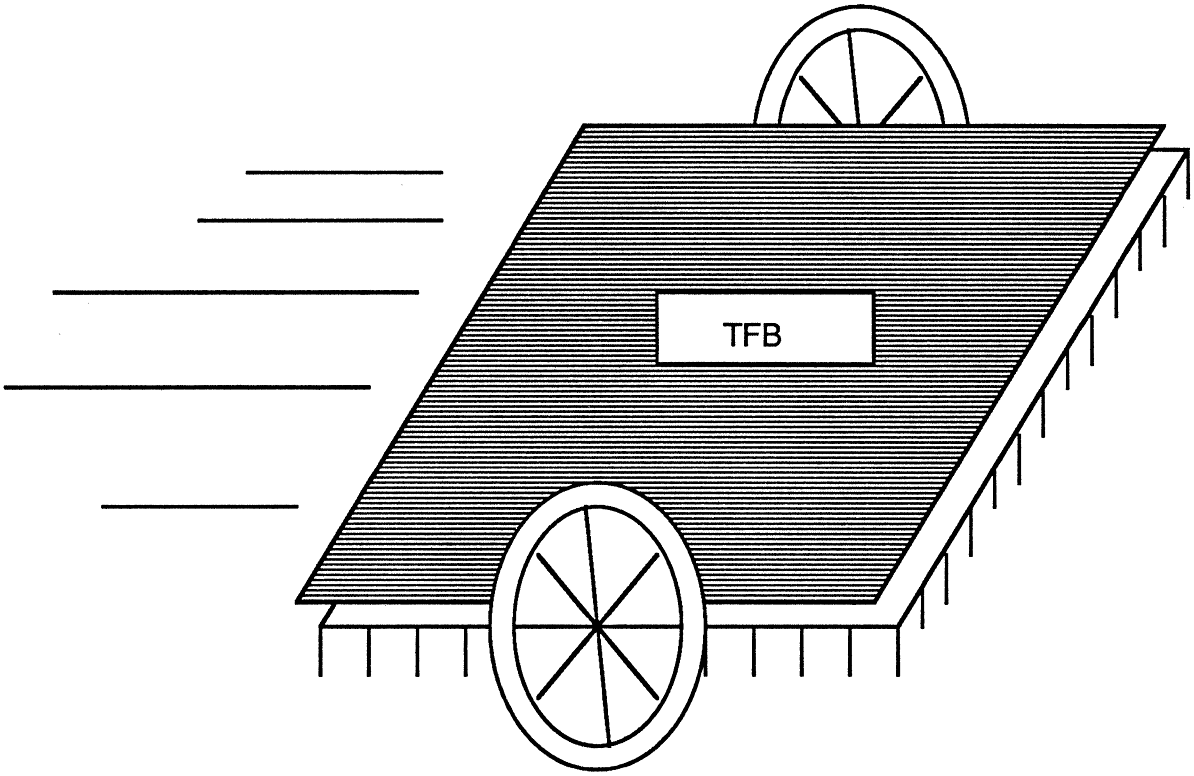
TFB SPECIFICATION



Frame buffer specification for 1 February 1986.
comments and questions are encouraged, direct them to
Toby Farrand at M/S 22Y x2524.

UPDATES TO THE SPECIFICATION

The following is a list of changes made to the frame buffer specification for the 28 June 1985 version of the document.

1 - Support for the planar color model has been dropped from section 1.3.
2 - The frame buffer memory map has been updated in section 2.0.
3 - Many of the control register definitions have changed in section 3.0.
4 - Several pin definitions have changed in section 4.0.
5 - Test mode operation has changed, and is explained in section 4.3.
6 - Planar mode accesses have been eliminated.
7 - Data transfer cycles are changed in section 5.2.
8 - Appendix A has been added to suggest possible configurations using the TFB.
9 - Timing information has been added in Appendix B.

The following is a list of changes made to the frame buffer specification for the 1 February 1986 version of the document. These changes apply only to the 1.1 version of the TFB.

1 - Support for variable depth color has been added to the chip.
2 - Support for multiplexed address and data buses has been added to the chip.
3 - The speed of the chip has been increased substantially.
4 - The bus interface has been simplified.
5 - The chip parameter descriptions have been rewritten and elaborated on.
6 - The SC1~ pin has been eliminated in favor of a pixel clock output pin.
7 - The DS~ pin has been eliminated in favor of a dedicated test mode pin.
8 - The definitions and timing of the WEN and CMA buses have been changed.
9 - All of the figures and diagrams have been updated.
11 - Several configuration paramenter have been added.
10 - The appendicies have been updated.

TABLE OF CONTENTS

1.0 INTRODUCTION

One distinguishing characteristic of Apple's computer products is the tight coupling our machines have between their memory and video systems. This tight coupling results in products which have superior graphics in terms of resolution, speed and cost. This architecture's costs are significant,however, as the video refresh circuitry typically consumes between 40-50% of the available bus bandwidth. The demand for increased processor speed, and high-resolution, deep-color displays make the overhead associated with Apple's traditional video implementation unacceptible. Consequently, the TFB frame buffer controller chip was designed to provide a flexible, high-speed, modular and inexpensive frame buffer design. The TFB design is now complete, and the chip is operational in two systems to date. This document provides a complete technical specification of the part from both a software, and a systems design viewpoint.

1.1 How to Read this Document

Not all of this document is relevant to all readers. All programmers and system designers using the TFB sytems should read sections 1 and 2, as these give general information on the TFB. Software engineers interested in programming the TFB should refer to section 3 for a description of the control registers which define the chip's operation. Systems designers should read section 4 for a description of the signals found on the TFB. Those interested in the actual operation of the TFB should read section 5 describing the three basic types of RAM accesses the TFB will make.

Once the engineer is familiar with the general operation of the TFB, reference to the application note in Appendix A can be helpful.

1.2 System Configuration

Although the TFB is designed to be flexible enough to be integrated into a wide range of designs, the frame buffer subsystem is likely be quite similar from system to system. The figure below shows a typical implementation of a frame buffer using the TFB. This figure shows the TFB in a traditional, tightly coupled CPU-Memory-Video system. The TFB could easily interface to any 32 bit bus.

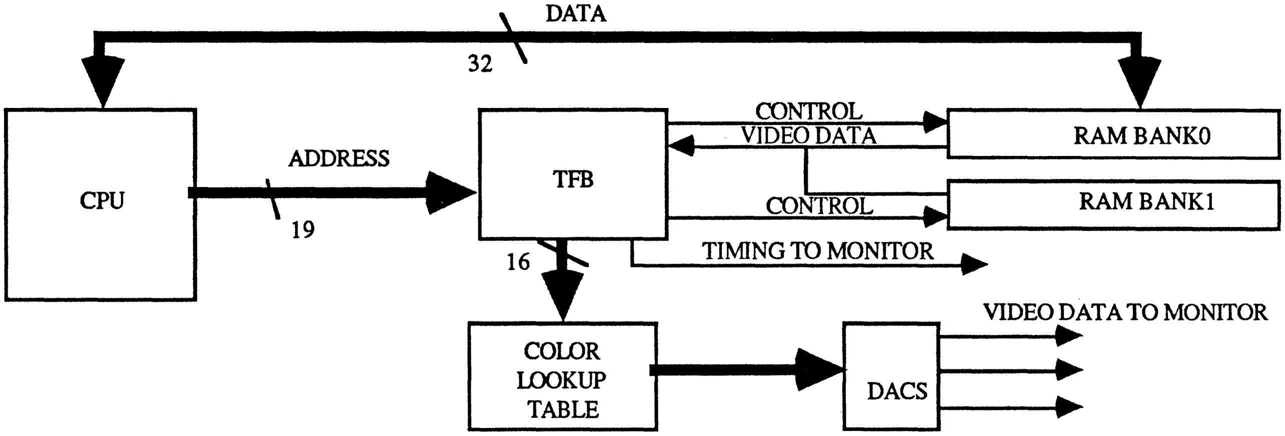


Figure 1.

The graphics/CPU system above consists of 5 major blocks:

The TFB is optimized for a 68020 bus interface, but it also has special features which support a NuBus interface. These include latched address, size and read signals, and optionally inverted address lines to accommodate the inverted sense of the NuBus.

The TFB itself is a gate array mounted in a 120-pin plastic PGA. It is expected to cost $15 in volume. This cost can be reduced by moving to a standard cell or full custom implementation.

The frame buffer memory will support either 256K or 512K bytes of dynamic RAM consisting of the NEC 256K bit video RAMs. By having a built-in shift-register and a separate serial port for video data, these RAMs allow up to 97% of the frame buffer memory bandwidth to be available to the processor. The NEC RAMs are currently packaged in a 24 pin, 400 mil DIP, and are expected to cost 20 - 30% more than standard "jelly bean" 256K parts. Several vendors are expected to supply this part in the near future in a range of packages.

No parity is provided for the frame buffer RAM. Since the parts are organized by four, 8 are required to fill out a 32-bit bus. Up to two ranks of memory can be accessed to yield a maximum RAM configuration of 512K bytes.

The color lookup table shown in the block diagram is needed for color systems. Several integrated color lookup tables and DACs are now available from various chip manufacturers.

1.3 Features

In addition to providing adequate memory bandwidth for high-resolution color images, the TFB supports a number of additional features:

- The TFB will operate with either 60Hz interlaced, RS170/NTSC compatible timing, or with 60Hz non-interlaced timing. PAL and SECAM timing should be achievable given the programable nature of the TFB's sync generation. Almost any non-standard video refresh rate can also be supported. For example, a TFB has been configured to provide a 120Hz interlaced refresh rate as well as a 80 Hz non-interlaced refresh rate.

- Merging of external video source with the frame buffer video stream is supported, though this capability is as yet untested.

- The TFB will support a wide range of screen resolutions, pixel depths, and pixel data rates up to 66 MBytes per second, all under software control. Pixel depths of 1, 2, 4,8 or 16 bits per pixel can be achieved using a single pixel clock, and no external logic. The master pixel clock on the TFB can be prescaled under software control to support any of these color depths. Pixel clocks up to 33MHz are achievable at 16 bits per pixel. Higher clock speeds can be achieved at shallower pixel depths as long as the 66MBytes per second data rate is not exceeded.

- The TFB provides a very simple processor/bus interface and is able to support processor clocks up to 15MHz. Since the TFB is currently implemented in a 2µ process, and 1.5µ processes are now becoming available, considerable speed improvement (40-50%) in the part is possible with very little effort just by moving to a smaller feature size process. Thus, future TFB's are likely to support faster processor/bus interfaces such that speed is limited by RAM technology and not the TFB itself.

