ETH
Eidgenössische
Technische
Hochschule
Zürich

Institut
für Informatik

H.Eberle

Hardware Description of the Workstation Ceres

Januar 1987

RC.87.70

ETH
Eidgenössische
Technische
Hochschule
Zürich

Institut
für Informatik

Eidg. Techn. Hochschule Zürich
Informatikbibliothek
ETH-Zentrum
CH-8092 Zürich

H.Eberle

Hardware Description of the Workstation Ceres

Eidg. Techn. Hochschule Zürich
Informatikbibliothek
ETH-Zentrum
CH-8092 Zürich

87.4.23

Address of the author:

Institut für Informatik
ETH-Zentrum
CH-8092 Zürich / Switzerland

© 1987 Institut für Informatik, ETH Zürich

Hardware Description of the Workstation Ceres

H. Eberle

Abstract

Ceres is a single-user computer based on the 32-bit microprocessor NS32000. The processor is oriented to the use of high-level languages. The hardware design concentrates on simplicity and modularity. The key features are the arbitrated memory bus and the high resolution bitmapped graphics display which promotes the attractivity of a programming workstation. This paper documents the hardware of the Ceres computer.

Contents

1 Introduction 3

2 Hardware Structure 4
- 2.1 Processor 4
- 2.2 Primary Memory 4
- 2.3 Secondary Memory 4
- 2.4 Input/Output Devices 4

3 Hardware Implementation 6
- 3.1 Processor Board 6
  - 3.1.1 Processor 6
  - 3.1.2 Memory Bus Arbiter 9
  - 3.1.3 Boot ROM and Standard Input/Output Devices 12
- 3.2 Memory Board 15
- 3.3 Display Controller Board 18
  - 3.3.1 Display Memory 18
  - 3.3.2 Display Refresh Controller 19
- 3.4 Disk Controller Board 23
- 3.5 Motherboard 23

4 Hardware Extensions 24

References 26

Appendices

A Circuit Diagrams 28
- A.1 Processor Board 28
- A.2 Memory Board 35
- A.3 Display Controller Board 37

B PAL Design Specifications 42

C Interface Connectors 47
- C.1 Processor Board 47
- C.2 Display Controller Board 48
- C.3 Motherboard 48

1 Introduction

Todays working tools of a software engineer at the Institut für Informatik of ETH Zürich are the programming language Modula-2 [1] and the workstation computer Lilith [2]. The architecture of Lilith is optimized for the development and execution of Modula-2 programs. The processor is a microprogrammed implementation of a stack machine based on bit-slice technology. From the Lilith project it was learned that the architecture of a computer should be designed according to the programming language used.

The rapidly evolving VLSI technology has provided the motivation to design a new workstation. The architecture should incorporate one of the recent microprocessors oriented to the use of high-level languages [3]. The design should be simple and modular, which also means expandable. The project has resulted in a compact 32-bit single-user workstation with a performance comparable to a VAX 11-780 [4]. With respect to the Lilith computer, the component count has been reduced by a factor of two.

The computer has been named CERES, an acronym for Computing Engine for Research, Engineering and Science. (In Greek mythology, Ceres is the name of the goddess of fertility.)

The Ceres project started in early 1984 when concepts of the hardware architecture were first proposed by Professor N. Wirth. As a result of close cooperation with the author, a prototype was produced a year later in the spring of 1985. The prototype was based on the 16-bit processor NS32016 from National Semiconductor. At that time, it was seen that future developments of integrated circuits would be concentrated on 32-bit processors; therefore, a second prototype based on the 32-bit processor NS32032 was developed which is software compatible with its family member NS32016. In the fall of 1985, the redesign was complete. By the end of 1986, a series of 30 computers has been built.

The venture to design a personal computer was also shared with Frank Peschel and Matthias Wille, who have ported the Lilith operating system MEDOS-2 [5]. A one-pass Modula compiler was developed by Professor N. Wirth [6]. Roger Burlet and Immo Noack made it possible that the Ceres computer can be manufactured today in series quantity.

This report is a technical manual which provides insight into the hardware implementation of the Ceres computer. The hardware designer will be able to add his own extensions, while the system software designer will have better knowledge of the underlying computer organization. Justification and comparative analysis of concepts are not subjects of this paper.

2 Hardware Structure

The Ceres hardware consists of a 32-bit processor based on the National Semiconductor Series 32000 chip set, primary memory, secondary memory, and miscellaneous input and output devices. These include a high resolution display, a serial keyboard, a mouse pointing device, a RS-232-C serial line interface, and a RS-485 serial line interface. Figure 2.1 is a block diagram of the hardware structure. This section gives a brief description of the main hardware characteristics.

2.1 Processor

The heart of the Ceres computer is a National Semiconductor NS32032 32-bit microprocessor. Two slave processors add the capabilities of virtual memory management and of floating point arithmetic. The processor operates at a clock rate of 10 MHz, resulting in a memory cycle time of 400 ns. It has an addressing range of 16 MBytes. Its repertoire includes 83 basic instructions with 9 addressing modes.

2.2 Primary Memory

The primary storage of Ceres consists of 2 MBytes of dynamic RAM, 256 KBytes of video RAM, and 32 KBytes of ROM. The former is implemented with 256 Kbit dynamic RAM chips. Parity checking makes it possible to detect single bit errors within a data byte. A special type of dynamic RAM, a 64 Kbit video RAM, is used to store the display bitmap. 64 Kbit chips form the ROM memory, which contains bootstrap and diagnostic software.

2.3 Secondary Memory

The secondary storage of Ceres consists of a Winchester hard disk drive and a floppy disk drive. The 5 1/4" hard disk has a formatted capacity of 40 MBytes, an access time of 40 ms, and a data transfer rate of 5 Mbits per second. For backup, a 3 1/2" floppy disk is available with a formatted capacity of 720 KBytes, an access time of 94 ms, and a data transfer rate of 250 Kbits per second.

2.4 Input/Output Devices

Display

The display is a high resolution raster scan monitor. It can display 819'200 dots which are stored in a matrix called bitmap that is 1024 dots wide and 800 dots high. The picture is refreshed with a frequency of 62.15 Hz (noninterlaced) which results in a nearly flicker-free image. The bitmap information is stored in a separate, dedicated memory implemented with video RAMs.

Keyboard and Mouse

In addition to a standard serial ASCII keyboard, an opto-mechanical, three-button mouse is provided which has a resolution of 380 counts per inch.

Communication

The standard RS-232-C serial interface works with asynchronous data transfer rates from 50 to 38'400 bps. A higher transmission speed can be obtained with two RS-485 serial ports which provide data transfer rates up to 230.4 Kbps. In a multipoint configuration, this interface allows the implementation of a low cost computer network.

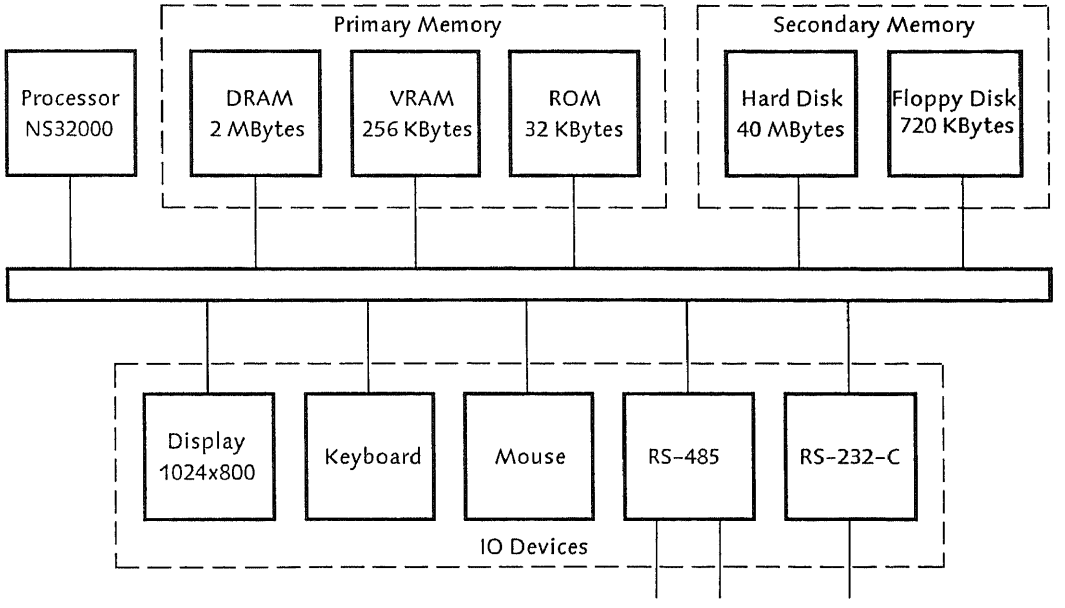


Figure 2.1 Hardware structure of the Ceres computer.

3 Hardware Implementation

The hardware of Ceres is physically divided into several boards which are interconnected by the memory bus:

- the *processor board* contains the processor chip set, the memory bus access and timing controller, the boot ROM, and various IO devices
- the *memory board* holds the dynamic RAM memory
- the *display controller board* comprises the video RAM memory for the displayed bitmap and the logic to serialize the bitmap data into the video refresh data
- the *disk controller board* combines a controller for both the hard disk and the floppy disk drives
- all boards communicate via the *motherboard* which contains the memory bus

Based on the circuit diagrams in appendix A, the hardware of Ceres is explained in the following sections. The specifications of the integrated circuits used can be referred to in the referenced data sheets.

3.1 Processor Board

3.1.1 Processor

The NS32032 central processing unit (CPU) has a uniform linear 16 MByte addressing range and a full 32-bit architecture and implementation [7]. Internal working registers, internal and external data paths, and ALU are all 32 bits wide. There are eight general purpose registers which provide local, high-speed storage for the processor, such as holding temporary variables and addresses. Eight special purpose registers are used for storing address and status information. The register set, the supported data types, and the instruction set are fashioned after high level language instructions [8]. Code generation is made easier by a high degree of symmetry. (Note: a processor's architecture is said to be symmetrical if every supported data type is provided with a complete set of operators and if each operator can use any addressing mode to access operands.)

A slave processor is an auxiliary processing unit which operates in coordination with the CPU. The NS32082 memory management unit (MMU) performs address translation, virtual memory management and memory protection [9]. The NS32081 floating-point unit (FPU) operates on two floating-point data types: single precision (32-bits) and double precision (64-bits). Arithmetic operations include Add, Subtract, Multiply, Divide, and Compare. Several Move and Convert instructions are also available [10].

The structure of the processor and its memory bus interface are illustrated in the block diagram in Figure 3.1. The following blocks may be distinguished:

- the *processor cluster* consists of the NS32032 CPU, the NS32082 MMU, and the NS32081 FPU
- the *timing control unit* generates the clock and reset signals for the processor and the memory bus
- the *memory bus interface* connects the address, data, and control signals of the local, multiplexed processor bus to the demultiplexed memory bus

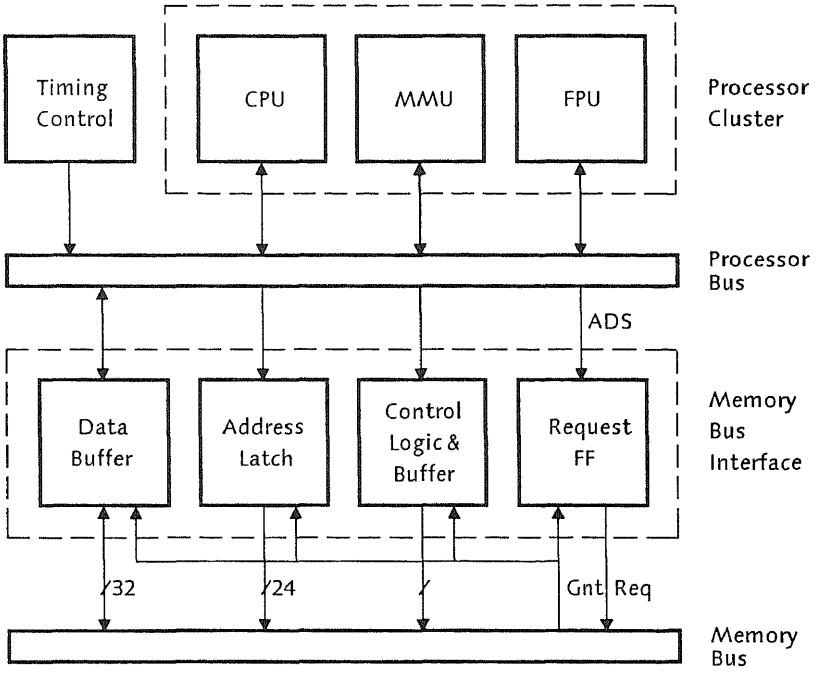


Figure 3.1 The processor and its memory bus interface.

The *processor cluster*, i.e. the CPU (u24) and its slave processors (u23, u25) are connected to a local, multiplexed address and data bus (ad0-ad23, d24-d31) that combines 32 bits of data with 24 bits of address. The local bus is required either for memory access (IO devices are memory mapped) or slave processor communication. In the latter case, only the two least significant bytes of the data bus are used. Note that the CPU is solely responsible for memory access; that is, operands of a slave processor instruction are always fetched from the memory by the CPU.

The NS32201 *timing control unit* (TCU, u36) provides a two phase, nonoverlapping 10 MHz clock (TCU.PHI1, TCU.PHI2) which is used by the processor chips [11]. A 10 MHz and a 20 MHz TTL compatible clock (TCU.CTTL, TCU.FCLK) are also generated (see Figure 3.2). The TCU also provides circuitry that meets the reset requirements of the processor chips. If the reset input line RSTI' is pulled low, the TCU asserts TCU.RST' which resets the processor chips. The RSTI' input signal is provided by the TL7705 (u35) which contains a power voltage sensor and a debounce circuit. It is activated at power-up or when the externally mounted reset button has been pressed. The reset and clock signals on the memory bus (RESET' and CLK, FCLK) are buffered versions of the corresponding TCU signals (u7).

The *memory bus interface* consists of a 32-bit wide data buffer, a 24-bit wide address latch, a buffer for several control lines, and a circuitry that requests a bus cycle when the processor wants to access the memory. The data buffer is made up of four 74ALS645 octal bus transceivers (u3-u6) [22]. Two additional 74ALS645s (u1, u2) are needed in order for the MMU with its 16-bit wide data bus to access the higher data word of the main memory. When the MMU accesses an odd memory word (A1=1), the higher data word of the

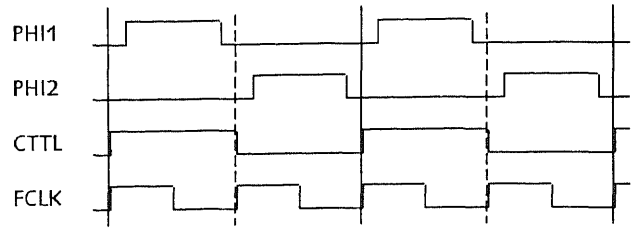


Figure 3.2 TCU clock signals.

memory bus (D16-D31) has to be mapped onto the lower word of the processor bus (ad0-ad15). The necessary signals, GW' and GD', for the buffer enable inputs are generated by part of a PAL16L8A PAL device (u21) [23, 24]. The MMU has to access memory in order to update its internal address translation cache from page table entries in memory or to update certain status bits within them.

The address latch uses three 74ALS573 octal D-type transparent latches (u9-u11). Using the address strobe signal MMU.ADS', the information of the multiplexed address/data bus is retained by the latches at the beginning of a bus cycle. At power-up or after a system reset, a flip-flop (u40a) with the output signal name BT.UP' is set. If BT.UP' is asserted and the CPU is accessing memory, the address information of the signal lines ad19-ad23 will not be gated to the memory address bus lines A19-A23; instead, another ALS573 (u8) sets these lines to high as long as BT.UP' enables this latch. This maps address locations 000000-07FFFF (hex) to F80000-FFFFFF (hex) where the boot ROM and IO devices can be found. Note that the processor starts program execution with a PC value zero after reset. The boot flip-flop is reset irreversibly under software control.

The following control signals are provided: ILO', AV', R/W' and BE0'-BE3'. ILO' is a buffered version of the corresponding CPU signal CPU.ILO' (u7), which indicates that an interlocked instruction is being executed. It is made available to external bus arbitration circuitry in order to implement the semaphore primitive operations for resource sharing. This signal is however not used in the present circuits. AV' marks a valid address on lines A0-A23 and is activated when a processor request has been granted (u34b, u7). R/W' indicates the direction of the data transfer as seen from the processor (u38c, u7). BE0'-BE3' facilitate individual byte accessing on the 32-bit data bus. Any data item, regardless of size, may be placed starting at any memory address; therefore, the 24-bit address A0-A23 is a byte address. While the data bus always transfers double-word data, the memory uses BE0'-BE3' to select the appropriate bytes. A PAL16L8A device (u21) contains the necessary logic to generate the byte enable signals. During a memory write cycle, these signals are defined by either the CPU (CPU.BE0'-CPU.BE3') or the MMU (A1). A CPU memory access can contain one, two, three or four bytes, while the MMU always accesses words. An MMU memory cycle can be identified if MMU.MAC' is low. During a memory read cycle, BE0'-BE3' are all active. This precaution must be taken to prevent floating data buffer inputs caused by nonselected memory devices.

The cycle request circuitry consists of a 74AS74 flip-flop (u22a). It is set by the address strobe signal MMU.ADS' which signals that the processor is starting a bus cycle. A CPU memory cycle request (signalled by CPU.REQ'=0) is acknowledged by the bus arbiter with an active CPU.GNT' signal. CPU.GNT' is used as an output enable of the buffers and latches of

the memory bus interface. The RDY signal is used to extend the current processor bus cycle. This is necessary if the CPU bus cycle request cannot be acknowledged immediately (u39b) or in case of a slow access (u34a).

