ELTEC
elektronik mainz

ELTEC-68K-SYSTEM

Documentation
PIG1/68K

Revision A dated 5/86

DO .68 01749

This document is edited and printed by:

ELTEC Elektronik GmbH, D-6500 Mainz

CONTENTS

How to use this documentation?

This documentation is divided into four parts to give only as much information as needed for a certain purpose.

Part A contains all general specifications of the product and its documentation like scope of delivery and technical specs and the general nomenclature used in this manual.

Part B decribes the procedure you should follow to adapt the product to your specific system and peripherals.

Part C gives detailed information to the user who wants to program the board himself.

Part D gives detailed information about the hardware-aspects of the board for service etc.

If you only use the board with a software package, that means as a user, not as a programmer, you should find complete information while reading only part B.

ELTEC Elektronik GmbH

E L T E C 6 8 K - S Y S T E M

DOCUMENTATION

PIG1/68k

PART A

DO.68 01749

1 Scope of delivery

1.1 Hardware

PIG1/68k.....Order No FE .68 01748
  Consisting of
    PIG1/68k with 1MByte frame buffer
    Dot clock generator of 55MHz
    Factory configured as described
      in part B, chapter 2
    Documentation PIG1/68k
    5 cables FE KAB 1888

PIG2/68k.....Order No FE .68 01750
  Board set consisting of
    PIG1/68k with 1MByte frame buffer
      and local extension of another
      1MByte frame buffer
    Dot clock generator of 55MHz
    Factory configured as documented
    Documentation-package
      consisting of documentations
      PIG1/68k and PIG2/68k
    2 cables FE KAB 1888
    8 cables (4 sets) FE KAB 1726

Cable (SMB-Coax to BNC, 3 mtr)....Order No FE KAB 01888

Cable-set (2 * SMB-Coax 20 cm)....Order No FE KAB 01726

Documentation PIG1/68k.....Order No DO .68 01749

Documentation PIG2/68k.....Order No DO .68 01751

1.2 Software support

PIGpac for OS-9/68000 8"/77T... .Order No AS OS9 01753

PIGpac for OS-9/68000 5"/40T... .Order No AS OS9 01762

PIGpac for OS-9/68000 5"/80T... .Order No AS OS9 01763

PIGpac is a complete, modern grafics package, which is completely window-oriented. It consists of an OS-9/68000 device driver with a terminal emulation and an interface to assembler and C language. Some samples and several monitor tables are included.

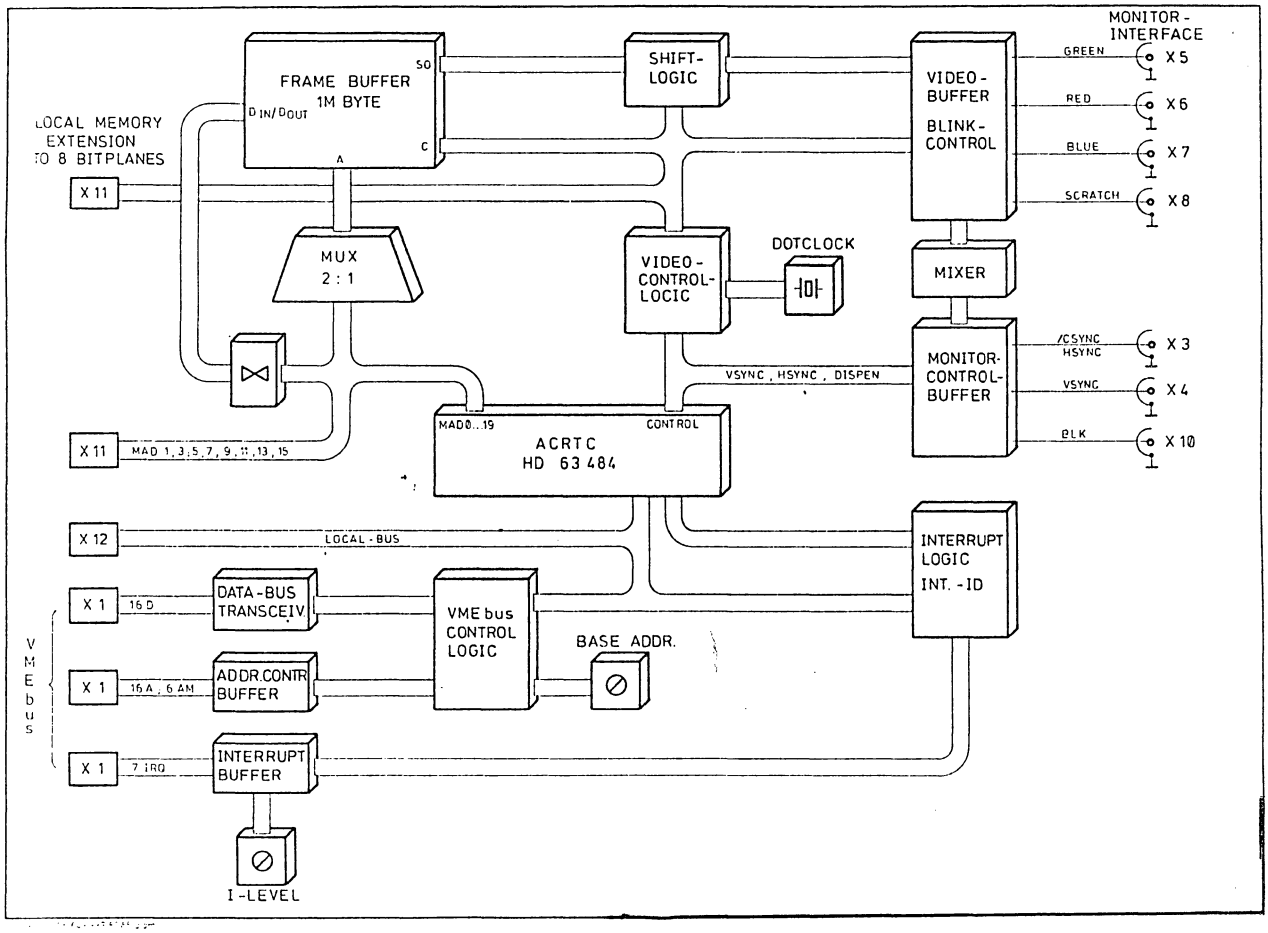
GKS-0A for OS-9/68000 8"/77T.....Order No AS OS9 01770

GKS-0A for OS-9/68000 5"/40T.....Order No AS OS9 01773

GKS-0A for OS-9/68000 5"/80T.....Order No AS OS9 01771

GKS-0A is an implementation of the Grafical Kernel System in accordance to ISO 7942 and DIN-Standard. A C-binding is provided.

Dia A.2.1: Block diagram PIG1/68k



2 Technical specifications

- -VMEbus interface in accordance to revision C with interrupter option on all levels
- -Grafics controller ACRTC HD63484
- -Black and white, greyscale and color operation
- -Maximum dotclock 64MHz in a l l bitplane configurations
- -Programmable sync format and flexible hardware interface to control nearly every raster scan monitor on the market
- -Modern dual-port video RAM architecture to provide highest drawing speed; useful especially with the ACRTC's powerful bit-block operations
- -Three bitplane configurations:
  - One bitplane with 8 MegaPixel capacity
  - Two bitplanes, each with 4 MegaPixel capacity
  - Four bitplanes, each with 2 MegaPixel capacity
- -Option PIG2/68k to get 8 bitplanes with no restriction in dotclock
- -Video-Zooming with factors of 1, 2 or 4 for both directions; separate vertical zooming with factors of 1 to 16
- -Pixel-by-pixel panning facility in both directions
- -Exact pixel-based blinking attribute
- -Access of local CPU-module prepared
- -Software support PIGpac
- -Software support GKS

3 VME Eurocard Specification PIG1/68k

Slave Data Transfer Options

A16 : D16

Interrupter Options

Any one of
I(1),I(2),I(3),I(4),I(5),I(6) or I(7) (STAT)

Environmental Options

Storage temperature: -55...+85 degree Celsius
Operating temperature: 0...+70 degree Celsius
Maximum operating humidity: 85 Percent relative

Power options

max 4.2 A (3.8 A typ) at +5 Volt

Physical configuration options

NEXP

4 Nomenclature of this manual

A hexadecimal notation is identified by a leading dollar sign ("$").
A binary notation is identified by a leading "B".

A logic low (high) level is identified by "L" ("H") or "0" ("1") independent of the assertion-type of the signal.

Names of level-controlled signals preceded by a slash ("/") indicate that this signal is active low.
Names of edge-controlled signals preceded by a slash ("/") indicate that this signal becomes active with the trailing edge.

Positions of jumpers refer to those shown in diagram dia B.1.1, which are identical to pin numbers. If not mentioned otherwise, "J401:1-2" for instance means, that jumper J401 must be set to connect pins 1 and 2.

ELTEC Elektronik GmbH

E L T E C 6 8 K - S Y S T E M

DOCUMENTATION

PIG1/68k

PART B

DO.68 01749

1 Introduction

This part of the manual gives you all information necessary to adapt the board to your system, ie to your VMEbus-computer and monitor. If you use ELTEC's PIGpac there should be no need to read other parts of this manual to get started.

Please follow this procedure for adaption:

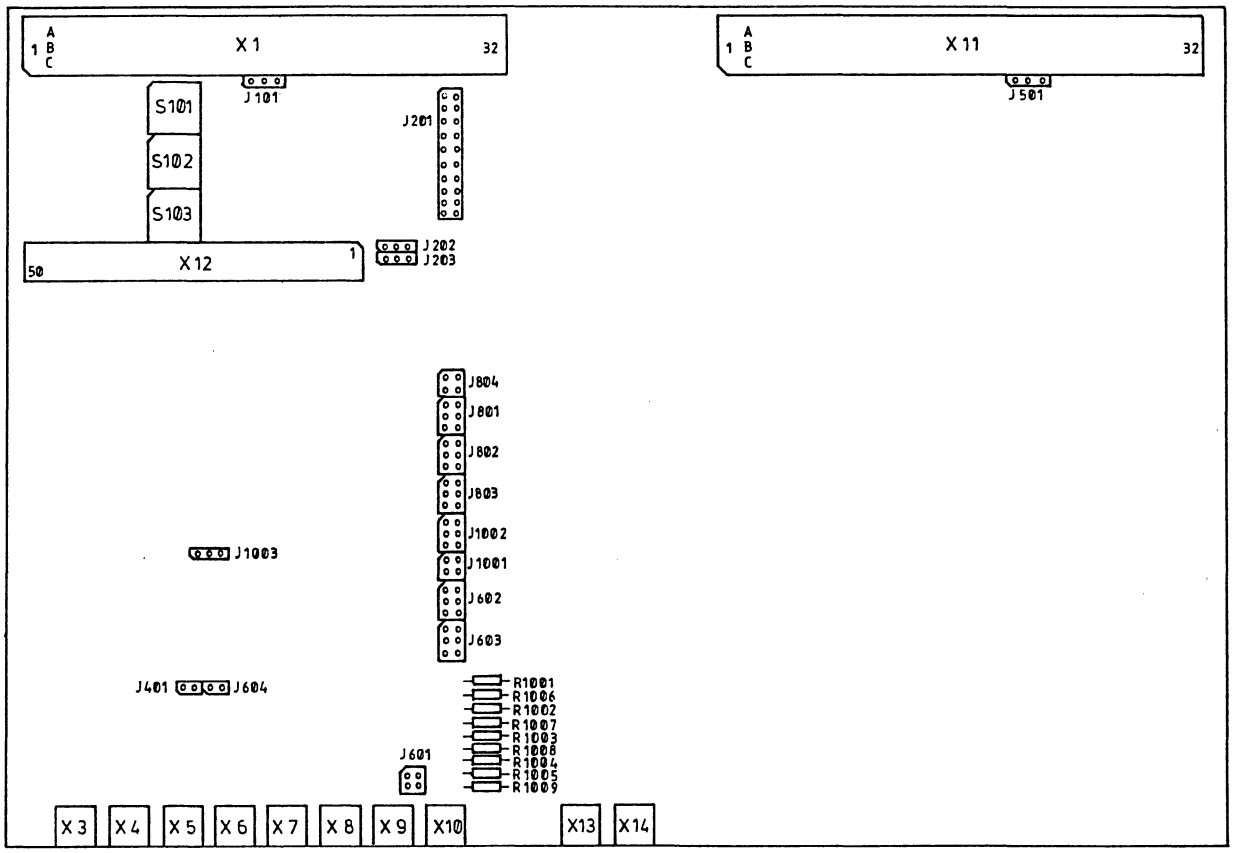
Adaption of the base address -	refer to chapter 3
Adaption of the address modifiers -	refer to chapter 3
Adaption of desired interrupt level -	refer to chapter 3
Adaption to your monitor -	refer to chapter 4
Blinking -	refer to chapter 4
Adaption of numbers of bitplanes -	refer to chapter 5
Master/slave operation -	refer to chapter 6
External access to local bus -	refer to chapter 7

Block diagram Dia A.2.1 gives an overview over the whole board.
Diagram Dia B.1.1 shows the position of all jumpers, connectors and switches.

In appendix B you will find a complete list of all jumpers and switches.
In appendix C you will find a complete list of all connectors.

In chapter B.2 you find the default setting as board is shipped.

Dia B.1.1: Switches, jumpers, connectors PIG1/68k



2 Default setting as board is shipped

Base address $FF8D00

S101:8
S102:D
S103:0 (Non-privileged; i.e. User-Mode I/O)
- Refer to chapter 3.1!

Address modifier short supervisory I/O

S103:0
J101:1-2 (AM2 decoded)
- Refer to chapter 3.2!

Interrupter I(7)

J201:1-2
- Refer to chapter 3.4!

Monitor interface

- Sync signals — separate syncs, positive polarity, TTL-level

  J1001:3-4
  J1002:5-6

- Video outputs — TTL-level

  R1001 == 0 Ohm
  R1002 == 0 Ohm
  R1003 == 0 Ohm
  R1004 == 0 Ohm
  R1005 == no resistor
  R1006 == no resistor
  R1007 == no resistor
  R1008 == no resistor
  R1009 == no resistor

- Dot clock — U601 == 55 MHz
- Refer to chapter 4!

Blinking

disabled

  J1003:1-2

- refer to chapter 4!

Number of bitplanes — 4 bitplanes

J602:1-2
J603:1-2
J801:1-2
J802:1-2
J803:1-2
J804:1-2
J804:3-4
J501:1-2
- refer to chapter 5!

Master/slave-mode — stand-alone

J401:1-2
J601:1-2
J604:1-2
- refer to chapter 6!

External access to local bus — disabled

J202:1-2
J203:1-2
- refer to chapter 7!

3 VMEbus-Interface

The slave PIG1/68k occupies 64 Byte in short I/O addressing range.

All on-board devices are byte- or word-oriented. While trying to access the board with longword bus transfer cycle or with single-byte transfer on even addresses, a bus-error is generated.

The interrupter module can generate interrupts on all VMEbus interrupt levels, selectable by a jumper. Several interrupt sources can be chosen by software.

The complete VMEbus interface logic is realized via PAL's. So it is possible to make changes in addressing the on-board devices, decoding of address modifiers etc very easy. ELTEC's firmware, however, expects the features documented herein.

The bus grant daisy chain is closed in the printed circuit, so there is no need for modifications on the backplane.

You always have to remove the jumper for the corresponding interrupt acknowledge daisy chain on the backplane.

3.1 Base address

The base address can be switched via hex-switches S101, S102 and S103 in steps of 64 Bytes (address-lines A15 thru A06 - short I/O).

For adaption to the desired base address its highest nibble must be switched with S101. S102 is for the next nibble (A11 thru A08). S103 determines with its lower bits A07 and A06 of the base address. The upper bits of S103 are used to decode adressmodifier AM2 and AM1. Please refer to chapter B.3.2!

3.2 Address modifier

The address modifiers AM2 and AM1 can be switched via hex-switch S103. Please refer to table Tab B.3.1!

AM0 is decoded in a PAL to assert only with AM0=H.

AM5 and AM3 are wired to H, AM4 is wired to L (16-Bit addressing).

With jumper J101 set to position 2-3 it is possible to decode Adressmodifier AM2 as "don't care" to decode the board in the non-proivileged and in the supervisory addressing range.

Please refer to appendix A (Address modifier in the VMEbus-specification) for further details.

Tab B.3.1: Switching the desired address modifier

AM-Code Hex	AM5	AM4	AM3	AM2	AM1	AM0	S103	Note
29	1	0	1	0	0	1	0-3	non-privileged
2B	1	0	1	0	1	1	4-7	reserved
2D	1	0	1	1	0	1	8-B	supervisory
2F	1	0	1	1	1	1	C-F	reserved

Note:
- The lower two bits of S103 define Address lines A07 and A06 of base address.
- Decoding of AM2 is only done if jumper J101 is in position 1-2

3.3 Device addresses

Table Tab B.3.2 shows the relative addresses of all on-board devices.

Tab B.3.2: Relative addresses of all devices

rel addr hex	Device	Transfer-type R/W	Length
0	HD63484 status reg	Read	Word
0	HD63484 address reg	Write	Word
2	HD63484 control regs	Both	Word
5	Hardware status reg	Both	Byte

3.4 Interrupter

The interrupter module on PIG1/68k is able to generate interrupts on one of any VMEbus interrupt-level. The selection is done by jumper J201. For details please refer to table Tab B.3.3! Please be sure to set the levels of request and acknowledge identically.

If you use the on-board interrupter, you must remove the jumper for the interrupt acknowledge daisy chain on the backplane.

For programming the internal interrupt sources and the interrupt-ID-Byte (vector) please refer to part C!

Tab B.3.3: Selecting the interrupt level

	Jumpering J201	
Desired interrupt level	VMEbus IRQ-Level	Interrupt-acknowledge level decoding
I(7) = /IRQ7	J201:1-2	Level 7 = not set
I(6) = /IRQ6	J201:3-4	Level 6 = J201:19-20
I(5) = /IRQ5	J201:5-6	Level 5 = J201:17-18
I(4) = /IRQ4	J201:7-8	Level 4 = J201:17-18 J201:19-20
I(3) = /IRQ3	J201:9-10	Level 3 = J201:15-16
I(2) = /IRQ2	J201:11-12	Level 2 = J201:15-16 J201:19-20
I(1) = /IRQ1	J201:13-14	Level 1 = J201:15-16 J201:17-18 J201:19-20

4 Adaption to your monitor

4.1 Monitor synchronisation

PIG1/68k is able to drive monitors with different logic-levels on the sync line(s). The voltage level is always TTL-level.

