-energized state, relay K2 applies the less stable 60 Hz or 50 Hz central processor power from the AC Distribution Panel and transformer T2 to the line frequencer timer shown on sheet 145. This less stable power source would then be used for lapsed time counting.

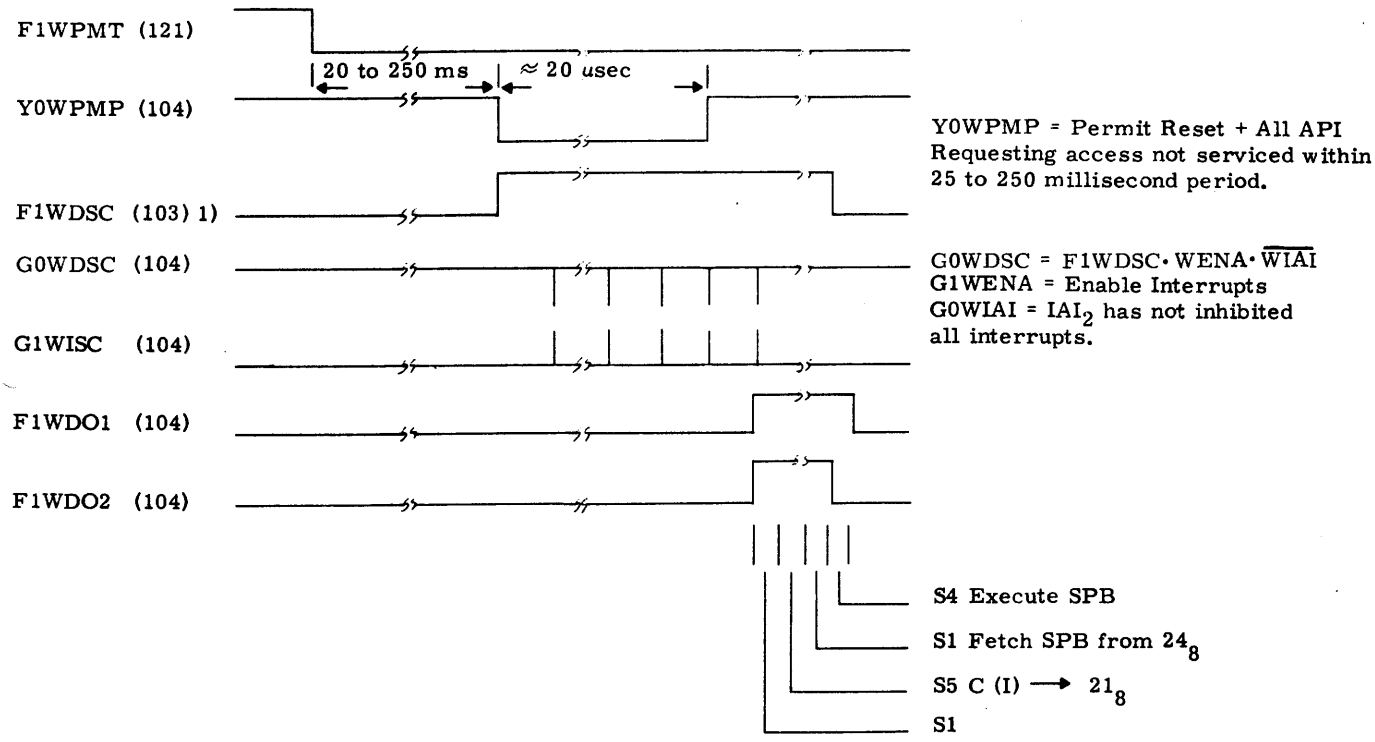


Fig. OPT. 12 Permit F/F Reset, Error Detection Timing

SYSTEMS ALARMS

The systems alarms option (4DP4800AS07) provides relay contact closure outputs to indicate on operator consoles, or other external equipment, various alarm conditions that occur within the computer system. The contact closure outputs are shown on sheet 143 of the arithmetic unit logic. The outputs are "form C" isolated contacts with a rating of 2 amps at 30 VDC or 1 amp at 100 VAC. The following alarm outputs are provided:

CORE PARITY - This contact closure output indicates when a parity error was detected while reading data from core memory. When a parity error is detected, relay K1 on the KALA1 board is energized. In addition to energizing relay K1, a parity error lights the CORE PRTY and ALARM CLEAR indicators on the Programming and Maintenance Console. Once a parity error has been detected, the alarm condition may be cleared by executing a JNP command, pressing the ALARM CLEAR switch, or pressing the ON-INIT switch.

CORE TEMP - This contact closure output indicates when the temperature within the core stack(s) is outside the operating limits of 45°C and 65°C. When the temperature is outside these limits, relay K2 on the KALA1 board is energized. Also, when the temperature is outside these limits, the CORE TEMP and ALARM CLEAR indicators on the Programming and Maintenance Console will be lighted. If a parity error occurs while an out of temperature condition exists, memory operation will be halted.

CSU OVERHEAT - This contact closure indicates when the temperature of the logic power supply, memory power supply, or auxillary memory regulator is approaching its upper limit or when the airflow from any of the blowers in the CSU has declined to a dangerous level. When any of these conditions are detected, relay K3 on the KALA1 board will be energized and the CAB TEMP and ALARM CLEAR indicators on the Programming and Maintenance Console will be lighted. These indications serve only as a warning, if the temperature continues to rise to the trip level, another set of thermostats will shut DC power down causing a BLM alarm.

The CABINET OVERHEAT (K9) relay is also controlled by K3 and temperature sensors from other cabinets may be connected to this alarm circuit.

STALL - This alarm provides an indication of a malfunction in the system program or hardware that causes a hang-up or stall condition in program sequencing. A detailed description of the stall alarm is contained in the DESC and GEN II - SSA portions of this document. When a stall condition is detected, relay K4 is de-energized and the STALL ALARM and ALARM CLEAR indicators on the Programming and Maintenance Console are lighted. The alarm condition may be cleared by the Initialize or STALL ENBL switches.

CONSOLE ENABLED - When the key switch on the Programming and Maintenance console is in the enabled position, relay K5 or the KALA1 board is de-energized indicating that the running program may be inadvertently manipulated from the console. When the console key switch is in the disabled position, relay K5 is energized indicating that the program cannot be disturbed from the console.

PROGRAMMABLE ALARM - This contact closure alarm output is controlled by the SALM and RALM commands and may be used to alarm any condition that can be sensed by the program. Executing RALM or pressing the ON/INIT switch energizes relay K6 on the KALA1 board and executing the SALM command de-energizes relay K6.

BLM - This alarm output indicates when DC power has been removed (either manually or automatically due to an out of tolerance condition). Relay K7 is energized when power is on and de-energized when power is removed or the 28V power supply has failed.

ANY ALARM - An additional contact closure output is provided to detect if any of the above alarms occur. As shown on sheet 143 of the arithmetic unit logic, when no alarms are present, relay K8 will be energized. If, however, if any of the above alarms occur, relay K8 will be de-energized.

ALTERNATE SOURCE TIMER - This contact indicates when the optional (4DP4800AS06) Alternate Source Timer located in the CSU Sequencer Unit has detected a failure in the stable 115 VAC alternate line frequency source and has switched the time source signal to the less stable system power source. When the stable alternate line frequency source is being used by the line frequency timer to provide pulses for elapsed time counting, relay K14 on the KALA1 board is energized. Should this source fail, the timer source is switched to the less stable system power source and relay K14 will be de-energized.

EXTERNAL ALARM - The customer may connect any normally closed alarm circuit to relay K10. The alarm condition would then de-energize relay K10 which would de-energize relay K8 providing the Hardware Alarm indication.

REMOTE INITIALIZE - This alarm relay is not used.

SPARE - As shown on sheet 143 of the arithmetic unit logic, two relays and contact closure outputs are provided. These are not normally used but are available for customer implementation.

ADJUSTABLE PULSE GENERATOR

This option provides logic level square wave pulses for application to external process control stations through the optional Pulse Source Initiator or as a high resolution clock API input for time counting.

The output frequency is controlled by a highly stable crystal controlled oscillator. Any one of the following output frequencies may be selected by a jumper pin on the SPGA5 board in slot B15FK (logic sheet 147); 200, 360, 500, 720, 1000, 1200, 2000, or an AUX (externally

wired for 900) pulses per second. The positive going and negative going pulse widths are of equal duration. The pulses may be started (SAPG - 25000502), and stopped (RAPG - 25000500), under program control or applied continuously by inserting a jumper pin on the SPGA5 board in slot B16FK in the I position.

The jumper pin is inserted in the I position when the Adjustable Pulse Generator is used with the Pulse Source Initiator (Process Digital Controller option) and the jumper pin is inserted in the S position when the Adjustable Pulse Generator is used as a high resolution clock.

A timing diagram that illustrates the operation of the adjustable pulse generator is contained on sheet 6.9 of the AU logic.

Fig. OPT. 13 illustrates the Adjustable Pulse Generator logic in simplified form. Control of the Adjustable Pulse Generator originates from a stable 5.760 MHz crystal controlled oscillator that free runs. The 5.760 MHz output is applied through three flip-flops (F1FOS1, 2, 3) which each act as a frequency divider providing a 720KHz output signal. Unless inhibited, this signal is applied as a clock to a counter providing the output pulses.

The 720 KHz clock pulses are inhibited in G0FC$\overline{\text{LK}}$ if the RAPG command has been executed and the jumper pin on the SPGA5 board (slot B16FK) is in the S position. Executing the RAPG command will set F1FOFF inhibiting the clock. Executing the SAPG command, pressing the INIT switch, or placing the jumper in the I position will reset F1FOFF and enable the clock pulses to the counter.

The counter consists of F1FB00 through F1FB16, F1FOVF, and F1FFX1. The counter is preset according to the frequency of the desired output as selected by the jumper pins on the SPGA5 in slot B16FK. The counter stages preset for each frequency selected are shown below:

To illustrate the operation of the counter, consider that it is initially cleared. It may have been cleared because of the execution of the RAPG command or due to incrementation. If the counter is cleared due to the execution of RAPG (i.e. F1FOFF is set), SAPG must be executed, INITialize must be pressed, or the jumper on the SPGA5 board placed in the I position for counter operation to begin (reset F1FOFF).

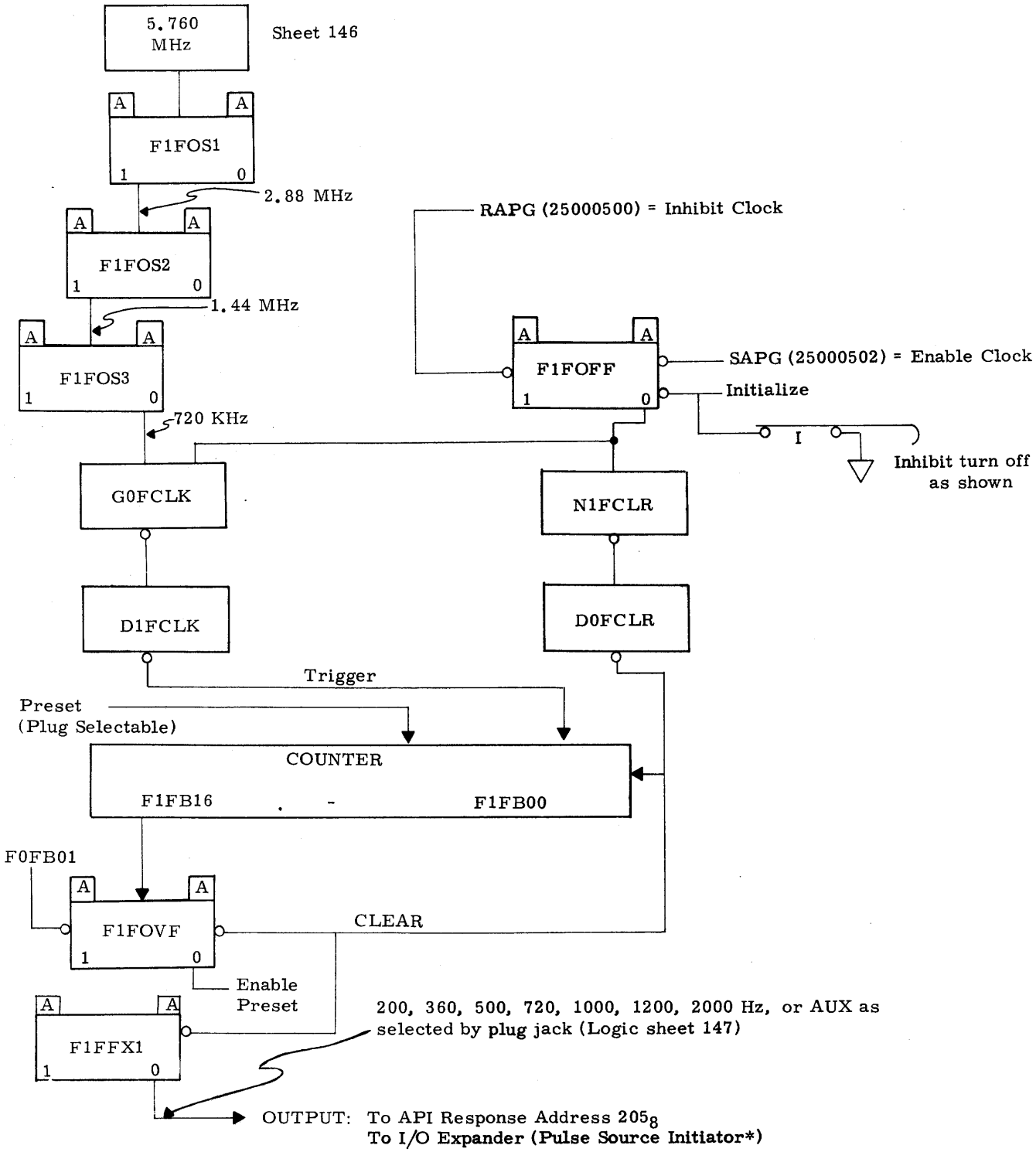
With F1FOFF reset and F1FOVF reset, the counter stages will be preset (D0FSET) according to the frequency selected by the jumper pins as described above.

The 720KHz clock signal increments the counter. The second clock pulse will set F1FB01 which will set F1FOVF and remove the preset signal leaving the counter stages in the preset state. The clock will continue to increment the counter until F1FB16 changes state. Since F1FB16 is preset for all jumper conditions, F1FB16 will reset causing F1FOVF to reset. Resetting F1FOVF will cause F1FFX1 to set changing the state of the output signal.

Resetting F1FOVF will again enable the counter to be preset according to the frequency selected by the jumper pins. Incrementing of the counter will continue setting F1FB01 which will set F1FOVF and remove the preset signal. The clock will continue incrementing the counter until F1FB16 is again cleared. Clearing F1FB16 will reset F1FOVF. Resetting F1FOVF will cause F1FFX1 to reset changing the output signal.

The above cycle continues to occur applying output pulses until RAPG is executed.

Frequency Hz	Jumper Pin	Counter Stages Preset
200	J1	FB16, 15, 14, 13, 12, 11, 07, 06, 05, 04, 03
360	J2	FB16, 15, 14, 13, 12, 11, 10, 04, 03
500	J3	FB16, 15, 14, 13, 12, 11, 10, 08, 05, 04
720	J4	FB16, 15, 14, 13, 12, 11, 10, 09, 03, 02
1000	J5	FB16, 15, 14, 13, 12, 11, 10, 09, 07, 04, 03
1200	J6	FB16, 15, 14, 13, 12, 11, 10, 09, 07, 06, 04, 02
2000	J7	FB16, 15, 14, 13, 12, 11, 10, 09, 08, 06, 03, 02
AUX (Prewired 900Hz)	J8	FB16, 15, 14, 13, 12, 11, 10, 09, 06, 05, 04



*A jumper pin on the ILDC5 board in slot B129 F-K (logic sheet 96.2) will select (B) the Adjustable Pulse Generator output or (G) the line frequency signal for application to the Pulse Source Initiator.

Fig. OPT.13 Adjustable Pulse Generator

READER COMMENTS

The General Electric Company solicits your comments on publications covering Process Computer equipment. Please explain any "no" responses in the COMMENTS section. Your comments and suggestions become the property of the General Electric Company.

- Name of Manual: _____
- What is your computer application: _____

  _____
- How is this publication used:

  Familiarization ☐ Reference ☐

  Training ☐ Maintenance ☐

  Other (Explain) _____

	YES	NO
• Does this publication meet your requirements	☐	☐
• Is the material:		
1) Presented in clear text	☐	☐
2) Conveniently organized	☐	☐
3) Adequately detailed	☐	☐
4) Adequately illustrated	☐	☐
5) Presented at appropriate technical level	☐	☐

- Please provide specific text references (page number, line, etc.) with your comments.

NAME _____ DATE _____

TITLE _____

COMPANY NAME _____

AND ADDRESS _____

COMMENTS:

No postage necessary if mailed in the U. S. A.

Staple

Communications concerning Technical Publications should be directed to:

Manager, Technical Publications
Utility and Process Automation Systems Operation
2255 West Desert Cove Road
Phoenix, Arizona 85029

Fold Fold

FIRST CLASS
Permit No. 4091
Phoenix, Arizona

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

GENERAL ELECTRIC COMPANY
UTILITY and PROCESS AUTOMATION
SYSTEMS OPERATION
2255 West Desert Cove Road
Phoenix, Arizona 85029

Attention: Technical Publications

Fold Fold

Cut Along Line

Additional Comments:

4032D AUTOMATIC PROGRAM INTERRUPT MODULE
TABLE OF CONTENTS

INTRODUCTION

The GE-PAC* 4032D Automatic Program Interrupt Module is designed to operate with the Model 4022D Arithmetic Unit to provide continuous surveillance of critical inputs without consuming central processor time and, on a priority basis, to execute predetermined instructions when these inputs have interrupted the running program. These inputs can be either internal or external to the central processor. Without this automatic recognition feature, the program would have to contain numerous checking routines to determine whether or not these functions require servicing. After an automatic interrupt has been serviced, control is returned to the running program without loss of information or sequencing.

Basically, when an interrupt occurs from a specific device or module it is sensed in the API logic by a change detector. The change detector records the fact that the interrupt has occurred and the API logic determines if the interrupt can be accepted at this time. If the interrupt is accepted, it is compared against other interrupts which may have occurred at approximately the same time and the accepted interrupts will be serviced on a priority basis. The highest priority interrupt will generate an address (API response address) corresponding to the channel on which the interrupt occurred and the response address is accessed as soon as the AU informs the API that it may interrupt. The contents of the response address (API response instruction) are placed into the "I" register of the AU, from where the instruction is executed. This instruction, fetched from the API response address, will then determine the sequence of events to occur as a result of the generated interrupt.