3.1.2 Memory Bus Arbiter

Processor, display controller, and DRAM refresh timer share access to the main memory and the memory bus. The device that controls the bus is known as the bus master. The transfer of bus mastership from one device to another is defined by a set of bus request and bus grant signals. The circuit is outlined in Figure 3.3. The arbiter consists of a *priority register* and a *bus control* unit that controls the timing of a memory cycle. The priority register is made up of a PAL16L8A (u16) and a 74AS573 octal transparent latch (u15). The bus control signals are generated by a finite state machine (FSM) built from two PAL16R8As (u13, u14). A detailed description of the bus control FSM including a state diagram is contained in appendix B. Note that the state machine is clocked by the fast clock (f = 20 MHz) in order to achieve higher granularity.

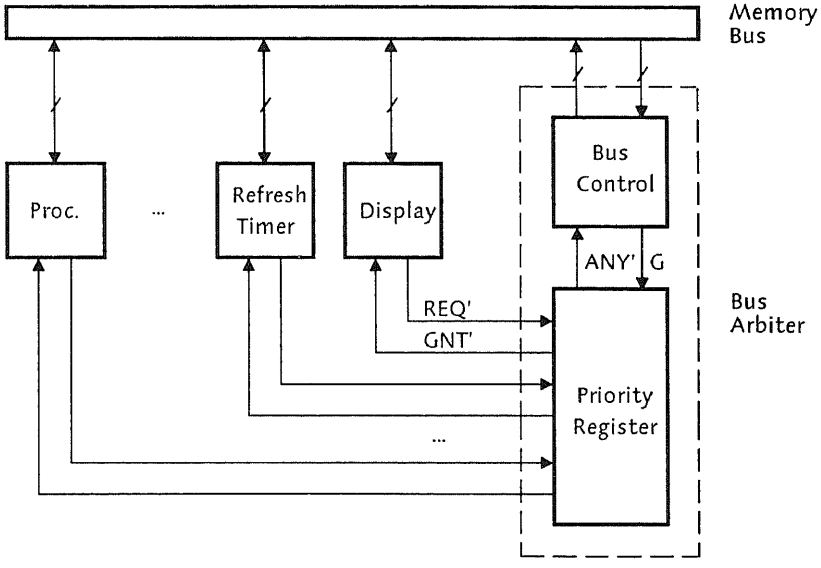


Figure 3.3 Memory bus arbiter.

The sequence of events during a read and a write memory cycle is shown in Figure 3.4. A full speed memory cycle is performed in four cycles of the processor clock CLK, labeled T1 through T4. Clock cycles not associated with a memory cycle are designated Ti (for "idle"). In order to acquire control of the bus (i.e. to become bus master), the device asserts its bus request signal that is fed into the priority register. The highest-order data applied at a request input is transferred to the appropriate grant output. If any request has been submitted to the priority register, ANY' becomes low, thereby informing the bus control FSM that a memory cycle has to be started. The FSM responds with a low G signal causing the state of the request lines to be latched by the priority register. At the end of a memory cycle,

the signal CLR.REQ' clears the processed request. The following bus master devices are provided (listed in descending priority):

DSP.REQ'	DSP.GNT'	Display refresh controller
REF.REQ'	REF.GNT'	DRAM refresh timer
REQ0'	GNT0'	not used
REQ1'	GNT1'	not used
REQ2'	GNT2'	not used
REQ3'	GNT3'	not used
CPU.REQ'	CPU.GNT'	Processor

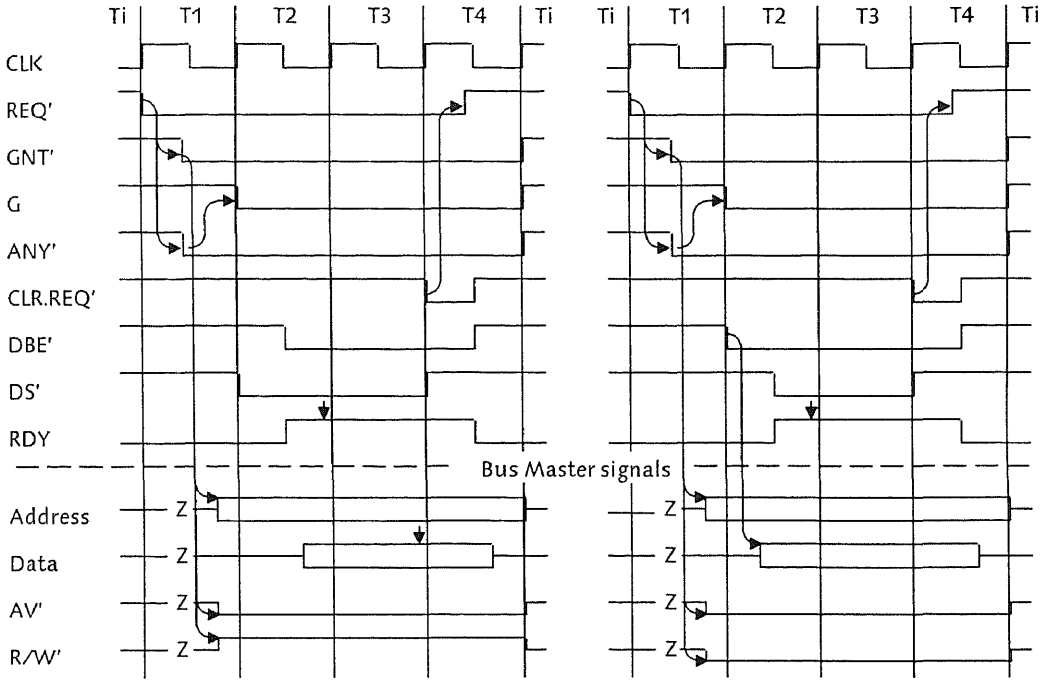


Figure 3.4 Read and write cycle timing.

The bus control FSM provides further control signals which are specifically introduced to suit the NS32000 processor chips, but are general enough to serve other master devices as well. The data buffer enable signal DBE' is used to control the data bus buffers. The leading edge of DBE' is delayed a half clock period during read cycles to avoid bus conflicts, for example, between data buffers and either the CPU or the MMU. DBE' goes inactive in the middle of T4, having provided for necessary data hold times. If the processor is performing a read cycle, the data bus is sampled at the end of T3. The data strobe DS' signals the beginning of a data transfer. This signal is used by the control circuitry for dynamic RAMs. The leading edge of DS' is delayed a half clock period during write cycles to guarantee the appropriate data setup time for the DRAMs. DS' returns to the high level at the beginning of T4. During a write cycle, the processor presents data starting at the beginning of T2 and ending at the end of T4.

To allow sufficient strobe widths and access times for any speed of memory or peripheral device, the bus control FSM provides cycle extension. Note that the arbitrated memory bus does not allow the use of the cycle extension capabilities of the TCU. The FSM uses the following wait input signals (listed in descending priority):

- a low IO.EN' during T2 causes the FSM to perform a so-called peripheral cycle, which is characterized by four added wait states (TW4, TW3, TW2, TW1). In addition, a read or write strobe signal (IO.RD', IO.WR') is generated which meets the setup and hold timing requirements of slower peripherals. IO.RD' and IO.WR' are decoded from R/W'
- if WAIT2' is sampled low during T2 two wait states (TW2, TW1) are inserted
- if WAIT1' is sampled low during T2 one wait state (TW1) is inserted
- CWAIT' initiates a continuous wait. As long as sampled low during T2 and TW1 one wait state (TW1) is inserted

Examples of cycle extension are shown in Figure 3.5. The processor is informed of an extended bus cycle by means of the RDY signal. At the end of T2, the RDY signal is sampled by the CPU or MMU. If RDY is high, the next T-states will be T3 and then T4 ending the bus cycle. If RDY is low, then another T3 state will be inserted after the next T-state, and the RDY line will again be sampled during the next T-state.

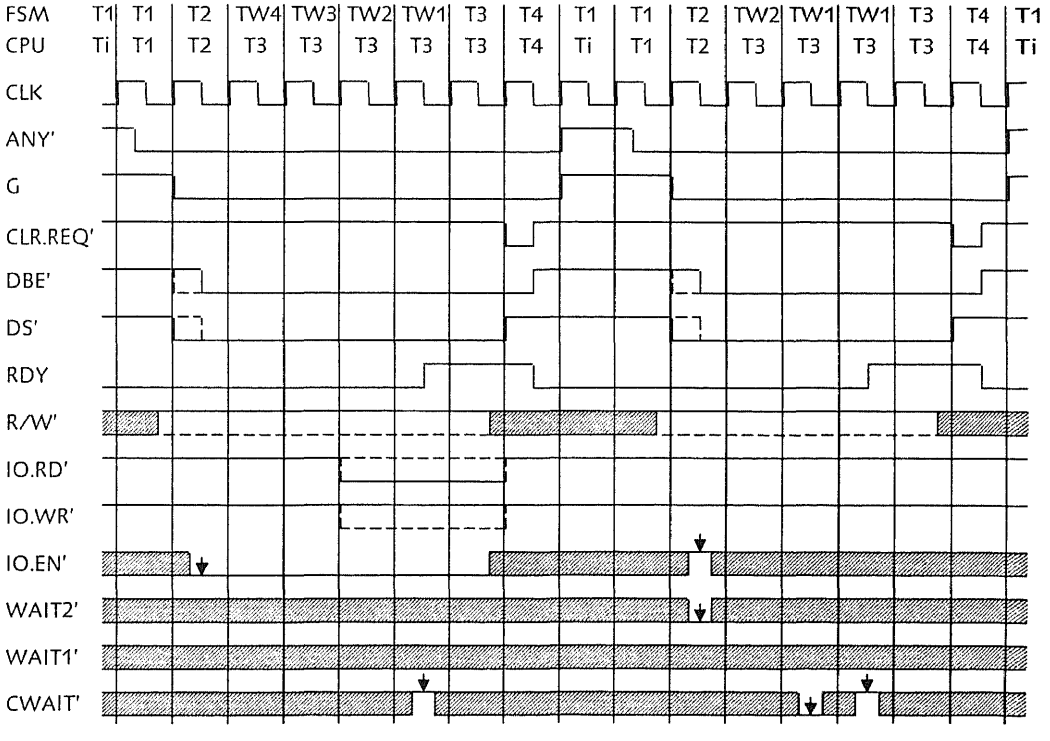


Figure 3.5 Cycle extension.

Although the processor has the lowest-order priority and thereby looses competition with any other bus masters, it is treated in a privileged way. Whenever no other master requests the bus, the processor is given control over the memory bus by default; as a result, there is no arbitration delay in case of a memory access by the processor.

The introduction of the address valid signal AV' is necessary for the following reasons. Since the processor also controls the memory bus during idle times, AV' is used to indicate a valid memory address during a memory bus cycle. Furthermore, bus masters such as the refresh timer for the DRAMs request so-called "dummy" bus cycles in order to prevent other devices to access the memory bus. AV' is then set to inactive, preventing any bus slave to decode the address.

The mentioned refresh timer is placed on the processor board. It is assumed that a central timer is responsible for refreshing all dynamic memory devices. The refresh timer consists of a 74LS393 dual 4-bit counter (u45) which divides the system clock CLK by 160. The refresh request line REF.REQ' is, therefore, asserted every 16 us. A memory refresh cycle is indicated by a low RFSH' signal.

3.1.3 Boot ROM and Standard Input/Output Devices

The *boot ROM* and several *standard IO devices* are also found on the processor board. Part of the address space is assigned to IO ports. This strategy is called memory-mapped IO with devices residing in the reserved IO address space loosely called IO devices. An *address decoder* provides the appropriate select signals. As can be seen in Figure 3.6 the IO devices include:

- a dual universal asynchronous receiver/transmitter (*UART*) that interfaces the serial keyboard and offers an additional RS-232-C serial port
- a dual-channel serial communications controller (*SCC*) providing two RS-485 serial interfaces
- a *mouse interface*
- a battery-backed real time clock (*RTC*)
- a *DIP-switch* holding system parameters
- an interrupt control unit (*ICU*) supporting up to eight interrupt sources

The width of the 32-bit data bus is not fully used by the peripheral devices. Their data paths are 4, 8, or 16 bits wide. The data bus interfaces are aligned with the 32-bit data bus using the lower-order data bits. An 8-bit peripheral unit, for example, is connected with data bits D0-D7; therefore, device register addresses are double-word addresses (i.e., address bits A0 and A1 are ignored).

A PAL20L8A (u12) and a 74ALS138 3- to 8-line decoder (u57) implement the *address decoder*. The PAL device provides the ROM and IO device enable signals ROM.EN' and IO.EN'. The reserved memory locations for ROM and IO devices are shown in Figure 3.7. To simplify future IO expansions, IO.EN' is also available on the memory bus. This signal further causes the arbiter to perform an extended, peripheral cycle. For the two uppermost 512 Byte-sized IO pages, additional select signals are generated (IO.PG0', IO.PG1'). The ICU resides in the uppermost IO page (IO.PG0'). This is required by the fact that the CPU reads the interrupt

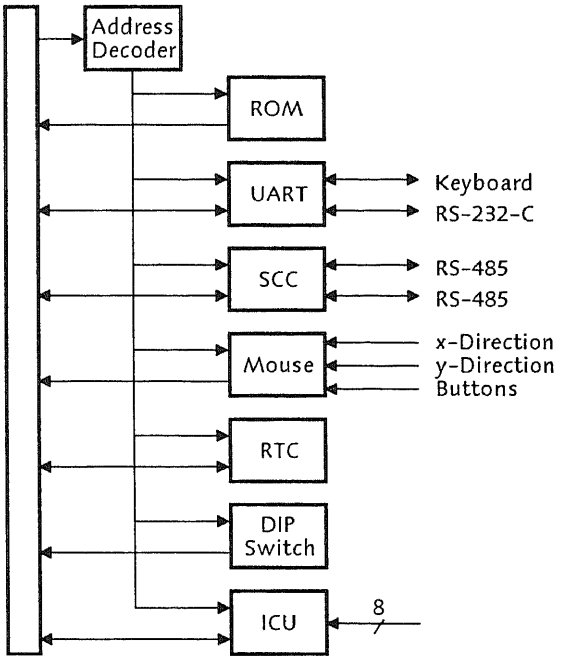


Figure 3.6 Boot ROM and standard IO devices.

vector from the fixed address FFFE00 (hex) [7]. The ICU chip select signal (ICU.CS') must not be activated when the CPU reads a dummy byte from address FFFF00 (hex) during a nonmaskable interrupt sequence; therefore, ICU.CS' is disabled if A8 is high (u62d). The next lower IO page (IO.PG1') is reserved for the other standard IO devices. A 74ALS138 (u57) provides eight select signals each having an address range of 64 Bytes.

The *boot ROM* is made up of four EPROM devices (u41-u44). The corresponding sockets can be configured for different ROM types (2764, 27128, 27256, 27512) with a range in total memory capacity from 32 KBytes to 256 KBytes. 150 ns parts are required in order to avoid wait states. The ROM data outputs are connected to the memory data bus with four 74ALS541 octal unidirectional buffers (u29-u32). The ROM address inputs are connected to the address lines A2-A17 (double word address).

A SC2681 *UART* (u47) provides two independent, full-duplex, asynchronous receiver/transmitter channels with software selectable baud rates up to 38.4 Kbps [12]. One channel is used for the keyboard. The receive and transmit data signals of the keyboard interface (KB.TxD', KB.RxD') are TTL compatible; the other channel implements a RS-232-C interface. A standard RS-232-C line driver (75188, u59) and a line receiver (75189, u60) [25] is used to provide the data transmission and the most common modem control signals: TxData, RxData, Request to Send, Data Terminal Ready, Clear to Send, Data Carrier Detected and Data Set Ready. Also provided on the UART is a programmable 16-bit counter/timer. Individual interrupt signals are output by the UART for the keyboard interface (KB.INT'), the RS-232-C interface (UART.INT') and the counter/timer (UART.C/T). The UART's crystal

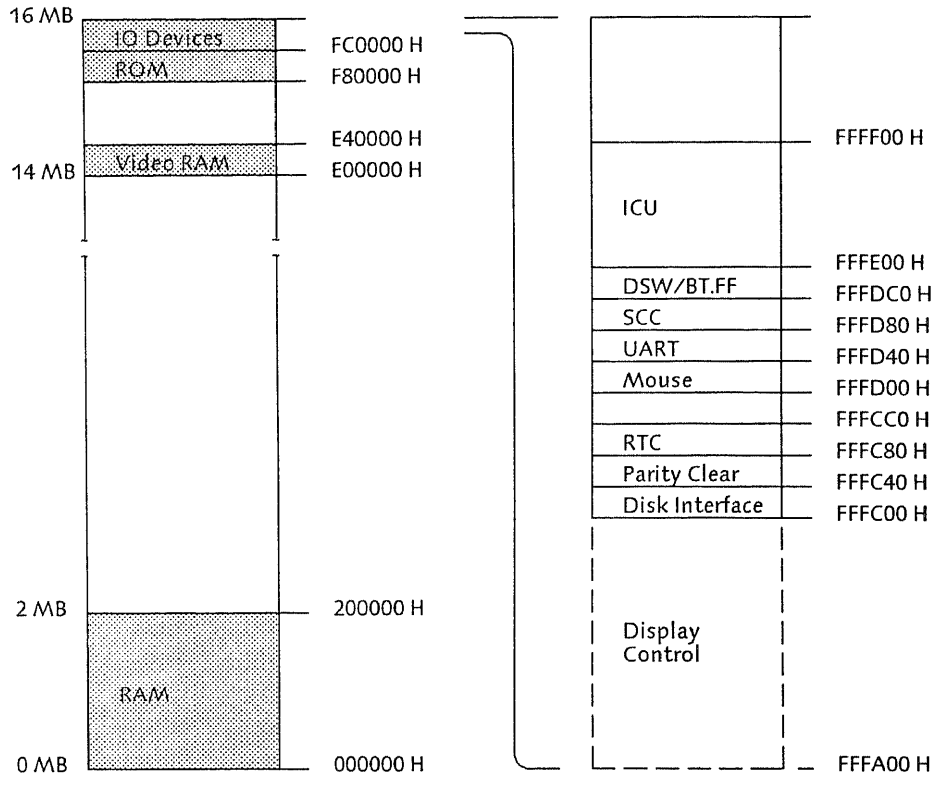


Figure 3.7 Memory map.

oscillator requires an external 3.6864 MHz crystal. A buffered version of this clock signal is also used by the SCC.