- An added advantage to making the video timing fully programmable is that a single hardware configuration can support a multitude of screen resolutions, color depths, and sizes, all under software control.

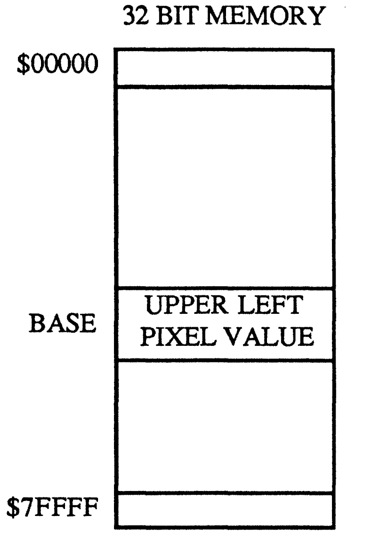
- Finally, internal syncronization of the pixel clock to the CPU clock makes it possible to decouple the CPU/bus system clock from the pixel clock. This provides more flexibility in determining what type of video is to be supported and what processor speed is chosen in a tightly coupled CPU-Memory-Video system.

2.0 DATA ORGANIZATION

One requirement of the TFB design was that, other than its use of the chunky color model, it should not change the fundamental way in which the frame buffer is viewed by the programmer. The traditional Apple frame buffer model is as a contiguous array of bytes. The memory map below shows the layout of the various frame buffer memory spaces, and how they relate to the screen image.

	8 BIT REGISTERS
$0000	LENGTH(7:0)
$0004	RFSH(2:0), INTERLACE, GENLOCK, SETUP(2:0)
$0008	BASE(15:8)
$000C	BASE(7:0)
$0010	SYNCINTERVAL(7:0)
$0014	VLINES(2:0),VFP(4:0)
$0018	VLINES(10:3)
$001C	VBACKPORCH(7:0)
$0020	SYNCINTERVAL8,VEQUAL(6:0)
$0024	HSYNCSTART(7:0)
$0028	HSYNCFINISH(7:0)
$002C	HALFLINE(7), HEARLY(6:0)
$0030	HALFLINE(7:0)
$0034	HPIXELS(1:0), HLATE(5:0)
$0038	HPIXELS(9:2)
$003C	RESET~, BASE(16), DEPTH(2:0),POLARITY,LENGTH(9:8)
	UNUSED
$1000 $1FFF	COLOR LOOKUP TABLE

CONTROL SPACE MEMORY MAP



RAM MEMORY MAP

Table 1.

The control address space is independent of the RAM data space for the frame buffer. The first 64 bytes of this space are reserved for control registers. Control registers are 8 bits wide, and located at every 4th address in the control space starting at address 0. A complete description of the control registers is given in section 3.0.

The frame buffer memory is a 32 bit wide, 256K-512K byte linear address space in which pixels are organized in a "chunky" manner. That is, a pixel's color value is determined by contiguous bits in memory rather than separate bit planes. The number of longwords of data in a horizontal scan line for the frame buffer is fully programmable, as is the screen's horizontal resolution. Either 256K or 512K byte memory configurations are possible.

3.0 CONTROL REGISTER DESCRIPTION

A set of 16 8-bit control registers in the TFB provides the parameters determining screen resolution, sync generation, and system configuration requred. Since some of the parameters require more than 8 bits for their definition, some registers contain bits for more than one parameter, specifically, registers 4, 14, 20, 2C, 34 and 3C. See the memory map above and the parameter descriptions below for details.

Throughout this document, a distinction is made between registers and parameters. Registers are always 8 bits wide and are located at a particular physical address as determined by the memory map above. Parameters, on the other hand, represent a collection of bits which together determine a particular function or characteristic of the TFB. Several parameters require more than 8 bits of definition, and therefore span more than one register. Some parameters require only a single bit of definition.

There is much interplay between different parameters, so determining parameter values for a particular frame buffer configuration tends to be an iterative process. Only after the configuration parameters are determined should the register values be set. This avoids the confusion that can arise between the definition of a parameter and its corresponding register.

The following is a description of the parameters required to configure the TFB. For each parameter the register bit locations are given as:

<parameter name>(<bit range>) <register address>:<bit range> <schematic name>.

3.1 System Configuration Parameters

These parameters give global definitions of how and where the pixel data is to be organized, how RAM is to be refreshed, whether the TFB is running interlaced or non-interlaced and so on.

15-4

BASE(16:0) $3C:2 $08:7-0 $0C:7-0 VSLC

This parameter gives the offset, in long words, from the base of the frame buffer memory, to the upper leftmost pixel to be displayed. If the frame buffer base is set so high that there is not enough memory to display the full screen, unpredictable data will be displayed after the point at which the 512K barrier is reached. If BASE is set to start in the 2nd rank when the 2nd rank is not present, unpredictable data will be displayed.

LENGTH(9:0) $3C:1-0 $0:7-0 LNGTH

This parameter equals rowbytes/4 for the screen width, where rowbytes is defined to be the number of bytes between successively scanned lines. For a byte per pixel, non-interlaced, 640 pixel wide screen, this parameter should be set to $A0 (decimal 160). Notice rowbytes must be divisible by 4. For an interlaced display with these characteristics, this parameter is set to $140 (decimal 320), since interlace displays successively scan every other line.

RFSH(2:0) $04:7-5 STAT(6:4)

This parameter equals the number of RAM refresh cycles to be executed per scan line time. If the TFB is running with NTSC timing, then the scan lines are 63.56μsec apart. Since RAM must be refreshed every 4000μsec there will be 62.9 scan lines per refresh period. Dividing 256 rows by 62.9 scan lines gives 4.07 rows refreshed per scan line. We round up to 5, so this field should contain a 101B.

Unpredictable results will occur if this parameter is set to zero.

INTERLACE $04:4 STAT3

If this bit is set HIGH, then the TFB runs in interlaced mode; otherwise the TFB runs in non-interlaced mode. This has implications on how scan line addresses get generated as described below in section 5.2.

GENLOCK $04:3 STAT2

When HIGH, this bit indicates that the horizontal and vertical sync signals are externally generated. When in genlock mode, many of the parameter definitions change. These changes will be outlined in an appendix to this document to be added later. The genlock capability of the TFB has not yet been tested, and due to the difficulty of the problem, no guarantees are made about the adequacy of the TFB's support for genlock. Normally this bit will be set to 0.

SETUP $04:2-0 STAT10-8

This 3 bit field determines the time required to synchronize the parts of the TFB concerned with pixel generation, to the parts of the TFB concerned with RAM timing. This field is calculated as follows:

SETUP :=($FF- (20 * Trunc((CPU2X period)/(EffPXCLK period))) DIV 4) - 56

Where the EffPXCLK = (16/depth of pixels)*PXCLK period.

Clearly some black magic is going on here. The operation of this field is described in detail in section 5.2. A full understanding of this equation requires careful study of the chip's schematics and timing, and is probably not worth the effort. Suffice to say that this field is needed to assure proper synchronization between the the pixel generation and RAM control portions of the TFB.

POLARITY $3C:3 PLRITY

This parameter determines the sense of the address bus as seen by the TFB. When LOW, this parameter causes the TFB to treat the address bus as active HIGH as it would be on a 68020 bus. When HIGH, this parameter causes the TFB to treat the address bus as active LOW as it would be on a NuBus. This parameter is initially LOW when RESET~ is asserted. In a NuBus based system, care should be taken to generate the proper addresses for the TFB after a RESET~ is asserted since the sense of the address lines will be inverted from normal operation.

In addition to changing the sense of the address lines, when HIGH, this parameter causes the SIZE0, A1 and A0 signals to be interpreted as if they were the TM0~, AD1~ and AD0~ signals of the NuBus. When POLARITY is tied HIGH, these NuBus signals may be directly tied to the TFB. SIZE1 should be tied HIGH when running in this configuration.

RELATIONSHIP BETWEEN POLARITY, BYTE SELECT PINS AND WEN SIGNALS

POL-ARITY	SIZE1~	SIZE0~	A0	A1	WEN3~	WEN2~	WEN1~	WEN0~
0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0
0	0	0	1	0	1	1	0	0
0	0	0	1	1	1	1	1	0
0	0	1	0	0	0	1	1	1
0	0	1	0	1	1	0	1	1
0	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0
0	1	0	0	0	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	0	1	0	1	1	0	0
0	1	0	1	1	1	1	1	0
0	1	1	0	0	0	0	0	1
0	1	1	0	1	1	0	0	0
0	1	1	1	0	1	1	0	0
0	1	1	1	1	1	1	1	0
1	X	0	0	0	0	1	1	1
1	X	0	0	1	1	0	1	1
1	X	0	1	0	1	1	0	1
1	X	0	1	1	1	1	1	0
1	X	1	0	0	0	0	1	1
1	X	1	0	1	0	0	0	0
1	X	1	1	0	1	1	0	0
1	X	1	1	1	0	0	0	0

Table 2.

DEPTH $3C:6-4 DEPTH

This parameter determines the depth of the pixels which the TFB is to generate. This parameter will cause the TFB to prescale the pixel clock, and multiplex the pixel data so as to produce the desired pixel depth. Note that all parameters which are given in pixel times, are given in relation to the scaled pixel clock, and not necessarily the pixel clock fed into the TFB. The pixel depth determined by the DEPTH parameter is given in the table below:

DEPTH parameter value	Pixel Depth
100	1 bit per pixel
101	2 bits per pixel
110	4 bits per pixel
111	8 bits per pixel
000	16 bits per pixel

Table 3.

It is important that this parameter be initialized well before the SOFTRESET~ parameter is set HIGH so as to allow time for the pixel clock generation circuitry to stabalize before the TFB is taken out of reset mode.

Refer to Table 4 for a description of how the DEPTH parameter affects the definition of the CMA bus .

