

Name	Reg. addr.	Chip	R/W	p/d	Function
BLTDDAT	000	A	er	d	Blitter output data (from blitter to RAM)
DMACONR	002	AP	r	p	Read DMA controller register
VPOS	004	A	r	p	MSB of the vertical position
VHPOS	006	A	r	p	Vertical and horizontal beam position
DSKDATR	008	P	er	d	Disk read data (from disk to RAM)
JOY0DAT	00A	D	r	p	Joystick/mouse position game port 0
JOY1DAT	00C	D	r	p	Joystick/mouse position game port 1
CLXDAT	00E	D	r	p	Collision register
ADKCONR	010	P	r	p	Read audio/disk control register
POT0DAT	012	P	r	p	Read potentiometer on game port 0
POT1DAT	014	P	r	p	Read potentiometer on game port 1
POTGOR	016	P	r	p	Read pot. port data
SERDATR	018	P	r	p	Read serial port and status
DSKBYTR	01A	P	r	p	Read disk data byte and status
INTENAR	01C	P	r	p	Read interrupt enable
INTREQR	01E	P	r	p	Read interrupt request
DSKPTH	020	A	w	p	Disk DMA address bits 16-18
DSKPTL	022	A	w	p	Disk DMA address bits 1-15
DSKLEN	024	P	w	p	Disk DMA block length
DSKDAT	026	P	w	d	Disk write data (from RAM to disk)
REFPTR	028	A	w	d	Refresh counter
VPOSW	02A	A	w	p	Write MSB of the vertical beam position
VHPOSW	02C	A	w	p	Write vertical and horizontal beam position
COPCON	02E	A	w	p	Copper control register
SERDAT	030	P	w	p	Write serial data and stop bits
SERPER	032	P	w	p	Serial port control register and baud rate
POTGO	034	P	w	p	Write pot. port data and start bit
JOYTEST	036	D	w	p	Write in both mouse counters
STREQU	038	D	s	d	Horizontal sync with VB and equal frame
STRVBL	03A	D	s	d	Horizontal sync with vertical blank
STRHOR	03C	DP	s	d	Horizontal sync signal
STRLONG	03E	D	s	d	Long horizontal line marker

The following registers can be accessed by Copper when COPCON=1.

Name	Reg. addr.	Chip	R/W	p/d	Function
BLTCNO	040	A	w	p	Blitter control register 0
BLTCNI	042	A	w	p	Blitter control register 1
BLTAFWM	044	A	w	p	Mask for the first data word from A
BLTALWM	046	A	w	p	Mask for the last data word from A
BLTCPFH	048	A	w	p	Address of the source data C bits 16-18
BLTCPFL	04A	A	w	p	Address of the source data C bits 1-15
BLTBPTH	04C	A	w	p	Address of the source data B bits 16-18
BLTBPTL	04E	A	w	p	Address of the source data B bits 1-15
BLTAPTH	050	A	w	p	Address of the source data A bits 16-18
BLTAPTL	052	A	w	p	Address of the source data A bits 1-15
BLTDPTH	054	A	w	p	Address of the destination data D bits 16-18
BLTDPTL	056	A	w	p	Address of the destination data D bits 1-15
BLTSIZE	058	A	w	p	Start bit and size of the blitter window
—	05A				unused
—	05C				unused
—	05E				unused
BLTCMOD	060	A	w	p	Blitter module for source data C
BLTBMOD	062	A	w	p	Blitter module for source data B
BLTAMOD	064	A	w	p	Blitter module for source data A
BLTDMOD	066	A	w	p	Blitter module for destination data D
—	068				unused
—	06A				unused
—	06C				unused
—	06E				unused
BLTCDAT	070	A	w	d	Blitter source data register C
BLTBDAT	072	A	w	d	Blitter source data register B
BLTADAT	074	A	w	d	Blitter source data register A
—	076				unused
—	078				unused
—	07A				unused
—	07C				unused
DSKSYNC	07E	P	w	p	Disk sync pattern

The following registers can always be written by the Copper:

Name	Reg. addr.	Chip	R/W	p/d	Function
COP1LCH	080	A	w	p	Address of the 1st Copper list bits 16-18
COP1LCL	082	A	w	p	Address of the 1st Copper list bits 1-15
COP2LCH	084	A	w	p	Address of 2nd Copper list bits 16-18
COP2LCL	086	A	w	p	Address of the 2nd Copper list bits 1-15
COPJMP1	088	A	s	p	Jump to the start of the 1st Copper list
COPJMP2	08A	A	s	p	Jump to the start of the 2nd Copper list
COPINS	08C	A	w	d	Copper command register
DIWSTRT	08E	A	w	p	Upper left corner of the display window
DIWSTOP	090	A	w	p	Lower right corner of the display window
DDFSTRT	092	A	w	p	Start of the bit plane DMA (horiz. pos.)
DDFSTOP	094	A	w	p	End of the bit plane DMA (horiz. pos.)
DMACON	096	ADP	w	p	Write DMA control register
CLXCON	098	D	w	p	Write collision control register
INTENA	09A	P	w	p	Write interrupt enable