The Z8530 *SCC* (u50) is a dual-channel, multiprotocol data communication peripheral [13, 14, 15]. The SCC can handle asynchronous and synchronous formats including SDLC. In the latter case, data rates up to 230.4 Kbps are possible. Both channels constitute an RS-485 serial line interface using DS3696 high-speed differential 3-state line transceivers (u61, u64) [26]. The SCC's "request to send" output (RTSA', RTSB') defines the data transmission direction. The "clear to send" input (CTSA', CTSB') is used to detect a line fault condition (LFA', LFB'), which is reported by the transceiver in case of a bus contention or fault situations that cause excessive power dissipation within the device. The SCC requires an external 6 MHz clock oscillator (u51). The 3.6864 MHz clocking signals for the receiver/transmitter channels are derived from the UART's oscillator circuit.

The *mouse interface* keeps track of the relative mouse position and holds the state of the three mouse buttons. A direction discriminator controls the up/down counter for the x- and y-directions. The three switches can be directly read on a parallel port and polled by software. The mouse interface is composed of the following components. A 74ALS138 3- to 8-line decoder (u56) provides select signals for the x-register (RX'), y-register (RY') and button state register (RB'). All registers are read-only. The state of the mouse buttons

(MB0', MB1', MB2') is isolated from the data bus (D0-D2) by a 74ALS244 octal buffer (u54) which is enabled by RB'. For each direction the mouse generates two phase-shifted signals (MXA, MXB resp. MYA, MYB). This information is evaluated by the direction discriminator which is realized with a PAL16R8A (u55). This device generates the necessary control signals for the x- and y-counters. Each counter is made up of two cascaded 74F779 8-bit counter chips (u17-u20) [27]. A built-in 3-state IO port eases the data bus interface of the counters.

The M3002 *RTC* chip (u66) contains a time of day clock and a calendar [16]. The register address and data are multiplexed over four data lines; therefore, no separate address lines are needed. External components include a 32.768 KHz crystal for the on-chip oscillator and a battery back-up to keep time and date when no external power is supplied. Because of the low power consumption of this device, the lithium cell provided guarantees a lifetime of more than 10 years.

The *DIP-switch* (u27, u28) holds 8 bits of information (read-only). The off position corresponds to a logic 1. The switches can be used to set the processor configuration, the size of installed memory, or a machine number in a network.

The Am9519A-1 *ICU* (u52) accepts up to eight maskable interrupt request inputs, resolves priorities and supplies programmable response bytes for each interrupt [17, 18]. The latter feature allows the CPU to acknowledge interrupt requests in the so-called vectored mode, interpreting the ICU's response byte as a vector value. An additional circuit (u62) is needed to distinguish between a "normal" access to the ICU's register (icu.cs') and an interrupt acknowledge cycle (icu.inta'). The group interrupt output ICU.INT' is synchronized with the rising edge of TCU.CTTL (u22b) in order to minimize the possibility of metastable states [19]. The ICU inputs the following interrupt signals (listed in descending priority):

INT0'	counter/timer (UART.C/T)
INT1'	two RS-485 channels (SCC.INT')
INT2'	RS-232-C interface (UART.INT')
INT3'	disk controller (DK.INT')
INT4'	keyboard (KB.INT')
INT5'	real time clock (RTC.INT')
INT6'	not used
INT7'	not used

Interrupt lines INT4'-INT7' are available on the backplane bus. In particular, INT6' and INT7' are provided for future IO device expansion.

The address decoder further generates the signals BT.CS' and PAR.CLR'. Any write access to an IO address assigned to BT.CS' clears the boot flip-flop (u40a). The parity error flag is reset during a hardware reset (u34c) or by accessing an address assigned to PAR.CLR' (u39c).

3.2 Memory Board

The Ceres memory board contains 2 MBytes of dynamic memory and is populated by 72 DRAM devices organized with a 36-bit wide data bus which allows for 32 bits of data plus byte parity. The memory is designed to accept 256 Kbit 120 ns dynamic RAM chips which

operate with the processor at 10 MHz with no wait states. Memory can be expanded by additional memory boards.

The organization of the memory is shown in a block diagram (Figure 3.8). In addition to the *memory array*, the following components are needed:

- the *board selection* logic allows the board to be activated for different address ranges
- the *memory control* logic takes care of the proper sequence of a memory cycle. Derived from the original address, the row and the column address together with the appropriate row and column address strobe signals are generated successively. Periodically the memory control logic is forced to refresh the dynamic memory chips. The content of a refresh counter is then sent as the address to the memory array
- the *error detection* unit generates parity bits (write-cycle) and checks the read information (read-cycle)
- the bidirectional *data line buffers* connect the data paths of the memory array and the memory bus

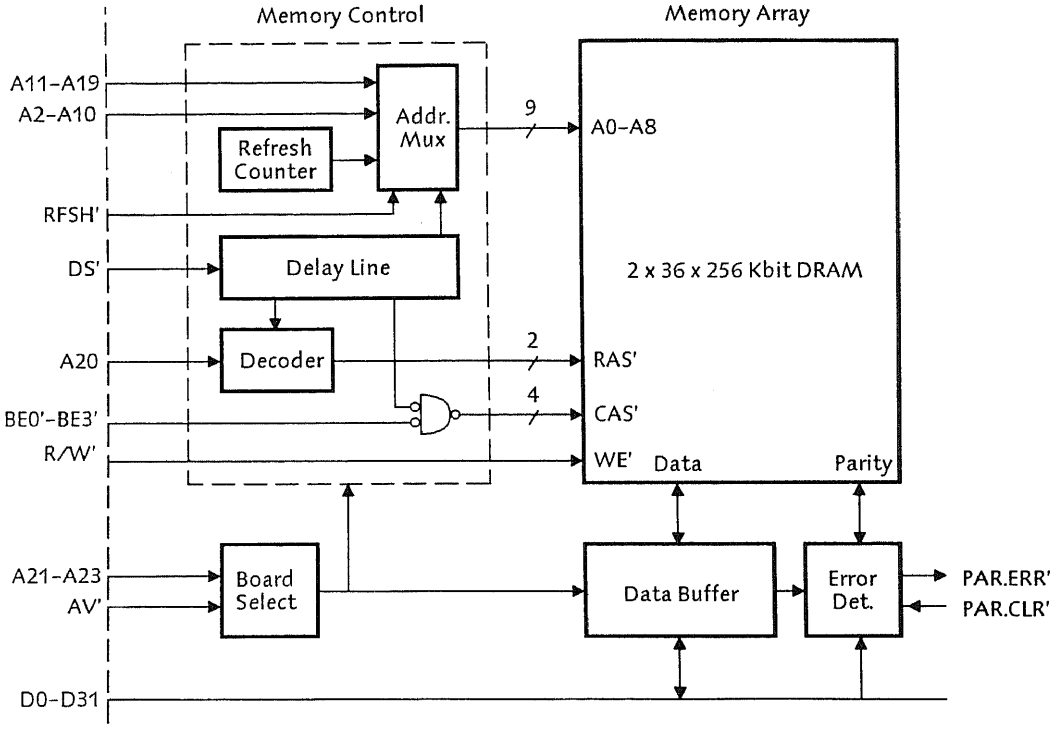


Figure 3.8 The 2 MByte dynamic memory.

The details of the implementation are explained in greater detail in the following section.

The *memory array* is divided into two banks each consisting of 36 DRAM devices. Address lines A2-A20 provide a double word address. Individual byte accessing is controlled by the byte enable signals BE0'-BE3'.

The *board selection* logic uses a 74ALS138 3- to 8-line decoder (u10). If the signal AV' indicates a valid address, the three most significant address bits A21-A23 are decoded and assign an address range of 2 MBytes to each decoder output. One of these is chosen as the board select signal MCS' by closing the appropriate jumper (u9).

Most functions of the *memory control* logic are provided by the DP8419 DRAM controller (u12) [20]. The higher order address bits A11-A19 serve as the row address and the lower order address bits A2-A10 as the column address. These addresses are output sequentially on the address lines a0-a8 which drive the memory devices. The address strobe signals RAS' and CAS' are generated by an internal delay line induced by the signal DS'. A timing diagram is shown in Figure 3.9. Individual control lines for each memory bank (RAS0', RAS1') and for each byte (CAS0'-CAS3') are generated using the signals A20 resp. BE0'-BE3'. A20 is used as an input to the internal bank decoder of the DRAM controller. BE0'-BE3' ORed with CAS' yield CAS0'-CAS3'. The write enable signal WE' is a buffered version of the bus signal R/W'. The DRAM controller uses high output current drivers for all address and control lines. External damping resistors reduce both overshoot and undershoot on these signal lines caused by the high capacity load of the memory devices. The DRAM controller performs a "normal" memory cycle, if the CS' input driven by the MCS' line is activated and the mode input M0-M2 is set to the so called auto access mode. This implies that the signal RFSH' must be inactive. If RFSH' is active, a refresh memory cycle takes place. The state of the address lines A2-A23 including MCS' is not relevant in this mode.

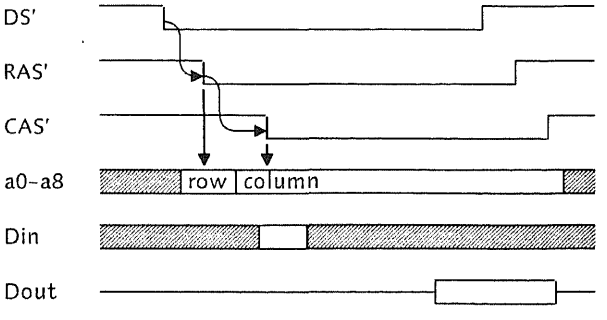


Figure 3.9 DRAM timing.

Error detection and *data line buffers* are made up of four Am29833 9-bit parity bus transceivers (u1-u4) [28]. The bidirectional tristate buffers need separate output enable signals for each direction. They are obtained through combination of the signals R/W', DBE' and MCS' (u6). The error detection circuit contains a parity generator and checker. The result of the parity checker is latched in an internal flip-flop which is triggered by the trailing edge of the signal DS' at the end of a read cycle (u6c). In case of a parity error, the open-collector output signal PAR.ERR' becomes active. The flip-flop can be reset by the signal PAR.CLR'. PAR.ERR' is connected with the nonmaskable interrupt signal of the CPU.

3.3 Display Controller Board

The design of the display refresh controller has mainly been influenced by the use of so-called video RAM devices which have been developed specifically for video applications. The multiport video RAM combines a standard 64 Kbit DRAM with an on-chip 256 bit shift register and the necessary controls to transfer data between the memory array and the shift register. The two ports (that is the memory array and the shift register) can be accessed simultaneously except during a data transfer between the array and the register.

The display controller board houses 256 KBytes of *display memory* and the *display refresh controller.* The display memory is made up of 32 VRAM chips organized with a 32-bit wide data bus as seen from the processor and a 8192-bit wide video data bus as seen from the display refresh controller. 64 Kbit 150 ns video RAM chips [21] are used which are accessed with one additional wait state. The display memory accommodates two bitmaps that can be displayed alternatively.

3.3.1 Display Memory

The organization of the display memory (Figure 3.10) is very similar to that of the memory board (Figure 3.8) with the exception of the error detection circuit which has been omitted. The following explanation of the implementation therefore concentrates on the differences.

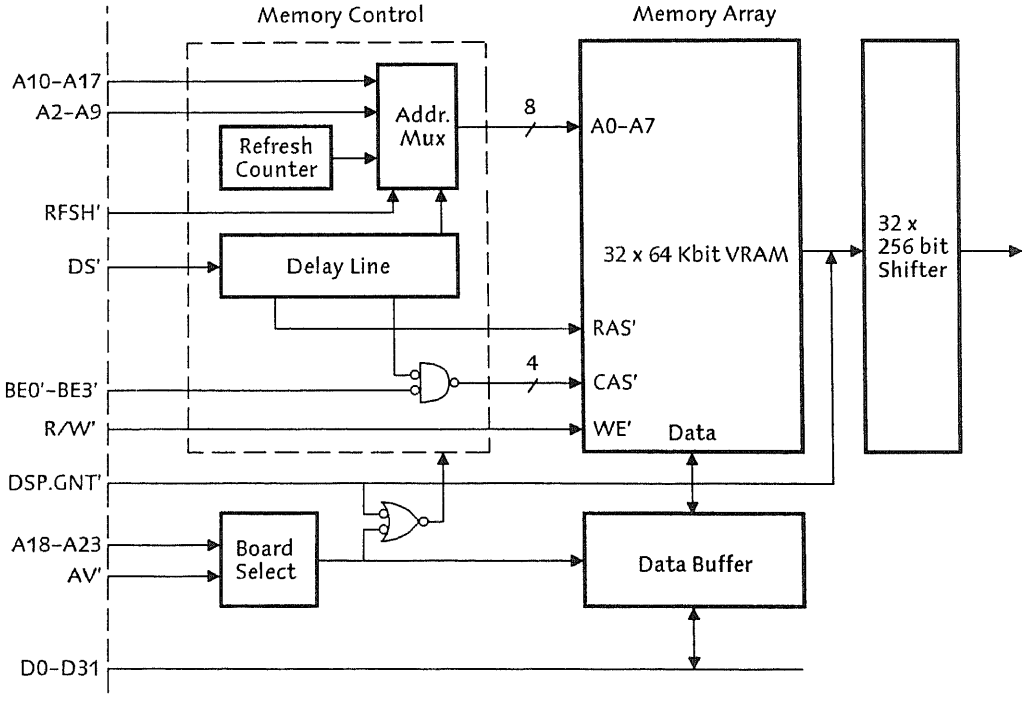


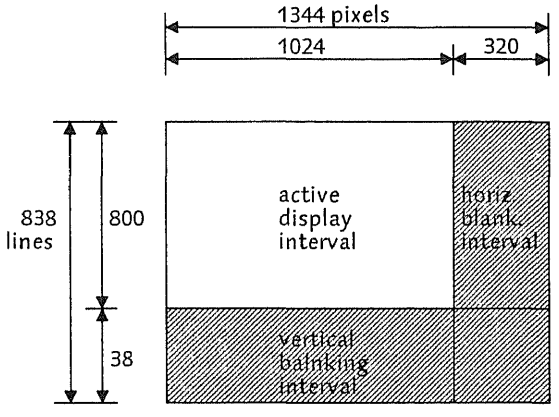
Figure 3.10 The display memory.

The *memory array* contains one bank only. The *board selection* logic is done with a PAL16L8A device (u7). Using address bits A18-A23, the decoder assigns the address range at E00000-E3FFFF (hex) to the display memory. When the display refresh controller accesses the display memory (DSP.GNT'=0) the address decoder is omitted (u11a). Since in this case no data are transferred on the memory bus, the data line buffers are not enabled. The *memory control* logic is again realized with a DP8419 DRAM controller (u8). Because of the smaller address range, fewer address lines are needed (A2-A17). The *data line buffers* are made up of four 74ALS645 8-bit bus transceivers (u1-u4).

3.3.2 Display Refresh Controller

In order to better understand the display refresh controller, the display parameters of the high resolution CRT-monitor are explained first (Figure 3.11). As can be seen from the illustration, the total frame time consists of the active display interval, the horizontal blanking interval (horizontal retrace), and the vertical blanking interval (vertical retrace). The pixel clock frequency is the product of pixels per line, lines per frame, and frames per second:

$$f(pixel) = 1344 * 838 * 62.15\ s^{-1} = 70\ MHz$$



Pixels per line	1344
Lines per frame	838
Displayed pixels per line	1024
Dislpayed lines	800
Frames per second	62.15
Pixel clock frequency	70 MHz
Pixel time	14.3 ns
Data transfer rate	6.4 MByte/s

Figure 3.11 The display parameters of the high resolution CRT-monitor.

The structure of the *display refresh controller* is shown in another block diagram (Figure 3.12). The following blocks can be distinguished:

- the *clock generator* circuitry provides the clock signals for the video shifter and the horizontal and vertical counters
- the *horizontal and vertical counters* keep track of the position of the displayed picture element (=pixel). Derived from the counters state, control signals such as the ones for the synchronization of the display monitor are generated; furthermore, the counters determine the memory array address of those display lines which have to be refreshed next
- the *display memory*, which is arranged as a 3-dimensional array of 32 memory devices organized as 256 words of 256 bits

- the *video shift register* transforms the data which are transmitted by the VRAMs as 32-bit entities into the bit-serial video data signal

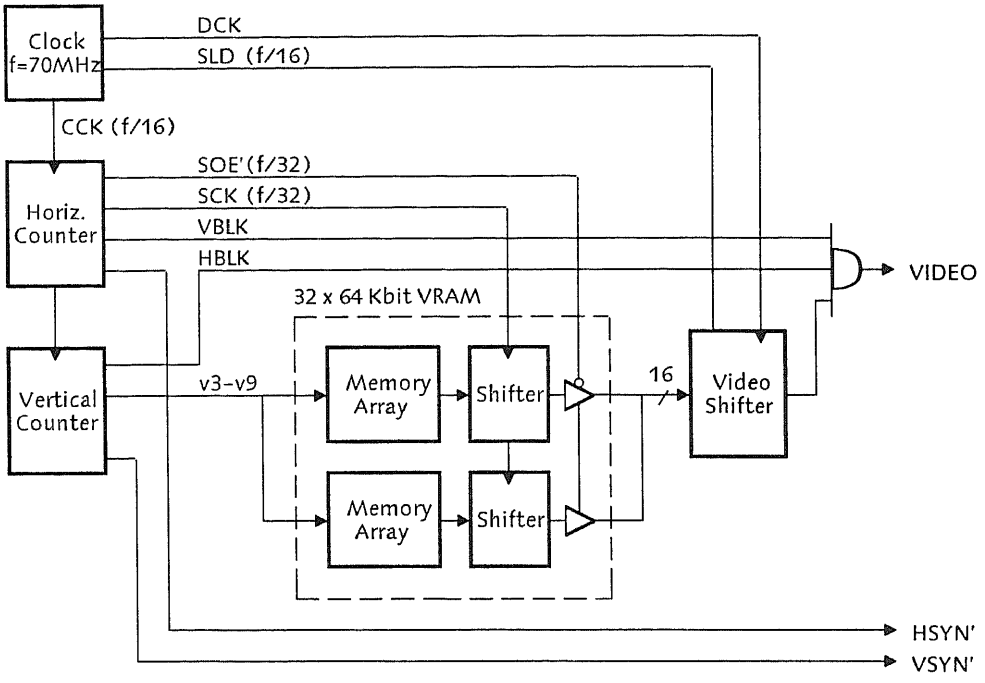


Figure 3.12 The display refresh controller.