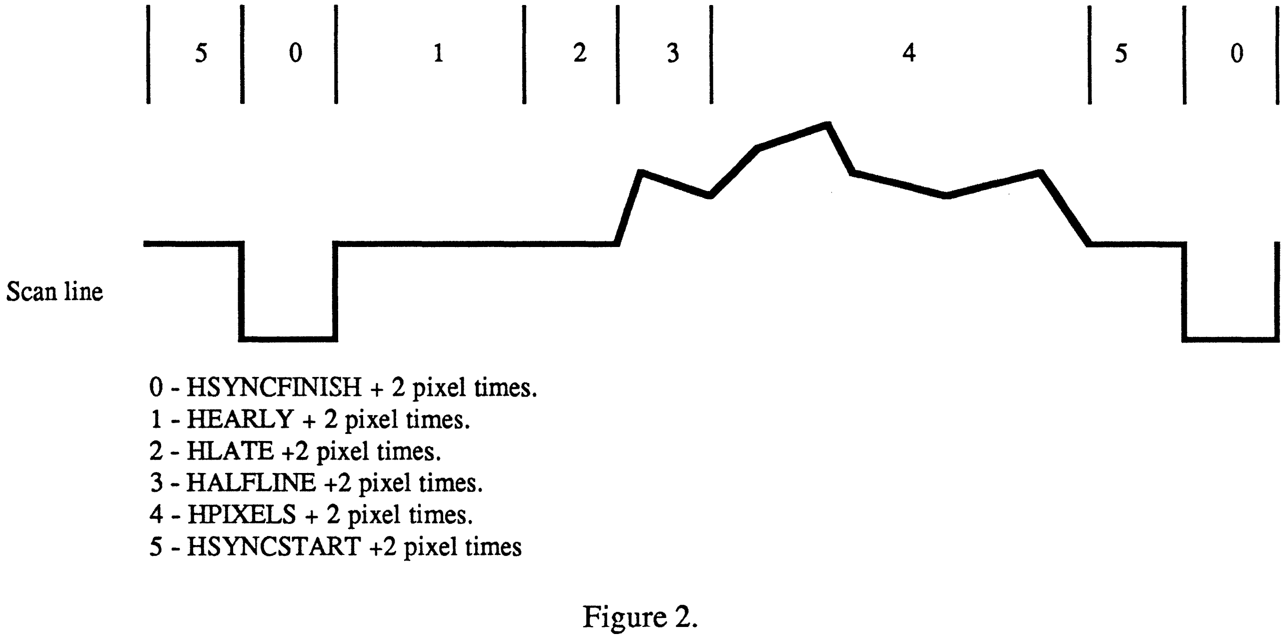
SOFTRESET~ $3C:7 STAT7

This bit is used to reset the pixel generation circuitry in the TFB. At system reset, this bit is cleared and the pixel generation circuitry of the TFB enters a reset mode waiting for SOFTRESET~ to be set. SOFTRESET~ should not be set until all of the configuration parameters are loaded with their proper values. At system reset, all configuration registers but this one are in an unknown state. No RAM refresh will take place until after SOFTRESET~ is set.

3.2 Horizontal Timing Configuration Parameters

The following seven parameters determine the timing characteristics of a single scan line. The first six parameters set the length of the various regions of the scan line. All region lengths are given in scaled pixel clock periods from the end of the last region. Each scan line is broken up into six regions which define the duration of the horizontal front porch, horizontal sync pulse, horizontal back porch, and the region of the horizontal scan line in which active video is displayed. Additionally, these first six parameters must indicate the midpoint of the scan line for use in generating the equalizing pulses found in the RS170 composite sync signal.

The figure below shows each of these regions and gives the parameter name which defines its duration.



0 - HSYNCFINISH + 2 pixel times.
1 - HEARLY + 2 pixel times.
2 - HLATE +2 pixel times.
3 - HALFLINE +2 pixel times.
4 - HPIXELS + 2 pixel times.
5 - HSYNCSTART +2 pixel times

Figure 2.

HSYNCSTART $24:7-0 HSS(7:0)

This parameter equals two less than the number of scaled pixel clocks from the beginning of region 5 above to the start of the next scan line. This is equal to the length of the horizontal front porch.

HSYNCFINISH $28:7-0 HSF(7:0)

This parameter equals two less than the number of scaled pixel clocks defining the duration of the horizontal sync pulse in region 0 above. This parameter also determines the duration of the equalizing pulses found in the RS170 composite sync signal. The equalizing pulses are equal to HSYNCFINISH/2 scaled pixel clock periods.

HEARLY $2C:6-0 HBDE(6:0)

This parameter equals two less than the length of region 1 above. The back porch is broken up into two regions. Region 2 must equal to the worst case time to finish a bus cycle, plus 3 CPU clock periods. HEARLY is equal to two less than the horizontal back porch length, less the time calculated above.

HLATE $34:5:0 HBPE(5:0)

This parameter equals two less than the length of region 2 above. This parameter, plus HEARLY described above, give the length of the horizontal back porch. Refer to section 5.2 for a complete description of the interplay of HEARLY and HLATE.

HALFLINE $2C:7 $30:7-0 HLFL(8:0)

This parameter equals two less than the length of region 3 above. The end of region 3 marks the middle of the scan line as measured from falling edge of HSYNC to falling edge of HSYNC for the next scan line. This "midpoint" determines the time at which the equalizing pulses found in the RS170 composite sync signal are to start.

HPIXELS $38:7-0 $34:7-6 HALE(9:0)

This parameter equals two less than the length of region 4 above. It is the number of pixels to be displayed from the midpoint of the scan line to the start of the horizontal front porch.

SYNCINTERVAL $24:7 $10:7-0 SINT(8:0)

This parameter has no bearing on the length of the horizontal scan line, but instead is used to determine the duration of the interval between the vertical serrations found in the RS170 composite sync signal. If composite sync is not being used, then this parameter need not be set to anything. A section of the RS170 composite sync signal is shown below:

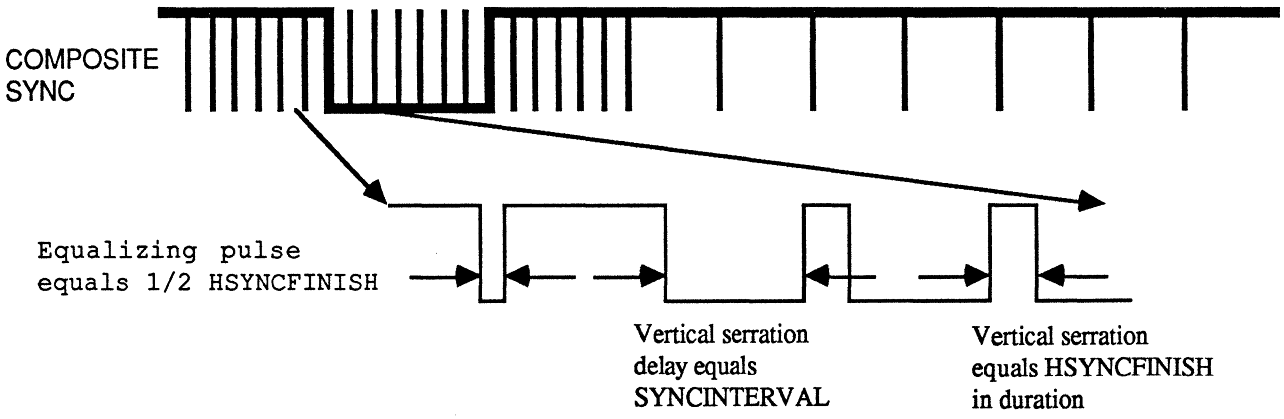


Figure 3.

The SYNCINTERVAL parameter determines the time from the middle of the scan line to the rising edge of the vertical serration in scaled pixel clock periods.

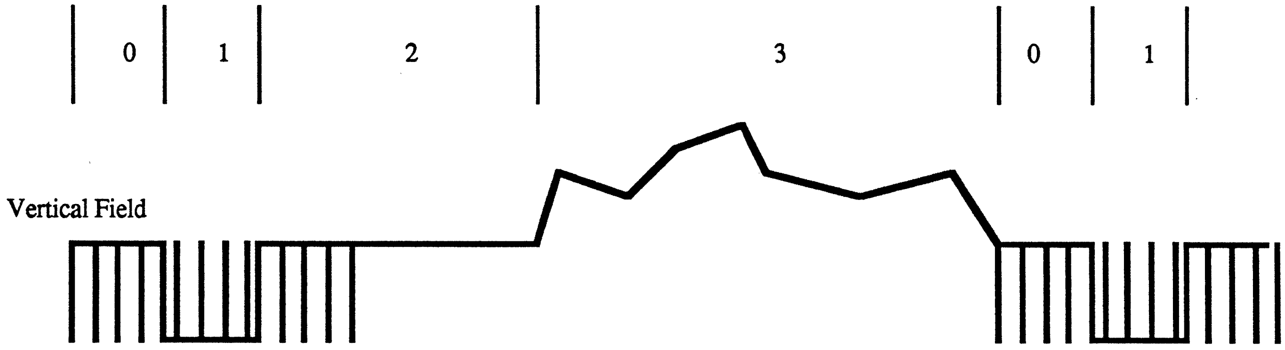
3.3 Vertical Timing Configuration Parameters

The next five parameters define the timing of the vertical sync generation circuitry. The vertical field is broken up into four regions, the duration of which is given by the four parameters below in terms of half scan lines. The four regions determine the vertical front porch, vertical sync pulse width, vertical back porch, and the number of active scan lines to be displayed.

Note that for interlaced displays, the number of half lines in a field must be odd in order to provide an offset from one field to the next. The INTERLACE parameter described earlier impacts address generation. Proper interlace timing still depends on placing interlaced compatible values in the vertical timing parameters.

In addition to defining the vertical timing, some of the following parameters have a bearing on where vertical serrations and equalizing pulses will appear in the composite sync signal. For strict adhearance to NTSC, some additional attention should be given to the lengths of the VFRONTPORCH and VSYNCFINISH parameters.

The figure below shows each of the vertical timing regions, and gives the parameter name which defines its duration.



0 - VFRONTPORCH + 1 half line time.
1 - VSYNCFINISH +1 half line time.
2 - VBACKPORCH +8 half line timeS.
3 - VLINES+1 half line time.

Figure 4.

VFRONTPORCH $14:4-0 VFP(4:0)

This parameter is equal to one less than the number of half lines in region 0 of the figure above. This is the "front porch" of the vertical field. The composite sync signal will have equalizing pulses inserted during the front porch portion of the vertical field.

VSYNCFINISH $20:6-0 VSEQ(6:0)

This parameter is equal to one less than the number of half lines in region 1 of the figure above. This is the vertical sync pulse portion of the vertical field. The composite sync signal will have vertical serations inserted for this number of half lines during the vertical sync period.

VBACKPORCH	$1C:5-0		VBP(5:0)

This parameter equals eight less than the number of half-lines in region 2 of the figure above. This is the 'back porch' portion of the vertical field.

VLINES	$14:7-5	$18:7-0	VAL(10:0)

This parameter equals one less than the number of half-lines in region 3 above. This is the active portion of the video field. For interlaced display, this parameter is set to half of the total number of lines to be scanned.