Name	Reg. addr.	Chip	R/W	p/d	Function
INTREQ	09C	P	w	p	Write interrupt request
ADKCON	09E	P	w	p	Audio, disk, and UART control register
AUD0LCH	0A0	A	w	p	Address of the audio data bits 16-18
AUD0LCL	0A2	A	w	p	On sound channel 0, bits 1-15
AUD0LEN	0A4	P	w	p	Channel 0 length of audio data
AUD0PER	0A6	P	w	p	Channel 0 period duration
AUD0VOL	0A8	P	w	p	Channel 0 volume
AUD0DAT	0AA	P	w	d	Channel 0 audio data (to the D/A converter)
—	0AC				unused
—	0AE				unused
AUD1LCH	0B0	A	w	p	Address of the audio data bits 16-18
AUD1LCL	0B2	A	w	p	On sound channel 1, bits 1-15
AUD1LEN	0B4	P	w	p	Channel 1 length of audio data
AUD1PER	0B6	P	w	p	Channel 1 period duration
AUD1VOL	0B8	P	w	p	Channel 1 volume
AUD1DAT	0BA	P	w	d	Channel 1 audio data (to the D/A converter)
—	0BC				unused
—	0BE				unused
AUD2LCH	0C0	A	w	p	Address of the audio data bits 16-18
AUD2LCL	0C2	A	w	p	On sound channel 2, bits 1-15
AUD2LEN	0C4	P	w	p	Channel 2 length of audio data
AUD2PER	0C6	P	w	p	Channel 2 period duration
AUD2VOL	0C8	P	w	p	Channel 2 volume
AUD2DAT	0CA	P	w	d	Channel 2 audio data (to the D/A converter)
—	0CC				unused
—	0CE				unused
AUD3LCH	0D0	A	w	p	Address of the audio data bits 16-18
AUD3LCL	0D2	A	w	p	On sound channel 3, bits 1-15
AUD3LEN	0D4	P	w	p	Channel 3 length of audio data
AUD3PER	0D6	P	w	p	Channel 3 period duration
AUD3VOL	0D8	P	w	p	Channel 3 volume
AUD3DAT	0DA	P	w	d	Channel 3 audio data (to the D/A converter)
—	0DC				unused
—	0DE				unused
BPL1PTH	0E0	A	w	p	Address of bit plane 1, bits 16-18
BPL1PTL	0E2	A	w	p	Address of bit plane 1, bits 1-15
BPL2PTH	0E4	A	w	p	Address of bit plane 2, bits 16-18
BPL2PTL	0E6	A	w	p	Address of bit plane 2, bits 1-15
BPL3PTH	0E8	A	w	p	Address of bit plane 3, bits 16-18
BPL3PTL	0EA	A	w	p	Address of bit plane 3, bits 1-15
BPL4PTH	0EC	A	w	p	Address of bit plane 4, bits 16-18
BPL4PTL	0EE	A	w	p	Address of bit plane 4, bits 1-15
BPL5PTH	0F0	A	w	p	Address of bit plane 5, bits 16-18
BPL5PTL	0F2	A	w	p	Address of bit plane 5, bits 1-15
BPL6PTH	0F4	A	w	p	Address of bit plane 6, bits 16-18
BPL6PTL	0F6	A	w	p	Address of bit plane 6, bits 1-15
—	0F8				unused
—	0FA				unused
—	0FC				unused
—	0FE				unused
BPLCON0	100	AD	w	p	Bit plane control register 0
BPLCON1	102	D	w	p	Control register 1 (scroll values)
BPLCON2	104	D	w	p	Control register 2 (priority control)
—	106				unused
BPL1MOD	108	A	w	p	Bit plane module for odd planes
BPL2MOD	10A	A	w	p	Bit plane module for even planes
—	10C				unused
—	10E				unused
BPL1DAT	110	D	w	d	Bit plane 1 data (to RGB output)
BPL2DAT	112	D	w	d	Bit plane 2 data (to RGB output)
BPL3DAT	114	D	w	d	Bit plane 3 data (to RGB output)
BPL4DAT	116	D	w	d	Bit plane 4 data (to RGB output)
BPL5DAT	118	D	w	d	Bit plane 5 data (to RGB output)
BPL6DAT	11A	D	w	d	Bit plane 6 data (to RGB output)
—	11C				unused
—	11E				unused
SPR0PTH	120	A	w	p	Sprite data 0, bits 16-18
SPR0PTL	122	A	w	p	Sprite data 0, bits 1-15
SPR1PTH	124	A	w	p	Sprite data 1, bits 16-18
SPR1PTL	126	A	w	p	Sprite data 1, bits 1-15
SPR2PTH	128	A	w	p	Sprite data 2, bits 16-18
SPR2PTL	12A	A	w	p	Sprite data 2, bits 1-15
SPR3PTH	12C	A	w	p	Sprite data 3, bits 16-18
SPR3PTL	12E	A	w	p	Sprite data 3, bits 1-15
SPR4PTH	130	A	w	p	Sprite data 4, bits 16-18
SPR4PTL	132	A	w	p	Sprite data 4, bits 1-15
SPR5PTH	134	A	w	p	Sprite data 5, bits 16-18
SPR5PTL	136	A	w	p	Sprite data 5, bits 1-15
SPR6PTH	138	A	w	p	Sprite data 6, bits 16-18
SPR6PTL	13A	A	w	p	Sprite data 6, bits 1-15
SPR7PTH	13C	A	w	p	Sprite data 7, bits 16-18
SPR7PTL	13E	A	w	p	Sprite data 7, bits 1-15
SPR0POS	140	AD	w	dp	Sprite 0 start position (vert. and horiz.)
SPR0CTL	142	AD	w	dp	Sprite 0 control reg. and vertical stop
SPR0DATA	144	D	w	dp	Sprite 0 data register A (to RGB output)
SPR0DATB	146	D	w	dp	Sprite 0 data register B (to RGB output)
SPR1POS	148	AD	w	dp	Sprite 1 start position (vert. and horiz.)
SPR1CTL	14A	AD	w	dp	Sprite 1 control reg. and vertical stop
SPR1DATA	14C	D	w	dp	Sprite 1 data register A (to RGB output)
SPR1DATB	14E	D	w	dp	Sprite 1 data register B (to RGB output)
SPR2POS	150	AD	w	dp	Sprite 2 start position (vert. and horiz.)
SPR2CTL	152	AD	w	dp	Sprite 2 control reg. and vertical stop
SPR2DATA	154	D	w	dp	Sprite 2 data register A (to RGB output)
SPR2DATB	156	D	w	dp	Sprite 2 data register B (to RGB output)
SPR3POS	158	AD	w	dp	Sprite 3 start position (vert. and horiz.)
SPR3CTL	15A	AD	w	dp	Sprite 3 control reg. and vertical stop
SPR3DATA	15C	D	w	dp	Sprite 3 data register A (to RGB output)

Construction of the second control word

Bit no.: 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Function: 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

SPR3DATAB	15E	D	w	dp	Sprite 3 data register B (to RGB output)
SPR4POS	160	AD	w	dp	Sprite 4 start position (vert. and horiz.)
SPR4CTL	162	AD	w	dp	Sprite 4 control reg. and vertical stop
SPR4DATA	164	D	w	dp	Sprite 4 data register A (to RGB output)
SPR4DATAB	166	D	w	dp	Sprite 4 data register B (to RGB output)
SPR5POS	168	AD	w	dp	Sprite 5 start position (vert. and horiz.)
SPR5CTL	16A	AD	w	dp	Sprite 5 control reg. and vertical stop
SPR5DATA	16C	D	w	dp	Sprite 5 data register A (to RGB output)
SPR5DATAB	16E	D	w	dp	Sprite 5 data register B (to RGB output)
SPR6POS	170	AD	w	dp	Sprite 6 start position (vert. and horiz.)
SPR6CTL	172	AD	w	dp	Sprite 6 control reg. and vertical stop
SPR6DATA	174	D	w	dp	Sprite 6 data register A (to RGB output)
SPR6DATAB	176	D	w	dp	Sprite 6 data register B (to RGB output)
SPR7POS	178	AD	w	dp	Sprite 7 start position (vert. and horiz.)
SPR7CTL	17A	AD	w	dp	Sprite 7 control reg. and vertical stop
SPR7DATA	17C	D	w	dp	Sprite 7 data register A (to RGB output)
SPR7DATAB	17E	D	w	dp	Sprite 7 data register B (to RGB output)
COLOR00	180	D	w	p	Color palette register 0 (color table)
COLOR01	182	D	w	p	Color palette register 1 (color table)
COLOR02	184	D	w	p	Color palette register 2 (color table)
COLOR03	186	D	w	p	Color palette register 3 (color table)
COLOR04	188	D	w	p	Color palette register 4 (color table)
COLOR05	18A	D	w	p	Color palette register 5 (color table)
COLOR06	18C	D	w	p	Color palette register 6 (color table)
COLOR07	18E	D	w	p	Color palette register 7 (color table)
COLOR08	190	D	w	p	Color palette register 8 (color table)
COLOR09	192	D	w	p	Color palette register 9 (color table)
COLOR10	194	D	w	p	Color palette register 10 (color table)
COLOR11	196	D	w	p	Color palette register 11 (color table)
COLOR12	198	D	w	p	Color palette register 12 (color table)
COLOR13	19A	D	w	p	Color palette register 13 (color table)
COLOR14	19C	D	w	p	Color palette register 14 (color table)
COLOR15	19E	D	w	p	Color palette register 15 (color table)
COLOR16	1A0	D	w	p	Color palette register 16 (color table)
COLOR17	1A2	D	w	p	Color palette register 17 (color table)
COLOR18	1A4	D	w	p	Color palette register 18 (color table)
COLOR19	1A6	D	w	p	Color palette register 19 (color table)
COLOR20	1A8	D	w	p	Color palette register 20 (color table)
COLOR21	1AA	D	w	p	Color palette register 21 (color table)
COLOR22	1AC	D	w	p	Color palette register 22 (color table)
COLOR23	1AE	D	w	p	Color palette register 23 (color table)
COLOR24	1B0	D	w	p	Color palette register 24 (color table)
COLOR25	1B2	D	w	p	Color palette register 25 (color table)
COLOR26	1B4	D	w	p	Color palette register 26 (color table)
COLOR27	1B6	D	w	p	Color palette register 27 (color table)
COLOR28	1B8	D	w	p	Color palette register 28 (color table)
COLOR29	1BA	D	w	p	Color palette register 29 (color table)
COLOR30	1BC	D	w	p	Color palette register 30 (color table)
COLOR31	1BE	D	w	p	Color palette register 31 (color table)

The registers from 1C0 to 1FC are unoccupied.

Accessing register address 1FE has no effect. The chips are not accessed (see Section 1.2.3).

