



Fermi National Accelerator Laboratory

MASTER TIMING CONTROLLER

HARDWARE DESCRIPTION

M.Fachin, C. Rotolo, C. Needles, P. Spentzouris

October 19, 1990

Table of Contents

1. INTRODUCTION	1
2. GENERAL DESCRIPTION	2
3. CLOCK GENERATION AND SYSTEM SYNCHRONIZATION.....	3
4. READ ADDRESS.....	5
5. TRIGGERS.....	6
5.1. Trigger Phase.....	6
5.2. Trigger Interface	6
5.3. Trigger WAIT	7
5.4. Trigger FIFO.....	7
6. MODES OF OPERATION.....	8
6.1. Run Mode.....	8
6.2. Test (Debugging) Mode.....	8
6.3. Calibration Mode.....	10
7. ERROR MONITORS.....	11
8. FRONT PANEL I/O AND DISPLAYS	11
8.1. Front panel LEDs	11
8.2. Coaxial connectors:.....	13
8.3. Ribbon cable connector:.....	13
9. FASTBUS INTERFACE	14
9.1. CSR0.....	14
9.2. CSR10 : READ_ADDRESS.....	15
9.3. CSR11 : PRDPATH.....	15
9.4. CSR12 : TOFFSET.....	16
9.5. CSR13 : CKPHASE	16
9.6. Fastbus Error Responses	16
10. MODULE CALIBRATION.....	16
11. MODULE INTERNAL SETTINGS.....	16
12. POWER REQUIREMENTS.....	16
13. MTC PRELIMINARY TESTING	17
14. AUTOMATED TESTS	17
14.1. Test Software.....	17
MTC features that are tested.....	18
Description of tests.....	18
APPENDIX A - Circuit Diagrams	
APPENDIX B - PAL equations used in the Fastbus Interface	
APPENDIX C - Parts List	
APPENDIX D - MTC Test Module Diagram	
APPENDIX E - Corrections to the Printed Circuit Board	

1. INTRODUCTION

The Master Timing Controller (MTC) module is part of the Silicon Strip Detector System, intended for use in experiments E771 and E789. In addition to generating the system clock coherent to the beam, the MTC provides the mechanism to maintain system synchronization, and acts as an interface between the readout system and the 1st level trigger system, generating hit addresses in response to trigger requests. The MTC provides a pipeline for triggers and is responsible for system reset (initialization). Figure 1 shows the MTC connections to the Sequencer modules and to the trigger system.

