Signetics

SCC63484
Advanced CRT Controller (ACRTC)

Preliminary Specification

Microprocessor Products

2

DESCRIPTION

The Signetics SCC63484 Advanced CRT Controller (ACRTC) is a CMOS VLSI microcomputer peripheral device capable of controlling raster scan type CRTs to display both graphics and characters. The ACRTC is a new generation CRT controller that is based on a bit-mapped technology and has more display control functions than those of an SCN2674 Advanced Video Display Controller (AVDC).

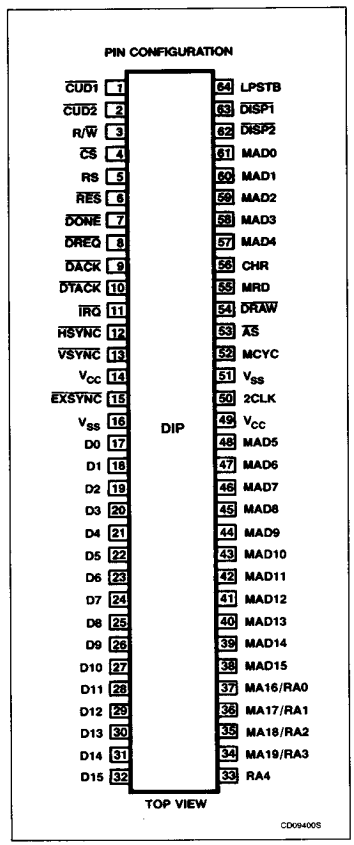
The ACRTC prepares the mechanisms to use in one of three modes: character only; graphic only and multiplexed character/graphic modes. Therefore, the ACRTC can be applied to many applications, from character-only display devices to large full-graphic systems.

The ACRTC can reduce CPU software overhead and enhance system throughput.

FEATURES

- **High-speed graphic drawings**
  - **Drawing rate: maximum 500ns/pixel (color drawing)**
  - **Drawn graphics: Dot, line, rectangle, polyline, polygon, circle, ellipse, paint, copy, etc.**
  - **Drawn colors: 16 bits/word, 1, 2, 4, 8, 16 bits/pixel (5 types) monochrome to max 64k colors**
- **Large frame memory space**
  - **Maximum 2Mbytes graphic memory**
  - **Maximum 128k-byte character memory separated from the MPU memory**
  - **Available to maximum 4096 × 4096 high-resolution CRT (1 bit/pixel mode)**
- **Various CRT display controls**
  - **Split screens (3 displays and 1 window)**
  - **Zooming up (1 to 16 times)**
  - **Scroll (vertical and horizontal)**
- **External synchronization**
  - **Synchronization between ACRTCs or between the ACRTC and external device; e.g. TV system or other controller**
- **DMA interface**
- **Two programmable cursors**
- **Three scan modes**
  - **Non-interlace, interlace sync., and interlace sync. and video modes**
- **Interrupt request to MPU**
- **256 characters/line, 32 rasters/line, 4096 rasters/screen**
- **Maximum clock frequency 8MHz**
- **CMOS, +5V single power supply**

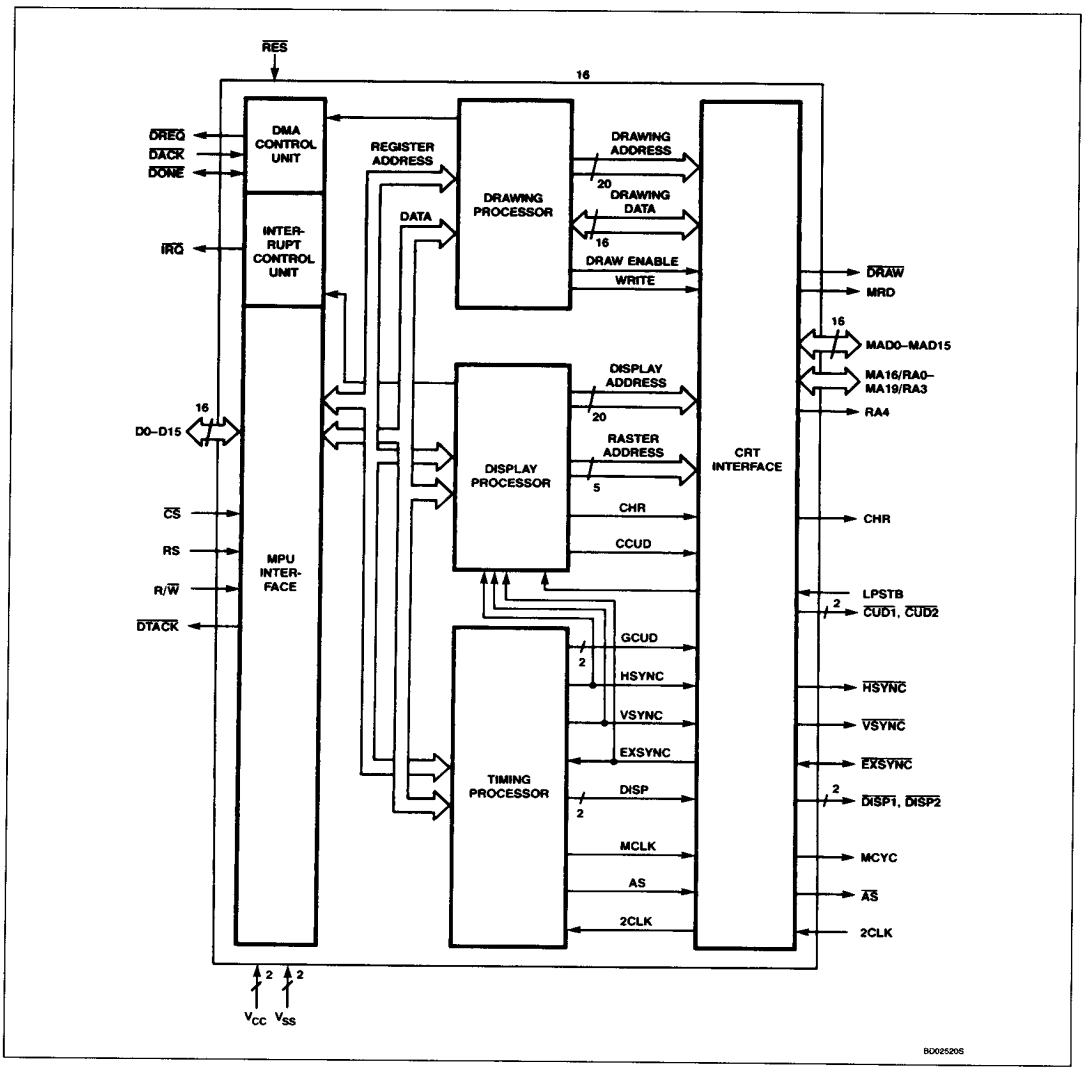
PIN CONFIGURATION



ORDERING INFORMATION

PACKAGE	V_{CC} = 5V ±5%, T_A = 0°C to +70°C
	8MHz
Plastic DIP	SCC63484C8N64
Ceramic DIP	SCC63484C8I64
Plastic LCC	SCC63484C8A68

BLOCK DIAGRAM



2

PIN DESCRIPTION

MNEMONIC	PIN NO.	TYPE	NAME AND FUNCTION
MPU Interface			
$\overline{RES}$	6	I	**Reset:** Hardware reset to the ACRTC.
D0 – D15	17 – 32	I/O	**Data Bus:** The bidirectional data bus for communication with the host MPU or DMAC. In 8-bit data bus mode, D0 – D7 are used.
R/$\overline{W}$	3	I	**Read/Write:** Controls the direction of host to/from ACRTC transfers.
$\overline{CS}$	4	I	**Chip Select:** Enables data transfers between the host and the ACRTC.
RS	5	I	**Register Select:** Selects the ACRTC register to be accessed and is normally connected to the least significant bit of the host address bus.
$\overline{DTACK}$	10	O	**Data Transfer Acknowledge:** Provides asynchronous bus cycle timing and is compatible with the SCN68000 Microprocessor $\overline{DTACK}$ input.
$\overline{IRQ}$	11	O	**Interrupt Request:** Generates interrupt service requests to the host MPU.
DMAC Interface			
$\overline{DREQ}$	8	O	**DMA Request:** Generates DMA service requests to the host DMAC.
$\overline{DACK}$	9	I	**DMA Acknowledge:** Receives DMA acknowledge timing from the host DMAC.
$\overline{DONE}$	7	I/O	**Done:** Terminates DMA transfer and is compatible with the DMAC $\overline{DONE}$ signal.
CRT Interface			
2CLK	50	I	**Clock:** Basic ACRTC operating clock derived from the dot clock.
MAD0 – MAD15	61 – 57, 48 – 38	I/O	**Address/Data Bus:** Multiplexed frame buffer address/data bus.
$\overline{AS}$	53	O	**Address Strobe:** Address strobe for demultiplexing the frame buffer address/data bus (MAD0 – MAD15).
MA16/RA0 – MA19/RA3	37 – 34	O	**Address Bits/Raster Address Outputs:** The high-order address bits for graphic screens and the raster address outputs for character screens.
RA4	33	O	**Raster Address Bit:** Provides the high-order raster address bit (up to 32 rasters) for character screens.
CHR	56	O	**Graphic/Character Screen Access:** Indicates whether a graphic or character screen is being accessed.
MCYC	52	O	**Memory Clock:** Frame buffer memory access timing — one half the frequency of 2CLK.
MRD	55	O	**Bus Direction Control:** Frame buffer data bus direction control.
$\overline{DRAW}$	54	O	**Drawing/Refresh Cycle:** Differentiates between drawing cycles and CRT display refresh cycles.
$\overline{DISP1}$ – $\overline{DISP2}$	62 – 63	O	**Display Enable Timing:** Programmable display enable timing used to selectively enable, disable and blank logical screens.
$\overline{CUD1}$ – $\overline{CUD2}$	1 – 2	O	**Cursor Timing:** Provides cursor timing determined by ACRTC programmed parameters such as cursor definition, cursor mode, cursor address, etc.
$\overline{VSYNC}$	13	O	**Vertical Synchronization:** CRT device vertical synchronization pulse.
$\overline{HSYNC}$	12	O	**Horizontal Synchronization:** CRT device horizontal synchronization pulse.
$\overline{EXSYNC}$	15	I/O	**External Synchronization:** For synchronization between multiple ACRTCs and other video signal generating devices.
LPSTB	64	I	**Light Pen:** Connection to an external light pen.

BLOCK DIAGRAM

The ACRTC consists of five major functional blocks. These functional blocks operate in parallel to achieve maximum performance. Two of the blocks perform the external bus interface for the host MPU and CRT, respectively.

MPU Interface

The MPU interface manages the asynchronous host MPU interface including the programmable interrupt control unit and DMA handshaking control unit.

CRT Interface

The CRT interface manages the frame buffer bus and CRT timing input and output control signals. Also, the selection of either display refresh address or drawing address outputs is performed. The other three blocks are separately microprogrammed processors which

operate in parallel to perform the major functions of drawing, display control, and timing.

Drawing Processor

This interprets commands and command parameters issued by the host bus (MPU and/or DMAC) and performs the drawing operations on the frame buffer memory. This processor is responsible for the execution of ACRTC drawing algorithms and conversion of logical pixel X-Y addresses to physical frame buffer addresses. Communication with the host bus is from separate 16-byte read and write FIFOs.

Display Processor

The display processor manages frame buffer refresh addressing based on the user-programmed specification of display screen organization. Combines and displays as many as four independent screen segments (three horizontal splits and one window) using an internal high-speed address calculation unit. Controls display refresh address outputs based on graphic (physical frame buffer address) or character (physical frame buffer address + row address) display modes.

Timing Processor

This generates the CRT synchronization signals and other timing signals used internally by the ACRTC. The ACRTC's software visible registers are similarly partitioned and reside in the appropriate internal processor, depending on function. The registers in the display and timing processors are loaded with basic display parameters during system initialization. During operation, the host primarily communicates with the ACRTC's drawing processor via the on-chip FIFOs.

OPERATION

Powerful visual interfaces are a key component of advanced system architectures. A proven technique uses raster-scanned CRT technology for the display of graphics and text information.

Systems which use first-generation CRT controllers (CRTCs) are constrained by hardware/software design time, manufacturing cost, and limited MPU bandwidth. To meet the functional requirements for powerful visual interfaces and to support their use in high volume, cost-sensitive applications, advanced circuit design and VLSI CMOS manufacturing technologies have been used to create a next generation CRTC, the SCC63484 ACRTC. The ACRTC concept is to incorporate major functionality on-chip, formerly requiring external hardware and software. In this way, both higher performance and reduced system cost benefits are achieved.

1. High-level command language increases performance and reduces software development cost.
   - — ACRTC converts logical X-Y coordinates to physical frame buffer addresses
   - — 38 commands including 23 graphic drawing commands — LINE, RECTANGLE, POLYLINE, POLYGON, CIRCLE, ELLIPSE, ARC, ELLIPSE ARC, FILLED RECTANGLE, PAINT, PATTERN and COPY
   - — On-chip 32-byte pattern RAM
   - — Conditional drawing function (8 conditions) for drawing patterns, color mixing and software mixing and software windowing
   - — Drawing area control with hardware clipping and hitting
   - — Maximum drawing speed of 2 million logical pixels per second is the same for monochrome and color applications
2. High resolution display with advanced screen control
   - — Up to 4096 by 4096 bit map graphic display and/or 256 line by 256 character by 32 raster character display
   - — Separate bit map graphic (2Mbyte) and character (128kbyte) address spaces with combined graphic/character display
   - — Three horizontal split screens and one window screen
   - — Size and position fully programmable
   - — Independent horizontal and vertical smooth scroll for each screen
   - — 1 to 16 zoom magnitude — independent X and Y zoom factors
   - — Logical pixel specification as 1, 2, 4, 8 or 16 bits for monochrome, gray scale and color displays
   - — Programmable address increment supports frame buffer memory widths to 256 bits for video bit rates > 500MHz. (ACRTC R mask is limited to 128 bits)
   - — Unique interleaved access mode for screen superimposition or 'flashless' displays
   - — ACRTC provides dynamic RAM refresh address
3. High-performance MPU interface
   - — Optimized interface with the SCN68000 MPU and DMAC
   - — 8- or 16-bit bus — compatible with other MPUs
   - — Separate on-chip 16-byte read and write FIFOs
   - — Maskable interrupts including FIFO status
4. Versatile CRT interface
   - — Full programmability of CRT timing signals
   - — Three raster scanning modes
   - — Master or slave synchronization to multiple ACRTCs or other video generating devices
   - — Two hardware cursors; three cursor modes
   - — Programmable cursor and display timing skew
   - — Eight user-defineable video attributes
   - — Light pen detection
5. VLSI CMOS process

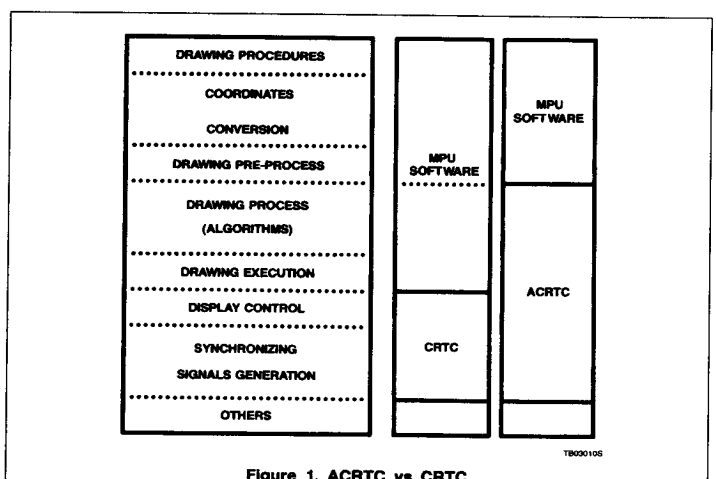


Figure 1. ACRTC vs CRTC

APPLICATIONS

The overall function of a visual interface is logically partitioned into layers. At the lowest layer are CRT timing and control signal generation. At the top layer are general purpose drawing procedures which provide a high-level interface to the user's operating system or application software. At this layer, a number of popular standards have emerged including GKS, CORE, NAPLP, GSX and others.

Figure 1 shows how the ACRTC performs the key functions or logical drawing algorithm and physical drawing execution. Formerly, these functions were performed by external hardware and/or MPU software.

As shown, the ACRTC reduces the 'gap' between device functionality and high-level graphics procedures. Since the ACRTC device itself provides capabilities closely related to those of high-level graphics packages, the effort (hardware and software design time and cost) required to develop a visual interface is significantly reduced.

Noting the traditional and emerging applications for visual interfaces, Figure 2 shows that a single ACRTC is suitable for a broad range of products in both alphanumeric and graphics areas. Multiple ACRTCs can achieve performance beyond that of any first-generation CRTC configuration.

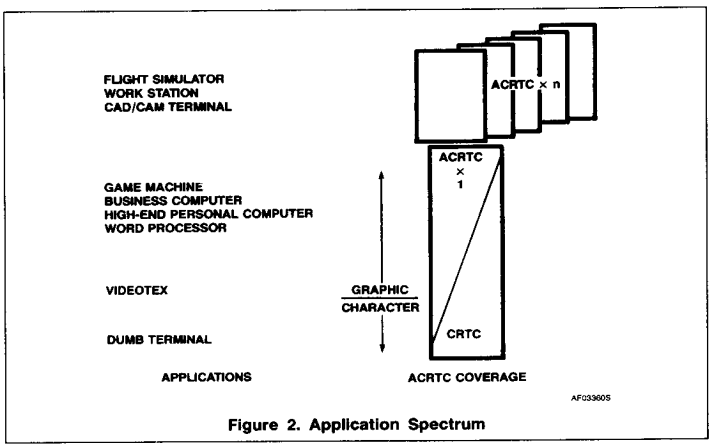


Figure 2. Application Spectrum

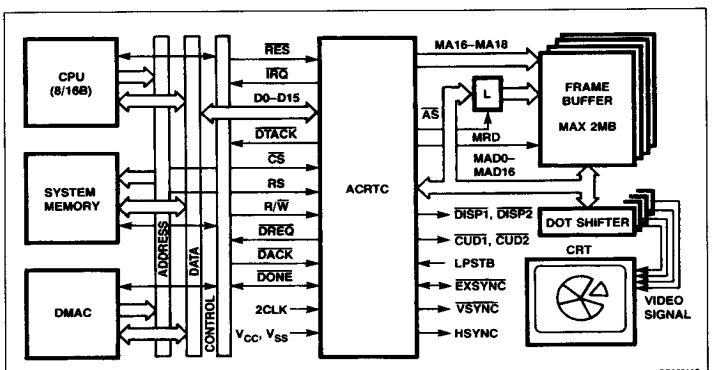


Figure 3. System Configuration

SYSTEM CONFIGURATION

Existing CRTCs provide a single bus interface to the frame buffer which must be shared with the host MPU. However, the refresh of large frame buffers and the requirement to access the frame buffer for drawing operations can quickly saturate this shared bus bandwidth.

As shown in Figure 3, the ACRTC uses separate host MPU and frame buffer bus interfaces. This allows the ACRTC full access to the frame buffer for display refresh, DRAM refresh and drawing operations while minimizing the ACRTC's usage of the MPU system bus. Thus, overall system performance is maximized. A related benefit is that a large frame buffer (2Mbyte for each ACRTC) is usable even if the host MPU has a smaller address space or segment size restriction.

The ACRTC can utilize an external DMA controller. This increases system throughput when large amounts of command, parameter and data information must be transferred to the ACRTC. Also, advanced DMAC features, such as the 'chaining' modes, can be used to develop powerful graphics system architectures.

However, more cost-sensitive or less performance-sensitive applications do not require a DMAC. The interface to the ACRTC can be handled completely under MPU software control.

While both ACRTC bus interfaces (host MPU and frame buffer) exploit 16-bit data paths for maximum performance, the ACRTC also offers an 8-bit MPU mode for easy connection to popular 8-bit bus structures.

VIDEO ATTRIBUTES

The ACRTC outputs 20 bits of video attributes on MAD0 – MAD15 and MA16/RA0 – MA19/RA3. These attributes are output at the last cycle prior to the rising edge of HSYNC and should be latched externally. Thus, video attributes can be set on a raster-by-raster basis (see Figure 4).

Attribute Code (ATC0 – ATC7:MAD0 – MAD7)

These are user-defined attributes. The programmed contents of the attribute control bits (ATR) of the display control register (DCR) are output on these lines. Note that the data written into ATR can be externally used after the completion of current raster scanning.

Attribute Code (ATC7 – ATC0) Application

ATC is one of the functions provided for user applications; the data employed depends on the system requirements. Application selections include:

1. Amount of horizontal dot shift for window smooth scroll
2. Horizontal width of crosshair cursor and the amount of horizontal dot shift (including block cursor)

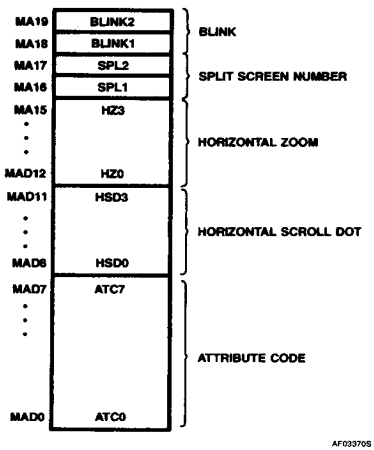


Figure 4. Video Attributes

3. Frame buffer specification in blocks (used for the base register)
4. Back screen color or character color code
5. Display screen selection during screen blink (used with SPL)
6. Interrupt vector address storage
7. Polarity selection of horizontal/vertical synchronization signal, etc.
8. Blinking signal like lamps used in the system
9. Code storage (maximum 8-bit) or selection signal, etc.

Horizontal Scroll (HSD0 – HSD3:MAD8 – MAD11)

These are used in conjunction with external circuitry to implement smooth horizontal scroll. These lines contain the encoded start dot address which is used to control the external shift register load timing and data. HSD usually corresponds to the start dot address of the background screens. However, if the window smooth scroll (SWS) bit of the OMR (operation mode register) is set to 1, HSD outputs the start dot address of the window screen segment. Note that HSD outputs the valid value only within the specified raster area. Changing the register contents during the scanning does not cause any external effects, because the value loaded at the beginning of the area is reserved.

Horizontal Zoom Factor (HZ0 – HZ3:MAD12 – MAD15)

These lines output the encoded (1 – 16) horizontal zoom factor as stored in the zoom factor register (ZFR). Horizontal zoom is accomplished by the ACRTC repeating a single display address and using the HZ outputs to control the external shift register clock. Horizontal zoom can only be applied to the base screen.

Split Position (SPL1 – SPL2:MA16 – MA17)

These lines present the encoded information showing the enabled background screen currently being displayed by the ACRTC.