Refer to table B.4.1 to see what to do to adapt PIG1/68k to the sync line(s) of your monitor. Please be sure to program the grafic display controller with a suitable parameter set. You have to use the corresponding dotclock generator U601, too.

Refer to Part C for further information about parameter sets for synchronisation and dotclock.

Refer to documentation of PIGpac to choose the correct parameter set for your monitor under control of this package.

To connect your monitor to PIG1/68k you should use the added coaxial cables.

Tab B.4.1: Monitor synchronisation PIG1/68k

HSY	VSY	CSY	Jumper	Connection
P	P	-	J1002:5-6	HSY: X3
			J1001:3-4	VSY: X4
P	N	-	J1002:5-6	HSY: X3
			J1001:1-2	VSY: X4
N	P	-	J1002:3-4	HSY: X3
			J1001:3-4	VSY: X4
N	N	-	J1002:3-4	HSY: X3
			J1001:1-2	VSY: X4
-	-	N	J1002:1-2	CSY: X3

Notes:
HSY.....Horizontal synchronisation
VSY.....Vertical synchronisation
CSY.....Composite synchronisation
P.....Positive logic-level (active high)
N.....Negative logic-level (active low)

4.2 Video signals

4.2.1 Separate lines

PIG1/68k is able to generate up to four bitplanes per pixel. Normally the voltage-level is TTL. Please notice, that it is possible to adapt the hardware to reduce the number of planes in order to get higher pixel capacity per plane. Please refer to chapter B.5!

There is a possibility to reduce the output voltage by changing resistors R1001 to R1004. In combination with the input-impedance (mostly 75Ohm) of the monitor the result is an "analogue" signal. A good approach for 75Ohm-input and 1Vpp is to use 100Ohm resistors instead of 0Ohm.

Table B.4.2 shows you the connectors for the bitplanes. In case of using a standard RGB-color monitor, one bitplane is not used. If you want to apply a color monitor with a separate white- or intensity-input you can drive it by this scratch-plane. In other cases you can use the plane to control the generate shapes to control the blink attribute (see chapter 4.3!).

Tab B.4.2: Bit planes and connectors

Monitor	No of planes	X5	X6	X7	X8
B/W	4	P3	P2	P1	P0/S
B/W	2	P3		P2	
B/W	1	P3			
Color	4	G	R	B	S
Color	2	G		R/S	
Color	1	G			

Notes:
P3-P0...Weight of greyscale modulation
to be connected to a monitor
with integrated D/A-Conversion
R.....Red-channel
G.....Green-channel
B.....Blue-channel
S.....Scratch-plane

4.2.2 Composite signal

Via resistors R1001 to R1008 it can be build a simple d/a-converter (R-2*R-ladder-network) generating an analogue signal at connector X5. Please notice, that this conversion is not always free of glitches. Respectively with dotclocks of more than 40MHz you should use this simple converter only as a first approach.

To get a 1Vp-p video-signal (at 75Ohm video input) with 16 greyscales R1001 to R1005 must be 100Ohm and R1006 to R1008 must be 50Ohm. If you want to generate a composite video signal, you have to add the composite synchronsignal via R1009 (=ca 150 Ohm). Please refer to chapter B.4.1, too!

Tab B.4.3: Components of video mixer

	Resistor	
Plane	Serial	Parallel
P3	R1001	--
P2	R1002	R1006
P1	R1003	R1007
P0	R1004	R1008
Terminator	R1005	

Note:
- P() denotes significance of bitplane

4.3 Blinking

The blinking attribute can be statically enabled with jumper J1003 set in position 2-3. If blinking is enabled, all shapes generated in the scratch plane blink against background (video-"black"). The blinking rate can be programmed in the ACRTC. Please refer to part C of this manual or to manual of PIGpac.

5 Number of bitplanes

In most cases PIG1/68k is used in the four bitplane mode, where you get four planes with a capacity of 2Mega-pixels each. Some hardware modifications can be done to define the attributes controlled by the several planes. Please refer to chapter 4!

In some cases it can be necessary to have a higher bitplane capacity. As documented in table Tab B.5.1 you have to make some adaptions.

The X-Y memory configuration depends on programming the ACRTC. Please refer to part C of this manual.

Jumper J501 is reserved for special purposes in combination with the display memory upgrade. It should be set to position 1-2 or left open.

Tab B.5.1: Adapting the bitplane number

Capacity per plane (pixel)	Set jumpers J602	J603	J801	J802	J803	J804
2Mega/4 planes	1-2	1-2	1-2	1-2	1-2	1-2 3-4
4Mega/2 planes	3-4	3-4	3-4	3-4	3-4	1-3 2-4
8Mega/1 plane	5-6	5-6	5-6	5-6	5-6	1-2 3-4

6 Master/slave operation

If you want to use several PIG1/68k in parallel to work on the same monitor, you must define one board as master, the other(s) as slave(s).

Please be sure to make the following adaptions:

Master
- Set jumper J601:1-2 and J601:3-4
- Set jumpers J401 and J604 (default)

Slave
- Set jumper J601:1-3
- Remove jumpers J401 and J604
- Remove U601

Connect master and slave via X9, X13, X14
(use cable-set FE KAB 1726!)

The stand-alone mode is nearly the same as the master mode with no connection to other PIG1/68k's and with jumper J601:1-2 only.

Of course it is possible to have several PIG1/68k in stand-alone-mode in one system, having each of them work to a separate monitor.

Please refer to PIGpac manual for software support!

7 External access to local bus

The complete local address- and databus and most of the lines of local controlbus are wired to connector X12. So it is possible to access to all local devices of PIG1/68k by a separate CPU and/or a DMA-controller, situated on a piggy-back-board, having full advantage of a separate bus. In low-cost applications a local CPU-module can access without the need of a separate VMEbus-backplane.

To make external access possible it is necessary to disable the VMEbus-buffers and -latches. Jumpers J202 and J203 are intended for this purpose.

Because all lines on X12 are decoupled VMEbus-lines, the buslogic and timing on the local bus is the same as on the VMEbus.

To enable the external access, please do the following modifications:

- Set J202:2-3, instead of J202:1-2
- Set J203:2-3, instead of J202:1-2

ELTEC Elektronik GmbH

E L T E C 6 8 K - S Y S T E M

DOCUMENTATION

PIG1/68k

PART C

DO.68 01749

1 Registers of the ACRTC

The ACRTC occupies two wordwide VMEbus address locations, which are documented in chapter B.3.3 of this manual. The ACRTC must be accessed as a wordwide device. In some cases only the lowest 8 bits are siginificant, however.

In this chapter a short introduction is given to the ACRTC's programming model. For more details you should read the data sheet of the controller and other literature, which is published by Hitachi Ltd.

The ACRTC has over two hundred bytes of accessible registers. These are organized as Hardware, Directly and FIFO accessible.

1.1 Hardware accessible registers

The lower address location of the ACRTC provides the status register, when a read-cycle is performed. The status register summarizes the ACRTC state and is used by the CPU to monitor the overall operation of the ACRTC. When a write-cycle is performed to the lower address-location, the address register is activated, in order to program the ACRTC with the address of the desired directly accessible internal register.

Only the lower 8 bits of both registers are significant.

1.2 Directly accessible registers

These registers are accessed by prior loading of the address register with the chosen register address. When the CPU accesses the higher VMEbus-address location, the chosen register is activated.

The FIFO entry enables access to FIFO accessible registers using the ACRTC's read and write FIFO's.

The Command Control Register CCR 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 OMR 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 DCR allows the independent enabling and diabling of each of the four ACRTC logical display screens (Base, Upper, Lower and Window). Also this register contains 8 bits of user defineable video attributes, the Attribute Register ATR.

The Timing Control RAM TCR contains registers which define ACRTC video timing. This includes timing specification registers for CRT control signals, logical display screen size and display period, blink timing and so on. The names of some important registers are:

- Horizontal Sync Register HSR
- Horizontal Display Register HDR
- Vertical Sync Register VSR
- Vertical Display Register VDR
- Split Screen Width Register SSW

The Display Control RAM DCR 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. The names of some important registers are:

- Memory Width Registers MWR0...MWR3
- Start AddressRegisters SAR0...SAR3
- Zoom Factor Register ZFR

1.3 FIFO accessible registers

For high performance drawing, key drawing processor registers are coupled to the CPU via the ACRTC's separate 16-byte read and write FIFO's.

ACRTC commands are sent from the CPU 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 DPR define detailed parameters of the drawing process, such as color control, area control and pattern RAM pointers. The DPR's are acessed using the ACRTC's Read Parameter Register (RPR) and Write Parameter Register (WPR) register access commands.

2 Frame-buffer interface of the ACRTC

2.1 Introduction

The ACRTC has a 16-bit wide data interface to the frame buffer lines (MAD0...MAD15). These 16 bits can be defined to be from 16 pixel in one plane to 16 planes of one pixel. On PIG1/68k all even MAD-lines are connected to the data-port of the frame buffer. All odd lines are used for the frame-buffer extension in the PIG2/68k-upgrade.

On PIG1/68k the frame buffer size is one MegaByte, which can be configured as one bitplane with eight MegaPixel, as two planes with four MegaPixels each or as four planes with two MegaPixels each. Please refer to part B of this manual for correct adaption, concerning the hardware-video-port.

According to the video-read-out configuration the ACRTC must be programmed with the correct Graphic Bit Mode GBM. It is important to have in mind, that the complete hardware is designed for generating 8 bitplanes with one ACRTC. If you use PIG1/68k alone, the effective GBM must be twice the desired number of bitplanes.

One of the design goals of PIG1/68k was to support the high drawing speed of the ACRTC by a suitable frame buffer architecture. Modern video RAM-chips have been used to reduce the need of time for display refresh to an absolute minimum: Only one cycle per raster scan line is necessary for that task. It is sufficient to program the ACRTC with a horizontal display width of one, to load 4096 bits (physical pixels) in the RAM's on-chip shift registers. In the four-bitplane-mode 1024 logical pixels are provided for one raster scan line, which is the maximum in that mode. In the two-bitplane-mode 2048 logical pixels are provided and in the one-bitplane-mode 4096 pixels per raster scan line are possible.

The dual-port-RAM architecture brings a very high throughput (factor 4 to 6 in comparison to conventional frame-buffer-concepts), but there are some restrictions, which should be mentioned here:

- The window (fourth screen) has always the same width as the other screens
- The horizontal pan feature (crawl) works always within one module of logical pixels
- The horizontal memory width must always be defined modulo the number of logical pixels
- The horizontal memory width has to be a power of two (512, 1024, 2048..)

Some of these restrictions can be compensated by using the powerful instruction set of the ACRTC efficiently. The bit-block-operations should be mentioned in this context. Because of the modern frame-buffer architecture they are working and thus can be used extensively without additional host-interaction.

2.2 Colors and Bitplanes

Controlling of the color mode is done with the Drawing Parameter Registers and within that set especially with the color registers COL0 and COL1. Please notice, that always all bits must be set according to the desired color. Please refer to table C.2.1 for details!

Tab C.2.1: Color control for PIG1/68k

Color	Control Byte	Register-value GBM=3	GBM=2	GBM=1
BLACK	$00	$0000	$0000	$0000
GREEN	$01	$0101	$1111	$5555
RED	$04	$0404	$4444	---
BLUE	$10	$1010	---	---
SCRATCH	$40	$4040	---	---
YELLOW	$05	$0505	$5555	---
CYAN	$11	$1111	---	---
MAGENTA	$14	$1414	---	---
WHITE	$15	$1515	---	---

Note:
- Color..... means the effect on the monitor when connected as described in chapter B!
- Register-value...means the effective value to be written into the appropriate registers with different Grafic Bit Modes GBM
- ---.....denotes impossible configurations

3 Examples to set up the ACRTC

In table C.3.1 you will find some set-up's for different monitors. It is important to have in mind, that the horizontal CRT sync timing is given in units of memory cycles. The length of one cycle is the dotrate, divided by 32. The length of the vertical parameters are given in units of raster scan lines. Please notice, like mentioned in the ACRTC data sheet, that some of the parameters must be subtracted by one before written into the ACRTC' registers. Table C.3.1 shows adjusted parameters.

Please note some general hints:

- The Grafic Address Increment parameter GAI should always be set to 2.
- The board should always be initialized as slave, even in standalone mode.
- The grafic bit mode GBM must always be twice the effective number of bitplanes, if no frame-buffer extension is used.
- The display start addresses must be adjusted by several so-called kluge-factors, which must be subtracted from the desired display start addresses. These factors depend on the ACRTC's mask-version and therefore can change. The given examples are evaluated with a S-mask.
- The horizontal display width HDW should always be set to one. The true parameter must be set in the ATR (Attribute Control) register.

Tab C.3.1: Set-up-examples for PIG1/68k (4 bitplane-mode)

Parameter Notes in ()	Monitor-Type 0	 1	 2	Name	ACRTC Reg
Hor Sync Width (1;2)	2	2	3	HSW	HSR
Hor Cycle (1;2)	27	45	41	HC	HSR
Hor Disp Width (1;2;4)	1	1	1	HDW	HDR
Eff Hor Disp Width (1;5)	19	31	31	-	ATR
Hor Disp Start (1;2)	2	6	3	HDS	HDR
Ver Sync Width (1;2)	3	3	10	VSW	VDR
Ver Cycle (1;2)	625	625	850	VC	VSR
Ver Disp Width (1;2)	512	512	785	SPx	SSW
Ver Disp Start (1;2)	67	67	40	VDS	VDR
Hor Plane Width (1;2)	4096	1024	1024	MW	MWR
Graphic Bit Mode (1;2)	3	3	3	GBM	CCR
GAI Mode (1;2)	2	2	2	GAI	OMR
Operation Mode (1;3)	410B	4108	4108	-	OMR
Kluge Factor: (2;6)					
Screen 0 (Upper)	4	4	4		SAR0
Screen 1 (Base)	0	4	4		SAR1
Screen 2 (Lower)	4	4	4		SAR2
Screen 3 (Window)	0	0	0		SAR3
Dotclock (7)	14	48	55		
Horiz. frequency in kHz	15.6	32.0	42.0		
Field frequency (8)	50	50	50		

Notes:
1...Parameter given not register-value
2...Decimal notation
3...Hexadecimal notation
4...Only one cycle because of Dual-port-RAM
5...Please refer to chapter C.4
6...To be subtracted from desired display start address
7...The dotclock is given in MHz with U601
8...In Hz; monitor-type 0 with interlaced fields

4 The Horizontal Display Width

As mentioned earlier, the effective horizontal display width (HDW) is programmed with register ATR. To be compatible with the ACRTC's normal nomenclature, it is recommended to calculate this parameter like to be written into register HDR.

Example:
If you want to have a horizontal display width of 1024 pixels on the screen with the four-bitplane-mode (GBM=3) the equivalent parameter HDW is 31 (32 minus 1).

The lower byte of register HDR should be set to one and the effective horizontal display width must be transformed in the following way:

First step: Complement HDW
Second step: Apply AND-mask of hex 3E
Third step: Write this value into register ATR

5 Crawl

Crawling is defined as repositioning the screen on the display memory in horizontal direction. In general always ACRTC's registers SARx are used for panning. If you want to use the horizontal pixel-by-pixel pan-feature the parameter Start Dot Address SDA within the SAR-registers are used. Those four bits define the offset of the screen position within that module which can be reached with the used grafic bit mode GBM. Please refer to table C.5.1!

Tab C.5.1: Significance of SAR-bits

GBM	SAR-bits D11	D10	D09	D08
3	x	x	x	1
2	x	x	2	1
1	x	4	2	1

Note:
GBM.....means grafic bit mode, which corresponds to the hardware-adapted number of bitplanes
x.......not used - should be set to zero
1,2,4...binary significance of Start Dot Address

6 Zoom

Programming the zoom-factor is very easy. The ACRTC has a special Zoom Factor Register ZFR. Four bits are used for the horizontal and another four bits are used for the vertical zoom factor VZF. The vertical zoom factor is used only within the ACRTC to modify vertical display refresh addressing. All 16 factors can be used. The horizontal zoom factor HZF is used by the ACRTC and external hardware, which supports only factors of 1,2 and 4. That is why only these factors should be used for horizontal zooming. Please refer to table C.6.1!

Tab C.6.1: Allowed horizontal zoom factors

Zoom factor	ZFR-Bits for HZF D15	D14	D13	D12
1	0	0	0	0
2	0	0	0	1
4	0	0	1	1

Note:
ZFR....means Zoom Factor Register
HZF....means Horizontal Zoom Factor
Zoom-factor 1 stands for unzoomed display

7 Blinking

Blinking is controlled via shapes generated in the scratch bitplane. Blinking must be hardware-enabled with Jumper J1003 as described in part B of this manual. With ACRTC's register BCR (Blink Control Register) the blinking rate can be modified and software-enabled/disabled. Only the BLINK2-feature is supported by the PIG-hardware.

8 Hardware status register

The hardware status register is an extra byte-wide register. It can be read and written. For address and bus-cycle please refer to chapter B.3!

It is intended to be used normally as ID/vector-register for the VMEbus interrupt-acknowledge cycle. But it can also be used for establishing semaphores and so on.

Please refer to PigPac manual for used and/or reserved bits.

ELTEC Elektronik GmbH

E L T E C 6 8 K - S Y S T E M

DOCUMENTATION

PIG1/68k

PART D

DO.68 01749

Preface

This part of the documentation is dedicated to all users, who want to deal with the hardware in detail, as in field service, for instance.
The hardware is designed modular. The several blocks are:

- VMEbus interface
  --> wiring diagram 1
- Control logic for VMEbus and local bus
  --> wiring diagram 2
- Status/vector register
  --> wiring diagram 3
- Advanced Cathode Ray Tube Controller ACRTC
  --> wiring diagram 4
- Frame-buffer interface
  --> wiring diagram 5
- Central timing logic
  --> wiring diagram 6
- Two-port-RAM timing generator
  --> wiring diagram 7
- Video shift-register
  --> wiring diagram 8
- Frame-buffer
  --> wiring diagram 9
- Monitor interface
  --> wiring diagram 10

1 VMEbus interface

Circuit diagram 1 shows the VMEbus interface.

The data bus is coupled via transceivers U101 and U102. The internal data bus is called ID00 to ID15. U103 and U104 generate the line /BADO (base address decoder out), if the base address and the appropriate address modifiers are decoded.

All internal VMEbus lines have the prefix "I" to show that they are of the same function, but internal.

