Daisy

IOP

Design Review

IOP Design Review - 4/13/84

IOP

* Based on Intel 80186 microprocessor

  8 MHz, enhanced 8086 family cpu

  Integrated DMA Controller, Interrupt Controller, Timers, Clock Generator, & Programmable Wait State and Chip Select Logic.

  Direct addressing capability to 1 MByte of memory and 64 KByte of I/O.

* Uses traditional up bus architecture

* Interrupt driven

* Normally - Ready System

* Can access both local memory & main memory

* Support external bus masters

* Can be extended through additional option slots

Local Memory

* 16 KBytes of EProm (0 wait state)
* 16 KBytes of SRAM (0 wait state)
* EProm contains booting & initialization code, Burdock kernel, & some diagnostic software
* SRAM is used for operating system software, interrupt vector table, & local buffering

Bus Masters

* Four Bus Masters
    1. IOP 80186
    2. PCE 80186
    3. RD-DMA Controller
    4. Ethernet Controller
* Bus arbitration is handled by special hardware

IOP Bus Bandwidth

* Data path is 16 bits wide (2 bytes, or 1 word)

* 8 MHz clock produces 125 ns T-states

* A minimum of 4 T-states (T1, T2, T3, T4) are required per bus cycle, i.e., a minimum bus cycle is 500 ns

* 80186 can potentially transfer data at 4 MBytes/s

* This potential is achieved on accesses to IOP's local RAM and EProm.

* However, most I/O is to/from main memory (thru A-Chip) which is slower and degraded by contention.

* Each wait state will degrade BW by about 20%

* From simulation studies, the average number of wait state for accessing main memory is 2.15 with 2 A-Chips and 2.98 with 1 A-Chip. [Ref = Daisy System Simulation & Performance Report by Abdo Kadifa, 5/20/84]

Integrated DMA Controller

* 2 DMA channels, each has a *request* input but no *acknowledge* output.

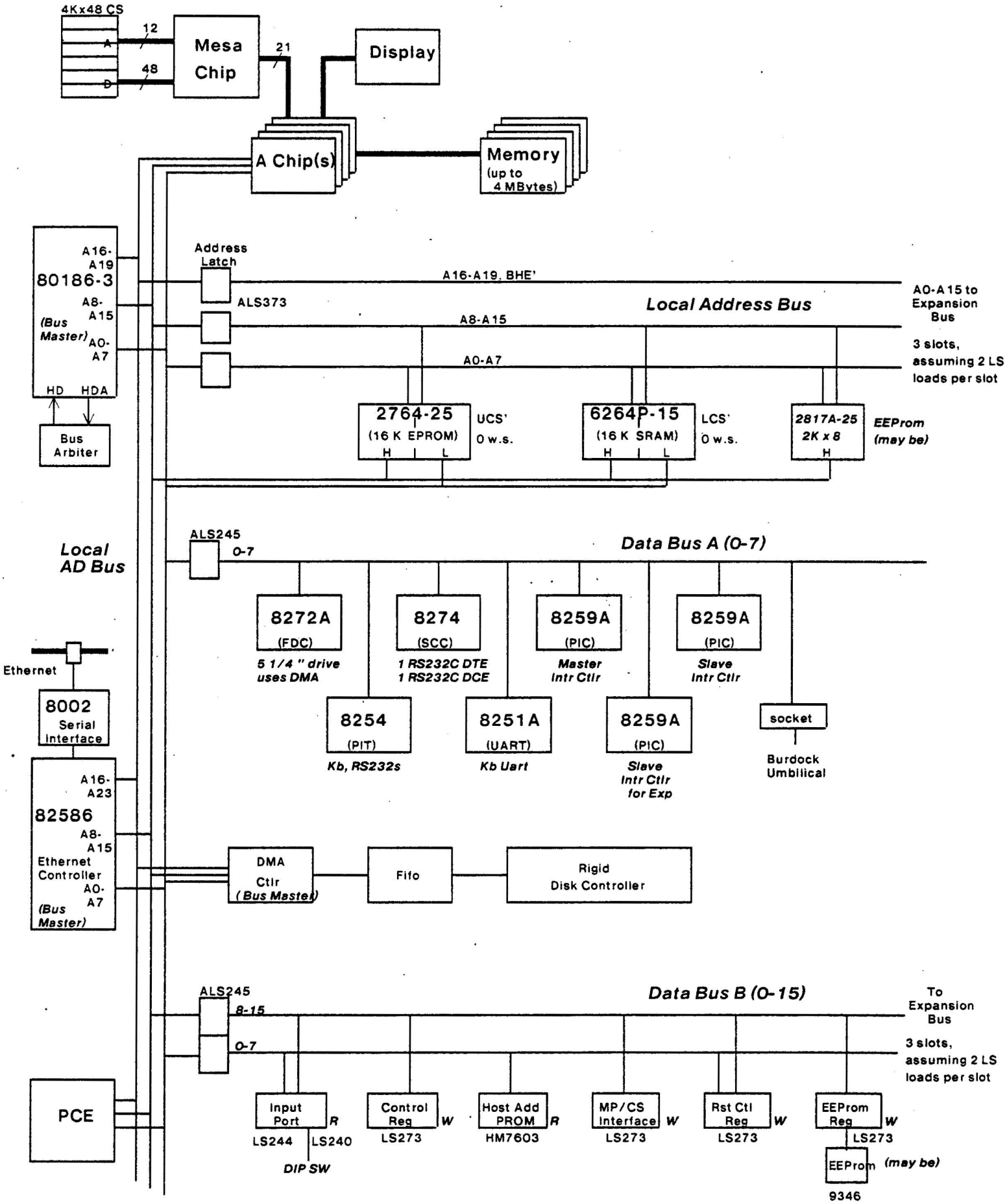
* Floppy Disk uses one channel

* Every integrated DMA transfer in 80186 consists of 2 independent bus cycles : a "fetch" cycle followed by a "deposit" cycle

  Fetch : Internal Temp Reg ← Source
  Deposit : Destination ← Internal Temp reg

* Potential transfer rate: 2 MBytes/s

* External HOLD has higher priority over an integrated DMA transfer



XEROX SDD	Project: Daisy	Daisy Block Diagram	File: DIOP40.silx	Designer: Tsang	Rev: F	Date: 6/01/84	Page: 40

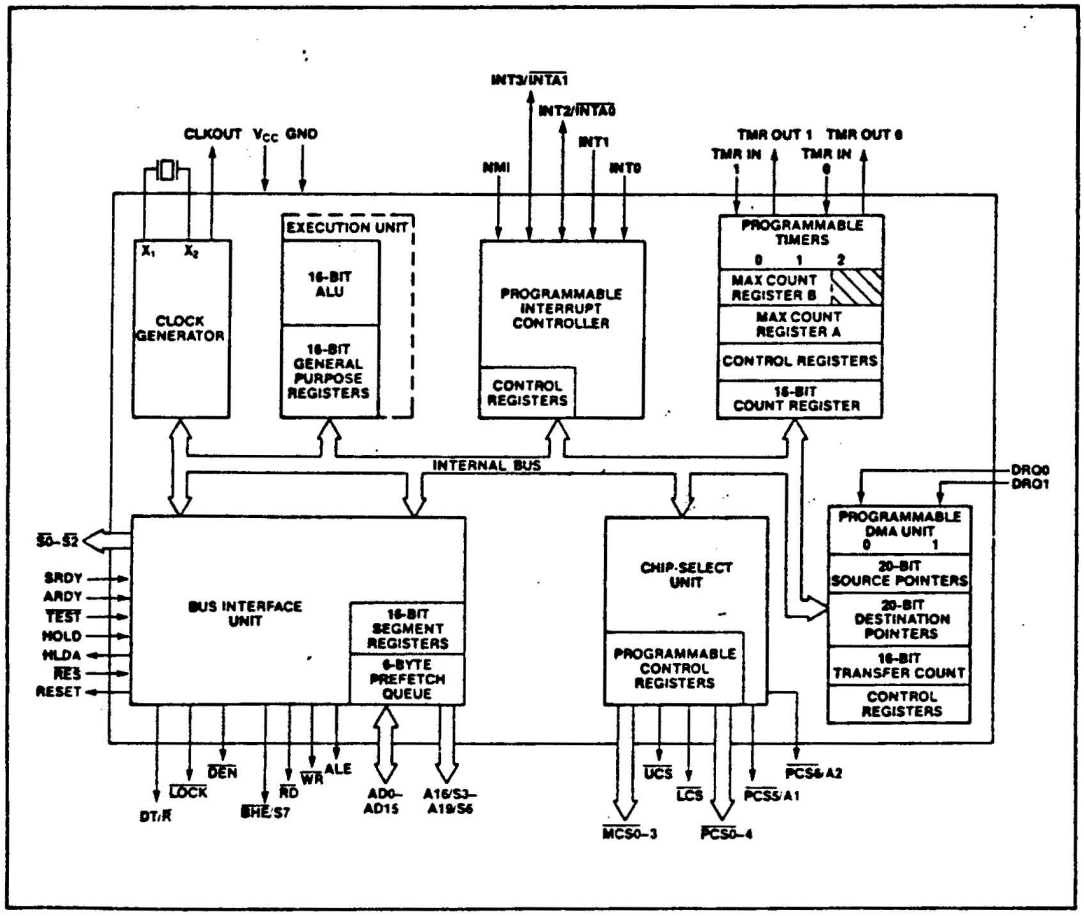
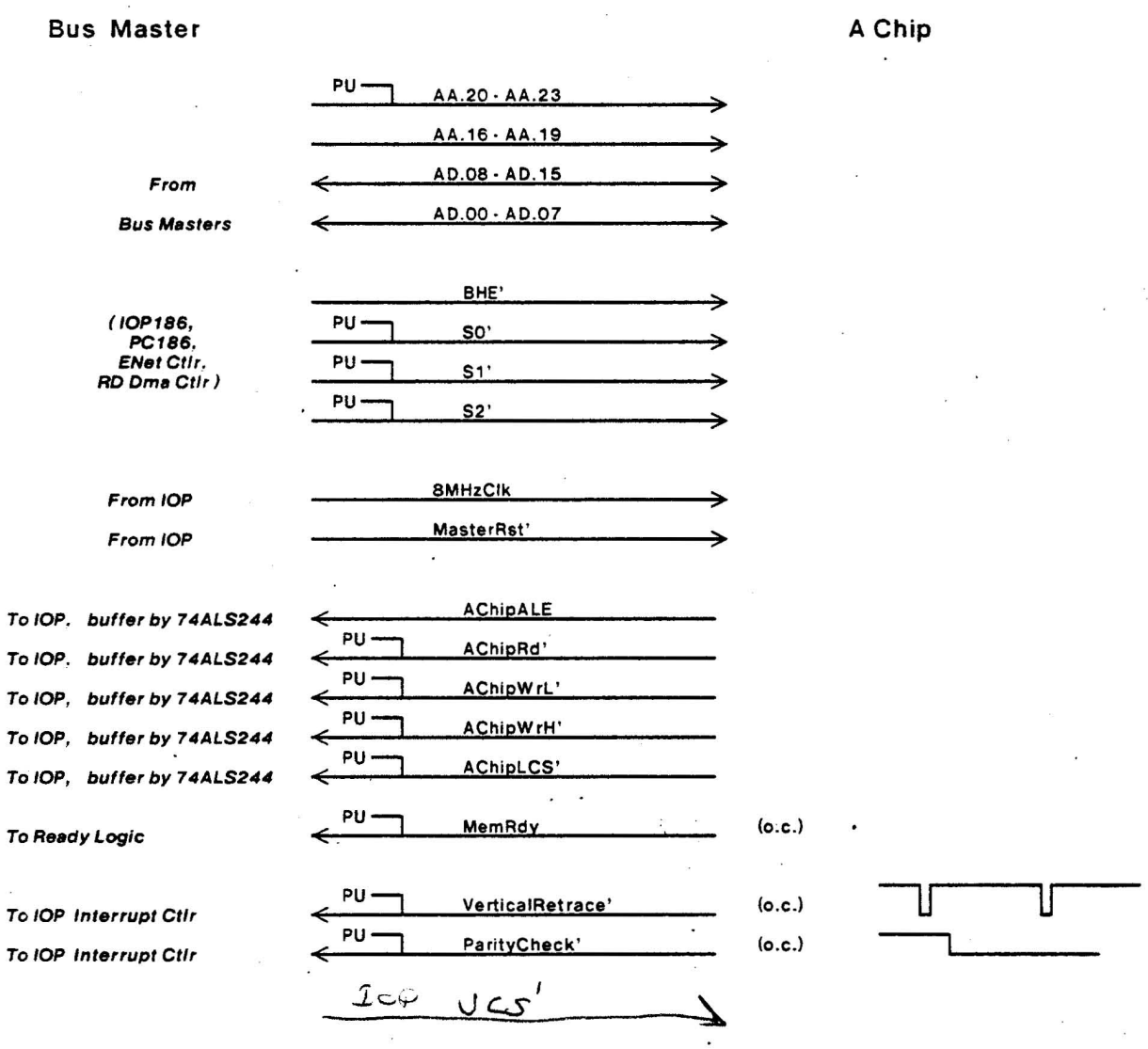


Figure 1. 80186 Block Diagram



PU = 10.0 K, pull-up on IOP board.

SelectA0', A1', A2' A3'
MapE'
MapF'

will be generated on Memory board(s).

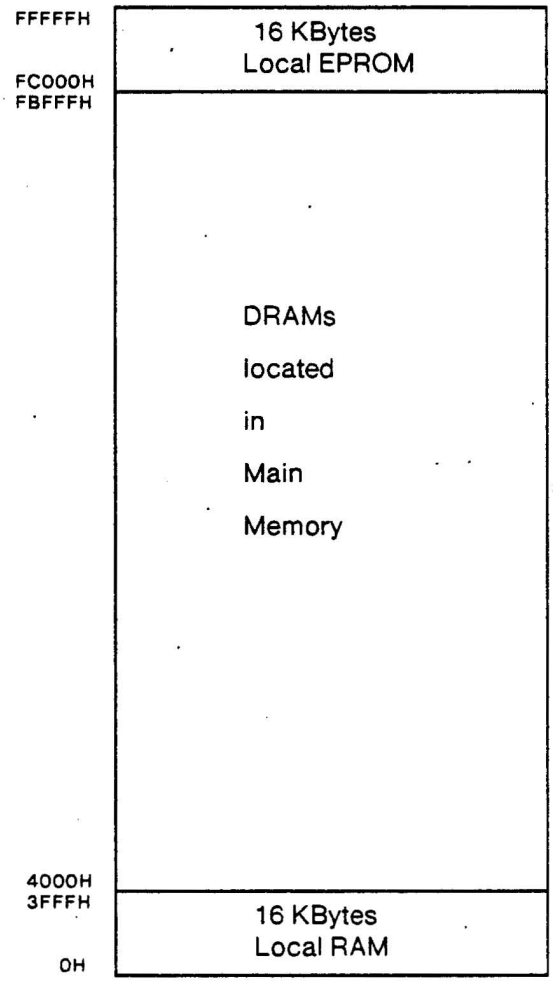
Why A chip is used to generate some of the 80186 control signals instead of using those available from 80186 ?

1. When the 80186 relinquish the bus to other bus master device(s), it floats the bus control signals and deactivates ALE, memory-select, and peripheral-select signals. So if an ext. bus master wants to address memory & peripheral devices, discrete chip select and ready generation logic must be used. But by using the A chip to generate these signals (ALE, LCS', ARdy), this logic can be saved.

2. When 80186 is set up to operate in "Queue Status" mode, the RD' line is ext. grounded, ALE = QS0, WR' = QS1. That is, they are not available from 80186 any more. Queue status mode is used for 8087 NDP extension.

IOP 80186 Memory Address Space

	ms																		ls
AA.23 - AA.20 = OFH	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
	1M	512K	256K	128K	64K	32K	16K	8K	4K	2K	1K	512	256	128	64	32	16	8	4	2



Function

0 wait state
UCS'

- StartUp ROM
- IOP BIOS

at least
1 wait state

0 wait state
AChipLCS'

- Local Memory
- Interrupt Vector Table
  (max 1K)

16 KBytes of SRAM (temp only)
uses MCS0', 0 w.s.
located at 10000H -- 13FFFH

Base address of mid-range memory must be a integer multiple of total block size.

XEROX SDD	Project: Daisy	80186 Memory Address Space	File: DIOP50.silx	Designer: Tsang	Rev: F	Date: 6/13/84	Page: 50

2.3 Address spaces

There are several types of address in Daisy, all of them used by the IOP at one time or another. The memory subsystem uses the *real address space*. On Daisy, this is 16Mb (24 bits), although typically, memory will only be provided for part of that. Application code in the Mesa machine uses the *Mesa virtual address space* (or just *virtual address space*). Virtual addresses are normally converted to real addresses by the Mesa microcode, but will occasionally be passed via IOCBs to the IOP, in which case the IOP has to perform the virtual-to-real translation. The *pagemap* to effect this translation lives at a well-known place in real memory. The *IO system address space* is 16Mb big, divided into 14Mb of real address space, 1Mb of *IOP logical address space* and 1Mb of *PC logical address space*. The logical address spaces are provided because of the 1Mb address limit of the 80186 processors; they are (partially) mapped into real memory by the A-chips. These allow the logical address spaces to be mapped in units of 128Kb to arbitrary 128Kb aligned blocks of real memory. For protection reasons, the mapping for both the PC and IOP logical address spaces will be under the control of the IOP.

High bandwidth devices (Ethernet and rigid disk) use the real address part of the IO system address space to access client data directly. Therefore, there is no need to copy data between device buffers and client buffers. In addition, because the IOP's logical address space is also available, the device controllers can pick up control information directly from the IOP as well, without the IOP having to convert the addresses of control blocks from logical to real addresses. Likewise the devices can read or write on behalf of the PC directly into the PC logical address space without the IOP having to convert those logical addresses to real.