SPL2	SPL1	
0	0	Background screen not enabled or displayed
0	1	Base screen
1	0	Upper screen
1	1	Lower screen

Even if the split screen display is prohibited, SPL is output if the area is specified.

Blink (BLINK1 – BLINK2:MA18 – MA19)

These lines are used to implement character and screen blink. The lines alternate from high to low periodically as defined in the blink control register (BCR). The blink frequency is specified in units of four field times. A field is defined as the period between successive VSYNC pulses.

ADDRESS SPACE

The ACRTC allows the host to issue commands using logical X-Y coordinate addressing. The ACRTC converts these to physical linear word addresses with bit field offsets in the frame buffer. Figure 5 shows the relationship between a logical X-Y screen address and the frame buffer memory, organized as sequential 16-bit words. The host may specify that a logical pixel consists of 1, 2, 4, 8 or 16 physical bits in the frame buffer. In the example, 4 bits per logical pixel is used allowing 16 colors or tones to be selected.

Up to four logical screens (upper, base, lower and window) are mapped into the ACRTC physical address space. The host specifies a logical physical start address, logical screen physical memory width (number of memory words per raster), logical pixel physical memory width (number of bits per pixel) and the logical origin physical address. Then, logical pixel X-Y addresses issued by the host or by the ACRTC drawing processor are converted to physical frame buffer addresses. The ACRTC also performs bit extraction and masking to map logical pixel operations (in the example, 4 bits) to 16-bit word frame buffer accesses.

The ACRTC has over two hundred bytes of accessible registers. These are organized as hardware-, directly-, and FIFO-accessible (see Figure 6 and Tables 1 and 2).

Hardware-Accessible

The ACRTC is connected to the host MPU as a standard peripheral which occupies two word locations of the host address space. The RS (register select) pin selects one of these two locations. When RS is low, reads access the status register and writes access the address register. The status register summarizes the ACRTC state and is used by the MPU to monitor the overall operation of the ACRTC. The address register is used to program the ACRTC with the address of the specific directly-accessible register which the MPU wishes to access.

Directly-Accessible

These registers are accessed by prior loading of the address register with the chosen register address. Then, when the MPU accesses the ACRTC with RS = 1, the chosen register is accessed. The FIFO entry enables access to FIFO-accessible registers using the ACRTC read and write FIFOs.

The command control register is used to control overall ACRTC operation such as aborting or pausing commands, defining DMA protocols, enabling/disabling interrupt sources, etc. The operation mode register defines basic parameters of ACRTC operation such as frame buffer access mode, display or drawing priority, cursor and display timing skew factors, raster scan mode, etc.

The display control register allows the independent enabling and disabling of each of the four ACRTC logical display screens (base, upper, lower, and window). Also, this register contains the 8 bits of user-defineable video attributes.

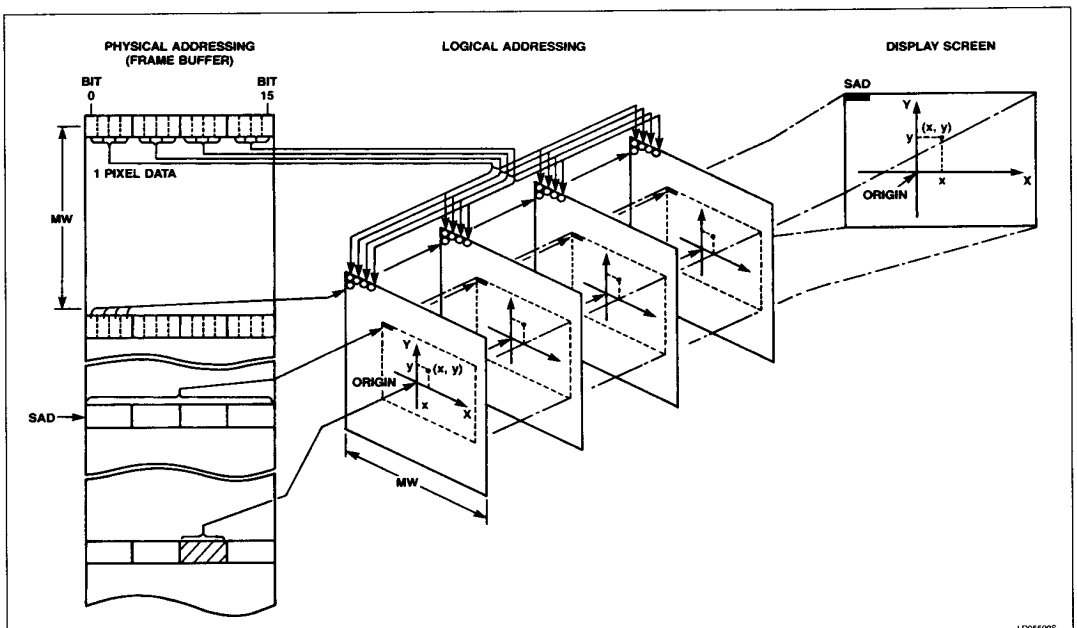


Figure 5. Logical/Physical Addressing

The timing control RAM contains registers which define ACRTC timing. This includes timing specification for CRT control signals (e.g. $\overline{\text{HSYNC}}$, $\overline{\text{VSYNC}}$), logical display screen size and display period, blink timing, etc. The display control RAM contains registers which define logical screen display parameters such as start addresses, raster addresses, and memory width. Also included are the cursor(s) definition, zoom factor, and light pen registers.

FIFO-Accessible

For high-performance drawing, key drawing processor registers are coupled to the host via the ACRTC's separate 16-byte read and write FIFOs. ACRTC commands are sent from the MPU via the write FIFO to the command register. As the ACRTC completes command execution, the next command is automatically fetched from the FIFO into the command register.

The pattern RAM is used to define drawing and painting 'patterns'. The pattern RAM is accessed using the ACRTC's read pattern RAM (RPTN) and write pattern RAM (WPTN) register access commands.

The drawing parameter registers define detailed parameters of the drawing process, such as color control, area control (hitting/clipping) and pattern RAM pointers. The drawing parameter registers are accessed using the ACRTC's read parameter register (RPR) and write parameter register (WPR) register access commands.

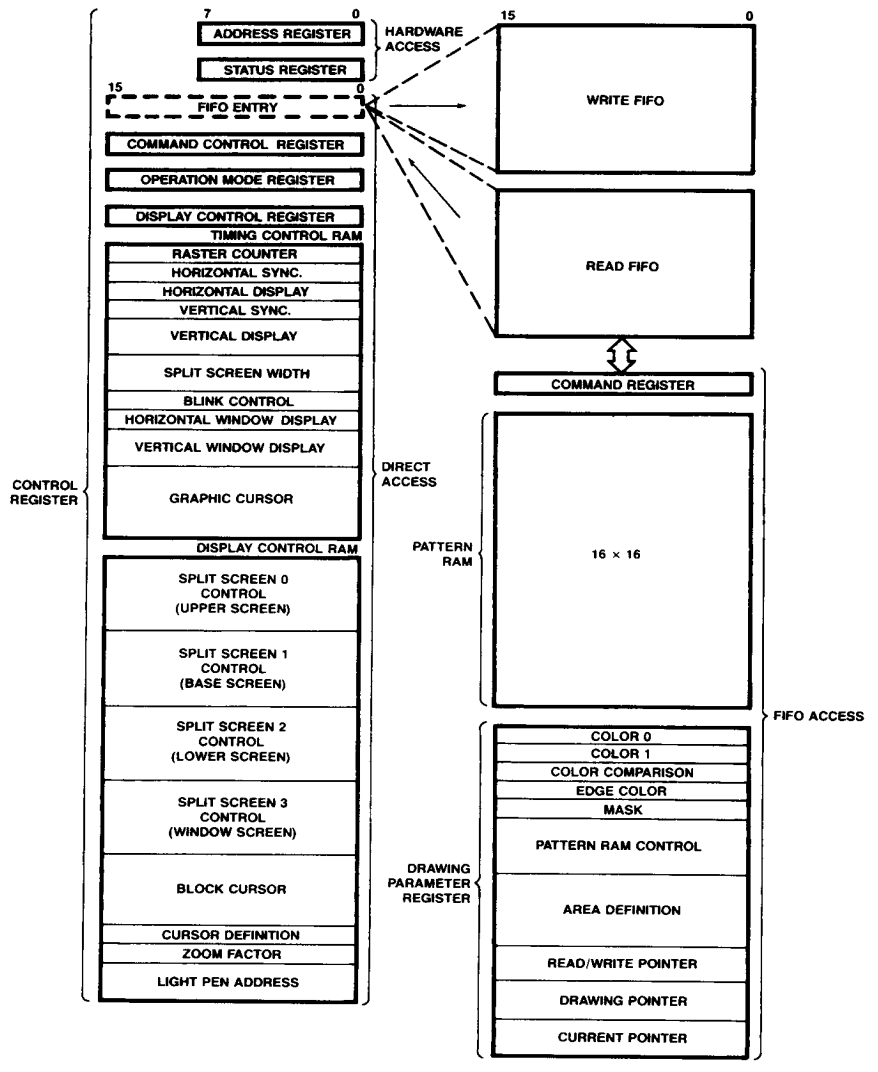


Figure 6. Programming Model

Table 1. Programming Model (Hardware Access, Direct Access Registers)

$\overline{CS}$	RS	$\overline{RW}$	REG. NO.	REGISTER NAME		ABBRE.	DATA (H) 15	14	13	12	11	10	9	8	DATA (L) 7	6	5	4	3	2	1	0
1	—	—	—	—		—	.....								.....							
0	0	0	AR	Address Register		AR	.....								Address							
0	0	1	SR	Status Register		SR	.....								CER	ARD	CED	LPD	RFF	RFR	WFR	WFE
		½	r00	FIFO Entry		FE	FE															
		½	r02	Command Control		CCR	ABT	PSE	DDM	CDM	DRC	GBM			CRE	ARE	CEE	LPE	RFE	RRE	WRE	WEE
		½	r04	Operation Mode		OMR	M/S	STR	ACP	WSS	CSK		DSK		RAM	GAI			ACM		RSM	
		½	r06	Display Control		DCR	DSP	SE1	SE0		SE2		SE3		ATR							
		—	r08 ↓ r7E	(undefined)		—	.....															
		1	r80	Raster Count		RCR	.....				RC											
		½	r82	Horizontal Sync.		HSR	HC								.....				HSW			
			r84	Horizontal Display		HDR	HDS								HDW							
		½	r86	Vertical Sync.		VSR	.....				VC											
		½	r88	Vertical Display		VDR	VDS								.....				VSW			
		½	r8A	Split Screen Width		SSW	.....				SP1											
		½	r8C				.....				SP0											
		½	r8E				.....				SP2											
		½	r90	Blink Control		BCR	BON1				BOFF1				BON2				BOFF2			
		½	r92	Horizontal Window Display		HWR	HWS								HWW							
		½	r94	Vertical Window Display		VWR	.....				VWS											
		½	r96				.....				VWW											
		½	r98	Graphic Cursor		GCR	CXE								CXS							
		½	r9A				.....				CYS											
		½	r9C				.....				CYE											
		—	r9E	(undefined)		—	.....															
		—	rA0 ↓ rBE																			
0	1	½	rC0	Upper Screen	Raster Addr. 0	RAR0	.....				LRA0				.....				FRA0			
		½	rC2		Memory Width 0	MWR0	CHR	.....			MW0											
		½	rC4		Start Addr. 0	SAR0	.....				SDA0				.....				SA0H/SRA0			
		½	rC6				SA0L															
		½	rC8	Base Screen	Raster Addr. 1	RAR1	.....				LRA1				.....				FRA1			
		½	rCA		Memory Width 1	MWR1	CHR	.....			MW1											
		½	rCC		Start Addr. 1	SAR1	.....				SDA1				.....				SA1H/SRA1			
		½	rCE				SA1L															
		½	rD0	Upper Screen	Raster Addr. 2	RAR2	.....				LRA2				.....				FRA2			
		½	rD2		Memory Width 2	MWR2	CHR	.....			MW2											
		½	rD4		Start Addr. 2	SAR2	.....				SDA2				.....				SA2H/SRA2			
		½	rD6				SA2L															
		½	rD8	Window Screen	Raster Addr. 3	RAR3	.....				LRA3				.....				FRA3			
		½	rDA		Memory Width 3	MWR3	CHR	.....			MW3											
		½	rDC		Start Addr. 3	SAR3	.....				SDA3				.....				SA3H/SRA3			
		½	rDE				SA3L															
		½	rE0	Block Cursor 1		BCUR1	BCW1				BCSR1				.....				BCER1			
		½	rE2				BCA1															
		½	rE4	Block Cursor 2		BCUR2	BCW2				BCSR2				.....				BCER2			
		½	rE6				BCA2															
		½	rE8	Cursor Definition		CDR	CM		CON1			COFF1			.....		CON2			COFF2		
		½	rEA	Zoom Factor		ZFR	HZF				VZF				.....							
		1	rEC	Light Pen Address		LPAR	.....								CHR	.....			LPAH			
		1	rEE				LPAL															
		—	rF0 ↓ rFE	(undefined)		—	.....															

NOTE: { 1 .... "High" level
0 .... "Low" level

HARDWARE ACCESS, DIRECT ACCESS REGISTERS

ABT: Abort
ACM: Access Mode
ACP: Access Priority
Address: Register No. of the control register
ARD: Area Detect
ARE: Area Detect Interrupt Enable
ATR: Attribute Control
CDM: Command DMA Mode
CED: Command End
CEE: Command End Interrupt Enable
CER: Command Error
CRE: Command Error Interrupt Enable
CSK: Cursor Display Skew
DDM: Data DMA Mode
DRC: DMA Request Control
DSK: $\overline{DISP}$ Skew
DSP: $\overline{DISP}$ Signal Control
FE: FIFO Entry
GAI: Graphic Address Increment Mode
GBM: Graphic Bit Mode
HC: Horizontal Cycle
HDS: Horizontal Display Start
HDW: Horizontal Display Width
HSW: Horizontal Sync. Width
LPD: Light Pen Strobe Detect
LPE: Light Pen Strobe Interrupt Enable
M/S: Master/Slave
PSE: Pause
RAM: RAM Mode
RC: Raster Count
RFE: Read FIFO Full Interrupt Enable
RFF: Read FIFO Full
RFR: Read FIFO Ready
RRE: Read FIFO Ready Interrupt Enable
RSM: Raster Scan Mode
SE0: Split Enable 0
SE1: Split Enable 1
SE2: Split Enable 2
SE3: Split Enable 3
STR: Start
VC: Vertical Cycle
VDS: Vertical Display Start
VSW: Vertical Sync. Width
WEE: Write FIFO Empty Interrupt Enable
WFE: Write FIFO Empty
WFR: Write FIFO Ready
WRE: Write FIFO Ready Interrupt Enable
WSS: Window Smooth Scroll
SP0, SP1, SP2: Split Screen 0 Width, Split Screen 1 Width, Split Screen 2 Width
BON1, BON2: Blink ON 1, Blink ON 2
BOFF1, BOFF2: Blink OFF 1, Blink OFF 2
HWS: Horizontal Window Start
HWW: Horizontal Window Width
VWS: Vertical Window Start
VWW: Vertical Window Width
CXS, CYS: Cursor X Start, Cursor Y Start
CXE, CYE: Cursor X End, Cursor Y End
FRA: First Raster Address
LRA: Last Raster Address
CHR: Character
MW: Memory Width
SDA: Start Dot Address
SAH/SRA: Start Address "High"/Start Raster Address
SAL: Start Address "Low"
BCW1, BCW2: Block Cursor Width 1, Block Cursor Width 2
BCSR1, BCSR2: Block Cursor Start Raster 1, Block Cursor Start Raster 2
BCER1, BCER2: Block Cursor End Raster 1, Block Cursor End Raster 2
BCA1, BCA2: Block Cursor Address 1, Block Cursor Address 2
CM: Cursor Mode
CON1, CON2: Cursor ON 1, Cursor ON 2
COFF1, COFF2: Cursor OFF 1, Cursor OFF 2
HZF, VZF: Horizontal Zoom Factor, Vertical Zoom Factor
LPAH: Light Pen Address "High"
LPAL: Light Pen Address "Low"

Table 2. Programming Model (Drawing Parameter Registers)

REGISTER NO.	READ/ WRITE	NAME OF REGISTER	ABBR.	DATA (H) 15	14	13	12	11	10	9	8	DATA (L) 7	6	5	4	3	2	1	0
Pr00	R/W	Color 0	CL0	CL0															
Pr01	R/W	Color 1	CL1	CL1															
Pr02	R/W	Color Comparison	CCMP	CCMP															
Pr03	R/W	Edge Color	EDG	EDG															
Pr04	R/W	Mask	MASK	MASK															
Pr05	R/W	Pattern RAM Control	PRC	PPY								PZCY				PPX			PZCX
↓				PSY								//////				PSX			//////
Pr07				PEY								PZY				PEX			PZX
Pr08	R/W	Area Definition **	ADR	XMIN															
↓				YMIN															
Pr0B				XMAX															
				YMAX															
Pr0C	R/W	Read/Write Pointer	RWP	DN				//////				RWPH							
Pr0D				RWPL														//////	
Pr0E	—	.....	—	—															
Pr0F				—															
Pr10	R	Drawing Pointer	DP	DN				//////				DPAH							
Pr11				DPAL															DPD
Pr12	R	Current Pointer **	CP	X															
Pr13				Y															
Pr14	—	.....	—	—															
Pr15				—															

NOTES:

////// .... Always set to "0"

** .... Set binary complements for negative values of X and Y axis.

DRAWING PARAMETER REGISTER

R: Register which can be read by Read Parameter Register command (RPR)
W: Register which can be written into by Write Parameter Register command (WPR)
—: Access is not allowed
CL0: Defines the color data used for the drawing when logical drawing data = 0
CL1: Defines the color data used for the drawing when logical drawing data = 1
CCMP: Defines the comparative color of the drawing operation
EDG: Defines the edge color
MASK: Defines the bit pattern used to mask bits upon which data transfer should not be performed
PSX, PSY: Pattern Start Point
PEX, PEY: Pattern End Point
PPX, PPY: Pattern Scan Start Point
PZX, PZY: Pattern Zoom
PZCX, PZCY: Pattern Zoom Count
XMIN, YMIN: Start point of Area definition
XMAX, YMAX: End point of Area definition
DN: Screen Number
RWPH: High-order 8 bits of Read Write Pointer Address
RWPL: Low-order 12 bits of Read Write Pointer Address
DPAH: High-order 8 bits of Drawing Pointer Address
DPAL: Low-order 12 bits of Drawing Pointer Address
DPD: Drawing Pointer Dot Address
X, Y: Position indicated by Current Pointer on X-Y coordinate

COMMANDS

The ACRTC has 38 commands classified into three groups — register access, data transfer, and graphic drawing (see Table 3). Five register access commands allow access to drawing processor, drawing parameter registers, and the pattern RAM. Ten data transfer commands are used to move data between the host system memory and the frame buffer, or within the frame buffer. Twenty-three graphic drawing commands cause the ACRTC to perform drawing operations. Parameters for these commands are specified using logical X-Y addressing.

All of the above commands, parameters and data are transferred via the ACRTC read and write FIFOs.

Assuming the ACRTC has been properly initialized, the MPU must perform two steps to cause graphic drawing. First, the MPU must specify certain drawing parameters which define a number of details associated with the drawing process. For example, to draw a figure or paint an area, the MPU must specify the drawing or painting 'pattern' by initializing the ACRTC pattern RAM and related pointers. Also, if clipping and hitting control are desired, the MPU specifies the 'area' to be monitored during drawing by initializing area definition registers. Other drawing parameters include color, edge definition, etc.

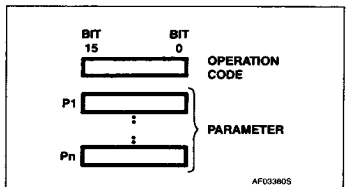


Figure 7. 16-Bit Interface

After the drawing parameters have been specified, the MPU issues a graphic drawing command and any required command parameters, such as the CRCL (circle) command with a radius parameter. The ACRTC then performs the specified drawing operation by reading, modifying and rewriting the contents of the frame buffer.

Command Format

ACRTC commands consist of a 16-bit opcode, optionally followed by one or more 16-bit parameters. When 8-bit MPU mode is used, commands, parameters, and data are sent to and from the ACRTC in the order of high-byte, low-byte.

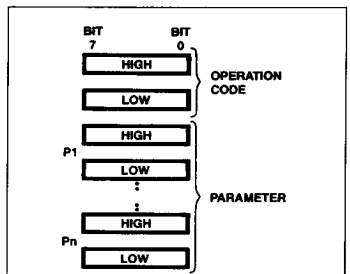


Figure 8. 8-Bit Interface

16-Bit Interface — In the case of 16-bit interface, first move the 16-bit operation code and then move necessary 16-bit parameters one by one (see Figure 7).

8-Bit Interface — In the case of 8-bit interface, first move the operation code's high byte followed by low byte, and then move the parameters in the same order (see Figure 8).