The various registers of the CIAs appear multiple times in the range from \$A00000 to \$BFFFFF. More about the addressing of the CIAs can be obtained from Section 1.2. Here are the addresses of the individual registers at their normal positions:

CIA-A	CIA-B	Name	Function
\$BFE001	\$BFD000	PA	Port register A
\$BFE101	\$BFD100	PB	Port register B
\$BFE201	\$BFD200	DDRA	Data direction register A
\$BFE301	\$BFD300	DDRB	Data direction register B
\$BFE401	\$BFD400	TALO	Timer A low byte
\$BFE501	\$BFD500	TAHI	Timer A high byte
\$BFE601	\$BFD600	TBLO	Timer B low byte
\$BFE701	\$BFD700	TBHI	Timer B high byte
\$BFE801	\$BFD800	E.LSB	Event counter bits 0-7
\$BFE901	\$BFD900	E.MID	Event counter bits 8-15
\$BFEA01	\$BFDA00	E.MSB	Event counter 16-23
\$BFEB01	\$BFDB00	—	Unused
\$BFEC01	\$BFDC00	SP	Serial port register
\$BFED01	\$BFDD00	IRC	Interrupt control register
\$BFEE01	\$BFDE00	CRA	Control register A
\$BFEE01	\$BFDE00	CRB	Control register B

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
00			SP	0	a	P		p			NSP		A	0	a	0
01			!	1	A	q		a	q		i	±	Á	N	á	ñ
02			"	2	B	R		b	r		€	²	Â	Ò	â	ò
03			#	3	C	S		c	s		£	³	Ã	Ó	ã	ó
04			\$	4	D	T		d	t		¤	⁴	Ä	Ô	ä	ô
05			%	5	E	U		e	u		¥	µ	Å	Ö	å	ö
06			&	6	F	V		f	v		¦	¶	Æ	Ø	æ	ø
07			'	7	G	W		g	w		§	·	Ç	×	ç	×
08			(8	H	X		h	x		¨	¸	È	Ù	è	ù
09)	9	I	Y		i	y		©	¹	É	Ú	é	ú
10			*	:	J	Z		j	z		ª	º	Ê	Û	ê	û
11			+	;	K	[k	[«	»	Ë	Ü	ë	ü
12			,	<	L	\		l			¬	¼	Ì	Ý	ì	ý
13			-	=	M]		m]		½	½	Í	Þ	í	þ
14			.	>	N	^		n	^		¾	¾	Î	ß	î	ß
15			/	?	O	_		o	_		¸	¸	Ï	À	ï	à

Normal configuration

\$000000	512K chip RAM	Copy of memory range \$FC0000 - \$FFFFFF
\$080000	copy of chip RAM	
\$100000	copy of chip RAM	
\$180000	copy of chip RAM	
\$200000	8MB fast RAM area	
\$A00000	CIAs	Starting address of CIA B Starting address of CIA A
\$C00000	512K expansion (Amiga 500/2000)	
\$C80000	Unused	
\$DC0000	Realtime clock (Amiga 500/2000)	Base address - realtime clock
\$DF0000	Custom chips	Base address - custom chips
\$E00000	Unused	
\$E80000	Expansion slot area	
\$F00000	ROM module	
\$F80000	256K copy of KickStart ROM	
\$FC0000	256K KickStart ROM	

DMACON Register addresses \$096 (write) and \$02 (read)

Bit	Name	Function (when set)
15	SET/CLR	Set/clear bits
14	BBUSY	Blitter busy (read only)
13	BZERO	Result of all blitter operations is 0 (read only)
12 and 11		Unused
10	BLTPRI	Blitter DMA has priority over processor
9	DMAEN	Enable all DMA (for bits 0 to 8)
8	BPLEN	Enable bit plane DMA
7	COPEN	Enable Copper DMA
6	BLTEN	Enable blitter DMA
5	SPREN	Enable sprite DMA
4	DSKEN	Enable disk DMA
3-0	AUDxEN	Enable audio DMA for sound channel x (the bit number corresponds to the number of the sound channel).

Register addresses: INTREQ = \$09C (write)
INTREQR = \$01E (read)
INTENA = \$09A (write)
INTENAR = \$01C (read)

Bit	Name	IL	Function
15	SET/CLR		Write/read (see DMACON register)
14	INTEN	(6)	Enable interrupts
13	EXTER	6	Interrupt from CIA-B or expansion port
12	DSKSYN	5	Disk sync value recognized
11	RBF	5	Serial receive buffer full
10	AUD3	4	Output audio data channel 3
9	AUD2	4	Output audio data channel 2
8	AUD1	4	Output audio data channel 1
7	AUD0	4	Output audio data channel 0
6	BLIT	3	Blitter ready
5	VERTB	3	Start of the vertical blanking gap reached
4	COPER	3	Reserved for the Copper interrupt
3	PORTS	2	Interrupt from CIA-A or the expansion port
2	SOFT	1	Reserved for software interrupts
1	DSKBLK	1	Disk DMA transfer done
0	TBE	1	Serial transmit buffer empty

The lower thirteen bits stand for the individual interrupt sources. The CIA interrupts are combined into a single interrupt. The bits in the DMAREQ register indicate which interrupts have occurred. A bit is set if the corresponding interrupt has occurred. In order to generate a processor interrupt, the corresponding bit must be set in the DMAENA register and the INTEN bit must also be set. The INTEN bit thus acts as the main switch for the remaining 14 interrupt sources which can be turned on or off with the individual bits of the INTENA register. Only when INTEN is 1 can any interrupts be generated.

If both the INTEN bit and the two corresponding bits in the INTENA and INTREQ registers are set, a processor interrupt is generated. The corresponding autovector numbers are listed in the IL (Interrupt Level) column in the table. Here are the addresses of the seven interrupt autovectors:

Vector no.	Address (Dec/hex)	Autovector level
25	100 / \$64	Autovector level 1
26	104 / \$68	Autovector level 2
27	108 / \$6C	Autovector level 3
28	112 / \$70	Autovector level 4
29	116 / \$74	Autovector level 5
30	120 / \$78	Autovector level 6
31	124 / \$7C	Autovector level 7

As you can see, the interrupts which require faster processing are given higher interrupt levels.