All of the TFB's parameters can be determined from just a few assumptions about the length of the scan line, the number of lines in a frame and so on. An interactive Pascal program has been written which will automate the generation of these parameters. A listing of this program is given in Appendix D.

3.4 Initialization Procedure

Before loading any values into the TFB's control registers, a soft reset should be issued to the TFB. All but the register at $3C can be be loaded in any order. Finally, the value for register $3C is loaded without taking the chip out of the soft reset state. The TFB should be taken out of the soft reset state only after all bits in all registers are in the desired state. Furthermore, the programmer should wait at lest three effective pixel clock periods between the time the last register is set, and the soft reset state is cleared.

4.0 SIGNAL DESCRIPTION

The TFB is housed in a 120 pin PGA. The following lists the TFB's signal names, pin numbers, I/O direction, current drive capability and timing information. All the outputs from the TFB are synchronous to the rising edge of either the pixel clock (PXCLK) or twice the CPU clock (CPU2X), so each output has a delay time listed from the rising edge of a particular clock. This is a worst case delay time assuming an 85 picofarad load.

One of the greatest drawbacks for CMOS is its reletively poor speed when driving signals off chip. For the TFB, this problem is particularly accute as it is trying to drive video data out of the chip at up to 33 MHz. To address this problem, the TFB provides a delayed clock output which can be used to clock the pixel data as it exits the TFB. This clock is a delayed version of the PIXCLK input to the TFB. Since most systems using the TFB will have the pixel clock going to only a few places, and most systems will have pixel data paths synchronous to the pixel clock, it is suggested that the delayed pixel clock output from the TFB be used as the system pixel clock where possible. The pixel clock oscillator should feed the TFB directly and go no place else. This will greatly ease the timing requirements for any pixel data path.

The delays listed below for PIXCLK referenced signals are referenced from the rising edge of the input PIXCLK. The delay listed for the PIXCLKOUT signal is a guaranteed minimum value. All PIXCLK referenced signals have guaranteed 10ns setup time to the PIXCLKOUT signal.

The format for the signals is as follows:

<pin name> <pin> <I/O> <drive> <clock> <delay>

For inputs which are synchronous, the <clock> field identifies what clock the signal is synchronous to, the <edge> field identifies the synchronizing edge of that clock, and the <delay> field identifies the minimum setup time required by that signal.

4.1 Inputs

RESET~ A9 INPUT N/A Both N/A

When RESET~ is asserted, the TFB is initialized to a known state. Many of the control parameters are cleared or set when RESET~ is asserted, so the control registers should be assummed to be in a random state.

The bus interface logic is initialized to a wait condition. No refresh of memory will take place until the video is enabled via the SOFTRESET parameter bit described above, after the RESET~ signal is asserted.

RESET~ is internally synchronized to both PIXCLK and CPU2X, so RESET~ must be asserted for at least five cycles of the slower of PIXCLK and CPU2X.

PIXCLK B5 INPUT N/A N/A N/A

Pixel data and the video sync signals are valid soon after the rising edge of PIXCLK. This signal is independent of the CPU clock. If video merge between the frame buffer and an external video source is desired, external circuitry must make sure that the external horizontal and vertical sync, and pixel clock are in phase. The minimum pixel time supported by the TFB is 30ns at 16 bits per pixel. For color depths less than 16 bits per pixel, the pixel rates supported scale linearly from 30ns/pixel.

The minimum pulse width for PIXCLK is 15ns.

CPU2X N12 INPUT N/A N/A N/A

The TFB's bus interface is optimized to work with a 68020 style bus. CPU2X is the main clock used in the frame buffer and has a frequency twice that of the clock driving the 68020. This clock is independent of the pixel clock and has a maximum frequency of 29MHz. The rising edge of this signal triggers all edges of the main CPU clock.

For operation on the NuBus, a 20 MHz clock can easily be derived from the 75/25% duty cycle Nubus clock. This 20 MHz clock should be used as the CPU2X clock into the TFB.

The minimum pulse width for this signal is 15ns.

RAMSEL~ L13 INPUT N/A CPU2X 0 ns

When RAMSEL~ is asserted, the TFB starts a RAM cycle. Addresses must be valid 20 ns before the next CPU clock edge following the assertion of RAMSEL~. For systems in which the TFB is tightly coupled with a 68020, RAMSEL~ can be a simple decode of the 68020 addresses. Glitching on RAMSEL~ is allowed as long as it is stable at the rising edge of CPU2X when PAS~ is asserted.

PAS~ N11 INPUT N/A CPU2X 0 ns

Accesses to the RAM address space must be started as early in the bus cycle as possible to get maximum performance from the RAM. This early start of the RAM cycle is facilitated by having two signals select the RAM address space. The RAMSEL~ signal described above is ignored unless PAS is asserted. PAS~ has compatible timing to that provided by the PAS~ signal coming from a 68020 or PMMU.

In addition to strobing the address lines into the TFB, the PAS~ signal also indicates to the chip when DTACK~ can be negated. DTACK~ is negated as soon as PAS~ is negated to indicate the termination of a cycle. This has important implications for anyone designing the TFB into a NuBus based system. The temptation will be to use START~ as a PAS~ signal, but since START~ is negated after the first cycle of the transfer, the DTACK~ signal will be inhibited from the TFB. The TFB PAS~ signal must be derived from the START~ signal such that it allows recognition of DTACK~ for generating the NuBus ACK~ signal.

TESTEN~ L6 INPUT N/A N/A N/A

When asserted, TESTEN puts the TFB into test mode. This mode can be used to examine some internal state of the TFB. During normal operation, this input must be tied HIGH.

READ/WR~ M9 INPUT N/A N/A N/A

When this signal is low at the time PAS~ is asserted during a RAM access, a write cycle is executed. Otherwise, a read cycle is executed. Note that this signal is internally latched as PAS~ is asserted to facilitate interface to the Nubus.

SIZ(1:0) N6,N7 INPUT N/A N/A N/A

When the POLARITY parameter is LOW, these pins have compatible functionality and timing to SIZ0,1 on the 68020. They indicate the number of bytes yet to be transferred in the current bus transaction.

When the POLARITY parameter is HIGH, these pins have compatible functionality and timing to TM0,1~ on the NuBus. Refer to Table 2 for a description of how these pins affect what byte lanes are written to when accessing the TFB RAM space.

Note that these signals are internally latched as PAS~ is asserted.

CTLSEL~ B9 INPUT N/A N/A N/A

When CTLSEL~ is asserted the control registers are selected for a write operation. Control registers are write only. CTLSEL~ also acts as a data strobe signal. Data is latched into the control registers on the rising edge of CTLSEL~.

A(18:0) C INPUT N/A N/A N/A

A(18:0) are the address bits used for accessing RAM and the control registers. These signals are latched as PAS~ is asserted. The polarity of these signals is determined by the POLARITY parameter described above. If the POLARITY parameter is LOW, then the addresses are considered by the TFB to be active HIGH as they would be in a 68020 system. If the POLARITY parameter is HIGH, then the addresses are considered by the TFB to be active LOW as they would be in a NuBus system.

D(7:0) C INPUT N/A N/A N/A

The TFB has an 8-bit data port through which the control registers may be written. Since the control registers are all mapped to 4 consecutive bytes (A(1:0) are not decoded), the TFB may be placed on any byte lane of the bus.

VD(31:0)	C	INPUT	N/A	N/A	N/A	

VD is the video data to be clocked from the video RAMs onto the chip.

4.2 Outputs

DTACK~	K12	OUTPUT	4mA	CPU2X	RISING	42ns

DTACK~ is an output asserted during state S2 of a 68020 or state S4 of a 68000 bus cycle. DTACK~ is asserted for all accesses to the control address space, and for accesses to the RAM space which are validated by PAS~ as described above. Since all accesses to the TFB or its associated memory are 32 bit accesses, only one data acknowledge signal is needed.

RAS(1:0)~	H11,H12	OUTPUT	8mA	CPU2X	RISING	40ns

Each of the RAS signals is designed to drive 8 RAM chips without buffers. Minimization of capacitance on the RAS and CAS lines leading from the TFB should be a top priority of any pcboard layout. RAS1~ selects the high 256K bytes in the RAM address space, while RAS0~ selects the low 256K bytes.

CAS(1:0)~	N9,L8	OUTPUT	8mA	CPU2X	RISING	30ns

Like RAS, each CAS signal is designed to drive 8 RAM chips without buffers. CAS1~ selects the high 256K bytes in the RAM address space, while CAS0~ selects the low 256K bytes.

DTOE(1:0)~	M8,N8	OUTPUT	4mA	ASYNC	N/A	40ns

During RAM read cycles, DTOE~ selects which bank of memory is to be enabled onto the system bus. If DTOE~ is asserted before RAS~ is asserted, then a data transfer cycle occurs which loads the shift register on the selected video RAM chip.

Each DTOE signal is able to drive 8 RAM chips without buffers. DTOE1~ selects the high 256K bytes in the RAM address space, while DTOE0~ selects the low 256K bytes.

WEN(3:0)~	C	OUTPUT	4mA	CPU2X	FALLING	50ns

These signals select which byte in a long word is to be written during RAM cycles. WEN3~,WEN2~,WEN1~,WEN0~ select the high (D31-24) upper middle (D23-16) lower middle (D15-8) and low (D7-0) bytes respectively.

Refer to Table 2 for a description of how the POLARITY parameter, and the byte select pins affect these signals.

SCLK	G2	OUTPUT	4mA	PXCLK	RISING	45ns

SCLK clocks the video data from the RAMs to the TFB. Each SCLK signal is capable of driving 8 RAM chips at full speed. Buffering of SCLK is recommended for any system using the 16 bit per pixel mode of operation, otherwise buffering is not necessary.

SOE(1:0)~	F2,F3	OUTPUT	4mA	PXCLK	RISING	35ns

SOE selects which bank is to put its data onto the VD pins of the TFB.

PCLKOUT~ F1 OUTPUT 4mA PXCLK N/A 10ns MIN

This is a delayed version of the PIXCLK input which can be used to ease the timing constraints of the CMA bus and the video sync signals.

RADD(7:0) C OUTPUT 4mA ASYNC N/A N/A

These are the RAM address bits provided by the TFB. Addresses come from either the refresh address generation circuitry on the TFB, the next scan line address generation circuitry on the chip, or the CPU addresses presented during a RAM access cycle.

