4.12 CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION

4.12.1 Introduction

The General Interface Card (CMI28) is an optional card for the Fairlight CMI Series IIX. Please note that for use in the Series IIX machines a **CMI-25 rev-3** motherboard is required (or equivalently modified old motherboard). The General Interface Card is designed to handle reading and generating of SMPTE code and to control four MIDI ports as well as controlling the CLICK feature. Because its main purpose is for SMPTE and MIDI it is often referred to as the **SMIDI Card.** In the Series IIX machines the card is plugged into the second slot on the left between the Master Card and a Channel Card.

The SMIDI card is connected to the General Interface Support Card, the CMI-29 via a 26-way cable. This card is housed in a box bolted to the back of the CMI. This support unit has the opto-couplers, and open collector buffers for receiving and transmitting MIDI as well as the analog circuitry for reading and generating SMPTE code to tape.

The SMIDI card in general is a microcomputer system with a Motorola 68000 microprocessor, either 8k or 16k words of ROM and either 8k or 32k words of static RAM. It is possible to extend this to 64k words. It has a DMA interface to the CMI with the capability of DMAing to either P1 or P2. In Series IIX machines the DMA is only on P1. The card has 4 ACIA's (68B50) for the 4 MIDI ports, two 68B40 Programmable Timer Modules as well as associated circuitry for reading and generating SMPTE code and click/sync in and multiple syncs out.

4.12.2 Memory Configuration

There are four 28-pin sockets for ROM and static RAM. The minimum configuration is 8k words of ROM (2 x 2764) and 8k words of static RAM (2 x 6264). The ROM can be configured for 16k words by breaking the link (LK1) between pin 27 of the ROM and +5v and join the link LK1 to A15 from the processor and plugging in the appropriate two 27128's. Similarly the RAM can be arranged to accommodate 32k static RAM chips (e.g., MK4856 pseudo-statics) by breaking the links LK2 and LK3 to +5v and connecting A14 and A15 to pins 26 and 1 of the RAM chips (via LK3 and LK2), respectively. Further, there is an option for 64k words of RAM; by soldering two 32k RAM chips on top of each other except for pin-20, the chip select, which should be connected to the pads provided from the select circuitry, the AND gates (D12). All these memory expansions will depend on the availability of these chips.

NOTE; when plugging in the ROM's, they should be labelled 'odd' and 'even'. The even one should be plugged into E5,6 (near the 68000) and the odd one into E8 (between the RAM chips).
Memory addressing: ROM starts at $000000 and RAM starts at $080000.

4.12 <u>CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION</u> (continued)

4.12.3 68000/6809 DMA Bus Interface

(Refer to Drawing CMI-28 rev 2 page 4.)

Communication between the 68000 processor and the 6809 CPU is achieved by DMA (Direct Memory Access) on the system bus. The 68000 waits until no higher priority device is occupying the bus and then either 6809 (P1 or P2) is temporarily hung while the 68000 executes a normal bus cycle writing to or reading from memory or a peripheral on the bus. In this manner the entire 64K address space of each 6809 processor appears as a small slice of the 16 megabyte address space of the 68000. Software then defines various protocols for the different processors to pass messages and data to one another by simply placing them in system memory.

The DMA interface provided on the 68000 SMPTE/MIDI Card is a very flexible one. It automatically handles either 8 or 16-bit data transfers (doing double cycles across the 8-bit CMI bus in the latter case) and can do so on either P1 or P2 cycles, selecting any desired memory mapping which has been set up on the Q256 memory card.

DMA is initiated by the 68000 when it accesses any address in the range $040000 to $05FFFF. These addresses are decoded by the LS259 (E12) on drawing CMI-28-1/7 and result in the $\overline{CMI}$ signal being asserted (low). Since the rest of the interface circuitry is not activated yet, PACK (to be explained later) will be low and a low will be presented at the data input of flip flop C12(a) whose function is to synchronise the transfer with the CMI bus.
Address line A16 is used to select which 6809 processor's bus cycle(s) are to be used for the transfer. The timing signals for both processors are input to LS241 buffer A7 which is wired as a muliplexer:-

If A16 is low, P2ϕ2 is enabled through to become
Pϕ2, ADD2 becomes PADD and so on.
If A16 is high, P1's timing signals are enabled instead.

By this means, the address range specified above is split in two: from $040000 to $04FFFF the transfer automatically occurs on P2 bus cycles, while from $050000 to $05FFFF it occurs on P1 cycles. Refer to the 6809 CPU documentation for more information on the interleaved P1/P2 CMI bus cycles.
Thus at the beginning of the data cycle of whichever processor is selected, the Pϕ2 signal clocks the LS74, recording the fact that a DMA cycle is required.

All DMA devices are interconnected on the motherboard in a "daisy chain". Each device is assigned a given priority in the chain and must wait until no higher priority device is already using the bus. The 6809 CPU is the always the last device in the chain. There are two separate daisy chains in the CMI system, one for each 6809 CPU. Since the 68000 SMIDI card can perform DMA on either CPU's cycles, it is a member of both chains. $\overline{ETL1}$, $\overline{ENL1}$ and $\overline{RDMA1}$ are the chain signals for P1, $\overline{ETL2}$, $\overline{ENL2}$, $\overline{RDMA2}$ are for P2. Which set are used is again selected by the state of A16 at the time of transfer.

4.12 CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION (continued)

The selected $\overline{ETL}$ (Enable This Level) signal is low when no higher priority device is occupying the bus. After the $\overline{CMI}$ signal has been latched, nothing happens until this signal is low, whereupon the $\overline{RDMA}$ (Request DMA) is driven low through the selected LS12 gate. Any DMA device pulls this open collector line low to to request bus access to the CPU. At the same time, the selected $\overline{ENL}$ (Enable Next Level) signal is inhibited. Normally, the low on $\overline{ETL}$ comes in and goes out again on $\overline{ENL}$ to indicate to lower priority devices that the bus is available but when the 68000 requires a transfer $\overline{ENL}$ is held high to hold up the lower devices.
The CPU acknowledges that it will hang and release the bus for the next cycle by asserting ACK1 or ACK2; the selected ACK signal becomes PACK. When a request has been generated (C12(a) $\overline{Q}$ hi) and this level is enabled (ENL lo), the rising edge of PACK clocks a low into flip flop B11(a) to generate $\overline{DCYCLE}$. This signal indicates that the next bus cycle is definitely going to be a 68000 DMA transfer and remains asserted until the end of the address phase of the actual DMA cycle.

The other half (b) of B11 is also clocked by PACK to generate the P1 or P2 $\overline{DMAC}$ (DMA Claim) signal as selected by A16. This signal goes to the Q256 RAM card to select the memory mapping which has been set up specifically for the 68000. In this way the 68000 may have access to part or all of the same physical memory space as the 6809 CPU or it may have access to an entirely different part of physical memory as required by software. The $\overline{DMAC}$ signal is asserted during the data cycle preceding the actual transfer.
The address phase of the DMA cycle is indicated when $\overline{ATB}$ (Address To Bus) is asserted by the LS10 B10. At this time the lower 15 bits of the 68000 address bus are enabled on to the CMI bus through the two LS244's A2 and A3 to select the required location within the 6809 address space. VMA is driven high through LS125 B1 to indicate a Valid Memory Address and the 68000 R/W line is driven through the same buffer to indicate a read or write cycle. When the 68000 performs 8-bit memory accesses, the $\overline{UDS}$ and $\overline{LDS}$ signals (upper and lower address strobes) indicate whether an even or odd address is being accessed. The sense of these signals are clocked into JK flip flop H12 at the beginning of $\overline{DCYCLE}$ to generate HIBYTE and LOBYTE. The latter signal becomes the least significant address line driven onto MA0 through A3.

In the case of 16-bit accesses, the hardware automatically requests two successive DMA accesses across the 8-bit CMI bus. Both $\overline{UDS}$ and $\overline{LDS}$ are asserted so that the JK outputs HIBYTE and LOBYTE simply toggle on each access. It does not matter which byte transfers first and in fact this depends on the initial state of N6. LOBYTE directs the data to or from the odd or even address and both signals control whether the higher or lower 8 data lines are directed to the data bus.

The data bus interface consists of Schmitt bidirectional bus transceiver LS640 A6 and bidirectional driver/latches C5 and C6 (LS646s). The data phase of the DMA transfer is indicated by the assertion of $\overline{DTB}$ (Data To Bus) at the rising edge of BRA when a DMA cycle is in progress. This is performed by flip flop MN4. $\overline{DTB}$ enables the bus transceiver A6 and the direction is determined by the 68000 R/W signal.

If the 68000 is writing to the CMI bus, C5 or C6 simply act as buffers to transfer the high or low 68000 data signals (PD0-15) through to A6. HIBYTE or LOBYTE plus $\overline{CMI}$ being asserted will drive the $\overline{G}$ input of the appropriate LS646 for the duration of the DMA cycle (LS02 and LS32 gates M2 and M1).

When the 68000 reads from the CMI bus, C5 or C6 must latch the data in from the bus to hold it until the 68000 terminates its own cycle and latches the data internally, about 50nS after the end of the DMA cycle. 100nS before the end of the data phase, the CMI timing signal CAS goes low, resulting in a rising edge on $\overline{BCAS}$. Data from memory is guaranteed to be valid at this time. B10 generates the LDATA (Latch Data) signal which is ANDed with either HIBYTE or LOBYTE to latch the data coming into the A side of C5 or C6. The output of the latch (B side of the selected LS646) is driven onto the PD lines until the 68000 completes its cycle and negates $\overline{CMI}$.

Termination of the transfer after single or double DMA cycles is controlled by the two flip flops in LS74 C10:

In the **single** (8-bit) transfer case, either UDS or LDS will be low. This will cause the LS10 A10 to output a high, and $\overline{DTACK2}$ will be generated as soon as LDATA occurs. The 68000 will then terminate its cycle immediately, after only one DMA cycle.

In the **double DMA cycle** (16-bit) case, both $\overline{UDS}$ and $\overline{LDS}$ are high so $\overline{DTACK2}$ will not be generated until the first flip flop in C10 is set. Initially this flip flop is reset. At the first LDATA pulse a high is clocked in but $\overline{DTACK2}$ is not generated because of the propagation delay through to the next flip flop. Since $\overline{DTACK2}$ is not asserted, the 68000 still waits with address and address/data strobes asserted. If writing, the data remains asserted by the 68000 but both address and data are removed from the CMI bus when $\overline{ATB}$ and $\overline{DTB}$ are negated respectively. If reading, the first byte read in is latched and held by C5 or C6. Since $\overline{CMI}$ will still be asserted and PACK will have been negated, the whole process of waiting for daisy chain priority and DMA requesting begins again in order to perform a second DMA cycle. The second cycle can be held up indefinitely by higher priority devices using the bus after the first cycle. When the second LDATA edge comes along the high on the LS10 output is clocked into the second C10 flip flop and $\overline{DTACK2}$ is asserted. On the next falling edge of PCLK, the 68000 recognises that $\overline{DTACK}$ has been asserted. On the second falling edge of PCLK the data is latched internally for a read, and the address and strobes are released. The low on BAS resets the flip flops at C10.

4.12.4 Debugging Notes for the DMA Circuitry

If the timing circuitry of the DMA interface is faulty, the most likely result is that $\overline{\text{DTACK2}}$ will never be generated and the 68000 will simply hang which makes debugging easy. In this case, check first that the address decoding is generating $\overline{\text{CMI}}$, then that the daisy chain signals are present. Then look for an 800nS pulse on $\overline{\text{DCYCLE}}$, indicating that DMA cycles are actually ocurring. Continue through to the $\overline{\text{ATB}}$, $\overline{\text{DTB}}$ and $\overline{\text{LDATA}}$ signals, checking not only that they are generated but also that they get to their respective destinations in the circuitry.

If the DMA cycles are being synchronised and timed correctly check that the address buffers and data buffer/latches are being enabled and clocked at the correct times.

If all timing circuitry is correct, the last possibilty is data or address bus shorts, open circuits or faulty drivers. Special test ROMs are available which cause the 68000 to repetitively copy bytes and words from one location to another in CMI memory. The 6809 monitor can then be used to deduce which data or addresses cause problems.

4.12.5 SMPTE/MIDI Card Peripheral Circuits

There are four different peripheral circuits on the SMIDI card. Firstly, there are the four ACIA's (G7-11) which are the MIDI ports A,B,C, and D. Then there is the Timer (bottom rev.2 G5,6) which is used in conjuction with the SMPTE read and generate circuits (which are the other two circuits) as well as the Click In and Out.

The ACIA's and Timers work from an 8-bit data bus with (asynchronous) interfacing circuitry. They are driven also by the E (enable) signal from the 68000. The frequency of this clock is one-tenth of the 68000 clock (10MHz) with a 60/40 duty cycle (6 clocks high, 4 clocks low)

Initially the flip-flops (F2) are cleared causing a high $\overline{\text{DTACK3}}$ output setting the LS646 transceiver (G4) into the transparent mode. The direction of data flow is determined by the R/W line with the IO selected. Without IO line selected it appears in write mode. The peripheral is selected by the LS138 enabled by the $\overline{\text{CS}}$' signal. The first flip-flop F2(a) is clocked on the first falling edge of E with the IO select and the data strobe high (ie either $\overline{\text{LDS}}$ or $\overline{\text{UDS}}$ low). The Q output of F2(a) is applied to the NAND gate (G3), asserting $\overline{\text{CS}}$'. Selecting the peripheral at this time ensures that the peripheral has adequate address setup time.

On the next falling edge of E, the $\overline{Q}$ output of F2(b) is clocked low asserting $\overline{DTACK3}$ and latching data in the transceiver (G4). The asserted $\overline{DTACK3}$ signal deselects the peripheral by causing $\overline{CS}$' to go high. Flip-flop F2(a) is cleared by IO going low when the access terminates. Clearing flip-flop F2(a) also initializes the interface circuitry for the next access.

The ACIA's are selected by E10 and appear at addresses; \$60020, \$60030, \$60040 and \$60050. They share a common interrupt level - level 3. Their transmit and receive data lines are wired to the 26-way connector to be connected to the MIDI drivers and opto-coupler receivers.

The programmable timer (G5,6) appears at the general address \$60000, and has an interrupt level-2 to the 68000.

RAM is fast enough (150ns) to not need a delay on the $\overline{DTACK}$ line, so that when RAM is selected $\overline{DTACK}$ is also enabled. Not so with ROM, a delay is needed and is provided by the LS161 counter (F1) which delays the enabling of the $\overline{DTACK}$ line by 12 processor clock cycles.