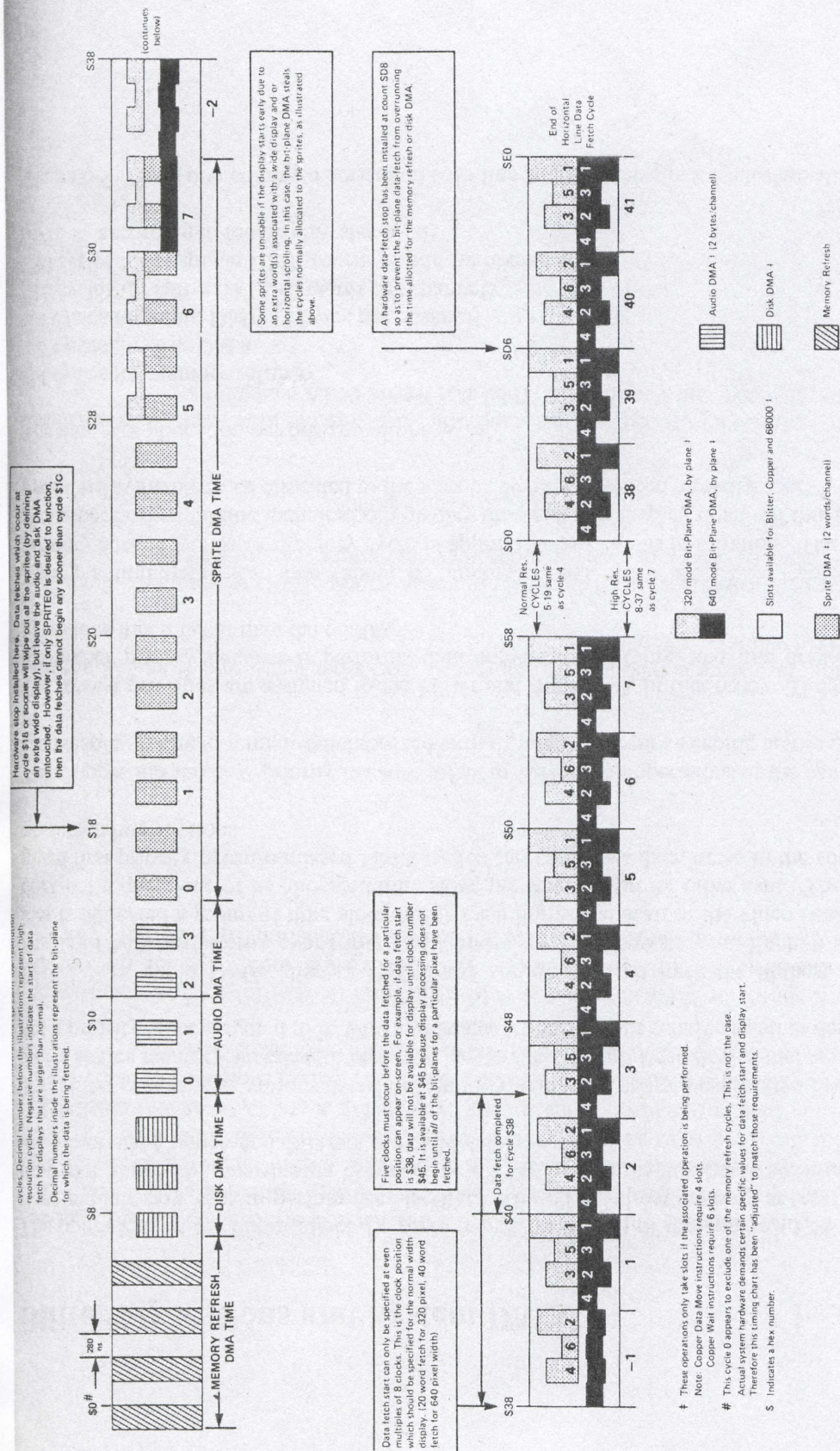


Figure 6-9: DMA Time Slot Allocation

REGISTERS USED BY THE DISK SUBSYSTEM

The disk subsystem uses two ports on the system's 8520 CIA chips, and several registers in the Paula chip:

CIAAPRA	(\$BFE001)	four input bits for disk sensing
CIABPRB	(\$BFD100)	eight output bits for disk selection, control and stepping
ADKCON	(\$DFF09E)	control bits (write only register)
ADKCONR	(\$DFF010)	control bits (read only register)
DSKPTH	(\$DFF020)	DMA pointer (32 bits)
DSKLEN	(\$DFF024)	length of DMA
DSKBYTR	(\$DFF01A)	Disk data byte and status read
DSKSYNC	(\$DFF07E)	Disk sync finder; holds a match word

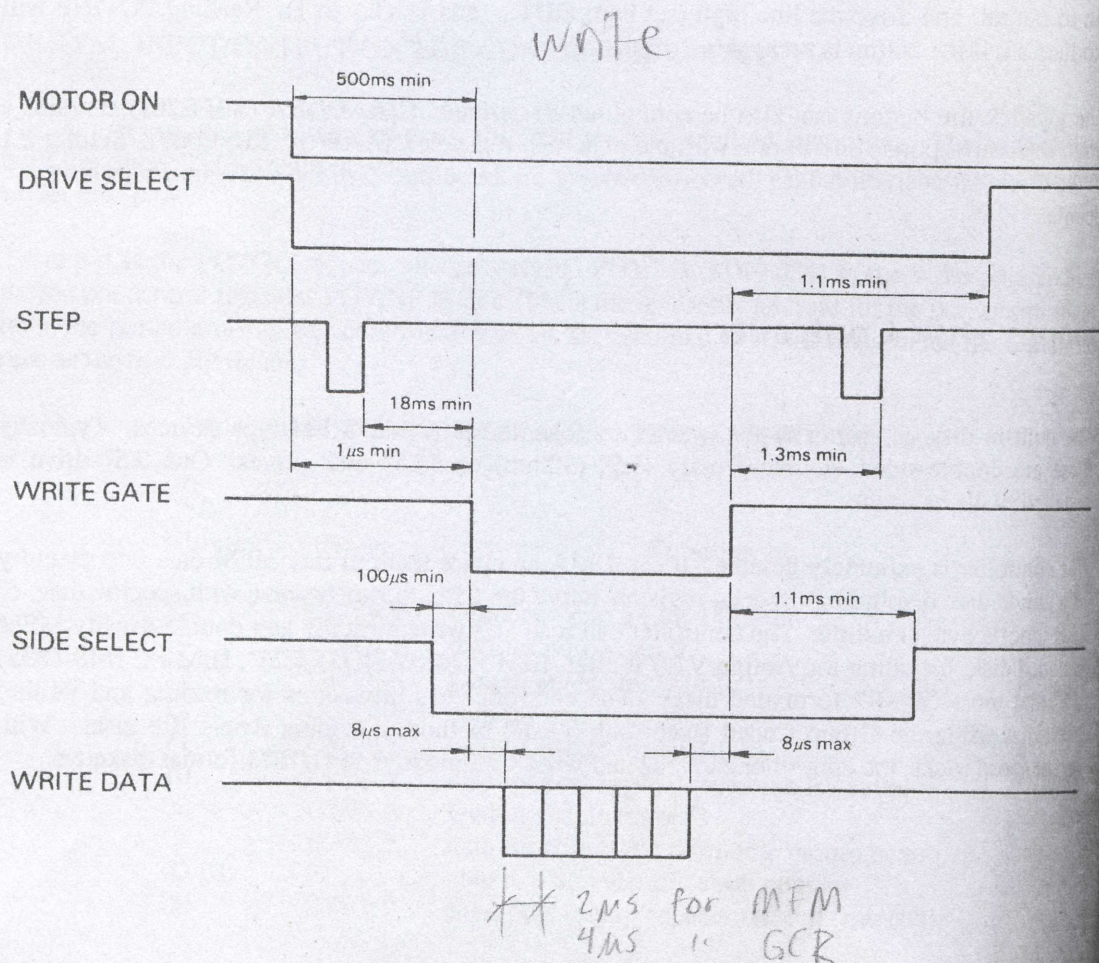
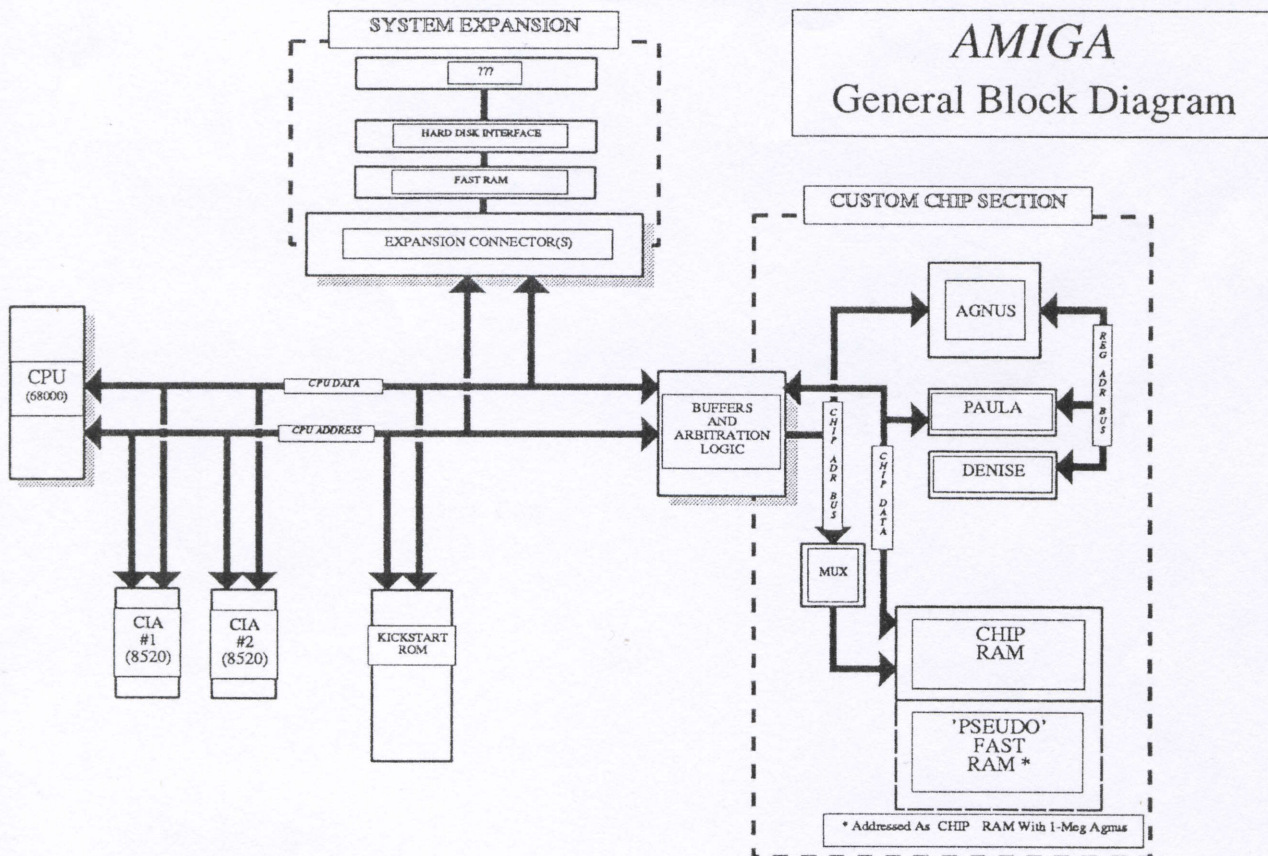