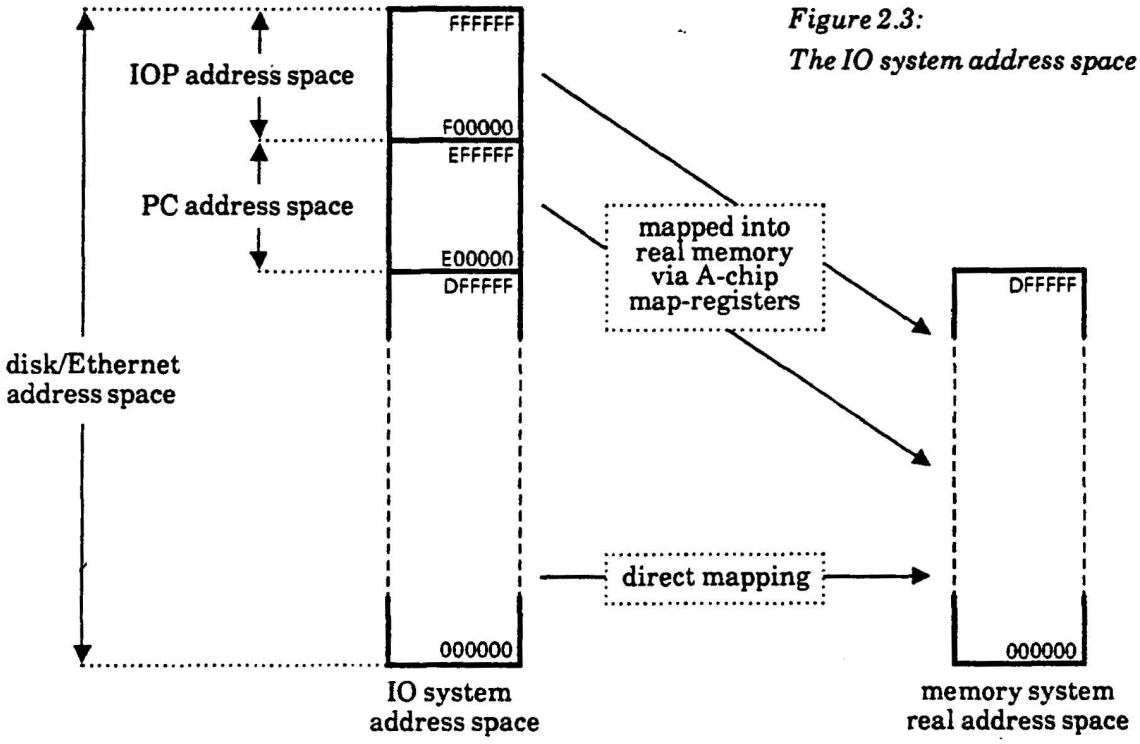


Figure 2.3:
The IO system address space

Note: although only 14Mb of real memory is shown, the A-chip map registers and the Mesa machine can access a full 16Mb. However, the top 2Mb cannot be directly addressed by the disk or Ethernet and so this part of real memory, if provided, would have to be treated specially - perhaps for the display bank or for resident code.

2.4 Outline of the A-chip map registers

The memory subsystem is controlled by a set of A-chips, each A-chip driving 1Mb of memory. Each A-chip has a set of 8 map registers which together cover the 1Mb logical address spaces. Each map register has two enables which indicate whether it should respond to the corresponding part of the IOP logical address space and/or to the corresponding part of the PC logical address space. If a map register is enabled, it can map that part of the appropriate logical address space to any 128Kb aligned block of the real memory it controls. With many A-chips in a system, care must be taken so that two A-chips are not both programmed to respond to a particular address.

Thus, by using the A-chip map registers appropriately, the IOP can access all of main memory (although only eight 128Kb blocks at a time). It also has to share the use of the map registers with the PC in a single A-chip system, whereas in a multi-A-chip system it might be convenient for different A-chips to respond to the same address for the two cases of a PC logical address and of a IOP logical address.

The mapping can also be disabled so that the disk and Ethernet can also supply real addresses to the A-chips. This is because the disk and Ethernet are capable of generating 24 bit addresses rather than the 20 of the IOP and PC 80186s.

2.5 Device controller types and address allocation

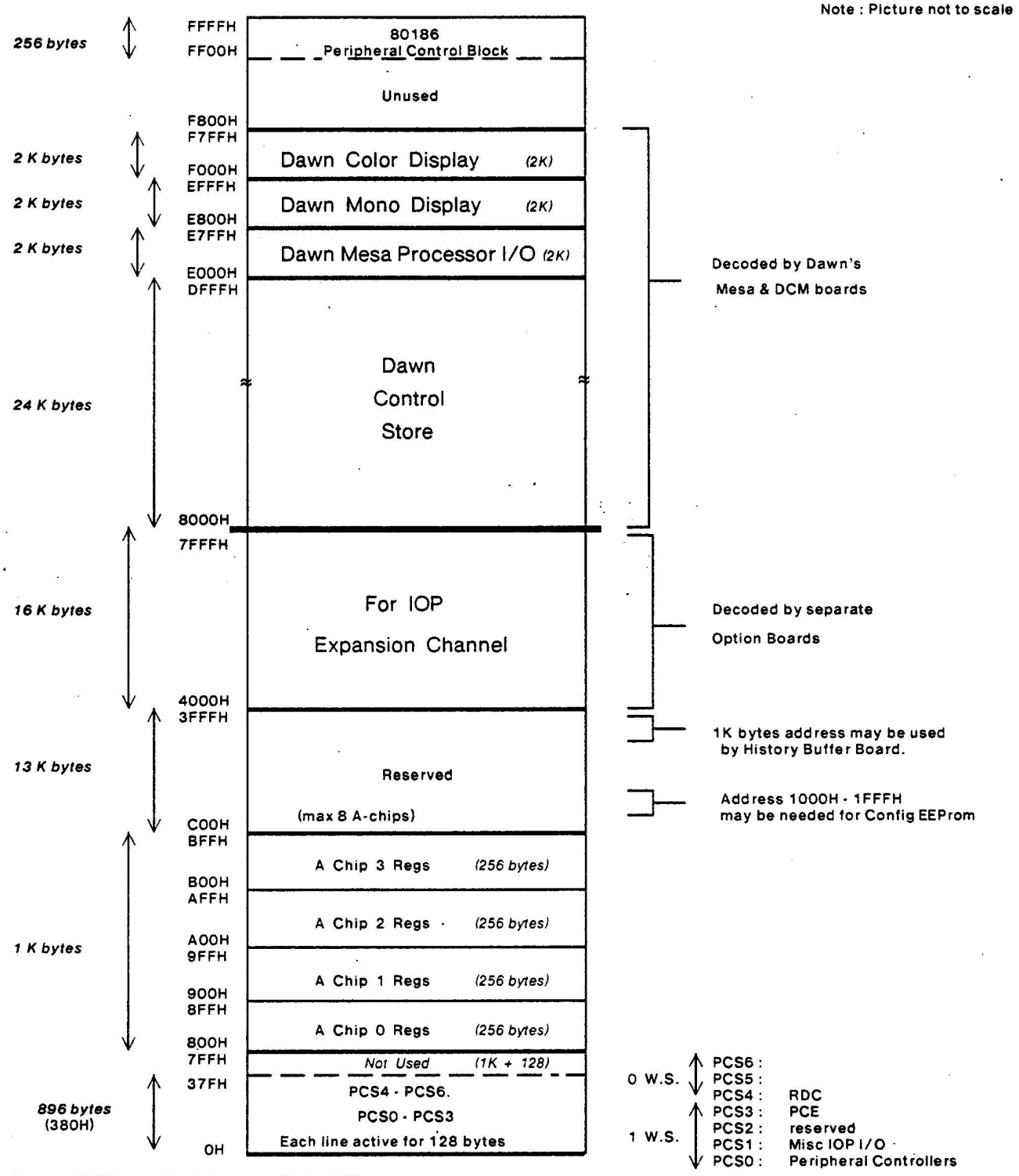
-- to be written

IOP 80186 I/O Address Space

				ms								ls							
A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
= 0				64K	32K	16K	8K	4K	2K	1K	512	256	128	64	32	16	8	4	2

64 KBytes of I/O Space

Note : Picture not to scale



Note : Base address of PCS's must be integer multiple of 1K.
Peripheral Control Block can be relocated to any 256 byte boundary.
0F8 to 0FF are reserved by Intel.
IBM PC I/O device addresses are located in the lowest 1K.

XEROX SDD	Project		File	Designer	Rev	Date	Page
	Daisy	80186 I/O Address Space	DIOP51.silx	Tsang	F	6/01/84	51

IOP I/O Controllers use the 80186 PCS lines.
Base address of PCS's starts at 0H in 80186 I/O space.
The seven PCS lines are allocated as follows :

		Address	Usages	
1 w.s.	PCS.0 :	0 - 7FH	I/O Controllers	
	PCS.1 :	80 - FFH	Misc IOP I/O	(Note: F8 to FFH reserved by Intel)
	PCS.2 :	100 - 17FH	PCE	
	PCS.3 :	180 - 1FFH	PCE	
0 w.s.	PCS.4 :	200 - 27FH	Rigid Disk	
	PCS.5 :	280 - 2FFH	unused	
	PCS.6 :	300 - 37FH	unused	

PCS.0

Address (Hex)	Read	Write
8259A Master Intr Ctlr		
0	IRR, ISR	ICW1, OCW2, OCW3
2	IMR	ICW2, ICW3, ICW4, OCW1
8259A Slave Intr Ctlr		
10	IRR, ISR	ICW1, OCW2, OCW3
12	IMR	ICW2, ICW3, ICW4, OCW1
8254 Timer		
20	Timer Counter 0	Timer Counter 0
22	Timer Counter 1	Timer Counter 1
24	Timer Counter 2	Timer Counter 2
26	—	Timer Mode
8251A Uart (Keyboard/Mouse)		
30	Uart Rx Data	Uart Tx Data
32	Uart Status	Uart Ctl Word
8274 RS232C Ctlr		
40	Ch A Rx Data	Ch A Tx Data
42	Ch A Status	Ch A Cmd/Parm
44	Ch B Rx Data	Ch B Tx Data
46	Ch B Status	Ch B Cmd/Parm
8272A FDC		
50	FDC Main Status Reg	*(illegal)*
52	FDC Data Reg Stack	FDC Data Reg Stack
54	Dma Read from FDC	Dma Write to FDC
56	Terminal Count to FDC	*(for polling only)*
8259A Expansion Intr Ctlr		
60	IRR, ISR	ICW1, OCW2, OCW3
62	IMR	ICW2, ICW3, ICW4, OCW1
8255A-5 PIO Burdock Interface		
70	Port A	Port A
72	Port B	Port B
74	Port C	Port C
76	*(illegal)*	Control Word

PCS.1

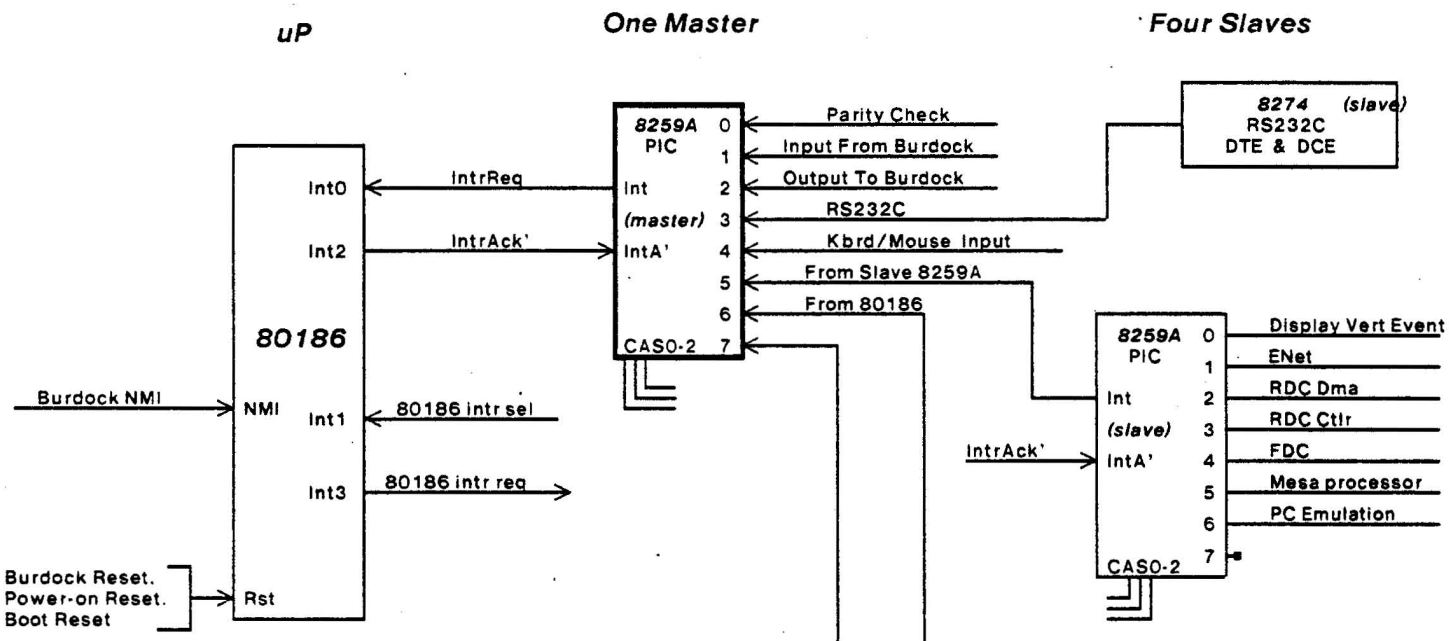
Address (Hex)	Read	Write
80	Input Port	Control Reg
90	Host Address PROM (90.92.94.96.98.9A.9C.9EH)	LED
A0	Clear Ring Latch	ENet Attn
B0	Clear Mesa Interrupt Latch	Mesa Processor/ Control Store Reg
C0	Clear ENet Interrupt Latch	***Reserved (may be used for WrConfigReg')***
D0	X	
E0	X	AllowPCCmd'
F0	HoldIOPCmd'	AllowRDCCmd'

Note : address 0F8H -- 0FFH are reserved by Intel.

XEROX SDD	Project Daisy	IOP I/O Controller Addresses	File DIOP53.silx	Designer Tsang	Rev E	Date 5/22/84	Page 53

	READ			WRITE				
	Input Port *(80H)*			Control Reg *(80H)*	LED *(90H)*	Mesa Proc/ Control Store Reg *(B0H)*	Reset Control Reg *(C0H)*	EEProm Config Reg *(D0H) (may be)*
Bit 15	8272A FDC Interrupt Request			Speaker Data	LED digit 3	Control Store Wr Enable	Enable Ethernet Int Loopback	Serial EEProm Enable
Bit 14	8272A FDC Dma Request			Enable 186 Timer 0	LED digit 3	Control Store Load/Shift'	Enable Keyboard Int Loopback	*unused*
Bit 13	Data From Control Store (1 bit only)			FDD Motor On	LED digit 3	Control Store Buffer En	Allow to Access Byte-wide EEProm	*unused*
Bit 12	Daybreak/ Daisy' ID			FDD In Use	LED digit 3	Data To Control Store (1 bit only)	*unused*	Serial EEProm Data In
Bit 11	Data From Config EEProm (1 bit only) *(may be)*			Allow Timer 1 to generate 8272A TC in real Dma	LED digit 2	Control Store Shift Clock	*unused*	*unused*
Bit 10	RS232 Ch A DSR' *(L active)*			*unused*	LED digit 2	*unused*	*unused*	*unused*
Bit 9	RS232 Ch A Ring Indicator' *(L active)*			Sel RS232 Ch A to use internal clock	LED digit 2	Halt Mesa Proc' *(L active)*	Reset Dma-Fifo'	*unused*
Bit 8	RS232 Ch B DTR' *(L active)*			Enable RS232 Ch B to send clock signals	LED digit 2	Interrupt Mesa Proc	Reset PCE-80186'	Serial EEProm Clock
Bit 7	SpkerTimer				LED digit 1		Reset RDC'	
Bit 6	Serial EEPromRdy				LED digit 1		Reset Mesa Proc'	
Bit 5	DipSw.5				LED digit 1		Reset Keyboard'	
Bit 4	DipSw.4				LED digit 1		Reset Burdock Ctlr 8255-5'	
Bit 3	DipSw.3				LED digit 0		Reset Keyboard Uart 8251A'	
Bit 2	DipSw.2				LED digit 0		Reset FD Ctlr 8272A'	
Bit 1	DipSw.1				LED digit 0		Reset RS232 Ctlr 8274'	
Bit 0	DipSw.0				LED digit 0		Reset ENet Ctlr 82586'	

Interrupts



80186 operates in

- iRMX mode
- as slave to ext intr ctlr
- internal vectoring

Master PIC 8259A operates in

- non-buffered mode
- cascade mode
- special fully nested mode
- all inputs are edge triggered

Slave PIC 8259A operates in

- non-buffered mode
- cascade mode
- all inputs are edge triggered

Vector Types

- max 256 allowed (limited by 8 bits vector), including predefined and reserved ones.
- 0 - 19 pre-defined for 80186.
- 20 - 31 reserved by Intel.

IOP interrupt types should start at 32.

Advantage of RMX mode: Interrupt vectors for 80186 internal peripherals are programmable. (The 80186 Interrupt Vector Register specifies the 5 m.s. bits, the lower 3 sig. bits are determined by the priority level of the internal device causing the interrupt in iRMX mode).

Disadvantages: Take 55 c.c. to process an interrupt (internal or external vectoring).