The response instruction may be the only instruction executed as a result of the interrupt, or it may branch to a service routine containing numerous instructions. The API logic guarantees that at least one instruction will follow the execution of the response instruction, but after that another interrupt may intervene. It is evident then, that when more than two instructions are required to service an interrupt, an SPB instruction (Save Place and Branch - saves the address at time of interrupt) should be contained in the API response address. This will provide a record of the address at the time of interrupt.

Each interrupt is associated with a fixed core memory address which is referred to as the "interrupt response address". Core Memory locations 200_8 through 377_8 are API response addresses. Each interrupt is assigned a priority level and the relative priority level of any interrupt is indicated by its API response address. The highest priority level is assigned to core location 200_8, and priority levels are assigned in descending priority as the memory location assignments increase to 377_8. The fact that an interrupt has occurred is recorded in the API logic. When no interrupt of a higher priority is awaiting service, or is being serviced, and if the interrupt is not inhibited, the recorded interrupt is serviced.

Interrupts may be prevented from being acknowledged under program control. Interrupts are divided into two basic categories, inhibitable and noninhibitable. according to the API response address. Those API's with response addresses 200 through 277 are noninhibited and those with response address 300 through 377 are inhibited.

Inhibitable interrupts may be inhibited by executing the IAI (25030000) command. All interrupts, both inhibitable and noninhibitable, may be inhibited by executing the IAI_2 (25000304) command, and inhibited in groups of 4 by enabling a corresponding mask register bit, which is controlled by the LMR and LMR_2 commands.

Any interrupt must await the completion of the instruction currently being executed by the AU, and certain instructions, because of operations which must follow them, inhibit the servicing of interrupts until the next instruction is executed. Among the non-interruptible instructions are: BTS, BTR, LDP, LPR, LDX, SPB, TXH, XEC, CLO, CLZ, CMO, CMZ, Quasi instructions, and instructions which control the API subsystem.

OPTIONS

The basic 4032D API module is part of the 4DP4800BS011 Basic Central Systems Unit. In addition, the following options are associated with the API module:

4DP4800AS101 - External API Buffer

4DP4800AS121 - 64 to 128 API and 24 to 48 TIM/TOM Adder

4DP4802AS04 - ASU API Neckdown

The API assignments associated with a system are also optional. These assignments are selected using jumper pins and/or wiring. The Release and Option Specification (4DPRO1000B) defines the jumper pin options and wire list connections for all API assignment options. The API assignments for a particular system are listed in the Hardware Address Summary drawing provided for each system.

This publication describes all of these options, as well as the basic API module.

REFERENCE DOCUMENTS

The following documents will be of assistance when reading the theory of operation.

Logic:

70C180954 - Basic API module including the the 4DP4800AS101 External API Buffer and the 4DP4800AS121 64 to 128 and 24 to 48 TIM/TOM Adder

70C181085 - 4DP4800AS04 ASU API Neckdown

*Registered Trademark of General Electric Company

Wiring:

70A120518 - 4032D Wire List

70A120519 - 4032D Pin Connection List

70A120520 - 4032D Signal List

Hardware Address Summary:

Unique for each system - provides API address assignments

Interface Module Logic:

70C180955 - Arithmetic Unit

70C180899 - 4078B I/O Expander

70C180908 - Bulk Memory Controller

70C180914 - 4780E Power Components

70C180909 - Basic I/O Buffer

Maintenance:

API Section of Computer Maintenance Manual

TERMINOLOGY

Throughout the following theory of operation, signal names containing an asterisk (*) or symbol (#) are used. The asterisk denotes any letter between B and R and refers to areas in the 4032D API Module. The symbol (#) can mean any number between 0 and 7. These devices allow the discussion to be general in nature and imply that signals in one portion of the overall circuit are treated identically to corresponding signals in another part.

Some signal names end with the letter L, meaning lower. If the corresponding signal on the upper half of a card is acted upon in the same manner or at the same time, the signal appears ending with L(U). This means either L or U.

Signal names with only six characters originate in the API module. With the exception of well known signals like SCO1 and SCA, all signals originating in the Arithmetic Unit are listed here preceded by the letters AU.

OPERATION

The various inhibits to the incoming interrupt, priority establishment and address generation, and the control sequencing of the API logic are described in detail in the following text. In general, the description of an Automatic Program Interrupt can be organized under the major classifications listed below:

- API Interface
- Interrupt Generation
- Interrupt Acceptance
- Priority Establishment and Address Generation
- Sequence Control

These classifications will be discussed individually in the order listed above. This sequence is depicted in the flow charts, Figs. OPR. 3 and OPR. 4. Reference should also be made to the logic schematic for the 4032D GE Drawing No. 70C180954, the block diagram, Fig. OPR. 2, and the timing diagrams, Figs. OPR. 10 and OPR. 11 during the following discussion.

API Interface

Fig. OPR. 1 illustrates the modules, logic drawings and connections of the modules which interface with the API module. These interface signals are briefly described in the following paragraphs.

- Arithmetic Unit to API

  1. API Lockout Control - This switch, located on the programming and maintenance console, inhibits all interrupts when placed in the lockout position.

  2. Initialize - This ensures the correct initial operating state of the API logic when computer is initialized by pressing the ON/INIT switch. The logic initialized is the sequence control flip-flops and the level change detector flip-flops.

  3. Timing - The basic 10 MHz clock signal and the sequential timing generated within the AU provide the timing in the sequencing control circuitry of the API module.

  4. Permit Interrupt Request - This signal allows all interrupts when the Permit Automatic Interrupt flip-flop in the AU is set by the PAI instruction.

  5. Interrupt Request Enable - This signal controls the timing for servicing requesting interrupts by the AU.

  6. Echo Interrupt Enable - This signal from the AU enables the generation of an echo interrupt when the current API response instruction is a DMT and the count passes from zero to a minus one, or when the API response location contains a TIM/TOM control word and the word count reaches a table full or empty condition.

  7. A-Register - Least significant 16 bits of the A-Register are transferred to load the API Mask Register by the LMR command.

  8. K_0, K_1, K_2 Decode - These signals are used for API Mask Register control (LMR and test).

- API Module to Arithmetic Unit

  1. API Memory Request - This signal allows addressing memory from the API module to obtain the response instruction.

  2. Decode for TIM/TOM Operation - Each unique interrupt, activated by a device using the TIM or TOM function, is wired to a matrix board located in the AU, which determines the device code (K bits) and the operation (S bits), TIM or TOM, for that device.

  3. Memory Addressing - The eight bits necessary to address memory locations 200_8 to 377_8.

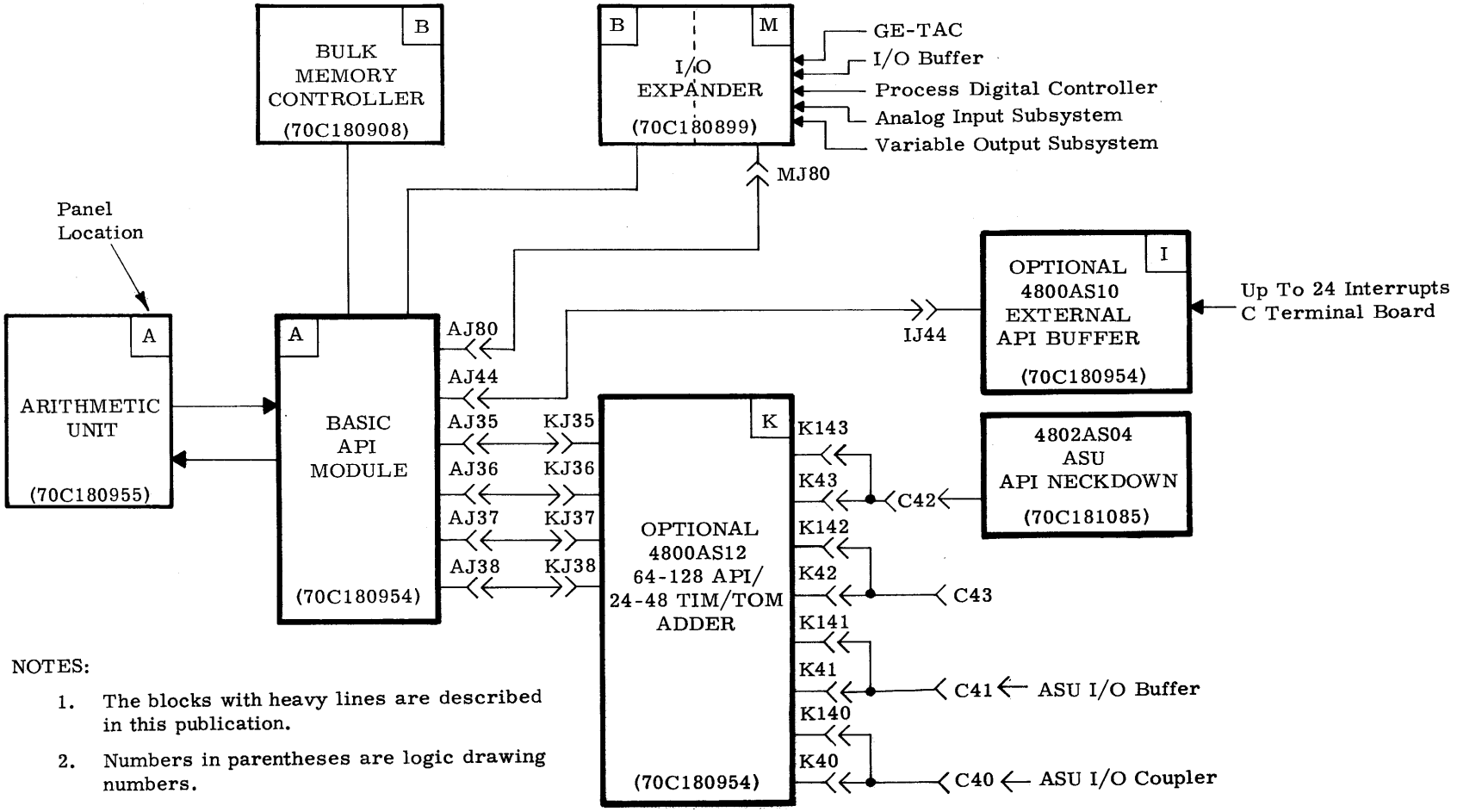
  4. Monitor to Watchdog - The API Watchdog timer, an option of the 4022D AU, provides hardware protection against the possibility of programming errors which inhibit the servicing of interrupts.

- Interrupt Inputs

  These are the signals that initiate the API operation.

INTERRUPT GENERATION

Within this description, the generation of an interrupt is considered as the interval from the time the interrupt is generated within a functional module or device, until the API level change detector flip-flop in the API logic is set. The setting of the level change detector flip-flop retains the fact that an interrupt has occurred on that channel and will retain this information until the request is acknowledged.



NOTES:

1. The blocks with heavy lines are described in this publication.
2. Numbers in parentheses are logic drawing numbers.

Fig. OPR. 1 API Interface

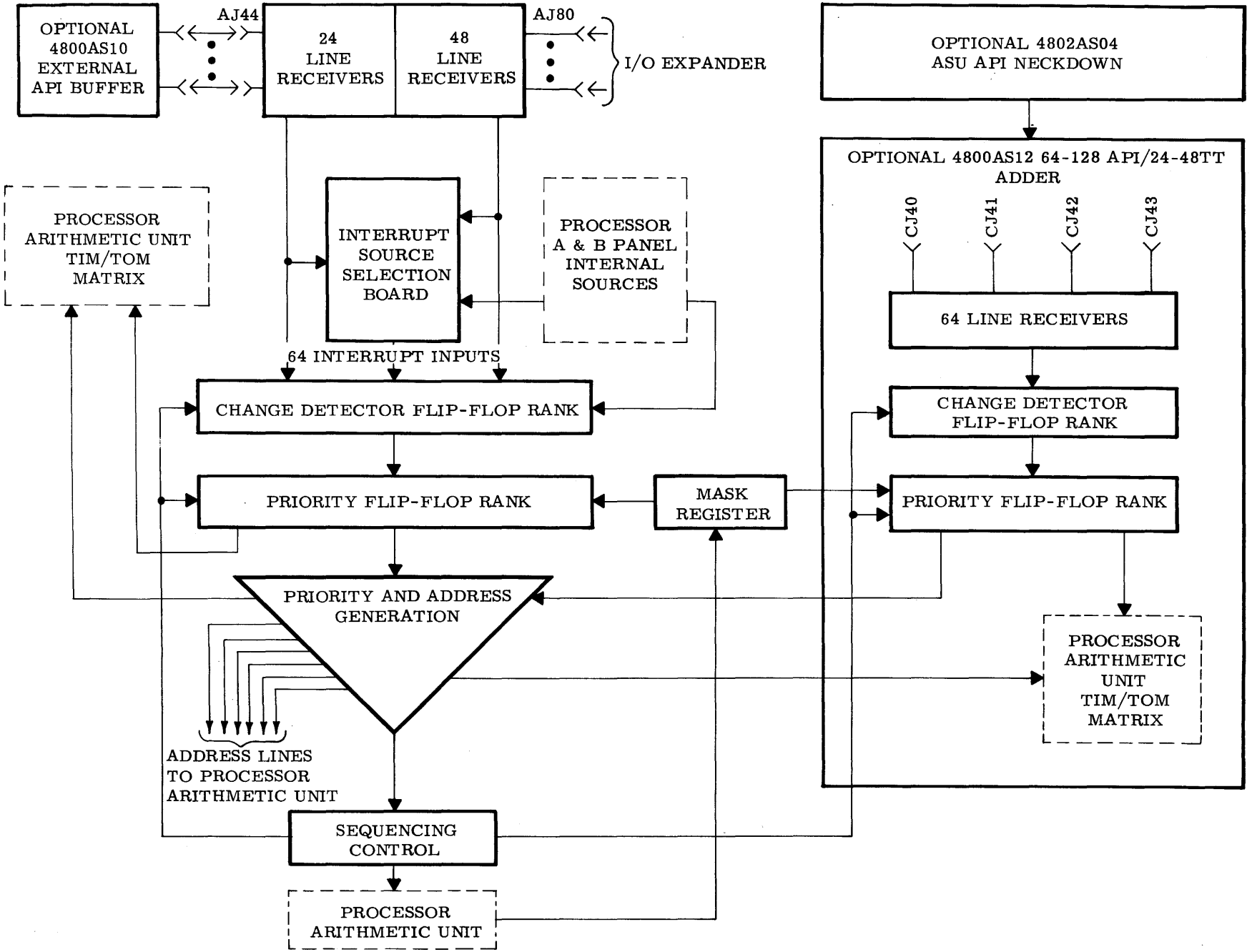


Fig. OPR. 2 API Block Diagram

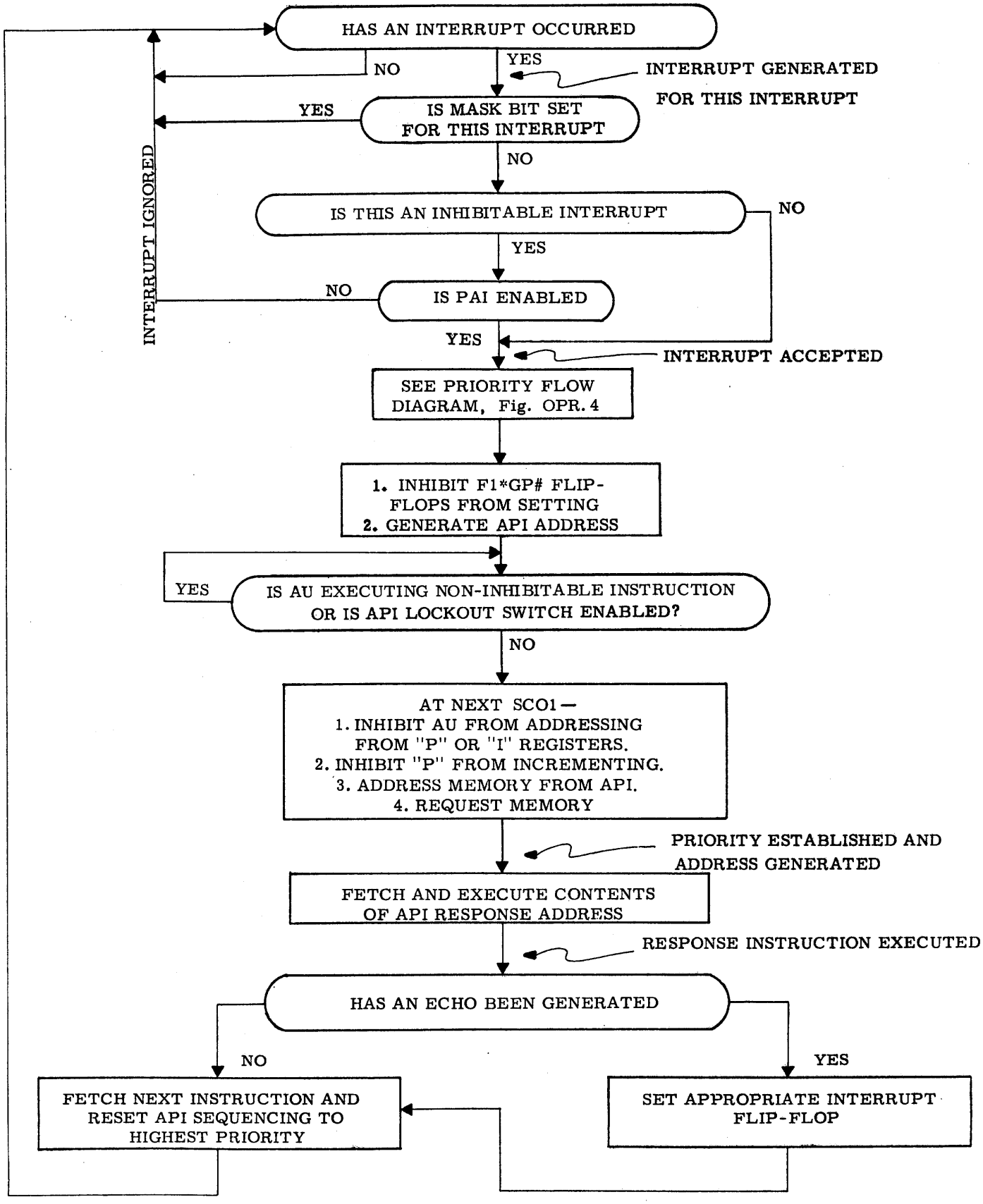


Fig. OPR. 3 API Flow Diagram

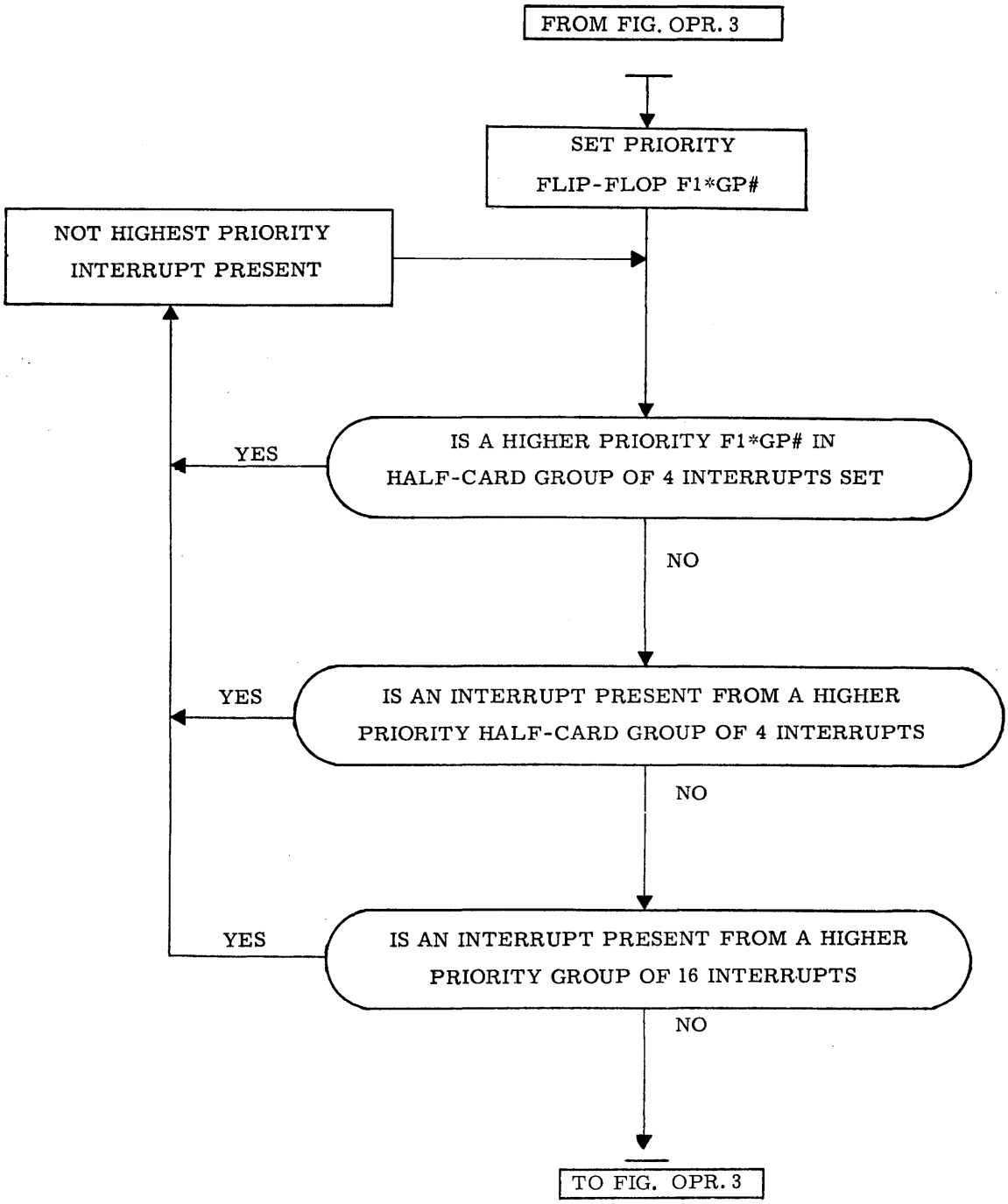


Fig. OPR. 4 Priority Establishment

Interrupt Sources

Interrupts may originate from a variety of sources, some of which may have switching levels not compatible to the API input level change detectors. Those which are incompatible require modification through termination and buffering logic. This termination and buffering is provided by the 4DP4800AS10 External API Buffer Adder and the 4DP4802AS04 ASU API Neckdown option. These options are described later in this section.

Examples of interrupt sources and the general functions of the instruction executed as they are serviced are:

Timekeeping (clock) interrupts are generated by the Line Frequency Timer, the Alternate Source Timer, or the Adjustable Pulse Generator. These periodic interrupts cause the execution of Full Operand instruction, DMT. This instruction acts as a one instruction subroutine which decrements a count preset by the program in the API response address. When

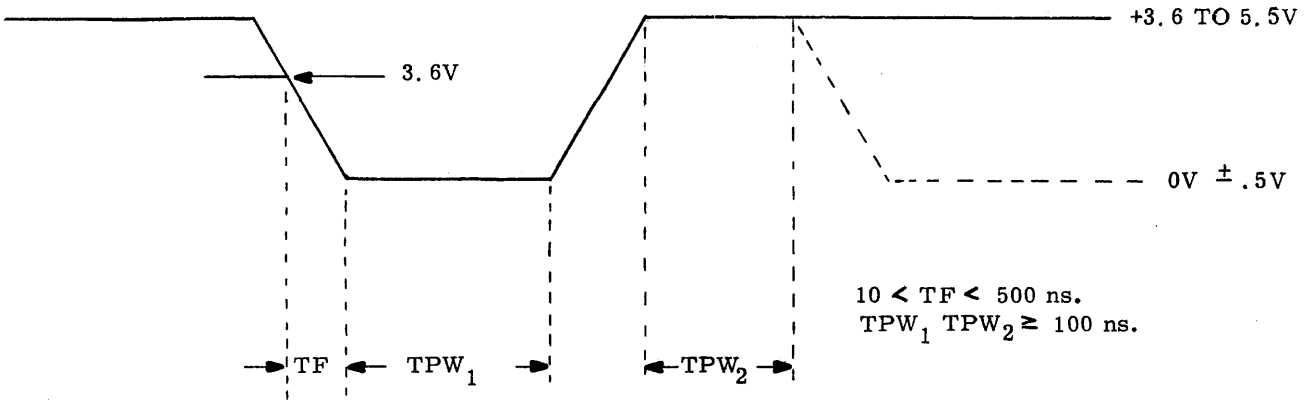


Fig. OPR. 5 Change Detector Input Signal Requirements

the count passes from zero to minus one, the DMT instruction causes an "echo" API to be generated, which informs the timekeeping program that the time period has elapsed.

Data Exchange API's are generated by input/output devices or communication channels to indicate that data is ready for transfer from the device or channel to the AU and core memory, or that the device or channel is ready to accept data. Such interrupts typically cause a TIM/TOM operation.

End-of-Record and Transfer Complete interrupts are generated by input/output devices to indicate that the transfer of a complete block of data has been accomplished. Such interrupts typically cause the Full Operand instruction, SPB, to be executed, thereby retaining data pertinent to the interrupted program, while a routine is executed which prepares for transfer of the next block of data, by storing new instructions or TIM/TOM control words in appropriate API response addresses, preparing new data tables, etc. Upon completion of the routine, return to the program is made through the use of the LDP or LPR instruction.

Echo interrupts are generated in the Central Processor when a DMT count has been fully decremented, or when a TIM table has become full, or a TOM table has become empty. In some cases, TIM/TOM generated Echo interrupts are connected to the same API module input point as the corresponding End-of-Record interrupt. Such interrupts typically cause execution of the SPB instruction as described under "End-of-Record and Transfer Complete".

Interrupt Detection

The interrupts, with the exception of the ones generating response address 200 & 203, are wired into the level change detectors as determined by the individual system requirements. Interrupt channel 203_8 does not have a direct input but comes from a signal in the AU labeled AUG1LAP3. This is necessary for the program load option which is discussed later in this text. The level change detector to which the interrupt is wired will determine the priority of the interrupt. The highest priority is accorded the interrupt connected to the level change detector which generates address 200_8 as the response address. The priority decreases in order as the response addresses increase. With the maximum API configuration, 128 points, the lowest priority will be address 377_8.

The level change detector flip-flops are enabled by a negative going, differentiated spike from the level change detector (d/dt). The negative going spike is generated whenever the input to the level change detector changes from a positive to a negative level. The parameters of the input signal to the level change detector must be within those illustrated by Fig. OPR. 5. The fall time (T_F) of the signal applied to the level change detector must be greater than 10 ns. and less than 500 ns. The time that the signal is at a zero volt level (TPW1) and the time that the signal is again positive (TPW2), following negative transition, must be equal to or greater than 100 ns.

INTERRUPT ACCEPTANCE

Interrupt acceptance, throughout this discussion, is considered as the sequence of events from the time the level change detector flip-flop is set until the acceptance of the interrupt. The acceptance of the interrupt is governed by the items discussed in the following paragraph.

Inhibit Automatic Interrupt Control (IAI)

All inhibitable interrupts, 300-377, are inhibited using the IAI (25030000) command. Priority Enable gate G0*PEL(U) is wired to the output of driver DISPAI. The IAI command inhibits DISPAI which holds all Priority FFs 300-377 reset. These Priority FFs are again permitted after the execution of a PAI (Permit Automatic Interrupt) instruction. The IAI instruction is normally used to prevent critical program routines from interruption by low priority interrupts.

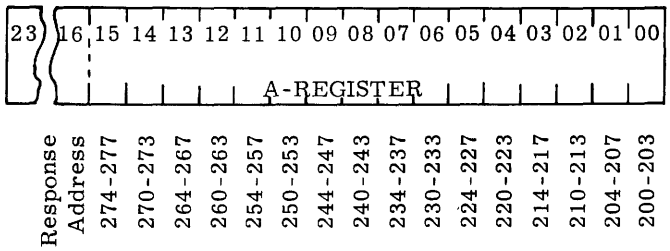
$\overline{G1*PEL(U)} = \overline{D1SPAI} + \overline{D1SPA2}$
$\overline{D1SPAI(2)} = \overline{AUF1WPMT}$
$\overline{AUF1WPMT}$ = Execution of IAI instruction.

Mask Register

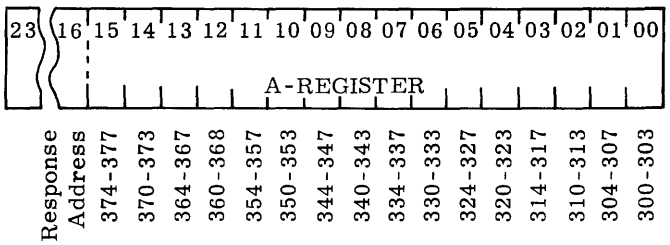
Half-card groups of four interrupts, whether wired as inhibitable or not, can be inhibited during the course of the running program under the control of the API Mask Register. One register bit or flip-flop is used for each group of four interrupts. There are two mask register flip-flops located on each change detector card. One flip-flop controls the lower half of the card via G0*PEL and the other controls the upper half of the card via G0*PEU.

Each mask register flip-flop is controlled by one of the least significant 16 bits of the "A" register. The mask register is loaded by Load Mask Register instructions LMR (25000302_8) and LMR_2 (25000300_8). LMR control API's 200-277, LMR_2 controls the inhibited interrupts, 300-377.

LMR_1



LMR_2



A "one" in the appropriate bit position of the mask register will inhibit the particular group of four interrupts, regardless of its inhibitable/non-inhibitable status. To permit interrupt requests from a given group of four interrupt points, the program must load a zero into the appropriate bit position of the mask register. For normally inhibitable interrupts, the Permit flip-flop must also be set.

The following logic description of the loading and controlling of the mask register by the LMR command is referenced to the timing diagram of Fig. OPR. 6. Operation of loading and controlling the upper 16 bits of the mask register by the LMR2 command is identical with the exceptions that 25000300_8 is decoded and corresponding signal names end with the numeral two (2).

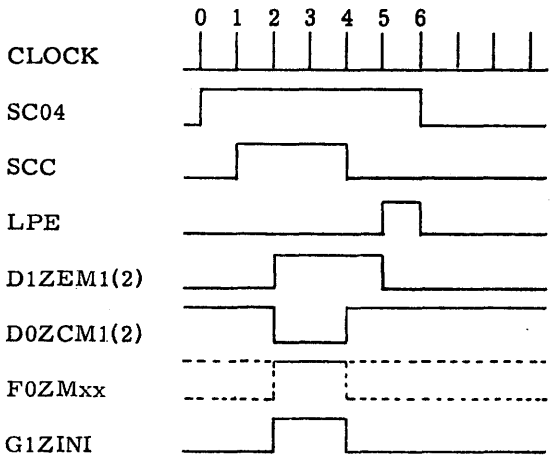
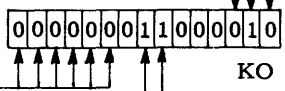


Fig. OPR. 6 Mask Register Timing Diagram

The Enable Mask gate G0ZEM1, on the mask register control card is enabled during sequence state 4 of the LMR command. The enabling of G0ZEM1 depends upon the following signals being true:

- $\overline{G0ZK01}$
  $\overline{G0ZK01} = \overline{AUG1NK01}$
- G1ZK02
  G1ZK02 = AUN0NK02
- $\overline{G0ZK04}$
  $\overline{G0ZK04} = \overline{G1ZK04}$
  $\overline{G1ZK04} = \overline{AUN0NK04}$
- AUG1DKS0
- AUD1NK22
- AUD1NK21
- G1ZBC1
  G1ZBC1 = AUD0SC04
  D0SC04 = $\overline{SC01} \cdot \overline{SC02}$
- AUD1TSCB



Enable Mask driver signal D1ZEM1 becomes true:

D1ZEM1 = G0ZEM1

During SCB · SCC time of sequence state 4 the Clear Mask driver D0ZCM1 goes true and clears all mask register flip-flops for interrupts 200-277 and temporarily prevents the accepting of interrupts.

D0ZCM1 = G1ZCLM·D1ZEM1

D1ZEM1 = G0CEM1

G1ZCLM = G0ZCLM

G0ZCLM = (D1ZEM1· AUD1TSCC) + (D1ZEM2· AUD1TSCC)

A given mask register flip-flop will remain cleared if the corresponding bit of the previously loaded A-register is a zero. However, if a particular group of four interrupts is to be inhibited, a "one" will have been loaded into the corresponding bit position of the A-register. The A-register bit is ANDed with the Enable Mask signal, D1ZEM1. With both signals true, the mask register flip-flop sets.

F1ZMxx = G0ZMxx

G0ZMxx = D1ZEM1· AUF1ARxx· $\overline{\text{D0ZCM1(2)}}$

The clear side output of each mask register flip-flop is wired to an input of its associated Priority Enable gate G0*PEL or G0*PEU. If a mask flip-flop is set, the zero signal will disable G0*PEL(U) and G1*PEL(U), preventing the setting of Priority flip-flops in the group of four interrupts:

$\overline{\text{G0*PEL(U)}}$ = F0ZMxx

$\overline{\text{G1*PEL(U)}}$ = $\overline{\text{G0*PEL(U)}}$

API acceptance is halted during the setting of the mask register to allow the priority and addressing logic to adjust to the highest priority of the interrupts permitted by the mask register. This is accomplished by "DC clearing" the Time Counter flip-flops, Priority Interrupt flip-flops and the Generate Echo flip-flop with signal D0SINT.

D0SINT = G1SINI

G1SINI = G0SPMT

G0SPMT = G1ZINI

G1ZINI = D0ZCM1 + D0ZCM2

Additional Inhibits

Acceptance of an interrupt must await the completion of any current interrupt being serviced. After an interrupt is accepted by the Priority FFs and during the time that an API is being serviced, D0SNIP goes false. This causes gates G1*NIL(U) to go false preventing the Change Detector FFs from being gated to the Priority FFs.

Acceptance of interrupt inhibited by $\overline{\text{G1*NIL(U)}}$

$\overline{\text{G1LNIL(U)}}$ = $\overline{\text{D0SNIP(1)}}$

$\overline{\text{D0SNIP}}$ = Priority established on an accepted interrupt.

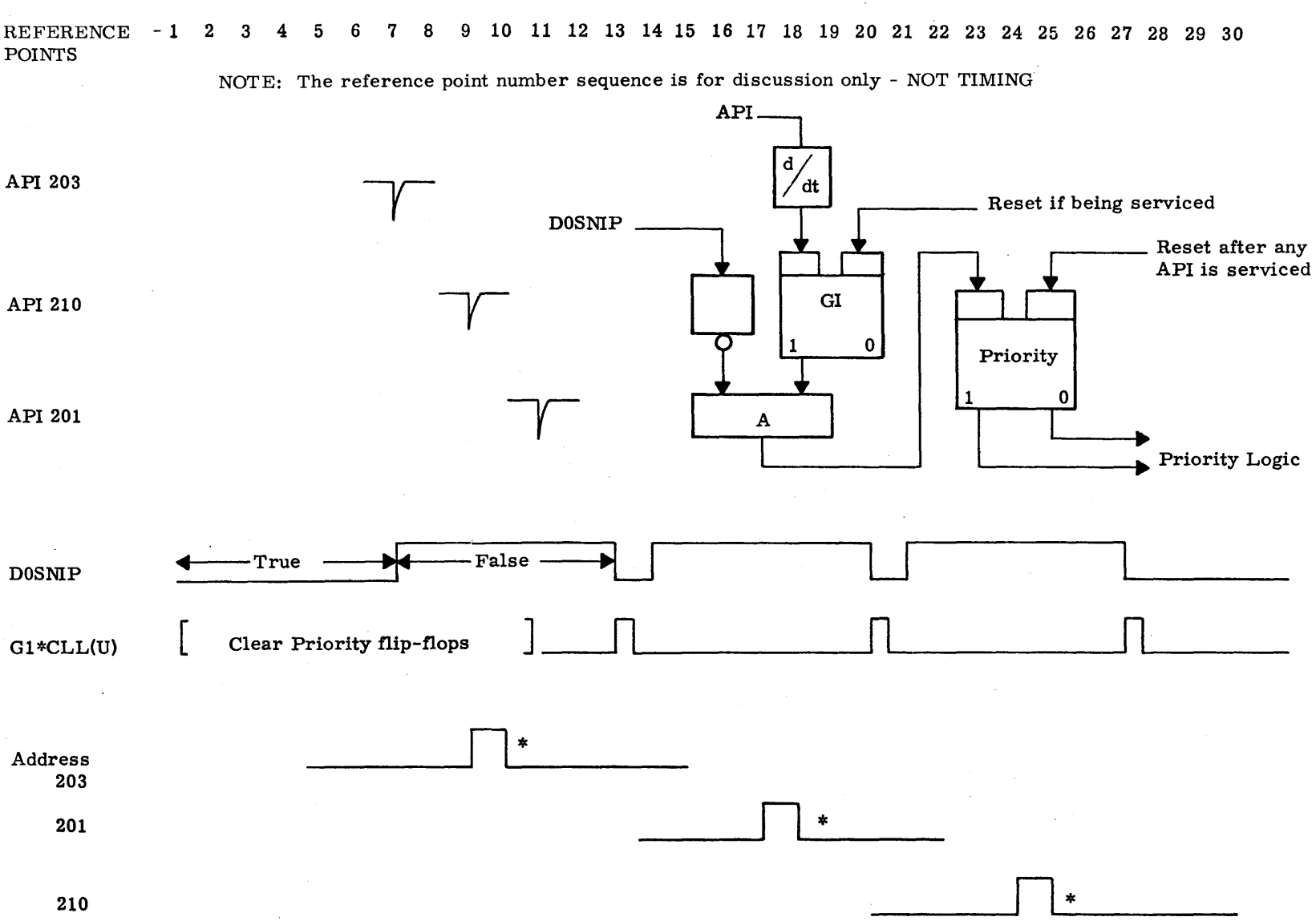
Interrupt acceptance may also be temporarily halted during a program load sequence described under the heading "Program Load" within this publication.