REGISTER ACCESS COMMAND

MNEMONIC	OPERATION CODE	PARAMETER	NO. WORDS	~ (CYCLES)
ORG	0 0 0 0 : 0 1 : 0 0 : 0 0 0 0 : 0 0 0 0	DPH DPL	3	8
WPR	0 0 0 0 : 1 0 : 0 0 : 0 0 0 : RN	D	2	6
RPR	0 0 0 0 : 1 1 : 0 0 : 0 0 0 : RN		1	6
WPTN	0 0 0 1 : 1 0 : 0 0 : 0 0 0 0 : PRA	n D_1,.....,D_n	n + 2	4n + 8
RPTN	0 0 0 1 : 1 1 : 0 0 : 0 0 0 0 : PRA	n	2	4n + 10

RN : Register number of the drawing parameter register ($0 – $13)
PRA: Pattern RAM address at which Read/Write operation starts ($0 – $F)
DPH: Drawing pointer register High word
DPL: Drawing pointer register Low word

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPH	DN		0	0	0	0	0	0	DPAH							
DPL	DPAL														DPD	

DPAH: Higher 8 bits of Drawing Pointer address
DPAL: Lower 12 bits of Drawing Pointer address
DPD: Dot position in the memory address
D, D_1,.....,D_n: Write data
n: Number of Read/Write data

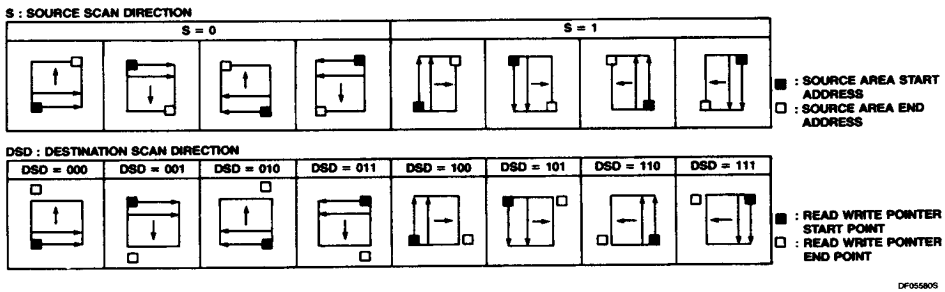
DN	SCREEN NO.
00	Upper Screen
01	Base Screen
10	Lower Screen
11	Window Screen

2

DATA TRANSFER COMMAND

MM: Modify Mode

MM	FUNCTION	
00	Replace	Replace drawing point data with modifier information
01	OR	OR drawing point data with modifier data and rewrite the result data to the frame buffer
10	AND	AND drawing point data with modifier data and rewrite the result data to the frame buffer
11	Ex-OR	Ex-OR drawing point data with modifier data and rewrite the result data to the frame buffer



AX: Number of word in X-axis direction-1
AY: Number of word in Y-axis direction-1
D: Write data
SAH: Source Start Address High word
SAL: Source Start Address Low word

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAH	0	0	0	0	0	0	0	0	(SAH)							
SAL	(SAL)												0	0	0	0

(SAH): Memory address Higher 8 bits
(SAL): Memory address Lower 12 bits
x: Number of word in X-axis direction
y: Number of word in Y-axis direction
↑: Rounding up

GRAPHIC DRAWING COMMAND

AREA : Area Mode
COL : Color Mode
OPM : Operation Mode

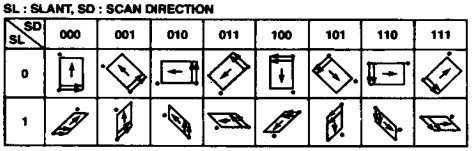
C: Circling Direction

C	DIRECTION
0	Counterclockwise
1	Clockwise

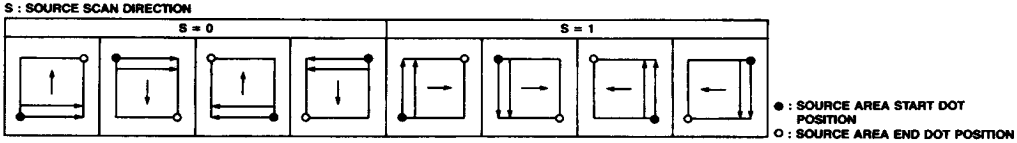
E: Definition of Edge color

E	DEFINITION
0	Edge color is defined by the data in the edge color register.
1	Edge color is defined by the data excluding the above.

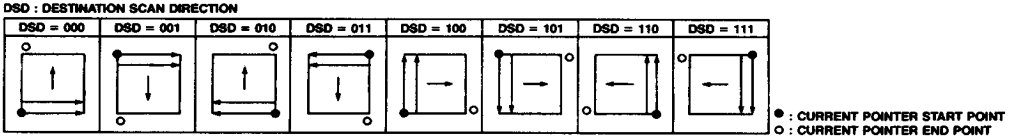
SL : SLANT, SD : SCAN DIRECTION



S : SOURCE SCAN DIRECTION



DSD : DESTINATION SCAN DIRECTION



DF05690S

X, X1, . . ., X_n : Absolute X-address from the original point
Y, Y1, . . ., Y_n : Absolute Y-address from the original point
dX : Relative X-address from the current pointer
dY : Relative Y-address from the current pointer
n : Number of notes
dX1, . . ., dXn : Relative X-address from each node
dY1, . . ., dYn : Relative Y-address from each node
r : Dot number on radius
a, b : $(DX)^2$: $(DY)^2$ = a : b
DX : X-direction dot number
DY : Y-direction dot number
Xc : Absolute X-address of the center point of arc/ellipse
Yc : Absolute Y-address of the center point of arc/ellipse
dXc : Relative X-address from the current pointer to the center point of arc/ellipse
dYc : Relative Y-address from the current pointer to the center point of arc/ellipse
Xe : Absolute X-address of the end point of arc/ellipse
Ye : Absolute Y-address of the end point of arc/ellipse
dXe : Relative X-address from the current pointer to the end point of arc/ellipse
dYe : Relative Y-address from the current pointer to the end point of arc/ellipse
Xs : Absolute X-address of the start dot position in source area
Ys : Absolute Y-address of the start dot position in source area
dXs : Relative X-address from the current pointer to the state dot position in source area
dYs : Relative Y-address from the current pointer to the state dot position in source area
P : 4(OPM = 000 – 011)/6(OPM = 100 – 111)
L, L0 : Dot number on straight line
d : Total dot number
A : Scan main direction dot number
B : Scan sub direction dot number

Table 3. ACRTC Command Table

TYPE	MNEMONIC	COMMAND NAME	OPERATION CODE	PARAMETER	NO. WORDS	(CYCLES)[3]
Register Access Command	ORG	Origin	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	DPH DPL	3	8
	WPR	Write Parameter Register	0 0 0 0 1 0 0 0 0 0 0 RN	D	2	6
	RPR	Read Parameter Register	0 0 0 0 1 1 0 0 0 0 0 RN		1	6
	WPTN	Write Pattern RAM	0 0 0 1 1 0 0 0 0 0 0 PRA	n D_1,...,D_n	n + 2	4n + 8
	RPTN	Read Pattern RAM	0 0 0 1 1 1 0 0 0 0 0 PRA	n	2	4n + 10
Data Transfer Command	DRD	DMA Read	0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0	AX AY	3	(4x + 8)y + 12[x · y/8↑] + (62~68)
	DWT	DMA Write	0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0	AX AY	3	(4x + 8)y + 16[x · y/8↑] + 34
	DMOD	DMA Modify	0 0 1 0 1 1 0 0 0 0 0 0 0 MM	AX AY	3	(4x + 8)y + 16[x · y/8↑] + 34
	RD	Read	0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0		1	
	WT	Write	0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0	D	2	8
	MOD	Modify	0 1 0 0 1 1 0 0 0 0 0 0 0 MM	D	2	8
	CLR	Clear	0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0	D AX AY	4	(2x + 8)y + 12
	SCLR	Selective Clear	0 1 0 1 1 1 0 0 0 0 0 0 0 MM	D AX AY	4	(4x + 6)y + 12
	CPY	Copy	0 1 1 0 S DSD 0 0 0 0 0 0 0 0	SAH SAL AX AY	5	(6x + 10)y + 12
	SCPY	Selective Copy	0 1 1 1 S DSD 0 0 0 0 0 0 MM	SAH SAL AX AY	5	(6x + 10)y + 12
Graphic Command	AMOVE	Absolute Move	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	X Y	3	56
	RMOVE	Relative Move	1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	dX dY	3	56
	ALINE	Absolute Line	1 0 0 0 1 0 0 0 AREA COL OPM	X Y	3	P · L + 18
	RLINE	Relative Line	1 0 0 0 1 1 0 0 AREA COL OPM	dX dY	3	P · L + 18
	ARCT	Absolute Rectangle	1 0 0 1 0 0 0 0 AREA COL OPM	X Y	3	2P(A + B) + 54
	RRCT	Relative Rectangle	1 0 0 1 0 1 0 0 AREA COL OPM	dX dY	3	2P(A + B) + 54
	APLL	Absolute Polyline	1 0 0 1 1 0 0 0 AREA COL OPM	n X1, Y1,...Xn, Yn	2n + 2	Σ[P · L + 16] + 8
	RPLL	Relative Polyline	1 0 0 1 1 1 0 0 AREA COL OPM	n dX1, dY1,...dXn, dYn	2n + 2	Σ[P · L + 16] + 8
	APLG	Absolute Polygon	1 0 1 0 0 0 0 0 AREA COL OPM	n X1, Y1,...Xn, Yn	2n + 2	Σ[P · L + 16] + P · Lo + 20
	RPLC	Relative Polygon	1 0 1 0 0 1 0 0 AREA COL OPM	n dX1, dY1,...dXn, dYn	2n + 2	Σ[P · L + 16] + P · Lo + 20
	CRCL	Circle	1 0 1 0 0 0 C 0 AREA COL OPM	r	2	8d + 66
	ELPS	Ellipse	1 0 1 0 1 1 0 C AREA COL OPM	a b dX	4	10d + 90
	AARC	Absolute Arc	1 0 1 1 0 0 0 C AREA COL OPM	Xc Yc Xe Ye	5	8d + 18
	RARC	Relative Arc	1 0 1 1 0 1 0 C AREA COL OPM	dXc dYc dXe dYe	5	8d + 18
	AEARC	Absolute Ellipse Arc	1 0 1 1 1 0 0 C AREA COL OPM	a b Xc Yc Xe Ye	7	10d + 96
	REARC	Relative Ellipse Arc	1 0 1 1 1 1 0 C AREA COL OPM	a b dXc dYc dXe dYe	7	10d + 96
	AFRCT	Absolute Filled Rectangle	1 1 0 0 0 0 0 0 AREA COL OPM	X Y	3	(P · A + B)B + 18
	RFRCT	Relative Filled Rectangle	1 1 0 0 0 1 0 0 AREA COL OPM	dX dY	3	(P · A + B)B + 18
	PAINT	Paint	1 1 0 0 1 0 0 E AREA 0 0 000		1	(18A + 102)B–58[1]
	DOT	Dot	1 1 0 0 1 1 0 0 AREA COL OPM		1	8
	PTN	Pattern	1 1 0 1 SL SD AREA COL OPM	SZ[2]	2	(P · A + 10)B + 20
	AGCPY	Absolute Graphic Copy	1 1 1 0 S DSD AREA 0 0 OPM	Xs Ys DX DY	5	((P + 2)A + 10)B + 70
	RGCPY	Relative Graphic Copy	1 1 1 1 S DSD AREA 0 0 OPM	dXs dYs DX DY	5	((P + 2)A + 10)B + 70

NOTES:

1. In case of rectangular filling

2. SZ:

15 8	7 0
SZy	SZx

SZy, SZx: Pattern Size

n: number of repetition X/Y: drawing words of X-direction/Y-direction
L/Lo/d: sum of drawing dots A/B: drawing dots of main/sub direction
E: [E = 0 (Stop at Edge color), E = 1 (Stop at excepting Edge color)] C: [C = 1 (clockwise), C = 0 (reverse)]
[↑]: rounding up
P = 4: OPM-000 – 011
6: OPM-100 – 111

3. Cycles: 2 clock cycle time

PROGRAM TRANSFER

Program transfer occurs when the MPU specifies the FIFO entry address and then writes commands/parameters to the write FIFO under program control (RS = high; R/$\overline{W}$, $\overline{CS}$ = low). The MPU writes are normally synchronized with the ACRTC FIFO status by software polling or interrupts.

Software Polling (WFR, WFE Interrupts Disabled)

1. MPU program checks the SR (status register) for write FIFO ready (WFR) flag = 1, and then writes 1-word op-code/parameters.
2. MPU program checks the SR (status register) for write FIFO empty (WFE) flag = 1, and then writes 1- to 8-word op-code/parameters.

Interrupt-Driven (WFR, WFE Interrupts Enabled)

1. MPU WFR interrupt service routine writes 1-word op-code/parameters.
2. MPU WFE interrupt service routine writes 1- to 8-word op-code/parameters.

In the specific case of register access commands and an initially empty write FIFO, MPU writes need not be synchronized to the write FIFO status. The ACRTC can fetch and execute these commands faster than the MPU can issue them.

COMMAND DMA TRANSFER

Commands and parameters can be transferred from MPU system memory using an external DMAC. The MPU initiates and terminates command DMA transfer mode under software control (CDM bit of CCR). Command DMA can also be terminated by assertion of the ACRTC $\overline{DONE}$ signal. $\overline{DONE}$ is treated as an input in command DMA transfer mode.

Using command DMA transfer, the ACRTC will issue cycle stealing DMA requests to the DMAC when the write FIFO is empty. The DMA data is automatically sent from system memory to the ACRTC write FIFO regardless of the contents of the address register.

NOTES:

1. Ensure that the write FIFO is empty and all the commands are terminated before starting the command DMA transfer.
2. The data DMA command cannot be executed in the command DMA transfer mode.

Register Access Commands

Registers associated with the drawing processor (pattern RAM and drawing parameter registers) are accessed through the read and write FIFOs using the register access commands.

Data Transfer Commands

Data transfer commands are used to move blocks of data between the MPU system memory and the ACRTC frame buffer or within the frame buffer itself. Before issuing these commands, a physical 20-bit frame buffer address must be specified in the RWP (read/write pointer) drawing parameter register.

Tables 4 and 5 list register access commands and data transfer commands. Figure 9 shows the data transfer command format.

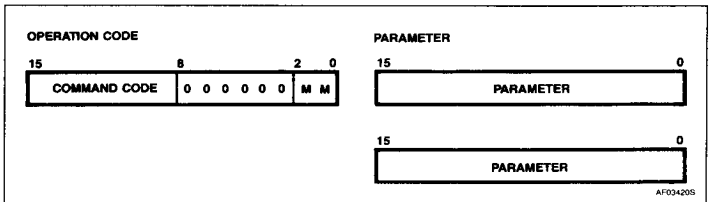


Figure 9. Data Transfer Command Format

Table 4. Register Access Commands

COMMAND	FUNCTION
ORG	Initialize the relation between the origin point in the X-Y coordinates and the physical address
WPR	Write into the parameter register
RPR	Read the parameter register
WPTN	Write into the pattern RAM
RPTN	Read the pattern RAM

Table 5. Data Transfer Commands

COMMAND	FUNCTION
DRD	Transfer data, by DMA transfer, from the frame buffer to the MPU system memory
DWT	Transfer data, by DMA transfer, from the MPU system memory to the frame buffer
DMOD	Transfer data, by DMA transfer, from the MPU system to the frame buffer subject to logical modification (bit-maskable)
RD	Read one word of data from the frame buffer specified by the read/write pointer (RWP), and load the word into Read FIFO
WT	Write one word of data to the frame buffer specified by the read/write pointer (RWP)
MOD	Perform logical operation on one word in the frame buffer specified by the read/write pointer (RWP) (bit-maskable)
CLR	Clear a rectangular area of the frame buffer with a data in the command parameter
SCLR	Initialize a rectangular area of the frame buffer with 1-word data subject to logical operation (bit-maskable)
CPY	Copy frame buffer data from one area (source area) to another area (destination area) specified by the read/write pointer (RWP).
SCPY	Copy frame buffer data from one area (source area) to another area (destination area) subject to logical modification by word. The source and destination areas must reside on the same screen (bit-maskable)

MODIFY MODE

The DMOD, MOD, SCLR and SCPY commands allow four types of bit-level logical operations to be applied to frame buffer data. The modify mode is encoded in the lower two bits (MM) of these op-codes. The bit positions within each frame buffer word to be modified are selectable using the mask register (MASK). Bits set to 1 are modifiable, ones to 0 are masked and not modifiable.

MM		Modify Mode
0	0	REPLACE frame buffer data with command parameter data
0	1	OR frame buffer data with command parameter data and rewrite to the frame buffer
1	0	AND frame buffer data with command parameter data and rewrite to the frame buffer
1	1	Ex-OR frame buffer data with command parameter data and rewrite to the frame buffer

GRAPHIC DRAWING COMMANDS

The ACRTC has 23 separate graphic drawing commands (see Table 6). Graphic drawing is performed by modifying the contents of the frame buffer based on microcoded drawing algorithms in the ACRTC drawing processor.

Most coordinate parameters for graphic drawing commands are specified using logical pixel X-Y addressing.

The complex task of translating a logical pixel address to a linear frame buffer word address, and further selecting the appropriate sub-field of the word is performed at high speed by ACRTC hardware. For example, a given logical pixel in 4 bits per logical pixel mode might reside in bits 8 – 11 of a frame buffer word.

Many instructions allow specification of X-Y coordinates with either absolute or relative X-Y coordinates (e.g., ALINE and RLINE). In both cases, two's complement numbers are used to represent positive and negative result values.

Absolute Coordinate Specification

The screen address (X, Y) is specified in units of logical pixels relative to an origin point defined with the ORG command (see Figure 10).

Relative Coordinate Specification

The screen address (dX, dY) is specified in units of logical pixels relative to the current drawing pointer (CP) position. A graphic drawing command consists of a 16-bit op-code and optionally 0 to 64k 16-bit parameters (see Figure 11). The 16-bit op-code consists of an 8-bit command code, an area mode specifier (3 bits), a color mode specifier (2 bits) and an operation mode specifier (3 bits).

Table 6. Graphic Drawing Commands

COMMAND	FUNCTION
AMOVE	Move the current pointer (CP) to an absolute logical pixel X-Y address
RMOVE	Move the CP to a relative logical pixel X-Y address
ALINE	Draw a straight line from the CP to a command-specified endpoint of the absolute coordinates
RLINE	Draw a straight line from the CP to a command-specified endpoint of the relative coordinates
ARCT	Draw a rectangle defined by the CP and a command-specified diagonal point of the absolute coordinates
RRCT	Draw a rectangle defined by the CP and a command-specified diagonal point of the relative coordinates
APLL	Draw a polyline (multiple contiguous segments) from the CP through command-specified points of the absolute coordinates
RPLL	Draw a polyline (multiple contiguous segments) from the CP through command-specified points of the relative coordinates
APLG	Draw a polygon which connects the start pointer (CP) and command-specified points of the absolute coordinates
RPLG	Draw a polygon which connects the CP and command-specified points of the relative coordinates
CRCL	Draw a circle of the radius, R, placing the CP at the center
ELPS	Draw an ellipse whose shape is specified by command parameters, placing the CP at the center
AARC	Draw an arc by using the CP as a start point with an end point and a center point of the absolute coordinates
RARC	Draw an arc by using the CP as a start point with an end point and a center point of the relative coordinates.
AEARC	Draw an ellipse arc by using the CP as a start point with an end point and a center point of the absolute coordinates
REARC	Draw an ellipse arc by using the CP as a start point with an end point and a center point of the relative coordinates
AFRCT	Paint a rectangular area specified by the CP and command parameters (absolute coordinates) according to a figure pattern stored in the pattern RAM (tiling).
RFRCT	Paint a rectangular area specified by the CP and command parameters (relative coordinates) according to a figure pattern stored in the pattern RAM (tiling).
PAINT	Paint a closed area surrounded by edge color using a figure pattern stored in the pattern RAM (tiling).
DOT	Mark a dot on the coordinates where the CP indicates
PTN	Draw a graphic pattern defined in the pattern RAM onto a rectangular area specified by the CP and by the pattern size (rotation angle: 45°)
AGCPY	Copy a rectangular area specified by the absolute coordinates to the address specified by the CP, (rotation angle: 90°/mirror turnover)
RGCPY	Copy a rectangular area specified by the relative coordinates to the address specified by the CP (rotation angle: 90°/mirror turnover)

The area mode allows versatile clipping and hitting detection. A drawing area can be defined, and should drawing operations attempt to enter or leave that area, a number of programmable actions can be taken by the ACRTC.

Absolute Coordinate Specification
Specifies the addresses (x, y) based on the origin point set by the ORG command.

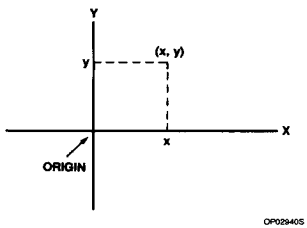


Figure 10. Absolute Coordinate Specification

Relative Coordinate Specification
Specifies the relative addresses (Δx, Δy) related to the current drawing point.

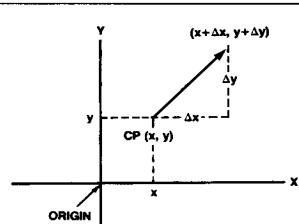


Figure 11. Relative Coordinate Specification

The color mode determines whether the pattern RAM is used indirectly to select color registers or is directly used as the color information.

The operation mode defines one of eight logical operations to be performed between the frame buffer read data and the color data in the pattern RAM to determine the drawing data to be rewritten into the frame buffer. Figure 12 shows the graphic drawing command format.

OPERATION MODE

The operation mode (OPM bits) of the graphic drawing command specify the logical drawing condition.

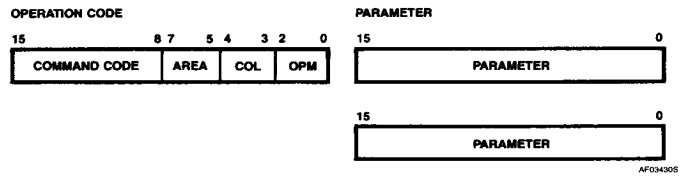


Figure 12. Graphic Drawing Command Format

Figure 13 shows examples of a drawing pattern applied with various OPM modes.

OPM			Operation Mode
0	0	0	REPLACE: * Replaces the frame buffer data with the color data
0	0	1	OR: * ORs the frame buffer data with the color data. The result is rewritten to the frame buffer
0	1	0	AND: * ANDs the frame buffer data with the color data. The result is rewritten to the frame buffer
0	1	1	Ex-OR: * Ex-ORs the frame buffer data with the color data. The result is rewritten to the frame buffer.
1	0	0	CONDITIONAL REPLACE (read data = CCMP): * When the frame buffer data at the drawing position is equal to the comparison color (CCMP), the frame buffer data is replaced with the color data
1	0	1	CONDITIONAL REPLACE (read data ≠ CCMP): * When the frame buffer data at the drawing position is not equal to the comparison color (CCMP), the frame buffer data is replaced with the color data
1	1	0	CONDITIONAL REPLACE (read data < CL): * When the frame buffer data at ** the drawing position is less than the color register data (CL), the frame buffer data is replaced with the color data
1	1	1	CONDITIONAL REPLACE (read data > CL): * When the frame buffer data at ** the drawing position is greater than the color register data (CL), the frame buffer data is replaced with the color data.

* Normally, the color register (CL0 or CL1) selected by the pattern pointer (PPX, PPY) is used for the color data, but the source area data is used in the graphic copy commands (AGCPY and RGCPY).

** Normally, the color register (CL0 or CL1) selected by the pattern pointer (PPX, PPY) is used for the color register data (CL), but the source area data is used in the graphic copy command (AGCPY and RGCPY).

COLOR MODE

The color mode (COL bits) specify the source of the drawing color data as directly or indirectly (using the color registers) determined by the contents of the pattern RAM.

COL		Color Mode
0	0	When pattern RAM data = 0, color register 0 is used When pattern RAM data = 1, color register 1 is used
0	1	When pattern RAM data = 0, drawing is suppressed When pattern RAM data = 1, color register 1 is used
1	0	When pattern RAM data = 0, color register 0 is used When pattern RAM data = 1, drawing is suppressed
1	1	Pattern RAM contents are directly used as color data

The color mode chooses the source for color information based on the contents (0 or 1) of a particular bit in the 16-bit by 16-bit (32-byte) pattern RAM. A sub-pattern is specified by programming the pattern RAM control register (PRC) with the start (PSX, PSY) and end (PEX, PEY) points which define the diagonal of the sub-pattern. Furthermore, a specific starting point for pattern RAM scanning is specified by PPX and PPY (see Figure 14).