CMA(16:0) C OUTPUT 4mA PCLKOUT FALLING 30ns

Pixel values are shifted through the CMA bits once every rising edge of PXCLK. Unpredicatable values are clocked out when CBLANK is high.

In order to facilitate use of the TFB in very high performance systems, the CMA bus generates pixel data in a rather unique manner. By multiplexing the 16-bit CMA bus down to 8 bits, the pixel generation speed of the TFB can be effectively doubled. When running in this configuration, the TFB can support up to 66MHz video rates from 1 to 8 bits per pixel. This process is facilitated by placing certain CMA bits onto more than one pin as per the following table:

CMA PINS

DEPTH	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
16BPP	PX15	PX14	PX13	PX12	PX11	PX10	PX09	PX08	PX07	PX06	PX05	PX04	PX03	PX02	PX01	PX00
8BPP	PX07	PX06	PX05	PX04	PX03	PX02	PX01	PX00	PX03	PX02	PX01	PX00	X	X	X	X
4BPP	PX03	PX02	PX01	PX00	X	X	X	X	PX01	PX00	X	X	X	X	X	X
2BPP	PX01	PX00	X	X	X	X	X	X	PX00	X	X	X	X	X	X	X
1BPP	PX00	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 4.

The table above shows the pixel value at each of the CMA pins for each of the possible programmed depths. An X entry indicates an unpredictable value.

HSYNC~ E2 OUTPUT 4mA PCLKOUT FALLING 30ns

This is a programmable horizontal sync signal. HSYNC~ is bidirectional so that the TFB can synchronize to an external video source. The direction of this signal is determined by the GENLOCK parameter described above.

VSYNC~ M6 OUTPUT 4mA PCLKOUT FALLING 30ns

This is a programmable vertical sync signal. VSYNC~ is bidirectional to allow genlocking. The direction is determined by the GENLOCK parameter described above.

CSYNC~ E1 OUTPUT 4mA PCLKOUT FALLING 40ns

This is an RS170 compatible composite sync signal.

CBLANK~ D1 OUTPUT 4mA PCLKOUT FALLING 30ns

This is an RS170 compatible composite blank signal.

5.0 BUS OPERATION

Three kinds of accesses can be made through the TFB to the video RAM:

- RAM cycle
- Video register data transfer cycle
- RAS only refresh cycle

5.1 RAM Cycle

For clarification of the RAM cycle, refer to the timings located in Appendix B. Notice that in the timings, the TFB states shown follow a gray code and lag the 68020 state by 2 states. For the purposes of this discussion, all references to states are references to 68020 states.

A read cycle is initiated when RAMSEL~ is asserted. Addresses must be valid at the time RAMSEL~ is asserted. Soon after the assertion of RAMSEL~, RAS~ for the proper bank according to the value of A18, is asserted. Two CPU2X clocks after the recognition of RAMSEL~, CAS is asserted to the RAM chips. On writes, the proper WEN signals will be asserted only if PAS~ is asserted by the beginning of S3. Likewise, PAS~ must remain asserted long enough for DTACK~ to be recognized by the processor (negation of PAS~ causes negation of DTACK~).

5.2 Data Transfer Cycle

The data transfer cycle is executed to set up the shift register found inside the NEC video RAMs for putting out the proper pixel data for display on the CRT. Data transfer cycles are requested by the pixel state machine part of the controller and have higher priority than CPU accesses so that, at times, CPU accesses will be forced to wait for a data transfer cycle to complete. At most, two data transfer cycles will be executed during a single scan line.

One data transfer cycle is always performed during horizontal sync. This instance of the cycle takes 10 CPU2X cycles to complete. Since no pixels are being sent to the CRT during this time, no synchronization is required between the RAM state machine and the pixel state machine. The horizontal sync data transfer cycle sets the shift register up to begin outputting data at the start of the next scan line.

Calculation of the start of the next scan line in memory is fairly complicated. At the start of each scan line, the longword address is calculated by summing together two out of five possible numbers:

<Next Scan Line Address> = (BASE or <previous scan line address>) +
(4*LENGTH or 8*LENGTH or 0)

If the next scan line is the start of a new field, then BASE is selected as the first addend; otherwise the previous scan line address is taken.

If the next scan line is the start of a new field, and the INTERLACE parameter is set, then 8*LENGTH is selected as the second addend.

If the next scan line is the start of a new field, and the INTERLACE parameter is set, and the current field is 'odd' (where the fields alternate between even and odd), then zero is selected as the second addend.

Finally, if the INTERLACE parameter is not set, then 4*LENGTH is selected as the next addend.

Often, a second data transfer cycle will be executed during the active portion of the scan line. Very careful synchronization between the bus controller and the pixel controller is required for this access so that pixel data being sent to the CRT is not disrupted. This means the exact time for this access to complete cannot be predicted.

As pixel data is scanned out, a counter on the TFB keeps track of which 32-bit longword in the shift register is currently being read out. If this 8-bit counter reaches $FF during the active portion of the scan line, then a data transfer cycle must be executed to reload the shift register. The data transfer cycle needed for reloading the shift register can be a lengthy one. As the shift register counter nears $FF, the TFB must detect the coming end of the shift register and request a data transfer cycle be started by the RAM state machine. This request must come early enough to allow any CPU access to memory to complete and to allow the data transfer cycle to nearly complete just as the last longword is read from the shift register. Since the ratio of the PXCLK period to the CPU2X period can have a very wide range depending upon the application, the setup time required by this data transfer cycle is somewhat programmable.

The setup time required for any data transfer cycle is a minimum of 24 CPU2X clock periods. When the SETUP parameter is equal to bits 2, 3 and 4 of the shift register counter, and bits 5,6 and 7 of the counter are high, a data transfer cycle is requested. For 16 bits per pixel operation in which PXCLK equals CPU2X, 24 PXCLK periods are required to have enough setup time for the data transfer, so SETUP is set to 2.

5.3 Refresh Cycle

After the data transfer cycle during each horizontal sync period, the TFB executes a number of refresh cycles to the RAM. Since horizontal scanning frequencies will vary from system to system, the number of refresh cycles executed is programmable and is given by the 3 bit RFSH parameter. Refresh addresses are generated on the TFB from a shift counter. The refresh cycles are CAS before RAS type cycles.

6.0 FUTURE DIRECTIONS

The TFB was designed as a research vehicle for quickly prototyping a wide range of frame buffer configurations. Like any design, the TFB has made tradeoffs between functionality and cost. In this its first design, most of these tradeoffs were made in favor of flexibility and functionality. Subsets of the chip (such as a black and white only TFB) can be turned around very quickly with considerable cost reduction. Several development efforts within Apple could benefit greatly from the speed with which the TFB could allow prototyping to continue.

It is clear, however, that the TFB has its limitations. While the TFB solves the problem of memory bandwidth associated with deep color frame buffers, it does not address the problem of performance of shallow color graphics in a deep color frame buffer. This is an area of ongoing research and is likely to require a radically different hardware approach from that of the TFB.

Some improvements could be made in the TFB without changing its architecture:

- Expand the addressability of the TFB from 512K to at least 1MB. This was not done in the original TFB due to pin count restrictions. Double buffering of video is desireable so more than 1MB of video RAM could be used effectively.

- Run the TFB processor interface off the CPU clock rather than twice the CPU clock. Since CMOS has problems running much above 30MHz, restructuring the TFB to run of the CPU clock would help in supporting 20MHz and faster processors.

- Support jelly bean RAMs and especially nibble mode RAMs. Processors such as the IP20 will support nibble mode accesses to speed cache fills.

7.0 CONCLUSIONS

The TFB has integrated much of the random control logic typically found in Apple 680xx products so that prototyping of entire systems can be done quickly and with few parts.

The TFB has provided yet another piece of evidence for the claim that design automation is here, now. The tools required for gate array design are well developed, and in place.

APPENDIX A

One of the central goals of the TFB design was for the frame buffer controller to maintain flexibility. At first glance, it is easy to overlook the many possible frame buffer configurations which the TFB will support, and how easy it is to design the TFB into almost any system. This appendix describes how to incorporate the TFB into two very different systems. We will illustrate the different aspects of designing with the TFB by drawing from two designs which have used the TFB.

The first design is a daughter card (known as YAD) for the YACC which reimplements the YACC's CPU, memory and video systems to use the TFB. The YAD provides the YACC with variable depth 1,2,4 or 8 bit per pixel chunky color.

The second design is a NuBus graphics card for the Milwaukee which uses the TFB. This design also supports variable depth color in addition to black and white. This card also supports a 17 inch display.

A.2.0 The Design Strategy

There are three areas of concern when integrating the TFB into any system:

- How will the TFB talk to the processor?
- How will the pixel output of the TFB be interpreted?
- How will the TFB be programmed?

A.2.1.0 Processor/Bus Interface

The TFB is optimized to be tightly connected with either a 68020 CPU or the NuBus. In either case, a minimal amount of external circuitry is needed to hook up the TFB.

Connecting the TFB to a 68020

Two signals must be generated to hook the TFB with a 68020. These are the selects for the two address spaces seen by the TFB. The RAMSEL~ signal can be a simple decode of the upper CPU addresses. They may glitch, but must be stable at the CPU2X edge at which PAS~ from the CPU is asserted.

CTLSEL~ indicates that the TFB control registers is being written to in the current cycle. The assertion of this signal is all that is needed for a control space access to take place. For accesses to the control registers, the data is latched at the rising edge of CTLSEL~ so this signal must also act as a data strobe.

The PAS~, DTACK~, READ, SIZE1, SIZE0 and RESET~ signals are all compatible with those found in a 68020 system.

The YAD schematics show that the RAMSEL~ and CTLSEL~ signals are both generated from a decoding PAL which was needed to decode other address spaces anyway. Note that the TFB does not require any timing relationship between the processor clock and the pixel clock.

Connecting the TFB to the NuBus

The NuBus interface is slightly more complicated than the 68020 interface because the PAS~ signal must be derived from the NuBus START~ signal and the DTACK~ signal must be messaged into the NuBus ACK~ signal. Additionally, doubling the NuBus clock would double the performance of such a NuBus graphics card. The schematics and PAL equations of ADG's NuBus graphics card are provided at the end of this document to show one way of implementing these signals.

From START~ is derived PAS~ to strobe the addresses into the TFB. START~ cannot be used directly because the TFB requires that PAS~ remain asserted until it is ok for DTACK to be negated. Also, the negation of PAS~ causes the address latches on the TFB to be transparent, so early termination of PAS~ would result in incorrect addresses being presented to the RAM chips. PAS~ must remain asserted until 20ns after CAS~ has been asserted to the RAM chips.