XEROX SDD	Project: Daisy	IOP Interrupts	File: DIOP52.silx	Designer: Tsang	Rev: E	Date: 5/22/84	Page: 52x

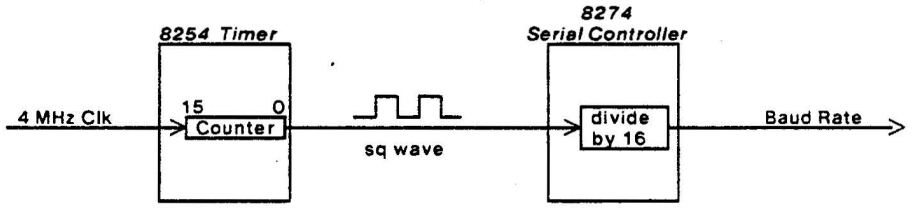
Timers

80186 : *(internal timers, 16 bits wide)*

- Timer0 : Speaker Circuit (Clock-In = Internal 2 MHz clock)
- Timer1 : FDC Terminal Count (Clock-In = 8272A FDC Dma Ack)
- Timer2 : *Reserved for Operating System* (Clock-In = Internal 2 MHz clock, no ext output)

8254 : *(8 bits wide)*

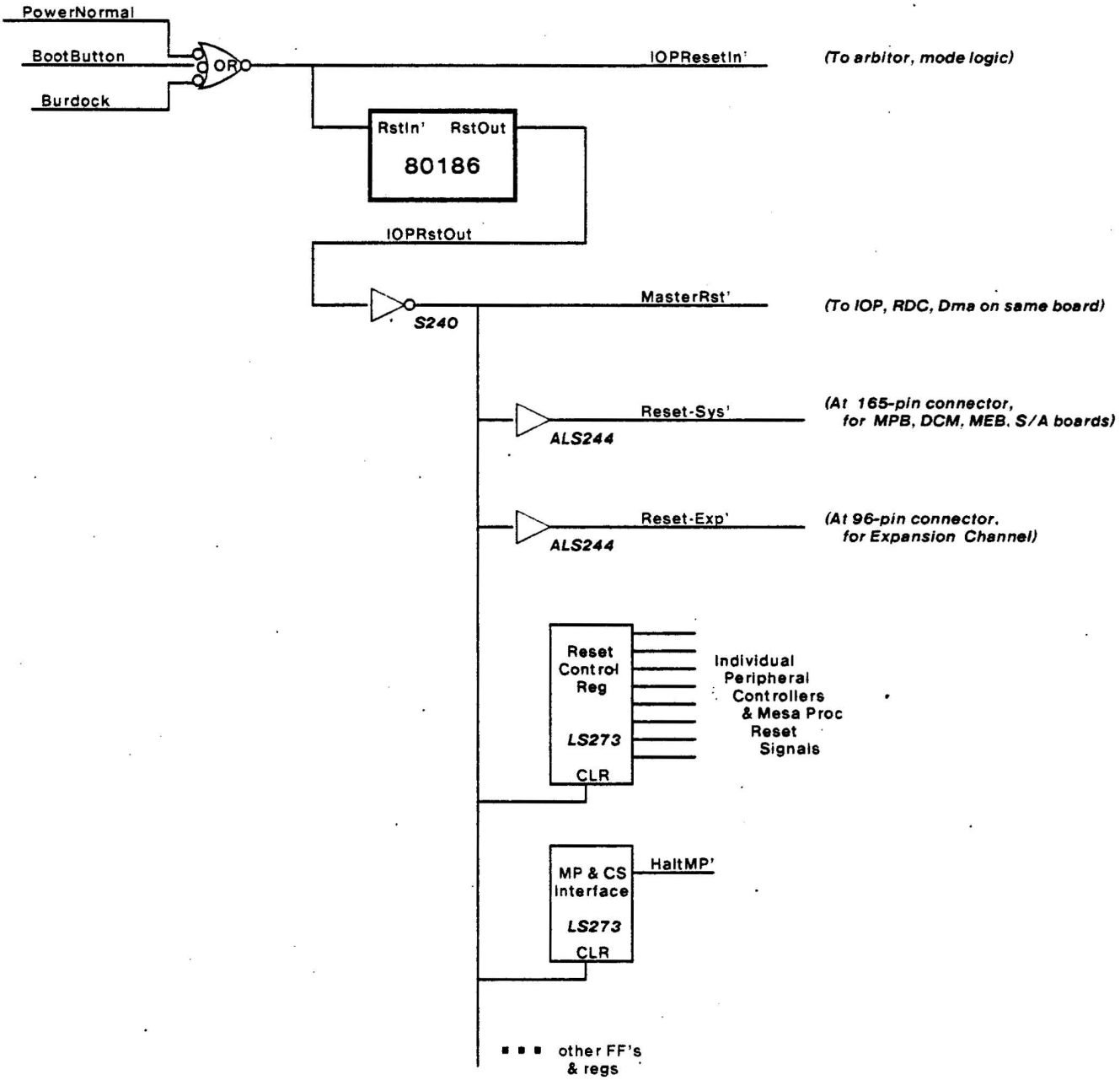
- Timer0 : RS232C DTE baud rate (clock in = 4.0 MHz)
- Timer1 : RS232C DCE baud rate (clock in = 4.0 MHz)
- Timer2 : Keyboard / Mouse baud rate (clock in = 4.0 MHz)



Baud Rate	Time Const	Error
19200	13	0.16 %
9600	26	0.16 %
7200	35	0.79 %
4800	52	0.16 %
3600	69	0.64 %
2400	104	0.16 %
2000	125	0 %
1800	139	0.08 %
1200	208	0.16 %
600	417	0.08 %
300	833	0.04 %
150	1667	0.02 %
134.5	1859	0.01 %
110	2272	0.03 %
75	3333	0.01 %
50	5000	0 %

$$\text{Time Constant} = \frac{4 \times 10^6}{(\text{baud rate}) \cdot (16)}$$

$$\%\ \text{Error} = \text{abs}\left[\frac{4 \times 10^6}{(\text{time const}) \cdot (\text{baud rate}) \cdot (16)} - 1\right] \times 100\%$$



- PowerNormal becomes H btwn 50 - 250 ms after all power supply outputs have exceeded 94% of nominal.
- Pressing the Boot Button will generate a L pulse of ~ 0.5 s.
- Reset signal from Burdock must be at least 500 ns wide (80186 requirement).
- Software resets of peripheral controllers should be at least 20 us long (due to keyboard reset requirement).
- When floppy disk controller is reset or during machine power up, the Write Current is cut-off.
- When rigid disk controller is reset or during machine power up, the Write Current is cut-off.

XEROX SDD	Project Daisy	Reset Logic	File DIOP494.silx	Designer Tsang	Rev F	Date 6/12/84	Page 494

Initialization and Processor Reset

Processor initialization or startup is accomplished by driving the $\overline{RES}$ input pin LOW. $\overline{RES}$ forces the 80186 to terminate all execution and local bus activity. No instruction or bus activity will occur as long as $\overline{RES}$ is active. After $\overline{RES}$ becomes inactive and an internal processing interval elapses, the 80186 begins execution with the instruction at physical location FFFF0(H). $\overline{RES}$ also sets some registers to predefined values as shown in Table 5.

Table 5. 80186 Initial Register State after RESET

Status Word	F002(H)
Instruction Pointer	0000(H)
Code Segment	FFFF(H)
Data Segment	0000(H)
Extra Segment	0000(H)
Stack Segment	0000(H)
Relocation Register	20FF(H)
UMCS	FFFB(H)

* Fetch 1st inst at CS:IP = FFFF:0
* Reloc Reg should = 20FF for
  - RMX Mode
  - Peripheral Control Block to start at 20FF00H in I/O space.

Local Bus Controller and Reset

Upon receipt of a RESET pulse from the $\overline{RES}$ input, the local bus controller will perform the following actions:

- Drive $\overline{DEN}$, $\overline{RD}$, and $\overline{WR}$ HIGH for one clock cycle, then float.

  NOTE: $\overline{RD}$ is also provided with an internal pull-up device to prevent the processor from inadvertently entering Queue Status mode during reset.

- Drive $\overline{S0}$-$\overline{S2}$ to the passive state (all HIGH) and then float.
- Drive $\overline{LOCK}$ HIGH and then float.
- Tristate AD0-15, A16-19, $\overline{BHE}$, DT/$\overline{R}$.
- Drive ALE LOW (ALE is never floated).
- Drive HLDA LOW. (Note 2)

Note 2: If HOLD = H after $\overline{RESET}$ = H, HLDA will go H 2 c.c. later.

Chip Select/Ready Logic and Reset

Upon reset, the Chip-Select/Ready Logic will perform the following actions:

- All chip-select outputs will be driven HIGH.
- Upon leaving RESET, the $\overline{UCS}$ line will be programmed to provide chip selects to a 1K block with the accompanying READY control bits set at 011 to allow the maximum number of internal wait states in conjunction with external Ready consideration (i.e., UMCS resets to FFFBH).
- No other chip select or READY control registers have any predefined values after RESET. They will not become active until the CPU accesses their control registers. Both the PACS and MPCS registers must be accessed before the $\overline{PCS}$ lines will become active.

DMA Channels and Reset

Upon RESET, the DMA channels will perform the following actions:

- The Start/Stop bit for each channel will be reset to STOP.
- Any transfer in progress is aborted.

Interrupt Controller and Reset

Upon RESET, the interrupt controller will perform the following actions:

- All SFNM bits reset to 0, implying Fully Nested Mode.
- All PR bits in the various control registers set to 1. This places all sources at lowest priority (level 111).
- All LTM bits reset to 0, resulting in edge-sense mode.
- All Interrupt Service bits reset to 0.
- All Interrupt Request bits reset to 0.
- All MSK (Interrupt Mask) bits set to 1 (mask).
- All C (Cascade) bits reset to 0 (non-cascade).
- All PRM (Priority Mask) bits set to 1, implying no levels masked.
- Initialized to non-iRMX 86 mode.

Timers and Reset

Upon RESET, the Timers will perform the following actions:

- All EN (Enable) bits are reset preventing timer counting.
- All SEL (Select) bits are reset to zero. This selects MAX COUNT register A, resulting in the Timer Out pins going HIGH upon RESET.

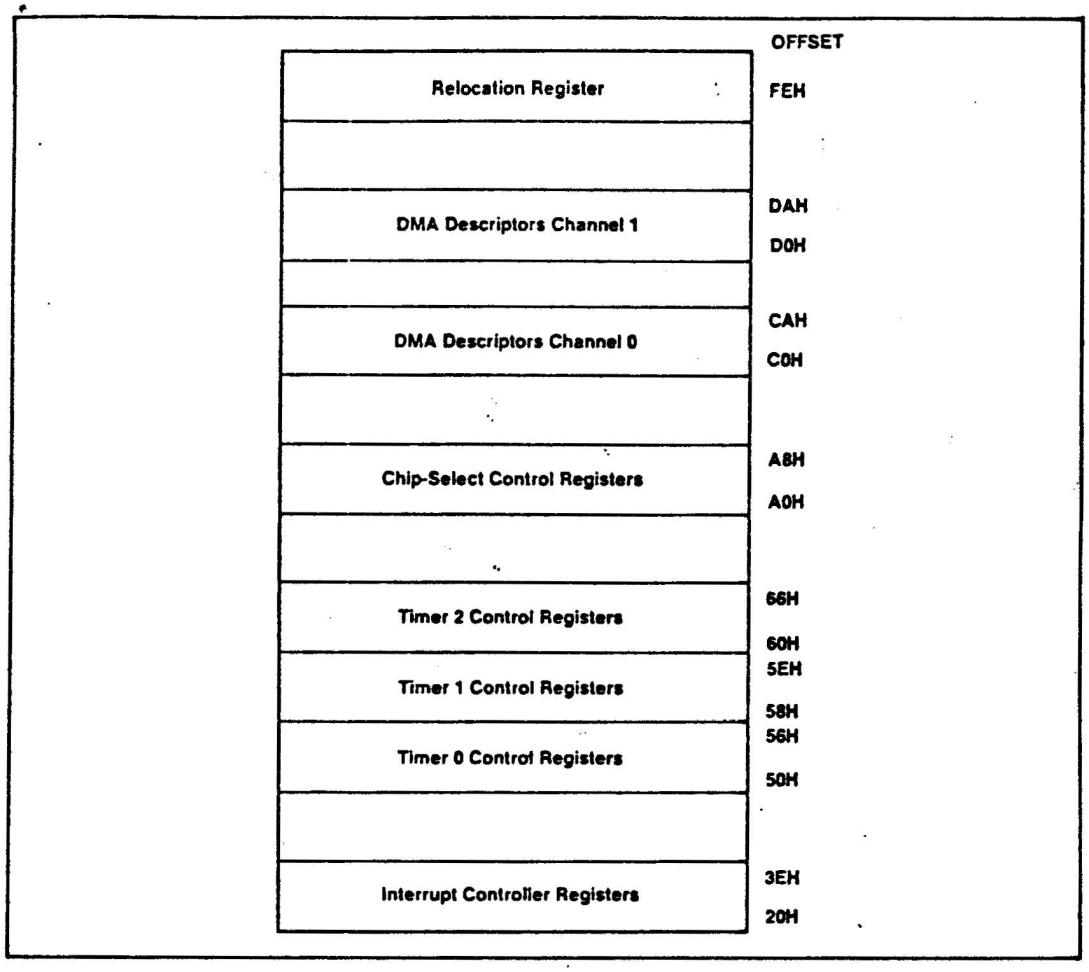


Figure A-1. 80186 Integrated Peripheral Control Block

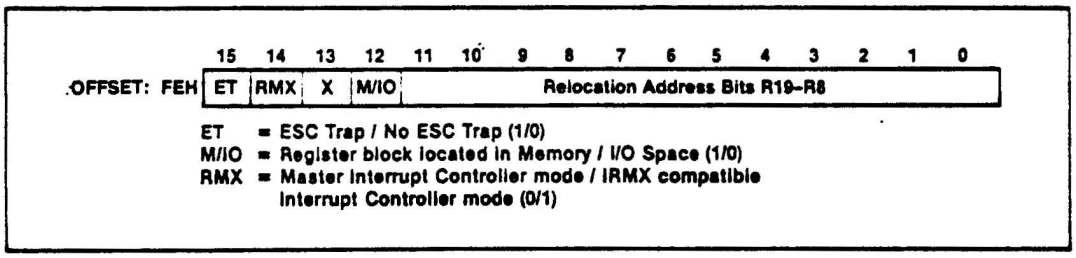


Figure A-2. 80186 Relocation Register Layout

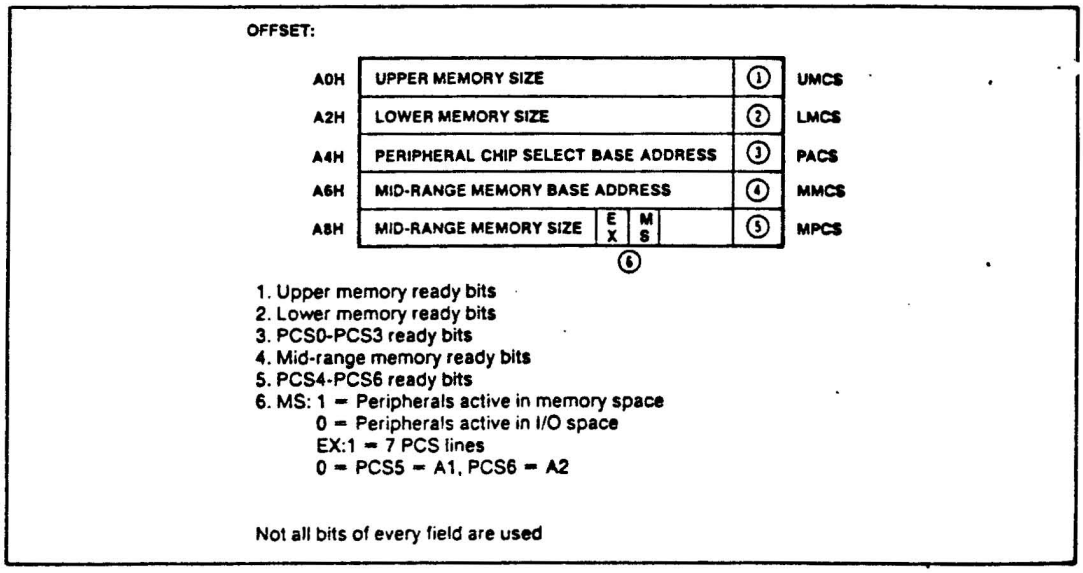
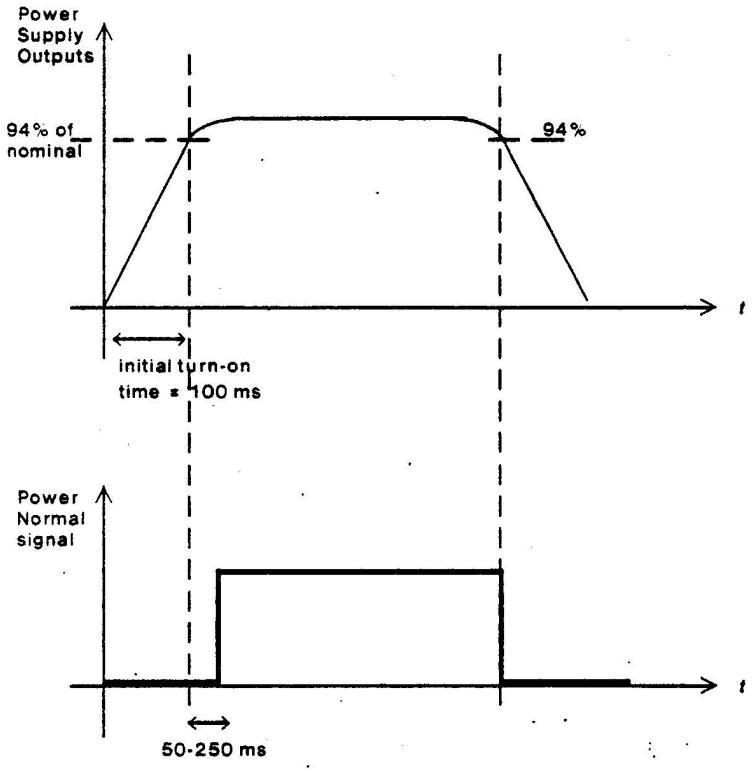


Figure 72. 80186 Chip Select Control Registers

Definition of Power Normal :