4.12.6 Interrupts

The 68000 has seven levels of interrupts. The priority for the interrupts is made by hardware through the 74LS148 ic (C2). The lowest level interrupt ($\overline{INT1}$) and the NMI ($\overline{INT7}$) are enabled and cleared by the CMI through the control port (B4). $\overline{INT2}$ is for the 68B40 Timer, $\overline{INT3}$ is for the ACIA's. $\overline{INT4}$ is for reading a SMPTE 'one' and $\overline{INT5}$ for a SMPTE 'zero' and are cleared by addressing location SMPTEWR on i.c. E10 (LS138). $\overline{INT6}$ is for $\overline{SMPTE}$ generation and is cleared by writing to the shift registers (C7,C8), i.e. by signal $\overline{SMPTERD}$.

4.12.7 SMPTE Generating Circuitry

An oscillator (3.84MHz) is divided by 10 (G2,G1) to provide a standard for generating the 3 different rates of SMPTE code (24, 25 and 30 frames per second). All three are denominators of 384,000. Further division, depending on the frame rate selected, is done by the Timer (G5,6) giving the signal CLK2, which is the bit rate for a SMPTE 'one' (ie 160 bits per frame). This is in turn divided by 2 (C11) giving CLK1 which is the bit rate for a SMPTE 'zero' (ie 80 bits per frame). When a SMPTE word is ready it is written to the Parallel-In-Serial-Out registers (C7,C8)at address \$60070 (through B9,B8 and B7). When this writing takes place the interrupt $\overline{INT6}$ if it has been asserted is now cleared. The data in the shift registers (C7,C8) is clocked out by CLK1, a 4-bit counter (D11) is also clocked which causes the interrupt on level-6 (INT6) when it reaches its terminal count of 16. Now, if a 'zero' is shifted out from C8 the flip-flop C11 is toggled at the rate determined by CLK2, but if a 'one' is shifted out from C8 the flip-flop C11 is toggled at the CLK1 rate. Thus, the word stored on the shift registers is outputted in SMPTE form.

4.12 CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION (continued)

4.12.8 SMPTE Reading Circuitry (Refer to Timing Diagram)

SMPTE code coming from tape, being converted to TTL signal levels by the CMI-29 board, is received by the CMI-28 through pin 17 of the 26-way connector. The circuitry consisting of the EXOR gates (C1) and the resistor-capacitor combination creating a pulse (at pin 6 of C1) for every up or down transition of the incoming signal.

The required output from this SMPTE data separator is to have one interrupt occur for every SMPTE 'one' read and another interrupt for every SMPTE 'zero' read. This process can be followed through with the timing diagrams. The 68B40 timer is set, according to the frame rate of the SMPTE being read, to 3/4 of the time for one bit cell. The circuit then detects whether there has been a transition in that time or not. If there has been a transistion then a 'one' is read, if no transition occurred then a 'zero' is read.

4.12.9 Sync In and Out

The SMIDI card also takes care of some of the sync-ing functions of the system. On revisions 1 (modified) and 2 of the CMI-28 SIDI board there are two 68B40 programmable timers (each with 3 timers inside), one wired on top of another. The input clock of 3rd timer in the bottom 68B40 (timer a) is wired to the Click/Sync input socket on the support box mounted to the rear panel of the mainframe. The output of this timer is fed into the inputs of the three timers in the top i.c. (timer b) providing a cascaded timer system. These four outputs are fed to the CMI-29 in the support box, to a 5-pin DIN socket via open-collector buffers.

```
                  _____________            |---timer 1b---> Sync 1 out
Click/          |             |           |
Sync IN--->|  timer 3a   |-------->|---timer 2b---> Sync 2 out
                |_____________|   |       |
                                  |       |---timer 3b---> Sync 3 out
                                  |
                                  |-------------------> Sync 4 out
```

4.12.10 General Interface Support Card CMI-29

This circuit board contains the analog circuitry required for the I/O for SMPTE and MIDI. There are 3 MIDI inputs (A, B & C) and 4 MIDI outputs (A, B, C & D). Provision has been made for a fourth MIDI input (D). The SMPTE input has a balanced line receiver. The signal is then filtered and converted to TTL compatible signals through the LM311 comparator. The SMPTE out signal is converted from a TTL to a balanced line signal. The SMPTE in and out signals are received and transmitted via two 3-pin XLR sockets and are connected to the board via the 20-way socket. The MIDI I/O circuitry is the standard current loop drivers (open-collector buffers (U10) and receivers (fast opto-couplers (U5-U8)).

There are two other output (5-pin DIN) sockets. One is the CLOCK output, containing the CLOCK, RESET/START and RUN/STOP TTL compatible signals. This CLOCK output is designed to control Roland-type drum machines, etc. The other is the multiple SYNC output. A click or sync signal received through the CLICK input is fed to the 68B40 Timers (see above). The outputs are connected to the DIN socket driven by open-collector buffers. You will notice that the SYNC out 4 signal is the same as that of the CLOCK and of CLICK out.

4.12 CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION (continued)

4.12.11 Pin Connections for the 26-way Connector between the CMI-28 and CMI-29.

Pin 1	MIDI out A.	
Pin 2	+5 volts.	
Pin 3	MIDI in A.	
Pin 4	SYNC out 1.	
Pin 5	MIDI out B.	
Pin 6	SYNC out 2.	
Pin 7	MIDI in B.	
Pin 8	SYNC out 3.	
Pin 9	MIDI out C.	
Pin 10	Digital Ground.	
Pin 11	MIDI in C.	
Pin 12	Digital Ground.	
Pin 13	MIDI out D.	
Pin 14	RESET/START.	
Pin 15	MIDI in D.	
Pin 16	RUN/STOP.	
Pin 17	SMPTE code in.	
Pin 18	Digital Ground.	
Pin 19	SMPTE code out.	
Pin 20	CLICK out; SYNC out 4.	
Pin 21	CLICK in.	
Pin 22	(CMI29) Analog Ground.#	(CMI28) n/c.*
Pin 23	(CMI29) +15 volts.#	(CMI28) CPU Halt switch.*
Pin 24	(CMI29) -15 volts.#	(CMI28) Digital Ground.*
Pin 25	(CMI29) n/c.#	(CMI28) CPU Reset switch.*
Pin 26	(CMI29) n/c.#	(CMI28) Digital Ground.*

NOTES:

these connections are from the CMI29 board to the Audio Board only.
* these connections (from the CMI28 board only) are for debugging purposes only. If two push-button switches are connected between pins 23 & 24 and pins 25 & 26, they can be used to manually halt and reset the 68000 processor, respectively.

4.12 <u>CMI28 GENERAL INTERFACE - FUNCTIONAL DESCRIPTION</u> (continued)

4.12.12 Pin connections for the 20-way connector (CMI-29 board)

Pin 1 n/c
Pin 2 Analog ground
Pin 3 -15 volts
Pin 4 +15 volts
Pin 5 Analog ground
Pin 6 n/c
Pin 7 n/c
Pin 8 n/c
Pin 9 n/c
Pin 10 Analog ground
Pin 11 SMPTE in+
Pin 12 SMPTE in-
Pin 13 SMPTE out+
Pin 14 SMPTE out-
Pin 15 Click/Sync in
Pin 16 Analog ground
Pin 17 Sync-4 out
Pin 18 Analog ground
Pin 19 Key + (MIDI D in)
Pin 20 Key - (MIDI D in)

see corrections

GENERAL INTERFACE CARD (SMPTE/MIDI/SYNC) SOFTWARE DOCUMENTATION

Introduction

11 march 1985

The General Interface Card (CMI28) is an optional card for the Fairlight CMI Series IIx. Please note that for use in the Series IIx machines a CMI-25 rev-3 motherboard is required (or equivalently modified old motherboard). The General interface card is designed to handle reading and generating of SMPTE code and to control four MIDI ports as well as controlling the CLICK feature. Because its main purpose is for SMPTE and MIDI it is often referred to as the 'SMIDI Card'. In the Series IIx machines the card is plugged into the second slot on the left between the Master Card and a Channel Card.
The SMIDI card is connected to the General Interface Support Card, the CMI-29 via a 26-way cable. This card is housed in a box which is either bolted to the back of the CMI, or used as a remote unit. This support unit has the opto-couplers, and open collector buffers for receiving and transmitting MIDI as well as the analog circuitry for reading and generating SMPTE code to tape.

The SMIDI card in general is a microcomputer system with a Motorola 68000 microprocessor, either 8k or 16k words of ROM and either 8k or 32k words of static RAM (though it is possible to extend this to 64k words). It has a DMA interface to the CMI with the capability of DMAing to either P1 or P2. Though in the Series IIx machine the DMA is only on P1. The card has 4 ACIA's (68B50) for the 4 MIDI ports, has two 68B40 Programmable Timer Modules as well as associated circuitry for reading and generating SMPTE code and for Click/sync in and multiple Syncs out.

CMI to SMIDI Card communication - the control latch

There are two versions with different control latches. Revisions 1 and 2 of CMI28 have a set of 6 D-flip-flops (74LS174) while rev. 3 has an 8 bit addressable latch (74LS259). The reason for the different latches is that in rev.s 1 and 2 there is the possiblity of race conditions occurring if the latch is addressed by different processors whereas by using the 74LS259 only one bit can change at any one time so avoiding any race conditions.

Writing to the Control Latch (LS174) (rev.s 1 & 2)

Writing a byte to the location $FCA0 by the system 6809's, or to location $05FCA0 by the SMIDI processor itself, will set the control latch.
Bit 0: normally high, a low will enable the INT1 interrupt
Bit 1: normally high, a low will clear the SMIDI-CMI interrupt#
Bit 2: Sync switch; low -> Click In=Sync 4 out; high ->timer
Bit 3: 68000 Halt; low -> halt enabled, high -> unhalt
Bit 4: 68000 Reset; low -> reset enabled, high -> disable reset
Bit 5: normally high, low will enable the INT7 (NMI) interrupt
Bits 6-8 not used
#Note; the SMIDI to CMI interrupt on the Rev.s 1&2 is on P1 level 0 only. The rev.3 card has two SMIDI to CMI interrupts.

Writing to the Control Latch (LS259) (rev.3)

Writing a byte to the same location as above has the following

effect. Bits D0-D2 act as a bit address and bit D3 determines whether the bit addressed will be high or low.
$00 - INT1 low (lowest priority interrupt on 68000 enabled)
$08 - INT1 high
$02 - SMIDI to CMI interrupt P1 level 0 disable (high)
$0A - SMIDI to CMI interrupt P1 level 0 enable (low)
$04 - SMIDI to CMI interrupt P1 level 3 disable (high)
$0C - SMIDI to CMI interrupt P1 level 3 enable (low)
$06 - Sync switch low - Sync Out 4 = Click/Sync In directly
$0E - Sync switch high - Sync Out 4 = (Click/Sync In)/timer
$03 - INT7 low (68000 NMI enabled)
$0B - INT7 high
$05 - HALT low (68000 held in halt mode - halt LED on)
$0D - HALT high (unhalt - LED off)
$07 - RESET low (68000 held in reset mode - reset LED on)
$0F - RESET high (release from reset - LED off)

Memory Configuration

There are four 28-pin sockets for ROM and static RAM. The minimum configuration is 8k words of ROM (2x2764)and 8k words of static RAM (2x6264). The ROM can be configured for 16k words by breaking the link (LK1) between pin 27 of the ROM and +5v and join the link LK1 to A15 from the processor and plugging in the appropriate two 27128's. Similarly the RAM can be arranged to accommodate 32k static RAM chips (e.g.MK4856 pseudo-statics) by breaking the links LK2 and LK3 to +5v and connecting A14 and A15 to pins 26 and 1 of the RAM chips (via LK3 and LK2), respectively. Further, there is an option for 64k words of RAM; by soldering two 32k RAM chips on top of each other except for pin-20, the chip select, which should be connected to the pads provided from the select circuitry, the AND gates (D12). All these memory expansions will depend on the availability of these chips.
Note; when plugging in the ROM's, they should be labelled 'odd' and 'even'; the even one should be plugged into E5,6 (near the 68000) and the odd one into E8 (between the RAM chips).
Memory addressing: ROM starts at $000000 and RAM starts at $080000.

68000/6809 DMA Bus Interface

Communication between the 68000 processor and the 6809 CPU is achieved by DMA (Direct Memory Access) on the system bus. The 68000 waits until no higher priority device is occupying the bus and then either 6809 (P1 or P2) is temporarily hung while the 68000 executes a normal bus cycle writing to or reading from memory or a peripheral on the bus. In this manner the entire 64K address space of each 6809 processor appears as a small slice of the 16 megabyte address space of the 68000. Software then defines various protocols for the different processors to pass messages and data to one another by simply placing them in system memory.
The DMA interface provided on the 68000 SMPTE/MIDI Card is a very flexible one. It automatically handles either 8 or 16 bit data transfers (doing double cycles across the 8-bit CMI bus in the latter case) and can do so on either P1 or P2 cycles, selecting any desired memory mapping which has been set up on the Q256 memory card.
DMA is initiated by the 68000 when it accesses any address in the

range $040000 to $04FFFF (ie. P2 DMA -not for Series II) and $050000 to $05FFFF (ie. P1 DMA).

SMPTE/MIDI Card Peripheral Circuits

There are four different peripheral circuits on the SMIDI card. Firstly there is the four ACIA's (G7-11) which are the MIDI ports A,B,C, and D. Then there is the Timer (bottom rev.2 G5,6) which is used in conjuction with the SMPTE read and generate circuits (which are the other two circuits) as well as the Click In and Out.
The ACIA's and Timers work from an 8-bit data bus with (asynchronous) interfacing circuitry. They are driven also by the E (enable) signal from the 68000. The frequency of this clock is one-tenth of the 68000 clock (10MHz) with a 60/40 duty cycle (6 clocks high, 4 clocks low)

The ACIA's are selected by E10 and appear at addresses; $60020, $60030, $60040 and $60050. They share a common interrupt level - level 3 ($\overline{INT3}$). Their transmit and receive data lines are wired to the 26-way connector to be connected to the MIDI drivers and opto-coupler receivers.
The programmable timers (G5,6) appear at the general address $60000, and $60010 and have an interrupt level-2 (INT2) to the 68000.