The timing of DTACK~ to ACK~ also is not direct. DTACK~ from the TFB comes very early and so it must be delayed before ACK~ can be generated. (Except, of course, for writes which can DTACK~ immediately.

Since 10MHz is well below the maximum operating frequency of the TFB, the NuBus graphics card doubles it to create an effective 20MHz rate. This is done through the use of a delay line. The TFB can handle an asymetric clock with no problem, so this 20MHz clock is quite acceptible.

Again note that the TFB does not require any synchronization between the NuBus clock and the pixel clock. Any synchronization is implicit in the DTACK signal.

A.2.2 Pixel Data Generation

The TFB generates video data at a rate of 16,8,4,2 or 1 bit every PXCLK period depending on the DEPTH parameter. It is the job of the pixel generation circuitry to interpret these bits in whatever way is appropriate for the type of video required.

A system may require a pixel rate which exceeds the 33MHz limit of the TFB. In this case, a minimal amount of external multiplexing circuitry allows you to trade off the pixel depth with the pixel rate. By multiplexing the 16 bits of TFB pixel data into 8 bits of pixel data, you can effectively double the 33MHz pixel rate limit (assuming the external multiplexors can switch at 66 MHz). This is done in the NuBus graphics card design so that just a couple of FTTL parts allows very high speed monitors to be supported.

As described in section 4.2 of the specification, the TFB assists in multiplexing the pixel data outside the TFB by placing relevent data on what would normally be unused pins of the TFB. This is described more thoroughly in section 4.2.

A.2.3.0 TFB Parameter Values

The values to be placed in the TFB parameters is, in general, anything but straight forward. Typically these value are rarely changed, and need not be calculated dynamically. Since there are many possible TFB configurations depending on pixel depth, pixel clock frequency, monitor scan rate etc., we will describe the parameters needed to configure the TFB for just one possible configuration.

In a 1 bit per pixel, 640X480 configuration, the parameter settings are as follows:

2 5 80
BASE = $1DA80
20000

This frame buffer configuration requires 38400 bytes of memory. We will assume a 512K byte system, and place the frame buffer in the high 38400 bytes of memory. Notice, BASE gives a longword address.

LENGTH = $014

Rowbytes in this example is 80. LENGTH equals 80/4 = 20 or 14.

RFSH = 3

The reasoning for this number was given in section 3.1.

INTERLACE = 1

We are running a non-interlace display

GENLOCK = 0

We generate our own syncs.

SETUP = 7

Assume a 35ns pixel clock. This is multiplied by 16 for 1 bit per pixel so the effective pixel clock period is 560ns. Assume a 70 ns CPU clock period, then by the equation in section 3.1:

SETUP := ($FF-Trunc (20 * 1/8) DIV 4) - 56 = 7

POLARITY = 1

Assuming these parameters are for a NuBus graphics card, this bit is set to indicate that the address lines are inverted and latched.

DEPTH = 100B

We are running 1 bit per pixel.

SOFTRESET~= 1

This bit should be set only when all other parameters are loaded.

The horizontal line is made up of 907 pixel times of which, the active scan line makes up 640 pixels. The rest of the pixels are divided between the sync pulse, front porch and back porch.

HSYNCSTART = $20

Horizontal back porch is 2.5 µsec long (71 pixel clocks).

HSYNCFINISH = $20

Sync duration is 2.5 µsec long (71 pixel clocks).

HEARLY = $77

First part of front porch is 121 pixel times.

HLATE = $02

Last part of front porch is 4 pixel times.

HALFLINE = $FC

Time to half line is 254 pixel times from end of front porch.

HPIXELS = $180

PXCLKs to end of line is 386.

SYNCINTERVAL = $1A4

This makes the vertical serration come just 2.35 µsec before the half line.

VFRONTPORCH

VEQUAL = 5

We need 3 equalizing pulses on either side of the vertical sync.

VBACKPORCH = $

We need 40 blanked lines.

VLINEŜ = $36F

VLINES equals one less than twice the total number of lines displayed in a frame:

480X2-1 = 959 -> $36F

VEQUAL = 5

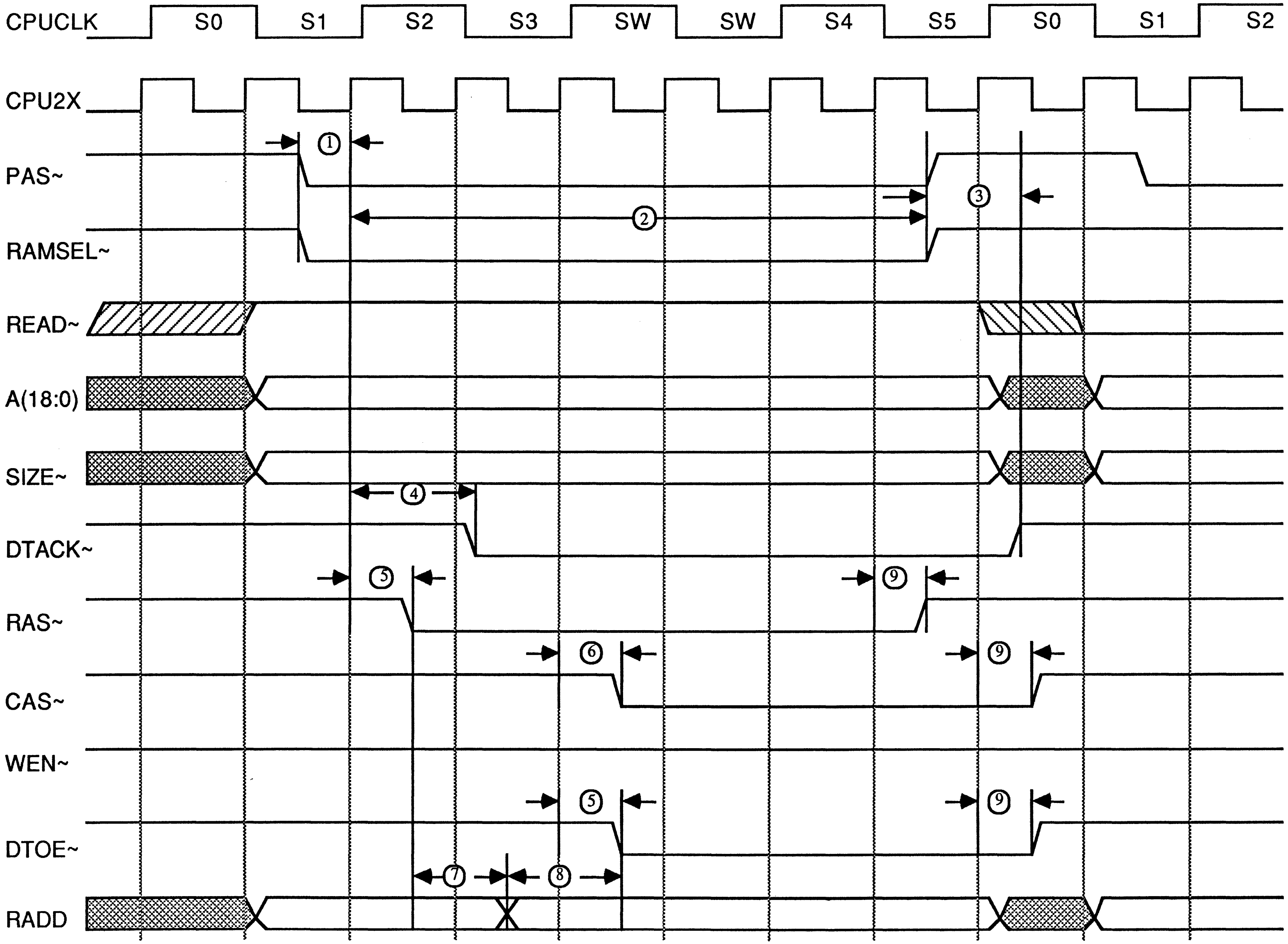
We need 3 equalizing pulses on either side of the vertical sync.

Whew, and that's all there is to it. There will soon be a simple program which will query the user about the system configuration and then generate the proper TFB parameter and register values.