Figure 8-7: Chinon Timing Diagram

A500 MEMORY MAP

Address	Description
\$000000	512K Chipmem
\$080000	Clone of above. (Reserved for Chipmem on Fatter Agnuses.)
\$100000	Clone of above. (Reserved for Chipmem on Superfat/AGA.)
↓	
\$200000	Up to 8M of Fastmem
↓	
↓	
↓	
↓	
↓	
↓	
↓	
↓	
↓	
↓	
\$A00000	Reserved.
↓	
\$BFD000	CIA-A. 16 registers, at offset +\$000, +\$100 etc.
\$BFE001	CIA-B. 16 registers, at offsets +\$001, +\$101 etc.
\$C00000	Slowmem
\$D80000	Reserved (some Slowmem board use this to reach 1.75M)
\$DC0000	Real Time Clock
\$DFF000	Custom chip registers. For A500, up to \$DFF1C0
\$E00000	Reserved.
\$E80000	Autoconfig space for exp. Boards. ROMs map to \$E90000+
\$F00000	Reserved.
\$FC0000	Kickstart (256K ROM)

First digit in address is number of MB. So, max 16M for CPUs with 24-bit address bus.



Bit 1, DSKBLK, indicates "disk block finished." It is used to indicate that the specified DMA task that you have requested has been completed. This bit generates a level 1 interrupt.

More information about disk data transfer and interrupts may be found in Chapter 8, "Inter Hardware."

Serial Port Interrupts

The following serial interrupts are associated with the specified bits of the interrupt registers.

Bit 11, RBF (for receive buffer full), specifies that the input buffer of the UART has data th ready to read. This bit generates a level 5 interrupt.

Bit 0, TBE (for "transmit buffer empty"), specifies that the output buffer of the UART n more data and data can now be written into this buffer. This bit generates a level 1 interrupt.

Hardware priority	Exec Software priority		Label
		Description	
1	1	software interrupt	SOFTINT
	2	disk block complete	DSKBLK
	3	transmitter buffer empty	TBE
2	4	external INT2 & CIAA	PORTS
3	5	graphics coprocessor	COPER
	6	vertical blank interval	VERTB
	7	blitter finished	BLIT
4	8	audio channel 2	AUD2
	9	audio channel 0	AUD0
	10	audio channel 3	AUD3
	11	audio channel 1	AUD1
5	12	receiver buffer full	RBF
	13	disk sync pattern found	DSKSYNC
6	14	external INT6 & CIAB	EXTER
	15	special (master enable)	INTEN
7	--	non-maskable interrupt	NMI

Figure 7-4: Interrupt Priorities

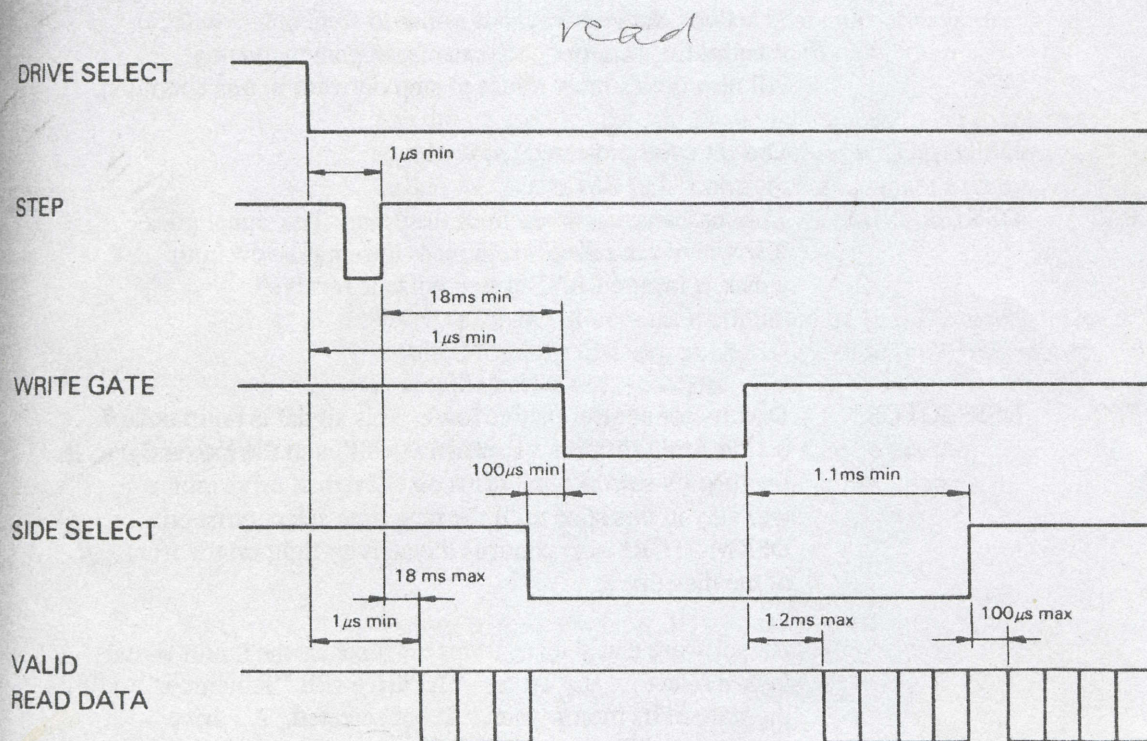
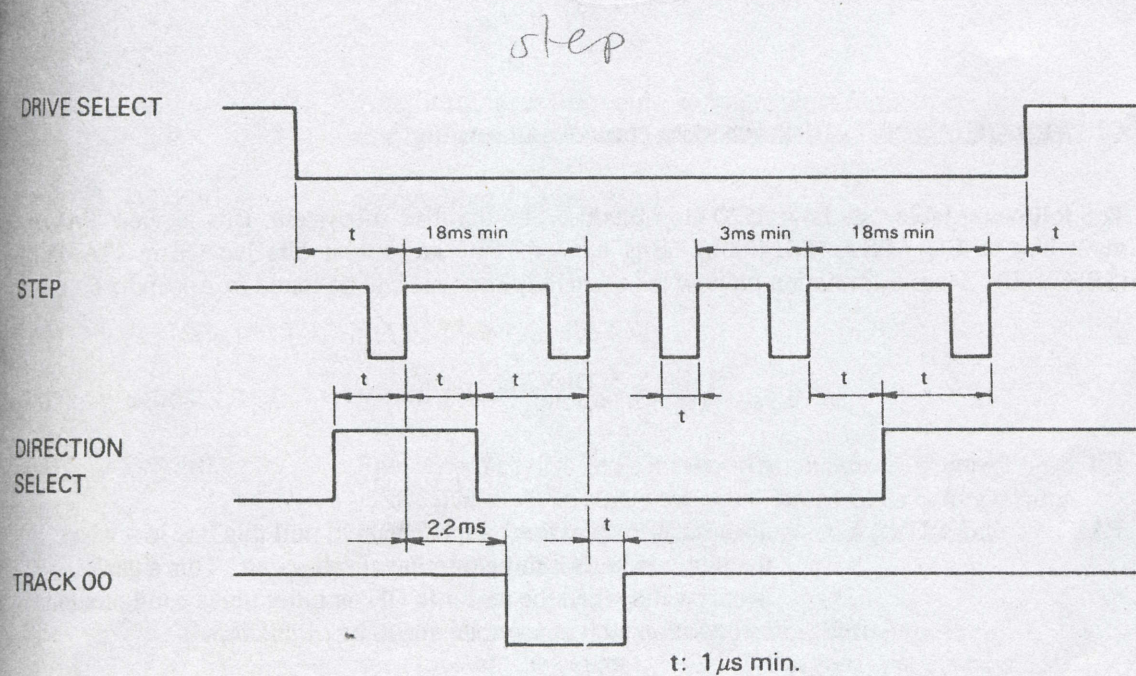


Figure 8-8: Chinon Timing Diagram (cont.)