Interrupts

The 68000 has seven levels of interrupts. The priority for the interrupts is made by hardware through the 74LS148 ic (C2). The lowest level interrupt ($\overline{INT1}$) and the NMI ($\overline{INT7}$) are enabled and cleared by the CMI through the control port (B4). $\overline{INT2}$ is for the 68B40 Timer, $\overline{INT3}$ is for the ACIA's. $\overline{INT4}$ (rev.2, $\overline{INT6}$ for rev. 3) is for reading a SMPTE 'one' and $\overline{INT5}$ for a SMPTE 'zero' and are cleared by addressing location $\overline{SMPTERD}$ on i.c. E10 (LS138). $\overline{INT6}$ (rev.2 INT4 for rev. 3) is for $\overline{SMPTE}$ generation and is cleared by writing to the shift registers (C7,C8), i.e. by signal $\overline{SMPTEWR}$.

SMPTE Generating Circuitry

An oscillator (3.84MHz) is divided by 10 (G2,G1) to provide a standard for generating the 3 different rates of SMPTE code (24,25 and 30 fps) All three are denominators of 384,000. Further division, depending on the frame rate selected, is done by the Timer (G5,6) giving the signal CLK2, which is the bit rate for a SMPTE 'one' (ie 160 bits per frame).This is in turn divided by 2 (C11) giving CLK1 which is the bit rate for a SMPTE 'zero' (ie 80 bits per frame). When a SMPTE word is ready it is written to the Parallel-In-Serial-Out registers (C7,C8) at address $60070 (through B9,B8 and B7). When this writing takes place the interrupt $\overline{INT6}$ if it has been asserted is now cleared. The data in the shift registers (C7,C8) is clocked out by CLK1, a 4-bit counter (D11)is also clocked which causes the interrupt on level-6 (INT6) when it reaches its terminal count of 16. Now, if a 'zero' is shifted out from C8 the flip-flop C11 is toggled at the rate determined by CLK2, but if a 'one' is shifted out from C8

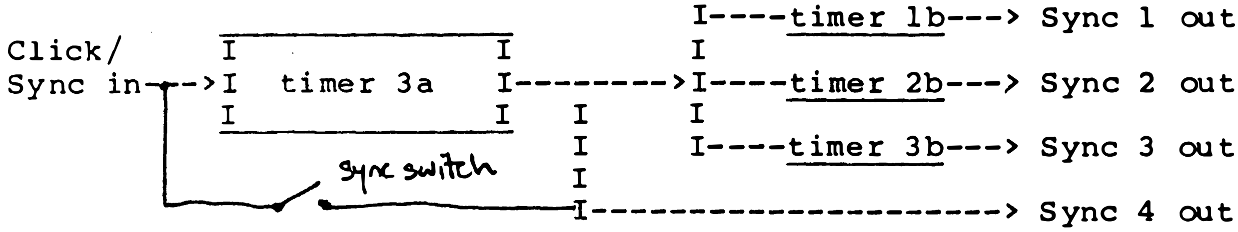
the flip-flop C11 is toggled at the CLK1 rate. Thus, the word stored on the shift registers is outputted in SMPTE form.
Timer 1a is used for the SMPTE reading. It is configured for single-shot double 8-bit mode MSB=$01 and LSB=$C2 (24fps); =$BB (25fps); =$9B (30fps). Timer 2a is used for SMPTE generation and is configured in the continuous 16-bit mode: MSB=$00 LSB=$31 (24fps); =$2F (25fps); =$25 (30fps).

SMPTE Reading Circuitry (Refer to Timing Diagram)

SMPTE code coming from tape, being converted to TTL signal levels by the CMI-29 board, is received by the CMI-28 and fed through a data separator.
The required output from this SMPTE data separator is to have one interrupt occur for every SMPTE 'one' read and another interrupt for every SMPTE 'zero' read. This process can be followed through with the timing diagrams. The 68B40 timer is set, according to the frame rate of the SMPTE being read, to 3/4 of the time for one bit cell. The circuit then detects whether there has been a transition in that time or not. If there has been a transistion then a 'one' is read, if no transition occurred then a 'zero' is read.
On the CMI28's rev.s 1 & 2 the SMPTE generate has the priority over the SMPTE read. A SMPTE 'one' read triggers INT4 and a SMPTE 'zero' read triggers INT5 while the SMPTE generation is on INT6 - i.e. at a higher priority to the read. It was found that the read would lose frames if SMPTE generation was occurring at the same time. Interrupts INT4 and INT6 were swapped for rev. 3 of the CMI28 and the problems with the read were avoided.

Sync In and Outs

The SMIDI card also takes care of some of the sync-ing functions of the system. On revisions 1 (modified to rev.2) and 2 of the CMI-28 SIDI board there are two 68B40 programmable timers (each with 3 timers inside), one wired on top of another. The input clock of 3rd timer in the bottom 68B40 (timer a) is wired to the Click/Sync input socket (on support box). The output of this timer is fed into the inputs of the three timers in the top i.c. (timer b) providing a cascaded timer system. These four outputs are fed to the CMI-29 in the support box, to a 5-pin DIN socket via open-collector buffers.



Note; timer 1a and timer 2a
are used for SMPTE

General Interface Support Card CMI-29

This circuit board contains the analog circuitry required for the i/o for SMPTE and MIDI. There are 3 MIDI inputs (A, B & C) and 4 MIDI outputs(A, B, C & D), though there is provision for a fourth MIDI input (D).

There are two other output (5-pin DIN) sockets. One is the CLOCK output, containing the CLOCK, RESET/START and RUN/STOP TTL compatible signals. This CLOCK output is designed to control Roland drum machines, etc. The other is the multiple SYNC output. A click or sync signal received through the CLICK input is fed to the 68B40 Timers (see above). The outputs are connected to the DIN socket driven by open-collector buffers. You will notice that the SYNC out 4 signal is the same as that of the CLOCK and of CLICK out.

QUICK REFERENCE FOR SMIDI ADDRESS MAP

$000000-$002000	- EPROM (8k words)
$020000-$03FFFF	- unused
$040000-$04FFFF	- P2 DMA (available on Series III not IIx)
$050000-$05FFFF	- P1 DMA
$060000-$06000F	- Timer A - SMPTE read & write
$060010-$06001F	- Timer B - Multi-sync
$060020-$06002F	- ACIA 1
$060030-$06003F	- ACIA 2
$060040-$06004F	- ACIA 3
$060050-$06005F	- ACIA 4
$060060-$06006F	- SMPTE Read Interrupt Clear
$060070-$06007F	- SMPTE Write Shift Register Load & INT Clear
$080000-$082000	- RAM (8k words)
$0A0000-$0BFFFF	- INTC - interrupt CMI (Series IIX only)
$0C0000-$0DFFFF	- SMPRST (for drum machines, etc)
$0E0000-$0FFFFF	- SMPR/H (for drum machines, etc)

Note: When reading and writing from the peripherals remember that they are on an 8 bit bus and word operations need to be done in separate byte instructions or MOVEP instructions. Note in particular that when reading from a peripheral an ODD address must be used, though for writes ODD or EVEN addresses are o.k.

FAIRLIGHT. 21feb85

MODIFICATIONS TO THE CMI-28 rev2. BOARD

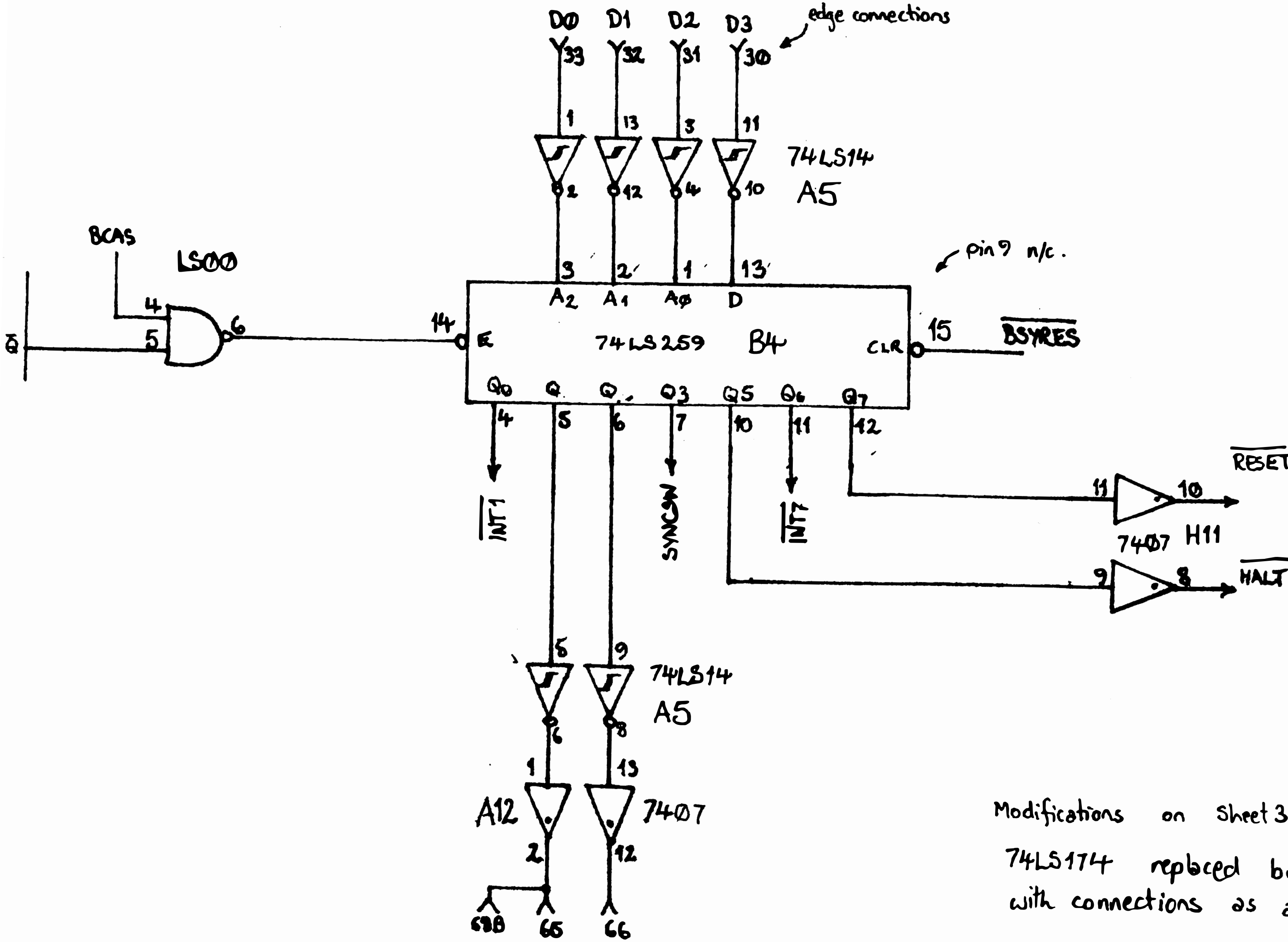
1/ I.C. D12 [illegible] 74LSØ2 on the rev.2 design. It now should be a 74LS32 with pin connections as per sheet 2.

2./. I.C. B4 (at present a 74LS174) should be a 74LS259 as per the attached sheet. Flip flop D1Ø(b) is no longer needed.

3/. Interrupt levels $\overline{INT4}$ and $\overline{INT6}$ have been swapped.
Now $\overline{INT4}$ should be connected to D1Ø (74LS74) pin 6
and $\overline{INT6}$ should be connected to E1 (74LSØØ) pin 3

4/. The biggest (and worse) mod is adding an extra 68B40 timer. At present we solder one on top of another (G5,6)
Three outputs of the extra timer are connected to the 26way connector
Also extra gates have been added to the in and out of the original timer; note the addition of the two tri-state gates (LS125) and one exclusive-OR gate (LS86)
refer to sheet 6.

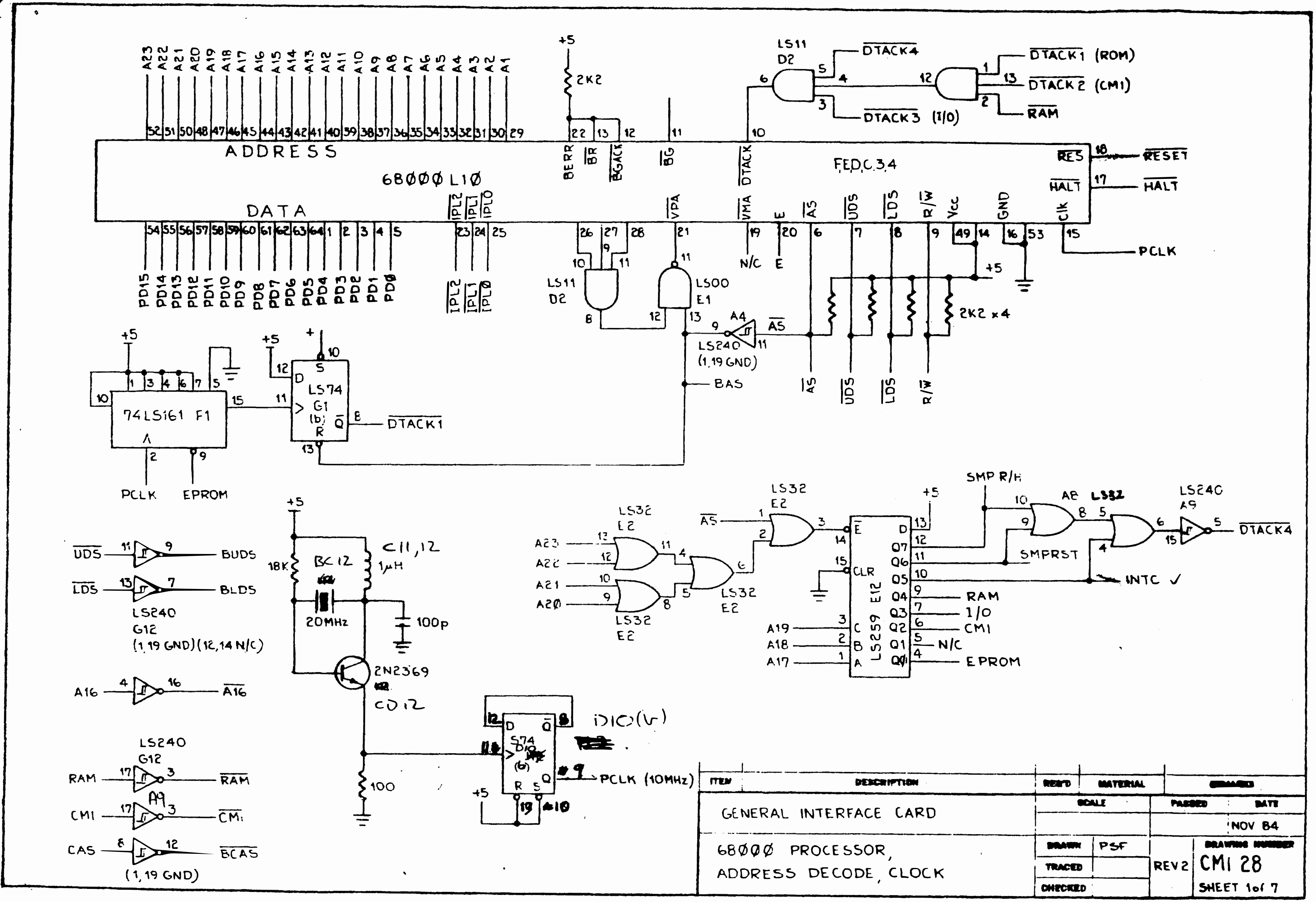
5/. Change the revision number to REV 3



Modifications on Sheet 3 CMI-28 rev 2.