Although an interrupt is considered accepted at this point, if none of the previously mentioned inhibits is present, the interrupt response may still be halted temporarily or indefinitely after the priority is established for the interrupt. This is described under the heading "Sequence Control", following the description of Priority Establishment and Address Generation.

PRIORITY ESTABLISHMENT AND ADDRESS GENERATION

If an interrupt is not inhibited by the previously mentioned conditions, the respective Priority flip-flop (located on the level change detector card) for that interrupt request is set. Since a number of interrupts may have been retained by the level change detector flip-flops, awaiting an inhibiting function to be terminated, a number of interrupts may be accepted by the Priority flip-flops at the same time. The priority circuitry determines which of the existing interrupts is of the greatest importance (highest priority) and enables that interrupt to establish its correlating response address and request an interrupt of the running program. Refer to Fig. OPR. 7 for typical timing of address generation relative to interrupt initiation during the following discussion.

Notice that D0SNIP (No Interrupt Present) is true (0 volt level) until the first API (203) occurs. D0SNIP true, continuously gates the Change Detector FFs into the Priority FFs. Until an API occurs, (in this case API 203) D0SNIP stays true because there are no Priority FFs set. When Change Detector FF 203 is set and gated to Priority FF 203, D0SNIP goes false, starting the API cycle, and in going false prevents any other Priority FF from becoming set. Change Detector 203 will be reset after Priority FF 203 is set. While 203 is being serviced API Change Detector FFs 210 and then 201 are set. Near the end of 203 execution (reference point 13) the Priority FFs are cleared and D0SNIP goes true which gates the Change Detector FFs to the Priority FFs. Because Priority FF 201 and 210 will set, D0SNIP will at once go false (reference point 14-20). During this time the Priority Logic will determine which API to service and reset (in this case 201). At the conclusion of the execution of API 201, the Priority FFs are again reset and D0SNIP goes false (reference point 21). At this time (reference point 21-27), Change Detector FF 210 is transferred to the Priority FF 210. Being the highest (only) priority, 210 is serviced, Change Detector FF 210 is reset and Priority FF 210 is reset at the end of the API cycle (which is during SC01 of the instruction following the API instruction in most cases). After API 210 is serviced, D0SNIP goes true and the API module awaits another API.



* The time during an API cycle, that memory is addressed is AUD1SAMW = SC01·SPI1

Fig. OPR. 7 Typical Priority Address Generation

Consider these two points:

1. After every API serviced, there is at least one instruction of a running program executed.

2. The Priority FFs are reset at the end of every API cycle in order to gate any newly set Change Detector FFs into the Priority FFs. This will reestablish the priority of the next API to be serviced.

After the Change Detector flip-flops are transferred to the Priority flip-flops, the lowest number Priority flip-flop which is set will be the one that will generate the API address (it being the highest priority) and all lower priority addresses will be inhibited from being generated. The first eight API's (Area B) will be discussed at length. Because each group of eight of the systems API's are near identical, there should be no problem correlating the discussion to all of the other areas (C-R).

Within Area B there are three major divisions:

1. Change Detector FFs F1B200-F1B207
2. Priority FFs F1BGP0-F1BGP3
3. Priority FFs F1BGP4-F1BGP7

Change Detectors:

The first stage after the input buffering is the Change Detector FFs. A negative going signal will cause these flip-flops to set. During normal operation, nothing can inhibit the input signal from setting the Change Detector FF and until the API is serviced the Change Detector FF will remain set.

After every API cycle D0SNIP goes true, gating the Change Detector flip-flops into the Priority FFs. If, or when, any Change Detector FF is set, and D0SNIP is true, another API cycle will begin. The major function of the Change Detector FFs is to remember that an API has occurred. It is reset only if it is serviced.

Priority FFs (Area B)

The Priority FFs, F1BGP0-F1BGP3 (a half card group), will be discussed in detail. The discussion will generally apply to all other half card groups, exceptions will be noted. After the Priority FFs are loaded with the Change Detector FF information, the process of priority establishment and address generation is begun. It is in fact almost instantaneous. The highest priority API which is set into the Priority FFs will inhibit all lower priority API's and the highest priority API address will be generated. Each Priority FF establishes priority (if higher) over the other Priority FFs in each half card group. If any Priority FF is set in (for example) the lower half card group of Area B (sh. 19), there will be a true output from gate D0BGB0 which will be applied to D0BGB1, the upper half card group on sh. 20. The result of this action is to inhibit any lower priority API in the lower priority group of 4 (F1BGP4-F1BGP7).

$$D0GBG0 = GP3+GP2+GP1+GP0$$

$$D0BGB1 = GP7+GP6+GP5+GP4\cdot\overline{D0BGB0}$$

This scheme carries on to the next half card group.

D0BGB0 + D0BGB1 inhibit D0CGB0 (sh. 22).

Thus far it can be seen that the higher priority inhibits the lower priority. This scheme becomes less obvious when observing D0EGB0 (sh. 28). There is no inhibit from higher priority areas to this gate. Should, for example, Priority FF 203 and Priority FF 232 be set, both will have an output from their respective D0*GB0 gates to gates in Area A (sh. 15). As shown on page 15 the D0BGB0 will exert priority over D0EGB0.

Each of the Priority FFs in the group of 4 exerts its priority over the others (except 3 or 7); e.g., on page 19, F1BGP0, when set, inhibits gates G0GBP1, G0BGP2, G0BGIL. The gates used for address generation, G0BGL0, G0BGL1, are enabled dependent on the Priority FFs which are set.

$$G0BGL0 = GP1 + GP3$$

$$G0BGL1 = GP3 + GP2\cdot\overline{GP1}\cdot\overline{GP0}$$

The group gates, D0*GB0(1), are used not only to inhibit lower priority group gates (which in effect inhibit lower priority addresses) but also are used in Area A to generate the address of the highest priority Priority FF which is set. Table OPR.1 lists all interrupts, the address generator gates which become true when the associated Priority flip-flop is the highest priority, and the address terms generated by those gates when decoded through Area A.

SEQUENCE CONTROL

With the priority established and response address generated for the recognized interrupt, an interrupt cycle is initiated. When the request generated by the recognized interrupt is permitted by the AU, the address generated by the API logic will access a core location containing the pre-determined response instruction. The response instruction will then be executed, servicing the interrupt. Following the response instruction, or routine (which may be interrupted if greater than two instructions), control will be returned to the running program.

The interrupt request is initiated by the interrupt control circuitry which forces execution of the instruction located at the interrupt response address. This is accomplished by allowing the API module to

AREA	ADDRESS BITS (AREA A OUTPUT) MX0_ 7	6	5	4	3	2	1	0	API#
B	1	0							*200
	1	0						G0BGL0	201
	1	0					G0BGL1		202
	1	0					"	"	203
	1	0				D0BGB1			204
	1	0				"		G0BGU0	205
	1	0				"	G0BGU1		206
	1	0				"	"	"	207
C	1	0			D0CGB0				210
	1	0			"			G0CGL0	211
	1	0			"		G0CGL1		212
	1	0			"		"		213
	1	0			D0CGB1	D0CGB1			214
	1	0			"	"		G0CGU0	215
	1	0			"	"	G0CGU1		216
	1	0			"	"	"	"	217
D	1	0		D0DGB0					220
	1	0		"				G0DGL0	221
	1	0		"			G0DGL1		222
	1	0		"			"	"	223
	1	0		D0DGB1		D0DGB1			224
	1	0		"		"		G0DGU0	225
	1	0		"		"	G0DGU1		226
	1	0		"		"	"	"	227
E	1	0		D0EGB0	D0EGB0				230
	1	0		"	"			G0EGL0	231
	1	0		"	"		G0EGL1		232
	1	0		"	"		"	"	233
	1	0		D0EGB1	D0EGB1	D0EGB1			234
	1	0		"	"	"		G0EGU0	235
	1	0		"	"	"	G0EGU1		236
	1	0		"	"	"	"	"	237
F	1	0	D0FGB0						240
	1	0	"					G0FGL0	241
	1	0	"				G0FGL1		242
	1	0	"				"	"	243
	1	0	D0FGB1			D0FGB1			244
	1	0	"			"		G0FGU0	245
	1	0	"			"	G0FGU1		246
	1	0	"			"	"	"	247
G	1	0	D0GGB0		D0GGB0				250
	1	0	"		"			G0GGL0	251
	1	0	"		"		G0GGL1		252
	1	0	"		"		"	"	253
	1	0	D0GGB1		D0GGB1	D0GGB1			254
	1	0	"		"	"		G0GGU0	255
	1	0	"		"	"	G0GGU1		256
	1	0	"		"	"	"	"	257
H	1	0	D0HGB0	D0HGB0					260
	1	0	"	"				G0HGL0	261
	1	0	"	"			G0HGL1		262
	1	0	"	"			"	"	263
	1	0	D0HGB1	D0HGB1		D0HGB1			264
	1	0	"	"		"		G0HGU0	265
	1	0	"	"		"	G0HGU1		266
	1	0	"	"		"	"	"	267
I	1	0	D0IGB0	D0IGB0	D0IGB0				270
	1	0	"	"	"			G0IGL0	271
	1	0	"	"	"		G0IGL1		272
	1	0	"	"	"		"	"	273
	1	0	D0IGB1	D0IGB1	D0IGB1				274
	1	0	"	"	"			G0IGU0	275
	1	0	"	"	"		G0IGU1		276
	1	0	"	"	"		"	"	277

a. Non-inhibitable API's

AREA	ADDRESS BITS (AREA A OUTPUT) MX0_ 7	6	5	4	3	2	1	0	API#
J	1	D0JGB0							300
	1	"						G0JGL0	301
	1	"					G0JGL1		302
	1	"					"	"	303
	1	D0JGB1				D0JGB1			304
	1	"				"		G0JGU0	305
	1	"				"	G0JGJ1		306
	1	"				"	"	"	307
K	1	D0KGB0			D0KGB0				310
	1	"			"			G0KGL0	311
	1	"			"		G0KGL1		312
	1	"			"		"	"	313
	1	D0KGB1			D0KGB1	D0KGB1			314
	1	"			"	"		G0KGU0	315
	1	"			"	"	G0KGJ1		316
	1	"			"	"	"	"	317
L	1	D0LGB0		D0LGB0					320
	1	"		"				G0LGL0	321
	1	"		"			G0LGL1		322
	1	"		"			"	"	323
	1	D0LGB1		D0LGB1		D0LGB1			324
	1	"		"		"		G0LGU0	325
	1	"		"		"	G0LGU1		326
	1	"		"		"	"	"	327
M	1	D0MGB0		D0MGB0	D0MGB0				330
	1	"		"	"			G0MGL0	331
	1	"		"	"		G0MGL1		332
	1	"		"	"		"	"	333
	1	D0MGB1		D0MGB1	D0MGB1	D0MGB1			334
	1	"		"	"	"		G0MGU0	335
	1	"		"	"	"	G0MGU1		336
	1	"		"	"	"	"	"	337
	1								
N	1	D0NGB0	D0NGB0						340
	1	"	"					G0NGL0	341
	1	"	"				G0NGL1		342
	1	"	"				"	"	343
	1	D0NGB1	D0NGB1			D0NGB1			344
	1	"	"			"		G0NGU0	345
	1	"	"			"	G0NGU1		346
	1	"	"			"	"	"	347
P	1	D0PGB0	D0PGB0		D0PGB0				350
	1	"	"		"			G0PGL0	351
	1	"	"		"		G0PGL1		352
	1	"	"		"		"	"	353
	1	D0PGB1	D0PGB1		D0PGB1	D0PGB1			354
	1	"	"		"	"		G0PGU0	355
	1	"	"		"	"	G0PGU1		356
	1	"	"		"	"	"	"	357
Q	1	D0QGB0	D0QGB0	D0QGB0					360
	1	"	"	"				G0QGL0	361
	1	"	"	"			G0QGB1		362
	1	"	"	"			"	"	363
	1	D0QGB1	D0QGB1	D0QGB1		D0QGB1			364
	1	"	"	"		"		G0QGU0	365
	1	"	"	"		"	G0QGU1		366
	1	"	"	"		"	"	"	367
R	1	D0RGB0	D0RGB0	D0RGB0	D0RGB0				370
	1	"	"	"	"			G0RGL0	371
	1	"	"	"	"		G0RGL1		372
	1	"	"	"	"		"	"	373
	1	D0RGB1	D0RGB1	D0RGB1	D0RGB1	D0RGB1			374
	1	"	"	"	"	"		G0RGU0	375
	1	"	"	"	"	"	G0RGU1		376
	1	"	"	"	"	"	"	"	377

b. Inhibitable API's

*API 200 thru 202 generates D0BGB0 which is used to inhibit lower priority gates but is not used directly in Area A.

Table OPR. 1 Response Address Generation

specify the address from which an instruction is to be fetched rather than by the normal Arithmetic Unit function of fetching the instruction. The "P" register incrementation is inhibited for this one instruction fetch to ensure that the program count is not lost.

The interrupt request from the address generator area causes the No Interrupt Present signals, D0SNIP and D0SNI1, to go false and enables the Time Counter Enable gate, G1STCE:

$$\left.\begin{matrix}\overline{\text{D0SNIP}} \\ \overline{\text{D0SNI1}}\end{matrix}\right\} = \text{G0ALRQ} + \text{G0AB06}$$

$$\text{G0AB06} = \text{G1AURQ} \cdot \overline{\text{G1ALRQ}}$$

$$\text{G1STCE} = (\text{G0ALRQ} + \text{G0AB06}) \cdot (\overline{\text{F1STC1}} + \overline{\text{F1STC2}})$$

D0SNIP upon going false (at the start of an API control cycle) prevents subsequent interrupts from being gated to the Priority FFs. This in turn stabilizes the priority establishment.

Time Counter Enable gate, G1STCE, arms Time Counter flip-flop, F1STC1, to set. F1STC1 sets on the next clock, arms F1STC2 to set and clears on the following clock at which time F1STC2 sets. On the third clock, F1STC1 sets again. With both F1STC1 and F1STC2 set, G1STCE goes false preventing further timer action. The delay of three clock times is necessary to ensure that the response address of the interrupt generating the sequence control cycle is firmly established in the address generator. Interrupts accepted by the Priority Granted flip-flops can continue to change the priority level and consequently the response address until the inhibiting driver D0SNIP goes false.

$$\text{F1STC1} = \text{G1STCE} \cdot \text{CLK}$$

$$\overline{\text{F1STC1}} = \text{G1STCE} \cdot \text{CLK}$$

(Status of JK flip-flops change on each clock when arming signal on both set and clear sides is the same.)

$$\text{F1STC2} = \text{G1STCE} \cdot \text{F1STC1} \cdot \text{CLK}$$

With F1STC1 and F1STC2 both true, gate G0STE3 is enabled which enables G1STE3:

$$\text{G0STE3} = \text{F1STC1} \cdot \text{F1STC2}$$

$$\text{G1STE3} = \text{G0STE3}$$

One output from G1STE3 provides the optional API watchdog timer in the AU with a monitor point. If F1STC1 and F1STC2 are not cleared within a preset time period, it indicates that a lengthy or continuous loop of non-interruptible instructions has prevented acknowledgement for too long a period.

Control Enabling Conditions

With the enabling of G1STE3, the F1SPI1 flip-flop is armed to set if AUG1WENA is also true. The interrupt response may be temporarily or indefinitely halted at this point, awaiting the Enable signal from the AU, AUG1WENA, to become true. The equation for enabling AUG1WENA is:

$$\text{AUG1WENA} = (\text{G0WI08} + \overline{\text{G1WS67}}) \cdot (\text{D0TSCA}) \cdot (\overline{\text{G1DQUA}}) \cdot (\text{G0WENE})$$

G0WI08 = Bit 8 of the "I" register true.

$\overline{\text{G1WS67}}$ = Bits 7 and 6 of the "I" register untrue.

D0TSCA = Time period T3 through T5 of the AU time counter.

$\overline{\text{G1DQUA}}$ = No quasi instruction being executed.

$$\text{G0WENE} = \overline{\text{F0XRMF}} \cdot \text{G1LSPI} \cdot \text{D1SC04} \cdot \overline{\text{G0NGS6}} \cdot \text{D1TSCC} \cdot \text{D1MSSI} \cdot \overline{\text{G0WLDX}}$$

$\overline{\text{F0XRMF}}$ = Remember flip-flop in AU, not set.

G1LSPI = API's not inhibited by console switch.

D1SC04 = Current instruction in sequence state 4.

$\overline{\text{G0NGS6}}$ = "S" bits of the instruction not equal to 6_8.

D1TSCC = Time period T1 through T3 of the AU time counter.

D1MSSI = No memory fence violation.

$\overline{\text{G0WLDX}}$ = Current instruction is not a GEN I, GEN II, LDP, or LPR and does not contain a bit 21.

The following summarizes the conditions that disable AUG1WENA, which halts interrupt response until AUG1WENA is enabled.

The API is withheld from servicing an interrupt immediately upon the completion of the following instructions to preserve program control:

- BRU Branch Unconditionally (14X*Y)
- BTS Branch if TSTF set (34X*Y)
- BTR Branch if TSTF reset (30X*Y)
- LDP Load Place (15X*Y)
- LPR Load Place and Restore (35X*Y)

- SPB Save Place and Branch (33X*Y)
- TXH Test X high or equal (24X0 -K)
- XEC Execute (04X*Y)
- Quasi entry instructions

The LDX (16X*Y) instruction is non-interruptible to facilitate the memory protect feature.

In order to preserve the "J" counter value, the following instructions prevent interruption immediately upon their completion:

- CLO Count least significant ones (05004137)
- CLZ Count least significant zeroes (05070137)
- CMO Count most significant ones (05004237)
- CMZ Count most significant zeroes (05070237)
- CNTO Count number of ones (05004337)
- CNTZ Count number of zeroes (05070337)

The JNR, JCB and JDR instructions are non-interruptible so that when a device is found ready, it is serviced immediately. If an interrupting TIM/TOM operation was allowed following one of these instructions, it would be possible for the program to return to a device no longer ready. The JNO instruction is non-interruptible only because the logic used in making other instructions non-interruptible is common.