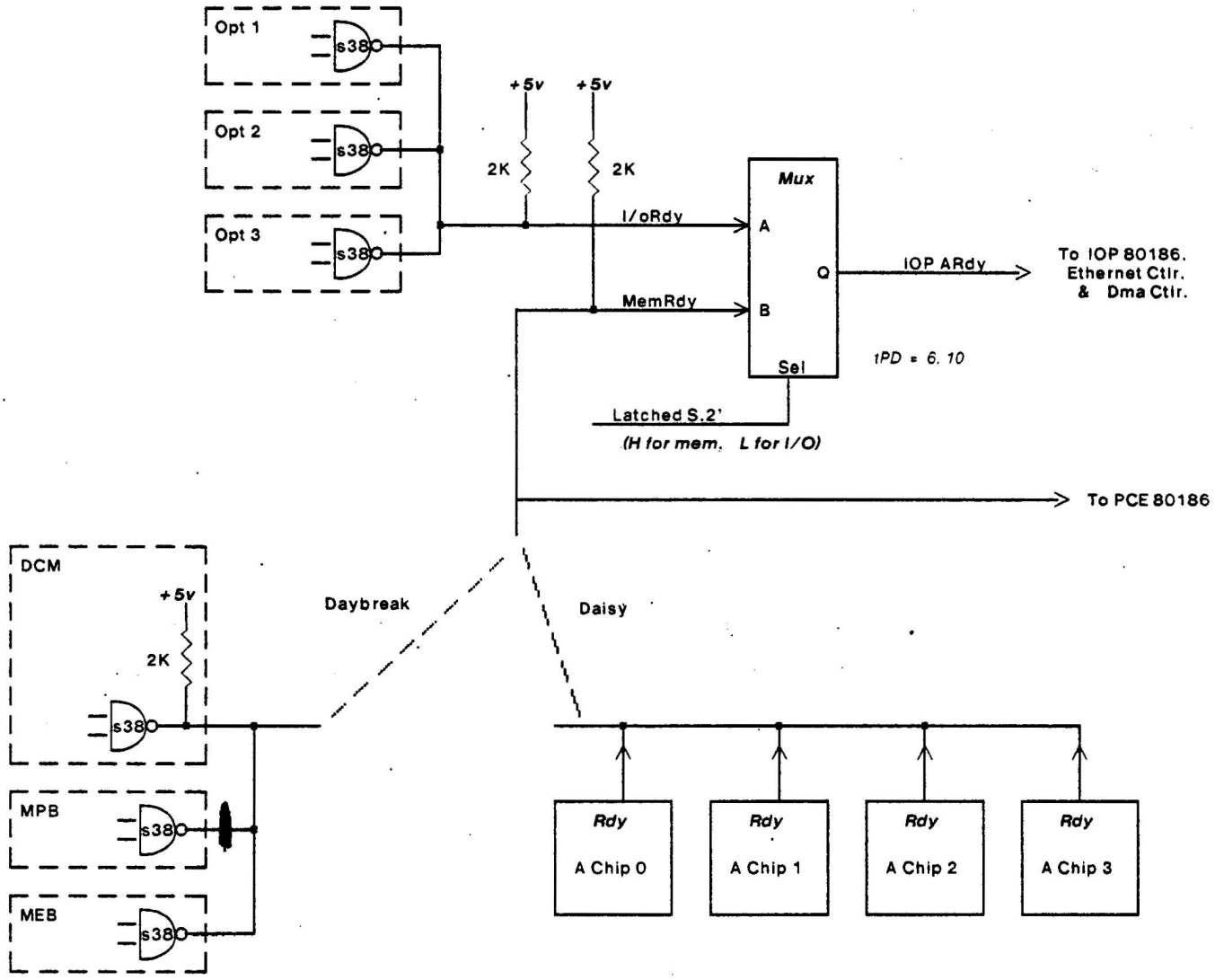
Outputs Nominal Voltages	*Voltage Regulation (at PS)*
+ 5.1 v	± 1 %
+ 12.0 v	± 3 %
- 12.0 v	± 3 %
- 5.2 v	± 3 % ???

From the power supply spec :

A PowerNormal shall be provided to indicate that ALL outputs are "UP AND USABLE" after the turn-on period, and also to indicate imminent loss of output during turn-off or AC input power failure.

1. During AC input turn-on, PowerNormal shall become logically H between 50 and 250 ms after all outputs have exceeded 94% of nominal.
2. During AC input turn-off or loss of AC input power, PowerNormal shall go logically L before any output decreases to 94% of nominal.
3. During momentary loss of input power for any duration, or for repeated AC input ON-OFF cycles, PowerNormal shall go L before any outputs decreases to 94% of nominal, and if PowerNormal goes L, it shall not go H until 50 to 250 ms after all outputs are above 94% of nominal.

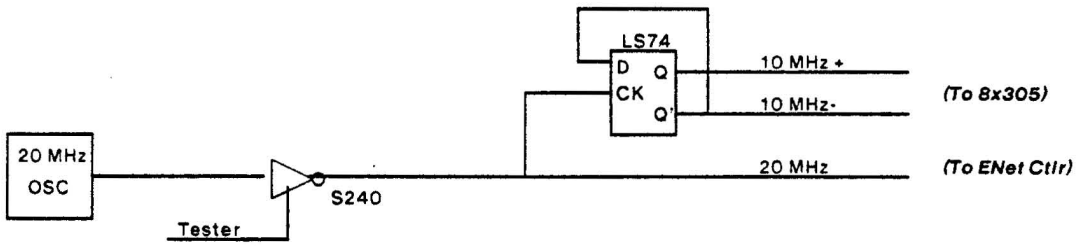
IOP 80186 Ready Logic :

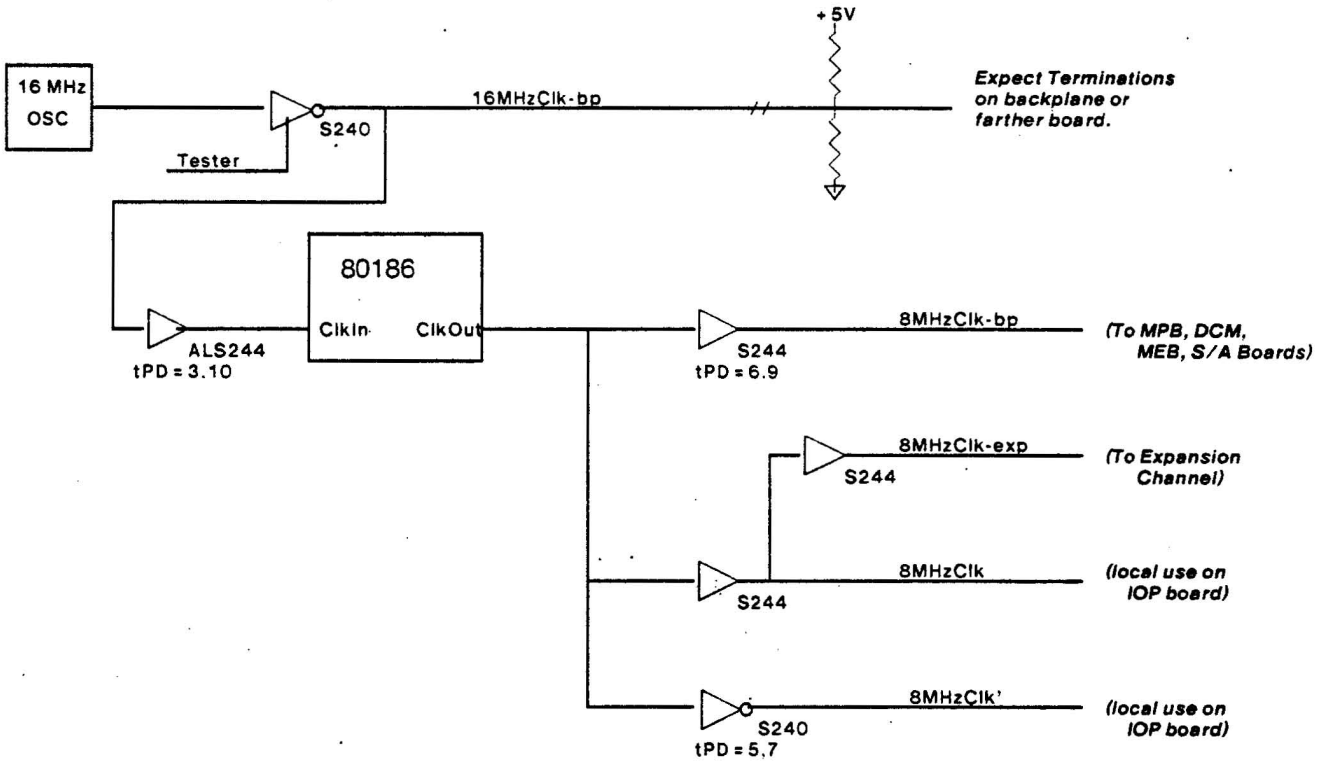


Note :

- IOP is a NORMALLY-READY system.
- Only the ARdy of 80186 is used. SRdy is not used.
- ARdy is separated into MemRdy and I/ORdy in order to reduce loadings & faster rise time.
- I/ORdy is connected to Expansion Slots only.
  AChip internal registers, Daybreak Mono Display, Daybreak Color Display, and Mesa Processor I/O are also located in IOP 80186 I/O space. But because they require no wait state, they don't need to pull I/ORdy low. So I/ORdy is NOT provided to them.
- All expansion devices must be in I/O space. So MemRdy in not provided to Expansion Slots.

XEROX SDD	Project Daisy	Ready Logic	File DIOP49.silx	Designer Tsang	Rev C	Date 06/12/84	Page 49



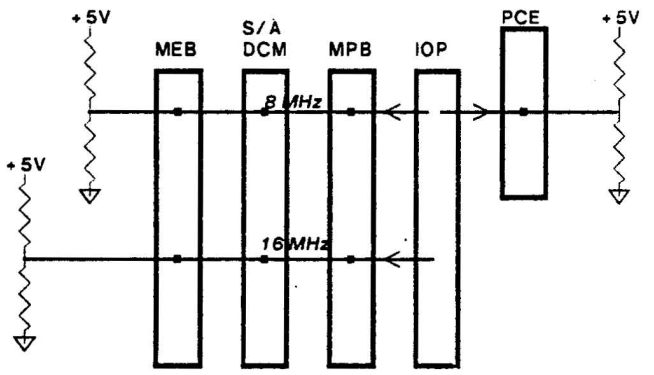


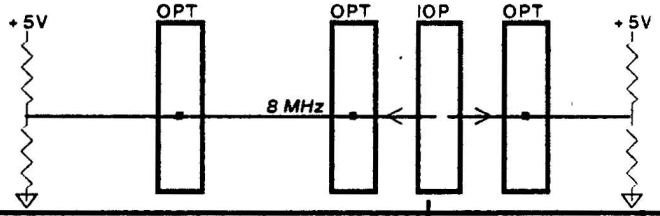
80186 ClkOut :

I_{OH} = 200 uA

I_{OL} = 2500 uA

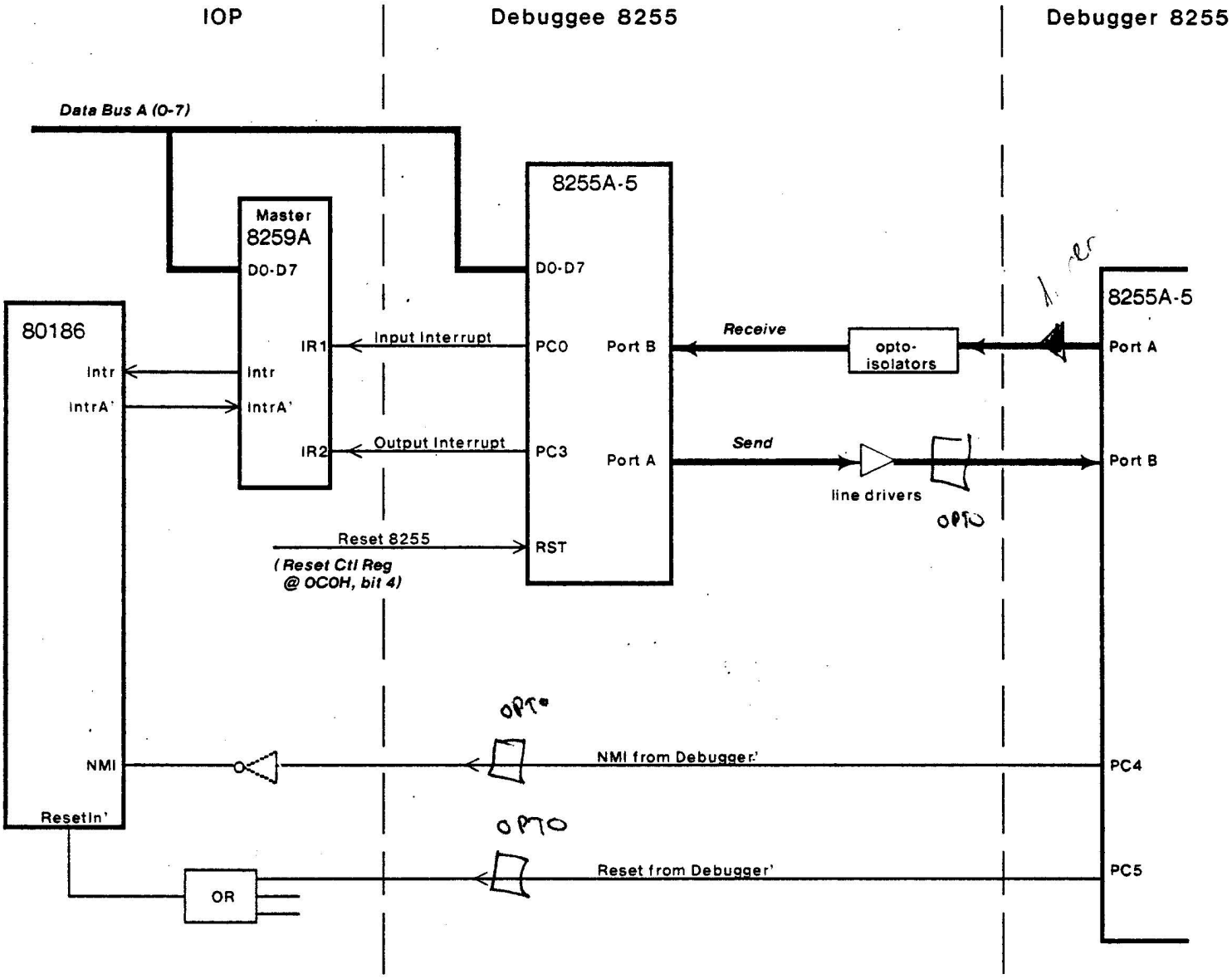
Terminations :



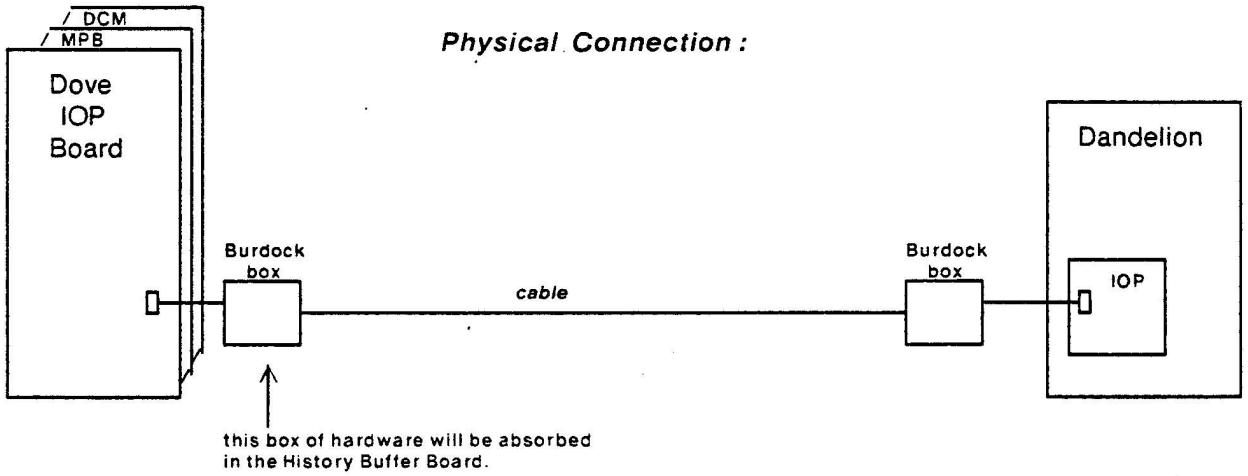


XEROX SDD	Project Daisy	Clock Distribution on IOP Board	File DIOP495.silx	Designer Tsang	Rev F	Date 6/01/84	Page 495x

IOP Burdock Port :



Physical Connection :