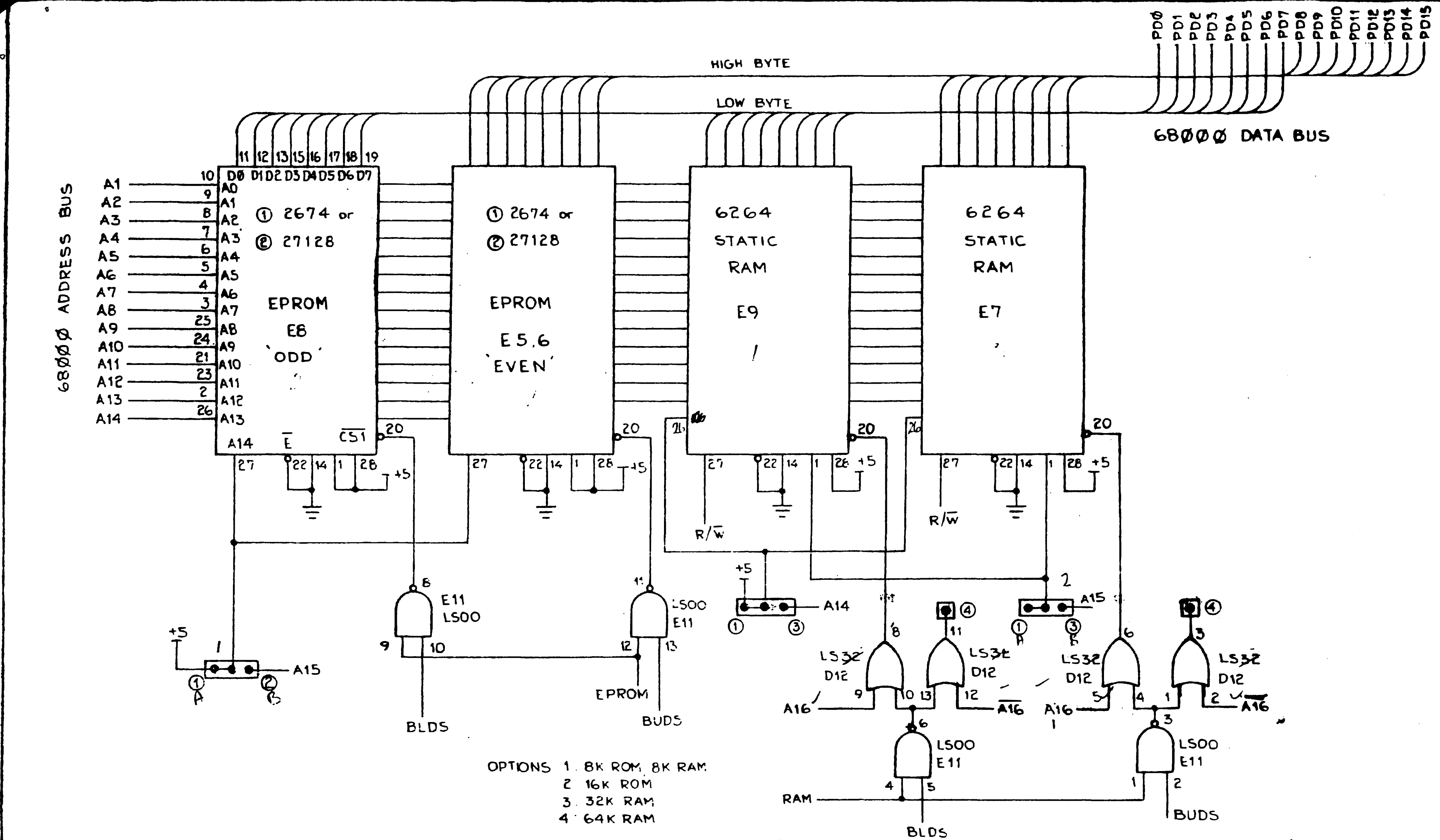
74LS174 replaced by 74LS259 with connections as above.

Flip-flop D10(b) not required nor is signal IN



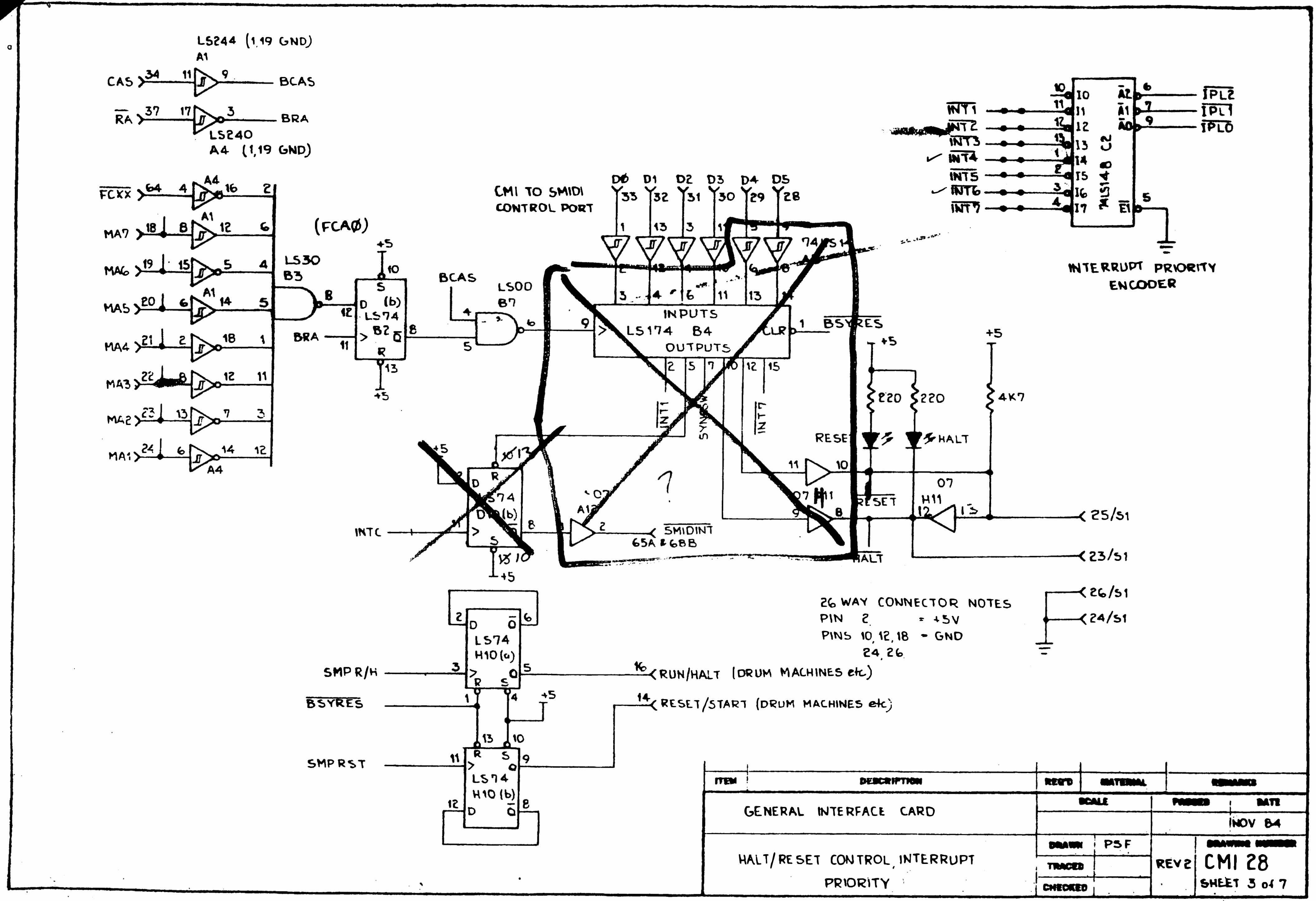
ITEM	DESCRIPTION	REQ'D	MATERIAL	
GENERAL INTERFACE CARD		SCALE	PASSED	DATE NOV 84
68000 PROCESSOR, ADDRESS DECODE, CLOCK		DRAWN PSF	REV 2	DRAWING NUMBER CMI 28
		TRACED		
		CHECKED		SHEET 1 of 7