Hex switches S101, S102 and S103 control the base address. Additionally S103 controls the address modifier AM2 and AM1. J101 can be used to disable decoding of AM2.

2 Control logic for VMEbus and local bus

U201 latches all control lines, which are necessary for internal use. U202 buffers control lines /DS0 and /SYSRESET. Circuit diagram 2 shows details of this module. PAL U204 decodes the devices and handles the VMEbus logic, including the VMEbus interrupt control logic. U203 decodes a valid interrupt acknowledge cycle (line /VIACKCY). U202 buffers the internal IRQ-line. With jumper J201 the desired interrupt level and the appropriate acknowledge level are selected.

All bus grant lines are connected in the printed circuit layout, so there is no need to jumper them on the backplane.

With J202 and J203 the VMEbus transceivers and buffers can be disabled for external access to the local bus.

3 Status/vector register

Circuit diagram 3 shows the status/vector register.

U301 is a 8 bit wide register, called PIG.STAT. The register can be programmed to contain the interrupt vector. Otherwise it can be used as a general purpose register to store semaphores etc.

4 Advanced Cathode Ray Tube Controller (ACRTC)

Circuit diagram 4 shows the ACRTC-module.

U401 is the central device of the board. It is decoded by the line /ENACRTC. NAND-gate U402 is used for external synchronisation of the board.

J401 is used for master/slave operation.

5 Frame-buffer interface

Circuit diagram 5 shows details of this module.

U501 is a buffer, which connects the multiplexed data/address-bus of the ACRTC with the databus of the frame-buffer. U502 is a registered multiplexer to address the frame-buffer. U503 and U504 are used to store some of the ACRTC's attribute-bits and the video-address-lines VA17-VA19. These lines are used to select the different RAM-banks. U505 stores the least significant video-address-line VA0, which must be done, because of the ACRTC's multiplexed memory bus.

Line SYNCR is used to synchronize other PIG-board(s).

6 Central timing logic

Circuit diagram 6 shows details of this module.

U601 is the central dot clock oscillator, which can/must be adapted to the desired monitor parameter set. Jumper J601 defines whether the dotclock comes from the on-board oscillator or from another PIG-board. U606 is clocked by the dotclock to generate other timing control lines. U608 buffers these lines to get a defined and stable timing relationship. The line 2CLK clocks the ACRTC and line CLK8 is used to control the RAM-timing. J602 and J603 must be set properly according to the desired bitplane mode.

U607 and U609 have nearly the same function as U606 and U608, but are clocked via a selectable frequency to control the horizontal video zoom capability.

U604 and U605 form a display-enable-signal, which is independent of the ACRTC. This must be done, because the ACRTC does not support the modern two-port-video-RAM's properly. The desired horizontal display width must be programmed with attribute lines MAD1 to MAD5. The ACRTC's horizontal width parameter determines only, how often a video refresh cycle will be done per raster scan line. Normally one video refresh cycle is necessary.

Jumpers J601 and J604 are used in master/slave-configurations.

7 Two-port-RAM timing generator

Circuit diagram 7 shows details of this module.

Shift register U702 generates basic timing signals, which are used in PAL U701 to form the RAM-control-lines.

8 Video shift-registers

For details see circuit diagram 8!

U802 and U803 store a group of 16 pixels, which are read out of the frame buffer at a time. Counter U801 generates several load-pulses for the shift registers. The chip is programmed with lines ATTSCx, which are the ACRTC's start dot address. With this circuitry it is possible to program the phase relationship between the line RAMCK, which serve as a clock to buffer the video data and the lines LDHI and LDLO, which load two shift register modules.

The first set of shift registers U806 to U809 is clocked by line CLKLO. The pixel-rate is multiplied by four in this stage. The next stage is built by shift register U810, which has different functions: In the four-bitplane-mode it serves only as a register; in other modes it increases the pixel-rate once more. It is important to notice, that jumpers J602, J603, J801, J802, J803 and J804 must be set properly according to the desired numer of bitplanes.

9 Frame-buffer

For details see circuit diagram 9!

U906 to U937 are dual-port-video-RAM's, containing the bitmap, generated by the ACRTC. U901 buffers the multiplexed RAM-address-lines. The RAM-chips are organized to 8 banks of 4 chips each. U902 controls the serial port of the several banks. U903 and U904 activate two out of 16 different /RASxx-lines to select two RAM-chips at a time for the ACRTC's random access to the frame-buffer. On PIG1/68k all odd lines of the ACRTC's memory port are connected to the frame-buffer, having all even lines reserved for the eight bit version of the board PIG2/68k.

In case of a RAM-refresh-cycle during the horizontal sync period, all chips will be activated at the same time. Refresh-control is achieved by line RASPOL.

10 Monitor interface

For details see circuit diagram 10!

Video-lines SDx are 'ANDed' with the synchronized display enable signal DISPEN, which is available at connector X10 as MBLK and at connector X11 for the local display memory extension. The video-lines are clocked the first time and then ANDed with the blink-enable line BLINKEN. After the second register stage they are driven with buffer U602 and connected to the resistor-network for optional greyscale operation.

The monitor sync signals are synchronized with U1005. With jumpers J1101 and J1102 the polarity can be selected. Another two buffers of U602 are used to drive both lines, which are available at connectors X3 (HSY) and X4 (VSY).

ELTEC Elektronik GmbH

E L T E C 6 8 K - S Y S T E M

DOCUMENTATION

PIG1/68k

APPENDIX

DO.68 01749

Appendix A: Address modifier VMEbus

hex Code	Address modifier 5	4	3	2	1	0	Access	Note
3F	H	H	H	H	H	H	Standard Supervisory Ascending	1
3E	H	H	H	H	H	L	Standard Supervisory Program	1
3D	H	H	H	H	L	H	Standard Supervisory Data	1
3C	H	H	H	H	L	L	Undefined	2
3B	H	H	H	L	H	H	Standard Non-Privileged Ascend	1
3A	H	H	H	L	H	L	Standard Non-Provileged Program	1
39	H	H	H	L	L	H	Standard Non-Privileged Data	1
38	H	H	H	L	L	L	Undefined	2
30-37	H	H	L	x	x	x	Undefined	2
2F	H	L	H	H	H	H	Undefined	2
2E	H	L	H	H	H	L	Undefined	2
2D	H	L	H	H	L	H	Short Supervisory I/O	1
2C	H	L	H	H	L	L	Undefined	2
2B	H	L	H	L	H	H	Undefined	2
2A	H	L	H	L	H	L	Undefined	2
29	H	L	H	L	L	H	Short Non-Privileged I/O	1
28	H	L	H	L	L	L	Undefined	2
20-27	H	L	L	x	x	x	Undefined	2
10-1F	L	H	x	x	x	x	Undefined	3
0F	L	L	H	H	H	H	Extended Supervisory Ascending	1
0E	L	L	H	H	H	L	Extended Supervisory Program	1
0D	L	L	H	H	L	H	Extended Supervisory Data	1
0C	L	L	H	H	L	L	Undefined	2
0B	L	L	H	L	H	H	Extended Non-Privileged Ascend	1
0A	L	L	H	L	H	L	Extended Non-Privileged Program	1
09	L	L	H	L	L	H	Extended Non-Privileged Data	1
08	L	L	H	L	L	L	Undefined	2
00-07	L	L	L	x	x	x	Undefined	2

Notes:
1....Defined by VMEbus Specification
2....Definition reserved
3....Defined by user

Appendix B: Jumpers, switches

Id No	Function	Refer to chapter
S101	VMEbus base address	B.3.1; D.1
S102	VMEbus base address	B.3.1; D.1
S103	VMEbus address modifier	B.3.2; D.1
J101	Decoding of AM2	B.3.2; D.1
J201	VMEbus interrupt level	B.3.4; D.2
J202	External access	B.7; D.2
J203	External access	B.7; D.2
J401	Master/slave-operation	B.6; D.4
J501	Operation of PIG2-config	B.5; D.5
J601	Master/slave-operation	B.6; D.6
J602	Selects no of bitplanes	B.5; D.6
J603	Selects no of bitplanes	B.5; D.6
J604	Master/slave-operation	B.6; D.6
J801	Selects no of bitplanes	B.5; D.8
J802	Selects no of bitplanes	B.5; D.8
J803	Selects no of bitplanes	B.5; D.8
J804	Selects no of bitplanes	B.5; D.8
J1001	Jumper for V-sync	B.4.1; D.10
J1002	Jumper for H-sync	B.4.1; D.10
J1003	Enables blinking	B.4.3; D.10

Appendix C: Connectors

Id No	Function	Type
X1	VMEbus-connector P1	96-pin DIN41612
X2	not used	
X3	H-sync output	Mini BNC
X4	V-sync output	Mini BNC
X5	Green output	Mini BNC
X6	Red output	Mini BNC
X7	Blue output	Mini BNC
X8	Scratch output	Mini BNC
X9	Dotclock output/input	Mini BNC
X10	Blanking output	Mini BNC
X11	Local frame buffer bus	64-pin DIN41612
X12	Local bus	2*25 pin female
X13	External synchronisation	Mini BNC
X14	Prescaler reset	Mini BNC

Connector X1 (VMEbus)

Signals in brackets have no connection to the board.

Pin No	Row A	Row B	Row C
1	D00	(/BBSY)	D08
2	D01	(/BCLR)	D09
3	D02	(/ACFAIL)	D10
4	D03	/BG0IN	D11
5	D04	/BG0OUT	D12
6	D05	/BG1IN	D13
7	D06	/BG1OUT	D14
8	D07	/BG2IN	D15
9	GND	/BG2OUT	GND
10	(SYSCLK)	/BG3IN	(/SYSFAIL)
11	GND	/BG3OUT	/BERR
12	(/DS1)	(/BR0)	/SYSRESET
13	/DS0	(/BR1)	/LWORD
14	/WRITE	(/BR2)	AM5
15	GND	(/BR3)	(A23)
16	/DTACK	AM0	(A22)
17	GND	AM1	(A21)
18	(/AS)	AM2	(A20)
19	GND	AM3	(A19)
20	/IACK	GND	(A18)
21	/IACKIN	(SERCLK)	(A17)
22	/IACKOUT	(SERDAT)	(A16)
23	AM4	GND	A15
24	A07	/IRQ7	A14
25	A06	/IRQ6	A13
26	A05	/IRQ5	A12
27	A04	/IRQ4	A11
28	A03	/IRQ3	A10
29	A02	/IRQ2	A09
30	A01	/IRQ1	A08
31	(-12V)	(+5VSTDBY)	(+12V)
32	+5V	+5V	+5V

Connector X11

This connector is used for PIG2/68k-configuration.

Pin No	Row A	Row B	Row C
1	DOTCLKP11	+5V	GND
2	GND	GND	MA0
3	MA1		MA2
4	MA3		MA4
5	MA5		GND
6	MA7		GND
7	VA18		-12V
8	VA19		+5V
9	VA0		+12V
10	RAMCK		
11	/OERAM		/OEBUF
12	/WERAM	GND	MRD
13	CLKLO	+5V	
14	/RAS1		
15	LDHI		
16	RASPOL		
17	/CAS		
18	/HSYNC		
19	LDLO		
20	DISPEN		
21	BLINKEN		
22	MAD1	GND	
23	MAD3		
24	MAD5		
25	MAD7		
26	MAD9		
27	MAD11		
28	MAD13		
29	MAD15		MAD18
30	MAD17		+5V
31	MAD19	GND	MA6
32	/MAD19	+5V	VA17

Connector X12

This connector is dedicated for external access to local bus. Lines in () are not used.

Pin No odd	Signal	Pin No even	Signal
1	reserved	2	reserved
3	RCTRL0	4	/GDCDONE
5	/GDCDREQ	6	/GDCDACK
7	/IDS0	8	/IWRITE
9	/IRES	10	(/IDS1)
11	(SYSCLK)	12	(A19)
13	(A18)	14	(A17)
15	(A16)	16	(A15)
17	(IA14)	18	(IA13)
19	(IA12)	20	(IA11)
21	(IA10)	22	(IA09)
23	(IA08)	24	(IA07)
25	(IA06)	26	(IA05)
27	IA04	28	IA03
29	IA02	30	IA01
31	/DBEN	32	/BADO
33	/ERR	34	/RDY
35	ID15	36	ID14
37	ID13	38	ID12
39	ID11	40	ID10
41	ID09	42	ID08
43	ID07	44	ID06
45	ID05	46	ID04
47	ID03	48	ID02
49	ID01	50	ID00

Appendix D: Sample program

```
*******************************************************************************
* Demonstration for PIG1 / HD 63484                                           *
*                                                                             *
* This program shows the initialization of the ACTRC and draws                *
*   on the screen one rectangle and some lines                                *
*                                                                             *
* (C) Eltec Elektronik GmbH Mainz                                             *
*                                                                             *
* rb 220586                                                                   *
*******************************************************************************

*         OS9-specific declarations

        use <oskdefs.d>
Stk     equ     100                     * some default stack size
        psect   pigdemo,(Prgrm<<8)!Objct,(ReEnt<<8)!1,0,Stk,pigdemo

*         end of OS9-specific declarations

acrtc   equ     $ff8d00                 * Hardware Adress of Master Status
memtot  equ     $100000                 * Total memory is 1 megawords
GBM     equ     3                       * log (2) of Bit/pixel for PIG2;
*                                         this can be 1, 2 or 3.
DN      equ     1<<30                   * Base screen is specified
linpix  equ     1024                    * Pixel/line
memwid  equ     linpix>>(4-GBM)         * 512 Words/line (16 bit;
*                                         8 bit with PIG1)
linnum  equ     memtot/memwid           * total number of lines
startad equ     memtot-(785*memwid)-4   * Start Address (upper left)
*                                         4 is 'Kluge factor'
drawpoi equ     DN+((memtot-memwid)<<4) * Drawing Pointer for (0,0)
*                                         on Base Screen

GAI     equ     2
*                                       Operation Mode Register OMR
OMR     equ     $4108+(GAI<<4)          * Start, Slave, GAI=2,Dual Access
*                                       Display Control Register DCR
DCR     equ     $4020                   * Enable Base Screen;
*                                         ATR=$20 i.e. HDW=31
```

```
*****************************************************************
*
*         Table for Display Control Ram ($c0 .. $de)
*
dispcon dc.w    0,memwid,startad>>16,(startad&$ffff) * upper
        dc.w    0,memwid,startad>>16,(startad&$ffff) * base
        dc.w    0,memwid,startad>>16,(startad&$ffff) * lower
        dc.w    0,memwid,startad>>16,(startad&$ffff) * window
        dc.w    0,0,0,0,0,0                          * Cursor, zoom

*****************************************************************
*
*        Drawing Parameter register contents
*
parreg: dc.w    $0000,$ffff,$ffff,$ffff,$ffff * Color Registers
        dc.w    $0000,$0000,$00f0       * Pattern Ram Control Registers
        dc.w    0,0,1023,2047           * Area boundaries

*****************************************************************

pigdemo: movem.l d0-d7/a0-a6,-(a7)
        move.l  #acrtc,a0               * Pointer to status reg. of ACRTC
        lea.l   2(a0),a1                * Pointer to data register
        move    #$2,(a0)
        move    #$8000,(a1)             * Abort (Reset ACRTC)
        move    #$4,(a0)
        move    #$0,(a1)                * slave mode, display stop
        bsr     intimct                 * Init Timing Control Ram

        bsr     clrscrn

        move    #300,d2                 * draw rectangle at <d2>,<d3>
        move    #200,d3
        bsr     rect

        bsr     drawtst                 * draw some lines
        movem.l (a7)+,d0-d7/a0-a6
        os9     F$Exit                  * OS9-specific finish
```