Not shown in the block diagram is the *display control register* and the *memory cycle request* circuitry needed to gain access to the "video port".

The *clock generator* circuitry uses a hybrid 70 MHz oscillator chip (u29). Its output provides the pixel or dot clock DCK. A 74AS163 4-bit counter (u27) acts as a divider of the dot clock frequency. It produces the clock signal CCK for the horizontal counter and the load signal SLD for the video shift register. The timing relationship of these signals is shown in Figure 3.13.

The *horizontal resp. vertical counter* is made up of two resp. three 74ALS163 4-bit counters (u17, u19, u20, u30, u32), a 27C64 EPROM (u18, u21), and a 74F378 6-bit latch (u22, u33). Horizontally, the counter represents the pixel position divided by 16, while the vertical counter state corresponds to the line position. The two EPROMs generate the waveforms of the horizontal and vertical control signals based on the counters state (Appendix A.3). The following signals are provided:

- HBLK and VBLK deactivate the video outputs during the horizontal and vertical blanking intervals
- HSYN' and VSYN' are responsible for the line and frame synchronization of the video beam

- HRQ and VRQ cause the VRAM shift register to be reloaded with a new bitmap block every 8 display lines, thereby allowing the counter output signals v3-v9 to be used as the bitmap block address
- VCK is the clock signal of the vertical counter
- HCLR' and VCLR' initialize the horizontal and vertical counters to the zero state at the end of a line resp. frame

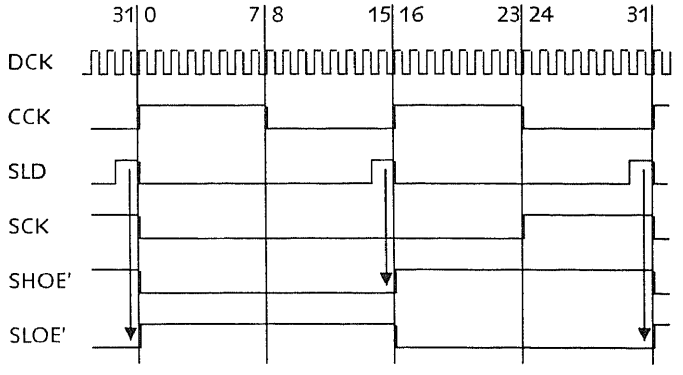


Figure 3.13 Clocking signals of the display refresh controller.

The horizontal counter also controls the clock signal SCK and the output enable signals SLOE' and SHOE' of the VRAM shift registers. The timing relationship can be seen in Figure 3.13. The 16-bit video shift register is loaded, alternating with the lower and the higher 16 bits of the VRAM shift register data outputs.

The *video shift register* is realized with a 74F676 16-bit shift register (u24) [27]. Its output, the serialized video data SOUT and the blanking signal BLK, first have to be synchronized with the dot clock DCK (74AS175 quad D-FF, u26) before SOUT can be masked by BLK (u0). The display control register provides the signal INV which, when set to 1, inverts the video data signal (u0).

In order to access the display memory, the display refresh controller periodically requests a memory cycle from the bus arbiter. The *memory cycle request* flip-flop (u14) asserts the DSP.REQ' line when the horizontal and vertical counters activate MRQ (MRQ = VRQ AND HRQ, u23c) and the display control register bit DSP.EN is set to 1. Another flip-flop is needed to synchronize the request signal with the system clock CLK. A granted memory cycle is indicated by an active DSP.GNT' signal. At the end of a memory cycle, the signal CLR.REQ' clears the request.

DSP.GNT' serves as the output enable of two 74ALS541 octal buffers (u5, u6) which gate the address and control signals to the memory bus. The address is made up of the vertical counter outputs v3-v9 and the display control register output a17. Signals v3-v9 define the display line that has to be scanned next; a17 determines in which half of the display memory the displayed bitmap is located. This address information is directed to the address lines A10-A17 which define the memory row of the VRAM that is to be loaded into the internal shift register. If the two address bits A8 and A9 equal 00 during this register transfer cycle, a total of 256 bits can be sequentially read out (it is possible to transfer 64, 128, 192 or 256

bits of a memory row into the shift register). All other address bits are neglected (A0-A7, A18-A23). As address decoding now has to be inhibited (AV'=1), the display memory also has to be selected (DCS') when DSP.GNT' is active (u11a).

The signal T'/OE', that is input to the VRAMs, has two functions (see Figure 3.14). First, it selects either shift register transfer or random-access operation when RAS' falls; therefore, during a memory access of the display refresh controller, T'/OE' equals DSP.GNT', which is already low as RAS' falls. Second, if a random-access operation is performed, it functions as an output enable after CAS' falls. For this reason, it can then be identical to CAS' (u11b).

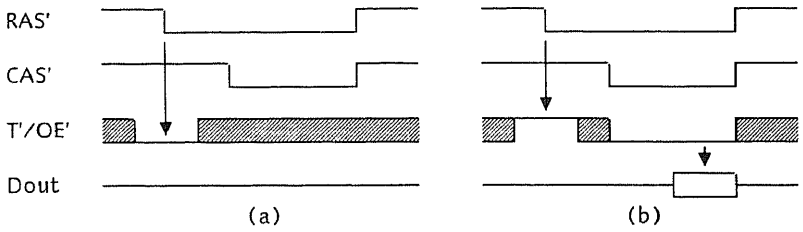


Figure 3.14 VRAM (a) shift register transfer and (b) random access operation.

The *display control register* is implemented with a 74ALS175 quad D-flip-flop (u12). The flip-flops are reset by a RESET' pulse. The address decoding is performed by half a PAL16L8A (u7). The register is located at FFFA00 (hex). The meaning of the three write-only bits is as follows:

Bit 0	0	Display Enable (initialized value)
	1	Display Disable
Bit 1	0	A17=0 (initialized value)
	1	A17=1
Bit 2	0	Normal Video (initialized value)
	1	Inverse Video

DSP.EN set to 0 (bit 0 of the control register) prevents any display requests. Nevertheless, the display is refreshed with the content of the VRAM shifters which will no longer be loaded. The shifter input signal SI now defines the video data signal. In order to guarantee a blank screen, SI is connected to INV (bit 2 of the control register). In the inverse mode SI is set to 1 in order to get an inverted video data signal of 0.

The design of a display refresh controller with a pixel frequency of 70 MHz requires special care. At a clock period of 14.3 ns gate delays of 5 ns are of considerable significance. The following provisions have been made:

- all registers generating critical signals are clocked by the same clock signal (u22, u33, u24, u26)
- synchronizers adjust different signal delays (u25, u26)

- all paths in a combinational circuit are of the same length (u0)

3.4 Disk Controller Board

The Western Digital WD1002-05 disk controller board [29] contains a Winchester interface (Seagate ST506 compatible) and a floppy interface (Shugart SA450 compatible). The Controller holds all of the logic required for a variable sector length (up to 1 KBytes), ECC correction, data separation, and host interface circuitry. The latter consists mainly of an 8-bit bidirectional parallel bus and appropriate control signals. Programmed IO is used to transfer sector data to and from an on-board sector buffer. Except for the board select signal DK.CS' and the interrupt request signal DK.INT, all signals of the host interface are "standard" memory bus signals. Additional circuitry for the signals DK.CS' (u57) and DK.INT (u63b) resides on the processor board.

3.5 Motherboard

Physical extensibility is obtained by placing the circuitry on several boards which are connected by a backplane (motherboard). The Ceres motherboard offers slots for six boards interconnected with a common, parallel backplane bus. Three slots are occupied by the already explained standard boards. Packaging flexibility is provided by requiring that the physical card position on the motherboard has no effect on the functioning of the system. This is accomplished by avoiding the use of daisy chain signals, which would require that there be no empty slots between boards and by having all signals independent of the backplane position. To avoid floating values pullup resistors are provided for the address and data signals. The backplane bus contains the following lines:

Address	A0-A23
Data	D0-D31
Control	
Data transfer	AV', BE0'-BE3', DS', DBE', R/W'
Bus arbitration	REQ0'-REQ3', GNT0'-GNT3', DSP.REQ', DSP.GNT', CLR.REQ'
Cycle extension	CWAIT', WAIT1', WAIT2'
IO Devices	IO.EN', IO.RD', IO.WR', DK.CS', DK.INT
Interrupts	INT4'-INT7'
Clock	CLK, FCLK
Miscellaneous	RESET', RESET.IN', RDY, ILO', PAR.ERR', PAR.CLR', RFSH'

The pin assignment of the bus lines is contained in appendix C.3.

4 Hardware Extensions

The modular, extensible multiboard arrangement invites not only the addition of more memory but, in particular, hardware which extends the versatility of the Ceres computer. The memory bus provides all necessary signals to either add new bus master or bus slave devices. In Figure 4.1, bus interfaces for both types are proposed. Note that the slave must be "synchronous" in the sense that it is always available and does not provide a completion signal.

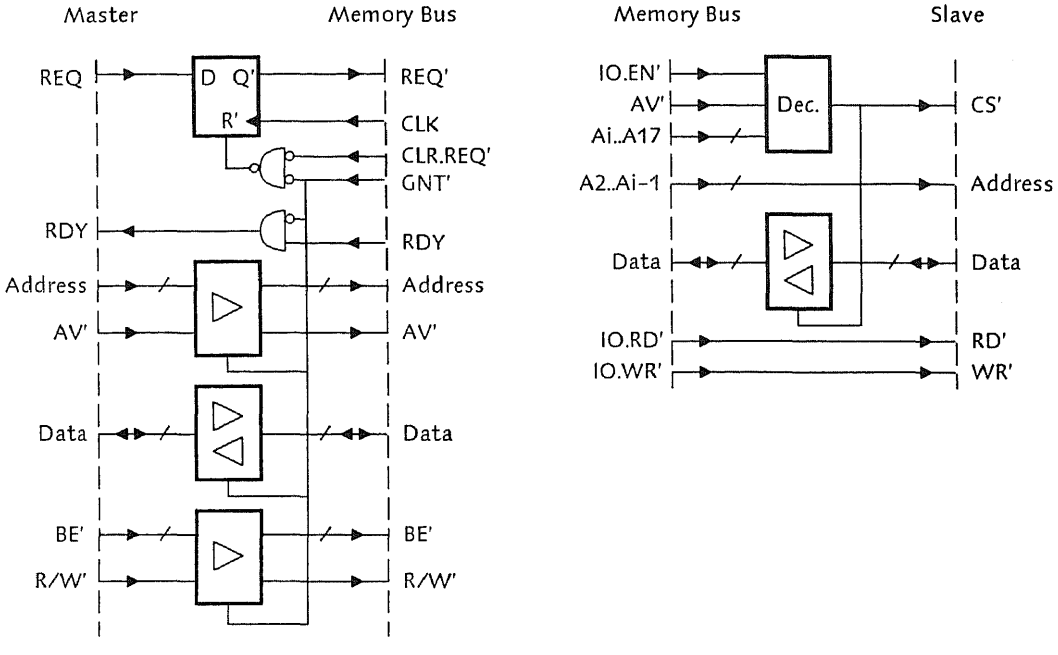


Figure 4.1 Interfaces for a bus master and a bus slave.

The timing specification of a basic memory cycle is presented in Figure 4.2. Based on a processor clock period of 100 ns, the unit of value is the nanosecond. For a peripheral cycle, four wait states are inserted between T2 and T3.

More information may be obtained from the circuit diagrams in appendix A. The pin assignments of the memory bus connectors are contained in appendix C.3.

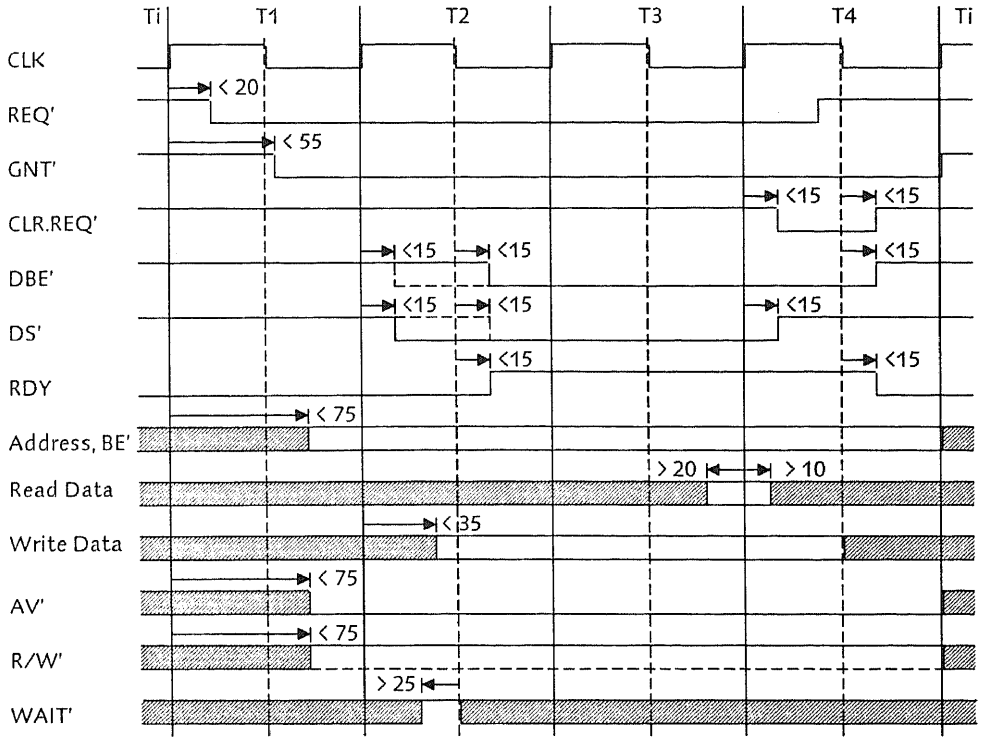


Figure 4.2 Timing specification of a basic memory cycle.

References

1. N. Wirth: Programming in Modula-2. (includes defining report)
Springer-Verlag, Heidelberg, New York, 1982.

2. R. Ohran: Lilith: A Workstation Computer for Modula-2.
Diss. ETH Nr. 7646, 1984.

3. N. Wirth: Microprocessor Architectures: A Comparison Based on Code Generation by Compiler.
Communications of the ACM, Oct. 1986, 978-990.

4. I. Berntsen: Benchmark Tests.
ETH Zürich, Institut für Automatik, Internal Report, 1986.

5. S.E. Knudsen: A Modula-2 Oriented Operating System for the Personal Computer Lilith.
Diss. ETH Nr. 7346, 1983.

6. N. Wirth: A Fast and Compact Compiler for Modula-2.
ETH Zürich, Institut für Informatik, Report No. 64, July 1986.

Data Sheets

7. NS32032-10 High-Performance Microprocessors.
National Semiconductor, October 1984.

8. Series 32000 Instruction Set Reference Manual.
National Semiconductor, 1984.

9. NS32082 Memory Management Unit.
National Semiconductor, November 1984.

10. NS32081-10 Floating-Point Unit.
National Semiconductor, November 1984.

11. NS32201-10 Timing Control Unit.
National Semiconductor, November 1984.

12. SCN2681 Dual Asynchronous Receiver/Transmitter (DUART).
Philips, May 1983.

13. Z8530 SCC Serial Communications Controller.
Zilog, August 1985.

14. Z8530/Z8030 Serial Communications Controller.
Technical Manual, Advanced Micro Devices, 1982.

15. Z8530 and Z8030 SCC Initialization: A Worksheet and an Example.
Zilog, September 1982.

16. M3002 Real Time Clock Circuit.
Microelectronic-Marin, 1984.

17. Am9519A Universal Interrupt Controller.
Advanced Micro Devices, 1980.

18. Am9519A Universal Interrupt Controller.
Technical Manual, Advanced Micro Devices, 1984.

19. NS32000 Series User Information.
National Semiconductor.

20. DP8419 High Speed Dynamic RAM Controller/Driver.
National Semiconductor, December 1984.

21. TMS4161 65536 Bit Multiport Memory.
Texas Instruments, July 1983.

Data Books

22. Advanced Low-Power Schottky/ Advanced Schottky.
Data Book, Volume 3, Texas Instruments, 1984.

23. PAL Programmable Array Logic.
Databooklet, National Semiconductor, 1983.

24. PAL Programmable Array Logic.
Handbook, Monolithic Memories, 1981.

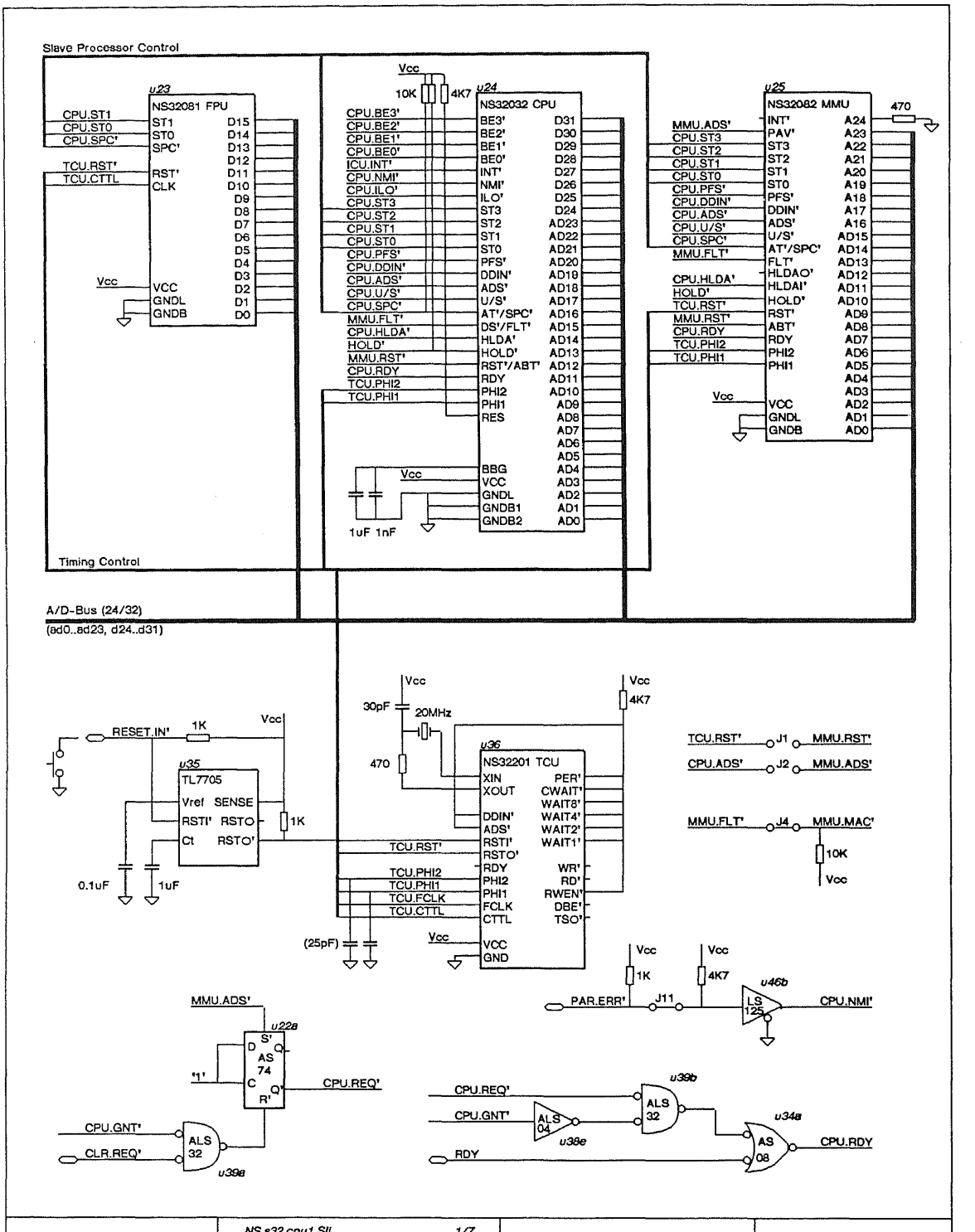
25. The Interface Circuits Data Book.
Texas Instruments, 1977.