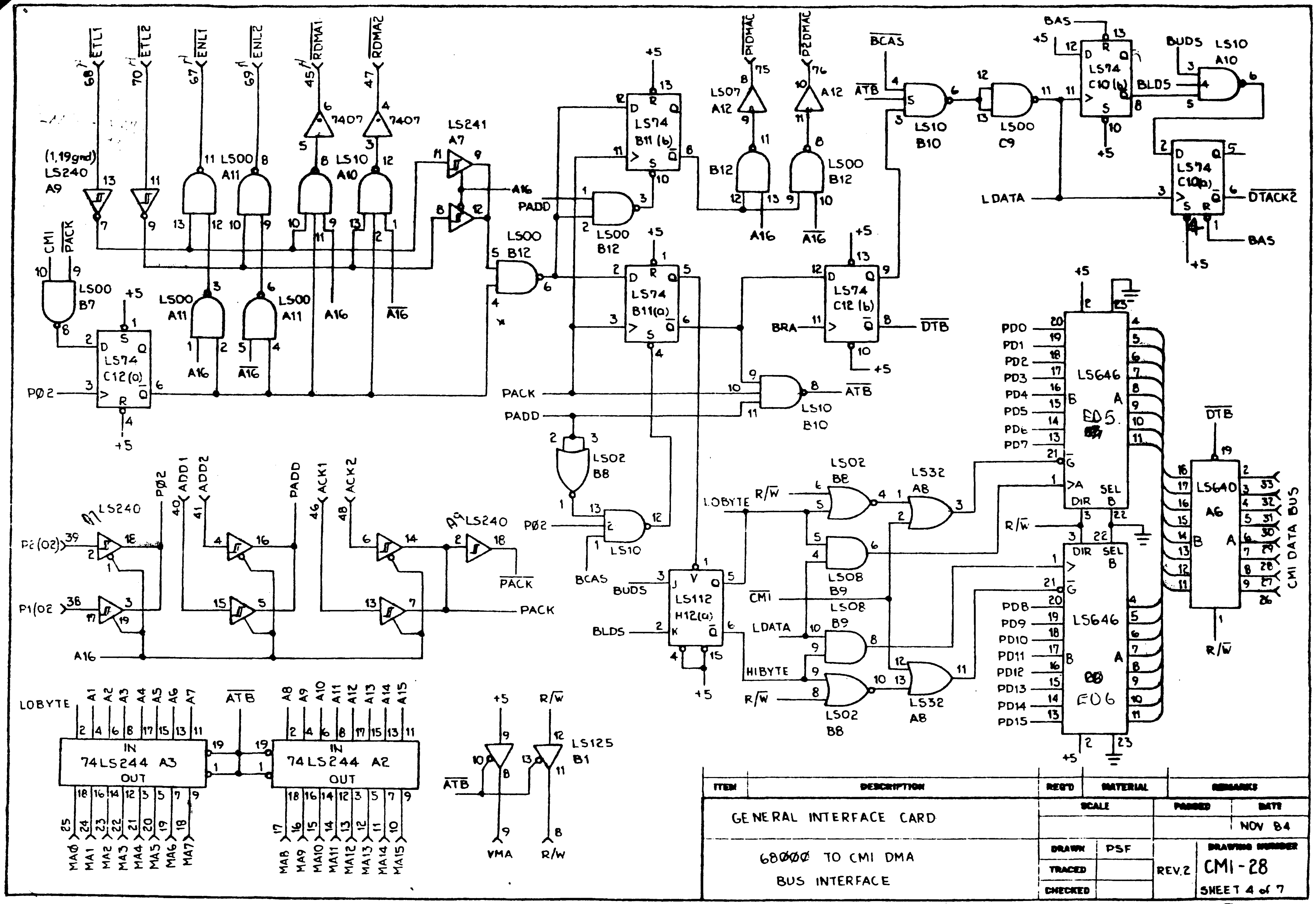
A3



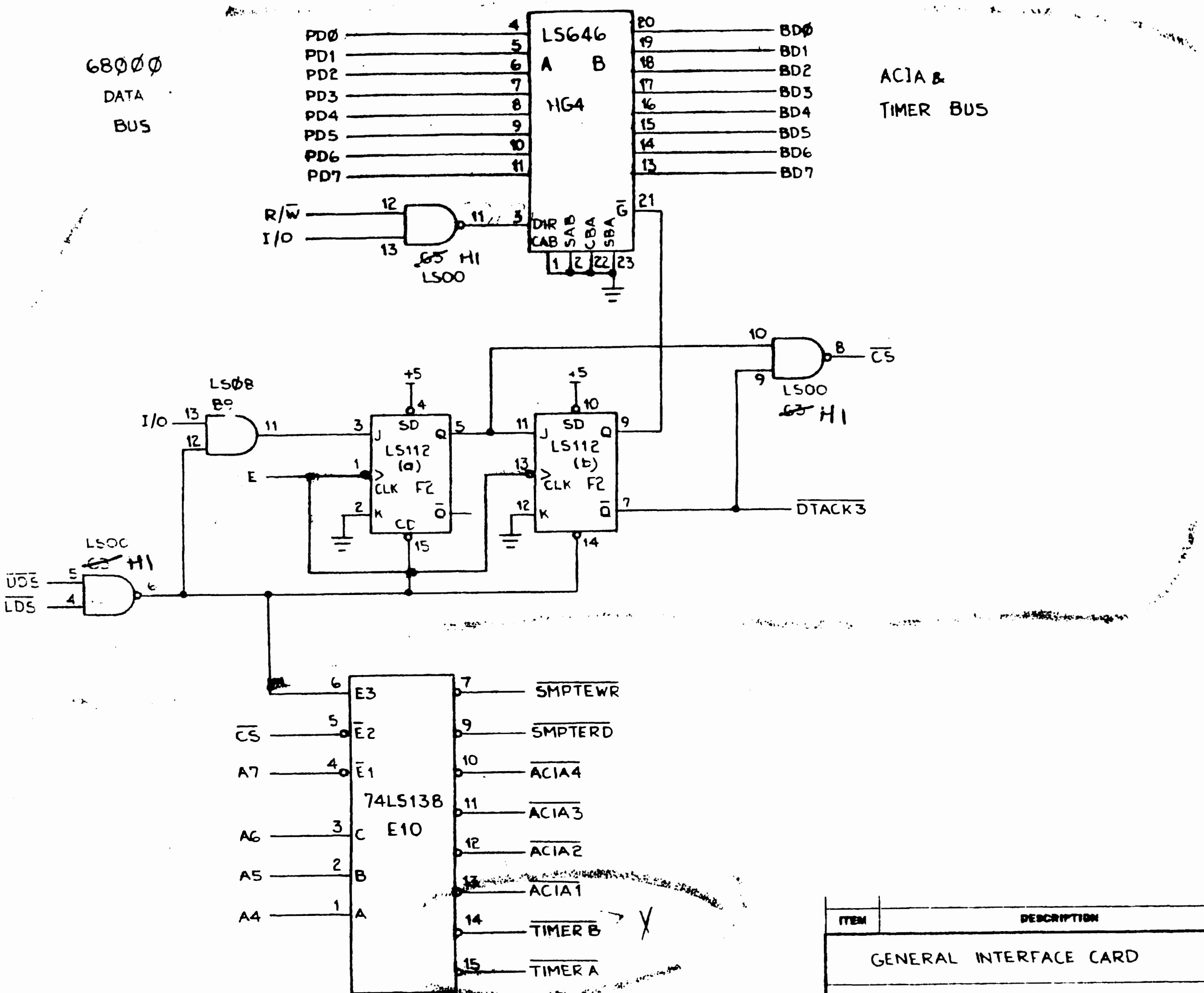
ITEM	DESCRIPTION	REQ'D	MATERIAL	REMARKS
	GENERAL INTERFACE CARD	SCALE	PASSED	DATE: NOV 84
	MEMORY	DRAWN: PSF / TRACED / CHECKED	REV 2	DRAWING NUMBER: CMI-28 SHEET 2 of 7

A3


LS244 (1,19 GND)
A1
CAS
BCAS
RA
BRA
LS240
A4 (1,19 GND)
FCXX
MA7
MA6
MA5
MA4
MA3
MA2
MA1
(FCAØ)
LS30
B3
LS74
B2
BCAS
BRA
LS00
B7
CMI TO SMIDI
CONTROL PORT
D0 D1 D2 D3 D4 D5
INPUTS
LS174 B4
OUTPUTS
CLR
BSYRES
74LS14
INT1
SYNCIN
INT7
RESET
HALT
220
220
4K7
+5
INTC
SMIDINT
65A & 68B
25/S1
23/S1
26/S1
24/S1
INT1
INT2
INT3
INT4
INT5
INT6
INT7
MLS148 C2
IPL2
IPL1
IPL0
INTERRUPT PRIORITY
ENCODER
26 WAY CONNECTOR NOTES
PIN 2 = +5V
PINS 10, 12, 18 - GND
24, 26
LS74
H10 (a)
SMP R/H
BSYRES
SMPRST
LS74
H10 (b)
RUN/HALT (DRUM MACHINES etc)
RESET/START (DRUM MACHINES etc)
ITEM
DESCRIPTION
REQ'D
MATERIAL
REMARKS
GENERAL INTERFACE CARD
SCALE
PASSED
DATE
NOV 84
DRAWN PSF
TRACED
CHECKED
DRAWING NUMBER
HALT/RESET CONTROL, INTERRUPT PRIORITY
REV 2
CMI 28
SHEET 3 of 7


GENERAL INTERFACE CARD
68000 TO CMI DMA
BUS INTERFACE
ITEM
DESCRIPTION
REQ'D
MATERIAL
REMARKS
SCALE
PASSED
DATE
NOV 84
DRAWN PSF
TRACED
CHECKED
REV.2
DRAWING NUMBER
CMI-28
SHEET 4 of 7
CMI DATA BUS
74LS244 A3
74LS244 A2
LS646
LS640 A6
LS125 B1