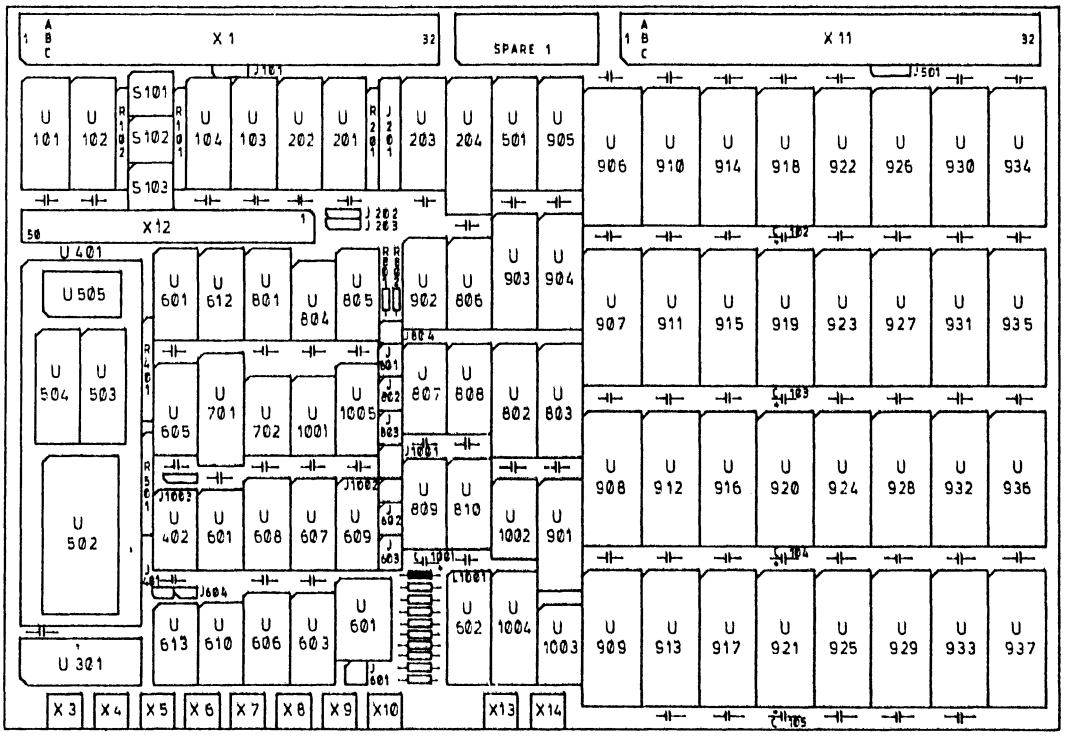
```
*
*       set RW POINTER to lower left
*
setrwp: move    #$080c,d1               * Write Pattern Ram
        bsr     wrfifo
        move.l  #drawpoi,d1
        swap    d1                      * write lower half register
        bsr     wrfifo
        move    #$080d,d1
        bsr     wrfifo
        swap    d1                      * upper half
        bsr     wrfifo
        rts
*
*        Move graphics cursor to x=<d2>; y=<d3>
*
amove:  move    #$8000,d1               * AMOVE command
        bsr     wrfifo
        move    d2,d1                   * X
        bsr     wrfifo
        move    d3,d1                   * Y
        bsr     wrfifo
        rts
*
*        write one byte into the fifo
*
wrfifo: move    #0,(a0)                 * status register is addressed
wrfif1: move    (a0),d6                 * read status register
        and     #2,d6                   * Write fifo ready?
        beq.s   wrfif1                  * Wait for it
        move    d1,(a1)                 * write data to fifo
        rts
```

```
*
*       Initialize Timing Control Ram;
*       This is for monitor type 2 (see PIG documentation, part C)
*
intimct: move   #$82,(a0)               * Timing Control RAM init.
         move   #(41<<8)+3,(a1)         * HorCycle, HorSyncWid  82
         move   #(3<<8)+1,(a1)          * HorDspStrt,HorDispWid 84
         move   #850,(a1)               * VertCycle             86
         move   #(40<<8)+10,(a1)        * VerDspStrt,VerSyncWid 88
         move   #785,(a1)               * Base Screen Width     8a

         move   #($9c-$8c),d0           * Clear Regs 8c..9c
clr9c    move   #0,(a1)
         sub    #2,d0
         bpl.s  clr9c

         move   #$c0,(a0)               * Display Control RAM
         move   #22,d0                  * 22 regs total
         lea    dispcon(pc),a2
displop  move   (a2)+,(a1)
         sub    #1,d0
         bne.s  displop

         move   #$2,(a0)                * Init control regs
         move   #GBM<<8,(a1)            * CCR: remove Abort, set bit/pixel
         move   #$4,(a0)                * OMR
         move   #OMR,(a1)               * Start, Slave, GAI=2,Dual Access
         move   #$6,(a0)                * DCR
         move   #DCR,(a1)
*
*        Initialize Origin to lower left
*
         move   #$0400,d1               * ORG command
         bsr    wrfifo
         move.l #drawpoi,d1             * (0,0) is lower left corner
         swap   d1
         bsr    wrfifo                  * transfer longword in 2 parts
         swap   d1
         bsr    wrfifo
         rts
```

```
*
*         Initialize parameter registers
*
         clr     d2                      * Init par registers
         lea     parreg(pc),a2
inpalop: move    #$0800,d1               * WPR Command
         or      d2,d1                   * Mask in Reg No.
         bsr     wrfifo
         move    (a2)+,d1                * Load item from table
         bsr     wrfifo
         add     #1,d2
         cmp     #12,d2
         bne     inpalop

         move    #$1800,d1               * WPTN Write Pattern Ram
         bsr     wrfifo
         move    #16,d1
         bsr     wrfifo                  * fill all 16 words with $ffff
         move    #16,d2                  * d2 is counter
patlop:  move    #$ffff,d1
         bsr     wrfifo
         sub     #1,d2
         bne.s   patlop
         rts

clrscrn: bsr      setrwp
         move    #$5800,d1               * Clear Screen to dark
         bsr     wrfifo
         clr     d1                      * Data is 'dark'
         bsr     wrfifo
         move    #memwid,d1              * X extension of area to clear
         bsr     wrfifo
         move    #linnum,d1              * Y
         bsr     wrfifo
         rts
```

```
rect:   bsr     amove                   * draw rectangle at <d2>,<d3>
        move    #$9400,d1               * RRCT command
rect1:  bsr     wrfifo
        move    #100,d1                 * X extension
        bsr     wrfifo
        move    #150,d1                 * Y extension
        bsr     wrfifo
        rts
*
*        Draw lines from lower left to the upper edge
*
drawtst: move   #1800,d4                * x-coordinate on upper edge
linlop: clr     d2                      * start at 0,0
        clr     d3
        bsr     amove                   * set starting point
        move    #$8800,d1               * ALINE command
        bsr     wrfifo
        move    d4,d1                   * X at endpoint
        bsr     wrfifo
        move    #1800,d1                * Y is always at maximum
        bsr     wrfifo
        sub     #100,d4                 * decrement X
        bne.s   linlop
        rts

        ends                            * this is specific for OS9
```

Appendix E: Layout diagram PIG1/68k



Appendix F: Parts list

The character in the part number identifies the type:

B.....Battery
C.....Condensator
D.....Diode, single LED, rectifier
F.....Fuse
J.....Jumper
K.....Relais
L.....Inductivity, transformer
Q.....Transistor, triac, thyristor
R.....Resistor (fixed/variable), Network
S.....Switch
T.....Test point
U.....Integrated circuit, display, quartz
X.....Connector

Other abreviations:

Ta.....Tantalum capacitor
Cer.....Ceramic capacitor
MF.....Metal film

Part no	Description
X1	96-pin male connector DIN 41612, rows a,b,c
X2	
X3	Mini-BNC
X4	Mini-BNC
X5	Mini-BNC
X6	Mini-BNC
X7	Mini-BNC
X8	Mini-BNC
X9	Mini-BNC
X10	Mini-BNC
X11	96-pin female connector DIN 41612, rows a,b
X12	2*25 pins female
X13	Mini-BNC-Buchse abgewinkelt
C	Block cap. 47nF/50V; AVX

Part no	Description
U101	74LS645-1 on socket
U102	74LS645-1 on socket
U103	AM25LS2521
U104	AM25LS2521
R101	Resistornetwork 8*3k3
R102	Resistornetwork 8*3k3
C101	Ta 10uF/16V
C102	Ta 10uF/16V
C103	Ta 10uF/16V
C104	Ta 10uF/16V
C105	Ta 10uF/16V
S101	Hex switch
S102	Hex switch
S103	Hex switch
J101	Jumper 1*3 pin

Part no	Description
U201	74LS373
U202	74LS641-1 on socket
U203	AM25LS2521
U204	PAL20L10 on socket (PIG1*2.x)
R201	Resistornetwork 8*3k3
J201	Jumper 2*10 pin
J202	Jumper 1*3 pin
J203	Jumper 1*3 pin

Part no	Description
U301	74LS374

Part no	Description
U401	HD63484-8 (S-mask or newer) on socket
U402	74F00
R401	Resistornetwork 8*330
J401	Jumper 1*2 pin

Part no	Description
U501	74LS245
U502	74LS604
U503	74LS374
U504	74LS374
U505	74LS74
R501	Resistornetwork 8*3k3
J501	Jumper 1*3 pin

Part no	Description
U601	QG 55MHz on socket
U602	74F244
U603	74F352
U604	74LS163
U605	74LS163
U606	74F163
U607	74F163
U608	74F175
U609	74F175
U610	74F20
U611	74LS74
U612	74F74
U613	74F243
J601	Jumper 2*2 pin
J602	Jumper 2*3 pin
J603	Jumper 2*3 pin
J604	Jumper 1*2 pin

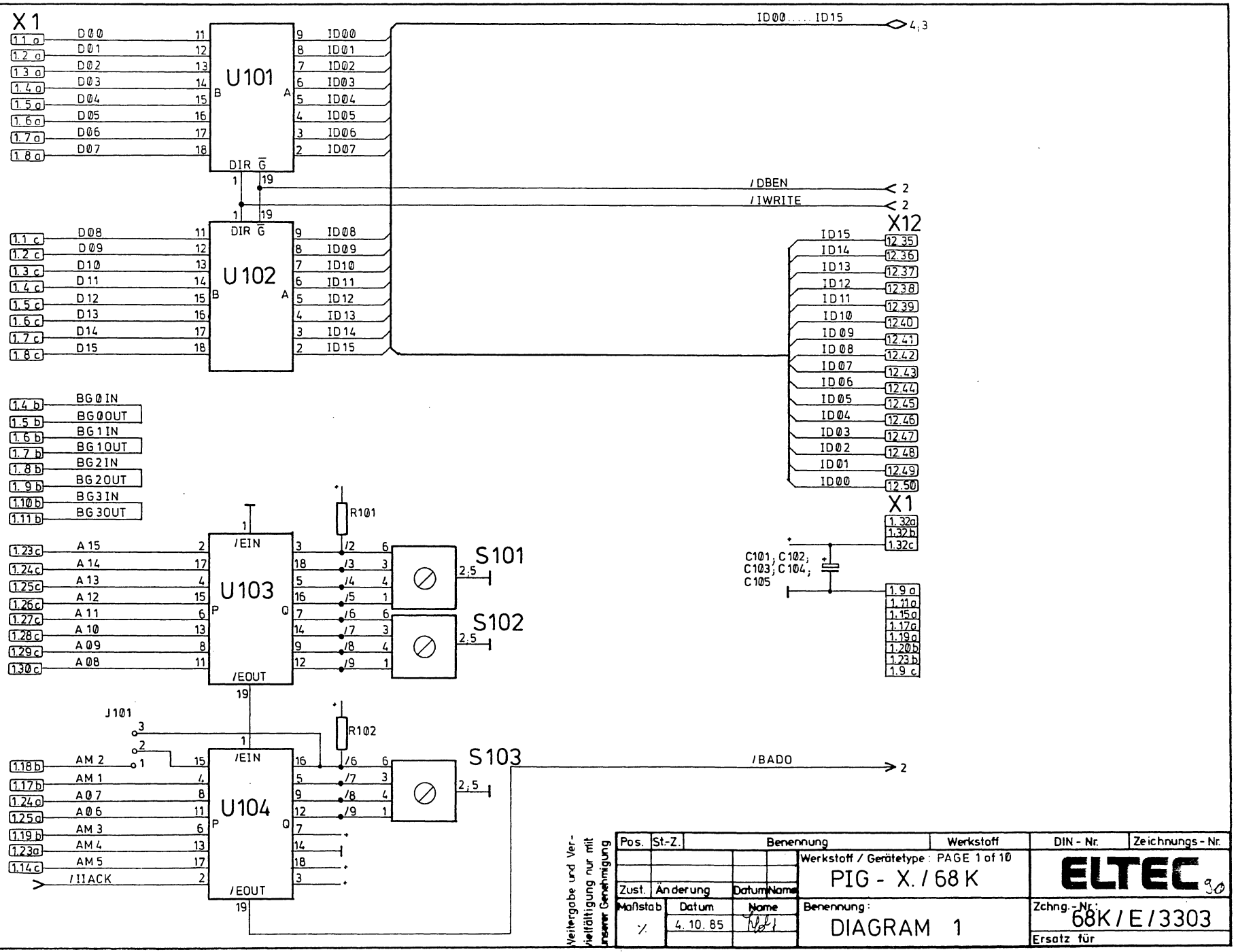
Part no	Description
U701	PAL16L8 on socket (PIG1*1.x)
U702	74F164

Part no	Description
U801	74F163
U802	74F374
U803	74F374
U804	74F00
U805	74F175
U806	74F194
U807	74F194
U808	74F194
U809	74F194
U810	74F194
J801	Jumper 2*3 pin
J802	Jumper 2*3 pin
J803	Jumper 2*3 pin
J804	Jumper 2*2 pin

Part no	Description
U901	AM2966
U902	74LS138
U903	74F538
U904	74F538
U905	AM2966
U906	uPD41264-15
U907	uPD41264-15
U908	uPD41264-15
U909	uPD41264-15
U910	uPD41264-15
U911	uPD41264-15
U912	uPD41264-15
U913	uPD41264-15
U914	uPD41264-15
U915	uPD41264-15
U916	uPD41264-15
U917	uPD41264-15
U918	uPD41264-15
U919	uPD41264-15
U920	uPD41264-15
U921	uPD41264-15
U922	uPD41264-15
U923	uPD41264-15
U924	uPD41264-15
U925	uPD41264-15
U926	uPD41264-15
U927	uPD41264-15
U928	uPD41264-15
U929	uPD41264-15
U930	uPD41264-15
U931	uPD41264-15
U932	uPD41264-15
U933	uPD41264-15
U934	uPD41264-15
U935	uPD41264-15
U936	uPD41264-15
U937	uPD41264-15

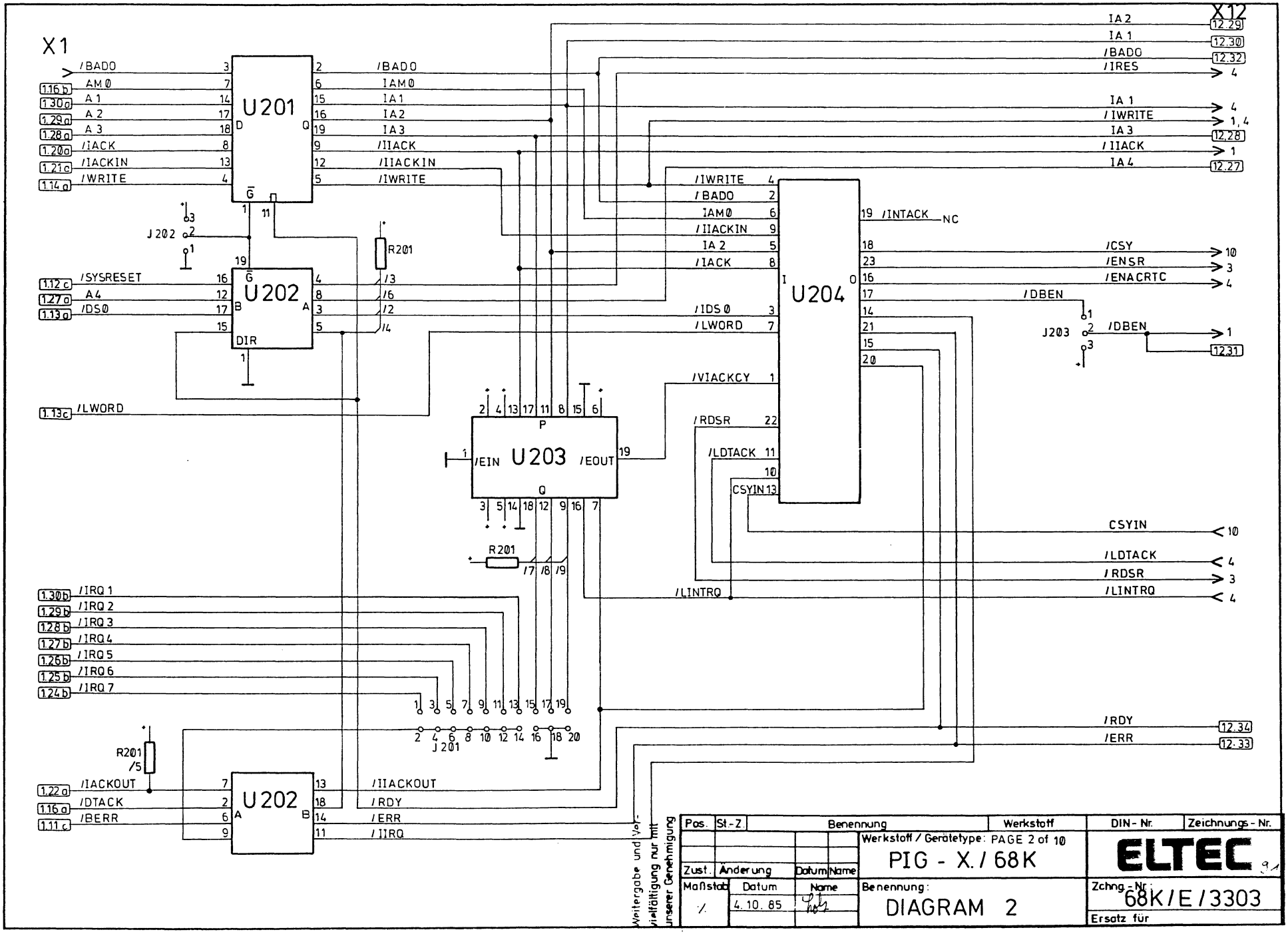
Part no	Description
U1001	74F74
U1002	74F08
U1003	74F08
U1004	74F374
U1005	74F175
R1001	0 Ohm on socket
R1002	0 Ohm on socket
R1003	0 Ohm on socket
R1004	0 Ohm on socket
R1005	no part with socket
R1006	no part with socket
R1007	no part with socket
R1008	no part with socket
R1009	no part with socket
L1001	100nH
C1001	Ta 33uF/16V
J1001	Jumper 2*2 pin
J1002	Jumper 2*3 pin
J1003	Jumper 1*3 pin