26. Interface Bipolar LSI/ Bipolar Memory/ Programmable Logic.
Data Book, National Semiconductor, 1983.

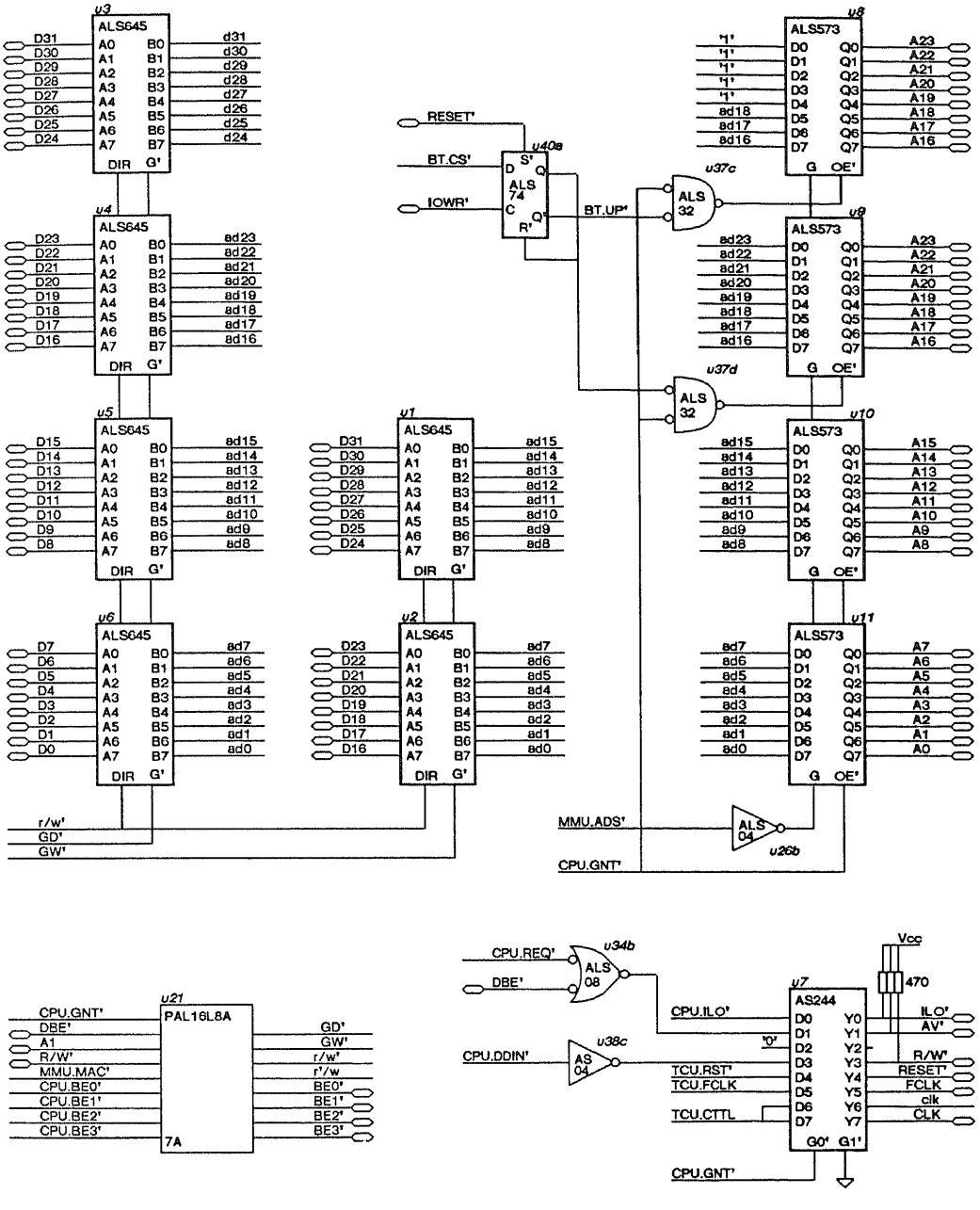
27. FAST TTL Logic series.
Book IC15N, Philips, 1984.

28. Bus Interface Product Specifications.
Databooklet, Advanced Micro Devices, October 1985.

29. WD1002-05/HDO Winchester/Floppy Disk Controller.
OEM Manual, Western Digital, July 1983.

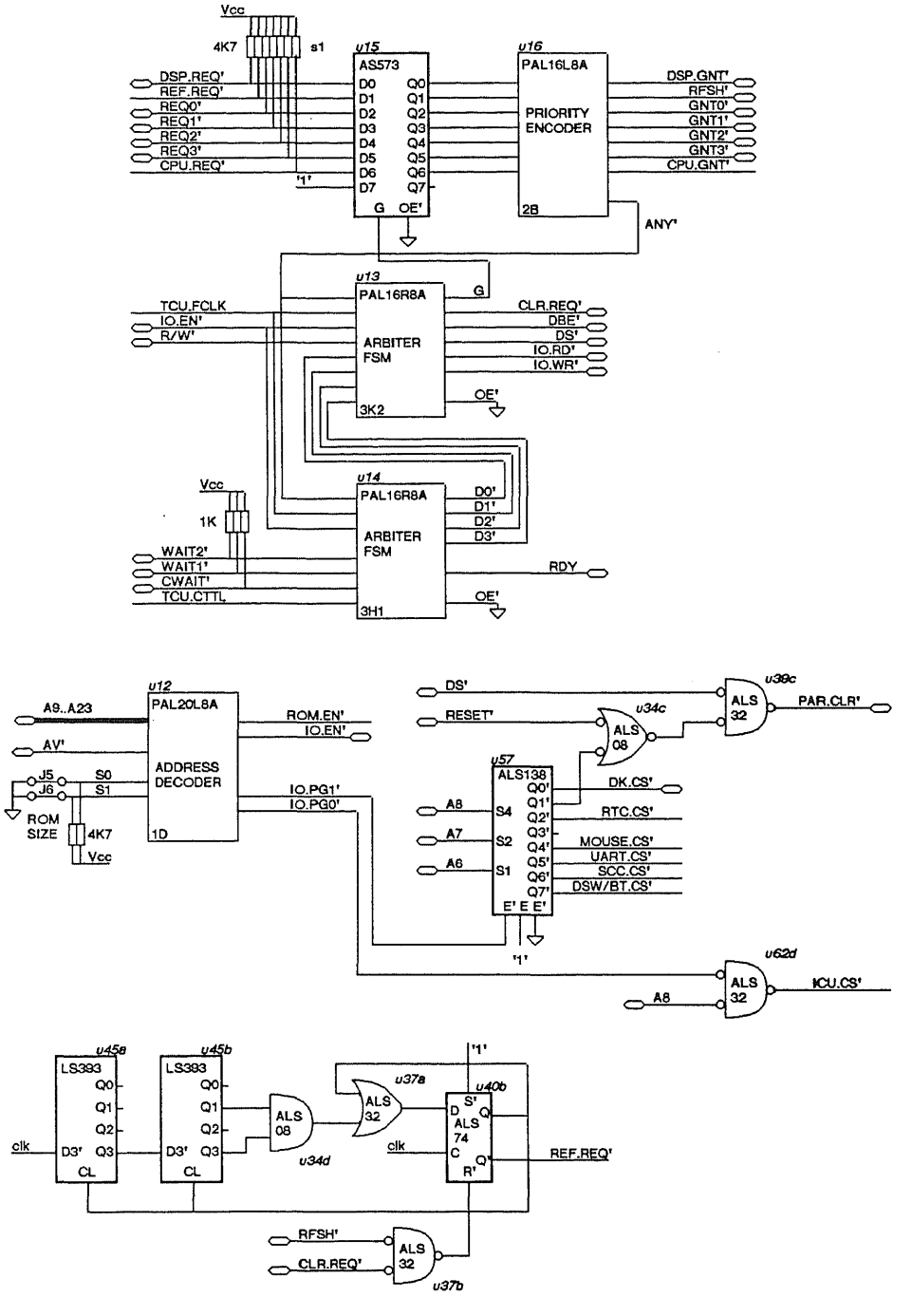


ETH Zurich	NS.s32.cpu1.SIL 1/7 Processor Cluster	Author: H.Eberle	Date: 3.7.85 REV. 3.12.85

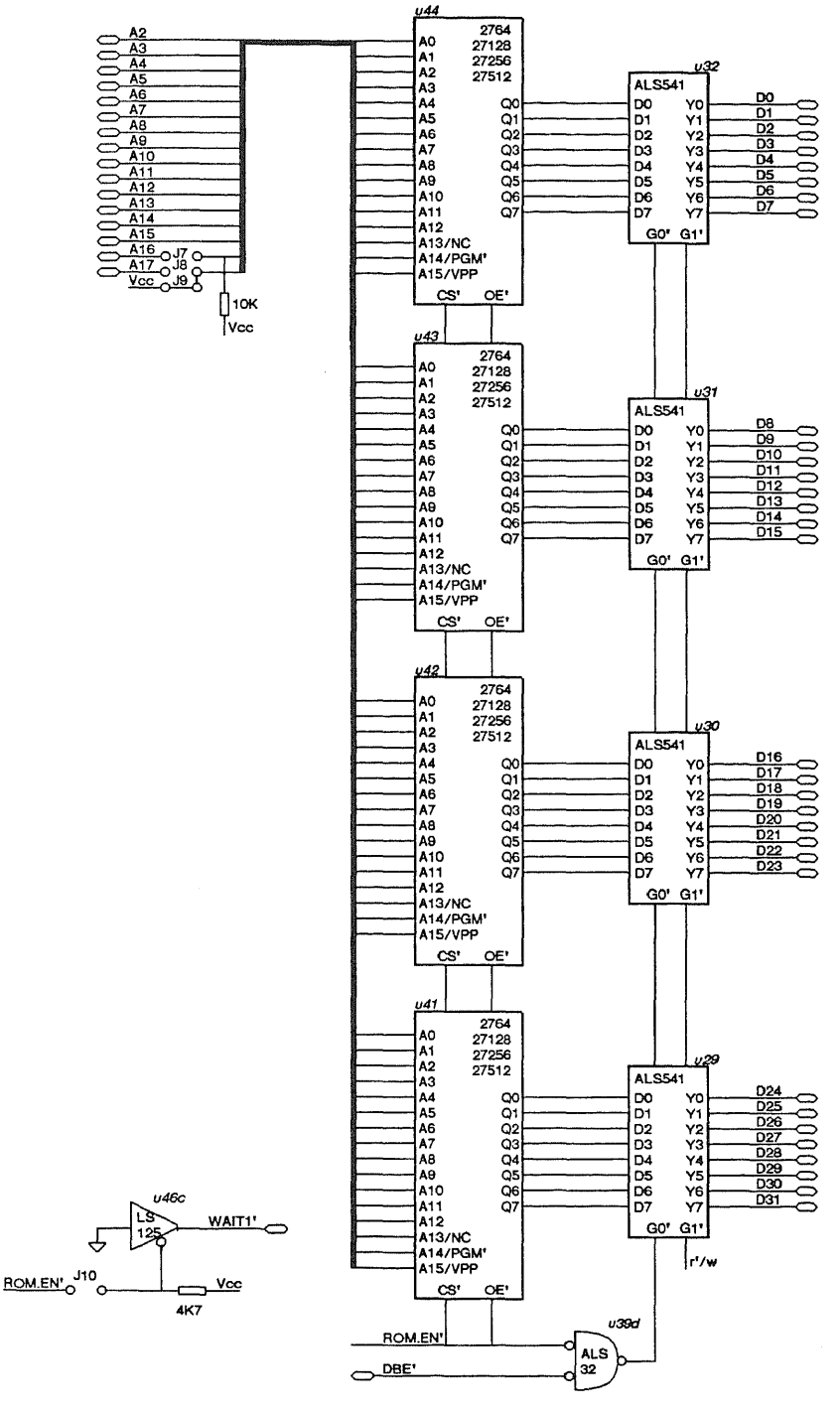


Termination resistors (270/560) are provided for TCU.FCLK, TCU.CTTL, FCLK, CLK and clk.

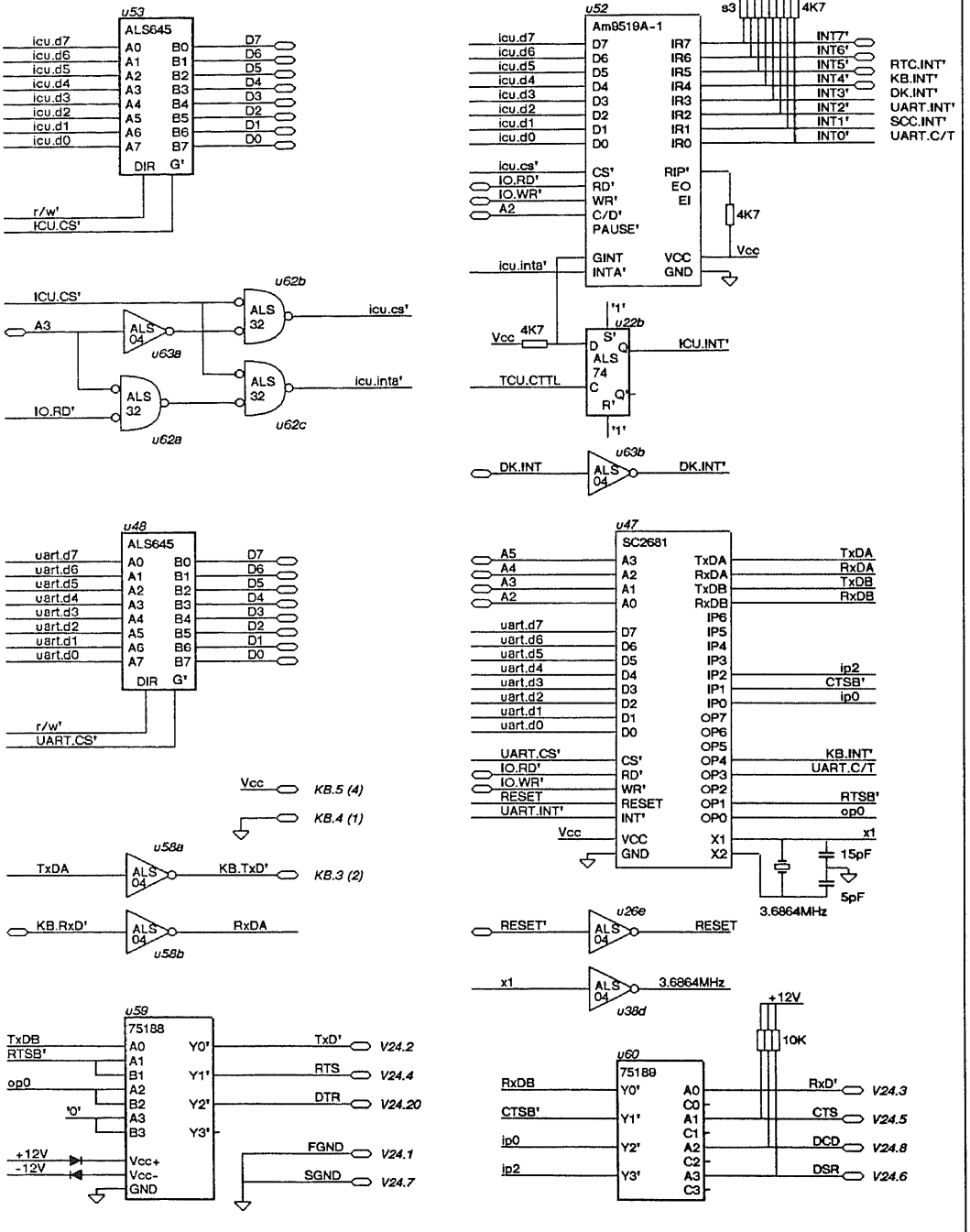
ETH Zurich	NS.s32.cpu2.SIL 2/7 Data Buffer, Address Latch Bus Control	Author: H.Eberle	Date: 3.7.85 REV. 21.5.86