Normally, the color registers (CL) should be loaded with on color data based on the number of bits per pixel. For example, if four bits/pixel are used, the 4-bit color pattern (e.g. 0001) should be replicated four times in the color register, as shown below.

Color Register = | 0001 | 0001 | 0001 | 0001 |

In this way, color changes due to changing dot address are avoided.

AREA MODE

Prior to drawing, a drawing 'area' may be defined (area definition register). Then, during graphics drawing operation, the ACRTC will check if the drawing point is attempting to enter or exit the defined drawing area. Based on eight area modes, the ACRTC will take appropriate action for clipping or hitting.

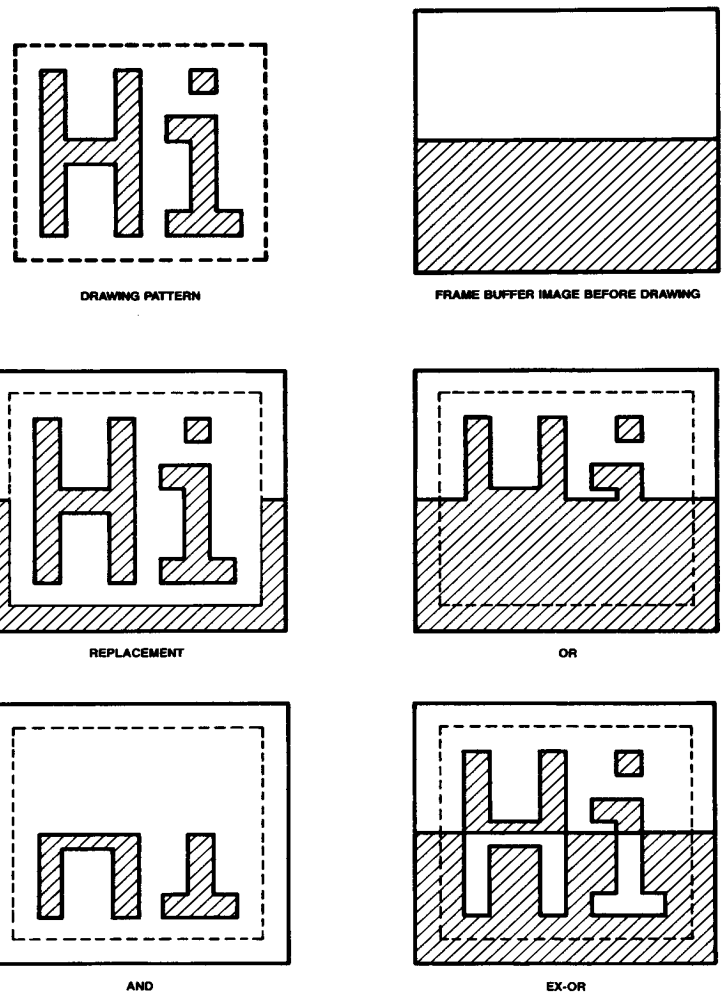


Figure 13. Operation Mode Example

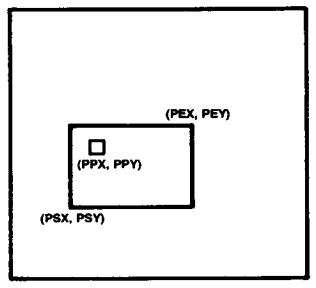


Figure 14. Pattern RAM

AREA			Drawing Area Mode
0	0	0	Drawing is executed without Area checking
0	0	1	When attempting to exit the area, drawing is stopped after setting ABT (abort bit)
0	1	0	Drawing suppressed outside the area — drawing operation continues and the ARD flag is not set
0	1	1	Drawing suppressed outside the area — drawing operation continues and the ARD flag is set at every drawing operation.
1	0	0	Same as area = 000
1	0	1	When attempting to enter the area, drawing is stopped after setting ABT (abort bit)
1	1	0	Drawing suppressed inside the area — drawing operation continues and the ARD flag is not set.
1	1	1	Drawing suppressed inside the area — drawing operation continues and the ARD flag is set at every drawing operation.

SYSTEM INTERFACE

Basic Clock

The ACRTC basic clock is 2CLK. 2CLK controls all primary ACRTC display and logic timing parameters. 2CLK, along with the specification of number of bits per logical pixel, the graphic address increment mode, and the display access mode, also determines the video data rate. The basic clock must be input, noting its cycle, max. and min. of "high" and "low" level width.

In any case, care must be taken not to stop the basic clock, fixing it at "high" or "low", nor to use 2CLK line in open state, which can destroy the LSI.

CRT INTERFACE

Frame Buffer Access

Access Modes — The three ACRTC display memory access modes are single, interleaved and superimposed.

1. Single access mode. A display (or drawing) cycle is defined as two cycles of 2CLK. During the first 2CLK cycle, the frame buffer display or drawing address is output. During the second 2CLK cycle, the frame buffer data is read (display cycles and/or drawing cycles) or written (drawing cycles).
   In this mode, display and drawing cycles contend for access to the frame buffer. The ACRTC allows the priority to be defined as display priority or drawing priority. If display priority, drawing cycles are only allowed to occur during horizontal/vertical flyback period. So, a 'flashless' display is obtained at the expense of slower drawing. If drawing priority, drawing may occur during display so high-speed drawing is obtained, however the display may flash.
2. Interleaved access mode (dual access mode 0). In this mode, display cycles and drawing cycles are interleaved. A display/drawing cycle is defined as four cycles of 2CLK. During the first 2CLK cycle, the frame buffer display address is output. During the second 2CLK cycle, the display data is read from the frame buffer. During the third 2CLK cycle, the frame buffer drawing address is output. During the fourth 2CLK cycle, the drawing data is read or written.
   Since there is no contention between display and drawing cycles, a 'flashless' display is obtained while maintaining full drawing speed. However, for a given configuration, frame buffer memory access time must be twice as fast as an equivalent single access mode configuration.
3. Superimposed access mode (dual access mode 1). In this mode, two separate logical screens are accessed during each display cycle. The display cycle is defined as four 2CLK cycles. During the first 2CLK cycle, the background (upper, base or lower) screen frame buffer address is output. During the second 2CLK cycle, the background screen display data is read. During the third 2CLK cycle, the window screen frame buffer address or the drawing frame buffer address is output. During the fourth 2CLK cycle, the window screen display or drawing data is read (display or drawing) or written (drawing). Note that the third and fourth cycles can be used for drawing (similar to interleaved mode) when these cycles are not used for window display.

Graphic Address Increment Mode (GAI) — During display operation, the ACRTC can be programmed to control the graphic display address in seven ways, including increment by 1, 2, 4, 8 and 16 words, 1 word every two display cycles and no increment.

NOTE:
The SCN63484 (R mask version) does not support 16-word increment mode.

Setting GAI to increment by 2, 4, 8 or 16 words per display cycle achieves linear increases in the video data rate; i.e. for a given configuration setting GAI to 2, 4, 8 or 16 words will achieve 2, 4, 8 or 16 times the video data rate corresponding to GAI = 1. This allows increasing the number of bits/logical pixel and logical pixel resolution while meeting the 2CLK maximum frequency constraint.

Table 7 shows the summary relationship between 2CLK, display access mode, graphic address increment, number of bits/logical pixel, memory access time and video data rate. The frame buffer cycle frequency (f_c) is shown by the following equation where:

f_v = Dot Clock
N = No. bits/logical pixel
D = Display access mode 1 for single access mode 2 for interleaved and superimposed access modes
A = Graphic address increment (½, 1, 2, 4, 8, 16)

$$f_c = (f_v \times N \times D)/(A \times 16)$$

DYNAMIC RAM REFRESH

When dynamic RAMs (DRAMs) are used for the frame buffer memory, the ACRTC can automatically provide DRAM refresh addressing. The ACRTC maintains an 8-bit DRAM refresh counter which is decremented on each frame buffer access. During $\overline{\text{HSYNC}}$ low, the ACRTC will output the sequential refresh addresses on MAD. The refresh address assignment depends on graphic address increment (GAI) mode as shown in Table 8. The ACRTC provides "0" output on the remaining address line of MAD and MA/RA.

DRAM refresh cycle timing must be factored into the determination of $\overline{\text{HSYNC}}$ low pulse width (HSW – specified in units of frame buffer memory cycles). If the horizontal scan rate if f_h (kHz), number of DRAM refresh cycles is N and the DRAM refresh cycle time is t_r (ms) then horizontal sync width (HSW) is specified by the following equation:

$$HWY \geq N \ / \ (t_r \times f_h)$$

Table 7. Graphic Address Increment Modes

DOT RATE ACCESS MODE		16MHz		32MHz		64MHz		128MHz	
Color No. (Bit/Pixel)	Memory Cycle	S	D	S	D	S	D	S	D
1	250ns	—	+½	+½	+1	+1	+2	+2	+4
	500ns	+½	+1	+1	+2	+2	+4	+4	+8
2	250ns	+½	+1	+1	+2	+2	+4	+4	+8
	500ns	+1	+2	+2	+4	+4	+8	+8	+16
4	250ns	+1	+2	+2	+4	+4	+8	+8	+16
	500ns	+2	+4	+4	+8	+8	+16	+16	—
8	250ns	+2	+4	+4	+8	+8	+16	+16	—
	500ns	+4	+8	+8	+16	+16	—	—	—
16	250ns	+4	+8	+8	+16	+16	—	—	—
	500ns	+8	+16	+16	—	—	—	—	—

For example, if the scan rate is 15.75kHz and the DRAMs have 128 refresh cycles of 2ms, HSW must be greater than or equal to 5.

$HSW \geq 128 / (2 \times 15.75) = 4.06$

EXTERNAL SYNCHRONIZATION

The ACRTC $\overline{EXSYNC}$ pin allows synchronization of multiple ACRTCs or other video signal generators. The ACRTC may be programmed as a single master device, or as one of a number of slave devices. To synchronize multiple ACRTCs, simply connect all the $\overline{EXSYNC}$ pins together.

For synchronizing to other video signals, the connection scheme depends on the raster scan mode. In non-interlace mode, $\overline{EXSYNC}$ corresponds to $\overline{VSYNC}$. In interlace modes, $\overline{EXSYNC}$ corresponds to VSYNC of the odd field (see Figure 15).

NOTES:

1. The ACRTC performs the synchronization every time it accepts the pulse input from $\overline{EXSYNC}$ in the slave mode. It is recommended that the synchronous pulse should be input from $\overline{EXSYNC}$ only when the synchronization gap between the synchronous signal of the master device and that of ACRTC is in the slave mode. $\overline{HSYNC}$ and $\overline{VSYNC}$ are also output in the slave mode.
2. The ACRTC needs to be controlled not to execute the drawing operation during $\overline{EXSYNC}$ input.

Table 8. GAI and DRAM Refresh Addressing

ADDRESS INCREMENT MODE	REFRESH ADDRESS OUTPUT TERMINAL
+0 (GAI = 101)	MAD0 – MAD7
+1 (GAI = 000)	MAD0 – MAD7
+2 (GAI = 001)	MAD1 – MAD8
+4 (GAI = 010)	MAD2 – MAD9
+8 (GAI = 011)	MAD3 – MAD10
+16 (GAI = 100)	MAD4 – MAD11
+½ (GAI = 111)	MAD0 – MAD7
(GAI = 110)	

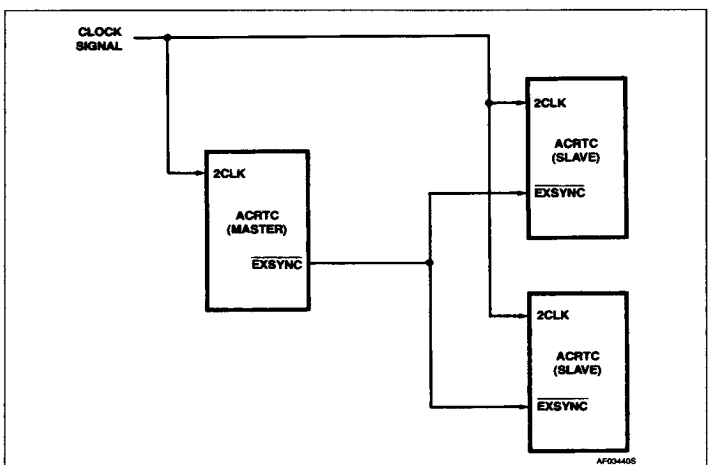


Figure 15. External Synchronization

MPU INTERFACE

MPU Bus Cycle

The ACRTC interfaces to the MPU as a peripheral occupying two addresses in the MPU address space. The ACRTC can operate as an 8- or 16-bit peripheral as configured during $\overline{RES}$. An MPU bus cycle is initiated when $\overline{CS}$ is asserted (following the assertion of RS and R/$\overline{W}$). The ACRTC responds to $\overline{CS}$ low by asserting $\overline{DTACK}$ low to complete the data transfer. $\overline{DTACK}$ will be returned to the MPU in between 1 and 1.5 2CLK cycles.

MPU WAIT states will be added in the following two cases.

1. If the ACRTC 2CLK input is much slower than the MPU clock, continuous ACRTC accesses may be delayed due to internal processing of the previous bus cycle. Be careful of $\overline{CS}$ "high" width.
2. If an ACRTC read cycle immediately follows an ACRTC write cycle, a wait state may occur due to ACRTC prepara-

tion for bus 'turn-around'. However, (e.g. 68000 system) MPUs normally have no instructions which immediately follow a write cycle with a read cycle.

For connection to synchronous bus interface MPUs, $\overline{DTACK}$ can simply be left open, assuming the system design guarantees that WAIT states cannot occur as described above. If WAIT states may occur, $\overline{DTACK}$ can be used with external logic to synthesize a READY signal.

DMA TRANSFER

The ACRTC can interface with an external DMA controller using three handshake signals. DMA request ($\overline{DREQ}$), DMA acknowledge ($\overline{DACK}$) and DMA done ($\overline{DONE}$). The ACRTC uses the external DMAC for two types of transfers, command/parameter DMA and data DMA. For both types, DMA transfers use the ACRTC read and write FIFOs.

Command/Parameter DMA

The MPU initiates this mode by setting bit 12 (CDM) in the ACRTC command control register to 1. Then, the ACRTC will automatically request DMA transfer for commands and their associated parameters as long the write FIFO has space. Only cycle steal request mode ($\overline{DREQ}$ pulses low for each data transfer) can be used.

Command/parameter DMA is terminated when the MPU resets bit 12 in CCR to 0, or the external $\overline{DONE}$ input is asserted. Note that the R mask version and the S mask version cannot perform command/parameter DMA transfer, so CDM (bit 12) should be set to 0.

Data DMA

Data DMA is used to move data between the MPU system memory and the ACRTC frame buffer. The MPU sets up the transfer by specifying the frame buffer transfer address (and other parameters of the transfer, such as 'on-the-fly' logical operations) to the ACRTC. Next, when the MPU issues a data transfer command to the ACRTC, the ACRTC will request DMA transfer to and from system memory. The ACRTC will request DMA, automatically monitoring FIFO status, until the DMA transfer command is completed.

Data DMA request mode can be cycle steal (as in command/parameter DMA), or burst mode in which $\overline{DREQ}$ is a low-level control output to the DMAC which allows multiple data transfers during each acquisition of the MPU bus.

INTERRUPTS

The ACRTC recognizes eight separate conditions which can generate an interrupt, including command error detection, command end,

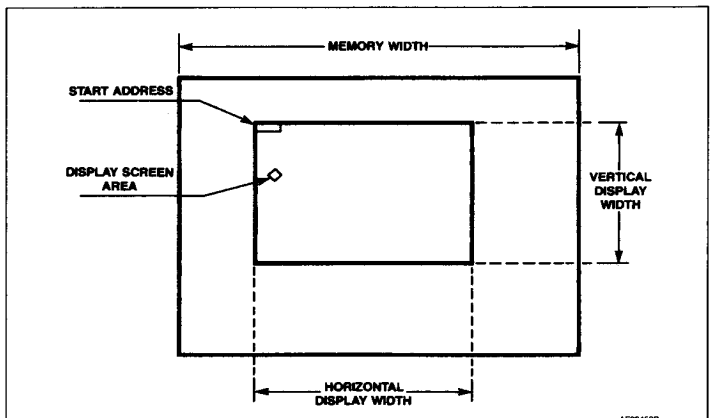


Figure 16. Display Screen/Frame Buffer Relationship

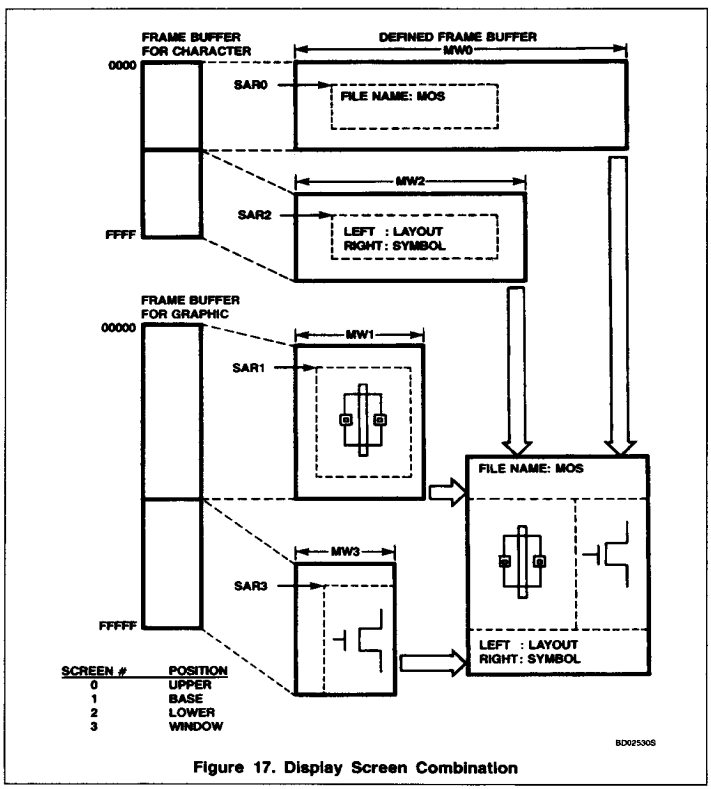


Figure 17. Display Screen Combination

drawing edge detection, light pen strobe, and four FIFO status conditions. Each condition has an associated mask bit for enabling/disabling the associated interrupt. The ACRTC removes the interrupt request when the MPU performs appropriate interrupt service by reading or writing to the ACRTC.

DISPLAY FUNCTION

Logical Display Screens

The ACRTC allows division of the frame buffer into four separate logical screens.

Screen Number	Screen Name	Screen Group Name
0	Upper Screen	Background Screens
1	Base Screen	
2	Lower Screen	
3	Window Screen	

In the simplest case, only the base screen parameters must be defined. Other screens may be selectively enabled, disabled and blanked under software control.

The background (upper, base, and lower) screens partition the display into three horizontal splits whose position is fully-programmable. A typical application might use the base screen for the bulk of user interaction, using the lower screen for 'status line(s)' and the upper screen for 'pull-down menu(s)'.

The window screen is unique, since the ACRTC gives the window screen higher priority than background screens. Thus, when the window, whose size and position is fully-programmable, overlaps a background screen, the window screen is displayed. One exception is the ACRTC superimposed access mode, in which the window has the same display priority as background screens. In this case, the window and background screen are 'superimposed' on the display. The ACRTC logical screen organization can be programmed to best suit a number of display applications. See Figures 16 through 18.

GRAPHIC/CHARACTER ADDRESS SPACES

The ACRTC controls two separate logical address spaces. The CHR pin allows external decoding if physically-separate frame buffers are desired. Each of the four logical screens (upper, base, lower, and window) is programmed as residing in the graphics address space or the character address space (see Figure 19).

ACRTC accesses to graphics screens are treated as bit-mapped using a 20-bit frame buffer address, with an address space of one megaword (1M by 16 bits).

ACRTC accesses to character screens are treated as character generator-mapped. In this case, a 64k word address space is used and five bits of raster address are output to an external character generator (see Figure 20).

Multiple logical screens defined as character can be externally decoded to use separate character generators or different addresses within a combined character generator. Also, each character screen may be defined with separate line spacing, separate cursors, etc.

CURSOR CONTROL

The ACRTC has two block cursor registers and a graphics cursor register. A block cursor is used with character screens. The cursor start and ending raster addresses are fully-programmable. Also, the cursor width can be defined as one to eight memory cycles.

A graphics cursor is defined by specifying the start/end memory in cycle the X dimension and the start/end raster in the Y dimension. The graphic cursor can output on character screens. The ACRTC provides two separate cursor outputs, $\overline{CUD1}$ and $\overline{CUD2}$. These are combined with two character cursor registers and a graphics cursor register to provide three cursor modes.

Block Mode

Two block cursors are output on $\overline{CUD1}$ and $\overline{CUD2}$, respectively (see Figures 21 and 22).

Graphic Mode

The graphic cursor is output on CUD1. Using an external cursor pattern memory allows a graphic cursor of various shapes. Two block cursors are multiplexed on CUD2 (see Figure 23).

Crosshair Mode

The horizontal and vertical components of the graphic cursor are output on CUD1 and CUD2, respectively. This allows simple generation of a crosshair cursor control signal (see Figure 24).

SCROLLING

Vertical Scroll

Each logical screen performs independent vertical scroll. On character screens, vertical smooth scroll is accomplished using the programmable start address raster (SAR). Line-by-line scroll is accomplished by increasing or decreasing the screen start address by one unit of horizontal memory width.
On graphics screens, vertical smooth scroll is accomplished by increasing or decreasing the screen start address by one unit of horizontal memory width (see Figure 25).

Horizontal Scroll

Horizontal scroll can be performed in units of characters for character screens and units of words (multi-logical pixels) for graphic screens by increasing or decreasing the screen start address by 1. For smooth horizontal scroll, the ACRTC has dot shift video attributes which can be used with an external circuit which conditions shift register load/clocking.

Since this dot shift information is output each raster, horizontal smooth scroll is limited to either the background screens or the window screen at any given time. However, horizontal smooth scroll is independent for each of the background screens (upper, base, lower). See Figures 25 through 27.

RASTER SCAN MODES

The ACRTC has three software-selectable raster scan modes — non-interlace, interlace sync, and interlace sync and video. In non-interlace mode, a frame consists of one field. In the interlace modes, a frame consists of two fields, the even and odd fields.

The interlace modes allow increasing screen resolution while avoiding limits imposed by the CRT display device, such as maximum horizontal scan frequency or maximum video dot rate.

Interlace sync mode simply repeats each raster address for both the even and odd fields. This is useful for increasing the quality of a displayed figure when using an interlaced CRT device such as a television set with RF modulator.

Interlace sync and video mode displays alternate even and odd rasters on alternate even and odd fields. For a given number of rasters/character, this mode allows twice as many characters to be displayed in the vertical direction as non-interlace mode (see Figure 28).