Instructions that control the API System are non-interruptible. These are IAI, IAI_2, LMR, LMR_2.

PAI can be interrupted by non-inhibitable API's. Prior to the execution of PAI it would be expected that the PMT FF is reset. This condition would keep the inhibited API's inhibited until after PAI is executed. IAI, IAI_2, inhibit AUG1WENA which prevents an API interruption.

LMR, LMR_2 are decoded on AI logic sheet 10. G1ZINI is generated, routed to G0SPMT, and there used to DC clear the Sequence Control logic. This action prevents the next instruction from being accessed by the API module.

AUG1WENA is also disabled as long as the API Lockout Switch, located on the Programming and Maintenance Console, is in the lockout position. This prevents the control sequence from being executed. The interrupt exercising priority at the time the switch is disabled will retain priority until the switch is again enabled. If no interrupt priority was established at the time, future interrupts will be detected and their respective Change Detector flip-flops set. The first interrupt establishing priority after the lockout switch is disabled, will retain priority and be executed when the switch is again enabled.

Since the detected interrupts cannot be serviced while the switch is disabled, subsequent interrupts detected at the level change detector cannot be serviced but will be remembered. If a second interrupt occurs on the same line, however, its incidence will not be known because the flip-flop is already set.

When no inhibiting condition is present to disable AUG1WENA, the F1SPI1 flip-flop is enabled to be set at the next clock:

$$F1SPI1 = G1STE3 \cdot AUG1WENA \cdot \overline{F1SPI2} \cdot CLK$$

Control Cycle

Setting F1SPI1 enables drivers D0SPI1, D1SCD1 and D1SCD2. On Change Detector cards, the D1SCD1 or D1SCD2 signal is ANDed with the group Priority driver signal from the Address Generator. The Level Change detector flip-flop for the interrupt currently being serviced is cleared. No others are affected.

Signal D0SPI1 is applied to the "P" register control and to sequence control circuits of the AU so that with the setting of F1SPI2 the "P" register does not increment and memory is addressed from the API area (AUG0SAMW is enabled).

$$F1SPI2 = G1STE3 \cdot F1SPI1 \cdot SC01 \cdot CLK$$

The most significant bit of the API response address, bit seven, is enabled as a result of the F1SPI1 flip-flop being set:

$$AUD1MA07 = AUG0SAMW = AUG1SWPI \cdot SC01$$

$$AUG1SWPI = D0SPI1 = F1SPI1$$

Memory is requested to fetch the response instruction as soon as the sequence counter enters SC01. With the address established by the API logic present on the memory address lines (AUD1SAMW), the instruction is fetched from the API response address and loaded into the "I" register of the AU from where it is executed. At the clock following the last pulse of this SC01 time, F1SPI1 clears and drivers D1SCD1 and D1SCD2 are disabled (refer to Fig. OPR. 8). This allows the Interrupt flip-flop for the interrupt that was just serviced to set again if another interrupt on the same line occurs. Driver D0SPI1 also disabled, disabling the addressing of memory from the API module (SAMW).

$$\overline{F1SPI1} = LPE \cdot SC01 \cdot CLK$$

$$\overline{D1SCD1}/\overline{D1SCD2}/\overline{D0SPI1} = \overline{F1SPI1}$$

At least one instruction must follow the interrupt response instruction before another interrupt can be serviced. This is accomplished by holding F1STC1, F1STC2 and F1SPI2 set during the execution of the API response instruction and during the fetch of the next instruction.

Clear Priority signals, D0SCP1 and D0SCP2, are enabled at SCB of SC01 for the instruction that follows the API response instruction. These signals "DC clear" F1STC1 and F1STC2. All Priority flip-flops on the Change Detector and Priority cards are also cleared, enabling the No Interrupt Present signals.

$$D0SCP1/D0SCP2 = F1SPI2 \cdot \overline{F1SPI1} \cdot CTAE$$

$$\overline{F1STC1}/\overline{F1STC2} = D0SCP1$$

$$\overline{F1*GP\#} = F1*GP\# \cdot \overline{G1*PEL(U)}$$

$$\overline{G1*PEL(U)} = D0SCP1(2) + D1SPAI(2) + AUD0NIN2 + F0ZMxx$$

The interrupt cycle is completed when the Priority Interrupt flip-flop, F1SPI2, clears at the following clock.

$$\overline{F1SPI2} = \overline{F1SPI1} \cdot CTAE \cdot SC01 \cdot CLK$$

SPB Control Cycle

Refer to Fig. OPR. 9 as an aid to this discussion. If the API response instruction is an SPB, a non-interruptible instruction (see note) the sequence control is terminated at T3 of the execution state (SC04) of the response instruction. This is accomplished by enabling the Initialize signal, D0SINT, during this period. D0SINT enabled, "DC clears" the F1STC1, F1STC2, F1SPI1, and F1SPI2 flip-flops. D0SINT is enabled by the following equation:

$$D0SINT = G1SINI = G0SPMT = G1ZINI$$

$$G1ZINI = AUD0NIN1 = AUG1MRTM + AUG1MIAP$$

The Priority flip-flops are not cleared by D0SCP1(2) but are cleared instead as a result of the SPB ANDed with SC04, TSCA, and SPI2.

$$AUG1MRTM + AUG1MIAP = SC04 \cdot DSPB \cdot TSCA \cdot SPI2$$

$$MRTM = AUD0NIN1 = AUG1LAP1 = AUD0LAP1,2$$

$$\overline{Priority\ FF} = \overline{G1*PEL(U)} = \overline{G0*PEL(U)} = AUD0NIN2 + AUD0LAP1,2$$

> NOTE
>
> An instruction which is said to be "non-interruptible" means that the instruction following it will be an instruction from a running program and not an API response instruction. ALL instructions executed from an API location, as a result of an API, are non-interruptible.

AUD0LAP1, 2 is used to reset Priority flip-flops in all areas except the first three (200, 201, 202). These three are reset by AUD0NIN2.

The Priority flip-flops are cleared during SC04 of the SPB response instruction and the level change detectors cannot establish a control cycle until AUG1WENA is enabled during SC04 of the command following the SPB. Because the SPB resets the PAI flip-flop (F1WPMT) only non-inhibitable Priority flip-flops (200-277) can be set by their change detectors until PMT is again set.

SPB, LPR, IAI Control Cycle

The execution of these instructions, as part of the running program, results in the generation of D0SINT which DC clears the API sequence control during SC04 rather than SC01 as does the other commands exemplified by the timing diagram, Fig. OPR. 8. The generation of D0SINT prevents an API cycle from starting, therefore, the SPB, LPR, or IAI is non-interruptible.

$$D0SINT = G1SINI = G0SPMT = AUG1CTAE \cdot AUG1WCPM$$

$$AUG1WCPM = (G0DSPB + G0WGS3 + G0XLPR) \cdot G0WISS \cdot D1TSCC \cdot D0WC04$$

Because these commands were not the result of an API fetch-execute cycle, F1SPI1, 2 will not be set. AUG1MRTM cannot be generated if SP12 is not set (SP12·SPB), therefore, the Priority flip-flops will not be reset as was the case when the SPB is executed as the result of an API. The inhibitable Priority flip-flops (300-377) however, are reset due to the reset of F1WPMT, which is the result of the execution of the commands being discussed (SPB, LPR, IAI).

In summary, there are four basic types of instructions which affect the API logic:

1. The instructions which inhibit AUG1WENA (non-interruptible instructions) prevent the next instruction from being an API response instruction.

2. Non-interruptible instructions, SPB, LPR, IAI not only inhibit AUG1WENA but also reset the API sequence control and reset F1WPMT (LPR bit 21=0). The reset of F1WPMT causes all inhibited (300-377) Priority flip-flops to be reset and held reset until F1WPMT is again set.

3. Instructions executed from an API location, as a result of an API are all non-interruptible, regardless of the instruction. The API sequence control times out and resets all of the Priority flip-flops during SC01 of the command following. An example of this (DMT) is shown in Fig. OPR. 8. One exception is covered in the next paragraph.

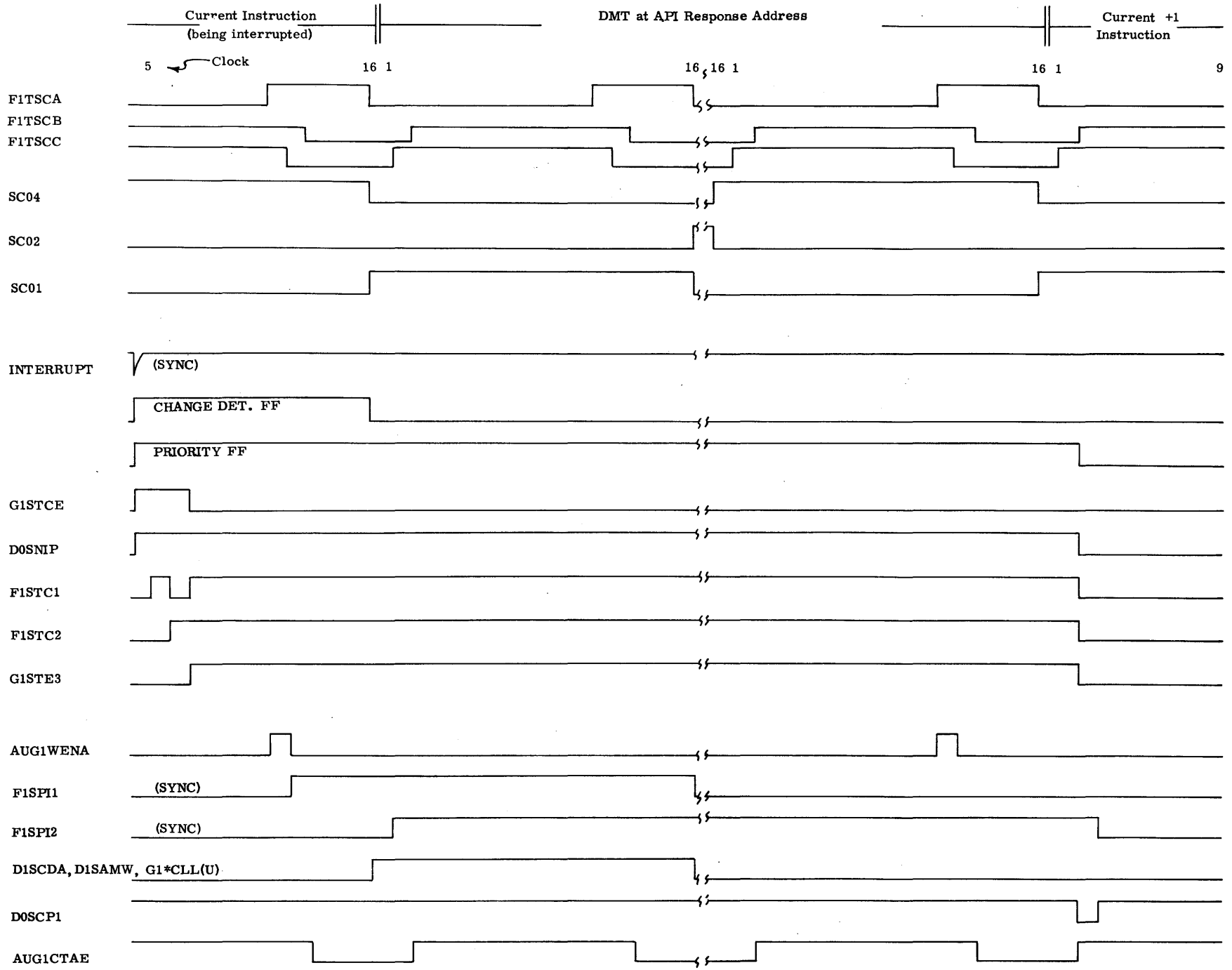


Fig. OPR. 8 API DMT Sequencing

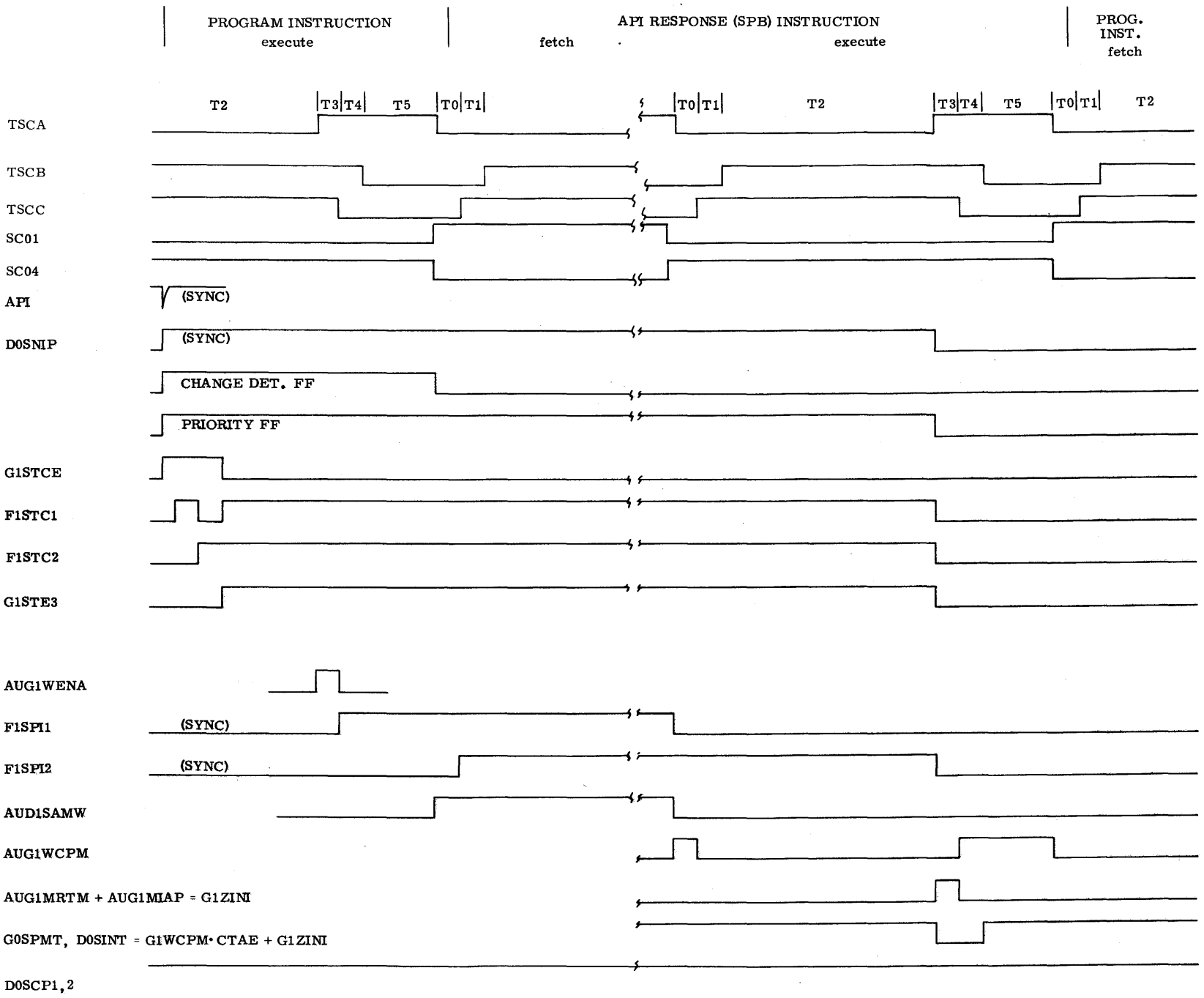


Fig. OPR. 9 API SPB Sequencing Timing

4. An SPB executed from an API location, as a result of an API, or from a trap location (20 or 24) as a result of a Memory Protect or Watchdog trap, will enable G1MTRM. The API sequence control and the Priority flip-flops will be cleared during SC04 of the SPB.

ECHO INTERRUPT GENERATION

Echo interrupts are special interrupts that are normally used to keep track of time counters and to signal the running program when table empty or table full conditions exist.

The API echo signal may be generated by a interrupting DMT or by a TIM/TOM function. The DMT echo occurs whenever a DMT instruction is executed from an interrupt response address and the count passes from zero to minus one. The TIM or TOM echo occurs when the word count reaches the table full or empty condition.

Any interrupt that occurs on a eight point card equipped with optional echo generator gates will cause an echo interrupt to occur when all of the following conditions are met.

- The output of the echo generator is wired to the input of another change detector.
- The first interrupt has been recognized as the one being processed by the API sequencer.
- The Generator Echo Interrupt flip-flop, F1SGEI is set.

For example, the Generate Echo gate for interrupt 233, designated G0EGE3, is enabled when:

G0EGE3 = D1EGB0 · G1EGP3 · D1ADEP · G1EGEL

The first three signals identify 233 as the interrupt being processed by the API sequencer in the same way the address generator recognized it as the interrupt for which to generate a response address.

That is:

D1EGB0 = (3+2+1+0) This term indicates a Priority FF on that half card group is set.

G1EGP3 = $(3 \cdot \overline{2} \cdot \overline{1} \cdot \overline{0})$ The exact Priority FF is now determined.

D1ADEP = $E \cdot \overline{B} \cdot \overline{C}$ Finally, the Group Priority signal from Area A indicates no higher priority FF is set.

Signal G1EGEL is true when the Generate Echo Interrupt flip-flop is set:

G1*GEL(U) = D0SGEI

D0SGEI = F1SGEI

The Generate Echo Interrupt flip-flop, F1SGEI, sets if the current API response instruction is a DMT and the count passes from a zero to a minus one or when the API response location contains a TIM/TOM control word and the word count reaches a table full or empty condition.

F1SGEI = F1SP12 · AUG1WEKO · CLK

AUG1WEKO = $\overline{\text{A1U23C}} \cdot \overline{\text{SCB}} \cdot$ SCA · AUG0WEK1 + AUG0WEK2

AUG0WEK1 = SC02 · DMT

AUG0WEK2 = SC01 · AUG1HECO (TIM/TOM)

SPECIAL CONTROL

Special controls are important factors in the operation of the API system. These include initialize control, API lockout control, inhibitable interrupt control, and control to inhibit all interrupts by instruction.

API ENBL Switch Control

The API ENBL switch located on the programming and maintenance console is operable only when the console key switch is in the enable position. The API ENBL switch is normally used when performing maintenance procedures. The switch is a latching type and when in the disable position, all interrupts are inhibited.

Ground from the API ENBL switch in the disable causes its Enable Interrupt gate signal, AUG0WENE, and the Enable API signal, G1WENA, to be disabled.