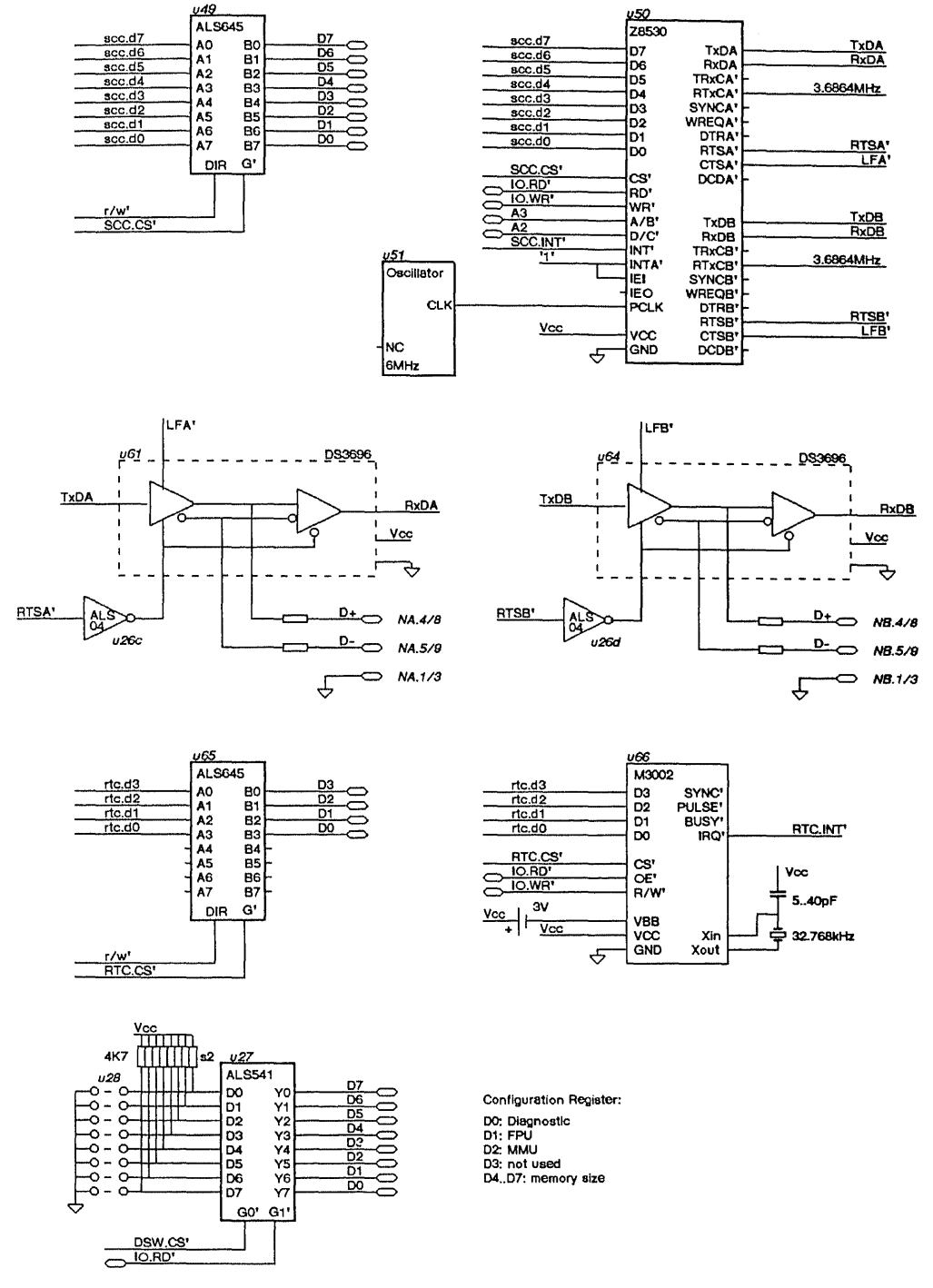
ETH Zurich	NS.s32.cpu3.SIL 3/7 Bus Arbiter, Addr. Decoder Refresh Timer	Author: H. Eberle	Date: 3.7.85



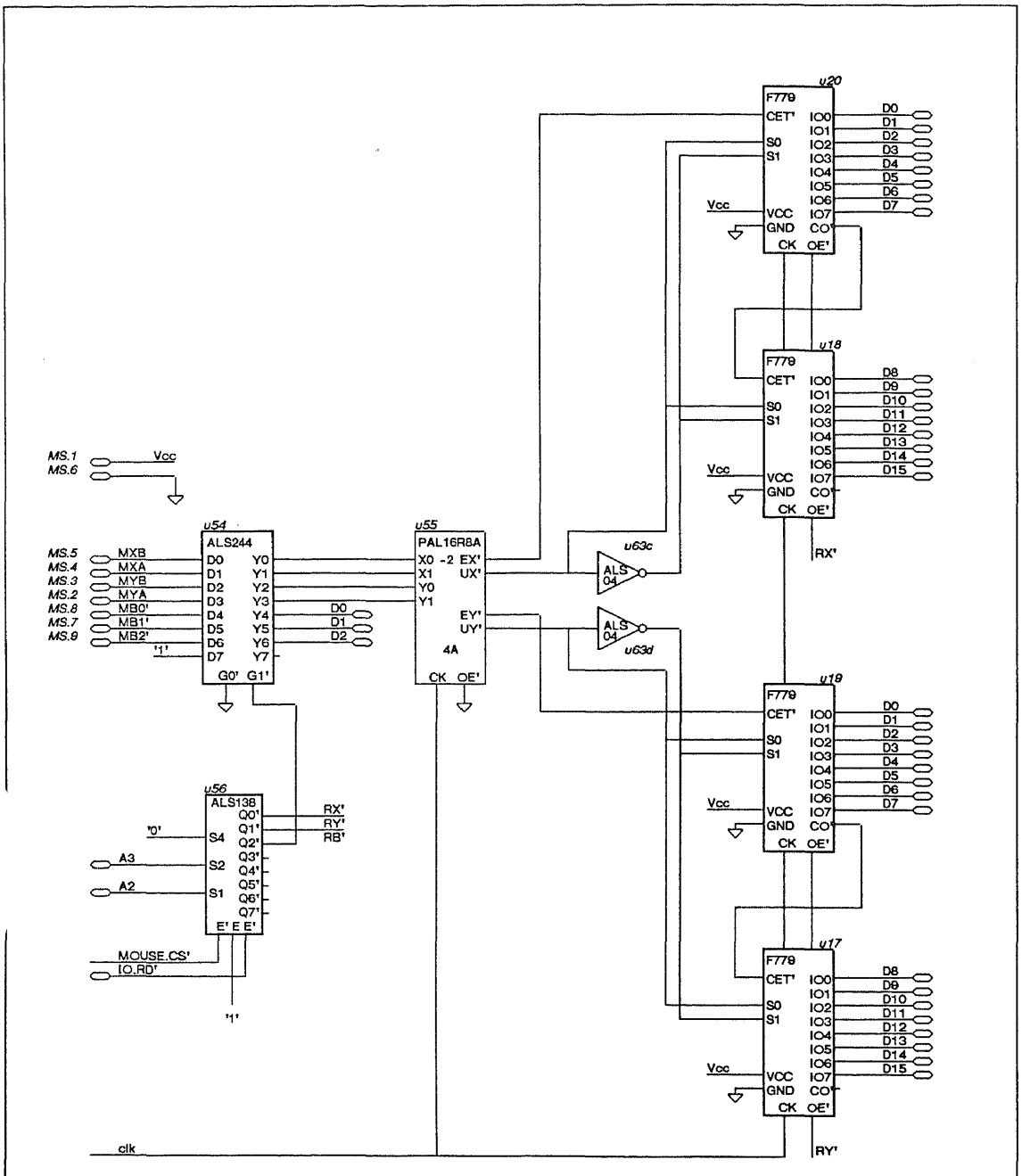
ETH Zurich	NS.s32.cpu4.SIL 4/7 ROM		Author: H.Eberle	Date: 3.7.85 REV 3.12.85



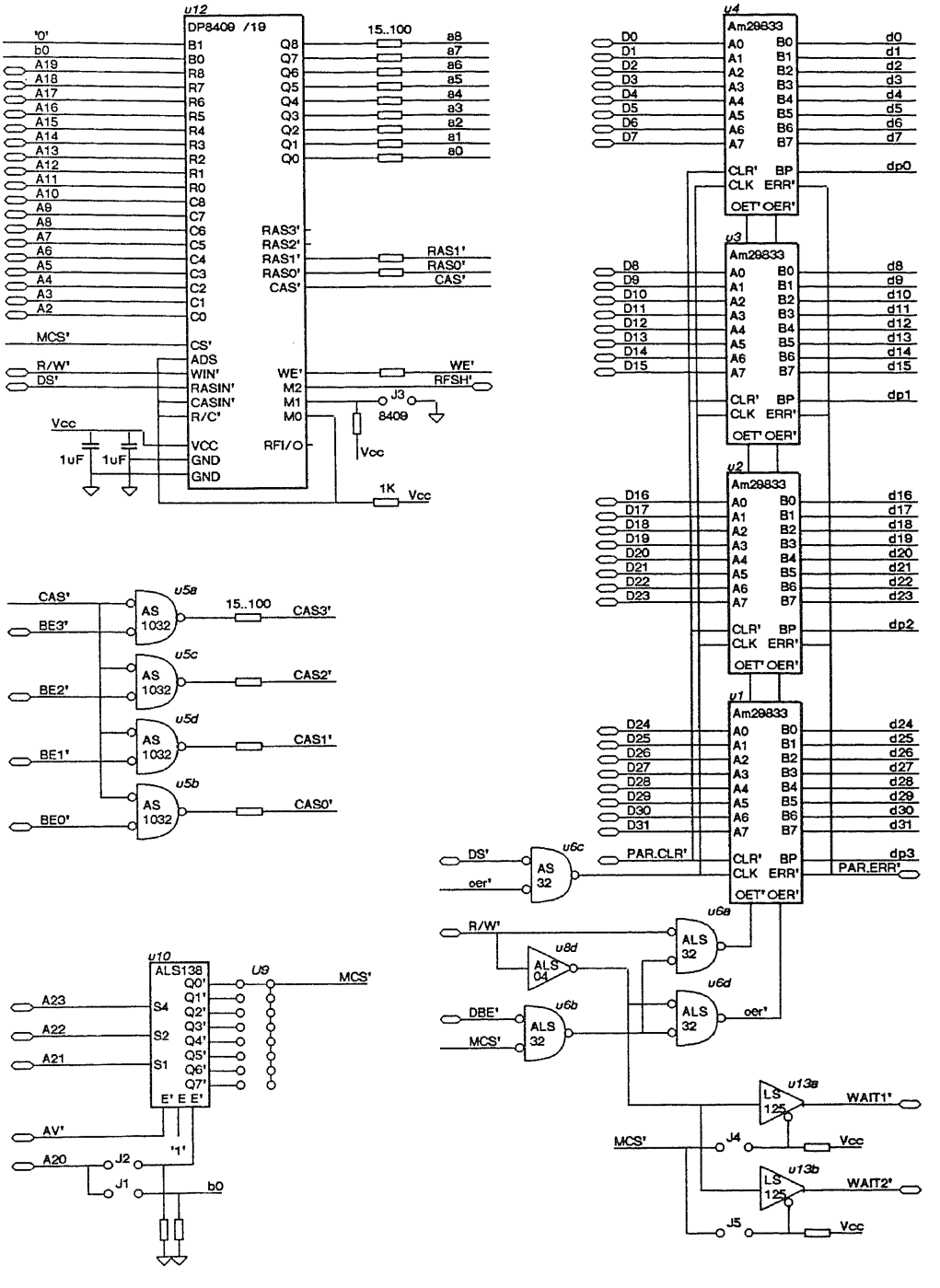
ETH Zurich	NS.s32.cpu5.SIL 5/7 ICU UART		Author: H.Eberle	Date: 3.7.85 REV. 21.5.86



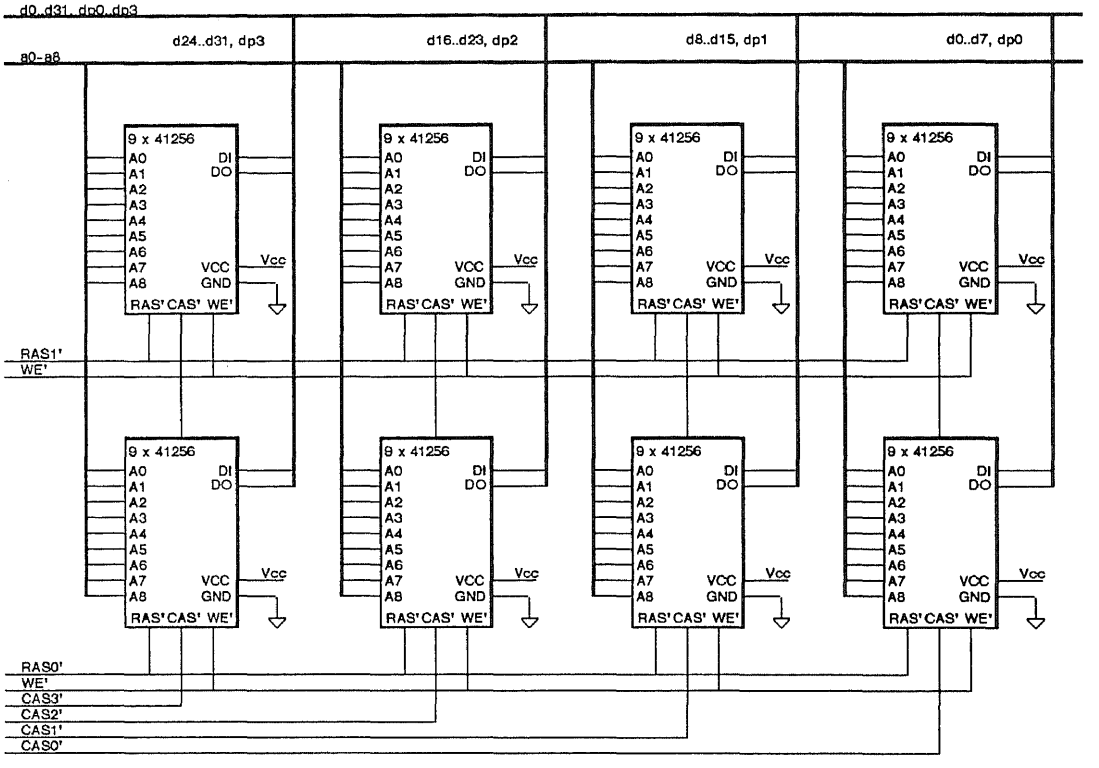
ETH Zurich	*NS.s32.cpu6.SIL* 6/7 SCC, RTC, DIP-switch	Author: H. Eberle	Date: 3.7.85 REV. 3.12.85



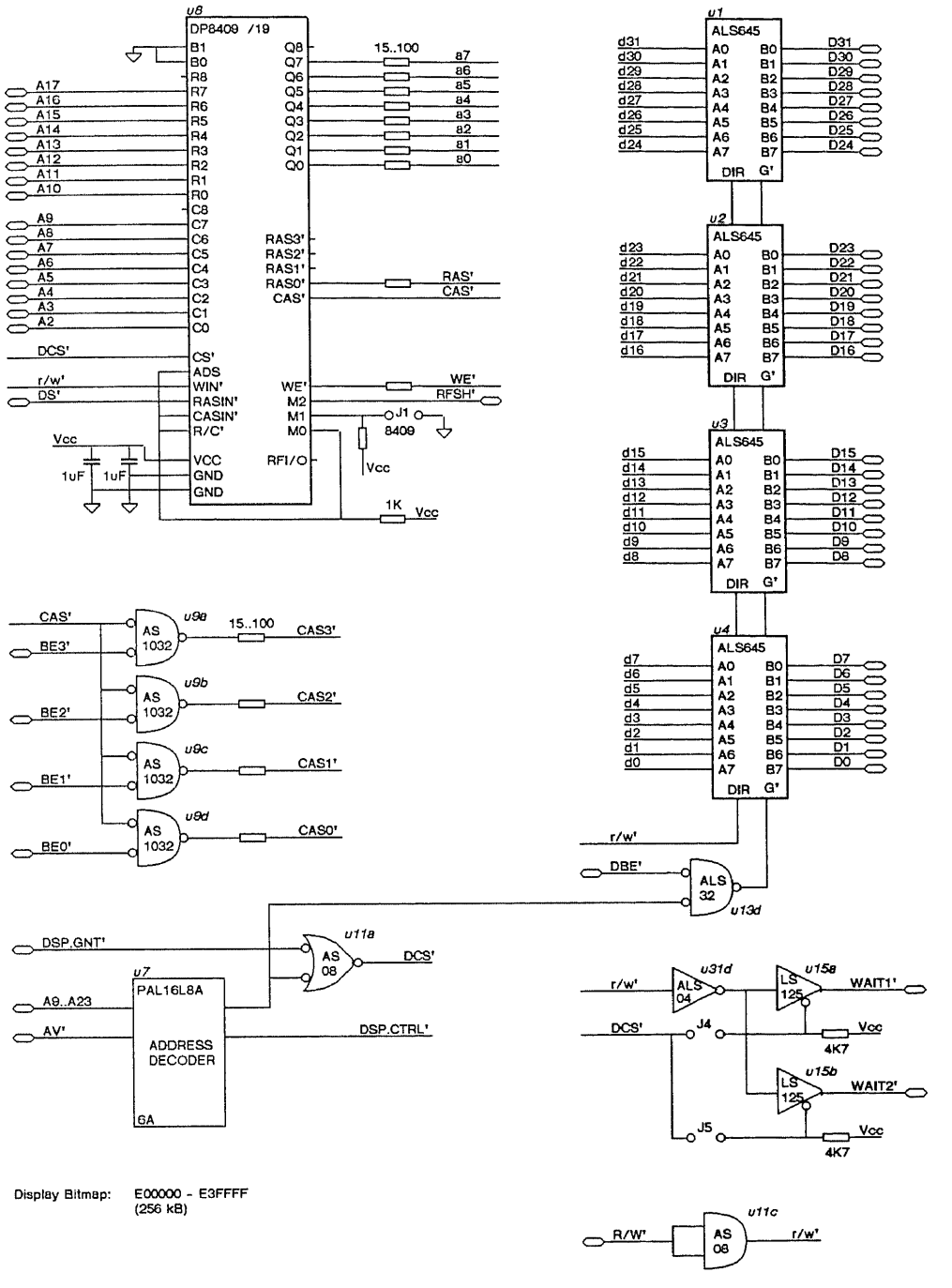
ETH Zurich	NS.s32.cpu7.SIL 7/7 Mouse-Interface	Author: I.Noack/H.Eberle	Date: 29.4.85 REV. 3.12.85