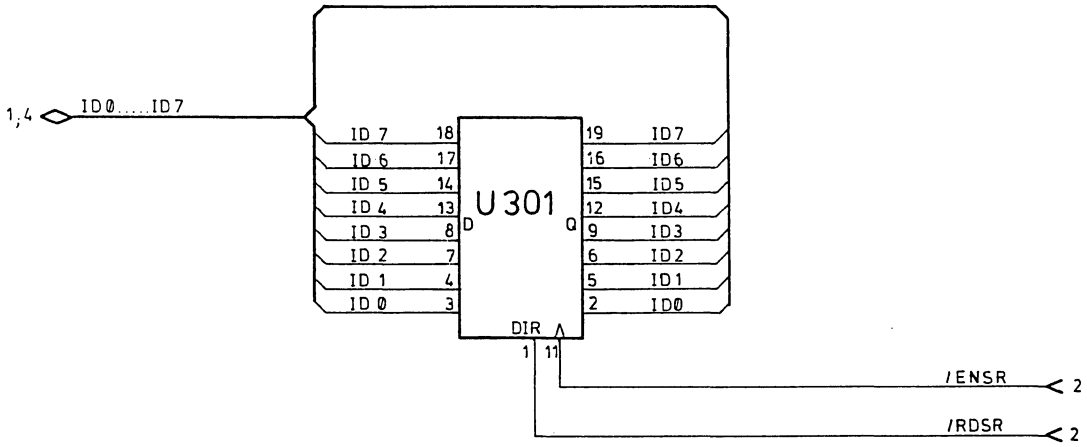
Appendix G: Circuit diagrams



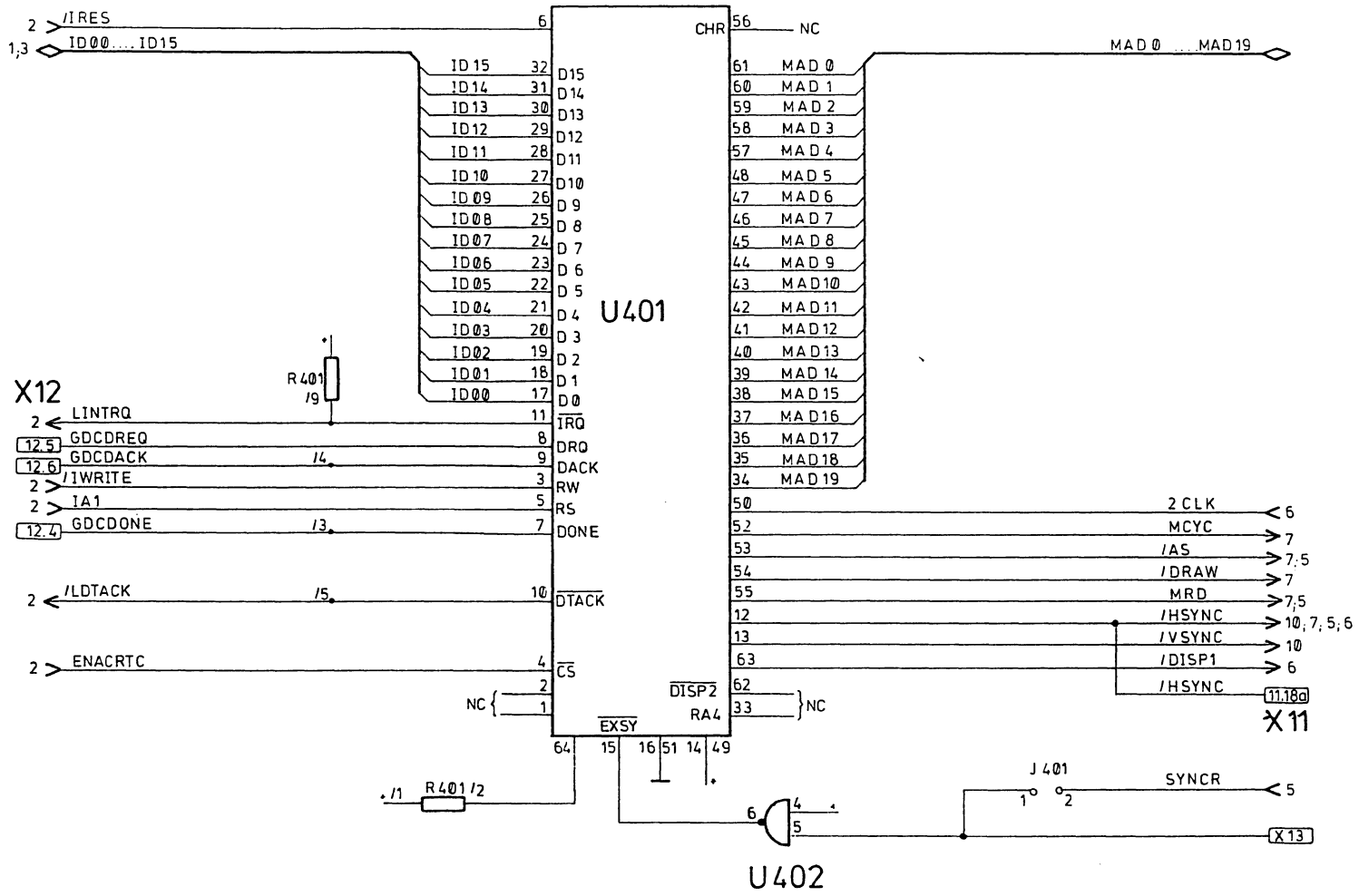
Pos.	St.-Z.	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype: PAGE 1 of 10 PIG - X. / 68 K		ELTEC 90	
Zust.	Änderung	Datum	Name		
Maßstab	Datum	Name	Benennung: DIAGRAM 1	Zchng.-Nr.: 68K / E / 3303	
%	4. 10. 85	Nöl		Ersatz für	

Weitergabe und Vervielfältigung nur mit ausdrücklicher Genehmigung


X1
X12
U201
U202
U203
U204
/BADO
AM0
A1
A2
A3
/IACK
/IACKIN
/WRITE
/SYSRESET
A4
/DS0
/LWORD
/IRQ1
/IRQ2
/IRQ3
/IRQ4
/IRQ5
/IRQ6
/IRQ7
/IACKOUT
/DTACK
/BERR
/IIACK
/IIACKIN
/IWRITE
IAM0
IA1
IA2
IA3
/IIACKOUT
/RDY
/ERR
/IIRQ
/IDS0
/VIACKCY
/RDSR
/LDTACK
CSYIN
/LINTRQ
/INTACK
/CSY
/ENSR
/ENACRTC
/DBEN
/IRES
IA4
/EIN
/EOUT
J201
J202
J203
R201
NC
Pos.
St-Z
Benennung
Werkstoff
DIN-Nr.
Zeichnungs-Nr.
Werkstoff / Gerätetype: PAGE 2 of 10
PIG - X./68K
ELTEC
Zust.
Änderung
Datum
Name
Maßstab
Datum
Name
4.10.85
Benennung:
DIAGRAM 2
Zchng-Nr.
68K/E/3303
Ersatz für
Weitergabe und Vervielfältigung nur mit unserer Genehmigung

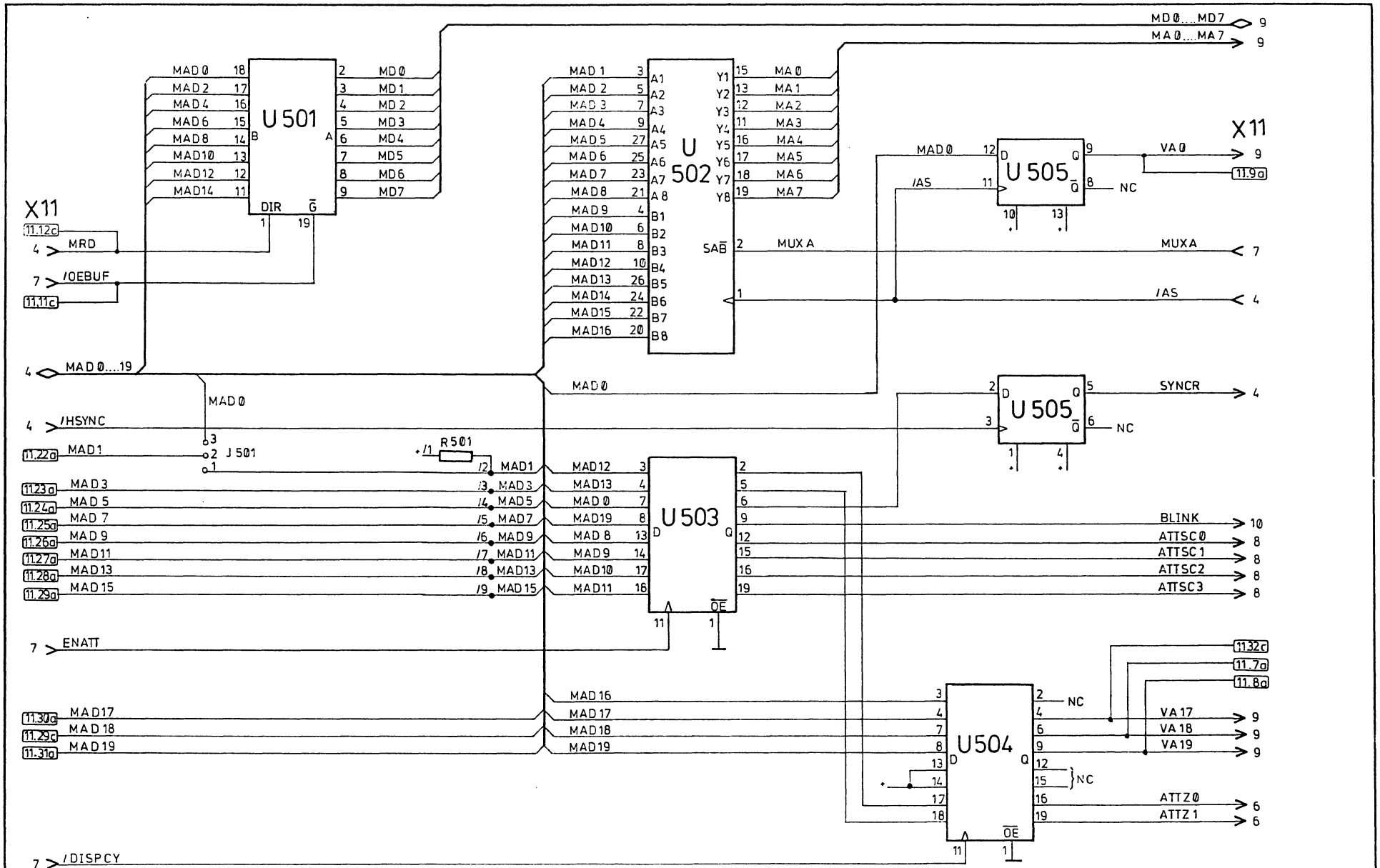


Pos.	St.-Z.	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype PAGE 3 of 10			
		PIG - X. / 68 K		ELTEC	
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
%	4. 10. 85		DIAGRAM 3	68K/E/3303	
				Ersatz für	



Pos.	St.-Z.	Benennung	Werkstoff	DIN-Nr.	Zeichnungs-Nr.
		Werkstoff / Geratetype: PAGE 4 of 10			
		PIG - X./68K			
Zust	Änderung	Datum	Name		
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
/	4.10.85		DIAGRAM 4	68K/E/3303	
				Ersatz für	

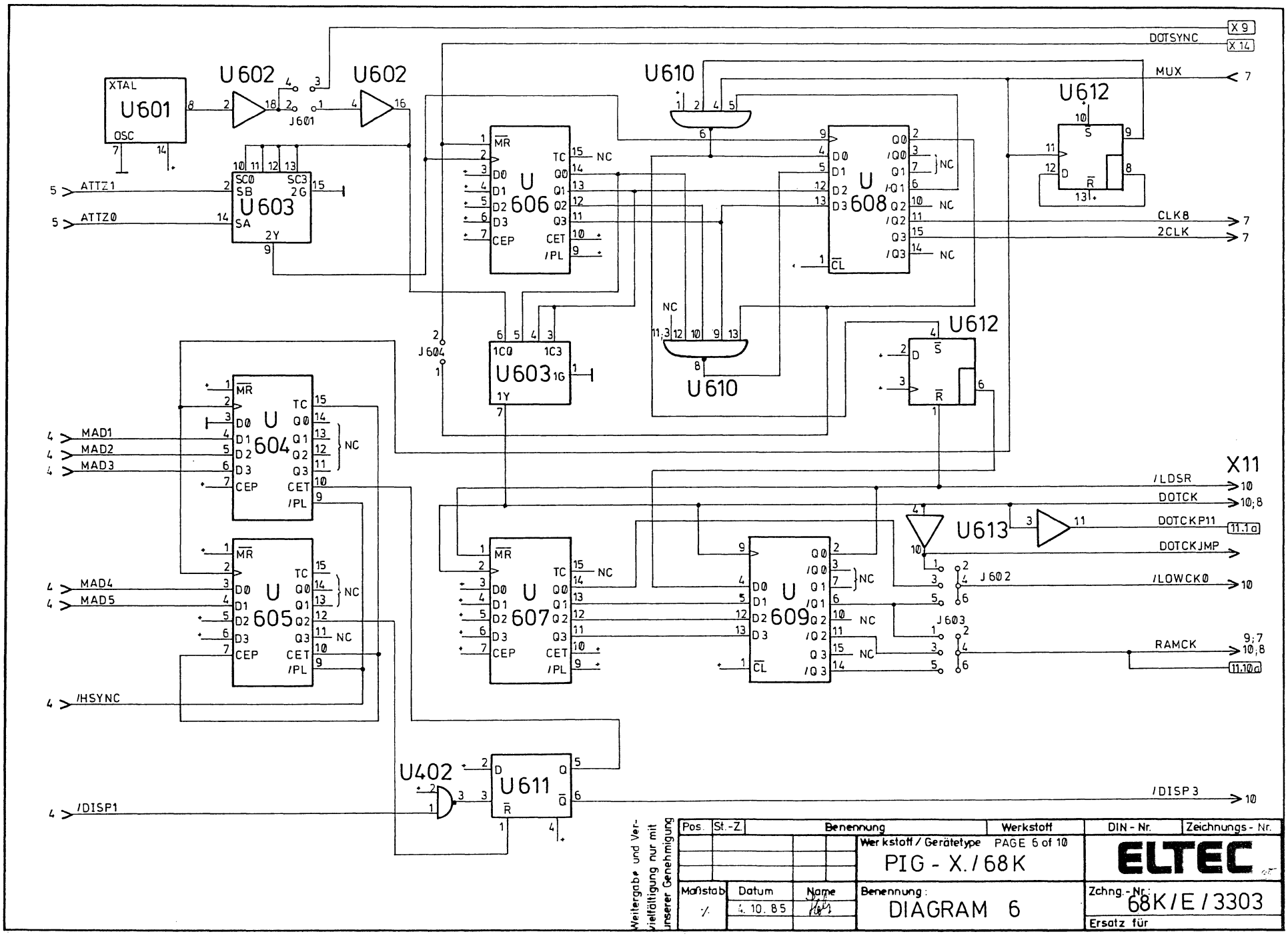
ELTEC



Pos	St.-Z.	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype PAGE 5 of 10			
		PIG - X./68K			ELTEC
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
%	4.10.85		DIAGRAM 5	68K/E/3303	
				Ersatz für	

Weitergabe und Vervielfältigung nur mit unserer Genehmigung

Bestell-Nr. 643432 90 95 0.0m

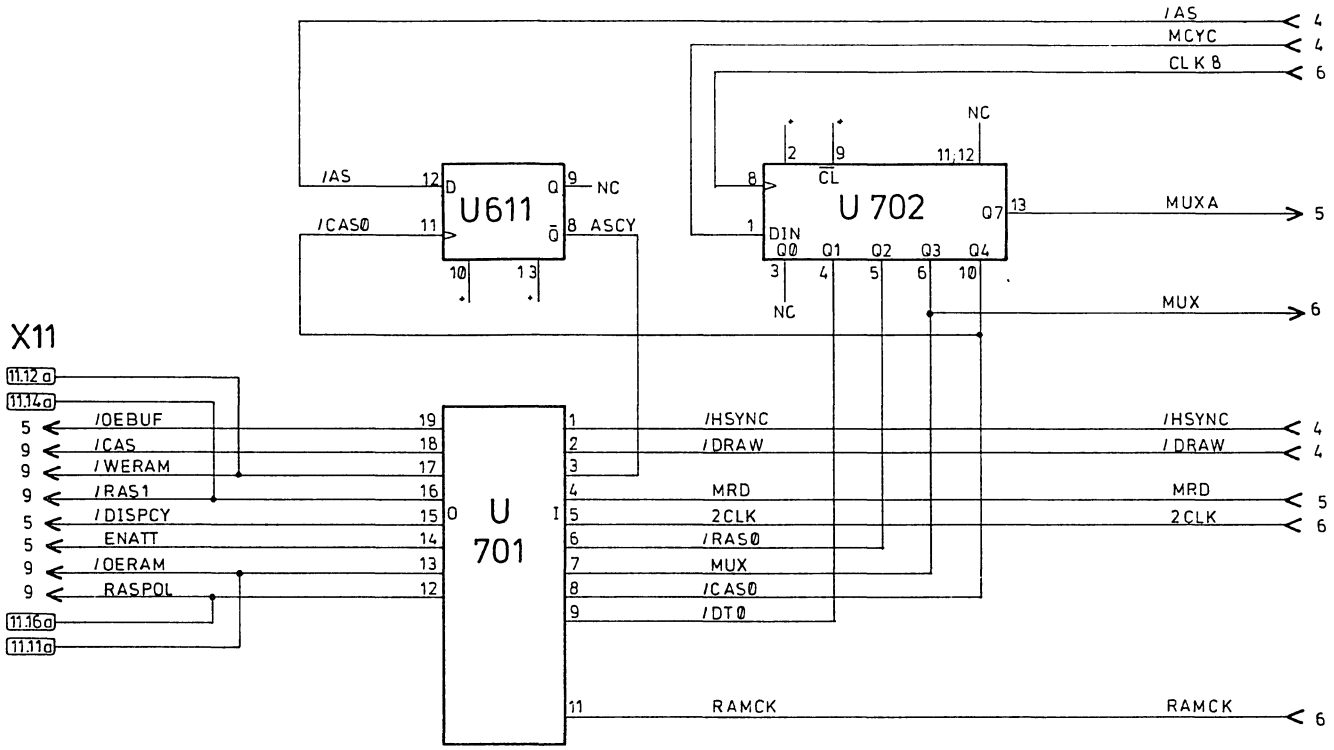


Pos.	St.-Z	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype PAGE 6 of 10			
		PIG - X./68K			

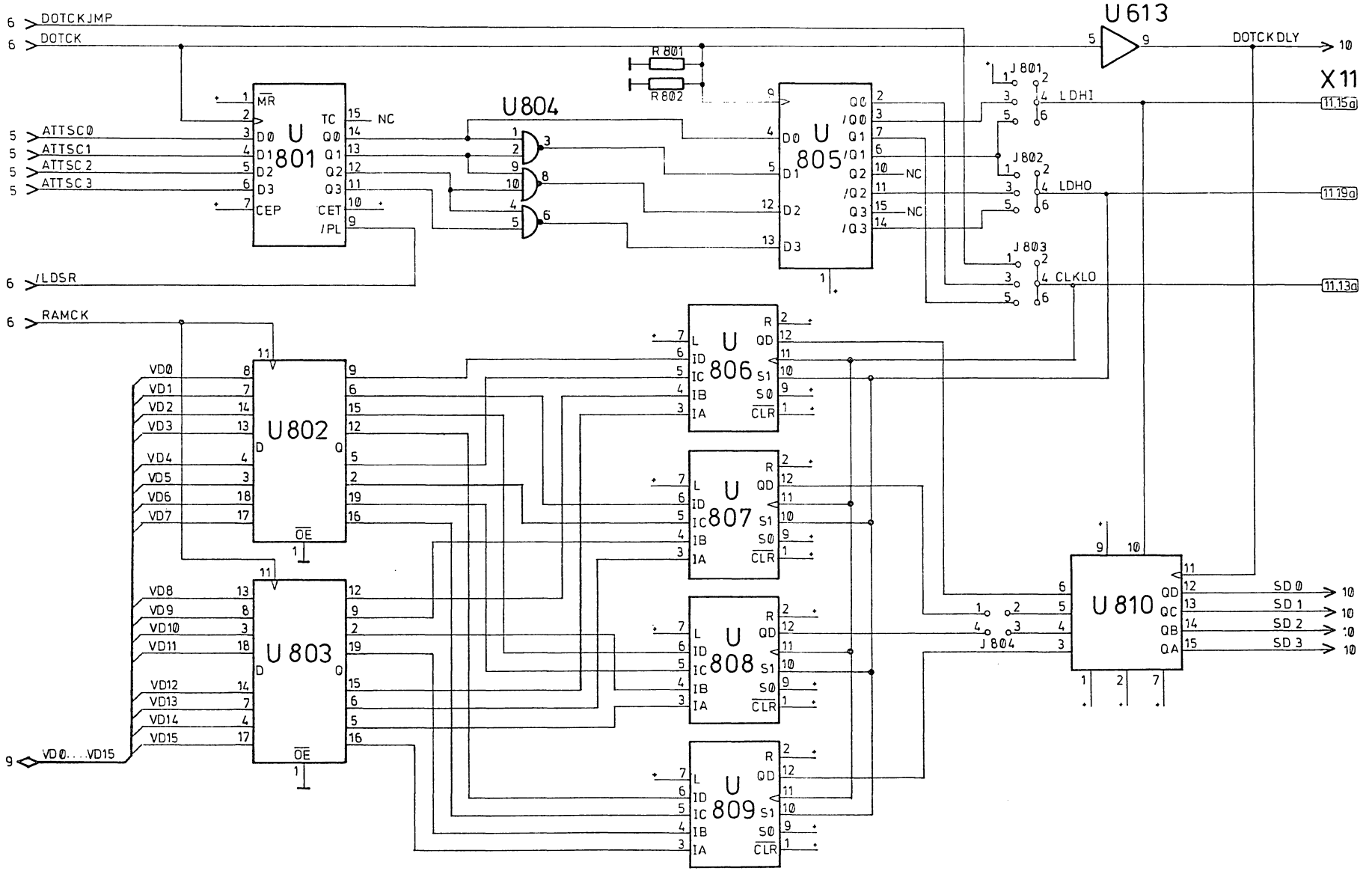
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:
%	4.10.85		DIAGRAM 6	68K/E/3303