$\overline{\text{AUG0WENE}}$ = AUSWCAPI

$\overline{\text{AUG1WENA}}$ = $\overline{\text{AUG0WENE}}$

The Priority Interrupt flip-flop, F1SPI1, in the sequence control area, is prevented from setting, thus prohibiting the addressing of memory from the API system.

IAI, Instruction Control

The Inhibit Automatic Interrupt instruction, IAI, is used to prevent critical program routines from interruption by low priority interrupts.

Only those interrupts classified as inhibitable (300-377) are affected by the IAI instruction. Any half-card group of four interrupts is rendered inhibitable when the appropriate input of the applicable Priority Enable gate G0*PEL or G0*PEU is wired to the output of the Permit Automatic Interrupt driver D1SPAI or D1SPA2.

Inhibitable interrupts are inhibited when the IAI_1 instruction is executed which clears the Permit flip-flop disabling D1SPAI and D1SPA2.

$\overline{\text{D1SPA1}} = \overline{\text{AUF1WPMT}}$

$\overline{\text{D1SPA2}} = \overline{\text{AUF1WPMT}}$

Inhibitable interrupts are allowed when the Permit flip-flop is set by the PAI (Permit Automatic Interrupt) instruction.

Initialize Control

When the computer is initialized (ON/INIT switch is pressed), the setting of Interrupt flip-flops due to erratic signals on API input lines is prevented. Initialize signals also clear all Priority flip-flops, the Time Counter flip-flops, Mask Register, and the Generate Echo flip-flop in the sequence control area.

AUD0NIN1 is applied to the first half card change detector group directly. All other half card groups are initialized by AUD0LAP1(2) which is a combination of AUD0NIN1 or Program Load. During Program Load it is necessary to hold all except the first half card group in an initialized state.

In the sequence control area, signal D0SINT clears the Time Counter flip-flops, Priority Interrupt flip-flops, and the Generate Echo flip-flop.

D0SINT = G1SINI

G1SINI = G0SPMT

G0SPMT = INITIALIZE

The INITIALIZE signal is from gate G1ZIN1.

The Mask Register is initialized by G1ZBGC as shown on logic sheet 10.

IAI_2 Instruction Control

The IAI_2 instruction inhibits all interrupts, both those that are inhibitable and those that are non-inhibitable by the IAI_1 instruction, provided the Permit flip-flop is clear. The decoding and control structure for the IAI_2 instruction, is a part of the control circuit for the mask register.

The Set Inhibit flip-flop gate, G0ZSIF, is enabled during sequence state 4 of the IAI_2 command (25000304_8). G0ZSIF is true when all of the following signals are true:

- $\overline{\text{G0ZK01}}$

  $\overline{\text{G0ZK01}} = \overline{\text{AUG1NK01}}$

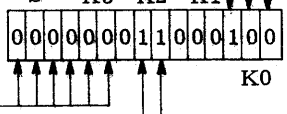
- $\overline{\text{G0ZK02}}$

  $\overline{\text{G0ZK02}} = \overline{\text{G1ZK02}}$

  $\overline{\text{G1ZK02}} = \overline{\text{AUN0NK02}}$

- G1ZK04

  G1ZK04 = AUN0NK04



- AUG1DKS0
- AUD1NK22
- AUD1NK21
- G1ZGS0

  G1ZGS0 = $\overline{\text{AUG1BC12}}$

  $\overline{\text{G1BC12}} = \overline{\text{SC01}} \cdot \overline{\text{SC02}}$

The Inhibit flip-flop is set by G0ZSIF if the Permit flip-flop is clear:

F1ZINH = G0ZSIF

F0ZINH = F1ZINH$\cdot \overline{\text{AUF0WPMT}} \cdot \overline{\text{AUD0NINI}}$

All interrupts are prevented from interrupting the program because API sequencing is halted. This is accomplished by "DC clearing" the Time Counter flip-flops, the Priority Interrupt flip-flops, and the Generate Echo flip-flop with signal D0SINT.

D0SINT = G1SINI

G1SINI = G0SPMT

G0SPMT = G1ZINI

G1ZINI = D0ZCM1 + D0ZCM2 + F0ZINH + AUD0NIN1

The Inhibit flip-flop is cleared either by the PAI instruction which sets the Permit flip-flop or when the computer is initialized:

$\overline{\text{F1ZINH}}$ = AUF0WPMT + AUD0NINI

API Test Hardware

To use the API test feature, the Change Detector card is pulled from its slot and plugged into one end of the special test extender (PX1000SXBA5). The other end of the extender is plugged into the vacated card slot. The cable attached to the test extender card is mated with the connector on the rear of the mask register control card. This opens the normal API input path and connects the inputs of the change detectors to outputs of gates in the API Test logic via cable. All other leads pass directly through the test extender. See Fig. OPR. 10.

Each of eight GEN II instructions is used to set one Interrupt flip-flop on the eight-point card under test. A decoded API test instruction enables an associated test gate. An interrupt is simulated when the gate's output goes from 3.6 volts to zero during the execution of the instruction. The list below shows which test gate is enabled and which Interrupt flip-flop sets for a given instruction.

Octal Code	Test Gate	Interrupt Flip-Flop	API
25000210	G0T200	F1*GI0	XX0
25000220	G0T201	F1*GI1	XX1
25000242	G0T202	F1*GI2	XX2
25000212	G0T203	F1*GI3	XX3
25000222	G0T204	F1*GI4	XX4
25000244	G0T205	F1*GI5	XX5
25000214	G0T206	F1*GI6	XX6
25000224	G0T207	F1*GI7	XX7

In addition to the decoding logic for the S and K bits of the instruction, all test gates are enabled with signal $\overline{AUG1BC12}$. This inhibits decoding during sequence states 1 and 2. Sequence state 2, however, will not normally be used because API test instructions are not usually indexed.

With all API inhibiting functions disabled, the interrupts may then be tested in groups of eight by executing the above commands each time the text extender is connected to a Change Detector card. Successful operation of the API channel may be determined if the corresponding API address is accessed with the execution of the command relating to that address. The first listed command (25000210) will correspond to the first API address of the group in all cases (200_8, 210_8, 220_8, etc., depending upon the location in which the test card is connected) and the following commands will access sequential addresses from the initial one mentioned.

JUMPER PIN SELECTION

Jumper pins, within the API module, are provided to select an alternate API input signal when the "standard" input is not implemented. Jumper pins are also provided to optionally disable the TIM/TOM function associated with an API response address permitting the operation of a SPB, BRU, or DMT type of full operand command as the API response instruction.

Jumper pin selection of secondary API inputs is provided on the NBTA5 boards in slots A28A-E and A20A-E and the NSLA1 board in slot A29. Inserting a jumper pin selects the secondary input.

Jumper pins on the NPDA5 in card slot A28 are provided to disable the TIM/TOM function associated with certain response addresses, permitting full operand instructions (DMT, SPB, BRU) to be executed from the response address. Inserting a jumper pin disables the TIM/TOM function.

A cabinet modification drawing is provided for each system illustrating the jumper pins to be inserted and the API response addresses affected.

EXTERNAL API BUFFER ADDER OPTION

This option (4DP4800AS10) provides for up to 24 external interrupts in the form of isolated, normally open, contact sets. The input to the Buffer is via the C-panel.

All interrupt inputs, except those assigned number 23 (K23) and 24 (K24), may either be used for contact closure or bypass the contact set if the input has sufficient line drive capability. External interrupts number 23 and 24 are restricted to inputs requiring contact closure. If the bypass feature is implemented, a shorting piggy-back is inserted on the KNDA1 PWB. Any contact closure input requiring signal settling time uses a piggy-back delay network to provide 70 to 150 ms. delay in the API response. The cabinet modification drawing for the API module specifies the part number of the piggy-backs and their placement on the KNDA1 board.

The option is implemented by installing the KNDA1 PWB and the I-panel to A-panel cable. Logic for this option is shown on sheets 80, 81, and 82 of the basic API logic, 70C180954.

EXPANDED API/TT ADDER OPTION

This plug-in option (4DP4800AS12) increases the API input capability from 64 to 128 and the available TIM/TOM channels from 24 to 48. The fundamental operation is the same as the basic API module.

The pre-assigned standard interrupt structure for the option is as follows:

1. Non-inhibitable interrupts are assigned addresses 220-227, 240-247 and 260-277.

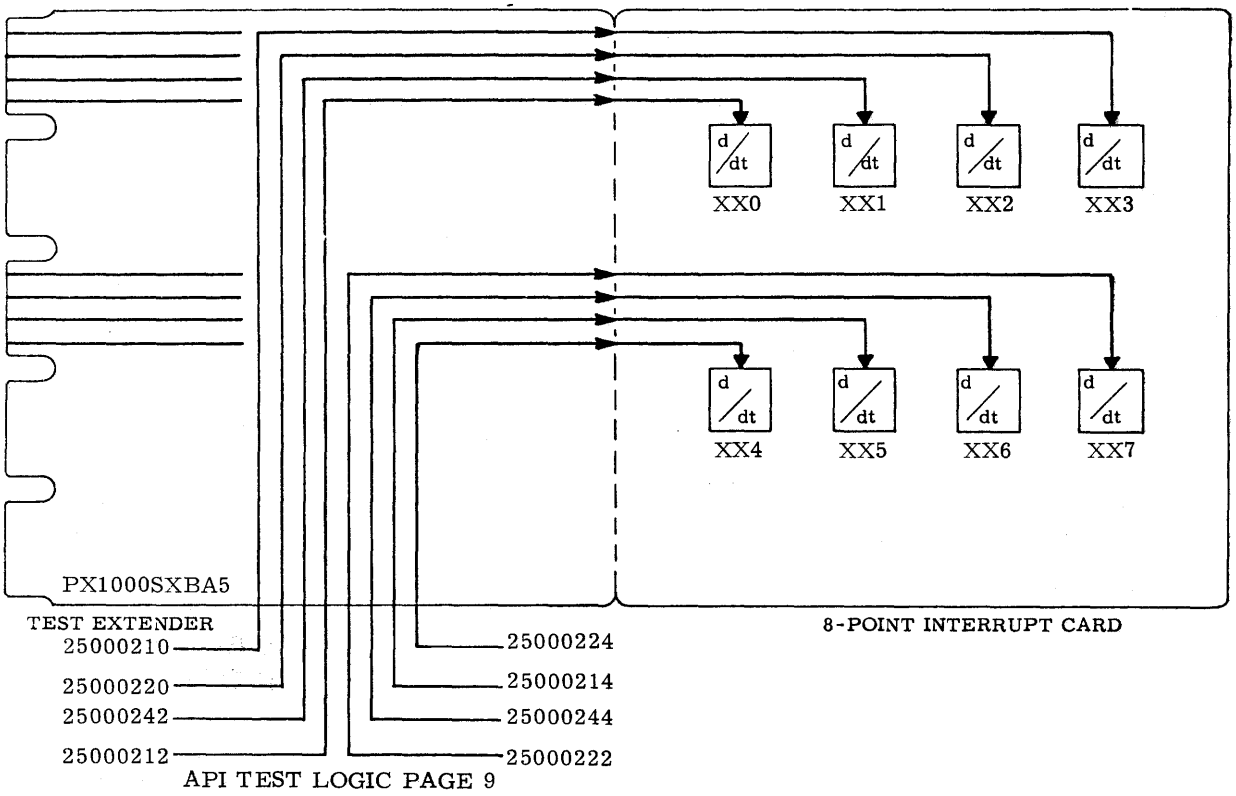


Fig. OPR. 10 Simplified Schematic of Interrupt Card on Test Extender

2. The assigned inhibitable interrupt addresses are 310-317, 330-337, 350-357 and 370-377.

This option is implemented by adding the K-panel, printed wiring boards and associated cables. The logic for this option is provided in 70C180954.

ASU API NECKDOWN OPTION

This option (4DP4802AS04) provides for up to 24 external interrupts in the form of isolated, normally open, contact sets. The inputs to the neckdown option is via terminal boards in the ASU cabinet. This option is shown in GE logic drawing 70C181085.

All interrupt inputs may either be used for contact closure or bypass the contact set if the input has an adequate line drive capability. If the bypass feature is implemented, a shorting piggy-back is inserted on the KNDA1. The piggy-backs, if required, are specified on the Cabinet Modification drawing for the API provided with each system.

Any contact closure input requiring signal settling time uses a piggy-back delay network to provide 70 to 150 ms. delay in the API response. Interrupts from the Neckdown module are routed through the C-panel of the ASU, via a cable to the C-panel of the CSU, and then to the API input.

PROGRAM LOAD

The program load feature is described in greater detail within the 4022D Arithmetic Unit description under the heading "Console". The general function and interrelation of the program load feature to the 4032D API is discussed in the following text.

The program load functions as a minimum or bootstrap loader; that is, it initiates the read-in of a basic loading program which can, in turn, be used to load another more sophisticated loader or operating routine.

A program load operation may be initiated from any of the following switches provided on the 4022D Maintenance Console:

TAPE - Loads from paper tape reader

CARD - Loads from card reader

BULK - Loads from a drum or disc

Executing a program load sequence results in the activation of the device that corresponds to the program load switch enabled. If a program load from one of the I/O Buffer devices (paper tape or card reader) is initiated, a TIM control word is stored into API response addresses 201_8 and 202_8 and location 000_8 is loaded with zeroes.

Upon placing the AUTO/MAN switch in AUTO and raising the API ENBL switch, the device is activated by an OPR command (25027000_8). A device enable code (21_8) was also placed in the "A" register during this sequence and is used to enable the paper tape reader but ignored by the card reader.

> **NOTE**
>
> **If the K_1K_0 address of the I/O Buffer device being loaded differs from 00_8, the correct address bits should be entered into the "B" register prior to switching to AUTO.**

On reaching the first character, an interrupt is generated which will transfer program control to the API response address to which the device is connected, either 201_8 or 202_8. The TIM control word which was stored in the response address will control both the packing of the words (two characters/word for the card reader and four characters/word for the paper tape reader) and the incrementing of the addresses into which the words are stored as they are read from the device. Each character read will generate an interrupt and the inputs will continue until the table is full.

During the loading procedure, all API response locations, 203_8 and higher, are inhibited. In addition, mask register bit zero flip-flop, F1ZM00, in the API logic is cleared. This permits the receipt of the program load device interrupt (as each character is read) in the event the mask register bit had been left in the set condition. D0ZCOM, the signal clearing the mask register bit zero, is only enabled during the period that the program load control logic is storing the TIM control words into the API response addresses, subsequent to the depressing of one of the program load switches. The inhibit on API responses for addresses 203_8 and higher remains until the program load switch is returned to its disabled position.

D0ZMAN = AUG0LSTR

G0LSTR = $\overline{\text{F1LPEN}}$·G1LPLF + F1LPL1 = Time period during which locations 000_8, 201_8 and 202_8 are automatically loaded by program load sequence.

AUD0LAP1/2 = G1LAPI = G1LPLF = Time period that program load switch is enabled.

API response location 203_8 is prevented from receiving interrupts during the program load sequence by a unique path. The level change detector input is contained in the program load section of the AU logic. The input is connected to G0LAP3 which is inverted and ANDed with G0LPLE to provide the level change input signal, G1LAP3, which is connected to the level change detector in the API. The program load enable signal, G0LPLE, enabled during the time that a program load switch is enabled, will prevent interrupts on that line from being detected.

(API)F1B203 = AUG1LAP3 (negative transition) = Interrupt·G0LPLE

The TIM control word automatically stored into the API response addresses is dependent upon the device program load initiated. When a program load from a paper tape reader is initiated, zeroes will be stored into locations 201_8 and 202_8. If a paper tape reader is utilized for program loading, it must be connected to one of these two response addresses. The TIM control word accessed by each interrupt will control the packing of four characters into the "B" register of the AU and the storing of these words beginning at address 001. (Y location of TIM control word plus one). Reference should be made to the TIM/TOM description of the 4022D Arithmetic Unit for specifics of the control word definition. Since the word count field of the control word is zeroes, a possible 63 words may be read from the paper tape reader by this program load sequence.

A card reader utilized for program loading must also be connected to either address 201_8 or 202_8. The TIM control word, accessed by each interrupt, will control the packing of two characters/word as a result of the program load sequence storing bits 15 and 17 into the two API response addresses. The beginning address into which the information read from the card is stored, as with the paper tape reader, will be address 001, since the Y field of the control word was set to zero. The maximum possible number of words (two 12 bit characters/word) will again be 63, since the word count was set to zeroes.

A program load from a bulk memory device differs somewhat from the paper tape and card reader loading operation. Location 000 is cleared, address 001 is loaded with an octal 100 and address 002 is cleared.

An OUT command for the bulk device (25041000_8) is placed into the "B" register and executed when the AUTO/MAN switch is switched to AUTO. Three control words (locations 000, 001, and 002), are accessed as a result of the execution of the OUT command to the controller of the bulk device. The three control words, will inform the bulk device to transfer 64 (octal 100) words to the core memory, beginning at address 000. In each of the program load sequences, the enabled program load switch must be returned to its disabled position before entering the next program.

READER COMMENTS

The General Electric Company solicits your comments on publications covering Process Computer equipment. Please explain any "no" responses in the COMMENTS section. Your comments and suggestions become the property of the General Electric Company.

- Name of Manual: _____
- What is your computer application: _____

_____

- How is this publication used:

Familiarization	☐	Reference	☐
Training	☐	Maintenance	☐

Other (Explain) _____

	YES	NO
• Does this publication meet your requirements	☐	☐
• Is the material:		
1) Presented in clear text	☐	☐
2) Conveniently organized	☐	☐
3) Adequately detailed	☐	☐
4) Adequately illustrated	☐	☐
5) Presented at appropriate technical level	☐	☐

- Please provide specific text references (page number, line, etc.) with your comments.

NAME _____ DATE _____

TITLE _____

COMPANY NAME _____

AND ADDRESS _____

COMMENTS:

No postage necessary if mailed in the U. S. A.

Staple

Communications concerning Technical Publications should be directed to:

Manager, Technical Publications
Utility and Process Automation Systems Operation
2255 West Desert Cove Road
Phoenix, Arizona 85029

Fold Fold

FIRST CLASS
Permit No. 4091
Phoenix, Arizona

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

GENERAL ELECTRIC COMPANY
UTILITY and PROCESS AUTOMATION
SYSTEMS OPERATION
2255 West Desert Cove Road
Phoenix, Arizona 85029

Attention: Technical Publications

Fold Fold

Cut Along Line

Additional Comments:

4078B I/O EXPANDER

TABLE OF CONTENTS

4078B I/O EXPANDER

INTRODUCTION

The 4078B Input/Output Expander is a part of the 4DP4800BS011 basic Central Systems Unit (CSU) for the GE-PAC* 4010 Process Computer System. It provides buffering, drive and expansion for the data, control signals, API, and test line interface between the Arithmetic Unit, and the I/O Buffer, process I/O controllers, and other I/O Modules.