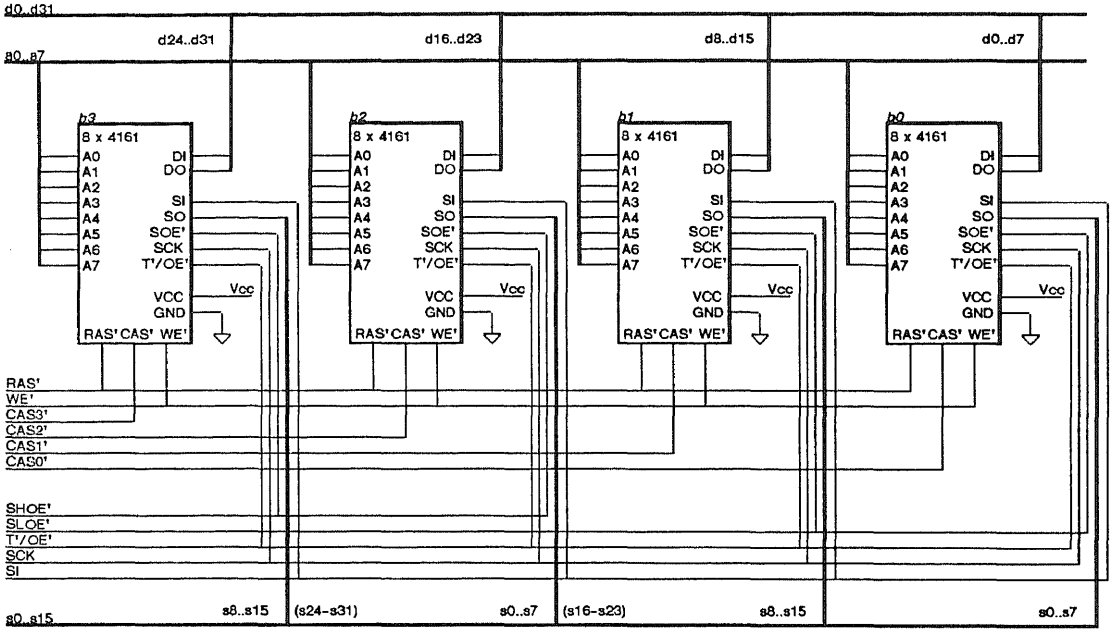
ETH Zurich	NS.s32.mem1.SIL 1/2 DRAM Controller Bus Buffers	Author: H. Eberle	Date: 8.4.85 REV. 9.10.85

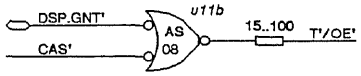


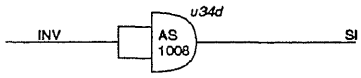
ETH Zurich	NS.s32.mem2.SIL 2/2 Memory (2 x 32 x 256 KBit)	Author: H. Eberle	Date: 29.3.85



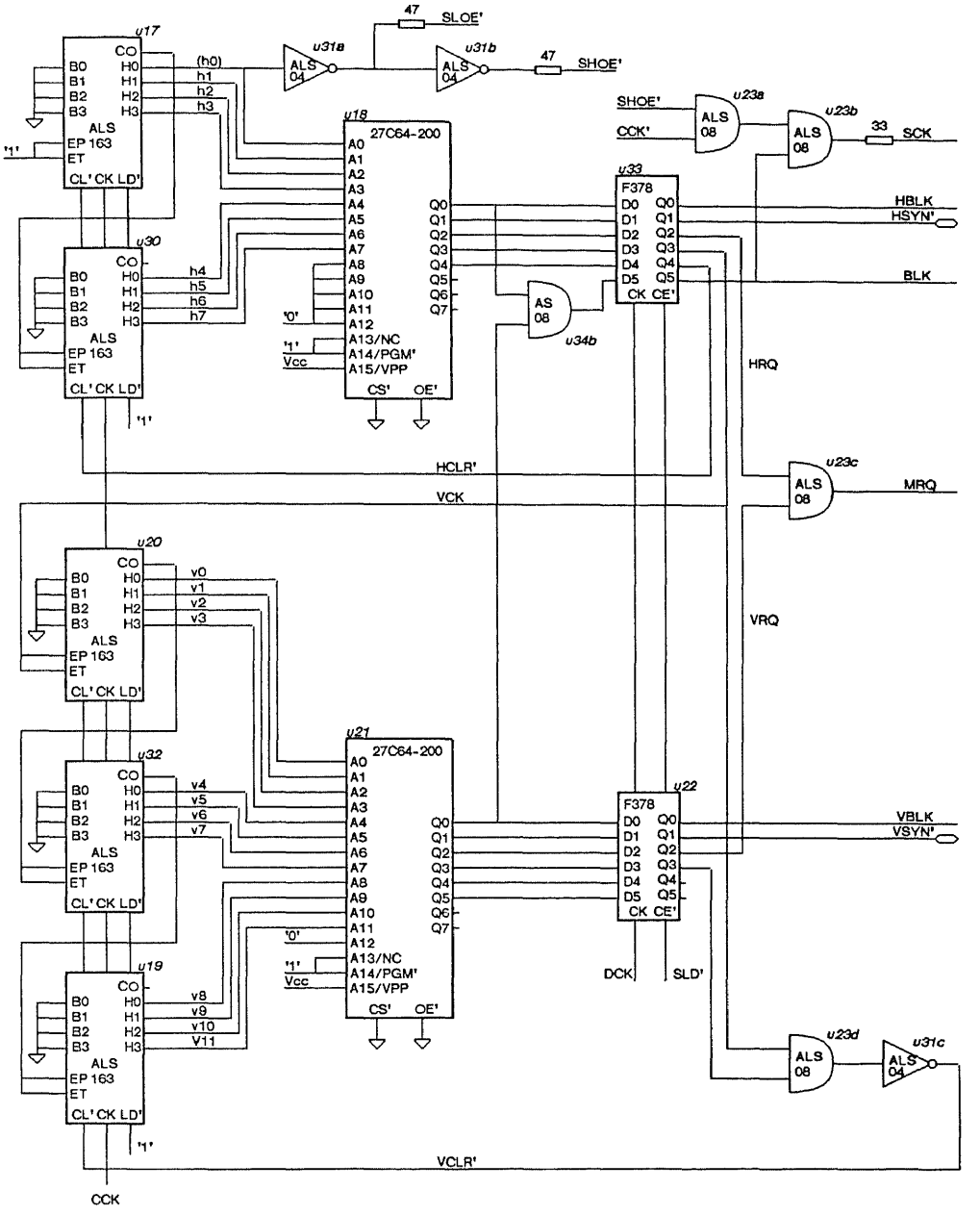
ETH Zurich	NS.s32.dpl1.SIL 1/4 DRAM Controller Bus Buffers	Author: N. Wirth H. Eberle	Date: 27.8.85 REV. 9.10.85



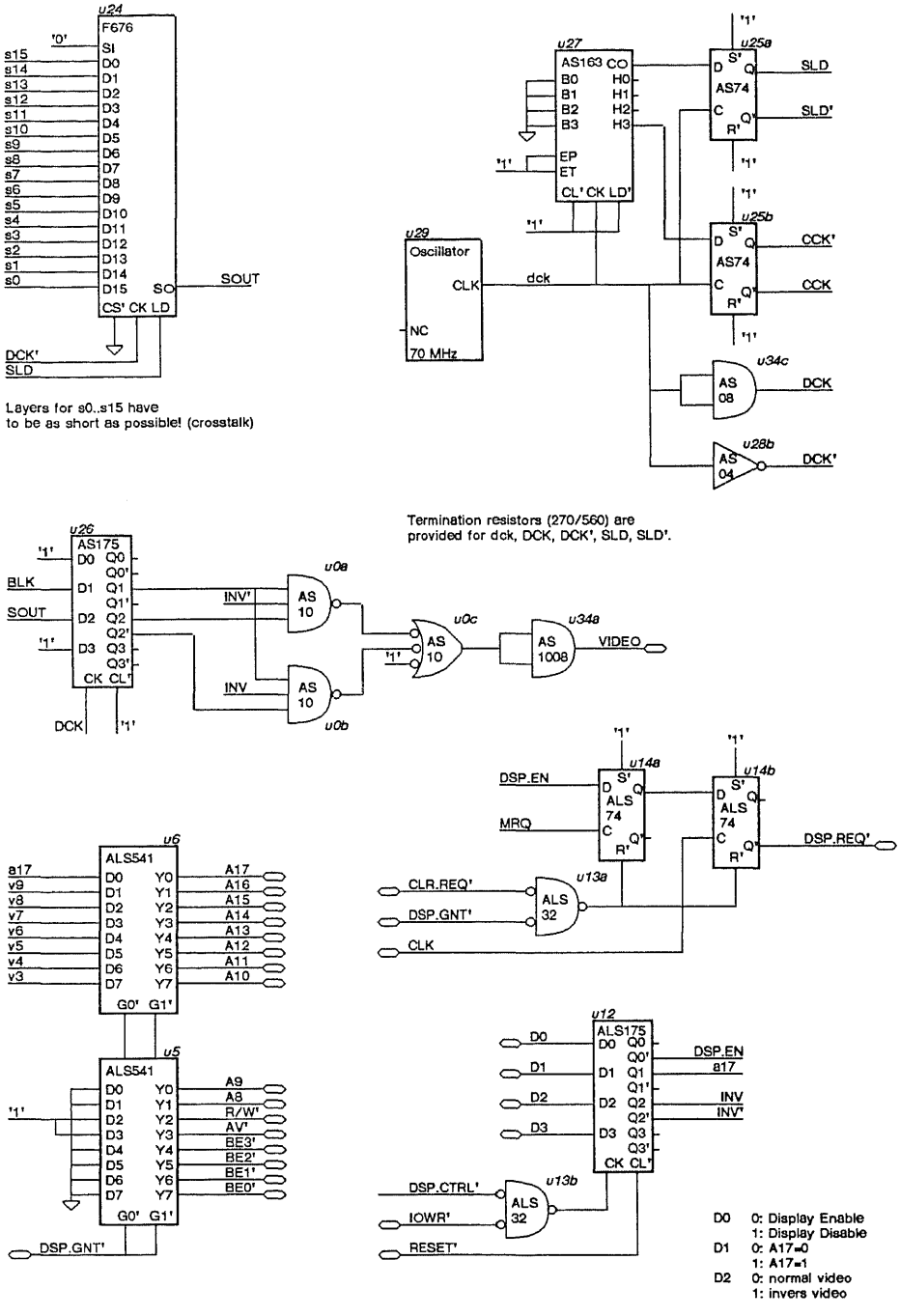




ETH Zurich	NS.s32.dpl2.SIL 2/4 Display Memory (1 x 32 x 64 KBit)	Author: N. Wirth H. Eberle	Date: 27.8.85 REV. 13.6.86



ETH Zurich	NS.s32.dpl3.StL 3/4 Counters	Author: N. Wirth H. Eberle	Date: 27.8.85 REV. 21.5.86

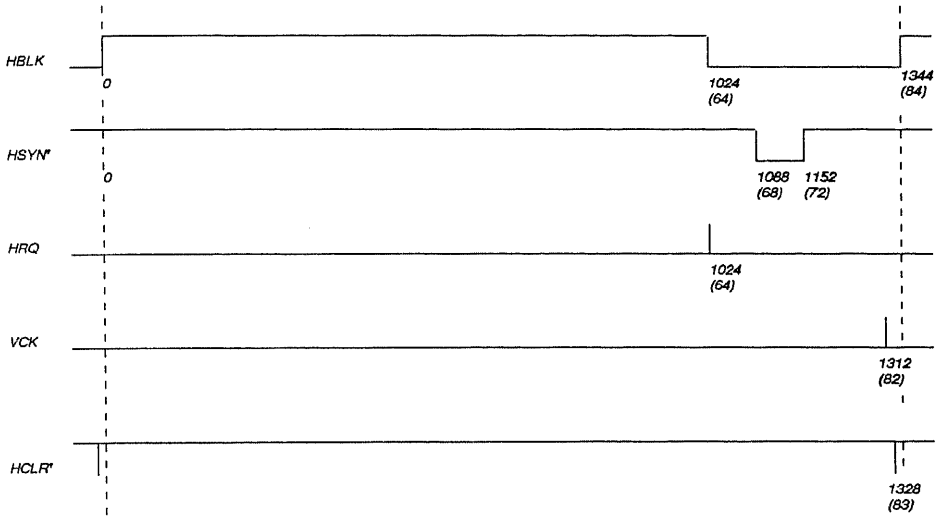


ETH Zurich	NS.s32.dpl4.SIL 4/4 Shifter, Clock Generator, Address Buffer, Status Register	Author: N. Wirth H. Eberle	Date: 27.8.85 REV. 13.6.86

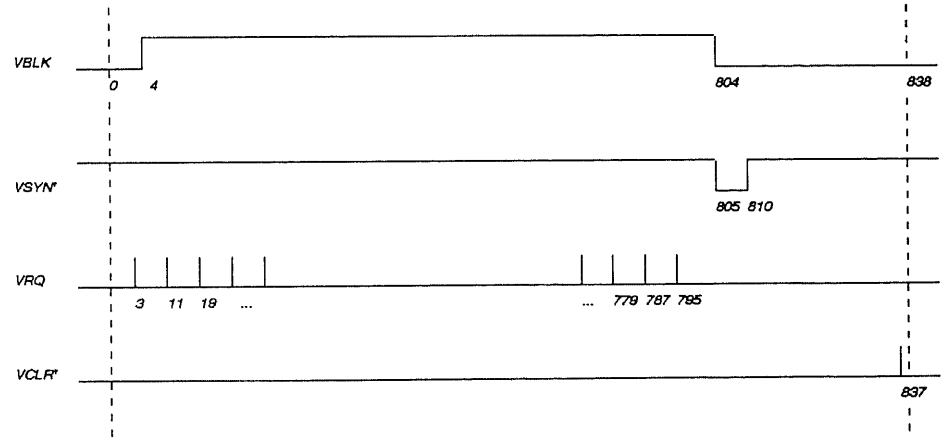
Horizontal timing

(actual H-ROM addresses are 1 less)

Resolution: 1024 x 800
Fh = 52.08 kHz
Fv = 62.15 Hz
Fp = 70 MHz



Vertical timing



ETH Zurich	*NS.s32.dptiming.SIL* Display Timing	Author: H. Eberle	Date: 4.2.86

B PAL Design Specifications

This section contains the PAL design specifications. The function of the PAL devices is described by using the input format of the PALASM PAL assembler [24].

NS.PAL1D: Address Decoder (ROM & IO Devices)

```
PAL20L8                                                    PAL DESIGN SPECIFICATION
ADDRESS DECODER                                                  H.EBERLE 24/07/85
NS.PAL1 D
IFI ETHZ
A23 A22 A21 A20 A19 A18 A17 A16 A15 A14 A13 GND
A12 A11 /IOPG1 /IOPG0 /IOEN A10 A9 /SEL1 /SEL0 /ROMEN /AV VCC
;
IF (VCC) ROMEN  = AV*A23*A22*A21*A20*A19*/A18*/A17*/A16*/A15*SEL1*SEL0
                + AV*A23*A22*A21*A20*A19*/A18*/A17*/A16*SEL1*/SEL0
                + AV*A23*A22*A21*A20*A19*/A18*/A17*/SEL1*SEL0
                + AV*A23*A22*A21*A20*A19*/A18*/SEL1*/SEL0
IF (VCC) IOEN   = AV*A23*A22*A21*A20*A19*A18
IF (VCC) IOPG0  = AV*A23*A22*A21*A20*A19*A18*A17*A16*A15*A14*A13*A12*A11*A10*A9
IF (VCC) IOPG1  = AV*A23*A22*A21*A20*A19*A18*A17*A16*A15*A14*A13*A12*A11*A10*/A9
```