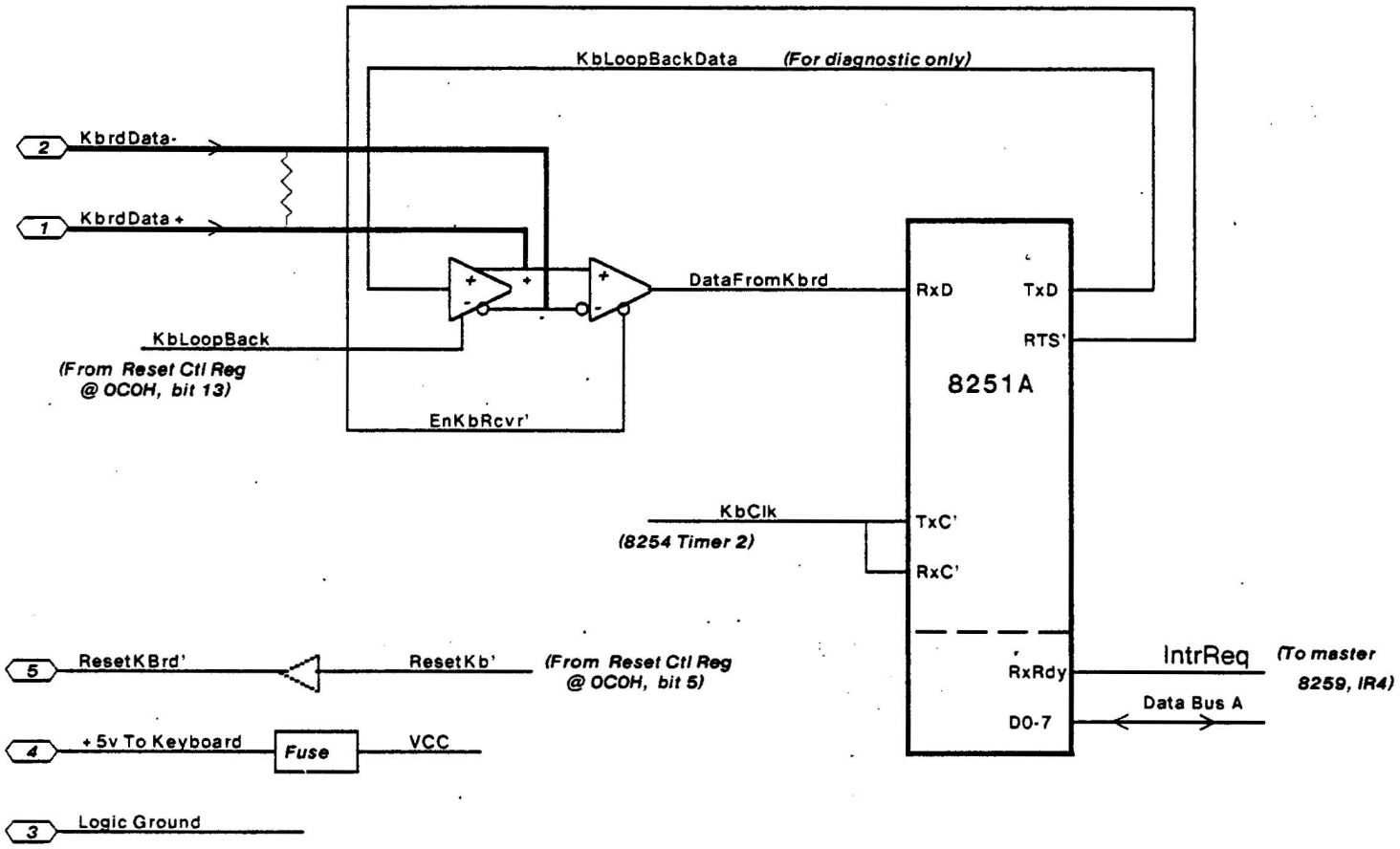


XEROX SDD	Project Daisy	IOP Burdock Interface	File DIOP498.silx	Designer Tsang	Rev F	Date 06/12/84	Page 498x

Keyboard/Mouse Controller

* Support low-profile keyboard
* 8251A UART based controller
* Serial & differential transmission from [redacted] keyboard processor at 9600 bps
* A Reset signal to allow IOP to reset keyboard processor
* Keyboard processor itself handles the mouse

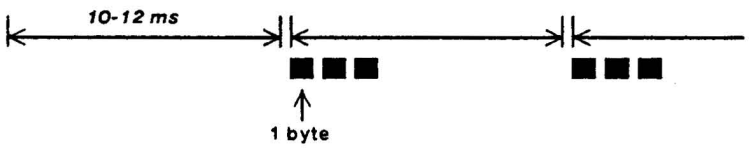
Keyboard/Mouse Interface :



Notes :

- Transmission rate : 9600 baud
- 1 start bit. 8 data bits. 1 stop bit.
  ~ 1 character per ms.
- kb takes ~ 10 - 12 ms per scan, then take
  ~ 5 ms to send 1 - 3 bytes of
  kb/mouse code to IOP.
- The receiver on the IOP is enabled by activating the RTS control signal of the 8274.
  Software must activate this signal in order to receive transmission from keyboard.

  The local driver should be disabled for normal operation.
- Transmission Picture :



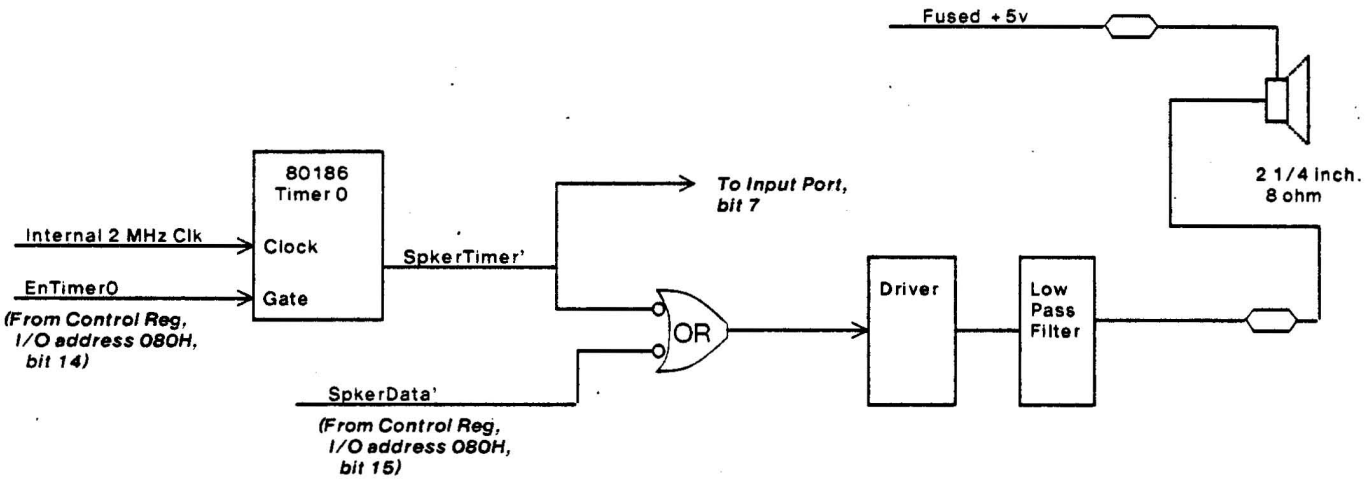
- When ResetKBrd' is active (ie, keyboard under reset), the driver at the keyboard end is disabled. So, if want to have local loopback for diagnostics without disconnecting the keyboard cable, MUST have the keyboard until reset.

  Then software can activate KbLoopBack signal to enable the local driver, and send TxD back to RxD.
- Only the receive channel (RxRdy) will generate interrupt upon receiving a character.
  When running diagnostic, should POLL TxRdy = H for transmission. TxRdy is not connected to intr ctlr.
- 6 wires connector,
  AMP DMP-R1A9S1A-xxx with right angle header.

  Connector :

KbrdData +	1
KbrdData-	2
Gnd	3
Vcc	4
ResetKBrd'	5
unused	6

Speaker Interface :



Two ways to generate sound from speaker :

1. Using the 80186 integrated Timer 0

   Set Control Reg, bit 14 = H, *(EnTimer0)*
   Control Reg, bit 15 = H, *(SpkerData)*

   Example :
   Set up Timer 0 for dual max count mode & set the ALT control bit.
   Then the timer output will be L if Max Count Reg B is being used
   for current count, H if Max Count Reg A is being used.

   This method does not tie up the processor.

2. Use software to write to the speaker directly and use software loop to generate the waveform. This will tie up the processor.

   Set Control Reg, bit 14 = L to disable the timer,
   then toggle Control Reg. bit 15, as desired.

Theory of Operation :

Send a stream of electrical pulses to drive the speaker's diaphragm in and out, producing the pulses of air movement that make up sound.

Volumn is related to voltage level (no adjustment for Daisy).
Frequency or pitch is related to shape of waveform.

Voice freq band : 200 - 3200 Hz

Mesa Processor & Control Store Interface

Model Only :

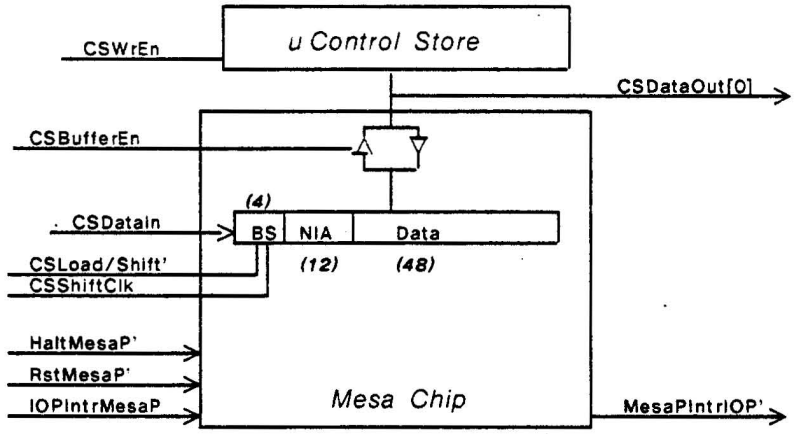
(to illustrate how the signals are used)

For more info, read the doc written by D.Davies. Stored at

Tundra

Daisy file drawer

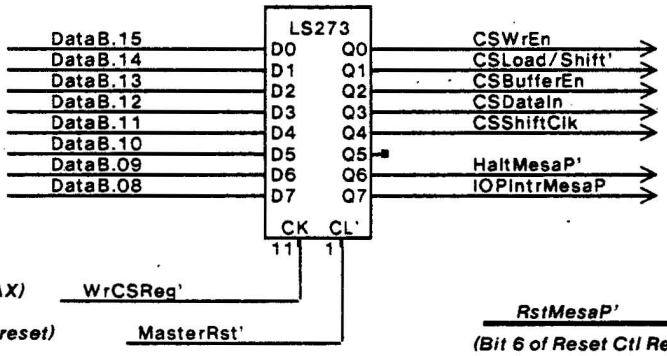
Sirius-IOP Interface



Note : Shift Reg shifts on L-to-H transition of clock.

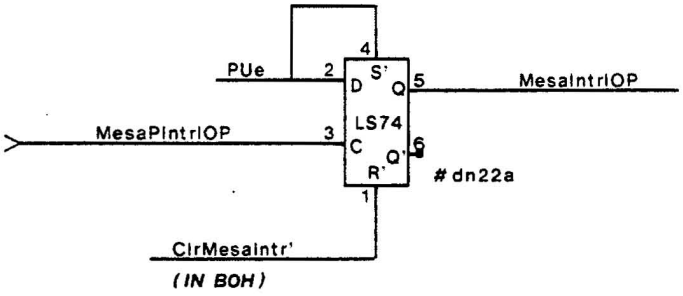
8-bit Register

From IOP



To Mesa Processor

From MP

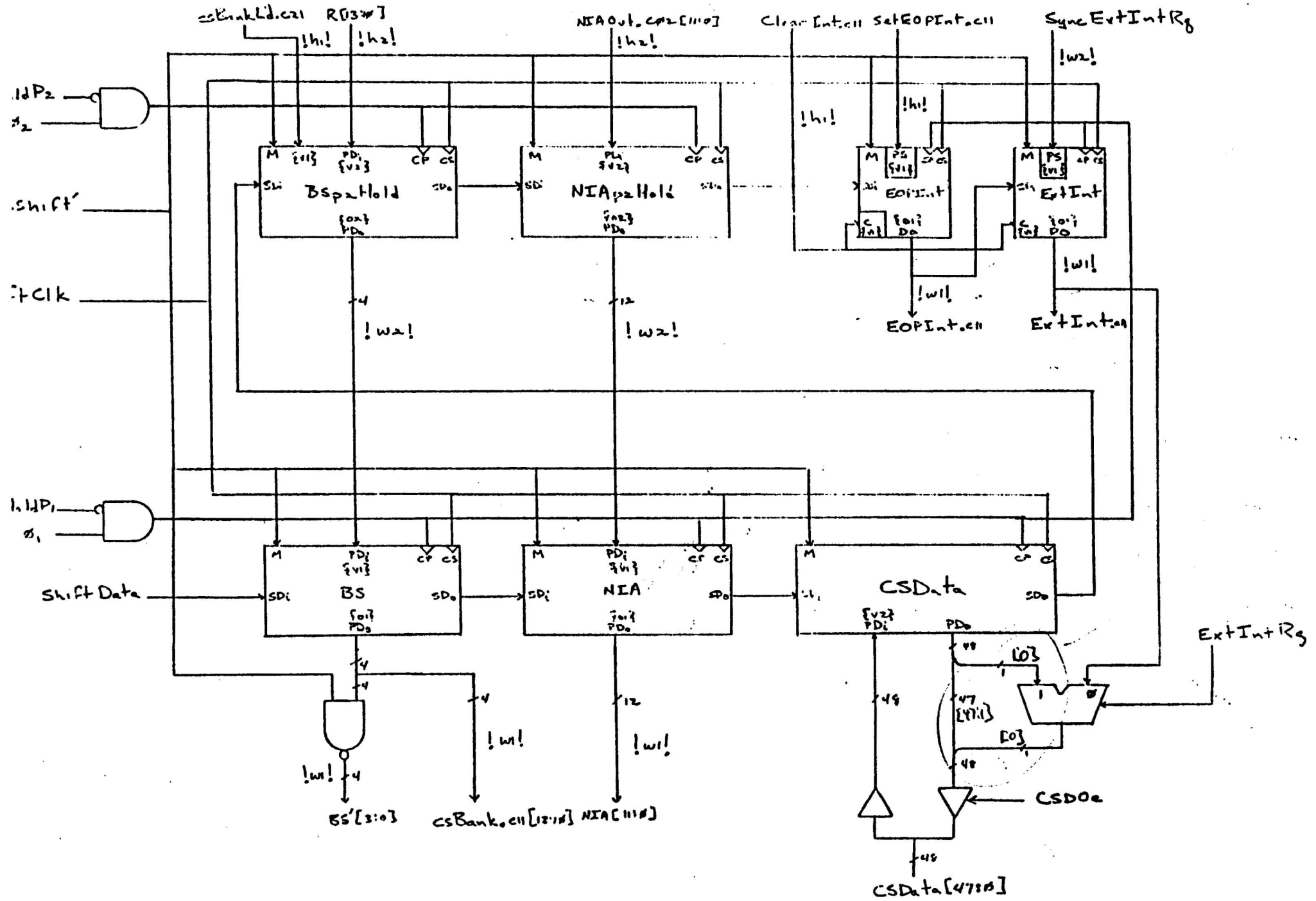


(To Slave 8259A, IR5)

From MP — *CSDataOut* — *(To Bit 13 of Input Port)*

XEROX SDD	Project: Daisy	Mesa Processor & Control Store Interface	File: DIOP48.silx	Designer: Tsang	Rev: F	Date: 6/13/84	Page: 48

Serial Interf. :



Expansion Channel

* Buffered Address Bus (16 bits only)
* Buffered Data Bus (16 bits)
* 16 KBytes (or 8 Kwords) of IOP 80186 I/O addresses are allocated for Option slots (04000H - 07FFFH)
* Interrupts are available (7 or 8, TBD)
* One 80186 integrated DMA channel (TBD)
* 3 slots in Daisy, 1 in Daybreak
* PCE Emulation slot is a special slot

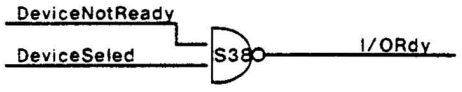
Expansion I/O Channel :

Signal Name	*Direction*	*Description*	*Loading Allowed Per Exp Slot*
A.15 - A.00	From IOP	Latch address bus. *(Not 20 bits because only I/O space).*	2 LS loads
Data.15 - Data .00	Bi-dir	Bi-directional 16-bit data bus.	2 LS loads
I/ORd'	From IOP	Read signal.	2 LS loads
I/OWrL'	From IOP	Write signal for Data.07 - .00 (low byte)	2 LS loads
I/OWrH'	From IOP	Write signal for Data.15 - .08 (high byte)	2 LS loads
ALE	From IOP	Address Latch Enable	2 LS loads
DEn'	From IOP	Data Enable. Used to enable Data Bus transceiver.	2 LS loads
DT/R'	From IOP	Data Bus direction signal. H for Transmit, L for Receive. For direction input of transceiver.	2 LS loads
I/ORdy	To IOP	Device Ready input signal to 80186. *See Note 1*	input
Reset'	From IOP	Reset signal. Low-level active. From 80186.	2 LS loads
8 MHz Clk	From IOP	Clock signal. 8 MHz, from 80186.	2 LS loads
IntrReq 0-7	To IOP	Interrupt Request inputs to 8259 Slave. Edge-triggered. *See Note 2*	input
ExpDmaReq'	To IOP	Dma Request input to 80186 integrated DMA Controller. Low-level active. *See Note 3*	input
ExpChanSel'	From IOP	Low if the address for 80186 I/O operations is within the range allocated for Expansion Slots. (4000H - 7FFFH)	2 LS loads

Notes :

- Three Expansion Slots for Daisy. One for Daybreak.
- All output signals are designed to drive 6 LS loads. two per Expansion Slots.
- All expansion devices lie in the 80186 I/O space. 16 KBytes of I/O addresses are allocated for expansion devices. From 4000H - 7FFFH.