Note that for interlace modes, the refresh frequency for a given dot on the screen is one-half that of the non-interlace mode. Interlace modes normally require the use of a CRT with a more persistent phosphor to avoid flickering display.

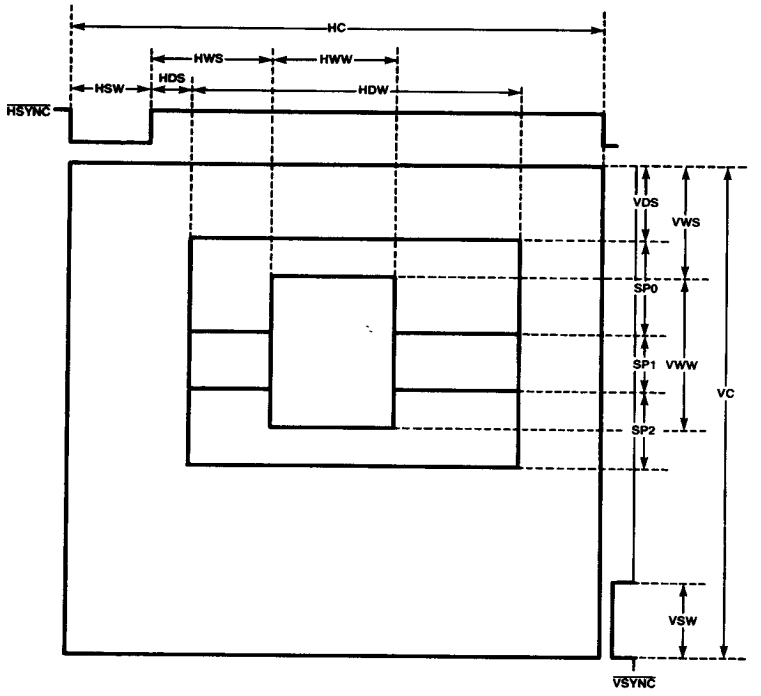


Figure 18. Display Screen Specification

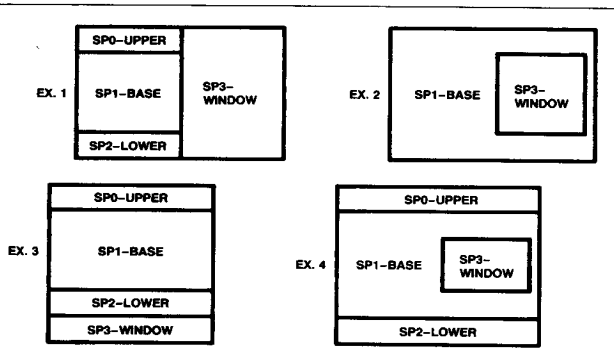


Figure 19. Screen Combination Example

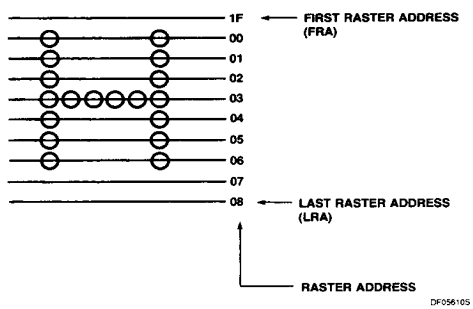


Figure 20. Character Screen Raster Addressing

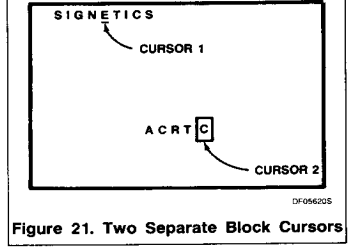


Figure 21. Two Separate Block Cursors

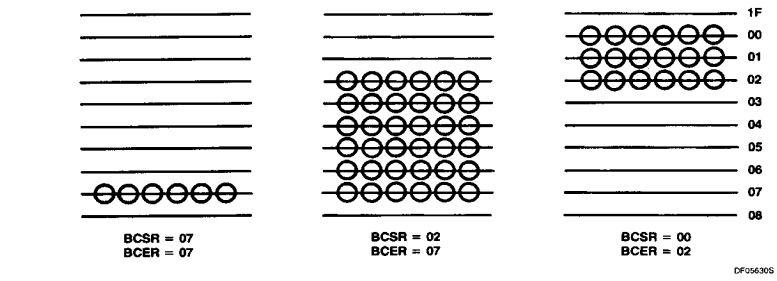


Figure 22. Block Cursor Examples

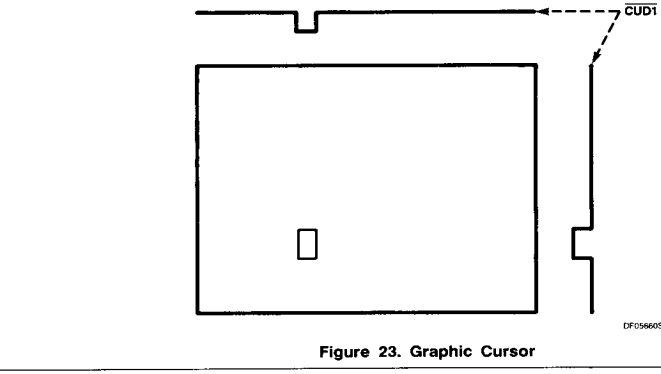


Figure 23. Graphic Cursor

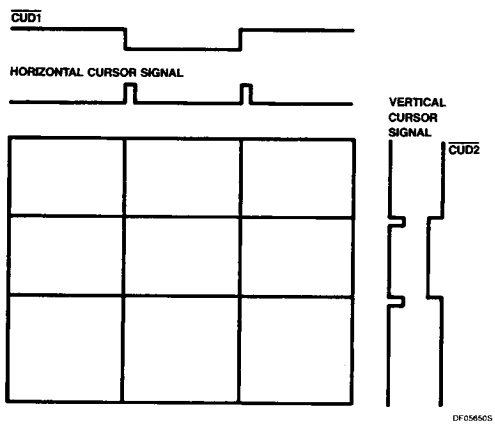


Figure 24. Crosshair Cursor

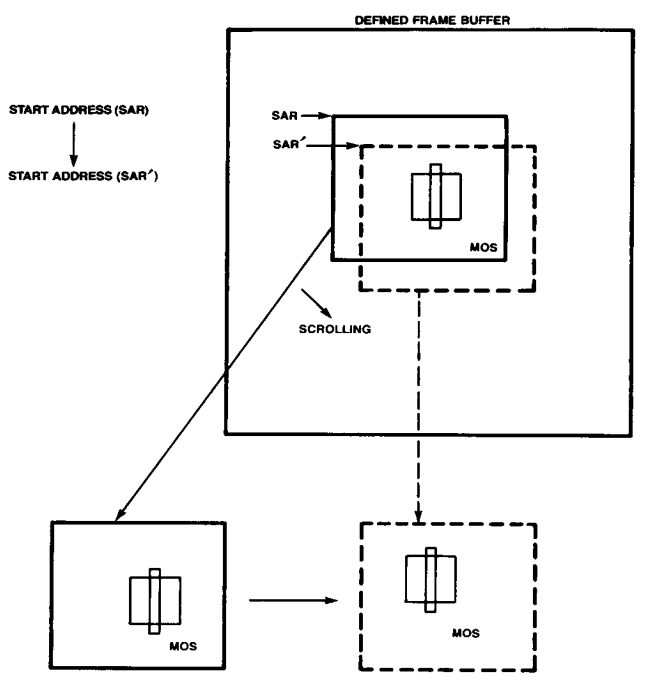


Figure 25. Scrolling By SAR (Start Address Register) Rewrite

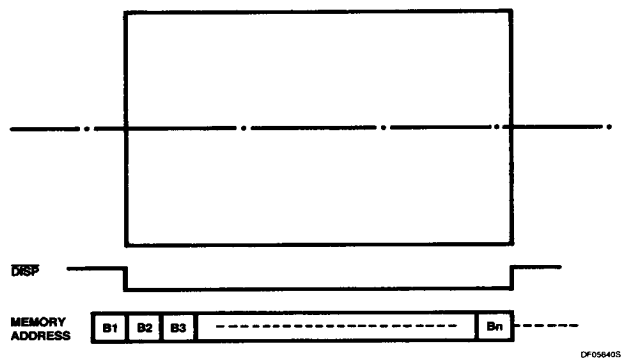


Figure 26. Horizontal Smooth Scroll — Base Screen

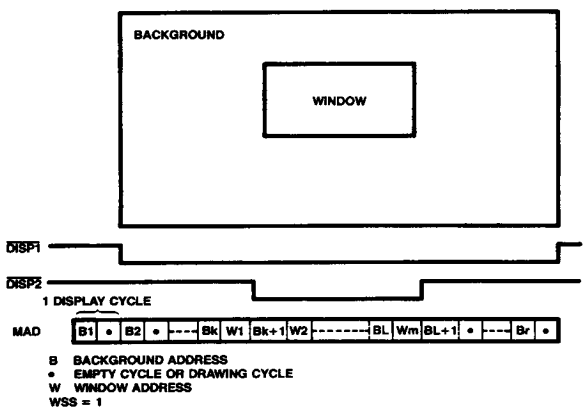


Figure 27. Horizontal Smooth Scroll — Window Screen

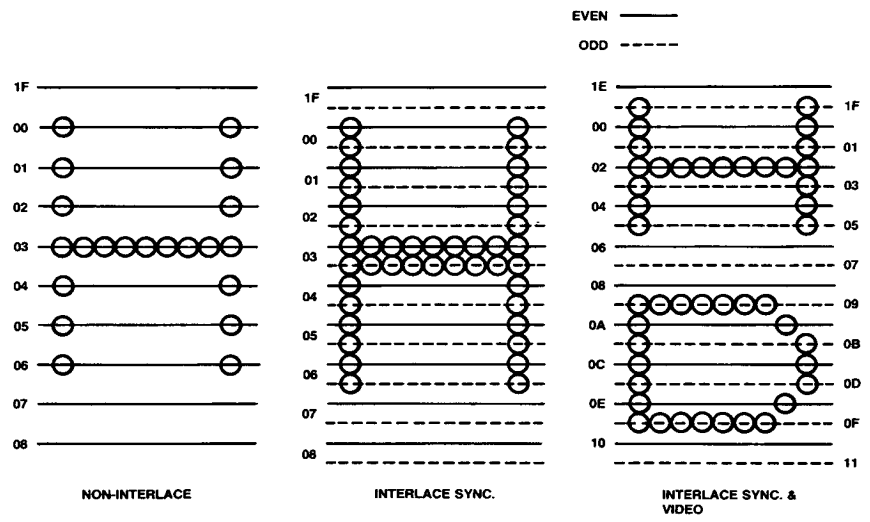


Figure 28. Raster Screen Modes

ZOOMING

The base screen (screen 1) is supported by the ACRTC zooming function. Note that ACRTC zooming is performed by controlling the CRT timing signals. The contents of the frame buffer area being zoomed are not changed. The ACRTC allows specification of a zoom factor (1 to 16) independently in the X and Y directions.

For horizontal zoom, the programmed zoom factor is output as video attributes. An external circuit uses this factor to condition the external shift register clock to accomplish horizontal zooming.

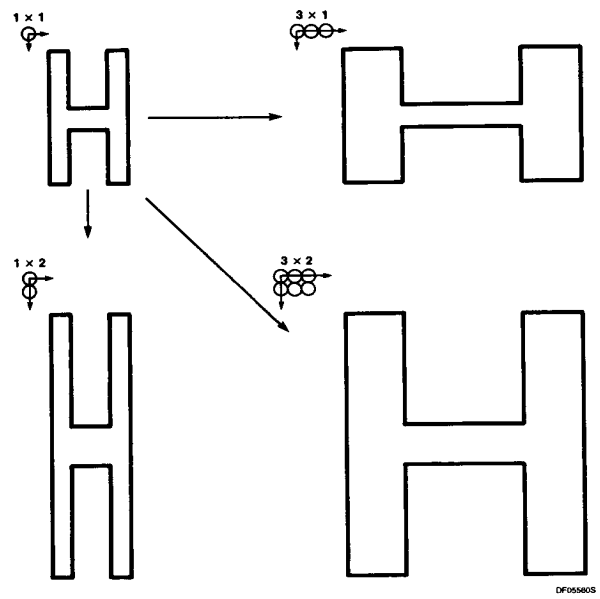
For vertical zoom, no external circuit is required. The ACRTC will scan a single raster multiple times to accomplish vertical zooming (see Figure 29).

LIGHT PEN

The ACRTC provides a 20-bit light pen address register and a light pen strobe (DPSTB) input pin for connection with a light pen.

A light pen strobe pulse will occur when the CRT electron beam passes under the light pen during display refresh. When this pulse occurs, the contents of the ACRTC display refresh address counter will be latched into the light pen address register along with a logical screen (character or graphic screen) designator. Also, an ACRTC status flag indicating light pen activity is set, generating an optional (maskable) MPU interrupt. Note that for superimposed access mode, when the light pen strobe occurs in an area in which the window overlaps a background (upper, base, or lower) screen, the background screen address will be latched. And even for all access modes, the drawing address will be latched.

Various system and ACRTC delays will cause the latched address to differ slightly from the actual light pen position. The light pen address can be corrected using software, based on system-specific delays. Or, if the application does not require the highest light pen pointing resolution, software can 'bind' the light pen address by specifying a range of values associated with a given area of the screen.



NOTES:
1. "U" mask limitation:
2. — when the parameters "a, b, and DX" for the ellipse command satisfy the following condition, $a \geq 2b(2DX - 1)$, the command fails to end and the ACRTC goes into a deadlock, resulting in a Flat Ellipse.

Figure 29. Zooming

ABSOLUTE MAXIMUM RATINGS[1]

SYMBOL	PARAMETER	RATING	UNIT
V_S	Supply voltage[2] range	−0.3 to +7.0	V
V_{IN}	Input voltage[2] range	−0.3 to V_{CC} + 0.3	V
I_{OUT}	Allowable output current[3]	+5	mA
I_{OUT}	Total allowable output current[4]	+120	mA
T_A	Operating temperature range	0 to +70	°C
T_{STG}	Storage temperature range	−55 to +150	°C

RECOMMENDED OPERATING CONDITIONS

SYMBOL	PARAMETER	MIN	TYP	MAX	UNIT
V_S	Supply voltage[2]	4.75	5	5.25	V
V_{IL}	Input low level voltage[2]	0	–	0.7	V
V_{IH}	Input high level voltage[2]	2.2	–	V_{CC}	V
T_A	Operating temperature	0	25	70	°C

NOTES:
1. Using an LSI beyond its maximum ratings may result in permanent destruction. LSIs should usually be used under recommended operating conditions. Exceeding any of these conditions may adversely affect the LSI's reliability.
2. This value is in reference to V_{SS} = 0V.
3. The allowable output current is the maximum current that may be drawn from, or flow out to one output terminal or one input/output common terminal.
4. The total allowable output current is the total sum of currents that may be drawn from, or flow out to output terminals or input/output common terminals.

TIMING MEASUREMENT

The timing measurement point for the output "low" level is defined at 0.8V throughout this specification. The output "low" level at stable condition (DC characteristics) is defined at 0.5V. The output "high" level is defined at V_{CC} − 2.0V (see Figure 30).

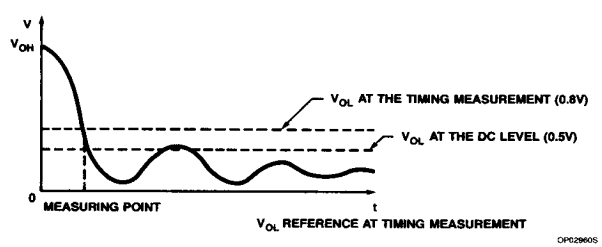


Figure 30. Timing Measurement

DC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ±5%, V_{SS} = 0V, T_A = 0 to +70°C unless otherwise noted.

PARAMETER		TEST CONDITIONS	8MHz		UNIT
			Min	Max	
Input high level	All inputs		2.2	V_{CC}	V
Input low level	All inputs		−0.3	0.7	V
Input leakage current	R/W, $\overline{CS}$, RS, $\overline{RES}$, $\overline{DACK}$ 2CLK, LPSTB	V_{IN} = 0 − V_{CC}	−2.5	2.5	μA
3-State (off-state) input current	D0 − D15, $\overline{EXSYNC}$, MAD0 − MAD15	V_{IN} = 0.4 − V_{CC}	−10	10	μA
Output high level	D0 − D15, MAD0 − MAD15, $\overline{CUD1}$, $\overline{CUD2}$ $\overline{DREQ}$, $\overline{DTACK}$, $\overline{HSYNC}$, $\overline{VSYNC}$, $\overline{EXSYNC}$	I_{OH} = −400μA	V_{CC} −1	—	V
Output low level	$\overline{DISP1}$, $\overline{DISP2}$ CHR, MRD, $\overline{DRAW}$, $\overline{AS}$, MCYC, RA4, MA16/RA0, MA19/RA3	I_{OL} = 2.2mA	—	0.5	V
Output leakage current (off-state)	$\overline{IRQ}$, $\overline{DONE}$	V_{OH} = V_{CC}	—	10	μA
Input capacity	D0 − D15, $\overline{EXSYNC}$, MAD0 − MAD15 R/$\overline{W}$, $\overline{CS}$, RS, $\overline{RES}$, $\overline{DACK}$, 2CLK, LPSTB	V_{IN} = 0V, T_A = 25°C, f = 1.0MHz	—	17	pF
Output capacity	$\overline{IRQ}$, $\overline{DONE}$	V_{IN} = 0V, T_A = 25°C, f = 1.0MHz	—	15	pF
Current Consumption		Chip not selected Display in progress Data bus in read/write operation Display in progress Command execution in progress	— —	100 100	mA mA

AC ELECTRICAL CHARACTERISTICS V_{CC} = 5V ±5%, V_{SS} = 0V, T_A = 0 to +70°C unless otherwise noted.

NO.	FIGURE	CHARACTERISTIC	TENTATIVE LIMITS 8MHz		UNIT
			Min	Max	
		Operation frequency of 2CLK	1	8	MHz
1	41	Clock cycle time	125	1000	ns
2	41	Clock high level pulse width	55	500	ns
3	41	Clock low level pulse width	55	500	ns
4	41	Clock rise time	—	10	ns
5	41	Clock fall time	—	10	ns
6	42, 43	R/$\overline{W}$ setup time	50	—	ns
7	42, 43	R/$\overline{W}$ hold time	0	—	ns
8	42, 43	RS setup time	50	—	ns
9	42, 43	RS hold time	0	—	ns
10	42, 43	$\overline{CS}$ setup time	40	—	ns
11	42, 43	$\overline{CS}$ high level width	60	—	ns
13	42	Read wait time	0	—	ns
14	42	Read data access time	—	80	ns
15	42	Read data hold time	10	—	ns
16	42	Read data turn off time	—	60	ns
17	42, 43	$\overline{DTACK}$ delay time (Z to L)	—	70	ns
18	42	$\overline{DTACK}$ delay time (D to L)	0	—	ns
19	42, 43	$\overline{DTACK}$ release time (L to H)	—	80	ns
20	42, 43	$\overline{DTACK}$ turn off time (H to Z)	—	100	ns
21	42	Data bus 3-State recovery time 1	0	—	ns
22	43	Write wait time	0	—	ns
23	43	Write data setup time	40	—	ns
24	43	Write data hold time	10	—	ns
25	44, 45	$\overline{DREQ}$ delay time 1	—	110	ns
26	44, 45	$\overline{DREQ}$ delay time 2	—	70	ns
27	44, 45	DMA R/$\overline{W}$ setup time	50	—	ns
28	44, 45	DMA R/$\overline{W}$ hold time	0	—	ns
29	44, 45	$\overline{DACK}$ setup time	40	—	ns
30	44, 45	$\overline{DACK}$ high level width	60	—	ns
32	44	DMA read wait time	0	—	ns
33	44	DMA read data access time	—	80	ns
34	44	DMA read data hold time	10	—	ns
35	44	DMA read data turn off time	—	60	ns
36	44, 45	DMA $\overline{DTACK}$ delay time (Z to L)	—	70	ns
37	44	DMA $\overline{DTACK}$ delay time (D to L)	0	—	ns
38	44, 45	DMA $\overline{DTACK}$ release time (L to H)	—	80	ns
39	44, 45	DMA $\overline{DTACK}$ turn off time (H to Z)	—	100	ns
40	44, 45	$\overline{DONE}$ output delay time	—	70	ns

2

AC ELECTRICAL CHARACTERISTICS (Continued)

NO.	FIGURE	CHARACTERISTIC	TENTATIVE LIMITS 8MHz		UNIT
			Min	Max	
41	44, 45	$\overline{DONE}$ output turn off time (L to Z)	—	80	ns
42	44	Data bus 3-State recovery time 2	0	—	ns
43	44, 45	$\overline{DONE}$ input pulse width	2	—	Clk Cyc
44	45	DMA write wait time	0	—	ns
45	45	DMA write data setup time	40	—	ns
46	45	DMA write data hold time	10	—	ns
48	46 – 49	$\overline{AS}$ low level pulse width	25	—	ns
49	47, 48	Memory address hold time 2	10	—	ns
50	46 – 49	$\overline{AS}$ delay time 1	—	60	ns
51	46 – 49	$\overline{AS}$ delay time 2	—	60	ns
52	46 – 49	Memory address delay time	—	70	ns
53	46 – 49	Memory address hold time 1	10	—	ns
54	46, 47, 49	Memory address turn off time (A to Z)	—	50	ns
55	47	Memory address data setup time	40	—	ns
56	47	Memory read data hold time	10	—	ns
57	46 – 49	MA/RA delay time	—	80	ns
58	46 – 48	MA/RA hold time	10	—	ns
59	46 – 50	MCYC delay time	—	50	ns
60	46 – 49	MRD delay time	—	70	ns
61	46 – 49	MRD hold time	10	—	ns
62	46 – 49	$\overline{DRAW}$ delay time	—	70	ns
63	46 – 49	$\overline{DRAW}$ hold time	10	—	ns
64	48	Memory write data delay time	—	70	ns
65	48	Memory write data hold time	10	—	ns
67	49 – 51	$\overline{HSYNC}$ delay time	—	70	ns
68	50	$\overline{VSYNC}$ delay time	—	70	ns
69	50	$\overline{DISP1}$, $\overline{DISP2}$ delay time	—	70	ns
70	50	$\overline{CUD1}$, $\overline{CUD2}$ delay time	—	70	ns
71	50	$\overline{EXSYNC}$ output delay time	20	70	ns
72	50	CHR delay time	—	70	ns
75	51	$\overline{EXSYNC}$ input pulse width	3	—	Clk Cyc
76	51	$\overline{EXSYNC}$ input setup time 1	50	—	ns
77	51	$\overline{EXSYNC}$ input hold time	30	—	ns
78	52	LPSTB uncertain time 1	70	—	ns
79	52	LPSTB uncertain time 2	10	—	ns
80	52	LPSTB input hold time	10	—	ns
81	52	LPSTB input inhibit time	4	—	Clk Cyc
82	53	$\overline{DACK}$ setup time for $\overline{RES}$	100	—	ns
83	53	$\overline{DACK}$ hold time for $\overline{RES}$	0	—	ns

AC ELECTRICAL CHARACTERISTICS (Continued)

NO.	FIGURE	CHARACTERISTIC	TENTATIVE LIMITS 8MHz Min	TENTATIVE LIMITS 8MHz Max	UNIT
84	53	$\overline{RES}$ input pulse width	10	—	Clk Cyc
85	54	$\overline{IRQ}$ delay time 1	—	150	ns
86	54	$\overline{IRQ}$ delay time 2	—	500	ns
87	49	ATR delay time 1	—	80	ns
88	49	ATR hold time 1	10	—	ns
90	49	ATR delay time 2	—	80	ns
91	49	ATR hold time 2	10	—	ns
100	42, 43	$\overline{CS}$ cycle time	4	—	Clk Cyc
101	42, 43	$\overline{CS}$ low level width	2	—	Clk Cyc
102	42, 43	$\overline{CS}$ high level width	2	—	Clk Cyc
104	44 45	$\overline{DACK}$ cycle time	4	—	Clk Cyc
105	44, 45	$\overline{DACK}$ low level width	2	—	Clk Cyc
106	44, 45	$\overline{DACK}$ high level width	2	—	Clk Cyc

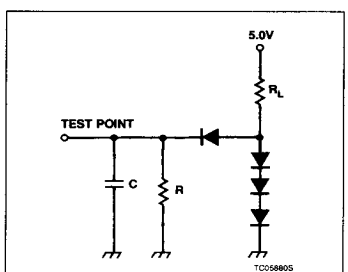


Figure 31. Test Load Circuit A

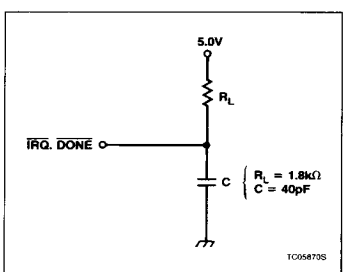


Figure 32. Test Load Circuit B

Signal	Load Condition
D0 – D15	R_L = 1.84Ω
$\overline{DTACK}$	C = 40pF
$\overline{DREQ}$	
MAD0 – MAD15	
MA16/RA0 – MA19/ RA3	
RA4	R = 10kΩ
$\overline{VSYNC}$, $\overline{HSYNC}$	
$\overline{EXSYNC}$	All diodes are 1S2074 or equivalent
MCYC, $\overline{AS}$, MRD	
$\overline{DRAW}$, CHR	
$\overline{DISP1}$, $\overline{DISP2}$	
$\overline{CUD1}$, $\overline{CUD2}$	

Power-On Sequence

The following condition needs to be satisfied when the power turns on.