Ersatz für

ELTEC



Pos.	St.-Z.	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype: PAGE 7 of 10		ELTEC	
		PIG - X./68K			
Zust.	Änderung	Datum Name			
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
x	4.10.85		DIAGRAM 7	68K/E/3303	
				Ersatz für	

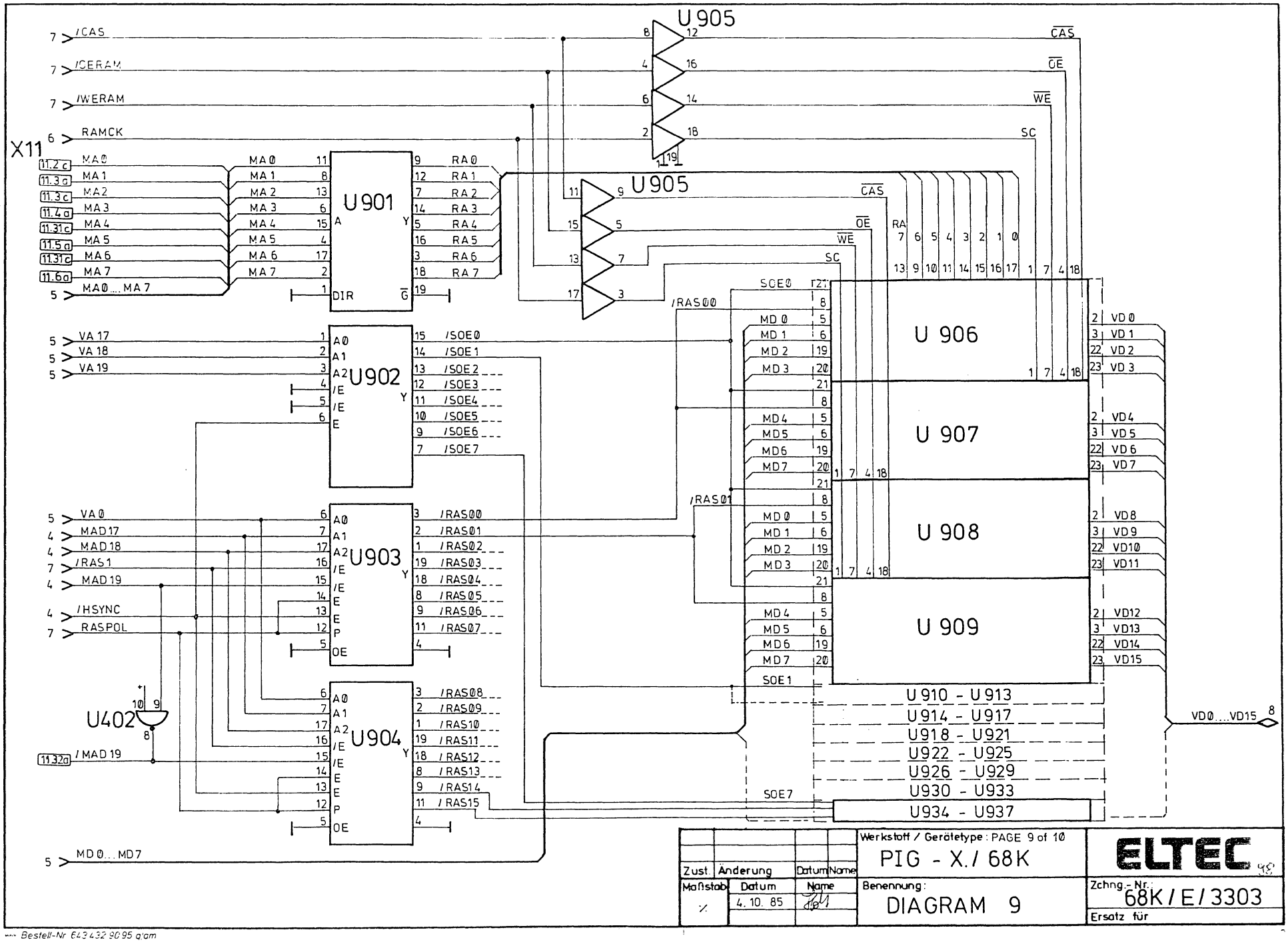


Pos.	St.-Z.	Benennung	Werkstoff	DIN-Nr.	Zeichnungs-Nr.
		Werkstoff / Gerätetype: PAGE 8 of 10			
		PIG - X./68K			
Zust.	Änderung	Datum	Name		
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
1/1	4.10.85		DIAGRAM 8	68K/E/3303	
				Ersatz für	

ELTEC

Weitergabe und Vervielfältigung nur mit unserer Genehmigung

Bestell-Nr. 643432 9095 g/gm



Zust.	Änderung	Datum	Name
Maßstab:	Datum: 4. 10. 85	Name:	
%			

Werkstoff / Gerätetype: PAGE 9 of 10

PIG - X./ 68K

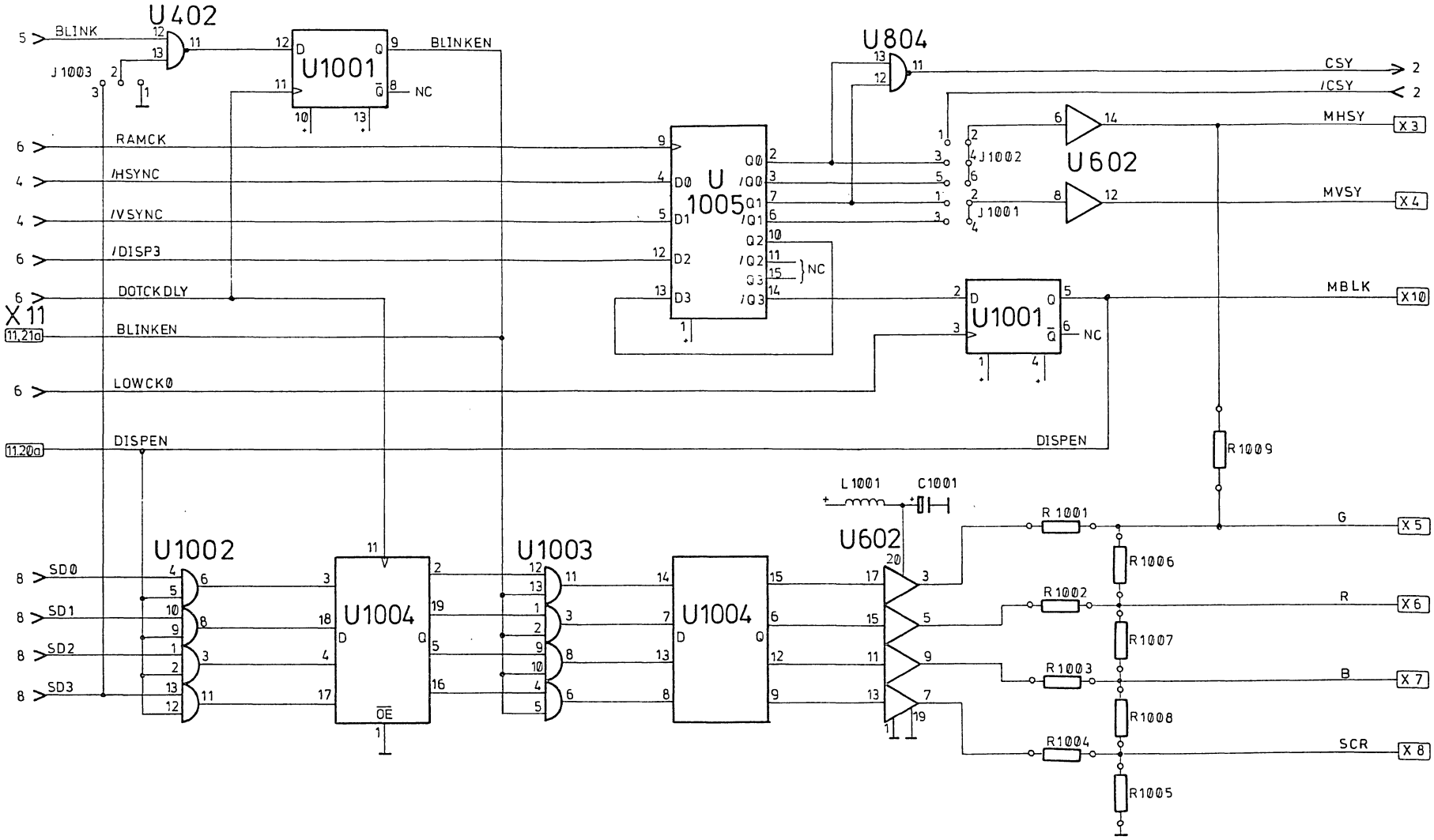
Benennung:

DIAGRAM 9

ELTEC 98

Zchng.-Nr.: 68K / E / 3303

Ersatz für



Pos.	St.-Z.	Benennung	Werkstoff	DIN - Nr.	Zeichnungs - Nr.
		Werkstoff / Gerätetype PAGE 10 of 10			
		PIG - X./68K			
Maßstab	Datum	Name	Benennung:	Zchng.-Nr.:	
/	4.10.85		DIAGRAM 10	68K/E/3303	
				Ersatz für	

ELTEC

Weitergabe und Vervielfältigung nur mit unserer Genehmigung

HD 63484

Advanced CRT Controller (ACRTC)

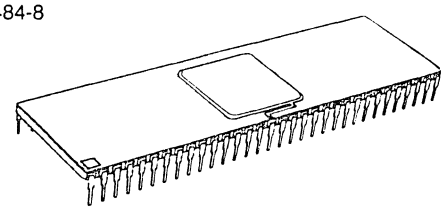
HITACHI

The 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 also a new generation CRT controller that is based on a bit-mapped technology and has more display control functions than those of an HD6845S (CRTC).

The ACRTC prepares the mechanisms to use at 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, as the key devices.

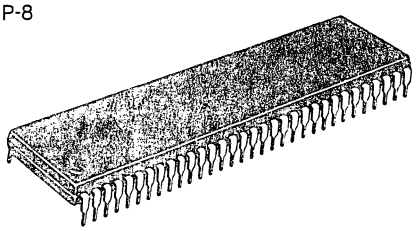
The ACRTC can reduce a CPU software overhead and enhance system throughput.

HD63484-4, HD63484-6, HD63484-8



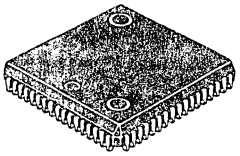
(DC-64)

HD63484P-4, HD63484P-6, HD63484P-8



(DP-64)

HD63484CP-4, HD63484CP-6, HD63484CP-8



(CP-68)

■ FEATURES

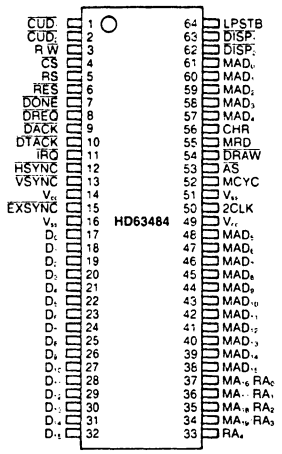
- High speed graphic drawings
  - Drawing rate : Maximum 500 ns/pixel (Color drawing)
  - Drawn graphics : Dot, Line, Rectangle, Poly-line, Poly-gon, Circle, Ellipse, Paint, Copy, etc.
  - Drawn colors : 16-bit/word
    1-, 2-, 4-, 8-, 16-bit/pixel (5 types) monochrome to max. 64k colors.
- Large frame memory space
  - Maximum 2M bytes graphic memory
    128k bytes character memory
    separated from the MPU memory
  - Available to maximum 4096 x 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 (ex. TV system or other controller)
- DMA interface
- Two programmable cursors
- Three scan modes
  - Non-interlace, Interlace Sync. and Interlace Sync. & Video modes
- Interrupt request to MPU
- 256 characters/line, 32 rasters/line, 4096 rasters/screen
- Maximum clock frequency 8 MHz
- CMOS, +5V single power supply

■ TYPE OF PRODUCTS

ACRTC	Clock Frequency (2CLK)
HD63484-4	4 MHz
HD63484-6	6 MHz
HD63484-8	8 MHz

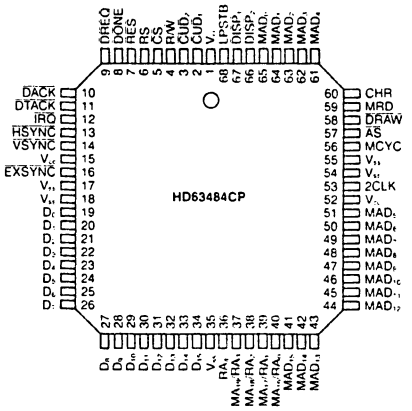
■ PIN ARRANGEMENT

• HD63484, HD63484P PIN ARRANGEMENT



(Top View)

• HD63484CP PIN ARRANGEMENT



(Top View)

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Rating	Unit
Supply Voltage	V_{CC}*	−0.3~+7.0	V
Input Voltage	V_{in}*	−0.3~V_{CC}+0.3	V
Allowable Output Current	$\|I_O\|$**	5	mA
Total Allowable Output Current	$\|\Sigma I_O\|$***	120	mA
Operating Temperature	T_{opr}	0~+70	°C
Storage Temperature	T_{stg}	−55~+150	°C

* This value is in reference to Vss = 0V.

** 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.

*** 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.

(Note) Using an LSI beyond its maximum ratings may result in its permanent destruction. LSI's should usually be used under recommended operating conditions. Exceeding any of these conditions may adversely affect its reliability.

■ RECOMMENDED OPERATING CONDITIONS

Item	Symbol	min	typ	max	Unit
Supply Voltage	V_{CC}*	4.75	5.0	5.25	V
Input "Low" Level Voltage	V_{IL}*	0	–	0.7	V
Input "High" Level Voltage	V_{IH}*	2.2	–	V_{CC}	V
Operating Temperature	T_{opr}	0	25	70	°C

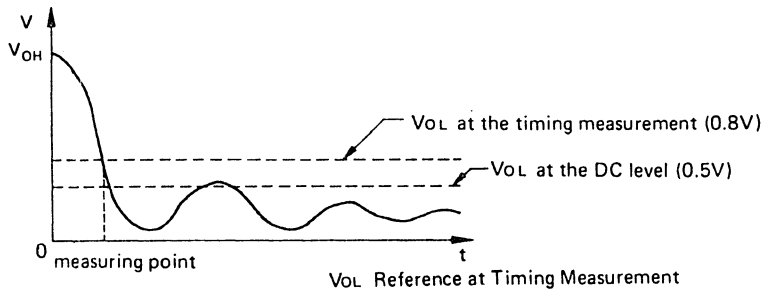
* This value is in reference to Vss = 0V.

■ 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.