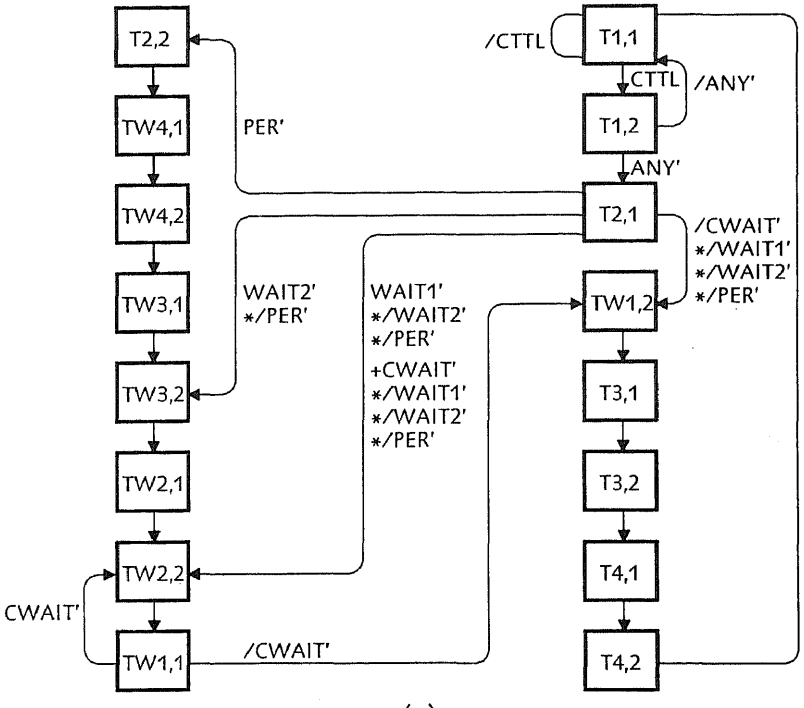
NS.PAL2B: Priority Encoder

```
PAL16L8                                                    PAL DESIGN SPECIFICATION
PRIORITY ENCODER                                                 H.EBERLE 18/06/84
NS.PAL2 B
IFI ETHZ
NC /DSPREQ /REFREQ /REQ0 /REQ1 /REQ2 /REQ3 /CPUREQ NC GND
NC /ANY /CPUGNT /GNT3 /GNT2 /GNT1 /GNT0 /REFGNT /DSPGNT VCC
;
IF (VCC) ANY    =  DSPREQ + REFREQ + REQ0 + REQ1 + REQ2 + REQ3 + CPUREQ
IF (VCC) DSPGNT =  DSPREQ
IF (VCC) REFGNT =  /DSPREQ* REFREQ
IF (VCC) GNT0   =  /DSPREQ*/REFREQ* REQ0
IF (VCC) GNT1   =  /DSPREQ*/REFREQ*/REQ0* REQ1
IF (VCC) GNT2   =  /DSPREQ*/REFREQ*/REQ0*/REQ1* REQ2
IF (VCC) GNT3   =  /DSPREQ*/REFREQ*/REQ0*/REQ1*/REQ2* REQ3
IF (VCC) CPUGNT =  /DSPREQ*/REFREQ*/REQ0*/REQ1*/REQ2*/REQ3
```

NS.PAL3H1 & NS.PAL3K2: Arbiter FSM

```
PAL16R8                                                    PAL DESIGN SPECIFICATION
MEMORY STATE MACHINE H1                                    H.EBERLE 04/06/86
NS.PAL3 H1
IFI ETHZ
FCLK /ANY /PER /WAIT2 /WAIT1 /CWAIT NC NC CTTL GND
/OE /D3 /D2 /D1 /D0 NC NC NC RDY VCC
;
D0   = /D3*/D2*/D1*/D0*CTTL
     + /D3*/D2*/D1*D0*ANY
     + /D3*/D2*D1*D0*PER
     + /D3*/D2*D1*D0*/PER*/WAIT2*/WAIT1*/CWAIT
     + D3*/D2*/D0
     + D2*D1*/D0
     + /D3*D2*/D1*D0*/CWAIT
D1   = /D3*/D2*/D1*D0*ANY
     + /D3*/D2*D1*D0*/PER
     + D3*/D2*D0
     + /D2*D1*/D0
     + /D3*D2*/D1
     + D3*D2*D1*/D0
D2   = /D3*/D2*D1*D0*PER
     + /D3*/D2*D1*D0*/PER*/WAIT2
     + D2*/D1*D0
     + D3*/D2*D1*D0
     + /D3*D2*D1
     + /D3*/D2*D1*/D0
     + D3*D2*D1*/D0
D3   = /D3*/D2*D1*D0*PER
     + /D3*/D2*D1*D0*/PER*WAIT2
     + D3*/D1
     + /D2*D1*/D0
     + D3*D2*D1*/D0
/RDY = /D3*/D2*/D1
     + /D3*/D2*D1*D0*PER
     + /D3*/D2*D1*D0*/PER*WAIT2
     + /D3*/D2*D1*D0*/PER*/WAIT2*WAIT1
     + /D3*/D2*D1*D0*/PER*/WAIT2*/WAIT1*CWAIT
     + D3
     + /D3*D2*D1*/D0
     + /D3*D2*/D1*D0*CWAIT
```

```
PAL16R8                                                    PAL DESIGN SPECIFICATION
MEMORY STATE MACHINE K2                                    H.EBERLE 12/07/86
NS.PAL3 K2
IFI ETHZ
FCLK /ANY /PER RD NC /D0 /D1 /D2 /D3 GND /OE NC G /IOWR /IORD /DS /DBE /CLEAR NC VCC
;
/G   = /D3*/D2*/D1*D0*ANY
     + /D2*D1
     + D3*/D1
     + /D3*D2
     + D3*D2*D1*/D0
CLEAR= /D3*/D2*D1*/D0
IORD = D3*/D2*RD*PER
     + /D3*D2*RD*PER
IOWR = D3*/D2*/RD*PER
     + /D3*D2*/RD*PER
DBE  = /D3*/D2*/D1*D0*/RD*ANY
     + D3*/D1*/RD
     + /D2*D1*/RD
     + /D3*D2*/RD
     + /D3*D2*D1*D0
     + /D3*D2*/D1*/D0
     + /D3*/D2*D1*/D0
DS   = /D3*/D2*/D1*D0*RD*ANY
     + /D3*/D2*D1*D0
     + D3*/D1
     + D3*/D2
     + /D3*D2
```



(a)

Zn	Q3..0	CTTL	ANY'	PER'	WAIT2'	WAIT1'	CWAIT'	Zn+1	D3..0	G	CLEAR'	DBE'	DS'	RDY	IORD'	IOWR'
T1,1	0000	0	X	X	X	X	X	T1,1	0000	1	1	1	1	0	1	1
		1	X	X	X	X	X	T1,2	0001	1	1	1	1	0	1	1
T1,2	0001	X	1	X	X	X	X	T1,1	0000	1	1	1	1	0	1	1
		X	0	X	X	X	X	T2,1	0011	0	1	R/W'	/R/W'	0	1	1
T2,1	0011	X	X	0	X	X	X	T2,2	1101	0	1	0	0	0	1	1
		X	X	1	0	X	X	Tw3,2	1010	0	1	0	0	0	1	1
		X	X	1	1	0	X	Tw2,2	0110	0	1	0	0	0	1	1
		X	X	1	1	1	0	Tw2,2	0110	0	1	0	0	0	1	1
		X	X	1	1	1	1	Tw1,2	0111	0	1	0	0	1	1	1
T2,2	1101	X	X	X	X	X	X	Tw4,1	1100	0	1	0	0	0	1	1
Tw4,1	1100	X	X	X	X	X	X	Tw4,2	1000	0	1	0	0	0	1	1
Tw4,2	1000	X	X	X	X	X	X	Tw3,1	1001	0	1	0	0	0	Y1	Y2
Tw3,1	1001	X	X	X	X	X	X	Tw3,2	1010	0	1	0	0	0	Y1	Y2
Tw3,2	1010	X	X	X	X	X	X	Tw2,1	1011	0	1	0	0	0	Y1	Y2
Tw2,1	1011	X	X	X	X	X	X	Tw2,2	0110	0	1	0	0	0	Y1	Y2
Tw2,2	0110	X	X	X	X	X	X	Tw1,1	0101	0	1	0	0	0	Y1	Y2
Tw1,1	0101	X	X	X	X	X	0	Tw2,2	0110	0	1	0	0	0	Y1	Y2
		X	X	X	X	X	1	Tw1,2	0111	0	1	0	0	1	Y1	Y2
Tw1,2	0111	X	X	X	X	X	X	T3,1	0100	0	1	0	0	1	Y1	Y2
T3,1	0100	X	X	X	X	X	X	T3,2	0010	0	1	0	0	1	Y1	Y2
T3,2	0010	X	X	X	X	X	X	T4,1	1110	0	0	0	1	1	1	1
T4,1	1110	X	X	X	X	X	X	T4,2	1111	0	1	1	1	0	1	1
T4,2	1111	X	X	X	X	X	X	T1,1	0000	1	1	1	1	0	1	1

Y1 = /(R/W'*/PER')
Y2 = /(/R/W'*/PER')

(b)

Figure B.1 Bus control state diagram (a) and truth table (b).

NS.PAL4A: Mouse Direction Discriminator

```
PAL16R8                                          PAL DESIGN SPECIFICATION
MOUSE DIRECTION DISCRIMINATOR                    H.EBERLE 01/04/84
NS.PAL4 A
IFI ETHZ
CLK X0 X1 Y0 Y1 NC NC NC NC GND
/OE /UX /EX /UY /EY /Y1N /Y0N /X1N /X0N VCC
;
X0N   = X0
X1N   = X1
Y0N   = Y0
Y1N   = Y1
;
EX    = /X1N*/X1*/X0N* X0 + /X1N*/X1* X0N*/X0
      + /X1N* X1*/X0N*/X0 + /X1N* X1* X0N* X0
      +  X1N*/X1*/X0N*/X0 +  X1N*/X1* X0N* X0
      +  X1N* X1*/X0N* X0 +  X1N* X1* X0N*/X0
UX    = X0*/X1N + /X0*X1N
;
EY    = /Y1N*/Y1*/Y0N* Y0 + /Y1N*/Y1* Y0N*/Y0
      + /Y1N* Y1*/Y0N*/Y0 + /Y1N* Y1* Y0N* Y0
      +  Y1N*/Y1*/Y0N*/Y0 +  Y1N*/Y1* Y0N* Y0
      +  Y1N* Y1*/Y0N* Y0 +  Y1N* Y1* Y0N*/Y0
UY    = Y0*/Y1N + /Y0*Y1N
```

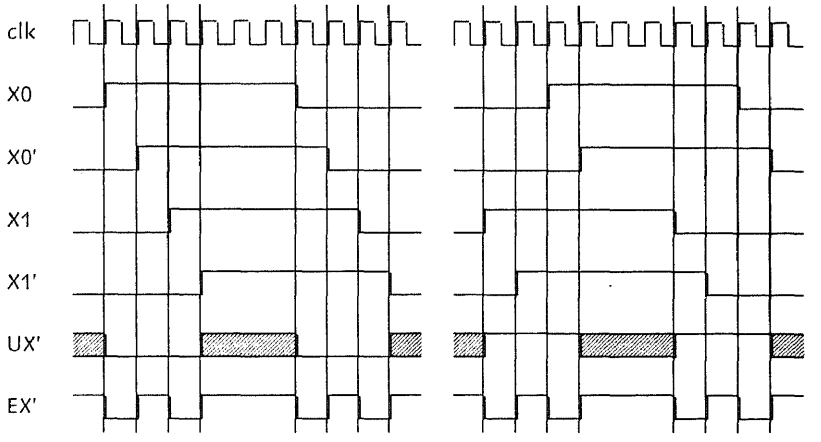


Figure B.2 Timing diagram of the mouse interface.

NS.PAL6A: Display Controller Address Decoder

```
PAL16L8                                                    PAL DESIGN SPECIFICATION
DISPLAY CONTROLLER ADDRESS DECODER                         H.EBERLE 10/12/85
NS.PAL6 A
IFI ETHZ
A23 A22 A21 A20 A19 A18 A17 A16 A15 GND
/AV /DPSTAT A14 A13 A12 A11 A10 A9 /MCS VCC
;
IF (VCC) MCS      = AV*A23*A22*A21*/A20*/A19*/A18
IF (VCC) DPSTAT  = AV*A23*A22*A21*A20*A19*A18*A17*A16*A15*A14*A13*A12*A11*/A10*A9
```

NS.PAL7B: Byte Enable & Other Glue Logic

```
PAL16L8                                                    PAL DESIGN SPECIFICATION
BYTE ENABLE & OTHER GLUE LOGIC                             H.EBERLE 20/07/86
NS.PAL7 B
IFI ETHZ
/CPUGNT /DBE A1 RD /MMUMAC /CPUBE0 /CPUBE1 /CPUBE2 /CPUBE3 GND
NC /BE3 /BE2 /BE1 /BE0 /R1 /R2 /GW /GD VCC
;
IF (CPUGNT) BE0  = MMUMAC*/A1 + /MMUMAC*CPUBE0 + RD
IF (CPUGNT) BE1  = MMUMAC*/A1 + /MMUMAC*CPUBE1 + RD
IF (CPUGNT) BE2  = MMUMAC* A1 + /MMUMAC*CPUBE2 + RD
IF (CPUGNT) BE3  = MMUMAC* A1 + /MMUMAC*CPUBE3 + RD
;
IF (VCC) GD       = CPUGNT*DBE*/MMUMAC + CPUGNT*DBE*/A1
IF (VCC) GW       = CPUGNT*DBE*MMUMAC*A1
;
; RD = R/W'; /R1 => r'/w; /R2 => r/w'
IF (VCC) R1       = RD
IF (VCC) R2       = /RD
```

C Interface Connectors

The computers interface to the environment consists of the following connectors:

Keyboard Interface	5-pin Audio-DIN connector
Mouse Interface	9-pin female Cannon connector
RS-485 Interfaces	9-pin female Cannon connector
RS-232-C Interface	25-pin female Cannon connector
Video Synchronization	4-pin Audio-DIN connector
Video Data	BNC connector

C.1 Processor Board

Keyboard Interface

Pin	Direction	Mnemonic	Signal Description
1		NC	No Connection
2	I	KB.RxD'	Receive Data from Keyboard
3	O	KB.TxD'	Transmit Data to Keyboard
4		GND	Ground
5		Vcc	+5V Power

Mouse Interface

Pin	Direction	Mnemonic	Signal Description
1		Vcc	+5V Power
2	I	MYA	Y Displacement
3	I	MYB	Y Displacement
4	I	MXA	X Displacement
5	I	MXB	X Displacement
6		GND	Ground
7	I	MB1'	Middle Switch
8	I	MB0'	Right Switch
9	I	MB2'	Left Switch

RS-485 Interfaces

Pin	Direction	Mnemonic	Signal Description
1		GND	Ground
2		NC	No Connection
3		GND	Ground
4	I/O	D+	Noninverted Data
5	I/O	D-	Inverted Data
6		NC	No Connection
7		NC	No Connection
8	I/O	D+	Noninverted Data (identical with pin 4)
9	I/O	D-	Inverted Data (identical with pin 5)

RS-232-C Interface

Pin	Direction	Mnemonic	Signal Description
1		FGND	Frame Ground
2	O	TxD'	Transmit Data
3	I	RxD'	Receive Data
4	O	RTS	Request to Send
5	I	CTS	Clear to Send
6	I	DSR	Data Set Ready
7		SGND	Signal Ground
8	I	DCD	Data Carrier Detect
9-19		NC	No Connection
20	O	DTR	Data Terminal Ready
21-25		NC	No Connection

C.2 Display Controller Board

Video Interface

Pin	Direction	Mnemonic	Signal Description
1	O	HSYNC'	Horizontal Synchronization
2		GND	Ground
3		NC	No Connection
4	O	VSYNC'	Vertical Synchronization

A separate, shielded coax-cable is used for the video data signal.

C.3 Motherboard

Memory Bus

Pin	Mnemonic	Signal Description
Aa1-Aa4	BE0'-BE3'	Byte Enable
Aa5	RFSH'	Refresh
Aa6	DS'	Data Strobe
Aa7	PAR.ERR'	Parity Error
Aa8	PAR.CLR'	Parity Clear
Aa9-Aa12	INT4'-INT7'	Interrupt Request
Aa13	NC	No Connection
Aa14	ILO'	Interlocked Operation
Aa15	CLR.REQ'	Clear Request
Aa16	DSP.REQ	Display Request
Aa17-Aa20	REQ0'-REQ3'	Request
Aa21	DSP.GNT'	Display Grant
Aa22-Aa25	GNT0'-GNT3'	Grant
Aa26, Aa27	GND	Ground
Aa28	-5V	-5V Power

Aa29	+12V	+12V Power
Aa30	-12V	-12V Power
Aa31, Aa32	+5V	+5V Power
Ab1-Ab32	GND	Ground
Ac1-Ac32	D0-D31	Data
Ba1, Ba2	+5V	+5V Power
Ba3	-12V	-12V Power
Ba4	+12V	+12V Power
Ba5	-5V	-5V Power
Ba6, Ba7	GND	Ground
Ba8	CWAIT'	Continuous Wait
Ba9	WAIT1'	1 Wait State
Ba10	WAIT2'	2 Wait States
Ba11	IO.EN'	IO Enable
Ba12	IO.WR'	IO Write
Ba13	IO.RD'	IO Read
Ba14	DK.CS'	Disk Card Select
Ba15	DK.INT	Disk Interrupt Request
Ba16-Ba19	NC	No Connection
Ba20	GND	Ground
Ba21	RESET'	Reset
Ba22	GND	Ground
Ba23	RESET.IN'	Reset Input
Ba24	GND	Ground
Ba25	CLK	Clock (10 MHz)
Ba26	GND	Ground
Ba27	FCLK	Fast Clock (20 MHz)
Ba28	GND	Ground
Ba29	RDY	Ready
Ba30	DBE'	Data Buffer Enable
Ba31	AV'	Address Valid
Ba32	R/W'	Read/Write
Bb1-Bb32	GND	Ground
Bc1-Bc24	A0-A23	Address
Bc25-Bc32	(A24-A31)	(Address)

The Memory Bus uses DIN41612 connectors with 3x32 circuits.

Disk Controller Interface

Pin	Mnemonic	
	Memory Bus	Disk Controller
1	D0	DAL0
3	D1	DAL1
5	D2	DAL2
7	D3	DAL3
9	D4	DAL4
11	D5	DAL5

13	D6	DAL6
15	D7	DAL7
17	A2	A0
19	A3	A1
21	A4	A2
23	DK.CS'	CS'
25	IO.WR'	WR'
27	IO.RD'	RD'
29	CWAIT	NC
31	R/W'	NC
33	DS'	NC
35	DK.INT	INTRQ
37	NC	DRQ
39	RESET'	MR'

All even numbered pins (2 through 40) are used as signal grounds.

Floppy Power & Reset Connectors

Mnemonic	Signal Description
+12V	+12V Power
GND	Ground
+5V	+5V Power
GND	Ground
RST'	Reset Input

The RST' signal is provided by an externally mounted reset switch.

Power Connector

Mnemonic	Signal Description
PFD	Power Fail Down (provided, but not used)
-12V	-12V Power, 0.7A maximal current
+12V	+12V Power, 5.0A maximal current
-5V	-5V Power, 0.7A maximal current
NC	No Connection
GND	Ground
GND	Ground
GND	Ground
+5V	+5V Power, 15.0A maximal current
+5V	+5V Power