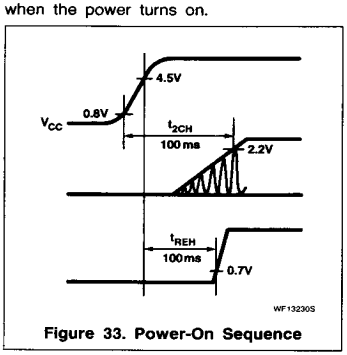


Figure 33. Power-On Sequence

Output Waveform

In case that ringing noise occurs beyond tolerance on the CRT data buses (MAD0 – MAD15, MA16/RA0 – MA19/RA2, RA4), damping resistors may be required for data buses as shown in Figures 34 and 35.

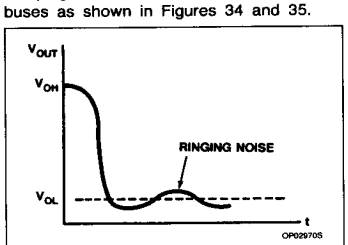


Figure 34. Ringing Noise

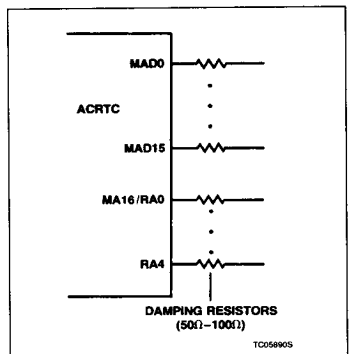


Figure 35. Damping Resistors

Power Supply Circuit

When designing the V_{CC} and V_{SS} pattern of the circuit board, the capacitors need to be located nearest to pin 14 (V_{CC}) and pin 16 (V_{SS}) or pin 51 (V_{SS}) and pin 49 (V_{CC}), as shown in Figure 36.

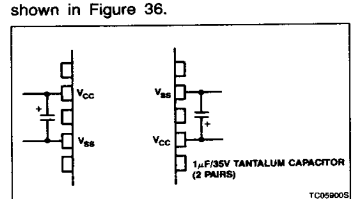


Figure 36. Designing Power Supply Current

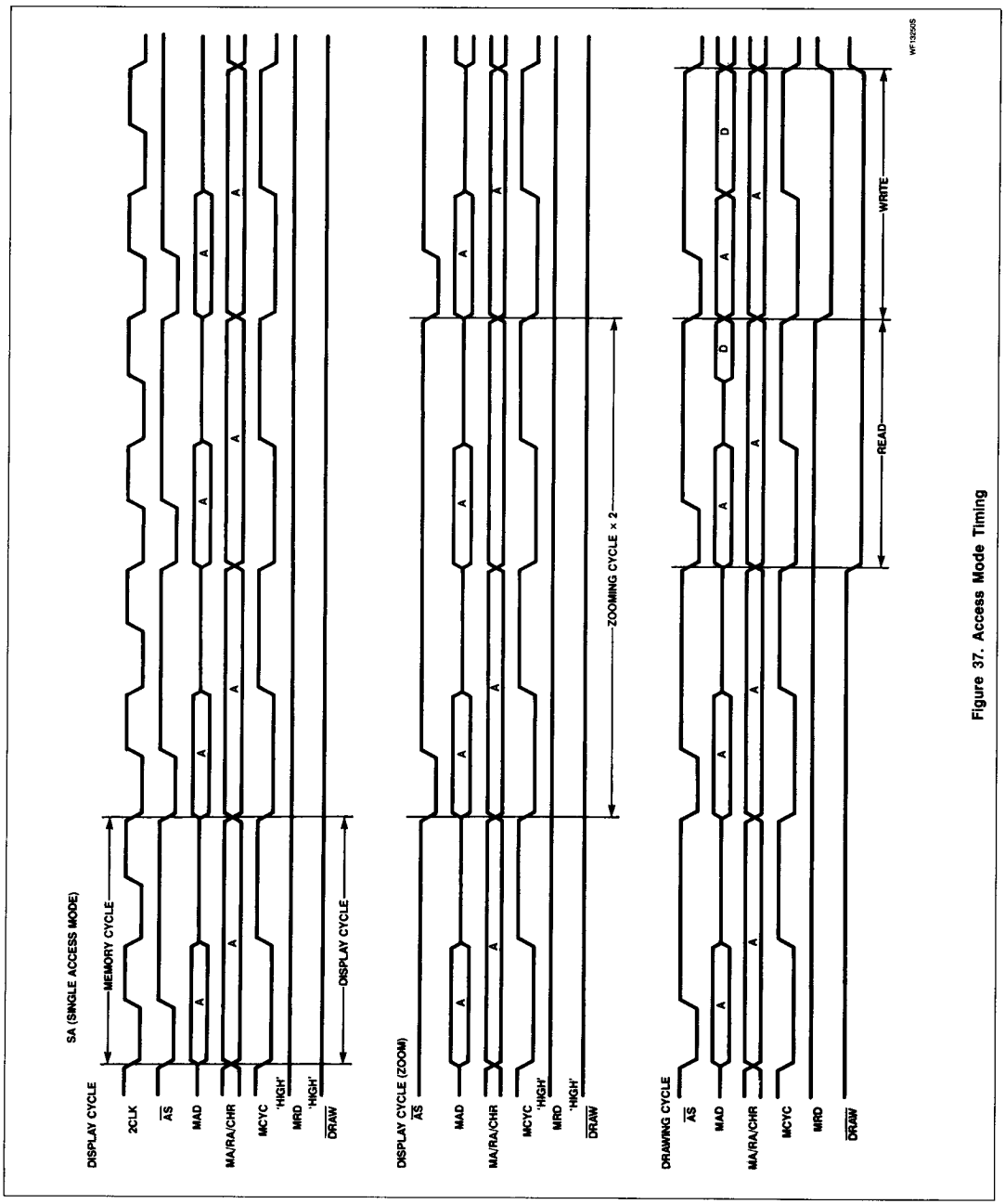


Figure 37. Access Mode Timing

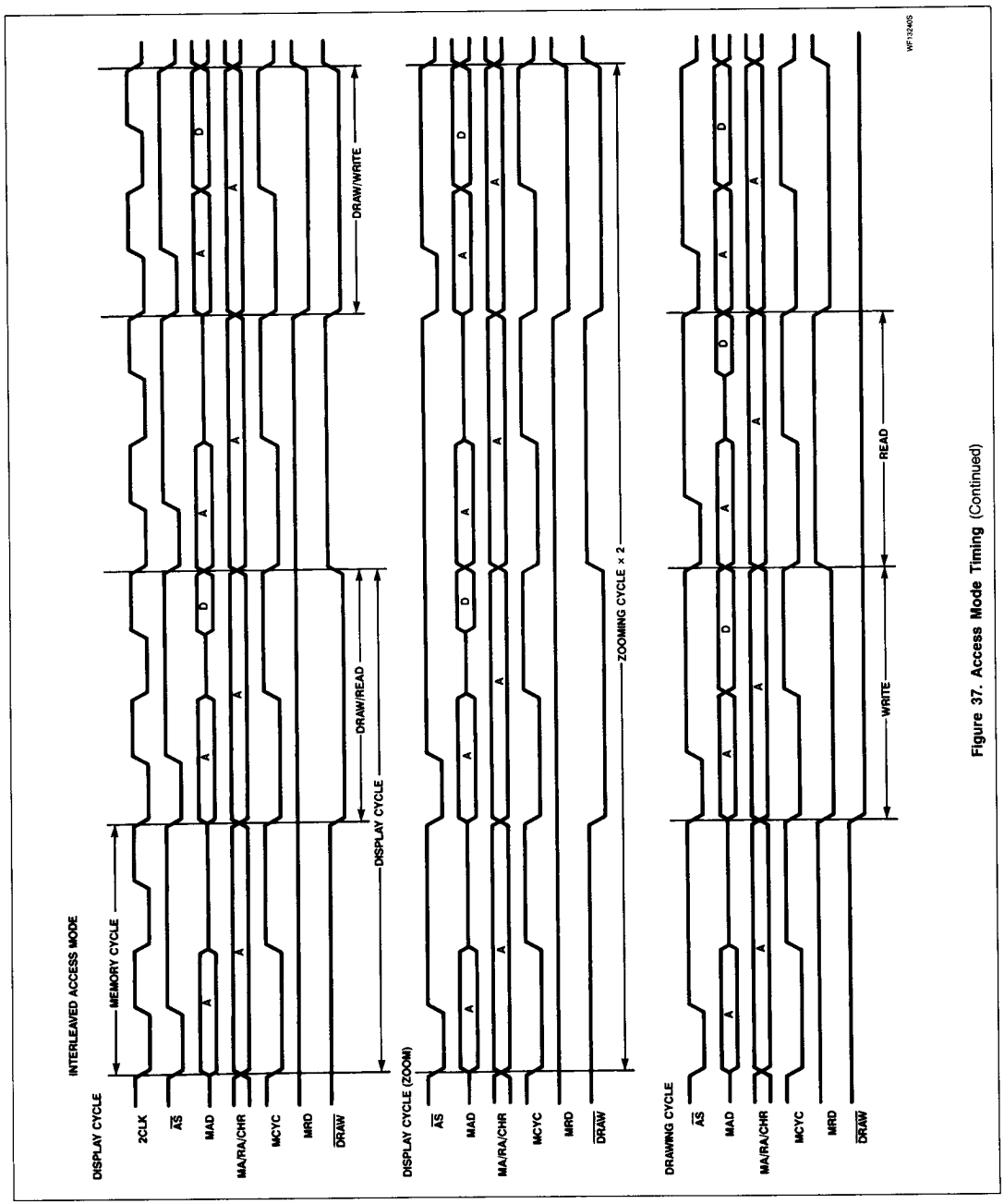


Figure 37. Access Mode Timing (Continued)

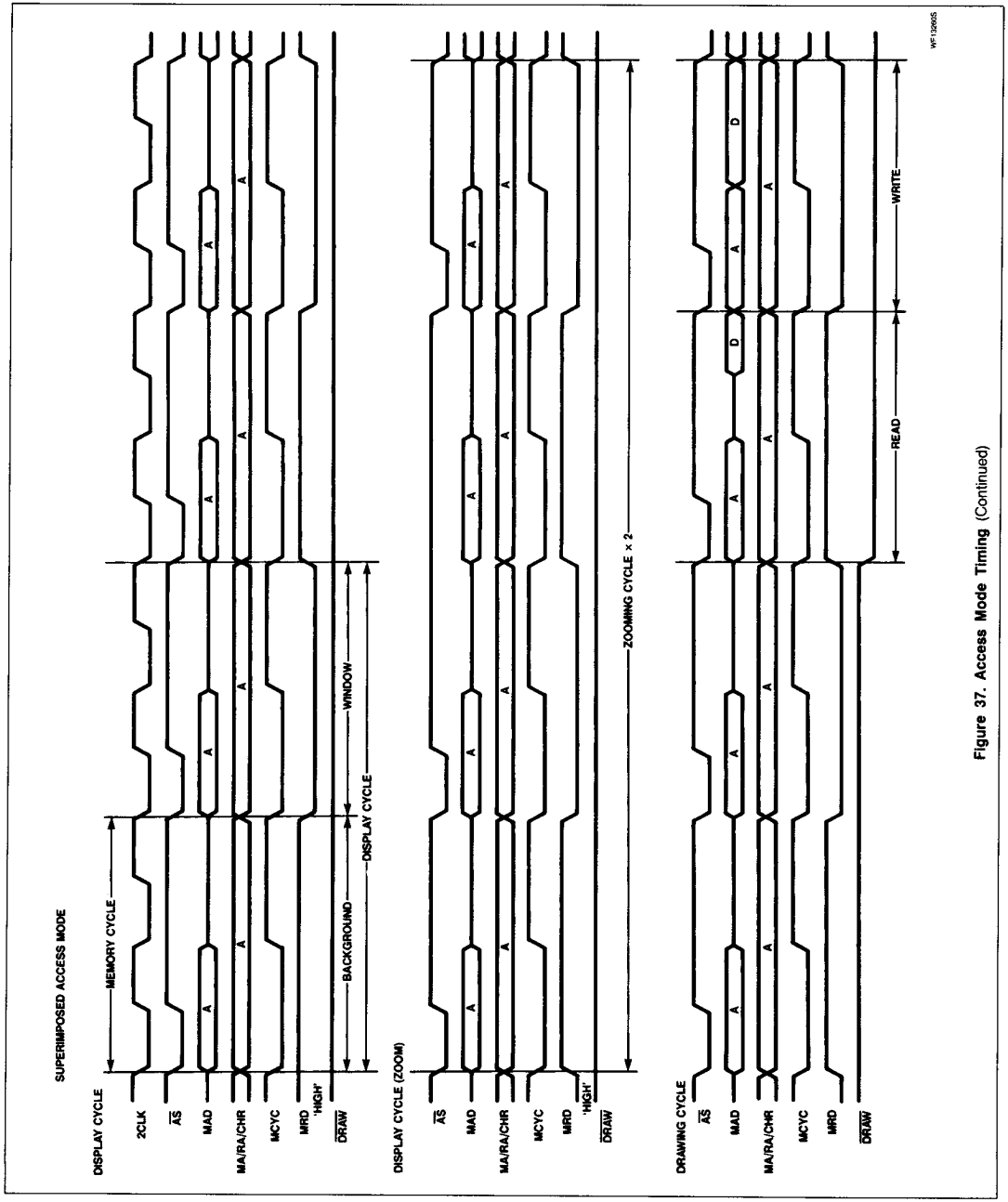


Figure 37. Access Mode Timing (Continued)

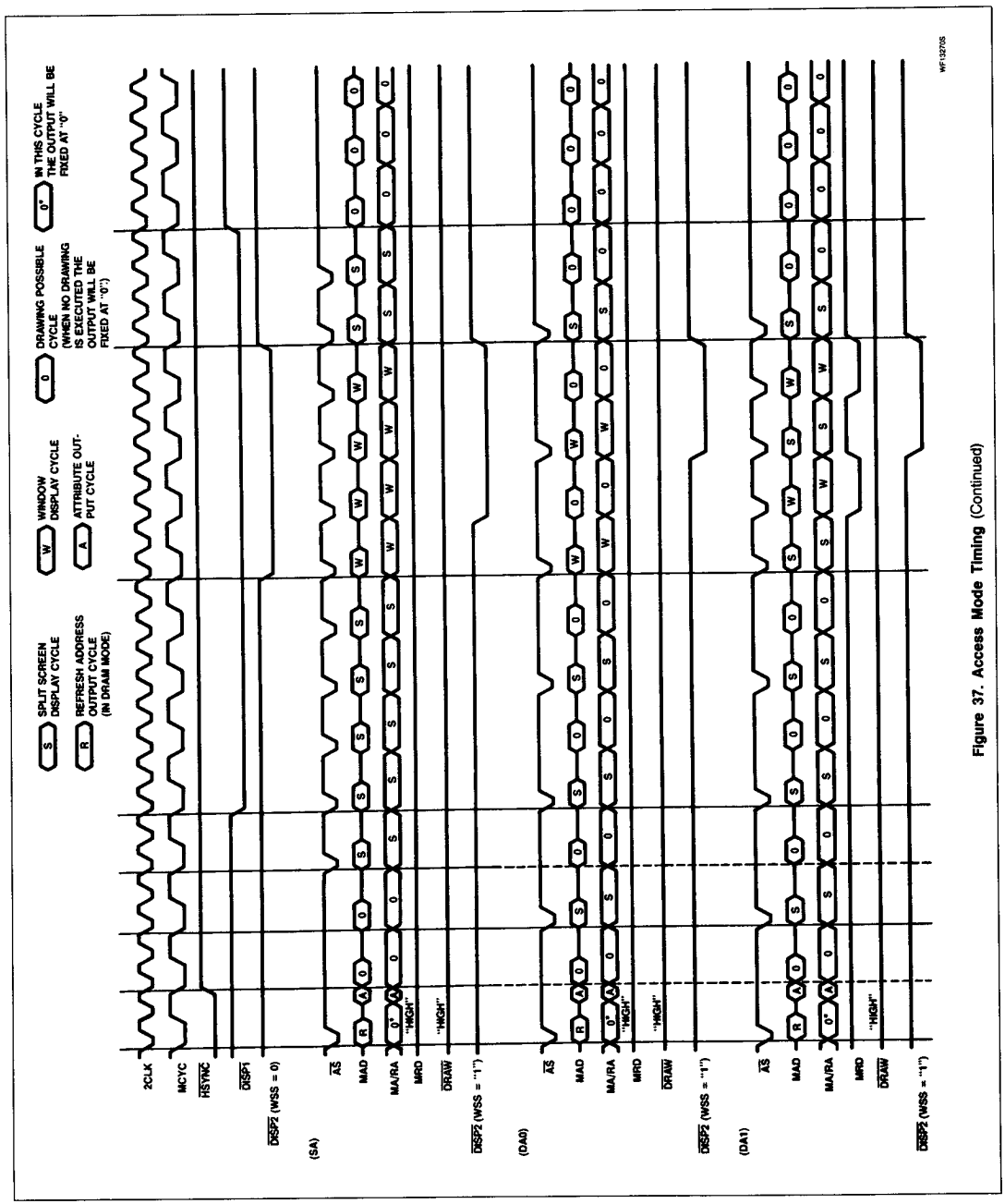


Figure 37. Access Mode Timing (Continued)