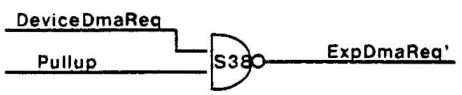
1. Suggested implementation for *I/ORdy*



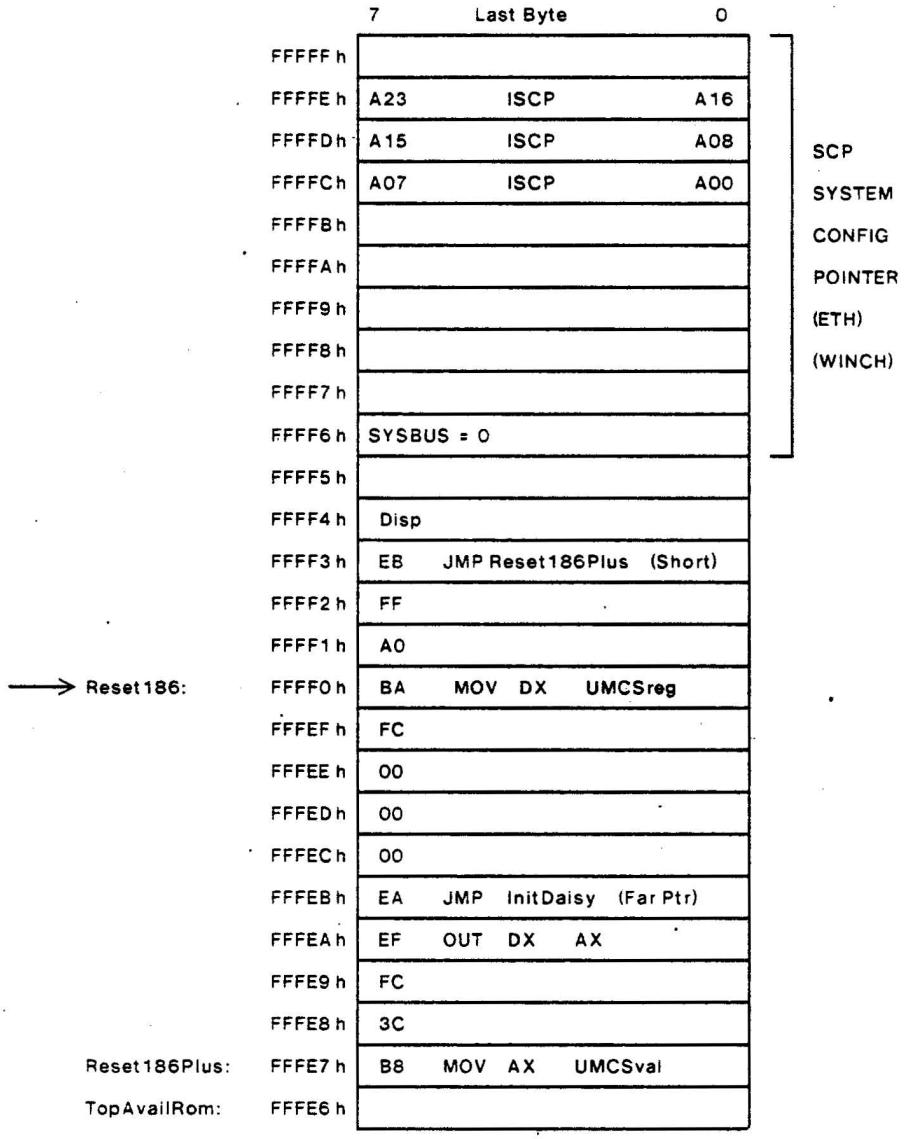
2. Interrupt Request inputs are edge-triggered

3. Suggested implementation for *ExpDmaReq'*

ExpDmaReq' is level triggered & sampled by 80186.



XEROX SDD	Project Daisy	Expansion I/O Channels	File DIOP491.silx	Designer Tsang	Rev E	Date 5/22/84	Page 491



Note : This is the way the uppermost bytes of EProm is set up in order to support ENet & RDC before Memory with mapping is available.

80186 Bugs List (as of 6/11/84)

SUMMARY OF 80186 ERRATA DESCRIPTIONS FROM INTEL

The following errata descriptions have been received from Intel and are repeated here for reference. Future developments will be reported over the Laurel mail system. There is currently a B-1 stepping, and a B-2 stepping. There are now 8 problems identified so far. They are described below. Note that the only problem fixed by going from B-1 to B-2 is problem number 5.

#1. DMA REGISTERS (problem in mask B-1 and B-2)

Any read of the upper 4 bits of the 20-bit pointer registers in the integrated DMA controller will yield all 0's. The DMA controller will continue to operate correctly if these registers

are read. This does not prevent the DMA controller from responding properly to all 20 bits of the DMA memory location when a DMA cycle is run. The upper 4-bits must still be programmed with their correct value.

SOLUTION:

If the content of these upper 4 bits is required, it can be determined by reading the DMA count register to determine the number of DMA transfers which have occurred and adding this to the value with which the register was programmed.

#2. QUEUE STATUS (problem in mask B-1 and B-2)

When executing any instruction that causes a non-sequential instruction fetch (JMP, Conditional JMP with true condition, Call, IRET, etc.), the processor may not indicate "empty the queue" status on the 80186 queue status lines. This only affects operation with the 8087, or any device which attempts to track 80186 execution using these signals. The queue status lines are not normally driven by the 80186. The part must be strapped into the queue status mode for these signals to be available outside the 80186.

SOLUTION:

There is no immediate solution except not to use the IRET instruction when using the 80186 with the 8087.

#3. IMPROPER INTERRUPT VECTORING (problem in mask B-1 and B-2)

When both the INT0 and the TEST/ line go high simultaneously, the 80186 will fetch the interrupt vector from the wrong location in the interrupt vector table. The TEST/ signal is used with the 8087 to allow concurrent execution between the two processors.

SOLUTION:

The TEST/ line should always be tied low. This will allow the part to operate correctly, while not locking up the processor when the wait instruction is executed. This also only affects operation with the 8087.

#4. NON-CONTIGUOUS INTA CYCLES (problem in mask B-1 and B-2)

When using DMA and the internal interrupt controller (cascaded, nested, fully nested, or RMX86 modes), it is possible to get a DMA cycle between the two INTA cycles.

SOLUTION:

If it is recognized that an interrupt is coming in, external logic should be used to block the DMA request lines until after the first INTA cycle has been completed. This will allow the second INTA cycle to be run before the DMA request is recognized. The user should be positive that interrupts are enabled (STI instruction), otherwise the DMA may never be serviced.

#5. STRING MOVE INSTRUCTION (problem in Mask B-1 only)

The problem relates to the interaction of a bus HOLD with the execution of the STRING MOVE instruction. When a HOLD is received by the 80186, it will release the bus to the requesting device. When the HOLD request is released, the 80186 should resume executing instructions. A HOLD request could come from either the external bus (using the HOLD input to the processor), or from the

internal DMA controller when a DMA transfer is pending.

IF THE 80186 FETCHED ONE OF THE STRING MOVE INSTRUCTIONS, SUCH AS MOVS, INS, AND OUTS, BEFORE RELEASING THE BUS, IT WILL NOT PROPERLY BEGIN THE STRING MOVE INSTRUCTION AFTER REGAINING CONTROL OF THE BUS.

The bug has only been observed with the string move instructions. The string move instruction does not need to be preceeded by a repeat prefix in order to fail. That is, both single iteration and multiple iteration string instructions fail.

The following conditions must occur for the string move instruction to fail:

1. The string move instruction must not have begun execution at the time the bus is relinquished. If the string move instruction had begun execution, the instruction will properly resume after bus control is regained.

2. The last bus operation performed before the HOLD is acknowledged must have been either a memory or I/O read cycle. NOT a memory or I/O write OR an instruction fetch. Thus, the string move instruction must be located in the prefetch queue at the time of the bus HOLD, because otherwise an instruction fetch would be required before the instruction could begin execution.

This problem is most likely to impact 80186 users who:

1. Have no control over the code sequences ececuted by the processor, i.e., those using a high-level language which generates the specific code sequence to be executed.

2. Build user reprogrammable systems, for example personal computers. Since these customers have little control over the programs their users will eventually attempt to run, they cannot prevent execution of the specific sequence which will allow the problem to occur.

PROBLEM CHARACTERIZATION

The string move instruction is performed by reading a value from a memory location (pointed to by the SI register) and writing the value to another memory location (pointed to by the DI register). After each iteration of the string move, both the SI and DI registers will be adjusted to point to the next element of the source or destination strings, respectively, so the next iteration of the instruction moves the next sequential byte or word.

The 80186 error is characterized by the string move instruction executing all the required reads, but not properly executing the required writes; thus the destination string is not the same as the source string. If the instruction is not executed properly by the 80186, the DI register will not be adjusted after every iteration of the move, since the memory write has not been performed. Thus, the SI register (which points to the source string) will be properly adjusted but the DI register (which points to the destination string) will not be properly adjusted by the number of bytes or words moved.

For the OUTS instruction, the destination for the data is the port addressed by DX. This register is not incremented by the instruction. There is, therefore, no way to determine from the register contents whether the correct number of I/O writes have

occurred.

POTENTIAL WORKAROUNDS: There are 5 possibilities

1. DO NOT USE THE HOLD INPUT OR INTEGRTED DMA CONTROLLER

   Since the problem will occur only upon returning from a bus HOLD, if a bus HOLD never occurs, the problem will not be observed. You can perform a pseudo-hold using external logic to prevent the processor's driving the bus, while forcing the processor to wait by manipulating its ready lines (of course this will not work with the integrated DMA unit).

2. DO NOT USE THE STRING MOVE INSTRUCTIONS

   Since the bus HOLD problem only interferes with the resumption of the string move instructions, if they are not used, the problem will not be observed. This may be difficult to do, however, since string move instructions are generated by some compilers, including the PL/M 86 compiler.

3. EXAMINE THE DI REGISTER AFTER A STRING MOVE AND CHK IT

   Since the DI register will not have been adjusted the proper amount if the string move instruction has not correctly resumed, its value can be compared with the proper value after instruction execution to insure the instruction has executed properly. This may also be difficult to do with code generated by compilers. In addition, this may not be used to determine the correct execution of the OUTS instruction, since this instruction never manipulates the register in tne first place.

4. COMPARE THE DEST. STRING WITH THE SOURCE STRING

   Comparing the destination string with the source string is a sure way to insure that they are equivalent, and that the string move instruction has executed properly.

5. AVOID INSTRUCTION SEQUENCES ALLOWING PROBLEM TO OCCUR

   The last bus cycle performed prior to the string instruction beginning execution MUST be a read cycle to prevent proper execution of the string instruction. If a write cycle is inserted immediately before the string in struction, or if an instruction fetch cycle is forced immediately before the instruction, it will execute properly. The example below shows how this may be done. Note that by jumping to the string move instruction, the queue is flushed, forcing instruction prefetch cycles between execution of the memory read and the string move instruction.

```
BAD SEQUENCE                                  GOOD SEQUENCE

pop   AX;      executes a memory read         pop   AX
rep   movs;    may fail if HOLD occurs        jmp   A
                                                A:  rep  movs
```

#6. IDIV PROBLEM (problem in mask B-1 and mask B-2)

"IDIV" instructions give incorrect results under certain conditions as follows: If the divisor is a memory reference and the value in memory is a negative number (i.e. MSB = 1), the result in AX (AL) is the 2's complement of the expected result. Remainder (DX or AH) is correct.

Note: If the divisor is a register reference. there is no failure.

```
EXAMPLE:    MOV   WORD   PTR   55. 0FFFEH   ;STORE 2
            MOV   AX,    40H
            MOV   DX,    0                          ;DOING 32 BIT/16 BIT
            IDIV  WORD   PTR   55                 ;SIGN-EXTEND AX THRU
                                                        ;THROUGH DX
```

SOLUTION:

A: In assembly language:

```
MOV   BX.   WORD   PTR   55
IDIV   BX
```

i.e., Move memory reference to a register and dthen use IDIV.

B: In high level language:

```
IF divisor < 0
  THEN Result = -(Numerator/(-Divisor))
   ELSE Result =    Numerator/Divisor
```

#7. LOCK PREFIX TIMING (problem on mask B-1 and mask B-2)

The LOCK/ signal does not remain active until the end of the last data cycle of a locked transfer. This may cause problems in some systems if. for example. the processor requests memory access from a dual ported RAM array and is denied immediate access (because of a DRAM refresh cycle, for instance). When the processor finally is able to gain access to the RAM array, it may have already dropped its LOCK/ signal. thus allowing the dual port controller to give the other port access to the RAM array instead.

SOLUTION:

A circuit to hold LOCK/ active until a RDY has been received by the 80186 should be used.

This circuit is shown in Fig. 15 of Application Note (AP-186). Order Number 210973-001.

#8. TIMER CONTROLLER REGISTER Found Jan. 1984

PROBLEM:

When writing a value into one of the control registers associated with timer n. it is possible to overwrite a register value associated with timer n+1. For example. writing into a register of timer 0 may alter a register of timer 1. Writing into any register of timer 2 does NOT affect a register of timer 0 or 1.

The problem can occur only when the following two conditions exist:

1. The write instruction. i.e.. MOV or OUT. to a register of timer n is followed by either an instruction fetch or a DMA cycle.

   and....

2. The lower byte of the address for either the instruction fetch or the DMA cycle equals the offset of one of the registers in timer n+1.

SOLUTIONS:

Any one of the five solutions outlined below will prevent the problem from occurring in your system.

1. Use only a single timer or use only timers 0 and 2.

2. Disable the DMA (e.g. set the DHLT bit or clear the stop/start bit) when writing to timer registers and use LOCK PREFIX on the write instruction to the timer register.

3. Write into registers in a sequential order since writing to timer n does not affect timer n-1 or other registers of timer n. For example, if your system is using all three timers, then write into the timer registers for 0, followed by the registers of timer 1, followed by the registers of timer 2.

4. Decode any write into any of the timer registers, and assert HOLD until the completion of the write.

5. Turn off DMA during all writes to the timer registers and locate the code in XXX0X .... XXX3X .... XXX7X .... XXXFX addresses.

80186 BUG #9 BHE/ DURING DMA

PROBLEM: When Read to either DMA control register is followed by DMA byte transfer from an odd to even or an even to odd Address, BHE/ may not go active on the odd byte Read or Write.

SOLUTION: Use a locked Read when reading the control registers.

---

186 STEPPING	BUGS PRESENT
B-1	1 through 9
B-2	1, 2, 3, 4, 6, 7, 8, 9 (String Mov fixed)
B-3	1, 2, 3, 4, 7, 8, 9 (IDIV fixed)
C	none!

NOTE ON THE C80186-3 (B2-STEP) RELAXED SPEC

The C80186-3 8 MHZ device has the following relaxed specifications:

VCC	5V tolerance 5% (not 10%)
TEMP	0 DEG. C TO 50 DEG. C (not 70 DEG. C)
DC PARA.	VCLI (clock input low voltage) MAX=0.3 (not 0.6)
AC PARAMETERS:	
TCLDX	min. 20 ns (not 10ns)
TCLAV	max. 59 ns (not 44ns)
TCHCTV	max. 73 ns (not 55ns)
TCLHAV	max. 67 ns (not 50ns)
TAVAL	min. TCLCH-40NS (not TCLCH-25ns)
TCHSV	max. 65 ns (not 55ns)
TCLSH	max. 65 ns (not 55ns)
TCVDEX	max. 70 ns DEN/ inactive delay, non-write cyc. (new)
TCLCSV	min. 12 ns (not 0ns)
TCHCSX	max. 47 ns (not 35ns)

Floppy disk subsystem

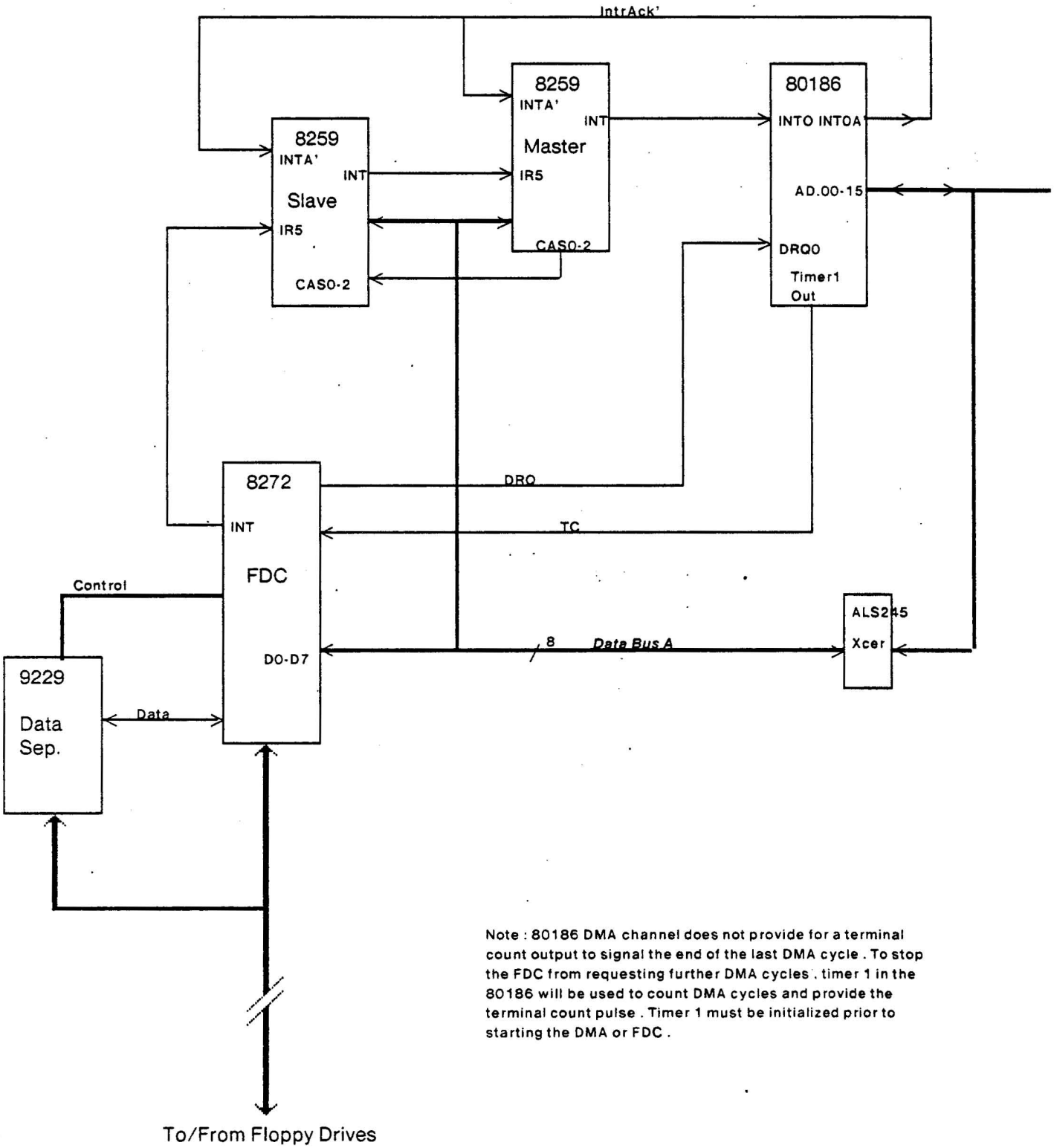
Provides support of one or two floppy disk drives

Controller implemented using integrated floppy disk controller and PLL

Floppy drive support

- One or two half-height , double sided , single/double density drives
- Controls Shugart SA455/465 compatible drives
- 48 tpi drives (IBM PC compatible) and 96 tpi drives supported
- Nominal unformatted capacity: 320 KBytes (48 tpi)
- 250 Kbit data transfer rate
- DMA using 80186 DMA channel
- IBM PC/ Industry standard $5\frac{1}{4}$ inch floppy diskette format

System Interface :

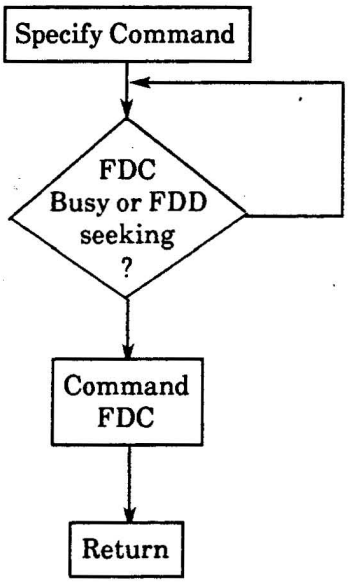


Note : 80186 DMA channel does not provide for a terminal count output to signal the end of the last DMA cycle . To stop the FDC from requesting further DMA cycles , timer 1 in the 80186 will be used to count DMA cycles and provide the terminal count pulse . Timer 1 must be initialized prior to starting the DMA or FDC .