Error



ITEM	DESCRIPTION	REQ'D	MATERIAL	REMARKS

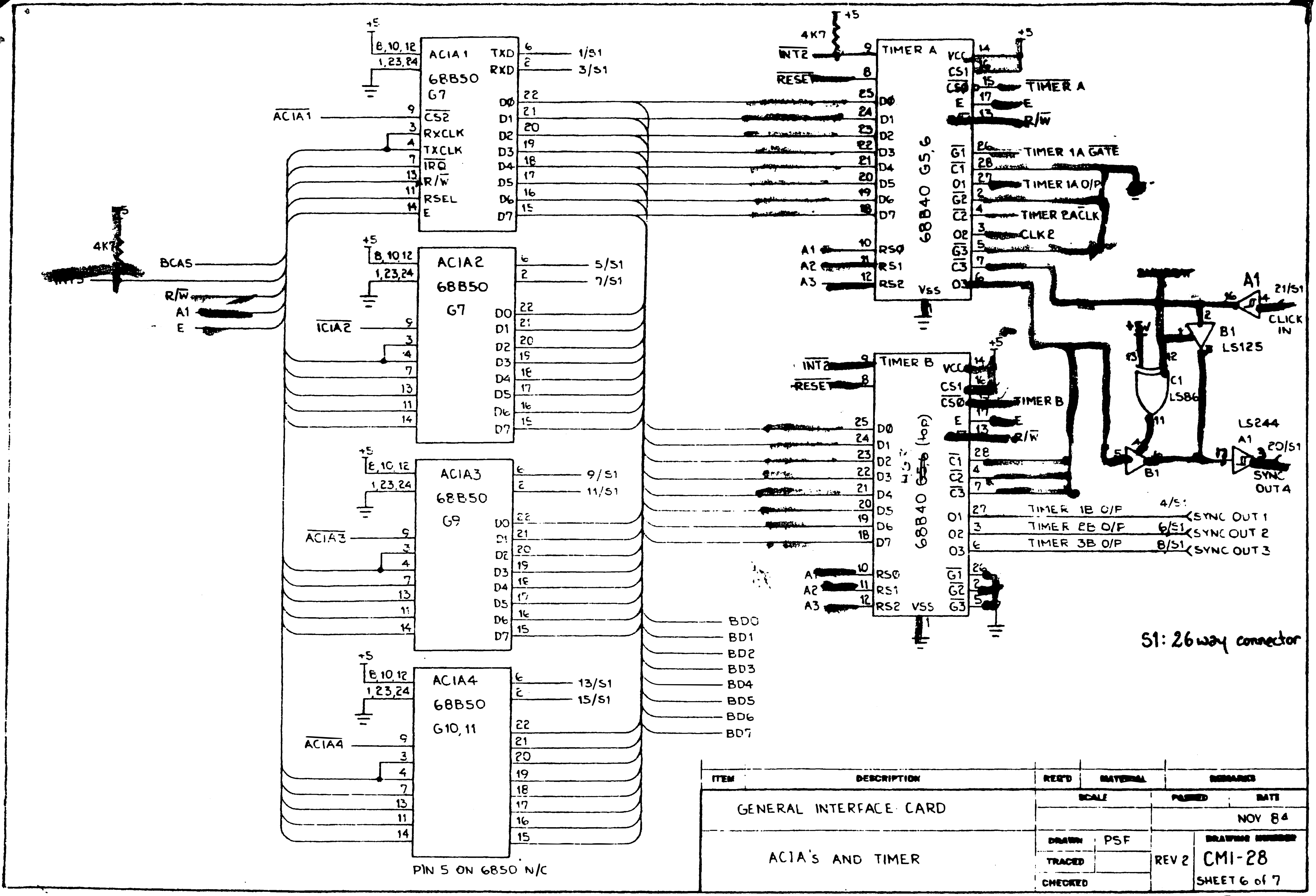
GENERAL INTERFACE CARD

PERIPHERAL SELECT

SCALE	PASSED	DATE
		NOV 84

DRAWN	PSF		DRAWING NUMBER
TRACED		REV.2	CMI-28
CHECKED			SHEET 5 of 7

A3



ITEM	DESCRIPTION	REQ'D	MATERIAL	REMARKS

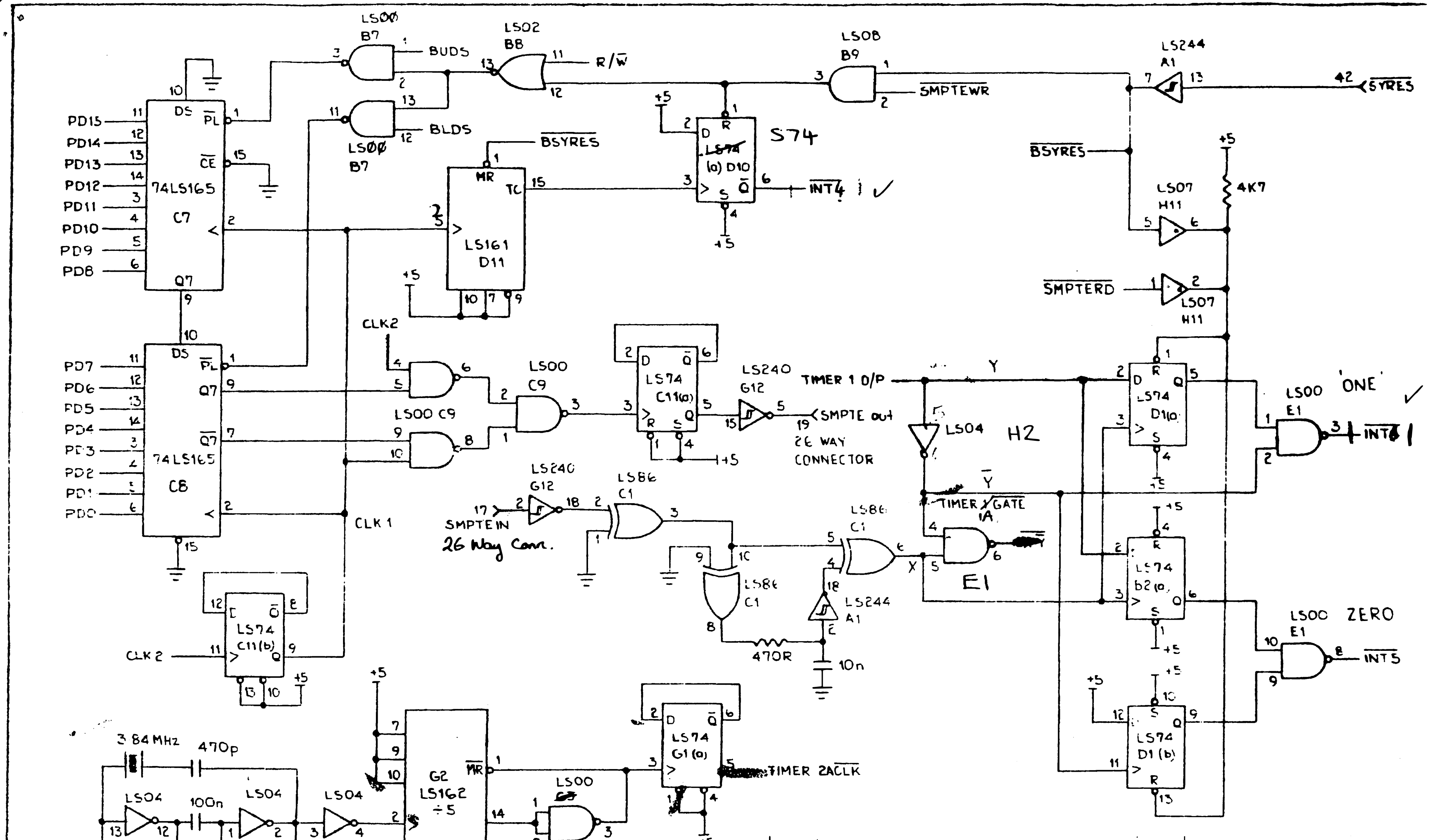
GENERAL INTERFACE CARD

ACIA's AND TIMER

SCALE	PASSED	DATE
		NOV 84

DRAWN: PSF
TRACED:
CHECKED:
REV 2
DRAWING NUMBER: CMI-28
SHEET 6 of 7

A3



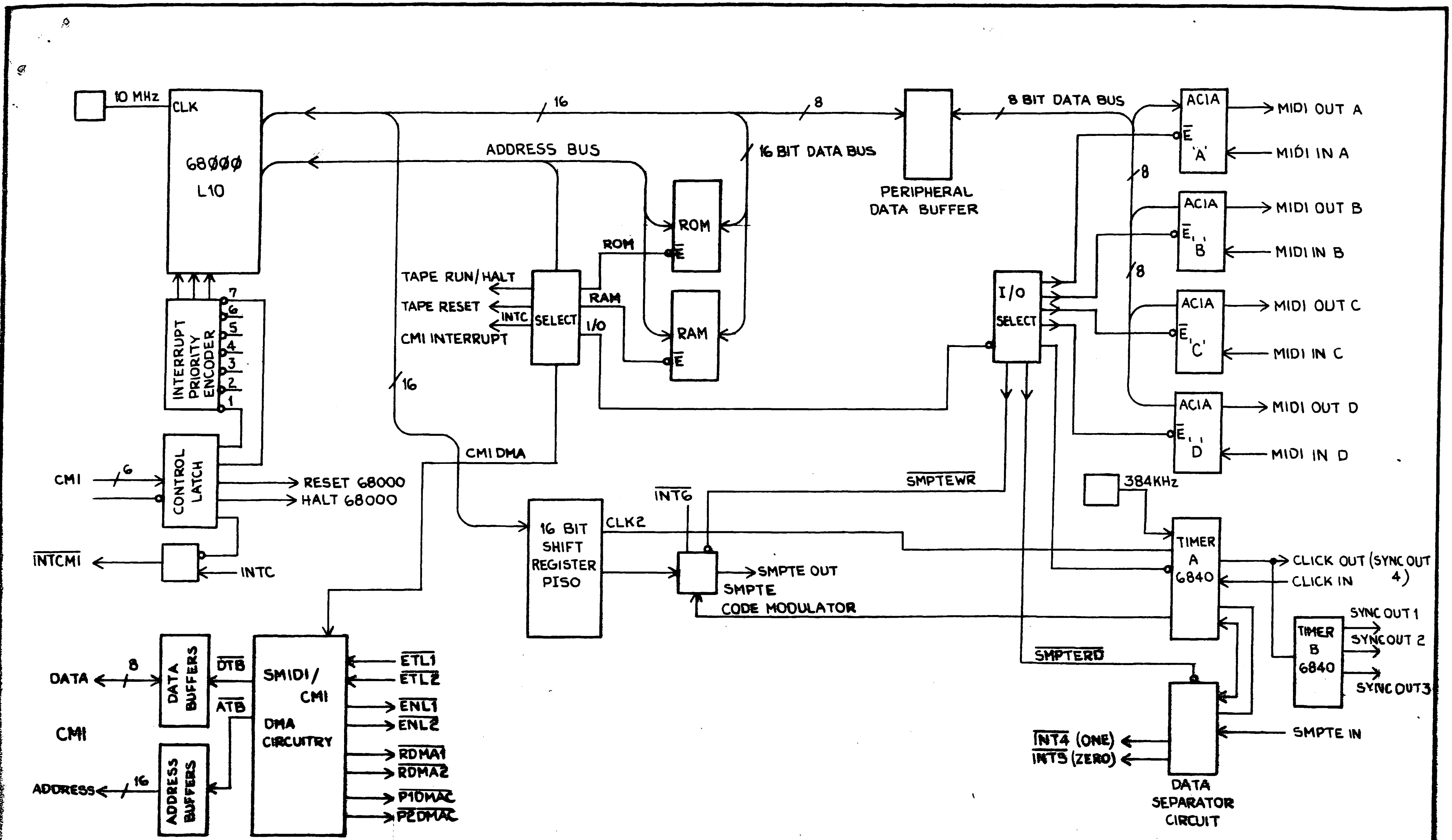
ITEM	DESCRIPTION	REQ'D	MATERIAL	REMARKS

GENERAL INTERFACE CARD

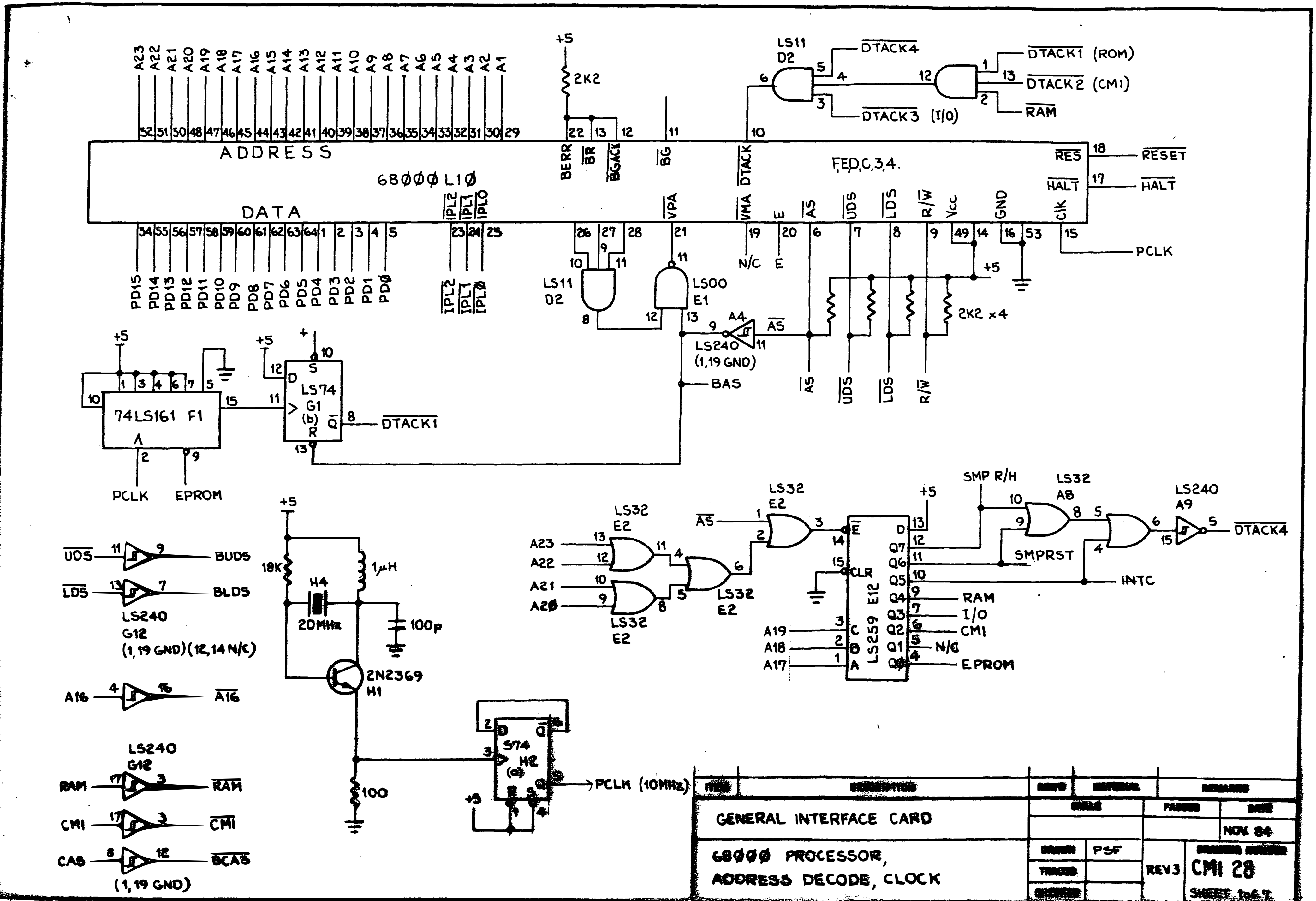
SMPTE CODE GENERATOR DATA SEPARATOR AND READER

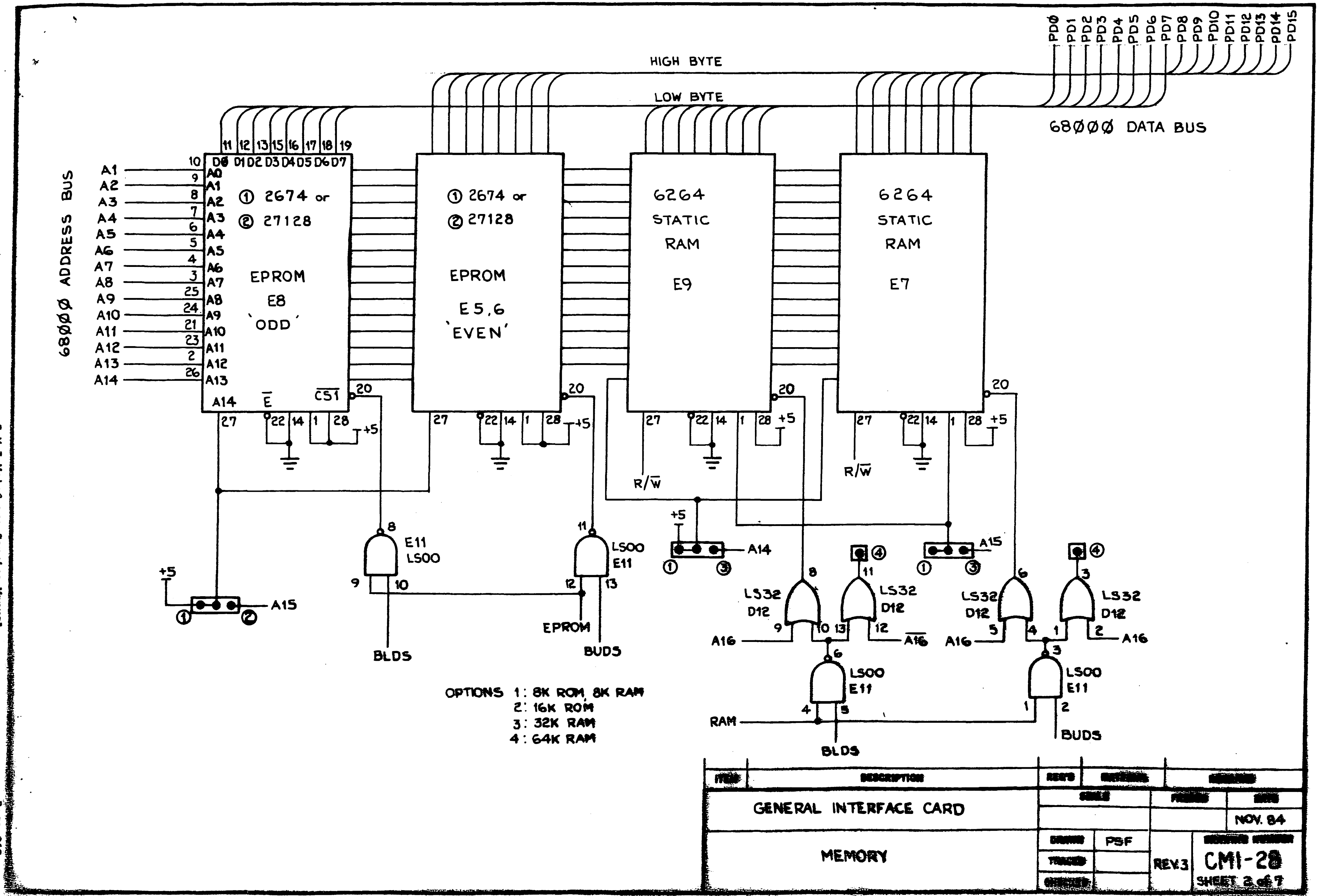
SCALE	
PASSED	
DATE	NOV 84
DRAWN	PSF
TRACED	
CHECKED	
REV	REV 2
DRAWING NUMBER	CMI - 28
SHEET	7 of 7

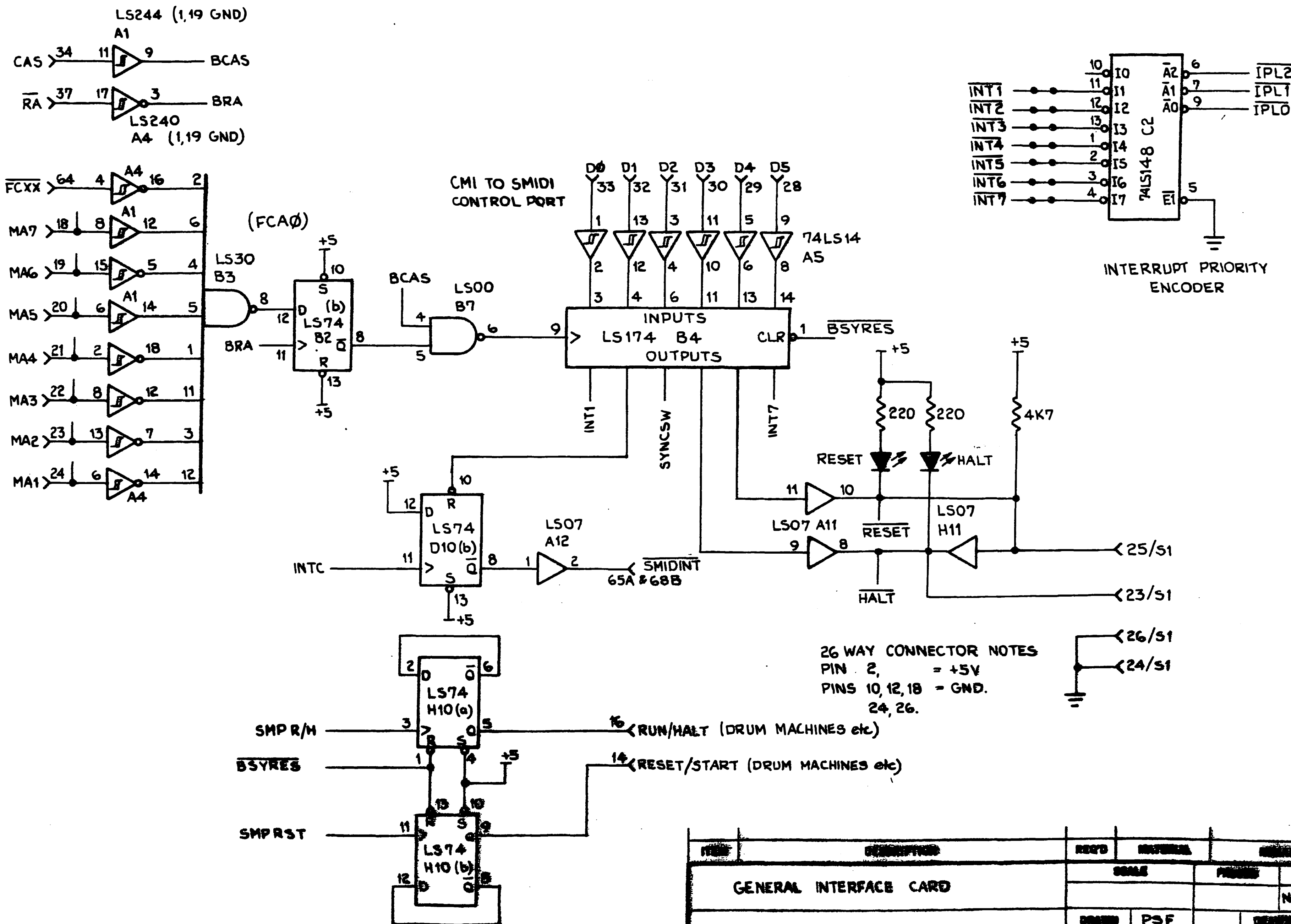
A3



GENERAL INTERFACE CARD	NOV. 84
BLOCK DIAGRAM.	DRAWN: PSF
	REV.3 CMI BD-1


68000 L10
ADDRESS
DATA
A23 A22 A21 A20 A19 A18 A17 A16 A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1
PD15 PD14 PD13 PD12 PD11 PD10 PD9 PD8 PD7 PD6 PD5 PD4 PD3 PD2 PD1 PD0
IPL2 IPL1 IPL0
BERR BR BGACK BG DTACK VPA VMA E AS UDS LDS R/W Vcc GND Clk
RES HALT
RESET
HALT
PCLK
FED,C,3,4.
+5
2K2
2K2 ×4
LS11 D2
DTACK4
DTACK3 (I/O)
DTACK1 (ROM)
DTACK2 (CMI)
RAM
N/C
LS00 E1
A4 LS240 (1,19 GND)
BAS
74LS161 F1
LS74 G1 (b)
DTACK1
PCLK
EPROM
UDS
BUDS
LDS
BLDS
LS240 G12 (1,19 GND)(12,14 N/C)
A16
RAM
CMI
CAS
BCAS
(1,19 GND)
18K
H4
20MHz
1μH
100p
2N2369 H1
100
S74 H2 (a)
PCLK (10MHz)
A23 A22 A21 A20
LS32 E2
SMP R/H
LS32 A8
SMPRST
INTC
LS240 A9
LS259 E12
A19 A18 A17
RAM
I/O
CMI
N/C
EPROM
GENERAL INTERFACE CARD
68000 PROCESSOR, ADDRESS DECODE, CLOCK
NOV 84
PSF
REV3
CMI 28