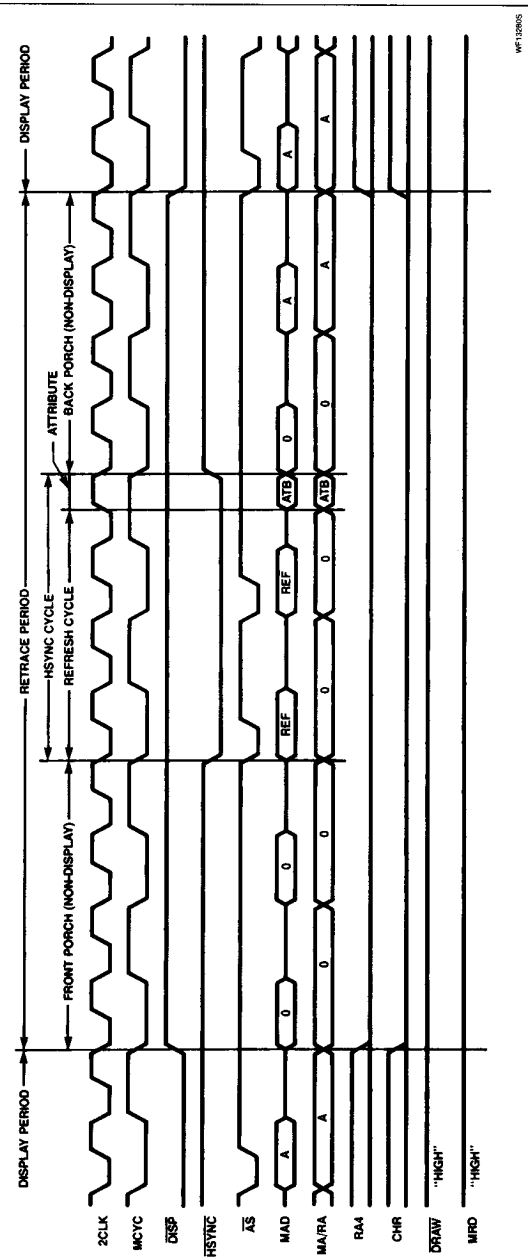


Figure 38. DRAM Refresh Timing

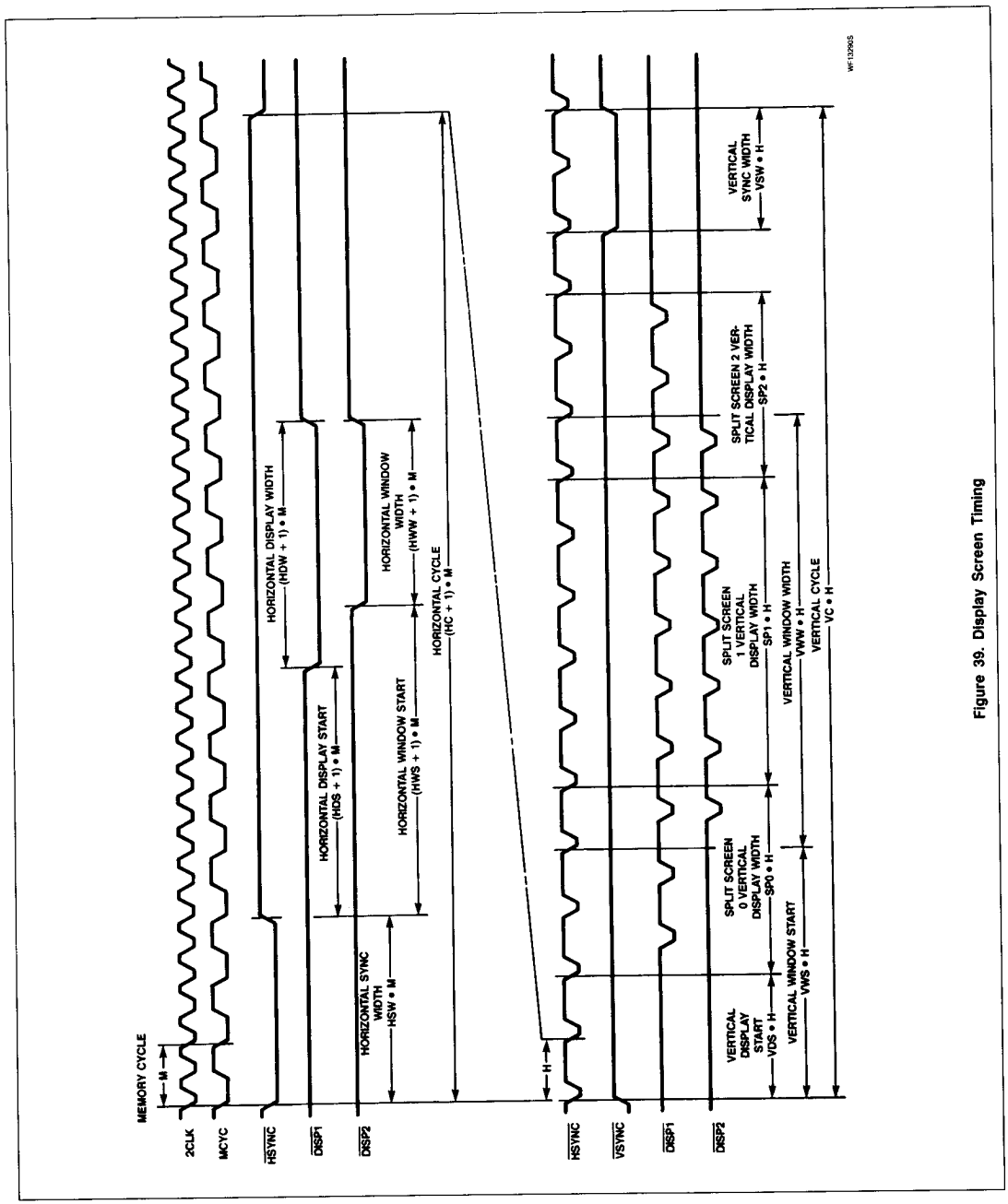


Figure 39. Display Screen Timing

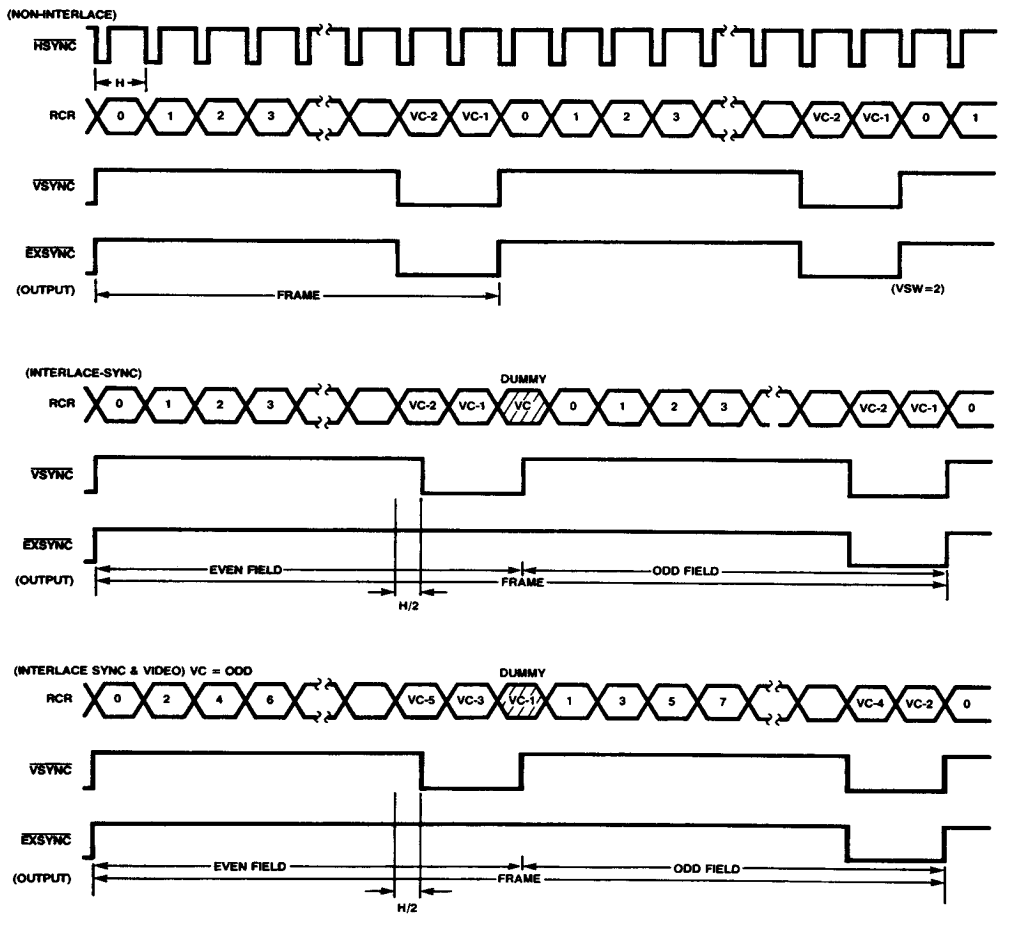


Figure 40. Raster Scan Timing

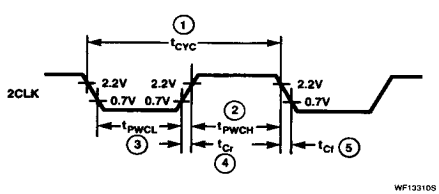
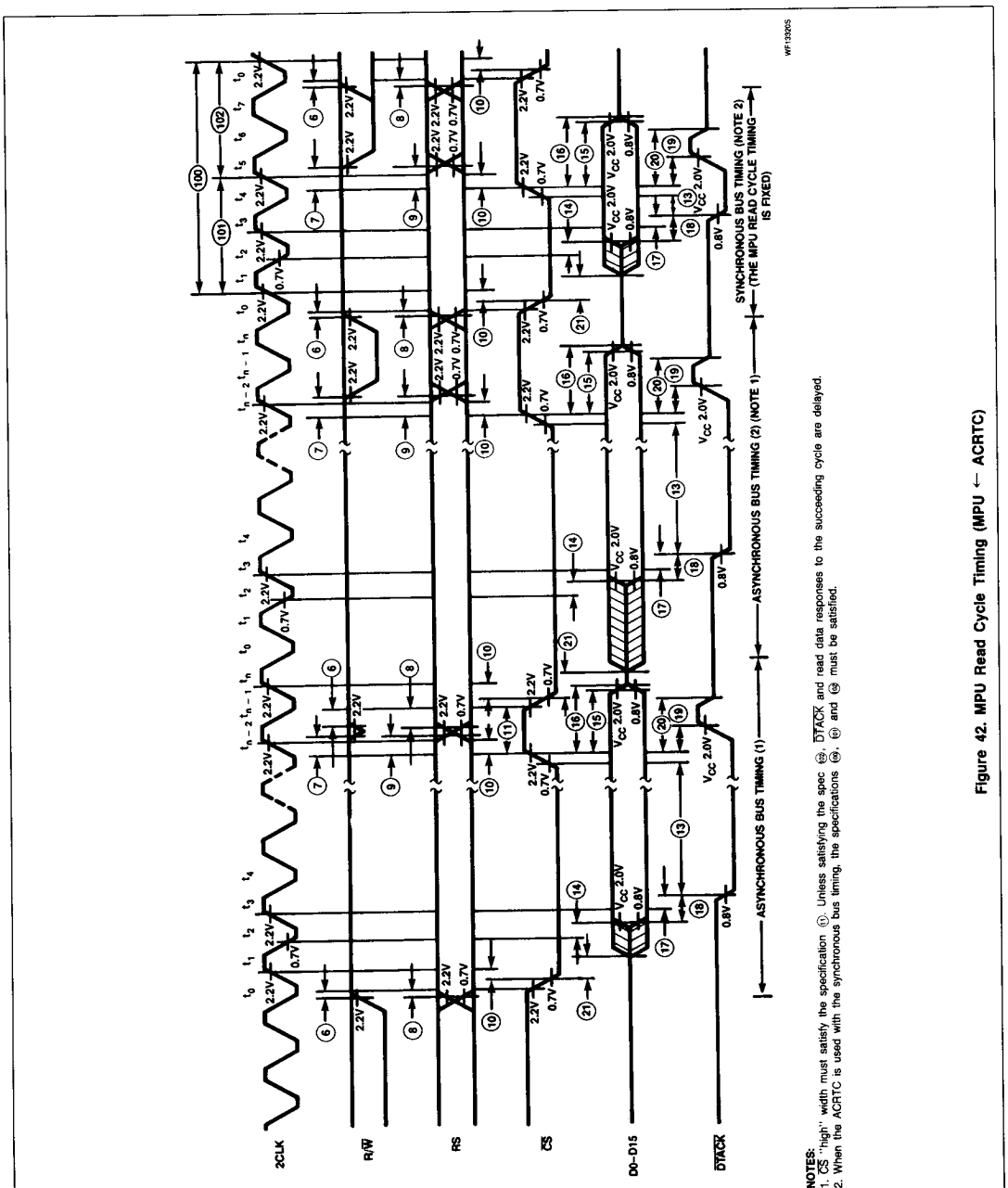
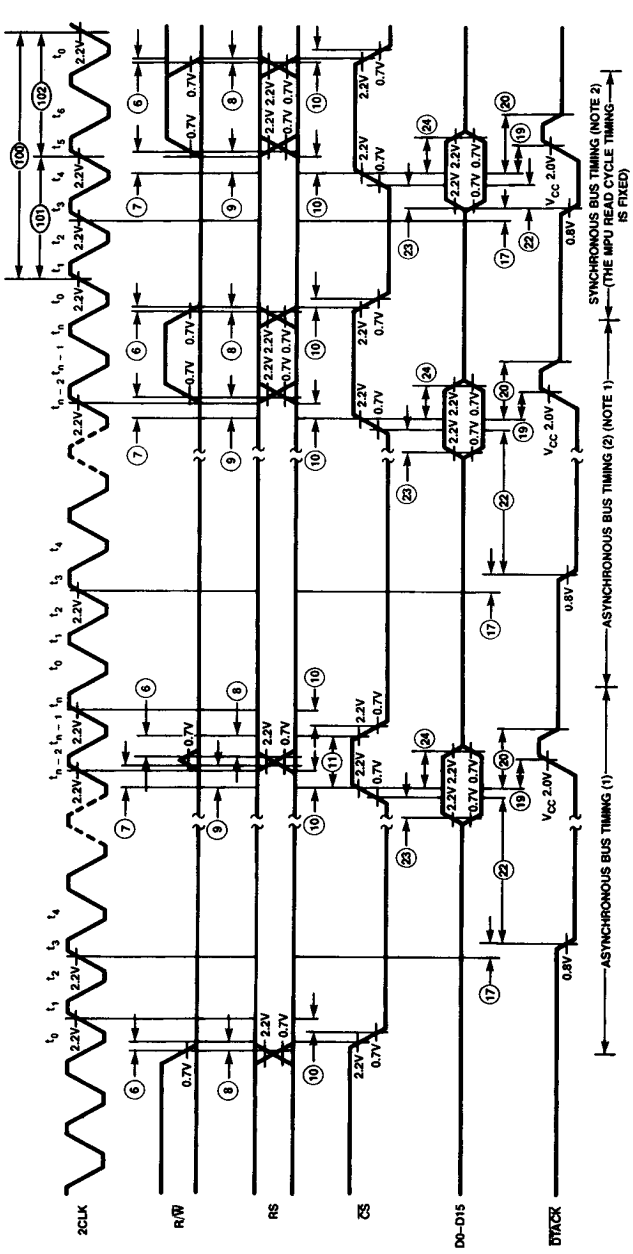


Figure 41. 2CLK Waveform



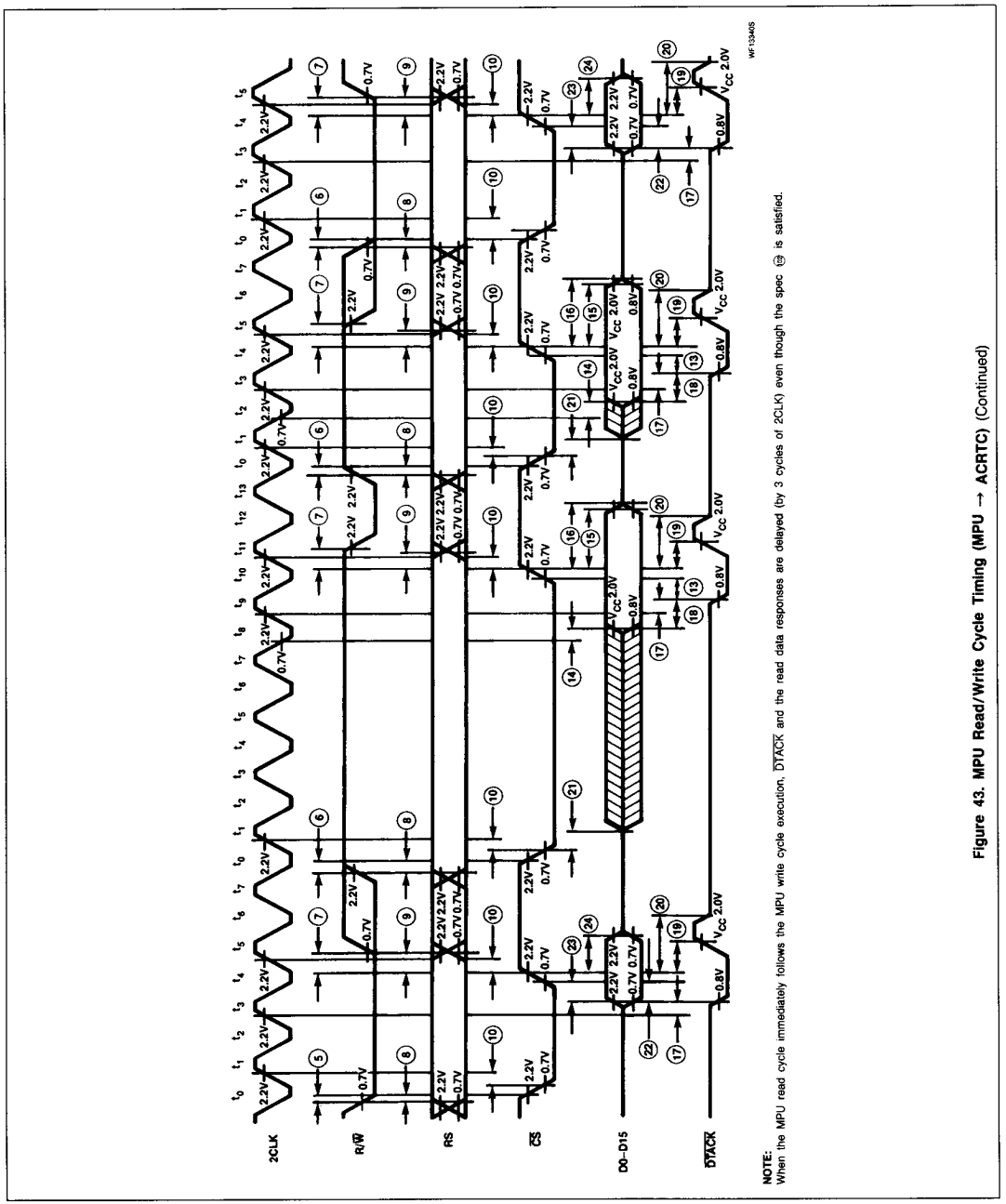
NOTES:
1. CS "high" width must satisfy the specification ⑪. Unless satisfying the spec ⑭, DTACK and read data responses to the succeeding cycle are delayed.
2. When the ACRTC is used with the synchronous bus timing, the specifications ⑭, ⑮ and ⑯ must be satisfied.

Figure 42. MPU Read Cycle Timing (MPU ← ACRTC)



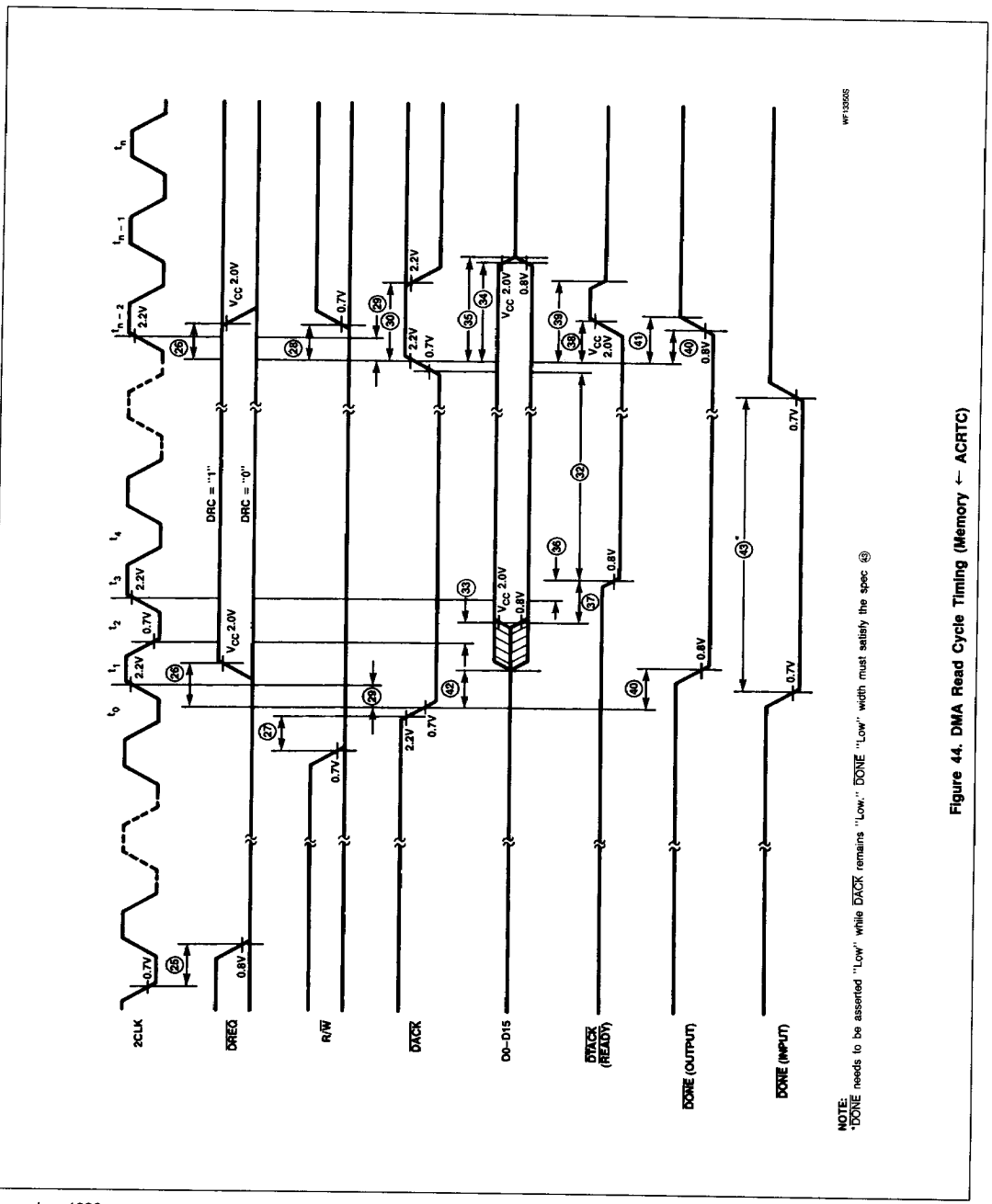
NOTES:
1. $\overline{CS}$ "high" width must satisfy the specification ⑪. Unless satisfying the spec ⑪, $\overline{DTACK}$ response to the succeeding cycle is delayed.
2. When the ACRTC is used with the synchronous bus timing, the specifications ⑨, ⑩ and ⑭ must be satisfied.