VOL Reference at Timing Measurement

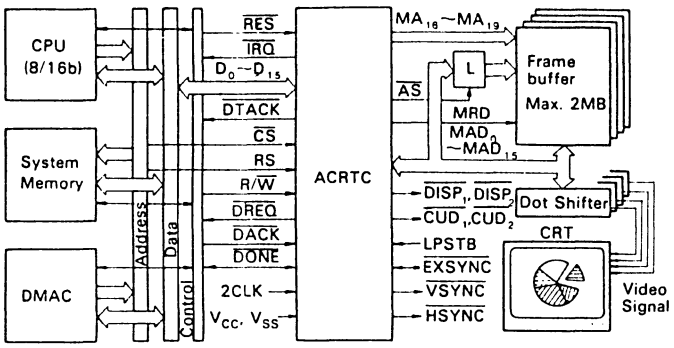


Figure 23 System Configuration

■ INTERNAL FUNCTIONS

• 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

Manages the asynchronous host MPU interface including the programmable interrupt control unit and DMA handshaking

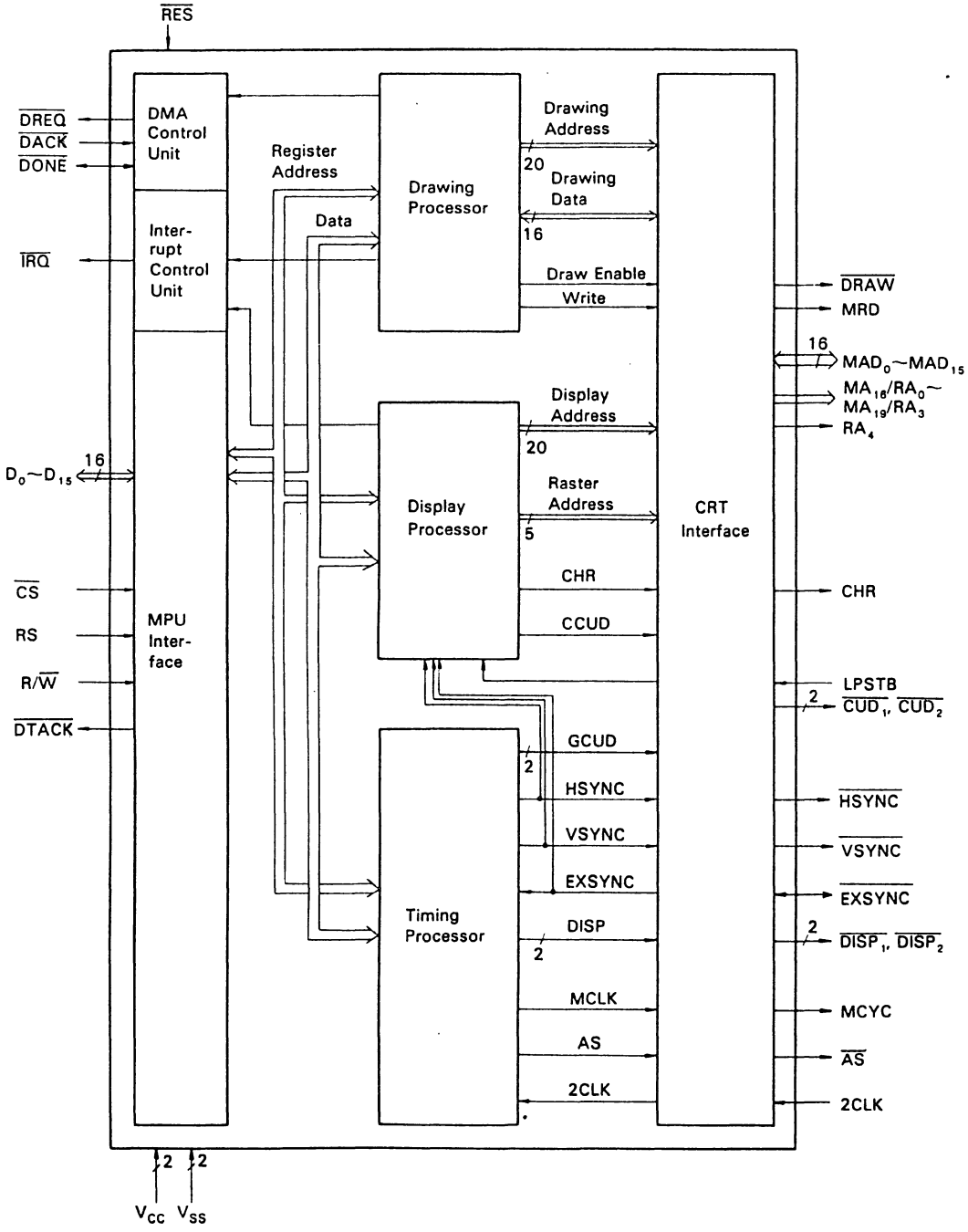


Figure 24 Block Diagram

control unit.

- 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
  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 via separate 16 byte read and write FIFOs.
- Display Processor
  Manages frame buffer refresh addressing based on the user programmed specification of display screen organization. Combines and displays as many as 4 independent screen segments (3 horizontal splits and 1 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
  Generates the CRT synchronization signals and other timing signals used internally by the ACRTC.
  The ACRTCs 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 ACRTCs Drawing processor via the on-chip FIFOs.

• SIGNAL DESCRIPTION

Following is a brief description of the ACRTC pin functions organized as MPU Interface, DMAC Interface, CRT Interface and Power Supply.

MPU INTERFACE

$\overline{RES}$ – Input
Hardware reset input to the ACRTC.

$D_0 \sim D_{15}$ – Input/Output
The bidirectional data bus for communication with the host MPU or DMAC. In 8 bit data bus mode, $D_0 \sim D_7$ are used.

R/$\overline{W}$ – Input
Controls the direction of host → ACRTC transfers.

$\overline{CS}$ – Input
Enables data transfers between the host and the ACRTC.

RS – Input
Selects the ACRTC register to be accessed and is normally connected to the least significant bit of the host address bus.

$\overline{DTACK}$ – Output
Provides asynchronous bus cycle timing and is compatible with the HD68000 MPU $\overline{DTACK}$ input.

$\overline{IRQ}$ – Output
Generates interrupt service requests to the host MPU.

DMAC INTERFACE

$\overline{DREQ}$ – Output
Generates DMA service requests to the host DMAC.

$\overline{DACK}$ – Input
Receives DMA acknowledge timing from the host DMAC.

$\overline{DONE}$ – Input/Output
Terminates DMA transfer and is compatible with the HD68450 DMAC $\overline{DONE}$ signal.

CRT INTERFACE

2CLK – Input
Basic ACRTC operating clock derived from the dot clock.

$MAD_0 \sim MAD_{15}$ – Input/Output
Multiplexed frame buffer address/data bus.

$\overline{AS}$ – Output
Address strobe for demultiplexing the frame buffer address/data bus ($MAD_0 \sim MAD_{15}$).

$MA_{16}/RA_0 \sim MA_{19}/RA_3$ – Output
The high order address bits for graphic screens and the raster address outputs for character screens.

RA_4 – Output
Provides the high order raster address bit (up to 32 rasters) for character screens.

CHR – Output
Indicates whether a graphic or character screen is being accessed.

MCYC – Output
Frame buffer memory access timing – one half the frequency of 2CLK.

MRD – Output
Frame Buffer data bus direction control.

$\overline{DRAW}$ – Output
Differentiates between drawing cycles and CRT display refresh cycles.

$\overline{DISP_1}$, $\overline{DISP_2}$ – Output
Programmable display enable timing used to selectively enable, disable and blank logical screens.

$\overline{CUD_1}$, $\overline{CUD_2}$ – Output
Provides cursor timing determined by ACRTC programmed parameters such as cursor definition, cursor mode, cursor address, etc.

$\overline{VSYNC}$ – Output
CRT device vertical synchronization pulse.

$\overline{HSYNC}$ – Output
CRT device horizontal synchronization pulse.

$\overline{EXSYNC}$ – Input/Output
For synchronization between multiple ACRTCs and other video signal generating devices.

LPSTB – Input
Connection to an external light pen.

VIDEO ATTRIBUTES

The ACRTC outputs 20 bits of video attributes on $MAD_0 \sim MAD_{15}$ and $MA_{16}/RA_0 \sim MA_{19}/RA_3$. These attributes are out-

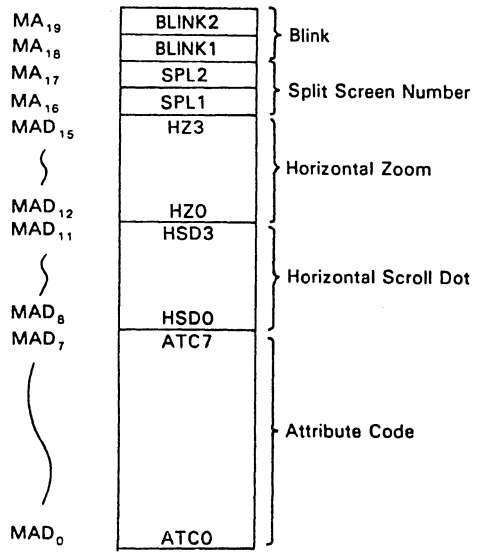


Figure 25 Video Attributes

put at the last cycle prior to the rising edge of $\overline{HSYNC}$ and should be latched externally. Thus, video attributes can be set on a raster by raster basis.

Attribute Code (ATC0~ATC7:$MAD_0 \sim MAD_7$)

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) 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 function to provide the with application to the user and appropriate data need to be employed depending on the system requirement.

Followings show some of application example.

(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).
(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 (max. 8 bit) or selection signal etc.

Horizontal Scroll Dot (HSD0~HSD3:$MAD_8 \sim MAD_{11}$)

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 OMR (Operation Mode Register) is set to 1, HSD outputs the start dot address of the window screen segment.

Note) 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:$MAD_{12} \sim MAD_{15}$)

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:$MA_{16} \sim MA_{17}$)

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:$MA_{18} \sim MA_{19}$)

The lines alternate from high to low periodically as defined in the Blink Control Register (BCR). the blink frequency is specified in units of 4 field times. A field is defined as the period between successive VSYNC pulses. These lines are used to implement character and screen blink.

◎ 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 26 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 screen 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.

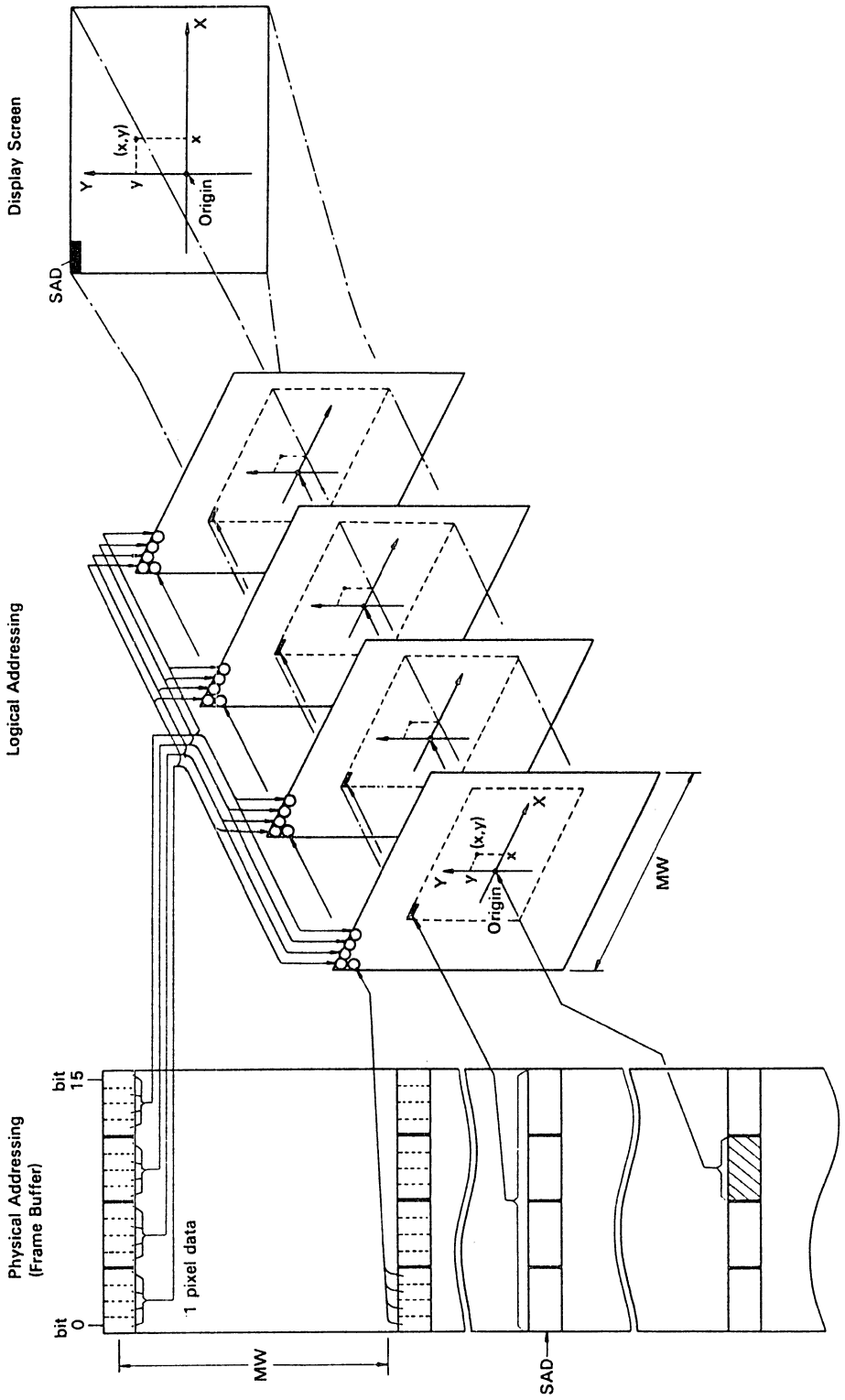


Figure 26 Logical/Physical Addressing

- **REGISTERS**

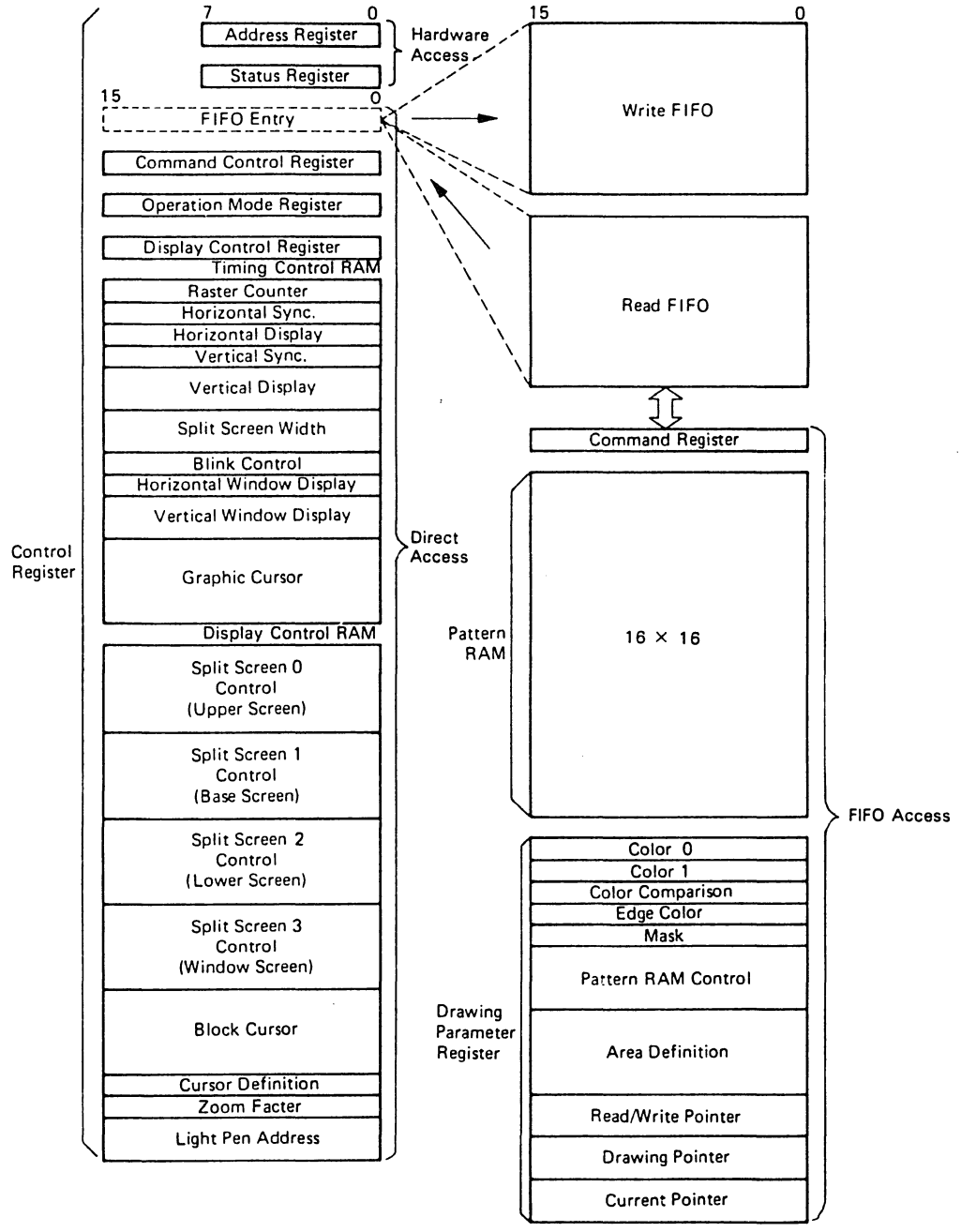


Figure 27 Programming Model

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

CS	RS	R/W	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
		0	r00	FIFO Entry	FE	FE															
		0	r02	Command Control	CCR	ABT	PSE	DDM	CDM	DRC		GBM		CRE	ARE	CEE	LPE	RFE	RRE	WRE	WEE
		0	r04	Operation Mode	OMR	M	S	STR	ACP	WSS	CSK		DSK	RAM	GAI			ACM		RSM	
		0	r06	Display Control	DCR	DSP	SE1		SE0		SE2		SE3	ATR							
		-	r08 ↓ r7E	(undefined)	—	.....								.....							
		1	r80	Raster Count	RCR	.....								RC							
		0	r82	Horizontal Sync.	HSR	HC								.....				HSW			
			r84	Horizontal Display	HDR	HDS								HDW							
		0	r86	Vertical Sync.	VSR	.....								VC							
		0	r88	Vertical Display	VDR	VDS								.....				VSW			
		0	r8A	Split Screen Width	SSW	.....								SP1							
		0	r8C			.....								SP0							
		0	r8E			.....								SP2							
		0	r90	Blink Control	BCR	BON1					BOFF1			BON2				BOFF2			
		0	r92	Horizontal Window Display	HWR	HWS								HWW							
		0	r94	Vertical Window Display	VWR	.....								VWS							
		0	r96			.....								VWW							
		0	r98	Graphic Cursor	GCR	CXE								CXS							
		0	r9A			.....								CYS							
		0	r9C			.....								CYE							
		-	r9E ↓ rBE	(undefined)	—	.....								.....							
0	1	0	rC0	Upper Screen Base Screen	RAR0	.....				LRA0				.....				FRA0			
		0	rC2		MWR0	CHR	.....							MW0							
		0	rC4		SAR0	.....				SDA0				.....				SA0H SRA0			
		0	rC6			SA0L															
		0	rC8	Lower Screen Base Screen	RAR1	.....				LRA1				.....				FRA1			
		0	rCA		MWR1	CHR	.....							MW1							
		0	rCC		SAR1	.....				SDA1				.....				SA1H SRA1			
		0	rCE			SA1L															
		0	rD0	Window Screen Base Screen	RAR2	.....				LRA2				.....				FRA2			
		0	rD2		MWR2	CHR	.....							MW2							
		0	rD4		SAR2	.....				SDA2				.....				SA2H SRA2			
		0	rD6			SA2L															
		0	rD8	Window Screen Base Screen	RAR3	.....				LRA3				.....				FRA3			
		0	rDA		MWR3	CHR	.....							MW3							
		0	rDC		SAR3	.....				SDA3				.....				SA3H SRA3			
		0	rDE			SA3L															
		0	rE0	Block Cursor 1	BCUR1	BCW1				BCSR1				.....				BCER1			
		0	rE2			BCA1															
		0	rE4	Block Cursor 2	BCUR2	BCW2				BCSR2				.....				BCER2			
		0	rE6			BCA2															
		0	rE8	Cursor Definition	CDR	CM			CON1		COFF1			.....		CON2		COFF2			
		0	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

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 : Commad End Interrupt Enable
CER : Command Error
CRE : Command Error Interrupt Enable
CSK : Cursor Display Skew
DDM : Data DMA Mode
DRC : DMA Request Control
DSK : $\overline{\text{DISP}}$ Skew
DSP : $\overline{\text{DISP}}$ 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 1 (cont.) 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	
				XMAX	
Pr0B				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				—	

▨ . . . 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 bit of Read Write Pointer Address
RWPL	: Low-order 12 bit of Read Write Pointer Address
DPAH	: High-order 8 bit of Drawing Pointer Address
DPAL	: Low-order 12 bit of Drawing Pointer Address
DPD	: Drawing Pointer Dot Address
X, Y	: Position indicated by Current Pointer on X-Y coordinate

The ACRTC has over two hundred bytes of accessible registers. These are organized as Hardware, Directly and FIFO accessible.