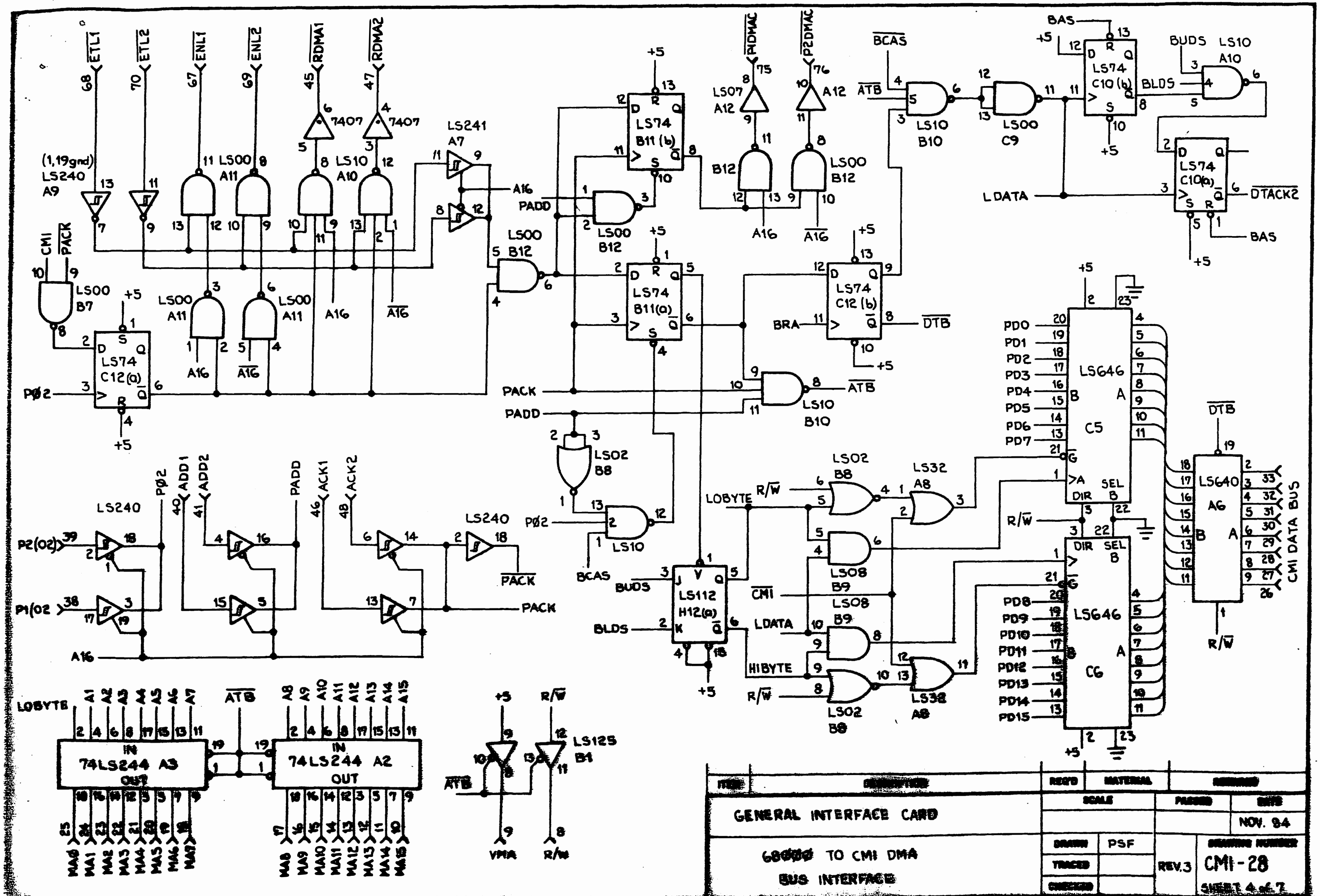

HIGH BYTE
LOW BYTE
68000 DATA BUS
PD0 PD1 PD2 PD3 PD4 PD5 PD6 PD7 PD8 PD9 PD10 PD11 PD12 PD13 PD14 PD15
68000 ADDRESS BUS
A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11 A12 A13 A14
① 2674 or
② 27128
EPROM
E8
'ODD'
EPROM
E5,6
'EVEN'
6264
STATIC
RAM
E9
E7
R/W
+5
A14
A15
A16
E11
LS00
LS32
D12
EPROM
BLDS
BUDS
RAM
OPTIONS 1: 8K ROM, 8K RAM
2: 16K ROM
3: 32K RAM
4: 64K RAM
GENERAL INTERFACE CARD
MEMORY
NOV. 84
PSF
REV.3
CMI-28
SHEET 2 of 7

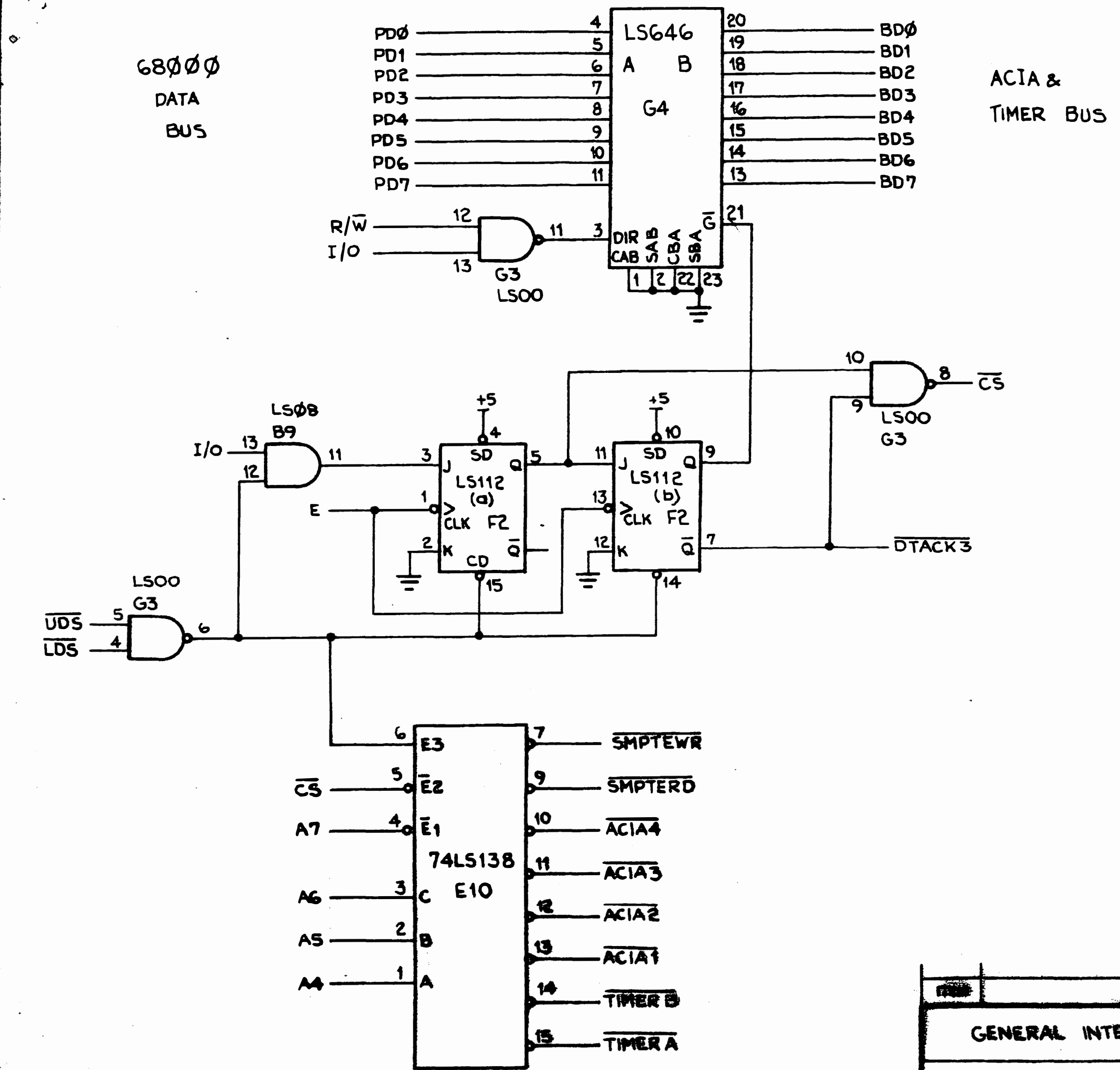


GENERAL INTERFACE CARD

HALT/RESET CONTROL, INTERRUPT PRIORITY

DATE	NOV. 84
DRAWN	PSF
	REV 3
DRAWING NUMBER	CMI 28
SHEET	3 of 7

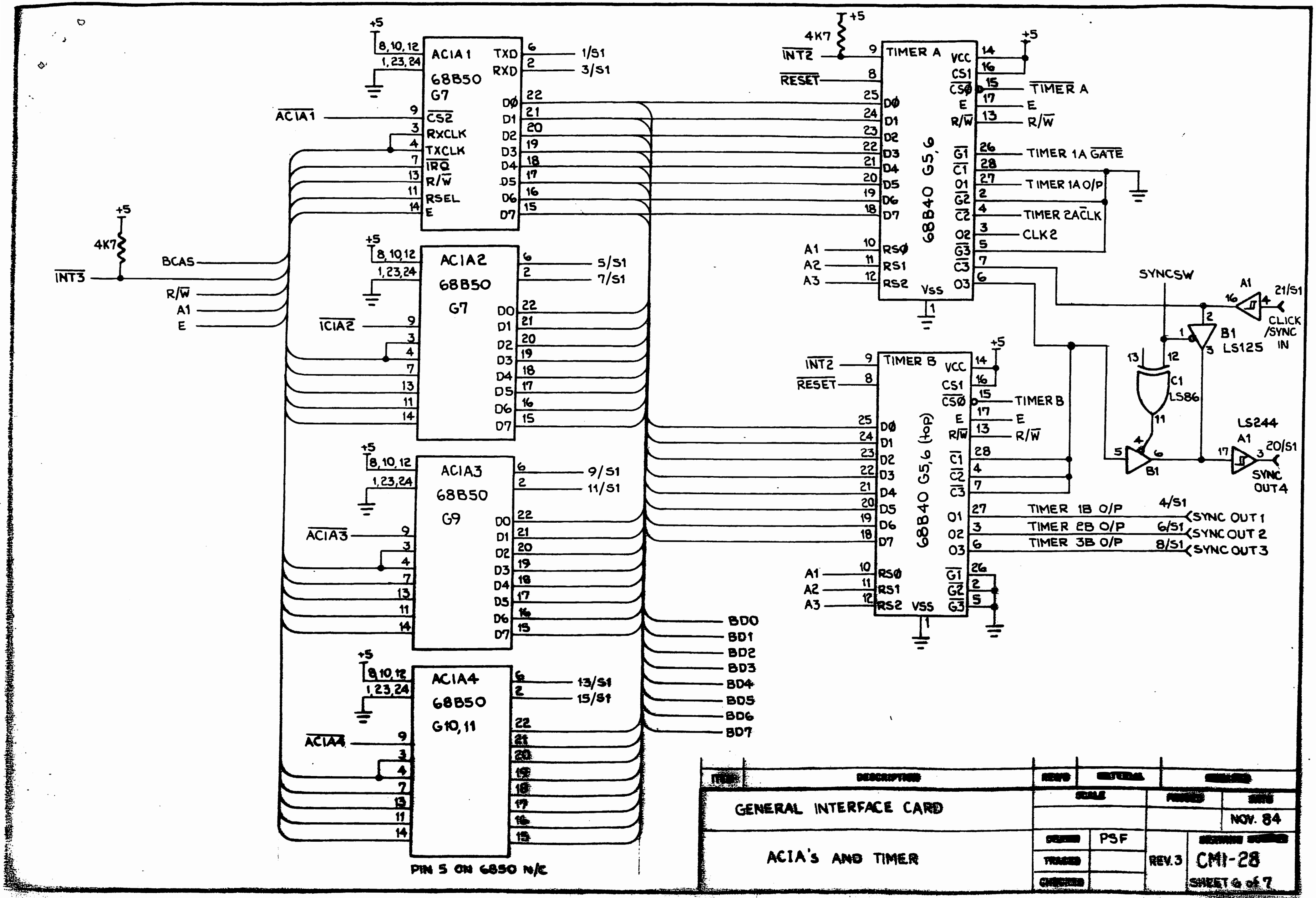

ETL1
ETL2
ENL1
ENL2
RDMA1
RDMA2
P1DMAC
P2DMAC
BCAS
ATB
BAS
BUDS
BLDS
LDATA
DTACK2
(1,19gnd) LS240 A9
LS00 A11
LS10 A10
7407
LS241 A7
A16
PADD
PACK
LS74 B11 (b)
LS74 B11 (a)
LS00 B12
LS07 A12
LS10 B10
LS00 C9
LS74 C10 (b)
LS74 C10 (a)
LS74 C12 (a)
LS74 C12 (b)
CMI PACK
LS00 B7
PØ2
BRA
DTB
LS02 B8
LS10
LS112 H12(a)
LOBYTE
HIBYTE
R/W
CMI
LS08 B9
LS32 A8
LS646 C5
LS646 C6
LS640 A6
CMI DATA BUS
PD0 PD1 PD2 PD3 PD4 PD5 PD6 PD7
PD8 PD9 PD10 PD11 PD12 PD13 PD14 PD15
LS240
P2(02)
P1(02
ADD1
ADD2
ACK1
ACK2
74LS244 A3
74LS244 A2
A1 A2 A3 A4 A5 A6 A7
A8 A9 A10 A11 A12 A13 A14 A15
MA0 MA1 MA2 MA3 MA4 MA5 MA6 MA7
MA8 MA9 MA10 MA11 MA12 MA13 MA14 MA15
LS125 B1
VMA
R/W
+5
ITEM
DESCRIPTION
REQ'D
MATERIAL
REMARKS
SCALE
PASSED
DATE
GENERAL INTERFACE CARD
NOV. 84
DRAWN PSF
TRACED
CHECKED
DRAWING NUMBER
68000 TO CMI DMA
BUS INTERFACE
REV.3
CMI-28
SHEET 4 of 7