Figure 43. MPU Read/Write Cycle Timing (MPU → ACRTC)



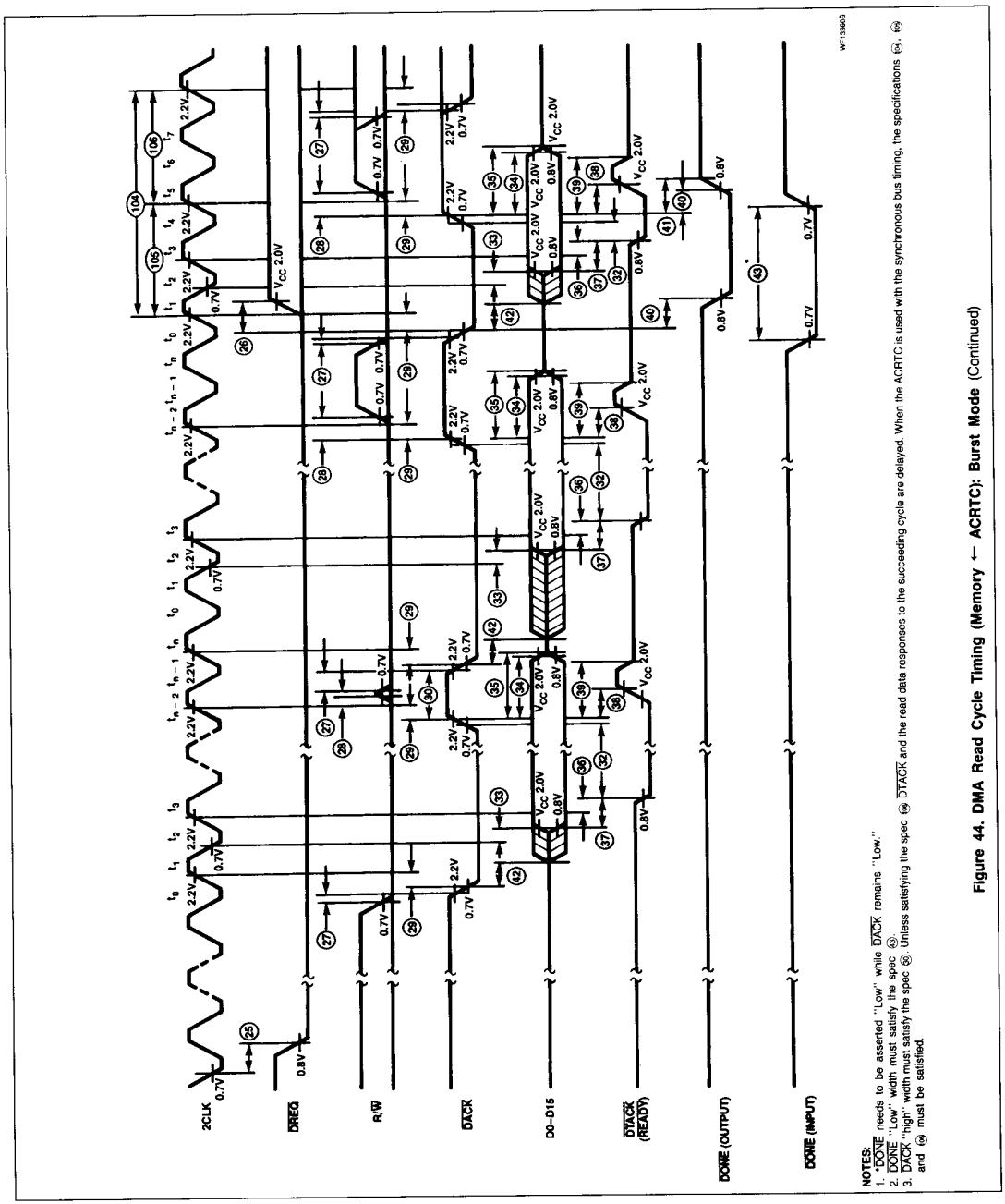
NOTE:
When the MPU read cycle immediately follows the MPU write cycle execution, $\overline{DTACK}$ and the read data responses are delayed (by 3 cycles of 2CLK) even though the spec ⑰ is satisfied.

Figure 43. MPU Read/Write Cycle Timing (MPU → ACRTC) (Continued)



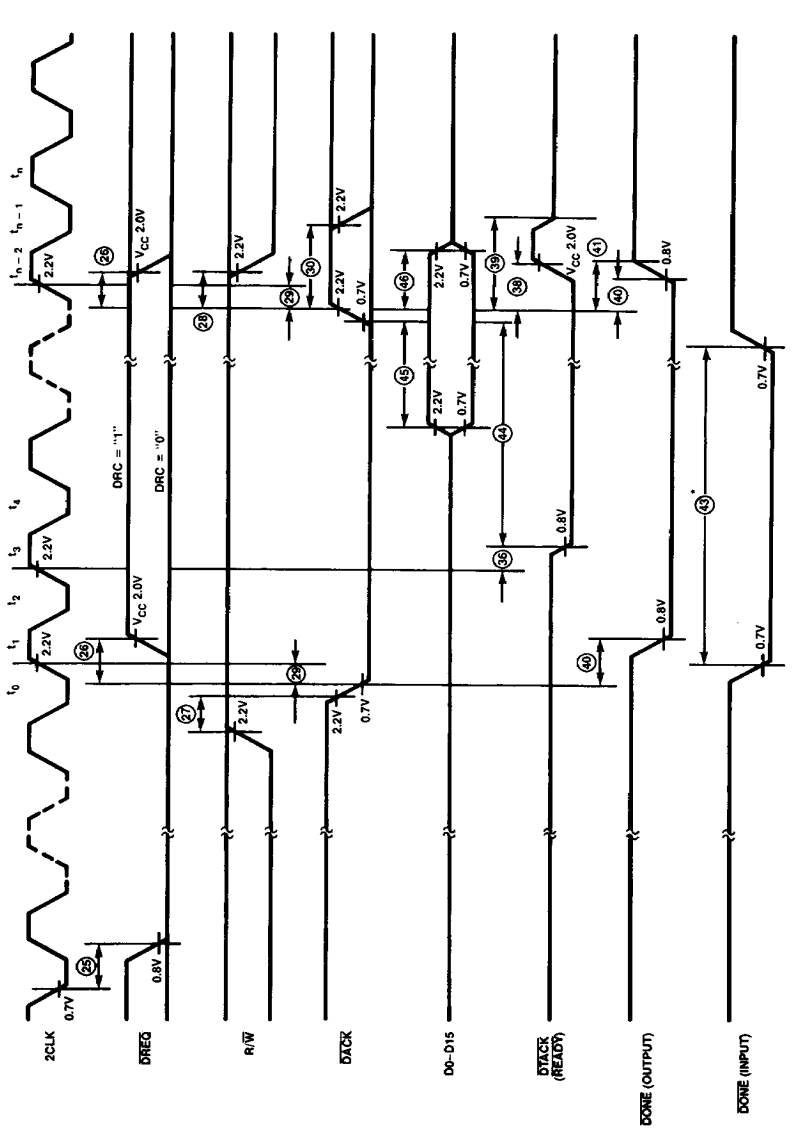
NOTE:
*DONE needs to be asserted "Low" while DACK remains "Low." DONE "Low" width must satisfy the spec ㊸.

Figure 44. DMA Read Cycle Timing (Memory ← ACRTC)



NOTES:
1. "DONE needs to be asserted "Low" while DACK remains "Low."
2. DONE "Low" width must satisfy the spec ㊸.
3. DACK "high" width must satisfy the spec ㊽. Unless satisfying the spec ㊾ DTACK and the read data responses to the succeeding cycle are delayed. When the ACRTC is used with the synchronous bus timing, the specifications ㊾, ㊾ and ㊾ must be satisfied.

Figure 44. DMA Read Cycle Timing (Memory ← ACRTC): Burst Mode (Continued)



NOTE:
*DONE needs to be asserted "Low" while DACK remains "Low". DONE "Low" width must satisfy the spec ㊸.

Figure 45. DMA Write Cycle Timing (Memory → ACRTC)

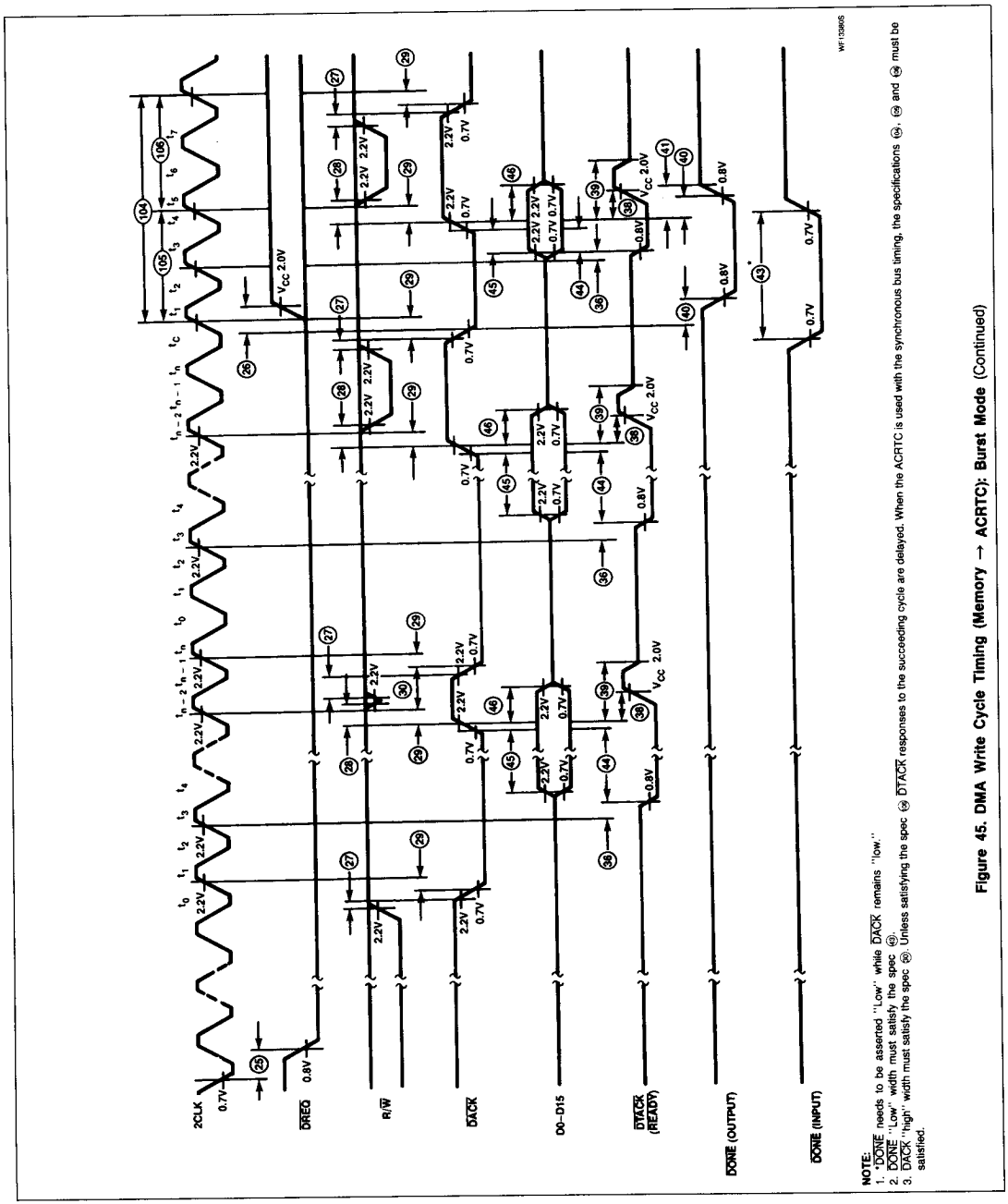


Figure 45. DMA Write Cycle Timing (Memory → ACRTC): Burst Mode (Continued)

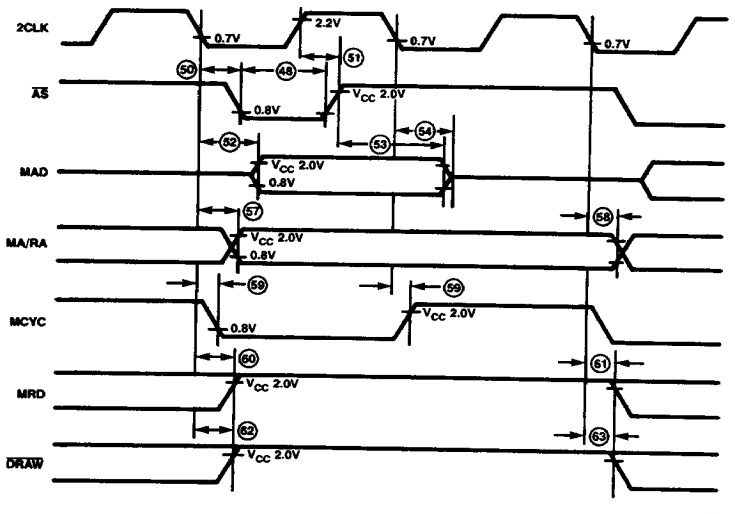


Figure 46. Screen Display Cycle Timing

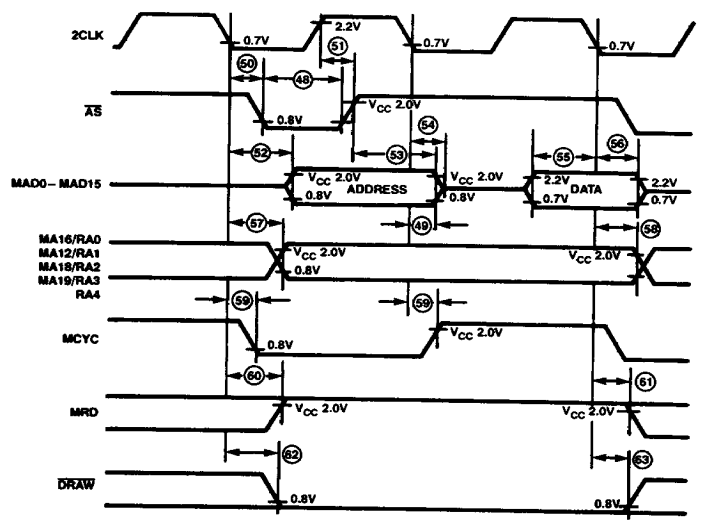


Figure 47. Frame Memory Read Cycle Timing (ACRTC ← Frame Memory)

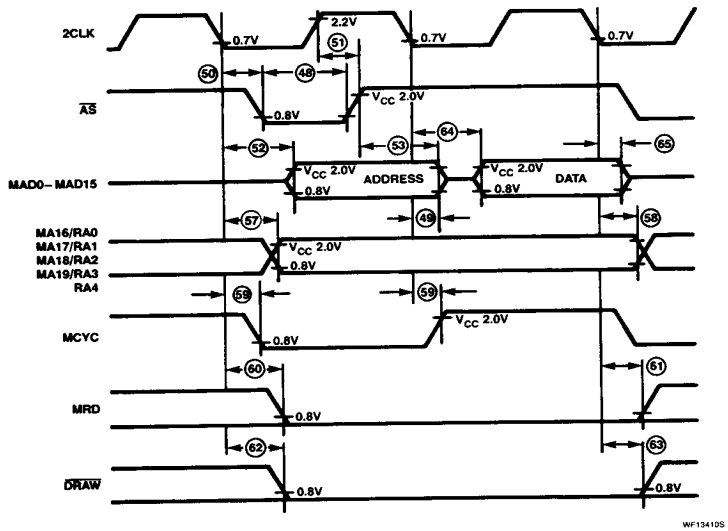
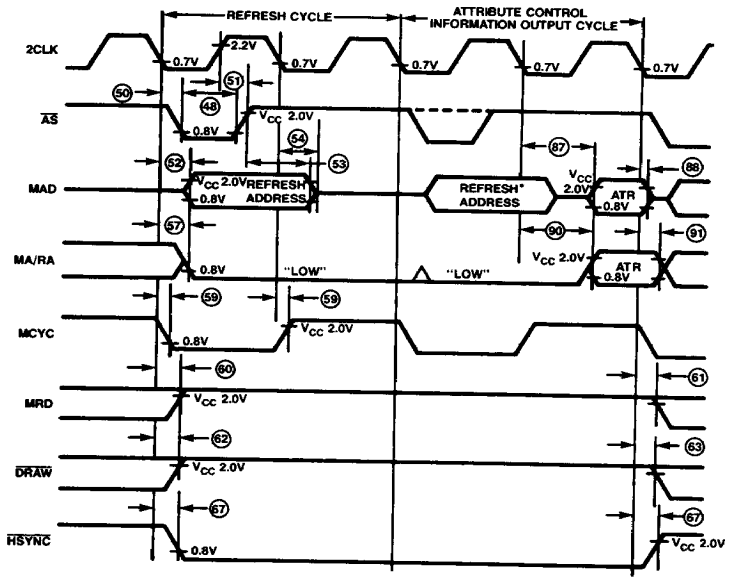


Figure 48. Frame Memory Write Cycle Timing (ACRTC → Frame Memory)



NOTE:
*When $\overline{AS}$ is "High", a "0" output is given.

Figure 49. Frame Memory Refresh/Attribute Control Information Output Cycle Timing

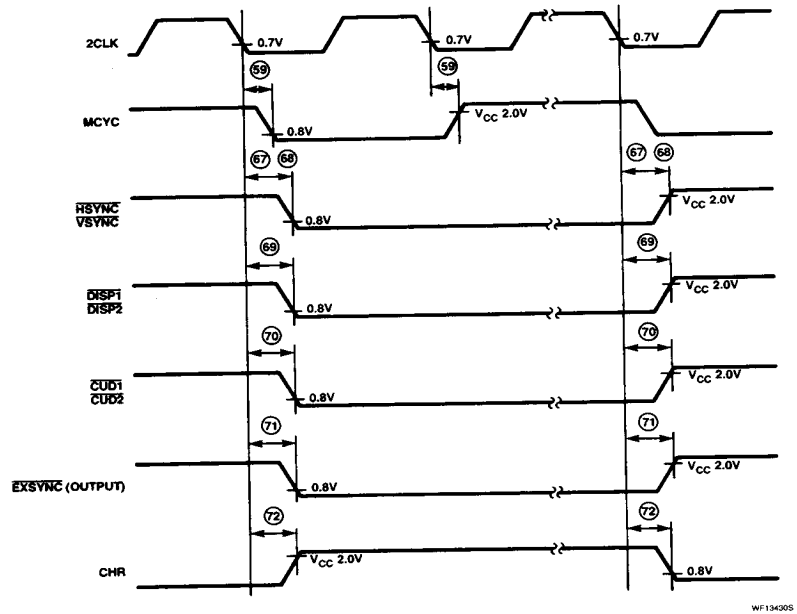
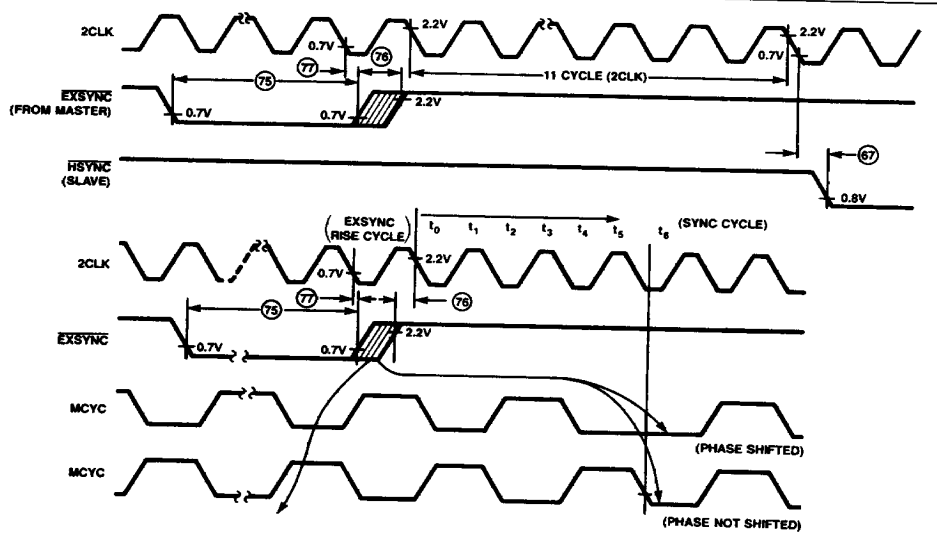
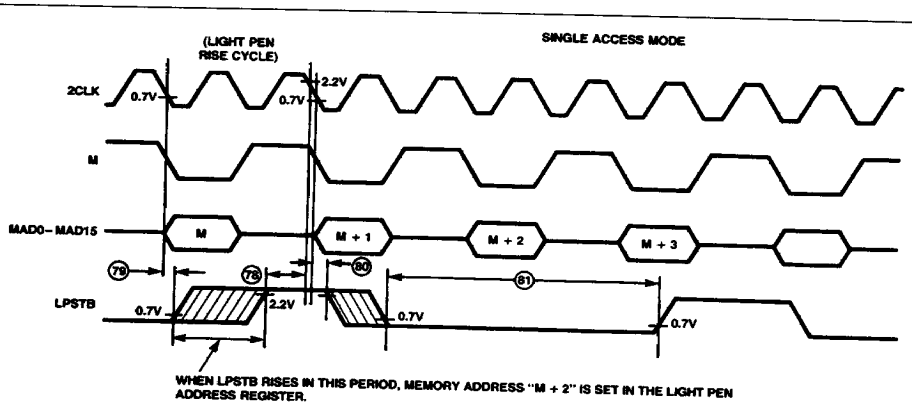


Figure 50. Display Control Signal Output Timing



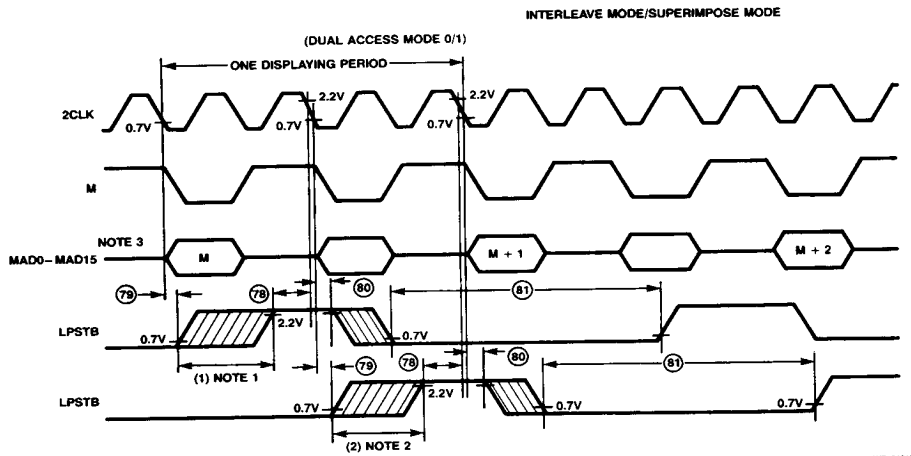
NOTE:
(When the leading edge of EXSYNC enters this period, ACRTC shifts the internal phase according to the above sequence.)

Figure 51. EXSYNC Input Timing



NOTE:
1. Memory address set in light pen address register during H-SYNC cycle may not be proper.

Figure 52. LPSTB Input Timing



NOTES:
1. When LPSTB rises in the period (1), memory address "M + 1" is set in the Light Pen Address register.
2. When LPSTB rises in the period (2), memory address "M + 2" is set in the "Light Pen Address register.
3. In the Interleave Mode, memory address "M", "M + 1", "M + 2" denote the display address. In the Superimpose Mode, memory address "M", "M + 1", "M + 2" denote the display address of the background screen.
4. Memory address set in light pen address register during H-SYNC cycle may not be proper.

Figure 52. LPSTB Input Timing (Continued)

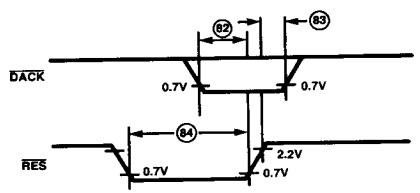


Figure 53. $\overline{RES}$ Input and $\overline{DTACK}$ Input Timing (System Reset and 16-Bit/8-Bit Selection)

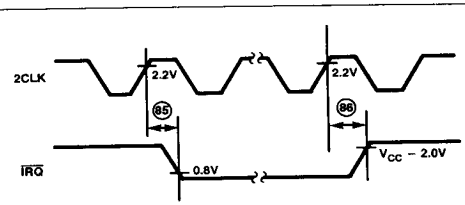


Figure 54. $\overline{IRQ}$ Output Timing