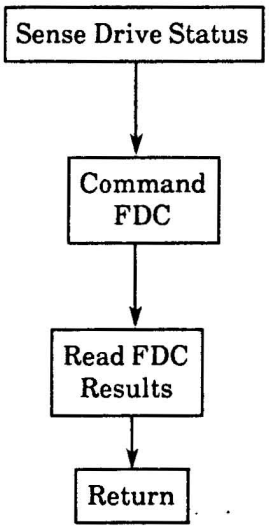
Floppy Diskette Summary

	48 TPI	96 TPI
Diskette Size	5¼ inches	5¼ inches
Sides	2	2
Tracks / Side	40	80
Tracks / Diskette	80	160
Sectors / Track	9	9
Bytes / Sector	512	512
Bytes / Track	4608	4608
Bytes / Side	184.32K	368.64K
Bytes / Diskette	368.64K	737.28K
Rotational Speed	300 RPM	300 RPM
Transfer Rate	250K bits/sec	250K bits/sec
Recording Method	MFM	MFM

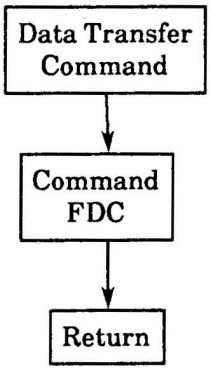
Generalized FDC Command Sequence



Note: No execution phase or result phase .

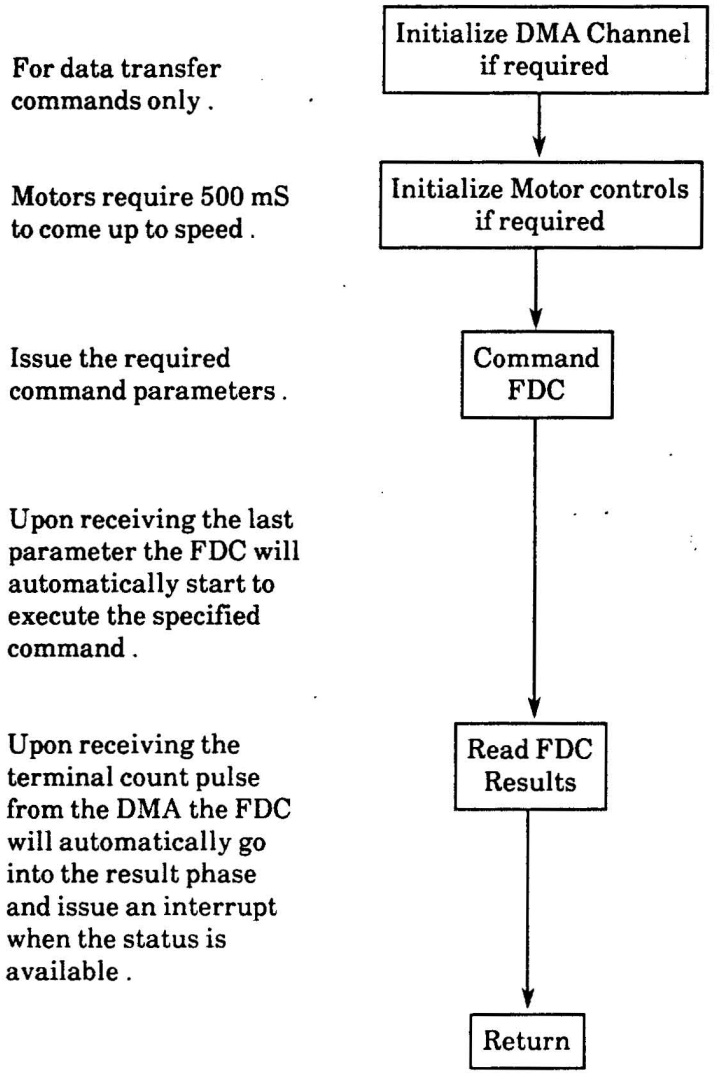


Note: No execution phase .

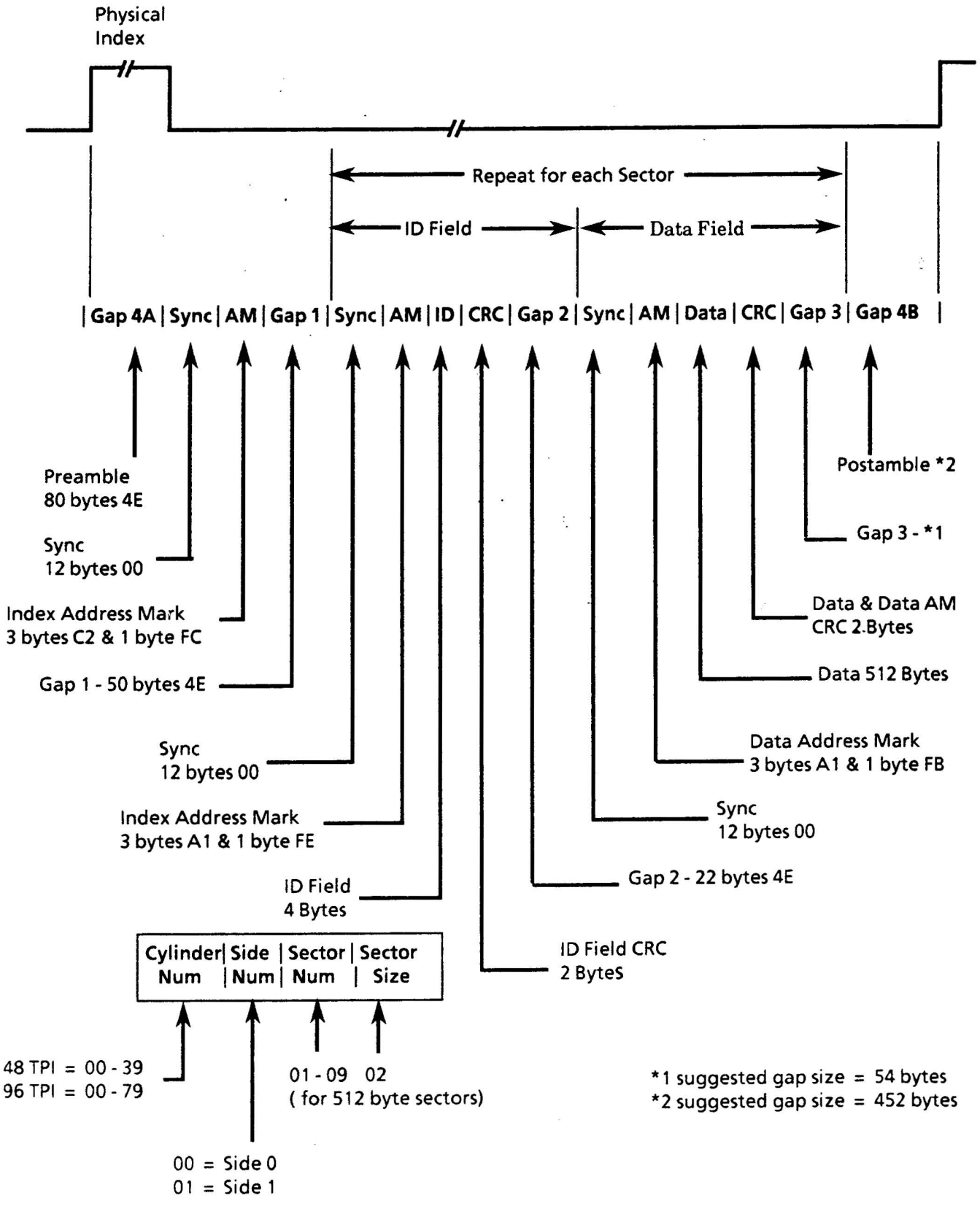


Note: FDC automatically enters execution phase upon completion of command phase .

Generalized Floppy I/O Sequence



5¼ Floppy Diskette Format



RS-232C Subsystem

Provides support of two RS-232C Channels

Baud Rates up to 9600 Baud in asynchronous or synchronous modes

Byte Synchronous protocols - IBM Bisync

Bit Synchronous protocols - SDLC / HDLC

Parity and CRC generation and checking

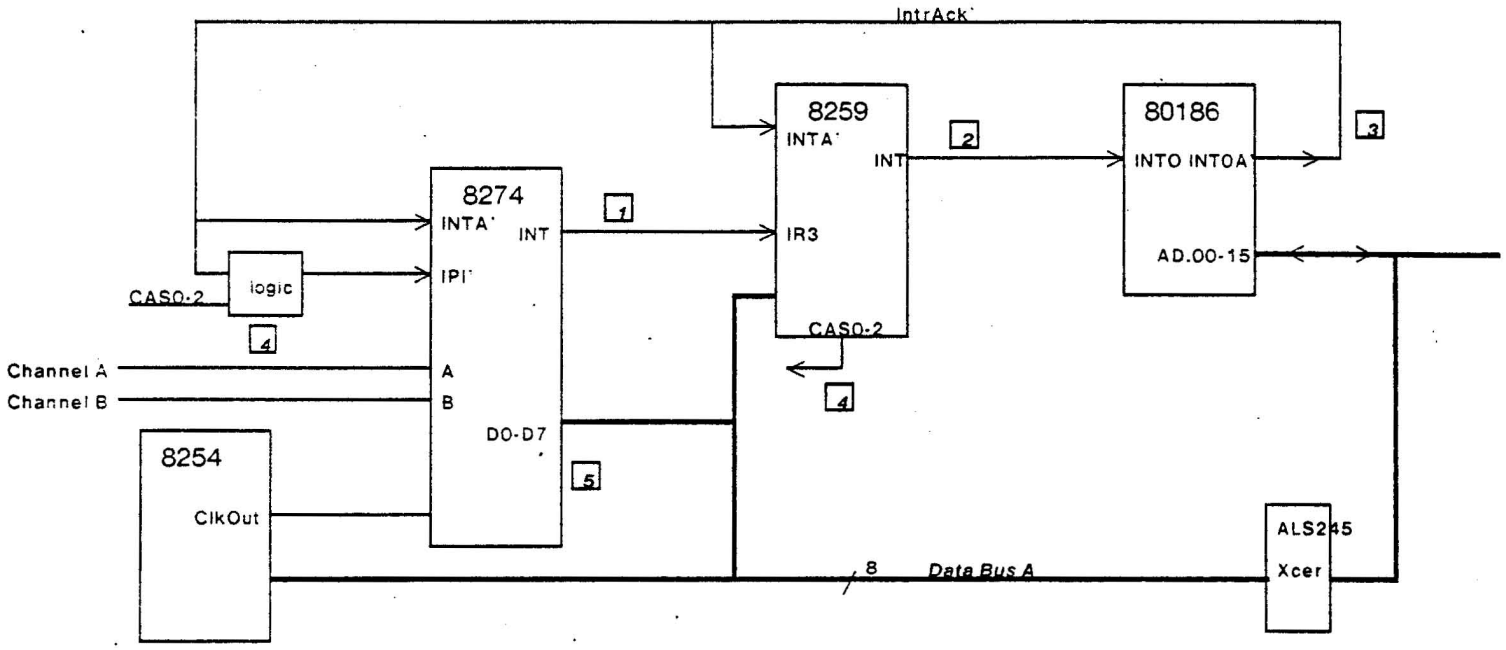
Controller generated Interrupt Vectors

1 Port configured as a DCE port

1 Port configured as a DTE port

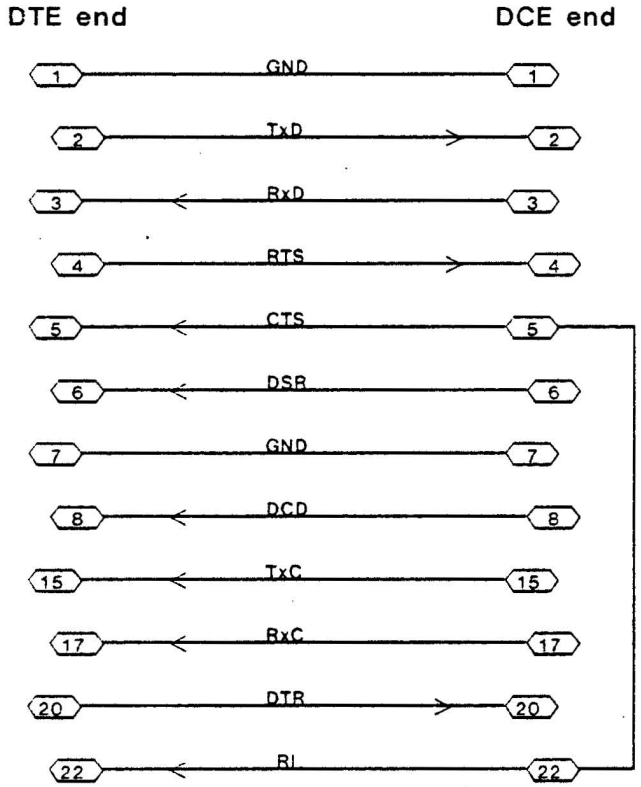
2 byte buffer in controller

System Interface :



1. When an internal interrupt condition occurs (and if it is accepted). 8274 will cause an interrupt request to the Master Interrupt Controller 8259.
2. Upon receiving the interrupt request from 8274, and if it meets the conditions at that time. 8259 will generate an interrupt to the 80186 uP.
3. The uP will respond with two contiguous Interrupt ack cycles when it can service the interrupt.
4. Then, upon receiving the first INTA pulse, the 8259 will generate the Cascade signals. CAS0-2, which is then decoded.
5. If the CAS0-2 are decoded for 8274, the 8274 will then deliver the interrupt vector on the Data-Bus-A during the second INTA cycle.

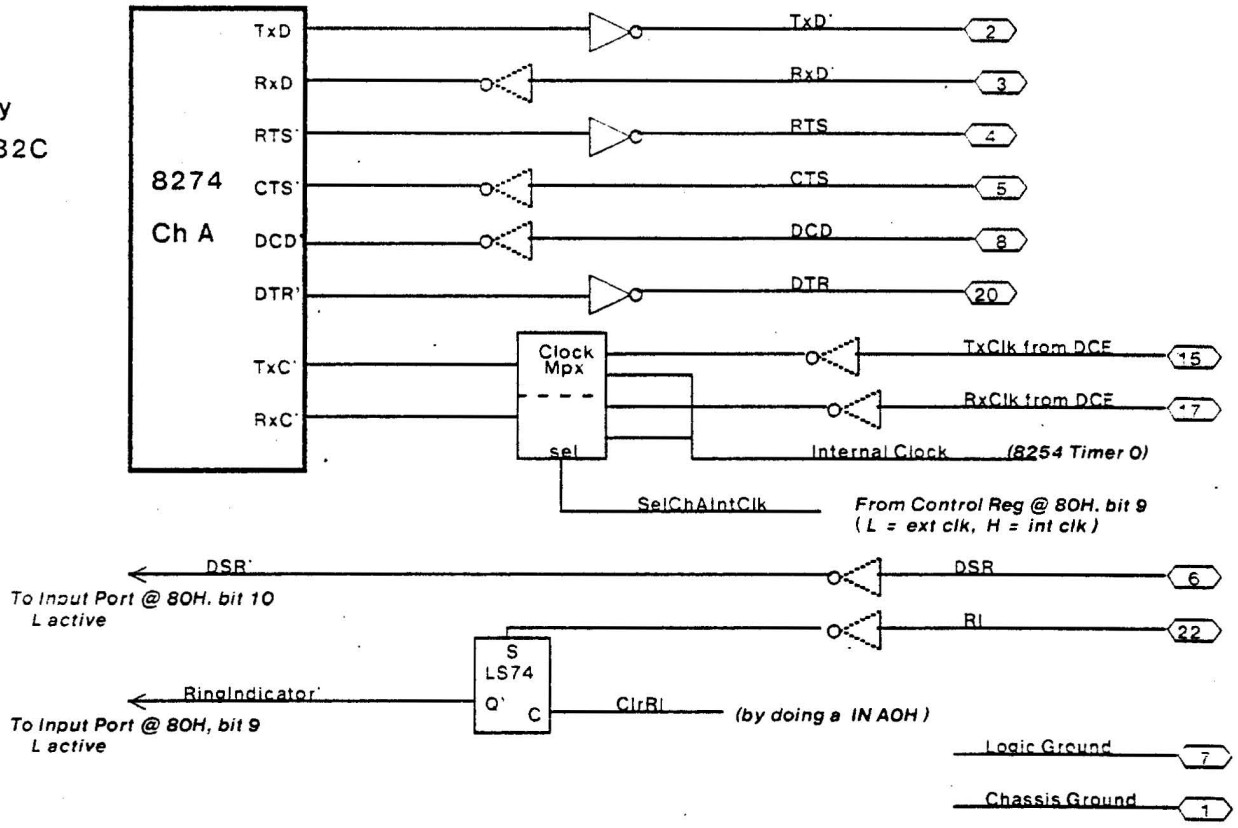
DTE to DCE Connection for External Loopback :