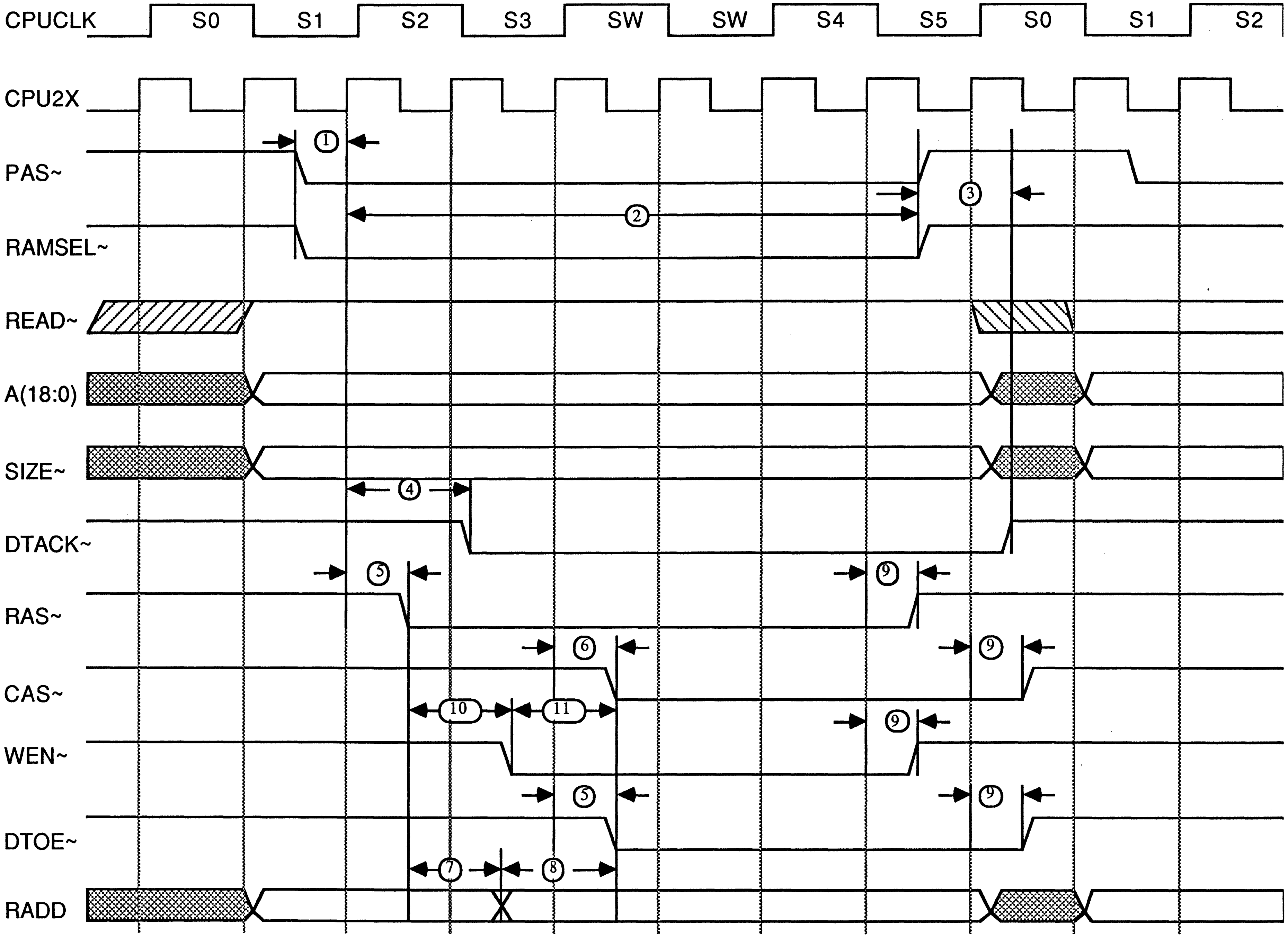
APPENDIX B
ELECTRICAL SPECIFICATIONS

NUM	DESCRIPTION	MIN	MAX
1	PAS~,RAMS~ to CPU2X setup time	0ns	–
2	PAS~,RAMS~ hold time after CPU2X	30ns	–
3	PAS~ negated to DTACK~ negated	–	26ns
4	CPU2X rising to DTACK asserted	–	42ns
5	CPU2X rising to RAS~, DTOE~ asserted	–	40ns
6	CPU2X rising to CAS~ asserted	–	30ns
7	RADD hold time after RAS~	1 CPU2X	–
8	RADD setup time to CAS~	10ns	–
9	CPU2X rising to RAS~,CAS~,DTOE~ negated	30ns	–
10	RAS~ to WEN~ delay	1 CPU2X	–
11	WEN~ to CAS~ setup time	10ns	–
12	CAS~ to RAS~ setup for refresh cycle	1 CPU2X	–
13	DATA hold time after CTLSEL~ negated	30ns	–
14	SOE0~ negated to SOE1~ asserted	1/2 PXCLK	–
15	CBLANK~,CSYNC,HSYNC~,VSYNC~, CMA setup time to PCLKOUT~ falling edge	10ns	–
16	CBLANK~,CSYNC,HSYNC~,VSYNC~, CMA hold time after PCLKOUT~ falling edge	10ns	–