- 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.
  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 ACRTCs 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 ACRTCs 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 ACRTCs Read Parameter Register (RPR) and Write Parameter Register (WPR) register access commands.

■ COMMANDS

The ACRTC has 38 commands classified into three groups — REGISTER ACCESS, DATA TRANSFER and GRAPHIC DRAWING.

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 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.

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.

Table 2 ACRTC Comrnand Table

TYPE	MNEMONIC	COMMAND NAME	OPERATION CODE	PARAMETER	= (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 0 RN	D	2	6
	RPR	Read Parameter Register	0 0 0 0 1 1 0 0 0 0 0 0 RN		1	6
	WPTN	Write Pattern RAM	0 0 0 1 1 0 0 0 0 0 0 0 PRA	n $D_1, \ldots, D_n$	n+2	4n+8
	RPTN	Read Pattern RAM	0 0 0 1 1 1 0 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 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	12
	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 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 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 1 0 0 C 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 0 0 0		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

*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 (Stp at excepting Edge color)] C: [C=1 (clockwise), C=0 (reverse)]
[↑]: rounding up
P = 4: OPM-000~011
6: OPM-100~111

*3) cycles: 2clock cycle time

REGISTER ACCESS COMMAND

Mnemonic	Operation Code	Parameter			# (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 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,·····,Dn		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,·····, Dn : Write data
n : Number of Read Write data

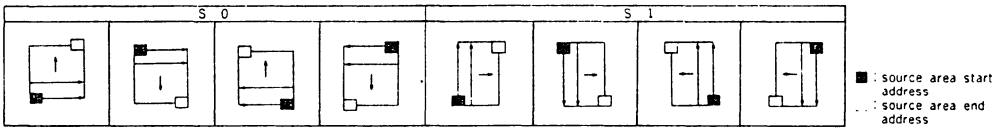
DN	Screen No.
00	Upper Screen
01	Base Screen
10	Lower Screen
11	Window Screen

DATA TRANSFER COMMAND

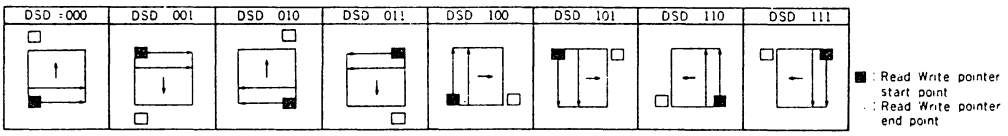
MM : Modify Mode

MM		Functior
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	EOR	EOR drawing point data with modifier data and rewrite the result data to the frame buffer

S : Source Scan Direction



DSD : Destination Scan Direction



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

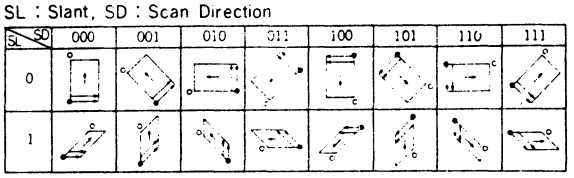
C : Circling Direction

C	Direction
0	Counterclockwise
1	Clockwise

E : Definition of Edge color

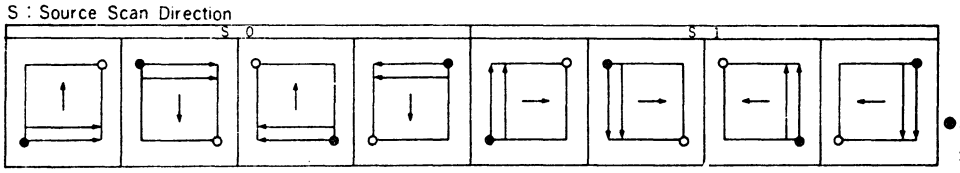
E	Definition
0	Edge color is defined by the data in the edge color register.
1	Edge color is defined by the data excluding the above.

SL : Slant, SD : Scan Direction



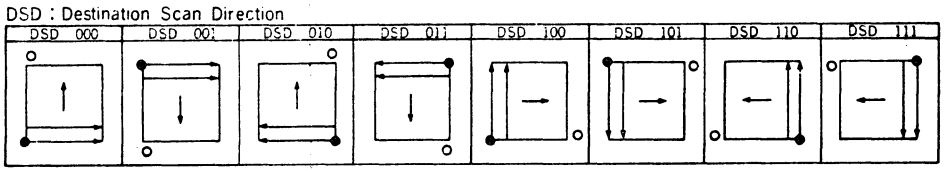
● : current pointer start point
: current pointer end point

S : Source Scan Direction



● : source area start dot position
: source area end dot position

DSD : Destination Scan Direction



● : current pointer start point
: current pointer end point

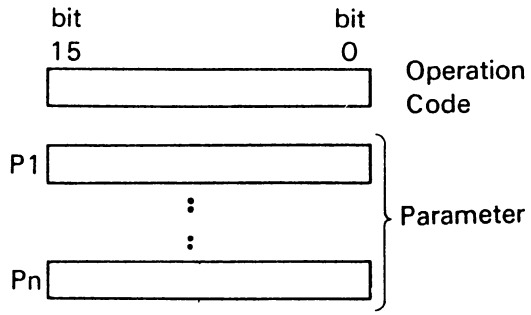
X, X1, . . . , Xn : Absolute X-address from the origin point
Y, Y1, . . . , Yn : Absolute Y-address from the origin point
dX : Relative X-address from the current pointer
dY : Relative Y-address from the current pointer
n : Number of nodes
dX1, . . . , dXn : Relative X-address from each node
dY1, . . . , 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 start dot position in source area
dYs : Relative Y-address from the current pointer to the start 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

• COMMAND FORMAT

ACRTC commands consist of a 16 bit op-code, optionally followed by 1 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.

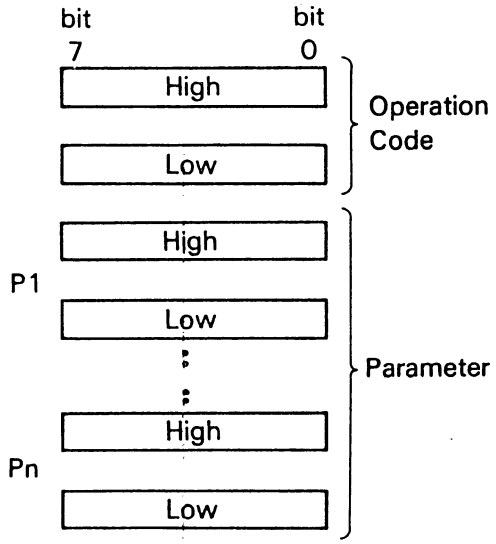
(a) 16 bit interface

In the case of 16 bit interface, first move the 16 bit operation code and then move necessary 16 bit parameters one by one.



(b) 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 those of parameters in the same order.



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 ACRTC FIFO status by software polling or interrupts.

- Software Polling (WFR, WFE interrupts disabled)
  - a) MPU program checks the SR (Status Register) for Write FIFO Ready (WFR) flag = 1, and the writes 1-word op-code/parameters.
  - b) 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)
  - a) MPU WFR interrupt service routine writes 1-word op-code/parameters.
  - b) 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 in 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.

Note)
- Make sure that the write FIFO is empty and all the commands are terminated before starting the Command DMA Transfer.
- The Data DMA Command cannot be executed in the Command DMA Transfer Mode.
- In the R mask and S mask version, the Command DMA Transfer is not in use.

• 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.

Table 3 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 4 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) tc another area (destination area) subject to logical modification by word. The source and destination areas must reside on the same screen. (bit maskable)

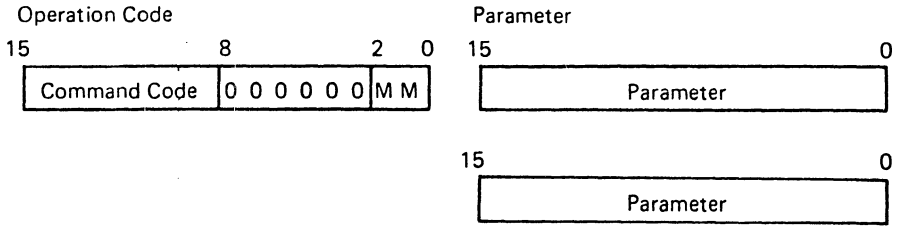


Figure 28 Data Transfer Command Format

MODIFY MODE

The DMOD, MOD, SCLR and SCPY commands allow 4 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 rewirte to the frame buffer.
1	0	AND frame buffer data with command parameter data and rewrite to the frame buffer.
1	1	EOR frame buffer data with command parameter data and rewrite to the frame buffer.

- **GRAPHIC DRAWING COMMANDS**

The ACRTC has 23 separate graphic drawing commands. Graphic drawing is performed by modifying the contents of the frame buffer based upon 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 (for example, a given logical pixel in 4 bits per logical pixel mode might reside in bits 8-11 of a frame buffer word) is performed at high speed by ACRTC hardware.

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

(a) 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.

(b) 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.
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).
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.
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.

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

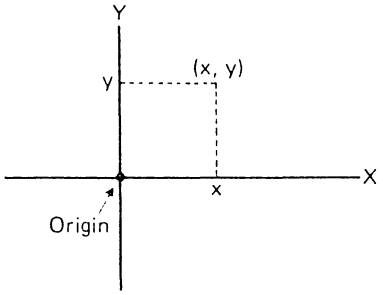


Figure 29 Absolute Coordinate Specification

(ii) Relative Coordinate Specification
Specifies the relative addresses (△x, △y) related to the current drawing point.

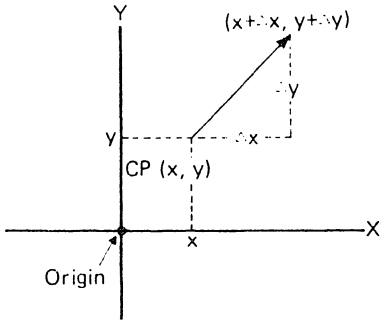


Figure 30 Relative Coordinate Specification

Table 5 Graphic Drawing Commands

Command	Function
AMOVE	Move the Current Pointer (CP) to an absolute logical pixel X-Y address.
RMOVE	Move the Current Pointer (CP) to a relative logical pixel X-Y address.
ALINE	Draw a straight line from the Current Pointer (CP) to a command specified end point of the absolute coordinates.
RLINE	Draw a straight line from the Current Pointer (CP) to a command specified end point of the relative coordinates.
ARCT	Draw a rectangle defined by the Current Pointer (CP) and a command specified diagonal point of the absolute coordinates.
RRCT	Draw a rectangle defined by the Current Pointer (CP) and a command specified diagonal point of the relative coordinates.
APLL	Draw a polyline (multiple contiguous segments) from the Current Pointer (CP) through command specified points of the absolute coordinates.
RPLL	Draw a polyline (multiple contiguous segments) from the Current Pointer (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 start pointer (CP) and command specified points of the relative coordinates.
CRCL	Draw a circle of the radius R placing the Current Pointer (CP) at the center.
ELPS	Draw a ellipse whose shape is specified by command parameters, placing the Current Pointer (CP) at the center.
AARC	Draw an arc by using the Current Pointer (CP) as a start point with an end point and a center point of the absolute coordinates.
RARC	Draw an arc by using the Current Pointer (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 Current Pointer (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 Current Pointer (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 Current Pointer (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 Current Point (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 Current Point (CP) indicates.
PTN	Draw a graphic pattern defined in the Pattern RAM onto a rectangular area specified by the Current Pointer (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 Current Pointer (CP). (rotation angle: 90°/mirror turnover)
RGCPY	Copy a rectangular area specified by the relative coordinates to the address specified by the Current Pointer (CP). (rotation angle: 90°/mirror turnover)

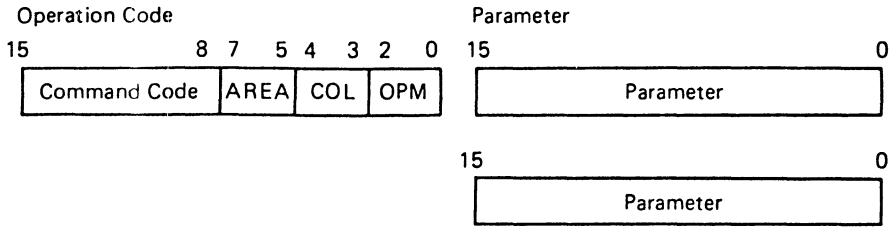


Figure 31 Graphic Drawing Command Format

OPERATION MODE

The Operation Mode (OPM bits) of the Graphic Drawing Command specify the logical drawing condition.

Figure 32 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 *	EOR: EORs 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).

Figure 32 shows examples of a drawing pattern applied with various OPM modes.

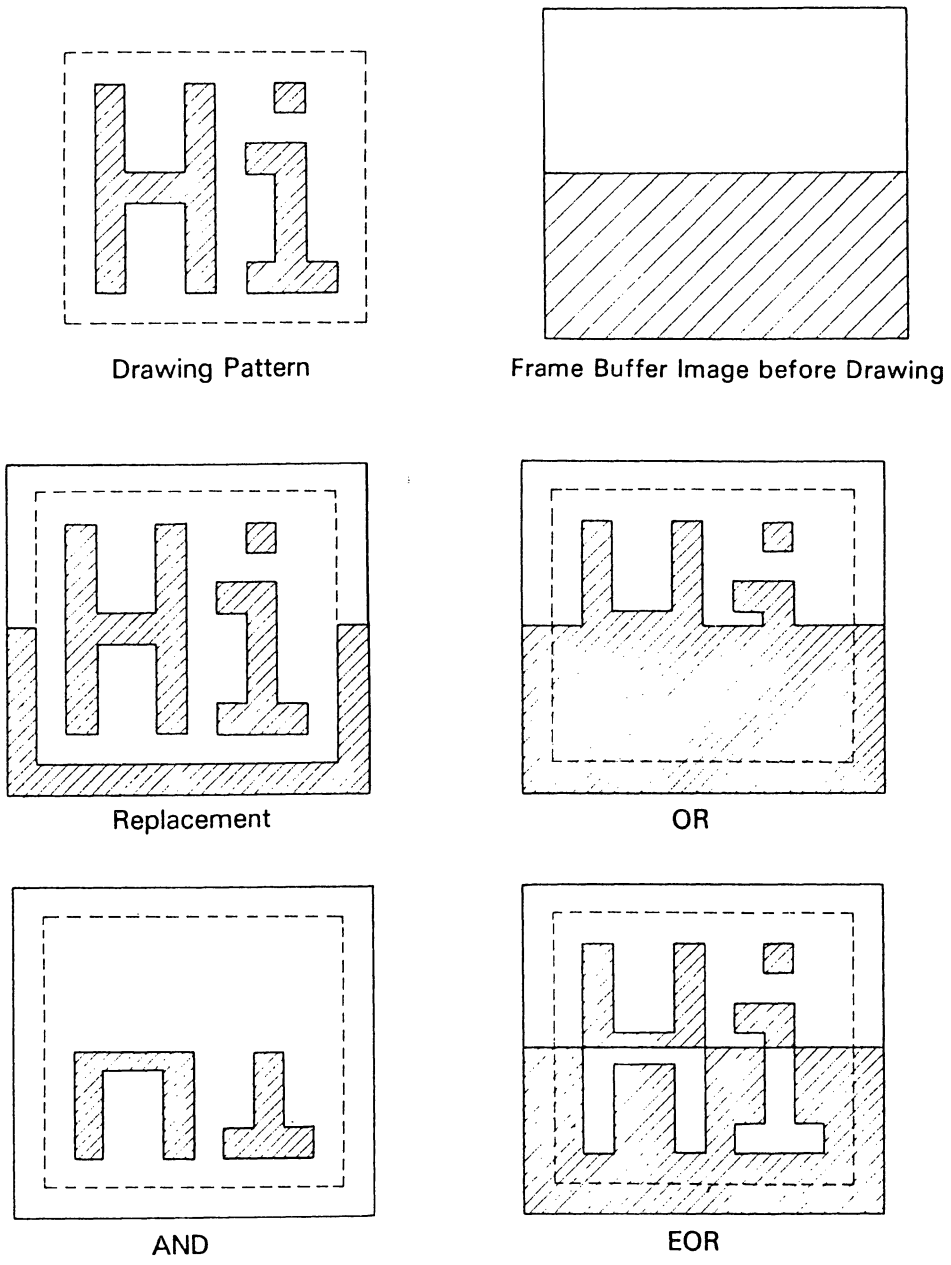


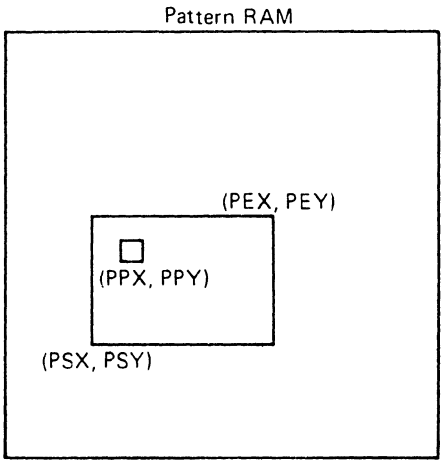
Figure 32 Operation Mode Example

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.



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

Color Register = | 0 0 0 1 | 0 0 0 1 | 0 0 0 1 | 0 0 0 1 |

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.

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 = 0 0 0.
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.