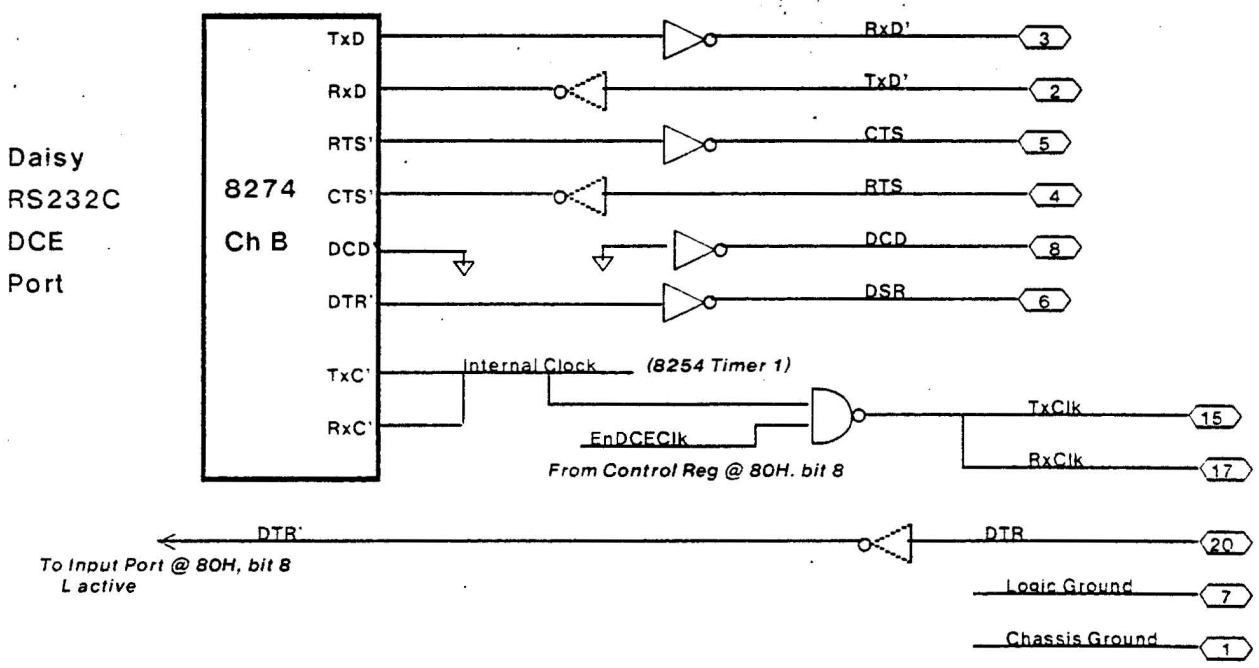
- If internal clock is used, loopback w/i a single RS232C channel can be performed by connecting :

TxD	*to*	RxD	(2 to 3)
RTS	*to*	CTS	(4 to 5)
DTR	*to*	DSR	(20 to 6)

Daisy
RS232C
DTE
Port



Daisy
RS232C
DCE
Port



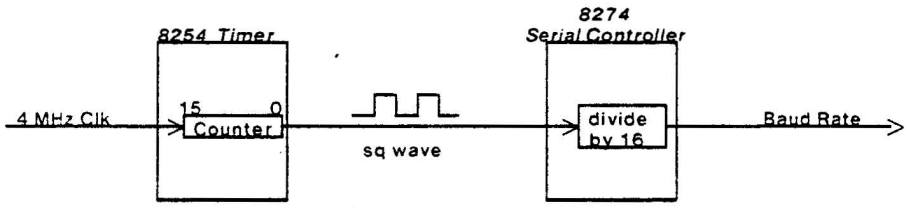
Features :

- Two independent full duplex channels
- Asynchronous, Byte Synchronous, & Bit Synchronous Operations
- Asyn transmission baud rate : up to 9600 Baud
- Synch transmission rate : up to 9600 Baud
- Interrupt driven operation, vector mode
- Internal Interrupt Priority : RxA, RxB, TxA, TxB, ExTA, ExTB
  or RxA, TxA, RxB, TxB, ExTA, ExTB

Timer

8254 : *(8 bits wide)*

Timer0 :	RS232C DTE baud rate	(clock in = 4.0 MHz)
Timer1 :	RS232C DCE baud rate	(clock in = 4.0 MHz)
Timer2 :	Keyboard / Mouse baud rate	(clock in = 4.0 MHz)



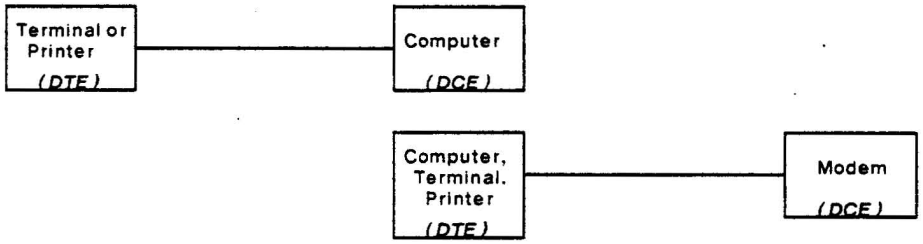
Baud Rate	*Time Const*	*% Error*
9600	26	0.16 %
7200	35	0.79 %
4800	52	0.16 %
3600	69	0.64 %
2400	104	0.16 %
2000	125	0 %
1800	139	0.08 %
1200	208	0.16 %
600	417	0.08 %
300	833	0.04 %
150	1667	0.02 %
134.5	1859	0.01 %
110	2272	0.03 %
75	3333	0.01 %
50	5000	0 %

$$\text{Time Constant} = \frac{4 \times 10^6}{(\text{baud rate}) \cdot (16)}$$

$$\%\ \text{Error} = \text{abs}\left[\frac{4 \times 10^6}{(\text{time const}) \cdot (\text{baud rate}) \cdot (16)} - 1\right] \times 100\%$$

Asynchronous Timer Parameters

Typical Connections :



Interface Signals :

Connector Pin No [1]	Signal Name [2]		Direction of Flow	Signal Type	Active Level [3]
1	GND	*Chassis Ground*	DTE —— DCE		
2	TxD	*Transmitted Data*	DTE → DCE	Data	- 12 V
3	RxD	*Received Data*	DTE ← DCE	Data	- 12 V
4	RTS	*Request To Send*	DTE → DCE	Control	+ 12 V
5	CTS	*Clear To Send*	DTE ← DCE	Control	+ 12 V
6	DSR	*Data Set Ready*	DTE ← DCE	Control	+ 12 V
7	SG	*Signal Ground*	DTE —— DCE		
8	DCD	*Data Carrier Detect*	DTE ← DCE	Control	+ 12 V
15	TxC	*Transmitter Clock*	DTE ← DCE	Timing	
17	RxC	*Receiver Clock*	DTE ← DCE	Timing	
20	DTR	*Data Terminal Ready*	DTE → DCE	Control	+ 12 V
22	RI	*Ring Indicator*	DTE ← DCE	Control	+ 12 V

Note :

1. RS232C interface uses a 25-pin DB type connector.
   Typically, DTE uses a MALE connector,
   DCE uses a FEMALE connector.

2. RS232C signal names are w/r to DTE.

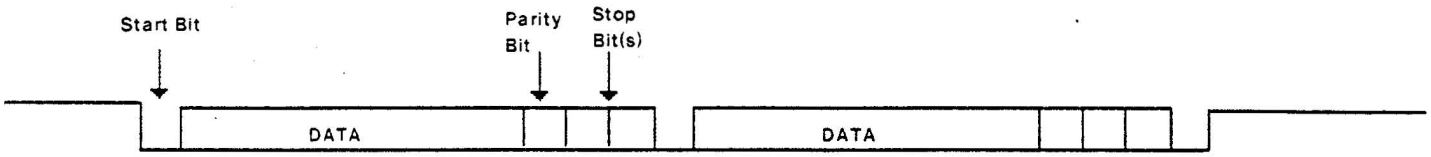
3. Signal voltages on the interface lines are nominally + 12v & -12v.
   * For Data signals, active => -12v = mark.
     inactive => + 12v = space.
   * For Control signals, active => + 12v = ON.
     inactive => -12v = OFF

4. Some mfgers use pin 9 for + 12v,
   pin 12 for -12v.

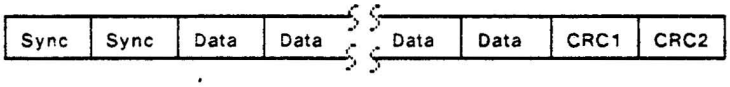
5. If a DTE were to be connected to another DTE (or a DCE to another DCE), the following pairs would have to be flipped :
   (2, 3)
   (4, 5)
   (6, 20)

6. In asynch mode, loopback w/i a single RS232C channel can be done by connecting
   2 to 3
   4 to 5 to 8 (and to 22 if implemented)
   20 to 6

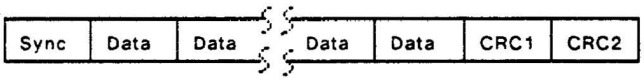
* In normal use, DTE brings up DTR, and DCE brings up DSR when they are powered on.
* In asynch, full duples environment. DTE then brings up RTS and awaits CTS from the DCE. (If the DCE is a modem, CTS is brought up along with DCD, once the incoming carrier has been detected).
* At this point, the DTE may transmit data to DCE over line 2, and the DCE may transmit data to DTE over line 3.
* In half-duplex system, RTS and CTS are used to determine which direction the data will be going over the wire, eg, the phone line.



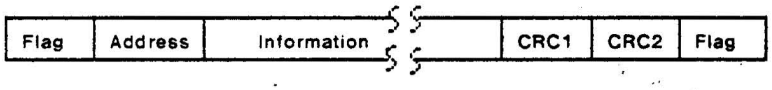
Asynchronous Mode



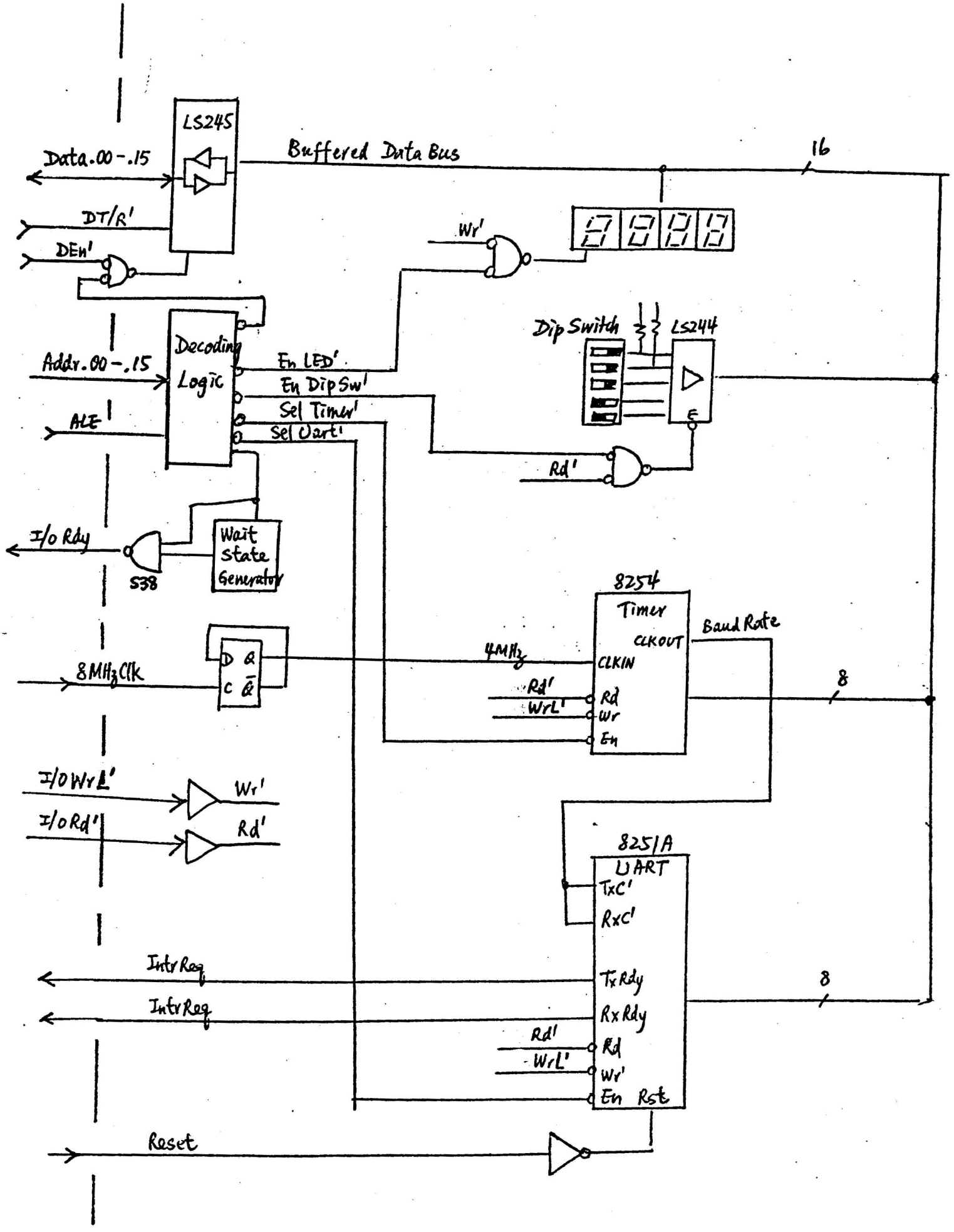
Bisync



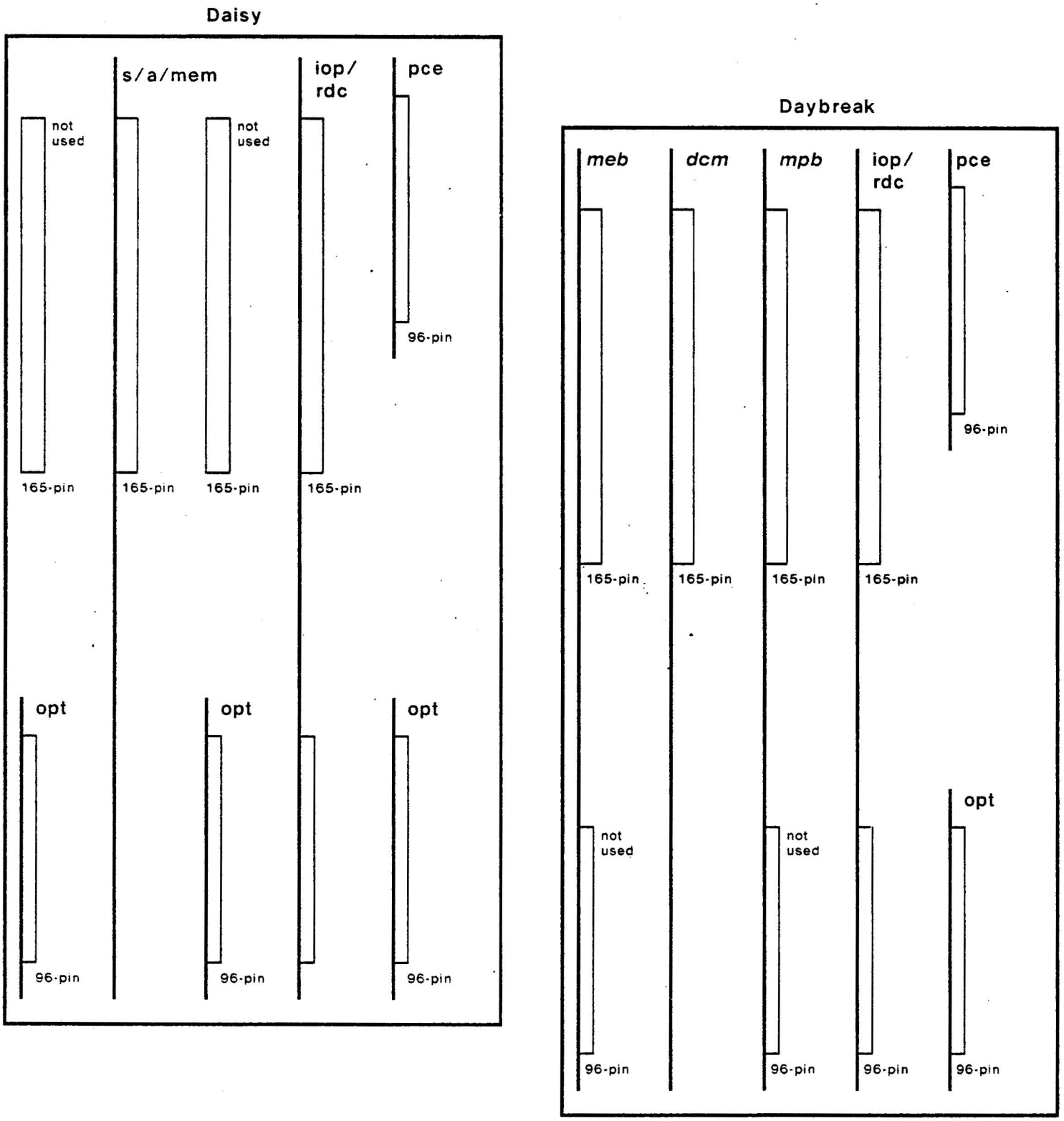
Monosync



SDLC/HDLC/X.25



Sample Implementation of a Option Board



Note: connector for FDD and RDD are TBD

Note: If MEB is only 10.9" x 10" then one extra option slot is available to Daybreak

XEROX EDC	*Project* Dove	Daisy/Daybreak Slots Usage	*File* BP02.sil	*Designer* ATsang/UNguyen	*Rev* B	*Date* 05/23/84	*Page* 02