All communications between the Arithmetic Unit and these devices occur as a result of the execution of GEN 2 commands or TIM/TOM.

MODEL NUMBERS/OPTIONS

The following model numbers and options are associated with the 4078B I/O Expander.

4800BS011 - Basic CSU (Includes basic 4078B)

4800AS131 - First external port adder

4800AS132 - Second external port adder

4800AS133 - Third external port adder

Each external port permits connection of an I/O Coupler or similar module.

REFERENCE DOCUMENTS

The following documents are associated with the I/O Expander.

Logic:

70C180899 - 4078B I/O Expander

Wiring:

70A120523 - A/B Panel Wiring

70A105910 - Cable between MJ89 and BJ89

70A105912 - Cable between MJ85 and BJ85

70A105913 - Cable between MJ83 and ZJ70 (to PDC)

70A105913 - Cable between MJ82 and WJ59 (to AIS)

70A105913 - Cable between MJ84 and ZJ79 (to VOC)

70A105914 - Cable between MJ87 and WJ69 (from AIS)

70A105914 - Cable between MJ88 and ZJ71 (from PDC)

70A105917 - Cable between MJ80 and AJ80 (API)

70A119917 - Cables, I/O Coupler to I/O Expander

Interface Logic:

70C180955 - Arithmetic Unit

70C180954 - Automatic Program Interrupt

70C181085 - ASU API Neckdown

70C180909 - Basic I/O Buffer

70C180975 - ASU I/O Coupler

70C180914 - CSU Power

SYSTEM INTERFACE

Fig. EXP. 1 illustrates the basic interface connections of the I/O Expander with the other system components. This drawing also provides logic and cable wiring drawing numbers associated with the interface modules. As shown in the figure, the I/O Expander interfaces with the Arithmetic Unit, Automatic Program Interrupt module, Process Digital Controller, Analog Input Subsystem, Variable Output Controller, GE-TAC* Controller, and the I/O Buffer within the CSU. In addition, up to three I/O Couplers or similar modules in the Auxiliary System Unit (ASU) cabinet may be connected to the I/O Expander.

Fig. EXP. 2 illustrates the location of the I/O Expander and associated interface modules within the GE-PAC 4010 Central Systems Unit. As shown by the card arrangement in Fig. EXP. 2, the I/O Expander is located in panels M and B of the CSU. Sheets 5 and 6 of the I/O Expander logic lists the interface connections.

OPERATION

The expander provides an expansion of communication paths thru which the Arithmetic Unit may communicate with input and output modules. In addition to transferring data to and from these devices, the Arithmetic Unit is able to determine the ready and error status of the devices, Initialize them, and manually clear alarms that exist within them.

The control signals associated with these functions, the actual data transfer paths, and the ready and error test line data paths thru the I/O Expander are described separately in the following paragraphs. This discussion is limited to the I/O Expander and Interface. For further details, refer to the theory of operation of the AU, API, or I/O Subsystem contained in this book set.

*Registered Trademark of General Electric Company

Control Signals

Data transfers and ready and error status monitoring are controlled by the execution of GEN 2 commands (OPR, ABT, OUT, IN, JNR, JDR, JCB, JNE, SEL) or TIM/TOM. Therefore, the S, K, and phase A and phase B timing signals are routed from the AU through the I/O Expander to the subsystem.

Table EXP. 1 lists the S (14 - 12) and K bit (11 - 0) control of the GEN 2 or TIM/TOM control word associated with various subsystems. Fig. EXP. 3 illustrates the logic path of these control signals through the I/O Expander and the interconnections with the AU and the subsystem.

The "Alarm Clear" signal originates at the ALARM CLEAR switch on the Programming and Maintenance console and is applied through the I/O Expander to the subsystems permitting alarms within the subsystem to be manually cleared. This interface is also shown in Fig. EXP. 3.

The Line Frequency or Adjustable Pulse Generator signal is applied from the AU through the I/O Expander to the Pulse Source Initiator option of the Process Digital Controller. This interface is also shown in Fig. EXP. 3.

Automatic program interrupt signals from the subsystem are also routed through the I/O Expander to the API module as shown in Fig. EXP. 3.

Data Input

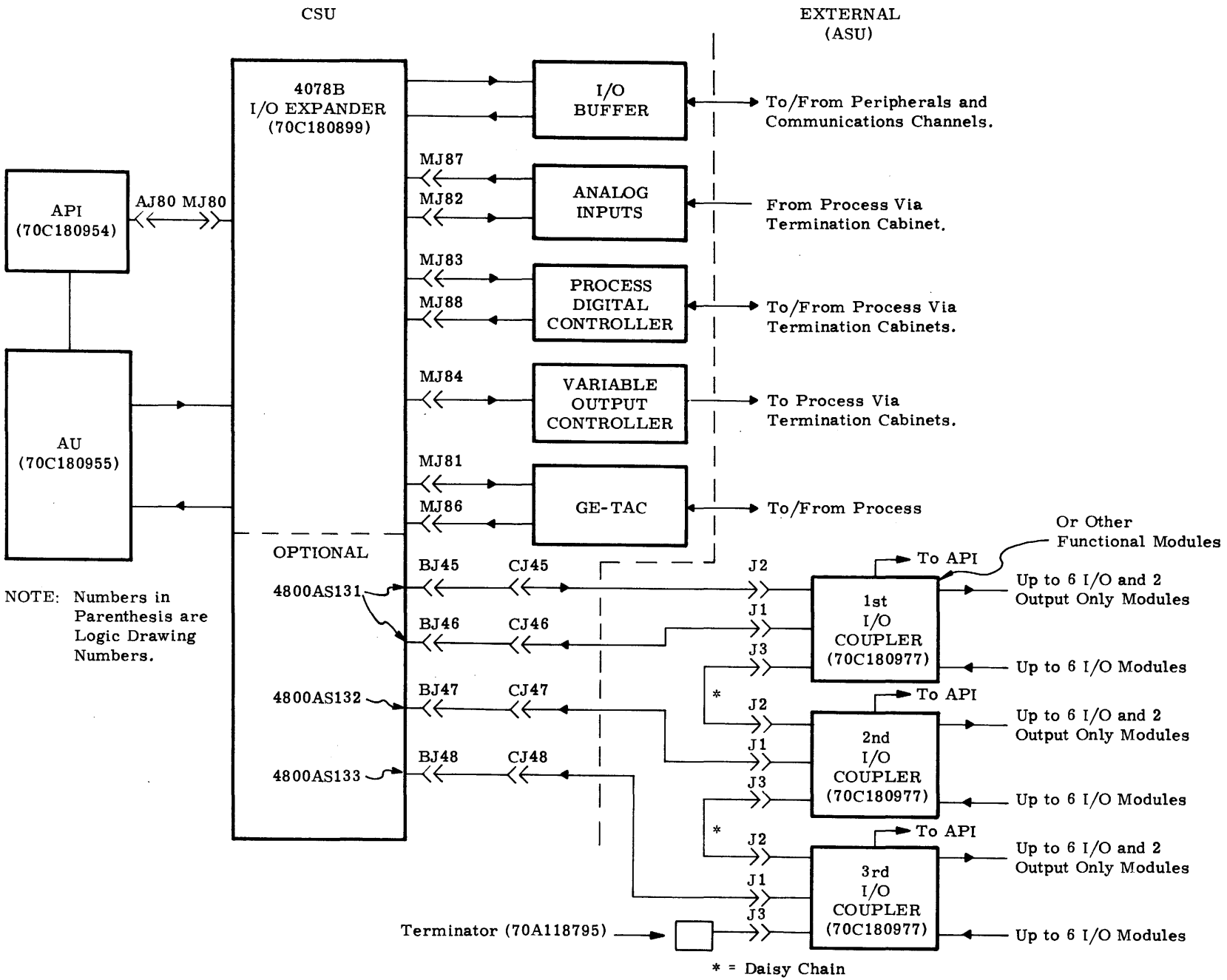
Fig. EXP. 4 illustrates the path of data from the devices connected to the I/O Buffer, Analog Input Subsystem, or Process Digital I/O Subsystem to the Arithmetic Unit. Input data transfers occur as a result of the execution of an IN command or the TIM function. Basic timing associated with data transfers is shown in Fig. EXP. 7. Logic drawing numbers and sheet references are provided in these figures for easy reference.

Data Output

Fig. EXP. 5 illustrates the path of data from the Arithmetic Unit to the Analog Input Subsystem, Process Digital I/O Subsystem, or the devices connected to the I/O Buffer. Output data transfers occur as the result of the execution of the OUT command or the TOM function. Basic timing associated with data transfers is shown in Fig. EXP. 7. Logic drawing numbers and sheet references are provided in these figures for easy reference.

Test Lines

Fig. EXP. 6 illustrates the test line (ready or busy) signal path from the devices through the I/O Expander to the Arithmetic Unit. Using GEN 2 instructions (JNR, JCB, JDR, JNE) the status of these lines can be monitored to provide program control.

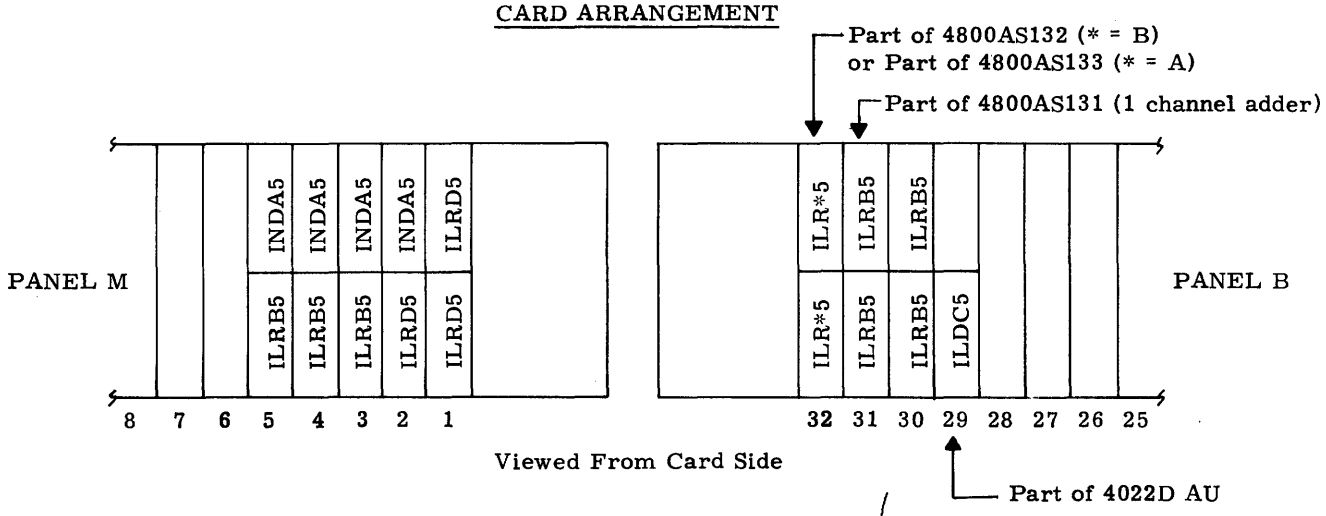


NOTE: Numbers in Parenthesis are Logic Drawing Numbers.

Fig. EXP. 1 I/O Expander Interface, Block Diagram

I/O EXPANDER

CARD ARRANGEMENT



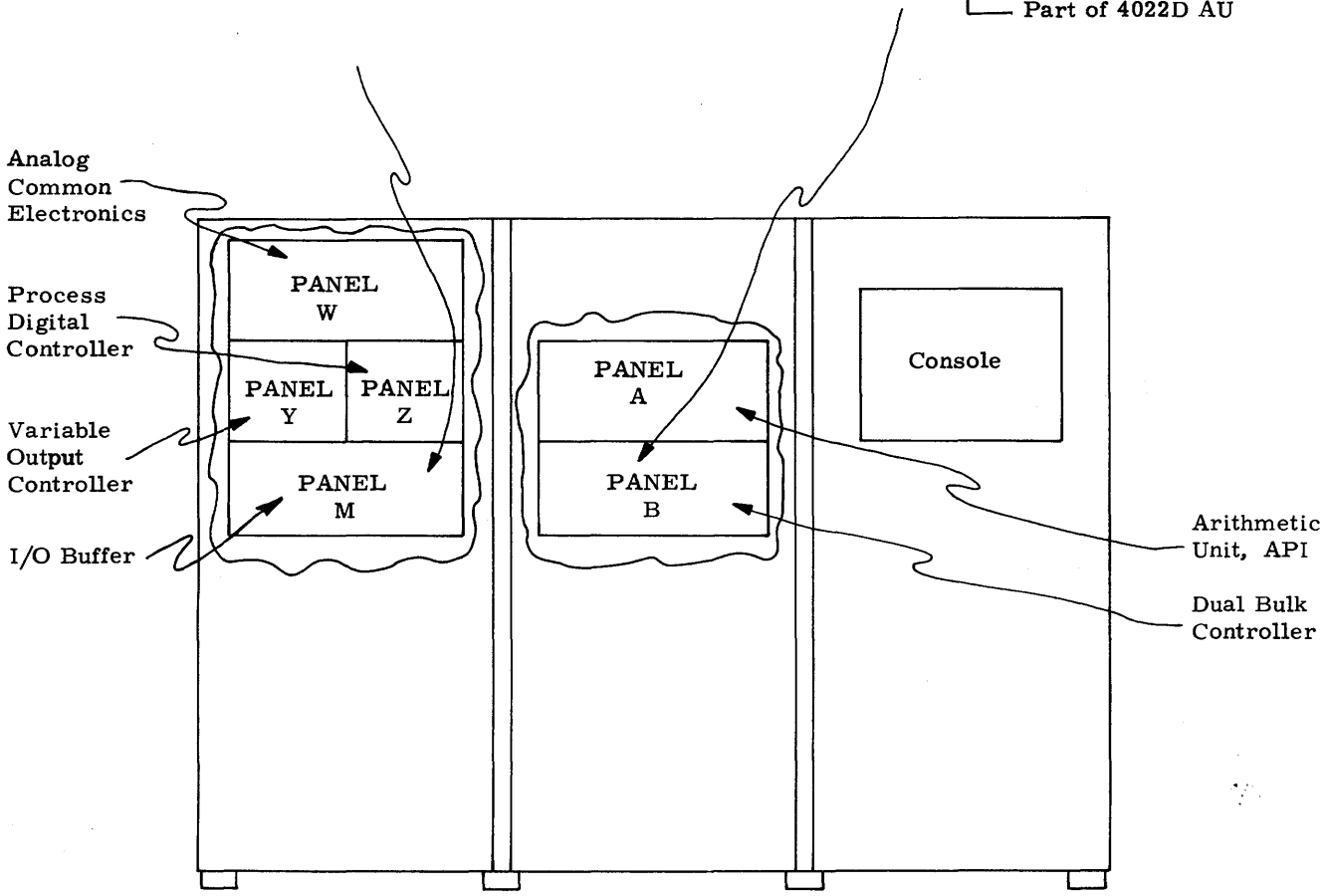


Fig. EXP. 2 GE-PAC 4010 Central Systems Unit

GEN 2 Command or TIM/TOM Matrix Bits	14 12 S	11 9 K3*	8 6 K2*	5 3 K1	2 0 K0
I/O BUFFER	2-OPR 3-ABT 4-OUT, TOM 5-IN/ TIM 6-JCB, JDR, JNR 7-JNE	7	0	Bits 3-0 = Channel Address Bits 5, 4 for JCB, JDR	
Process Digital Controller	0-SEL 1-ACT 3-ABT 4-OUT, TOM 5-IN, TIM 6-JNR 7-JNE	2	0	IN = Group Address OUT, ABT, SEL } Sub Operation	
Analog Input Controller	1-ACT 4-OUT, TOM 5-IN, TIM 6-JCB, JDR	4	0	Bit 4, 5 for JCB, JDR	
Variable Output Controller	1-ACT 4-OUT, TOM 6-JNR 7-JNE	2	7	Not Used	

* The K3, K2 device codes listed are those most common. Always refer to the Hardware Address Summary for actual device codes used.