REGISTERS USED BY THE DISK SUBSYSTEM

The disk subsystem uses two ports on the system's 8520 CIA chips, and several registers in the Paula chip:

CIAAPRA	(\$BFE001)	four input bits for disk sensing
CIABPRB	(\$BFD100)	eight output bits for disk selection, control and stepping
ADKCON	(\$DFF09E)	control bits (write only register)
ADKCONR	(\$DFF010)	control bits (read only register)
DSKPTH	(\$DFF020)	DMA pointer (32 bits)
DSKLEN	(\$DFF024)	length of DMA
DSKBYTR	(\$DFF01A)	Disk data byte and status read
DSKSYNC	(\$DFF07E)	Disk sync finder; holds a match word

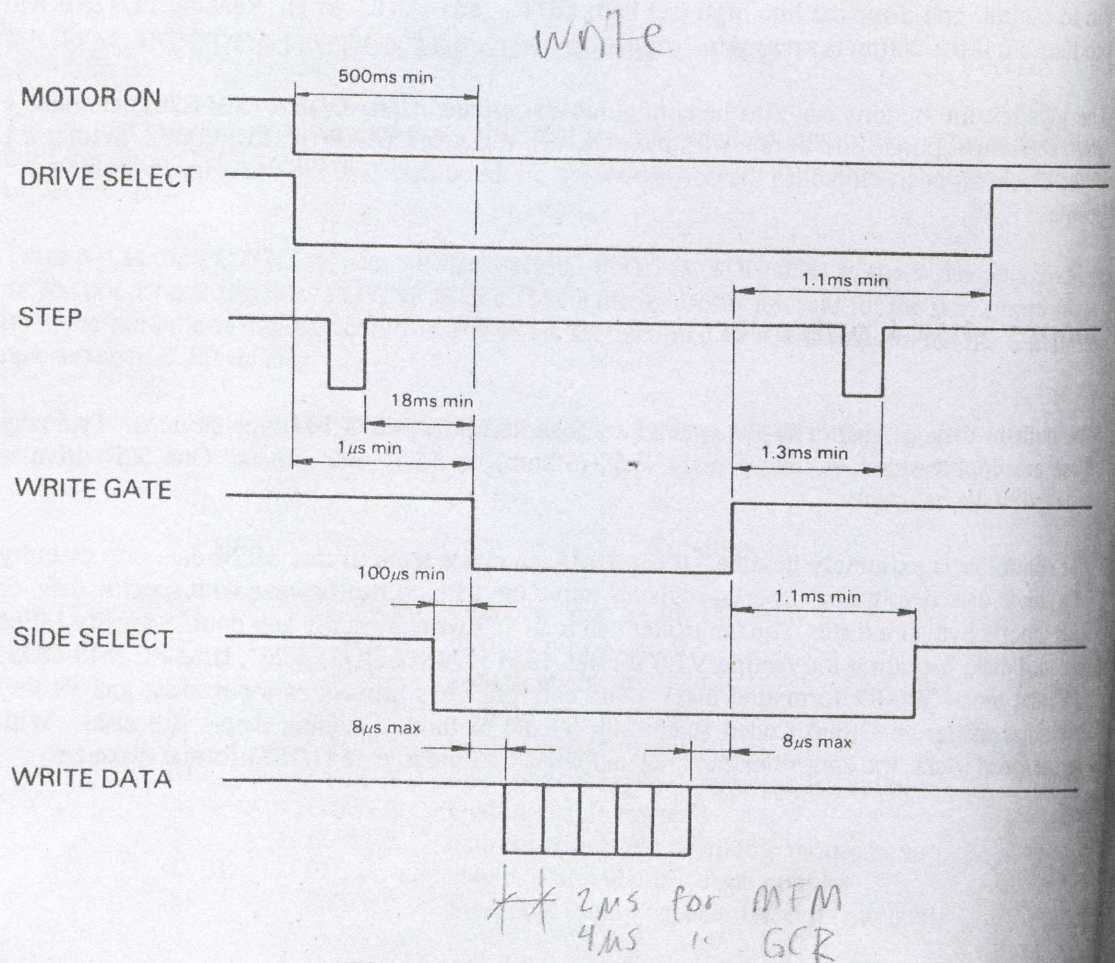


Figure 8-7: Chinon Timing Diagram

BLITTER PRIMER (PAL OCS)

List of all custom registers and bits relevant to Blitting!

\$DFF0xx		
00	BLTDDAT	Blitter Destination Data, last word result after blit
02	DMACONR	DMA Control Read
14	BLTDONE	(0 when ready again)*
13	BZERO	(1=all blitter output was 0)
1E	INTREQR	Interrupt Request, read.
6		Set when a blitter finished interrupt has been triggered*
2E	COPCON	Copper Control Write.
1		CDANG (1=copper can write to blitter registers). Cleared upon machine reset.
40	BLTCON0	Blitter Control 0:
15-12		A shift 0-15
11-8		Enable blitter channel ABCD
7-0		Minterm, bits 7-0*
42	BLTCON1	Blitter Control 1:
15-12		B shift 0-15
4		Exclusive Fill Enable (excludes last (left) edge)*
3		Inclusive Fill Enable (includes last (left) edge)*
2		Fill Carry In (invert fill)
1		DESC (1=descending mode)*
0		Line (1=line mode)
44	BLTAFWM	Blitter source A first word mask*
46	BLTALWM	Blitter source A last word mask*
48	BLTCPT	Longword pointer to source C. Bit 0 is IGNORED!
4C	BLTBPT	Longword pointer to source B. Bit 0 is IGNORED!
50	BLTAPT	Longword pointer to source A. Bit 0 is IGNORED!
54	BLTDPT	Longword pointer to dest D. Bit 0 is IGNORED!
58	BLTSIZE	Blitter Size, up to 1024x1024 pixels.
15-6		Height, 0=1024.
5-0		Width in words, 0=64 words.
60	BLTCMOD	Modulo in bytes, always "plus to skip margin". Minus requires masking.
62	BLTRMOD	Modulo in bytes, always "plus to skip margin". Minus requires masking.
64	BLTAMOD	Modulo in bytes, always "plus to skip margin". Minus requires masking.
66	BLTDMOD	Modulo in bytes, always "plus to skip margin". Minus requires masking.
70	BLTCDAT	Initial data constant for source if source channel disabled in BLTCON0.
72	BLTBDAT	Initial data constant for source if source channel disabled in BLTCON0.
74	BLTADAT	Initial data constant for source if source channel disabled in BLTCON0.
96	DMACON	DMA Control write.
15		Set/Clr (1=Set bits, 0=clear bits)
10		BLTPRI (1=full (instead of partial) priority over CPU)
9		DMAEN (1=enable DMA bits 8-0)
6		BLTEN (1=enable blitter DMA)
9A	INTENA	Interrupt Enable write.
15		Set/Clr
14		INTEN (1=enable interrupts)
6		BLIT (1=enable Blitter Ready level 3 interrupt)
9C	INTREQ	Interrupt Request write.
15		Set/Clr
6		Clear this bit at the end of the interrupt routine to clear the interrupt request.

* NOTES:

BLTDONE:
For compatibility with pre-fat Agnus machines, read this bit once before reading it again and relying on its value.

DESC:

In descending mode, the blitter
- decrements the pointers
- subtracts the modulus
- shifts to the left
Use ONLY descending mode to fill areas.

Fill enable bits:

It is not defined what happens if both EFE and IFE are 1!

BLTAXWM:

For 1-word-wide blits, both masks are logically combined with AND.

Blitter operation order:

1. Mask
2. Shift
3. Logical combination of sources
4. (Area Fill)
5. BZERO result.

Register loading order:

1. Wait for BLTDONE=0.
2. Load any BLTxDAT registers (if used) AFTER loading registers that shift it. (BLTCON0 & 1)
3. Load BLTADAT (if used) BEFORE loading registers that mask it. (BLTAXWM)
4. Load BLTSIZE last to start the blit.