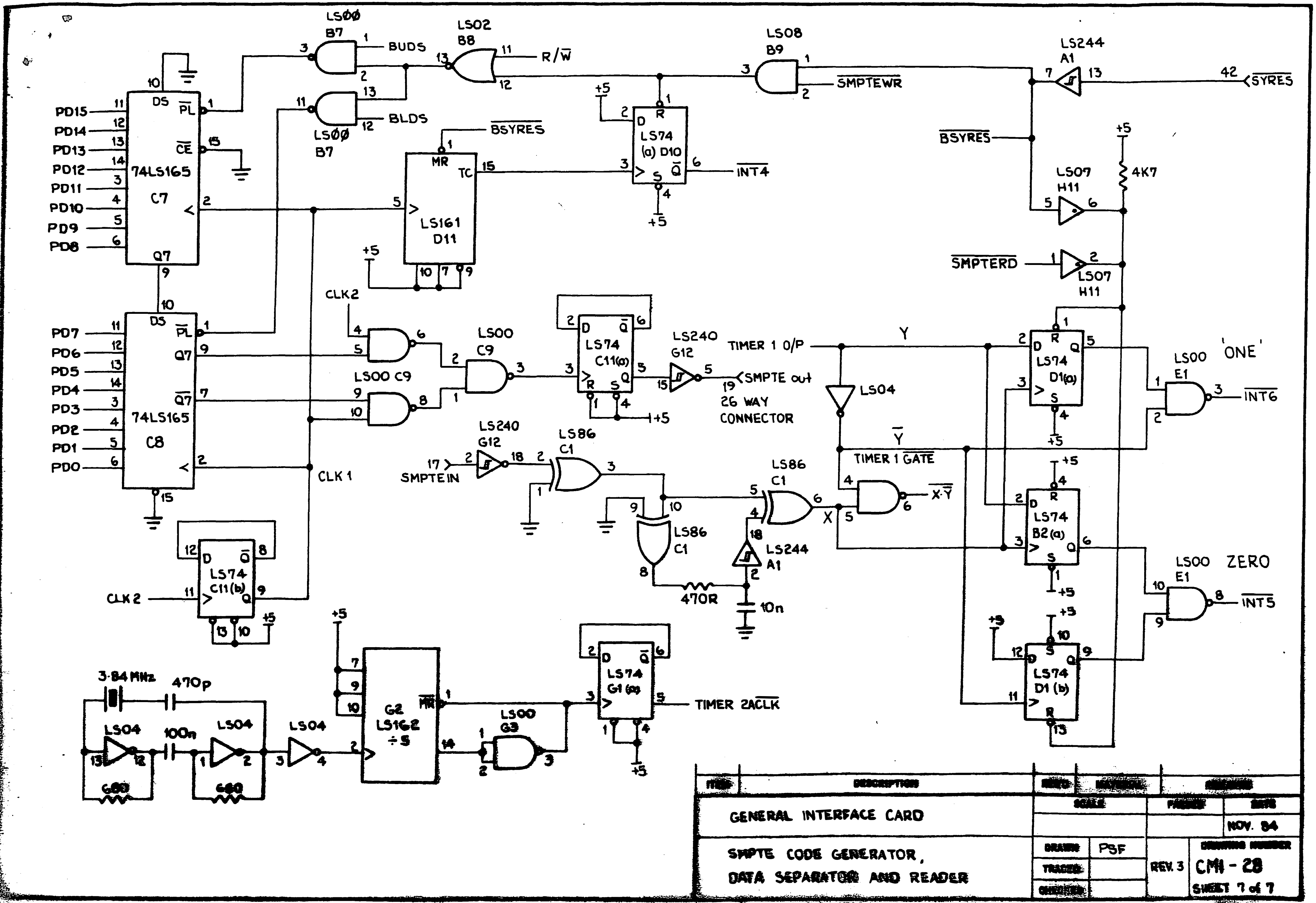


GENERAL INTERFACE CARD

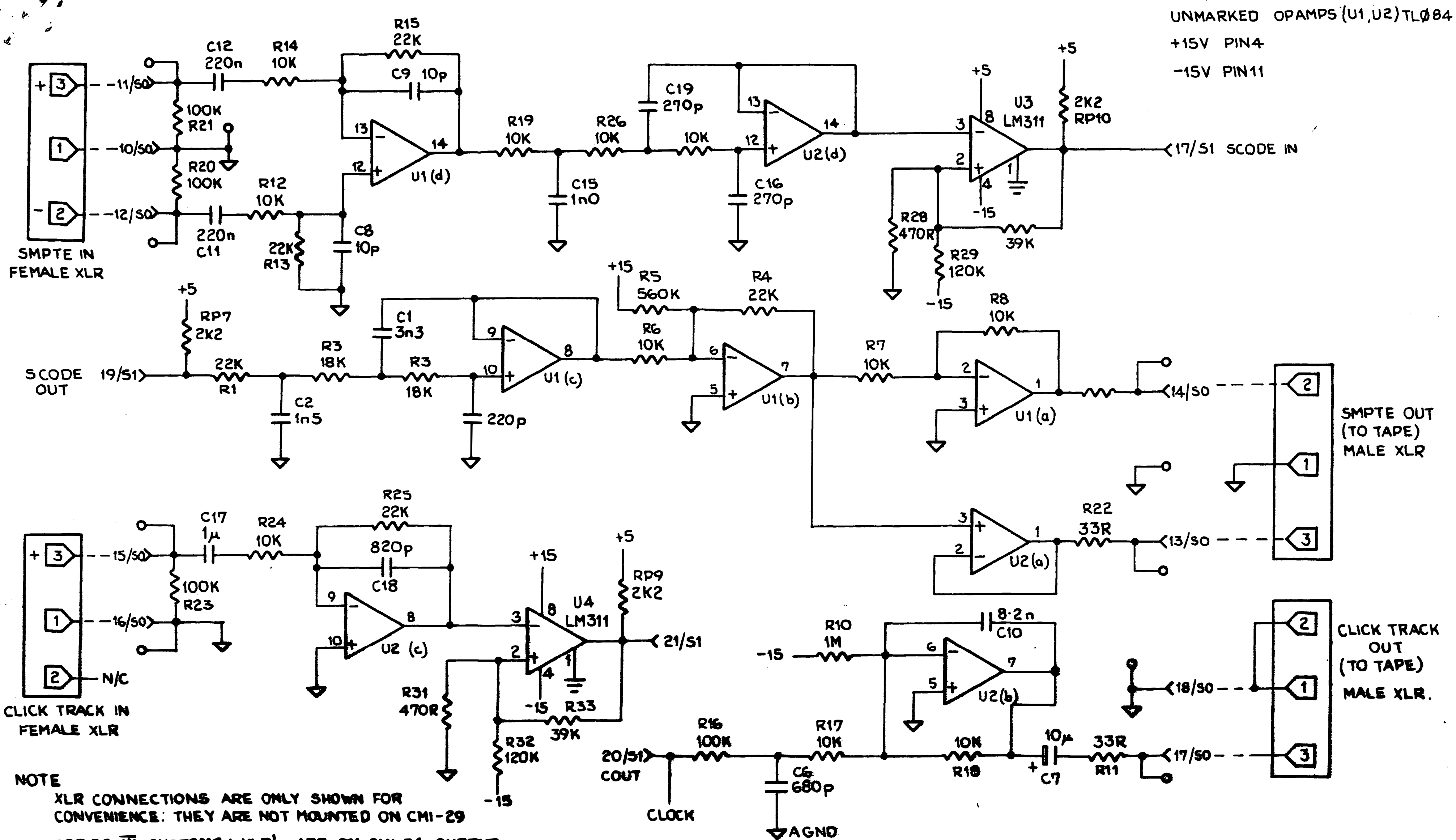
PERIPHERAL SELECT

DATE	NOV. 84
DRAWN	PSF
TRACED	
CHECKED	
REV.3	CMI - 28
	SHEET 5 of 7


ACIA1
68B50
G7
ACIA2
68B50
G7
ACIA3
68B50
G9
ACIA4
68B50
G10,11
TXD
RXD
CS2
RXCLK
TXCLK
IRQ
R/W
RSEL
E
1/S1
3/S1
5/S1
7/S1
9/S1
11/S1
13/S1
15/S1
4K7
+5
INT3
BCAS
R/W
A1
E
ACIA1
ICIA2
ACIA3
ACIA4
PIN 5 ON 6850 N/C
BD0
BD1
BD2
BD3
BD4
BD5
BD6
BD7
TIMER A
68B40 G5,6
INT2
RESET
VCC
CS1
CS0
E
R/W
RS0
RS1
RS2
VSS
TIMER A
TIMER 1A GATE
TIMER 1A O/P
TIMER 2A CLK
CLK2
A1
A2
A3
SYNCSW
A1
21/S1
CLICK /SYNC IN
B1
LS125
C1
LS86
LS244
A1
20/S1
SYNC OUT4
TIMER B
68B40 G5,6 (top)
TIMER B
TIMER 1B O/P
TIMER 2B O/P
TIMER 3B O/P
4/S1
6/S1
8/S1
SYNC OUT 1
SYNC OUT 2
SYNC OUT 3
GENERAL INTERFACE CARD
ACIA's AND TIMER
NOV. 84
PSF
REV.3
CMI-28
SHEET 6 of 7



CMI Mainframe Service Manual



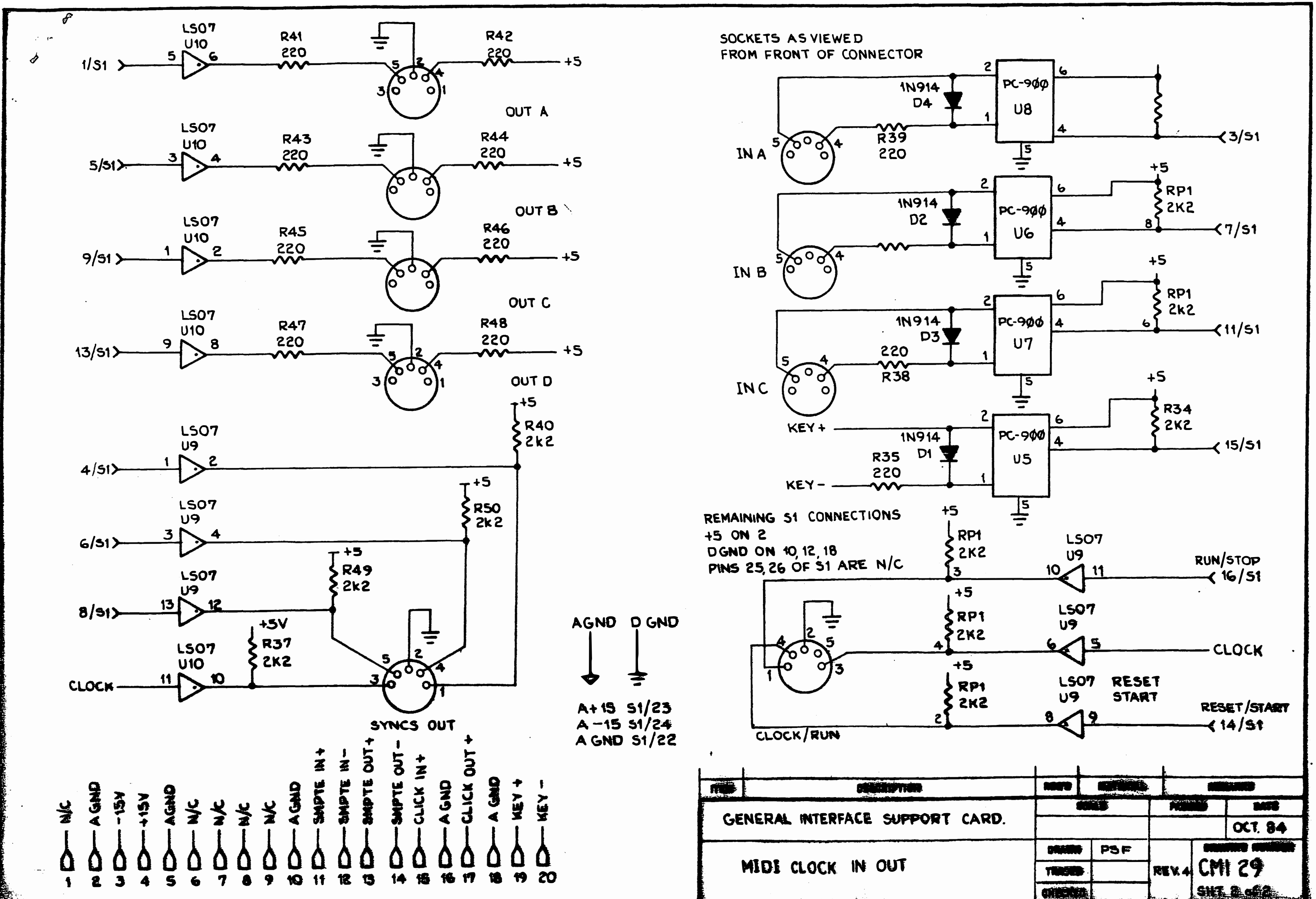
NOTE

XLR CONNECTIONS ARE ONLY SHOWN FOR CONVENIENCE: THEY ARE NOT MOUNTED ON CMI-29

SERIES III SYSTEMS: XLR's ARE ON CMI-36 OUTPUT BOARD CONNECTED TO SØ VIA RIBBON CABLE

SERIES II SYSTEMS: XLR's ARE MOUNTED IN EXTERNAL BOX CONNECTED TO SØ DIRECTLY OR TO PCB PINS INDICATED —o

ITEM	DESCRIPTION		MATERIAL	REMARKS
GENERAL INTERFACE SUPPORT CARD		SCALE		DATE NOV '84
TIME CODE / CLICK TRACK IN OUT		DRAWN P.F.		CMI 29
		TRACED		
		CHECKED	REV 4	SHEET



GENERAL INTERFACE SUPPORT CARD.

MIDI CLOCK IN OUT

OCT. 84

PSF

REV. 4 CMI 29

SHT 2 of 2