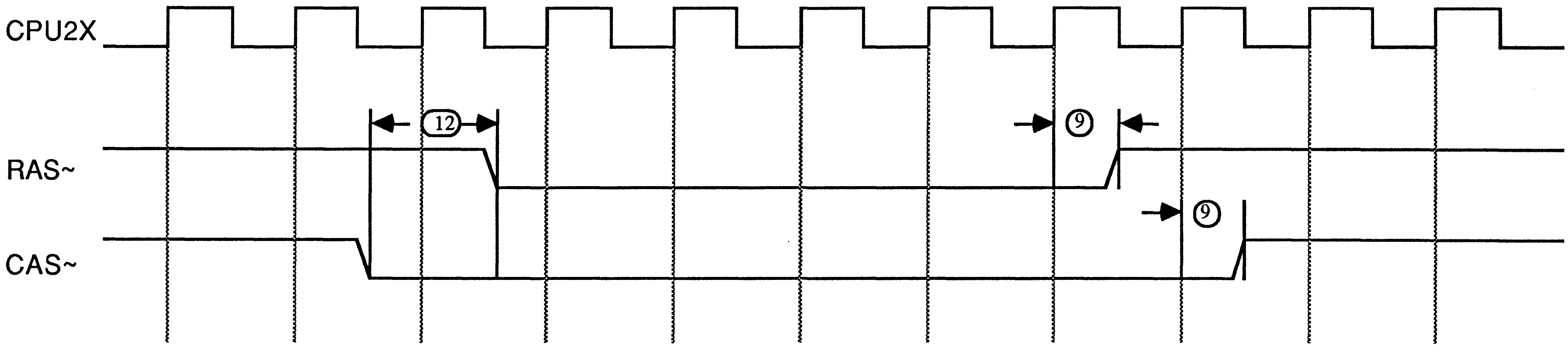
TFB RAM READ CYCLE



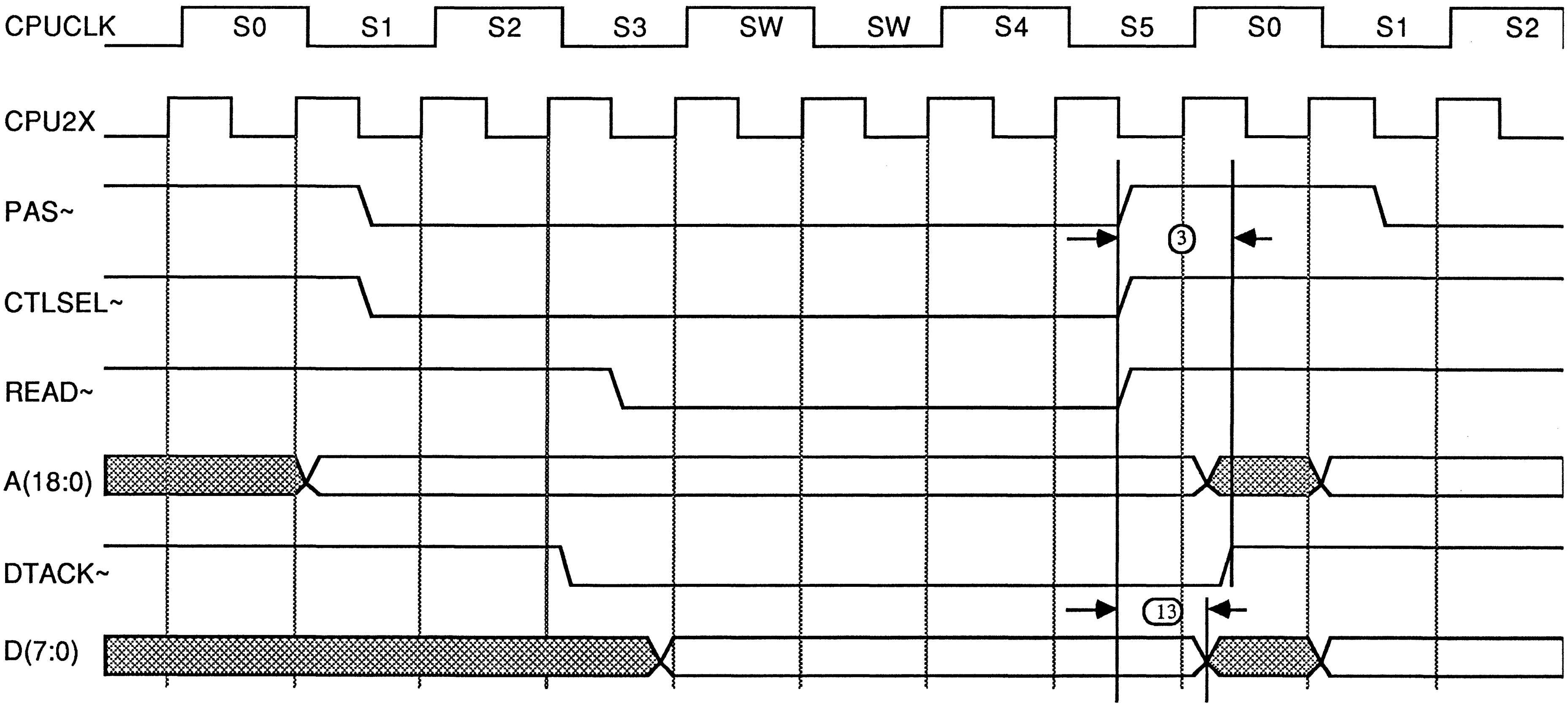
TFB RAM WRITE CYCLE



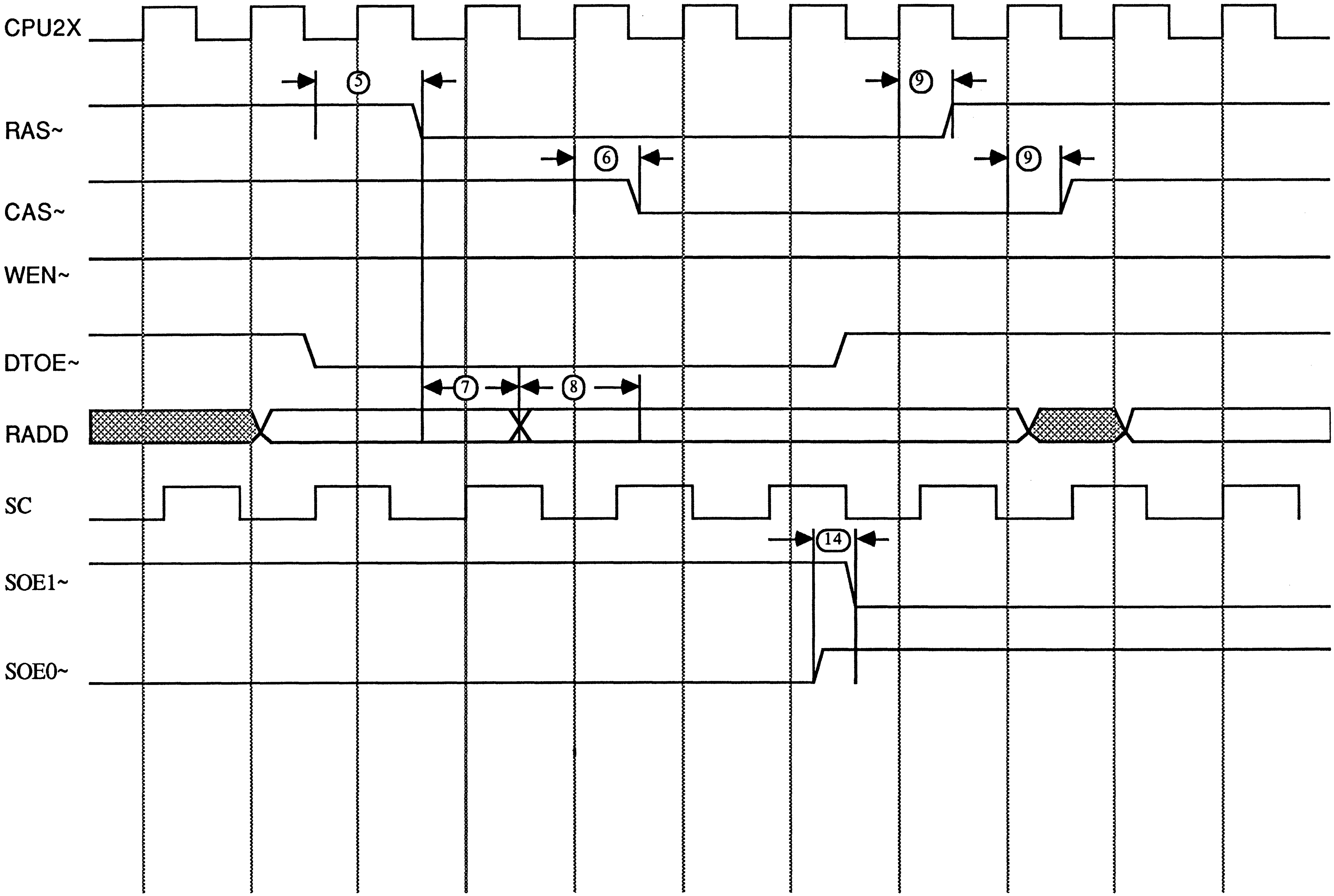
TFB RAM REFRESH CYCLE



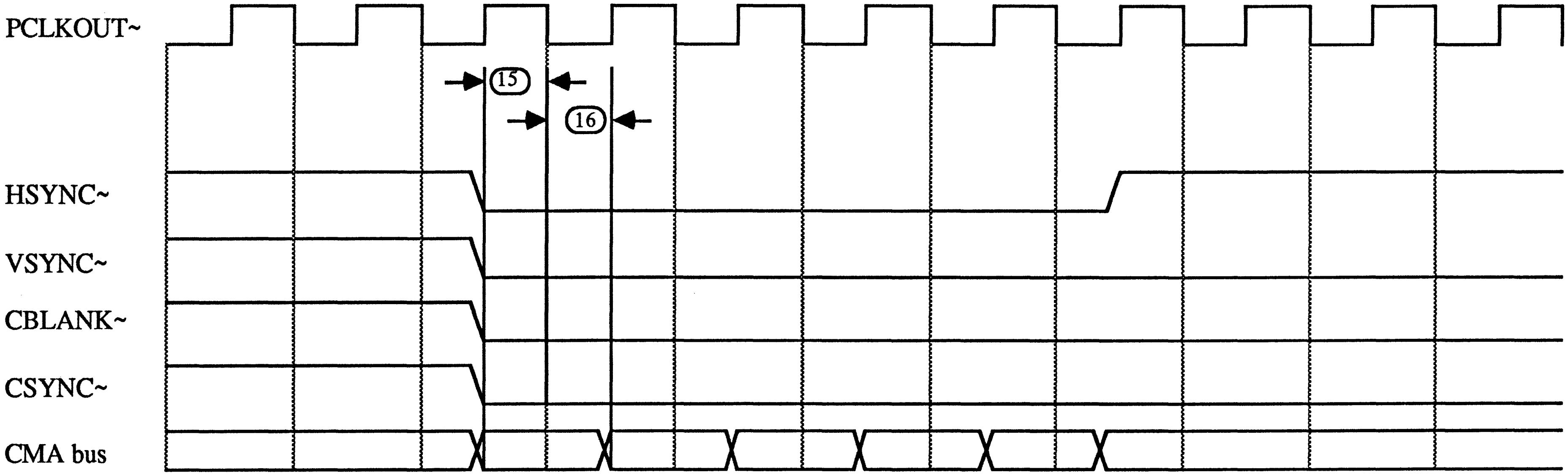
TFB CONTROL SPACE WRITE CYCLE



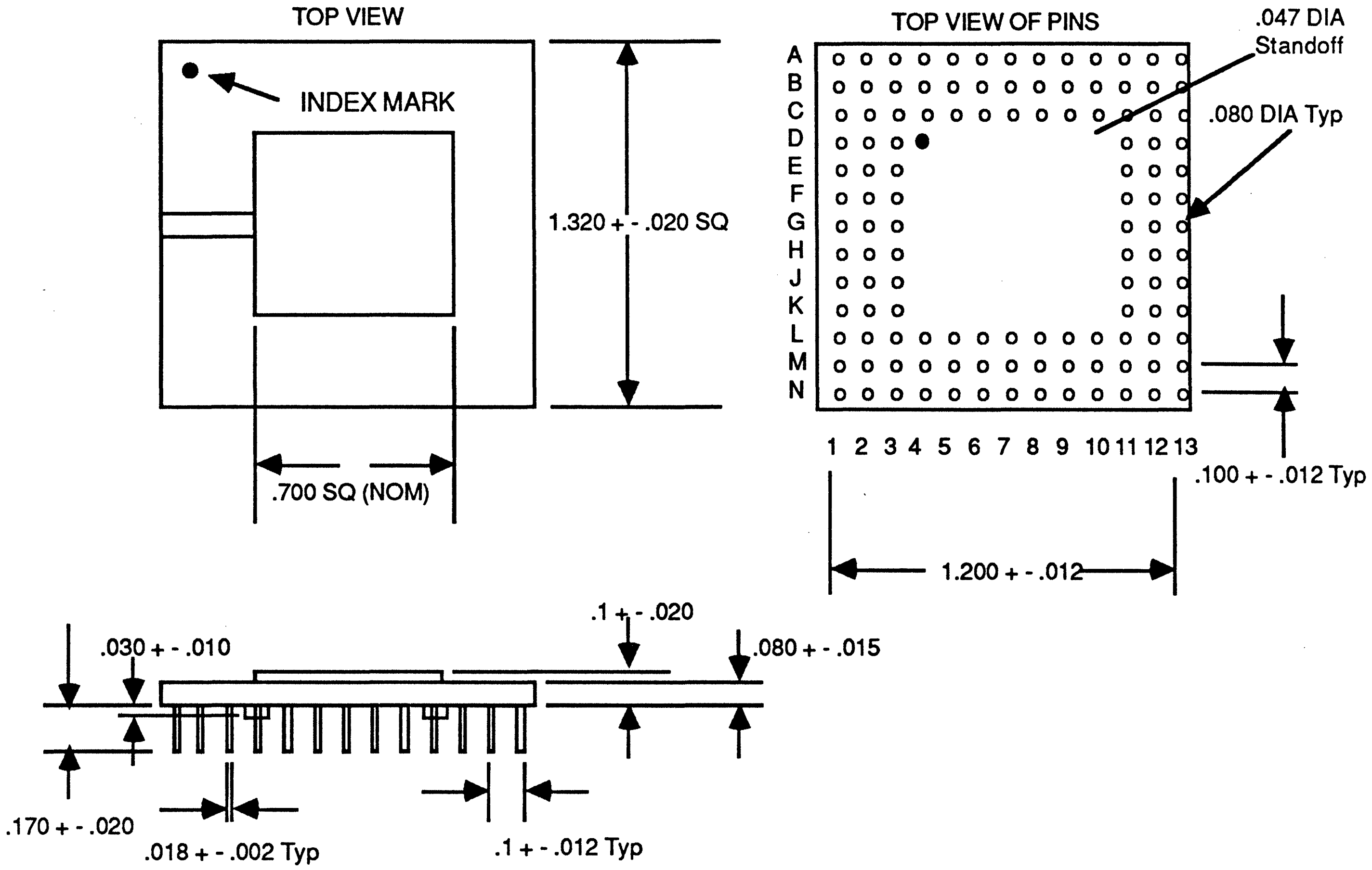
TFB RAM DATA TRANSFER CYCLE



TFB SYNC TIMINGS



APPENDIX C
MECHANICAL INFORMATION



PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL
NO.	NAME	NO.	NAME	NO.	NAME	NO.	NAME	NO.	NAME
A1	VDAT31	B12	VDAT16	E11	VDAT00	J11	WEN3~	M3	A08
A2	VDAT28	B13	VDAT03	E12	CMADD01	J12	WEN1~	M4	A06
A3	VDAT27	C1	VDAT10	E13	CMADD02	J13	WEN0~	M5	A03
A4	VDAT24	C2	VDAT11	F1	SERCLK0	K1	RADD2	M6	VSYNC~
A5	CMADD11	C3	VDAT15	F2	SEROE1~	K2	RADD0	M7	SIZ0
A6	CMADD08	C4	VDAT29	F3	SEROE0~	K3	A15	M8	DTOE1~
A7	GND	C5	VDAT25	F11	CMADD03	K11	D01	M9	READ
A8	CMADD14	C6	CMADD10	F12	CMADD04	K12	DTACK~	M10	A00
A9	RESET~	C7	VDD	F13	CMADD05	K13	WEN2~	M11	D06
A10	VDAT23	C8	CMADD12	G1	GND	L1	A17	M12	D03
A11	VDAT20	C9	VDAT22	G2	PCLKOUT~	L2	A14	M13	D02
A12	VDAT17	C10	VDAT18	G3	VDD	L3	A12	N1	A13
A13	VDAT06	C11	VDAT07	G11	CMADD07	L4	A09	N2	A10
B1	VDAT13	C12	VDAT05	G12	VDD	L5	A05	N3	A07
B2	VDAT14	C13	VDAT02	G13	CMADD06	L6	TESTEN~	N4	A04
B3	VDAT30	D1	CBLANK~	H1	RADD7	L7	GND	N5	A02
B4	VDAT26	D2	VDAT09	H2	RADD6	L8	CAS0~	N6	SIZ1
B5	PIXCLK	D3	VDAT12	H3	RADD5	L9	A01	N7	VDD
B6	CMADD09	D11	VDAT04	H11	RAS1~	L10	D07	N8	DTOE0~
B7	CMADD15	D12	VDAT01	H12	RAS0~	L11	D04	N9	CAS1~
B8	CMADD13	D13	CMADD00	H13	GND	L12	D00	N10	A18
B9	CTLSEL~	E1	CSYNC~	J1	RADD4	L13	RAMSEL~	N11	PAS~
B10	VDAT21	E2	HSYNC~	J2	RADD3	M1	A16	N12	CPU2X
B11	VDAT19	E3	VDAT08	J3	RADD1	M2	A11	N13	D05