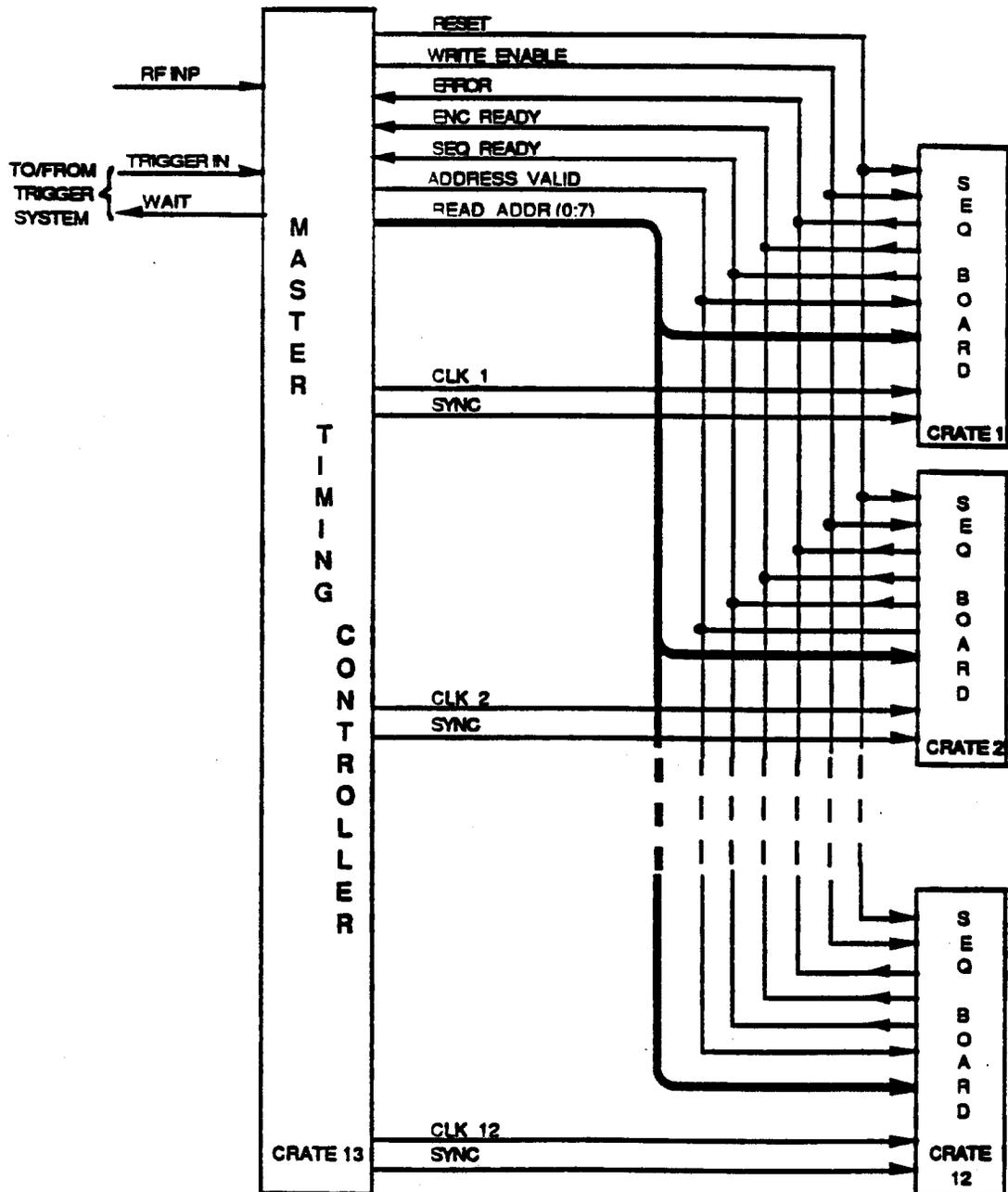


Fig. 1 - MTC connections to the Sequencers and 1st level trigger system.

2. GENERAL DESCRIPTION

The Master Timing Controller generates the system clock, controls system synchronization, and generates the hit addresses upon the receipt of Level 1 triggers from the experiments Trigger system. A block diagram of the MTC is shown in Figure (2). The MTC receives the 53 MHz Tevatron RF and establishes a near 50% duty cycle clock whose phase is adjustable relative to the incoming RF. This CLOCK along with a SYNC pulse (occurring every 256 clock cycles) are distributed to each Sequencer and eventually to all Delay/Encoders (D/Es). D/Es use this clock and sync to determine write addresses for incoming data. Being synchronized, the MTC knows the current D/E write address and generates a read or "hit" address when a trigger is received. The hit address generated is offset from the write address based upon a calibration of the Trigger decision time.