So: Wait, BLTCON, BLTxDAT, BLTAXWM, BLTSIZE. Pointers and Modulos may be loaded at any time before BLTSIZE.

Registers that are preserved between blits:

- ones that can be loaded once before a blit loop and never changed:
All from \$DFF040 to \$DFF074, except pointers that will end up where the next word would be in the blit, if the blit had been 1 pixel higher. This includes Fill and Line drawing modes, where Line Drawing pointers will be at the next pixel, if the line had been one pixel longer (usually, delta Y).

Blitter speed:

$t = (n \cdot H \cdot W / 7.09)$ microseconds

- where:

H=height in pixels
W=width in words
n=number of clock ticks per blitter cycle

Blitter cycle: (see also Cycle Sequence Diagram below)

Calculate according to which DMA channels are enabled. Minimum is 4 ticks; add:

- A 0 ticks
- B 2 ticks
- C 0 ticks
- D 0 ticks
- C+D 2 ticks

This means an ABCD blit has a cycle of 4+2+2 ticks=8 ticks. For example, using BLTAPT and BLTBDAT is more efficient than using BLTBPT and BLTADAT.

Line mode always takes 8 ticks PER PIXEL.

Blitter and other DMA

Disk, Audio, Display, Sprite, and Copper DMA all have priority over the blitter.
- disable all DMA that is not used, for example Disk or Audio DMA channels.
- enable certain sprites or bitplanes just where they are needed, by putting a copper WAIT instruction on the scanline above and below, with a MOVE to enable and disable use, respectively.
- enabling 4 bitplanes will leave only 50% of the cycles of a display line for blitter use; enabling 6 leaves only 25% of the cycles; assuming BLTPRI is enabled. If not, cycles left are shared between CPU and blitter.
- Blitter DMA steals up to 80 cycles/line, sprite up to 16, audio DMA 4 and Disk DMA 3.
- this means that you should only start and stop bitplane DMA for those scan lines that contain pixels.
- it also means that DMA-intensive blits are best done while Denise displays few or 0 bitplanes. So blit your bobs after the copper reaches the last display line, and save the clearing of the screen (1 DMA channel) for when displaying more bitplanes.

CPU and other DMA (including Blitter DMA)

- odd CPU (NOTE: CPU!) cycles are for internal operation, even are for memory operations, if no DMA (like bitplane or blitter DMA) steals it.
- If BLTPRI is enabled, Blitter will take every memory cycle** left from the other DMA (above). If it is disabled, Blitter will grant every 4th cycle left, to the CPU.
- what this means is that if more than 4 bitplanes are on, bitplane DMA will start stealing cycles from Blitter and CPU. If the Blitter is then running with BLTPRI enabled, CPU will stop until Blitter is finished, because Blitter uses all the even cycles. If BLTPRI is disabled, it will get every 4th cycle left - 12.5% of the cycles - if 6 bitplanes are enabled.
- this indicates that there is little or no use disabling BLTPRI with 5 or 6 bitplanes enabled. It also indicates that CPU code running while many bitplanes are on, should avoid accessing memory. So clear the screen with a blit and calculate internally then. (MULS/DIVS are great, really!)
- upto 4 bitplanes affect Blitter/CPU relatively little, an estimated 3-4% slowdown for each bitplane added.

** See this Cycle Sequence Diagram for cycle usage.

USEx bits	DMAx	Cycle Sequence
F	ABCD	A0 B0 C0 -- A1 B1 C1 D0 A2 B2 C2 D1 D2
E	ABC	A0 B0 C0 A1 B1 C1 A2 B2 C2
D	AB D	A0 B0 -- A1 B1 D0 A2 B2 D1 -- D2
C	AB	A0 B0 -- A1 B1 -- A2 B2
B	A CD	A0 C0 -- A1 C1 D0 A2 C2 D1 -- D2
A	A C	A0 C0 A1 C1 A2 C2
9	A D	A0 -- A1 D0 A2 D1 -- D2
8	A	A0 -- A1 -- A2
7	BCD	B0 C0 -- B1 C1 D0 -- B2 C2 D1 -- D2
6	BC	B0 C0 -- B1 C1 -- B2 C2
5	B D	B0 -- B1 D0 -- B2 D1 -- D2
4	B	B0 -- B1 -- B2
3	CD	C0 -- C1 D0 -- C2 D1 -- D2
2	C	C0 -- C1 -- C2
1	D	D0 -- D1 -- D2
0		-- -- -- --

Notes:

- 3-word blit shown. Core cycle utilization (from D0 until D1) is underlined.
- No fill.
- No competing bus activity.

Observations:

- Full blit (F) uses all memory cycles in 8 ticks; D and B also, but in 6 ticks.
- B and D are more efficient than 7
- 9 is more efficient than 5 and 3
- UNLESS the CPU has a lot to do during the blit and there is no new blit queued in the copper or interrupt(!), in which case BLTPRI can be disabled for better CPU utilization. This time can rarely be so utilized, except when blits are small or CPU is sorting/processing lists or clearing/filling memory.
- Clear (1) finishes in the same time as a Copy (9) - except the former leaves every other cycle available for CPU use. So don't waste it!

BLTCON registers (longword)

- for normal blits - has the form:

\$aummb000 (or \$aummb002 for Descending mode)

- where:

a=A shift, 0 to \$F
b=B shift, 0 to \$F
u=USEx bits, see Cycle Sequence Diagram above.
m=Minterm, see below

MINTERM:

Use the following diagram to calculate the 8-bit minterm. For those combinations that should yield a 1 bit in the destination, enter a 1 in the D column, and for all others, 0. Then those 8 bits vertically form the Minterm.

In the below example, we want the destination to be 1 ONLY if A or C is 1:

Bit	Channel
ABC->D	
0	000 0
1	001 1
2	010 0
3	011 1
4	100 1
5	101 1
6	110 1
7	111 1

= \$fc

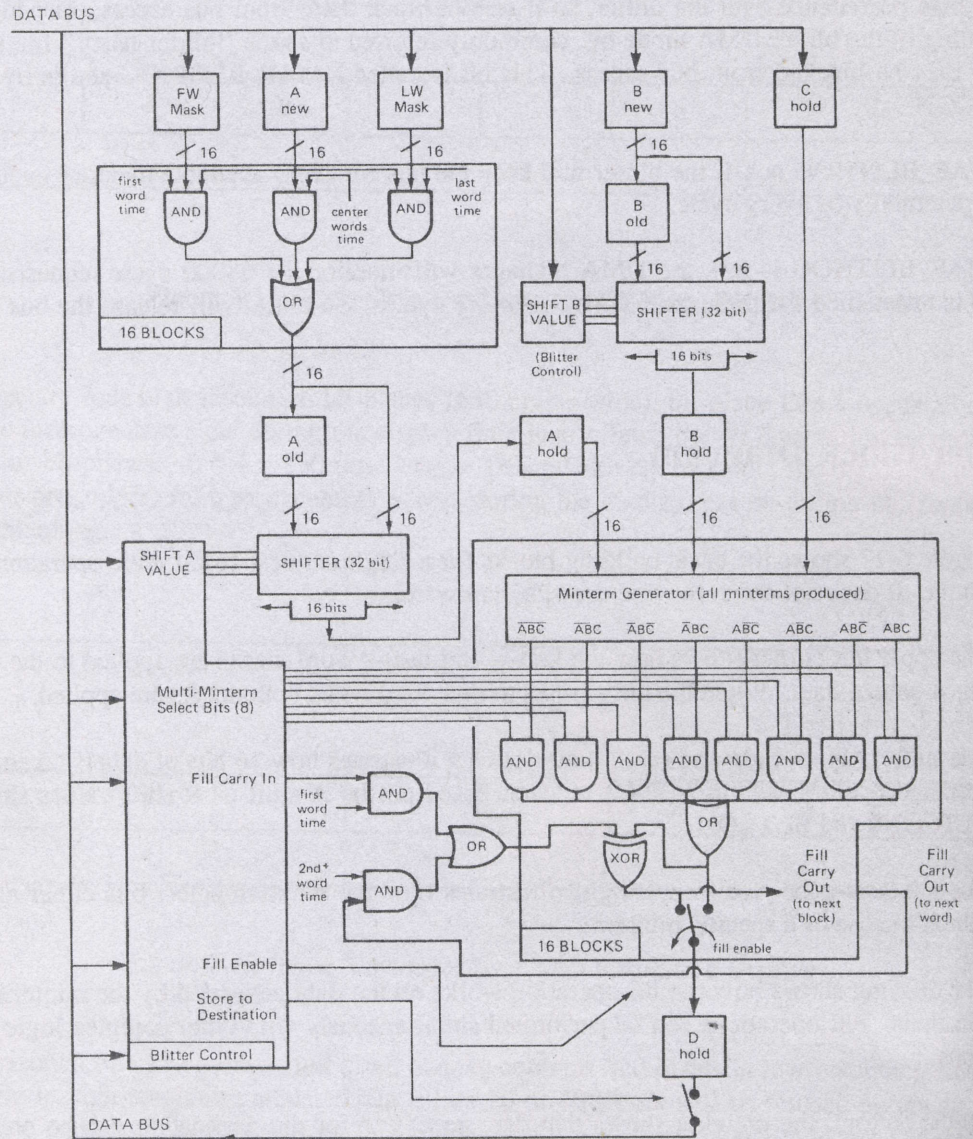


Figure 6-13: Blitter Block Diagram

Editor:

- T** [line] - Top of file/jump to line
- B** - Bottom of file
- L** [text] - Search for text
- ZL** [lines] - Zap number of lines from cursor
- P** [lines] - Print number of lines from cursor

Memory:

- M** [.size][addr] - Edit memory
- D** [addr] - Disassemble
- H** [.size][addr] - Hex dump
- N** [addr] - ASCII dump
- @D** [addr] - Disassemble line
- @A** [addr] - Assemble line
- @H** [.size][addr] - Hex dump line
- @N** [addr] - ASCII dump line
- S** [.size] - Search
- F** [.size] - Fill
- C** [.size] - Copy
- Q** - Compare

Insert:

- ID** - Insert disassemble
- IH** [.size] - Insert hex dump
- IN** - Insert ASCII

Assemble:

- A** - Assemble source in editor
- @A** [addr] - Assemble to memory
- AO** - Assemble optimize
- AD** - Assemble debug
- =S** - Symbol table print
- =** *(show source stats)*

Monitor:

- J** [addr] - Jump to address
- G** [addr] - Go to address
- K** [steps] - Single step n steps
- X** [register] - View/edit register values
- ZB** - Zap breakpoint buffer

Disk:

- RS** [drive] - Read sector
- RT** [drive] - Read track
- WS** [drive] - Write sector
- WT** [drive] - Write track
- CC** [drive] - Calculate boot checksum




Others:

- E** - Load extern files
- V** [path] - View directory
- >** - Direct output (to PRT: or DFn: ...)
- ?** [expr] - Calculate value

68000 Timings

Addressing Mode			Byte, Word	Long
Register				
Dn	Data Register Direct		0(0/0)	0(0/0)
An	Address Register Direct		0(0/0)	0(0/0)
Memory				
(An)	Address Register Indirect		4(1/0)	8(2/0)
(An)+	Address Register Indirect with Postincrement		4(1/0)	8(2/0)
(-An)	Address Register Indirect with Predecrement		8(1/0)	10(2/0)
(d16, An)	Address Register Indirect with Displacement		8(2/0)	12(3/0)
(d8, An, Xn)*	Address Register Indirect with Index		10(2/0)	12(3/0)
(xxx) W	Absolute Short		8(2/0)	12(3/0)
(xxx) L	Absolute Long		12(3/0)	16(4/0)
(d8, PC)	Program Counter Indirect with Displacement		8(2/0)	12(3/0)
(d16, PC, Xn)*	Program Counter Indirect with Index		10(2/0)	14(3/0)
#<data>	Immediate		4(1/0)	8(2/0)

*The size of the index register (Xn) does not affect execution time.

Source		Destination									
	Dn	An	(An)	(An)+	(-An) 	(d16, An)	(d8, An, Xn)*	(xxx), W	(xxx), L	(xxx), L	(xxx), L
Dn	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1) 	12(2/1)	14(2/1)	12(2/1)	16(3/1)	16(3/1)	16(3/1)
An	4(1/0)	4(1/0)	8(1/1)	8(1/1)	8(1/1)	12(2/1)	14(2/1)	12(2/1)	16(3/1)	16(3/1)	16(3/1)
(An)	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)	20(4/1)	20(4/1)
(An)+	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)	20(4/1)	20(4/1)
(-An) 	10(2/0)	10(2/0)	14(2/1)	14(2/1)	14(2/1)	18(3/1)	20(3/1)	18(3/1)	22(4/1)	22(4/1)	22(4/1)
(d16, An)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)	24(5/1)	24(5/1)
(d8, An, Xn)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)	26(5/1)	26(5/1)
(xxx), W	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)	24(5/1)	24(5/1)
(xxx), L	16(4/0)	16(4/0)	20(4/1)	20(4/1)	20(4/1)	24(5/1)	26(5/1)	24(5/1)	28(6/1)	28(6/1)	28(6/1)
(d16, PC)	12(3/0)	12(3/0)	16(3/1)	16(3/1)	16(3/1)	20(4/1)	22(4/1)	20(4/1)	24(5/1)	24(5/1)	24(5/1)
(d8, PC, Xn)*	14(3/0)	14(3/0)	18(3/1)	18(3/1)	18(3/1)	22(4/1)	24(4/1)	22(4/1)	26(5/1)	26(5/1)	26(5/1)
<data>	8(2/0)	8(2/0)	12(2/1)	12(2/1)	12(2/1)	16(3/1)	18(3/1)	16(3/1)	20(4/1)	20(4/1)	20(4/1)

MISC.

Table 8-11. Multiprecision Instruction Execution Times

Instruction	Size	op Dn, Dn	op M, M
ADDX	Byte, Word	4(1/0)	18(3/1)
	Long	8(1/0)	30(5/2)
CMPM	Byte, Word	—	12(3/0)
	Long	—	20(5/0)
SUBX	Byte, Word	4(1/0)	18(3/1)
	Long	8(1/0)	30(5/2)
ABCD	Byte	6(1/0)	18(3/1)
SBCD	Byte	6(1/0)	18(3/1)

Table 8-14. Exception Processing Execution Times

Exception	Periods
Address Error	50(4/7)
Bus Error	50(4/7)
CHK Instruction	40(4/3)+
Divide by Zero	38(4/3)+
Illegal Instruction	34(4/3)
Interrupt	44(5/3)*
Privilege Violation	34(4/3)
RESET**	40(6/0)
Trace	34(4/3)
TRAP Instruction	34(4/3)
TRAPV Instruction	34(5/3)

+ Add effective address calculation time.

* The interrupt acknowledge cycle is assumed to take four clock periods.

** Indicates the time from when RESET and HALT are first sampled as negated to when instruction execution starts.

Table 8-12. Miscellaneous Instruction Execution Times

Instruction	Size	Register	Memory
ANDI to CCR	Byte	20(3/0)	—
ANDI to SR	Word	20(3/0)	—
CHK (No Trap)	—	10(1/0)+	—
EORI to CCR	Byte	20(3/0)	—
EORI to SR	Word	20(3/0)	—
ORI to CCR	Byte	20(3/0)	—
ORI to SR	Word	20(3/0)	—
MOVE from SR	—	6(1/0)	8(1/1)+
MOVE to CCR	—	12(1/0)	12(1/0)+
MOVE to SR	—	12(2/0)	12(2/0)+
EXG	—	6(1/0)	—
EXT	Word	4(1/0)	—
	Long	4(1/0)	—
LINK	—	16(2/2)	—
MOVE from USP	—	4(1/0)	—
MOVE to USP	—	4(1/0)	—
NOP	—	4(1/0)	—
RESET	—	132(1/0)	—
RTE	—	20(5/0)	—
RTR	—	20(2/0)	—
RTS	—	16(4/0)	—
STOP	—	4(0/0)	—
SWAP	—	4(1/0)	—
TRAPV	—	4(1/0)	—
UNLK	—	12(3/0)	—

+Add effective address calculation time.

Table 8-13. Move Peripheral Instruction Execution Times

Instruction	Size	Register → Memory	Memory → Register
MOVEP	Word	16(2/2)	16(4/0)
	Long	24(2/4)	24(6/0)