Table EXP. 1 S and K Bit Functions

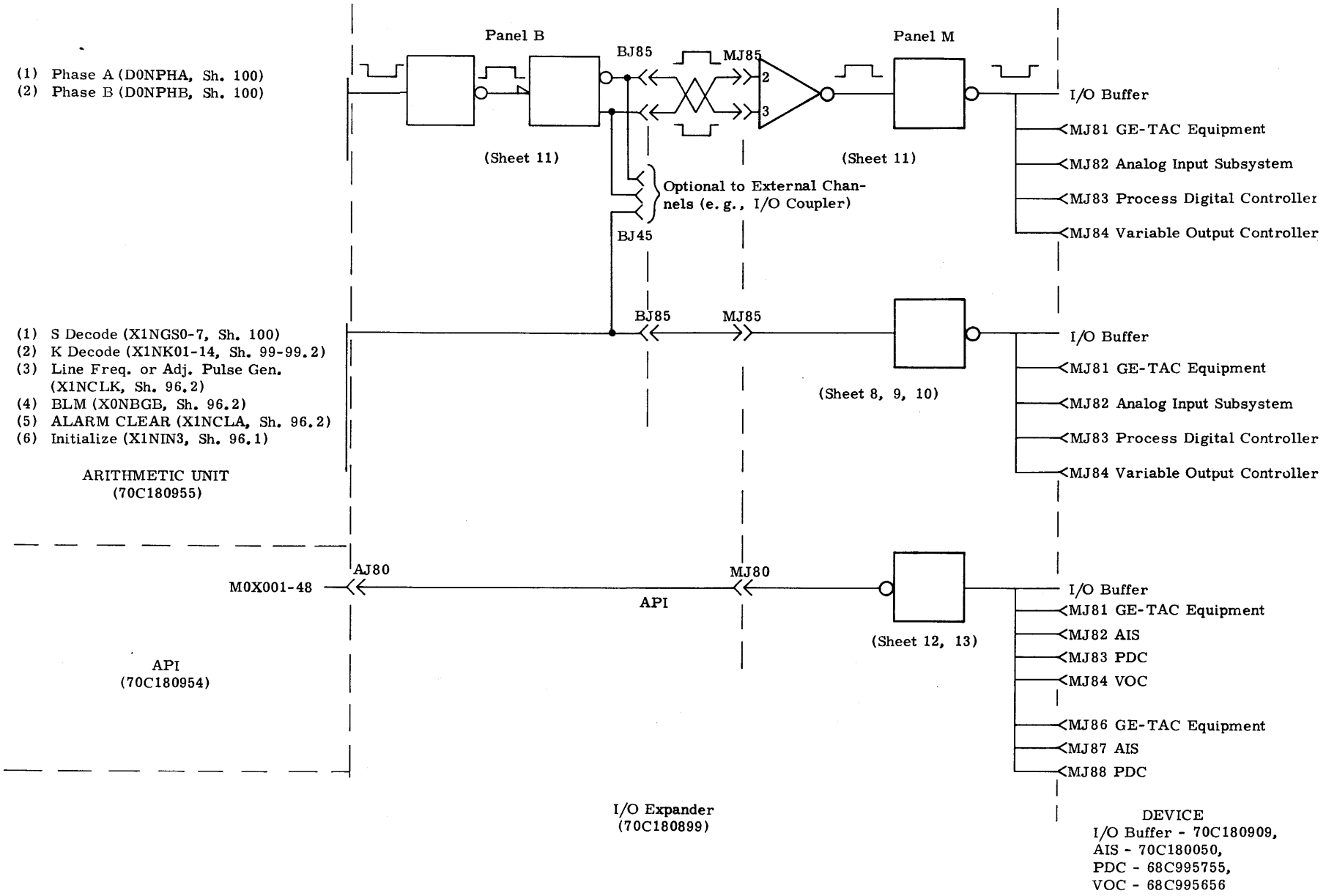
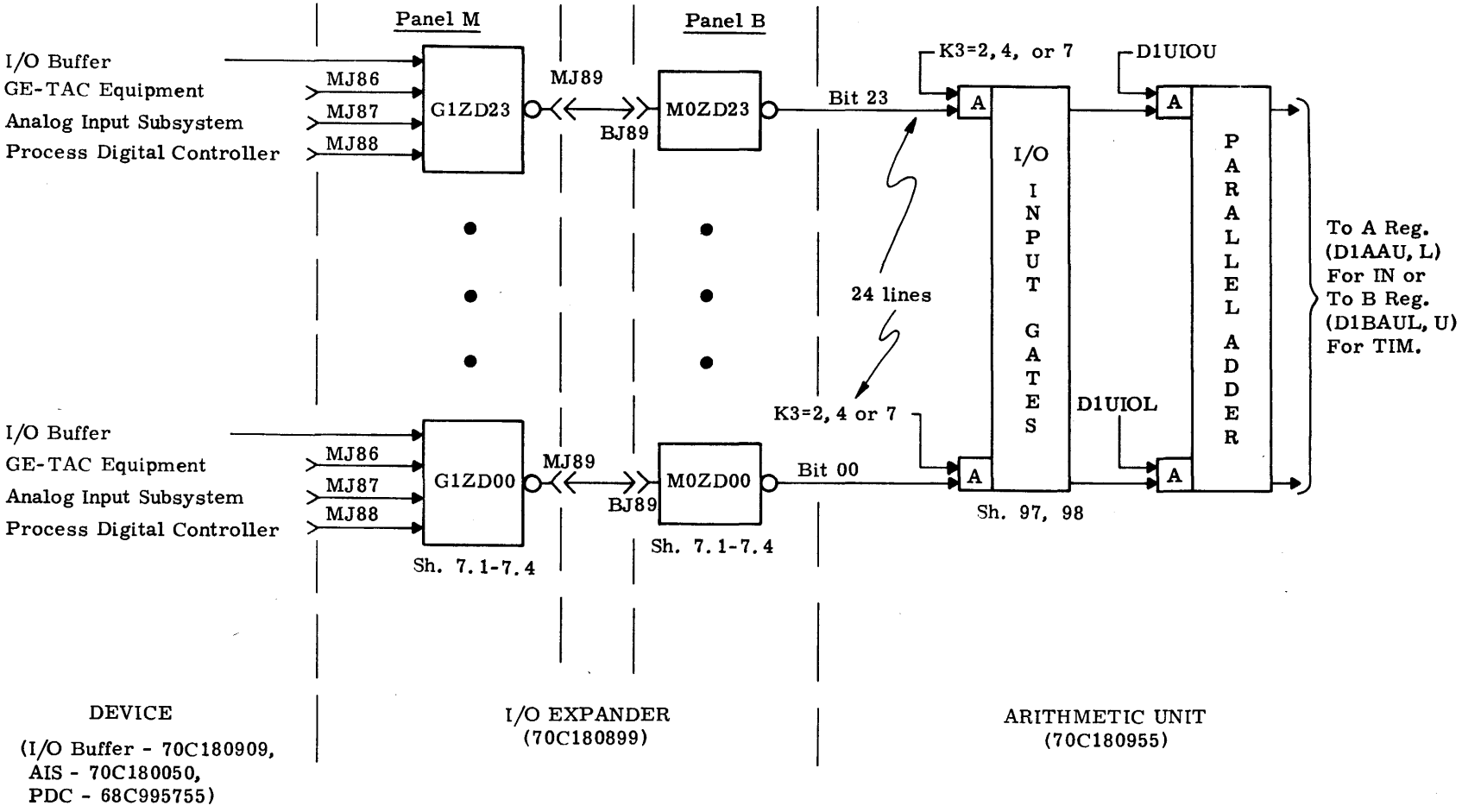


Fig. EXP. 3 Control Signals



NOTE: Data flow from optional external channel does not pass thru panel M. This data is routed directly to panel B (M0YD00-24) as shown on logic sheets 7.5 thru 7.8.

Fig. EXP. 4 IN/TIM Data Flow

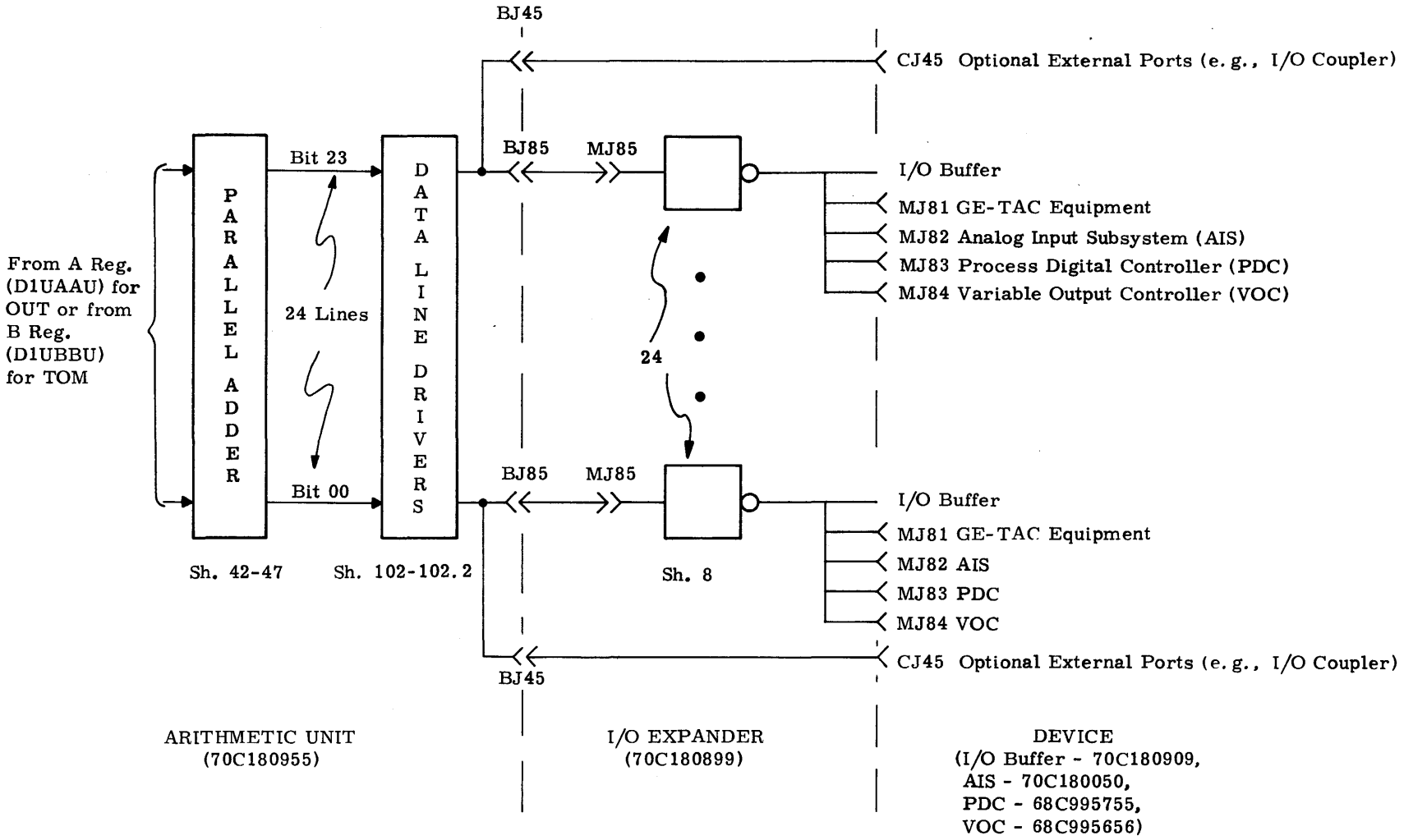
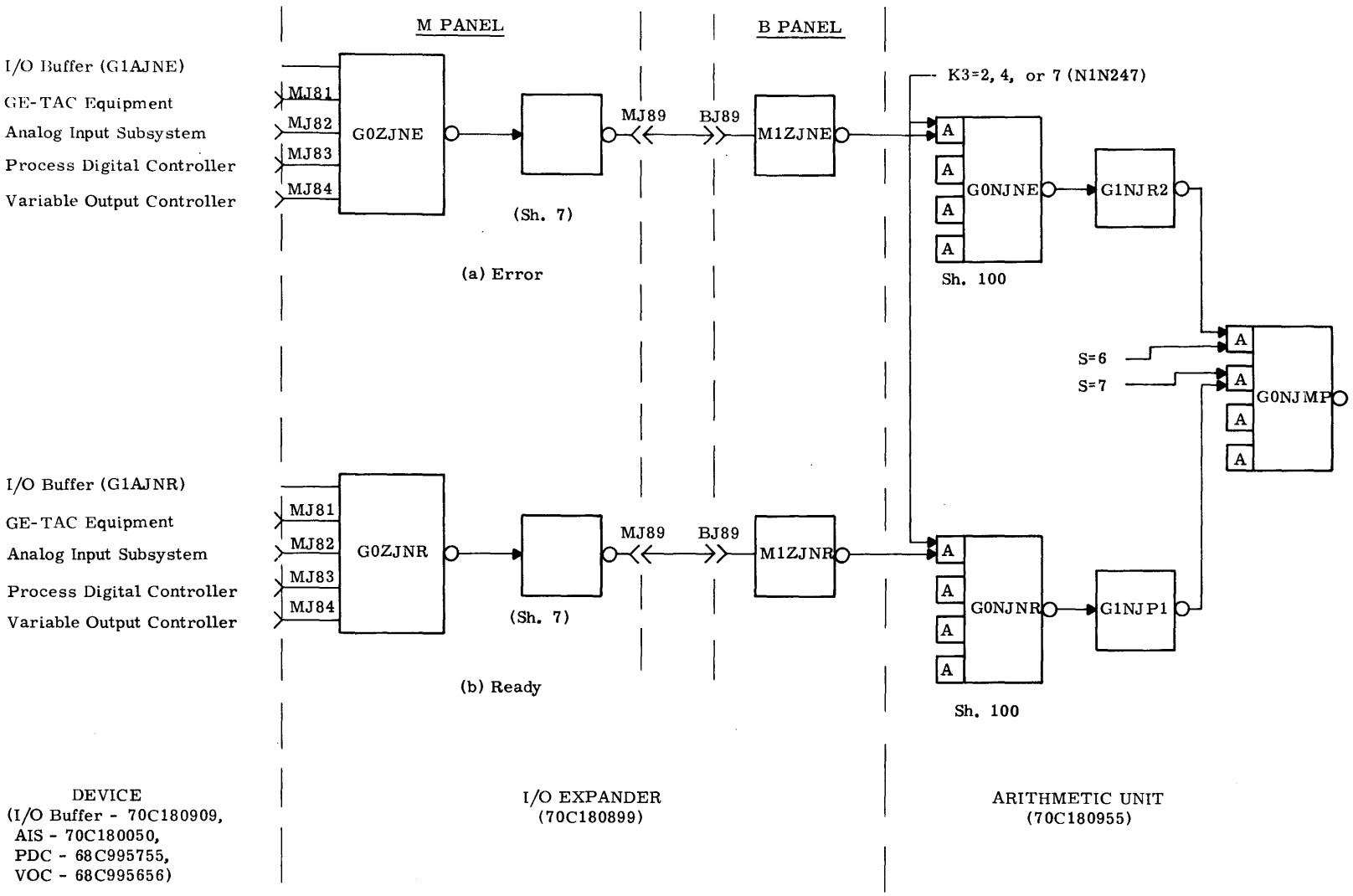
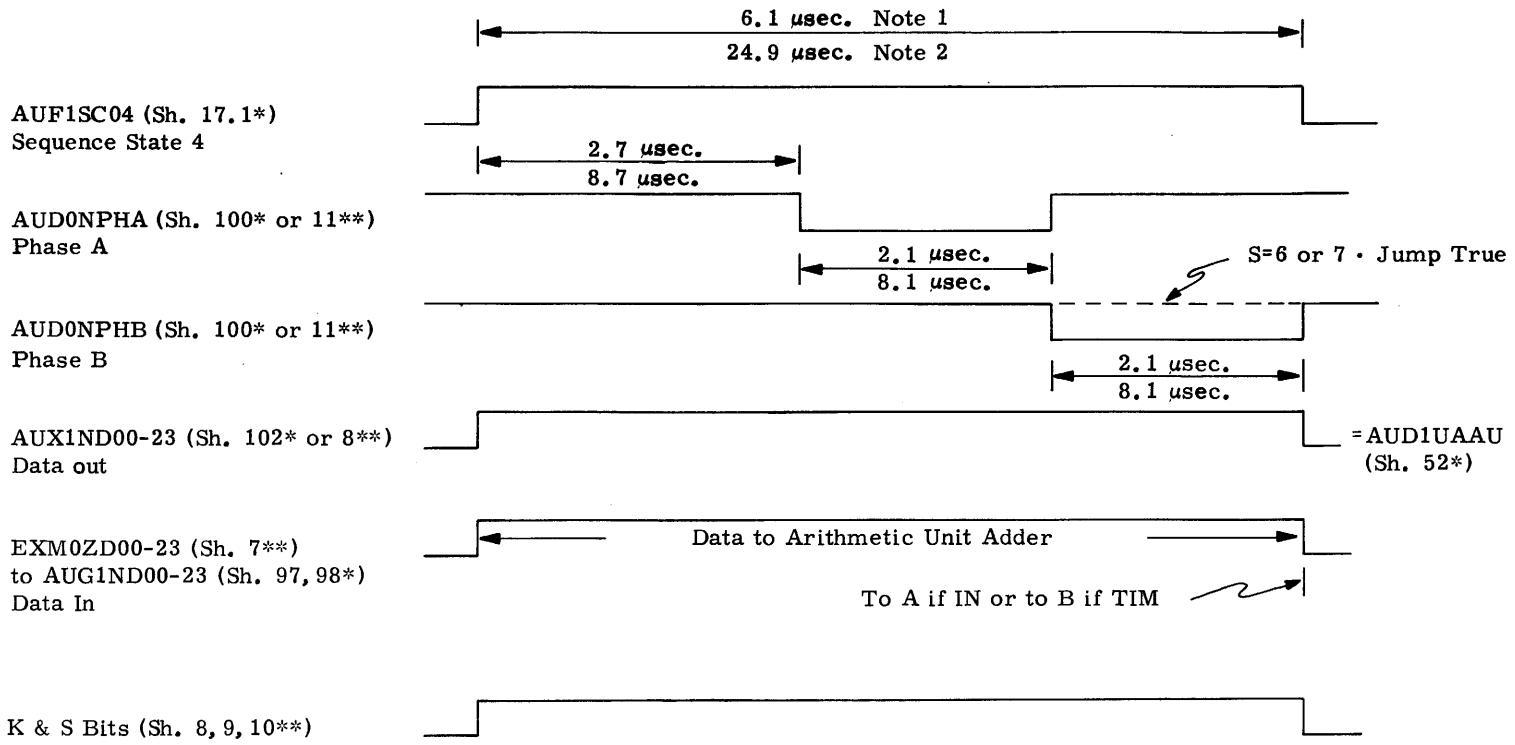


Fig. EXP. 5 OUT/TOM Data Flow



NOTE: Test lines from optional external channels do not pass thru panel M. These signals are routed directly to panel B (M1YJNE and M1YJNR) as shown on logic sheet 7.

Fig. EXP. 6 Test Lines



Notes:

1. High-Speed GEN II - K3 = 4 or 7 e.g., Peripherals, Analog Input Scanner, etc.
2. Low-Speed GEN II - K3 = 1 or 2 e.g., Process Digital Controller, Variable Output Controller, etc.

* AU logic, 70C180955

** I/O Expander logic, 70C180899

Fig. EXP. 7 Basic Timing

READER COMMENTS

The General Electric Company solicits your comments on publications covering Process Computer equipment. Please explain any "no" responses in the COMMENTS section. Your comments and suggestions become the property of the General Electric Company.

- Name of Manual: _____
- What is your computer application: _____

  _____
- How is this publication used:

  Familiarization ☐ Reference ☐

  Training ☐ Maintenance ☐

  Other (Explain) _____

	YES	NO
• Does this publication meet your requirements	☐	☐
• Is the material:		
1) Presented in clear text	☐	☐
2) Conveniently organized	☐	☐
3) Adequately detailed	☐	☐
4) Adequately illustrated	☐	☐
5) Presented at appropriate technical level	☐	☐

- Please provide specific text references (page number, line, etc.) with your comments.

NAME _____ DATE _____

TITLE _____

COMPANY NAME _____

AND ADDRESS _____

COMMENTS:

No postage necessary if mailed in the U. S. A.

Staple

Communications concerning Technical Publications should be directed to:

Manager, Technical Publications
Utility and Process Automation Systems Operation
2255 West Desert Cove Road
Phoenix, Arizona 85029

Fold Fold

FIRST CLASS
Permit No. 4091
Phoenix, Arizona

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

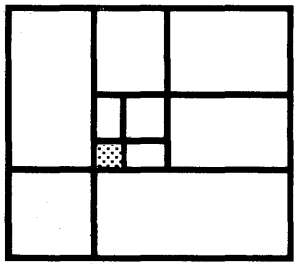
GENERAL ELECTRIC COMPANY
UTILITY and PROCESS AUTOMATION
SYSTEMS OPERATION
2255 West Desert Cove Road
Phoenix, Arizona 85029

Attention: Technical Publications

Fold Fold

Cut Along Line

Additional Comments:



GENERAL ELECTRIC