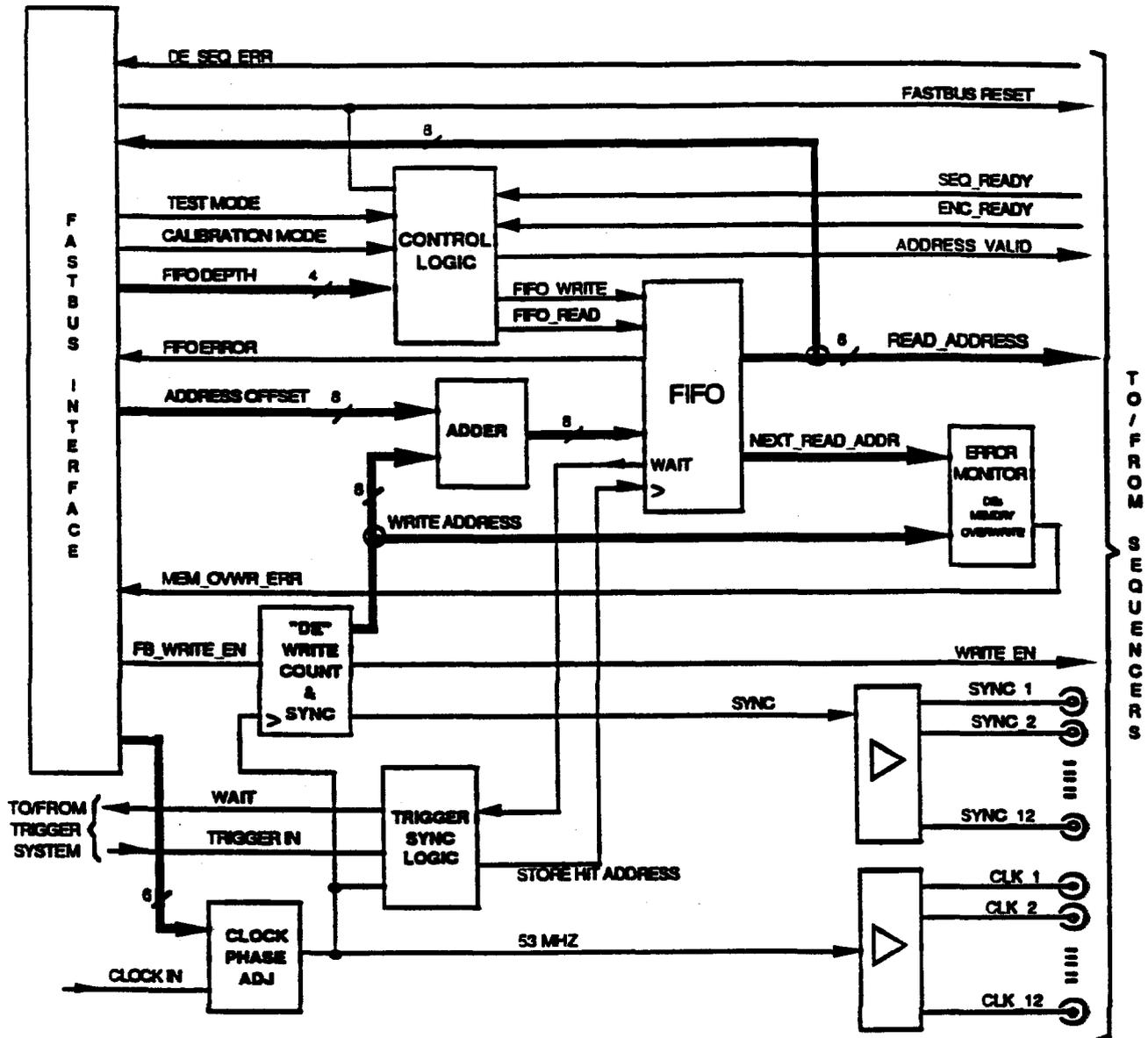


Fig. 2 - MTC internal block diagram

In the case of D/E's being busy, hit addresses are placed into a high speed FIFO queuing up to seven trigger requests which can occur as close together as successive RF buckets. The read or "hit" address output from the FIFO is broadcast to all D/E modules at a rate determined by the ENC_READY signal summed from all D/E modules.

The ability to pipeline triggers makes the system truly deadtimeless at trigger rates up to the readout bandwidth. With knowledge of the read address and the D/E's current write address, the MTC detects fatal D/E memory overwrite errors (256 clock cycles, or 4.8 μ s, is how much time it takes to overwrite the memory). However, to prevent such a condition, the system is throttled by sending a WAIT signal to the Trigger system — the MTC trigger pipeline depth can be set by Fastbus and will generate a trigger WAIT at this depth.

The MTC provides trigger synchronization monitoring and incorporates features for system calibration and debugging.

3. CLOCK GENERATION AND SYSTEM SYNCHRONIZATION

Figure 3 shows the Clock and the Sync generation block diagram.

The MTC expects a continuous NIM 53 MHz RF from the accelerator, and generates a fixed duty cycle near 50% that is fanned out individually to each crate. In order to maintain constant phase relationship to the RF, it is suggested that the sinusoidal RF signal be processed by EG&G Model 140/N Zero Crossing Discriminator, producing NIM level pulses, prior to being fed to the MTC, which in turn reshapes the 53 MHz pulses with a 8 ns delay line to achieve the desired duty cycle. To account for slow drifts of the RF with respect to the actual beam, as well as equipment changes, the MTC provides a 6-bit programmable delay line with 0.5 ns resolution for clock phase adjustment. This delay line is programmed through FASTBUS.

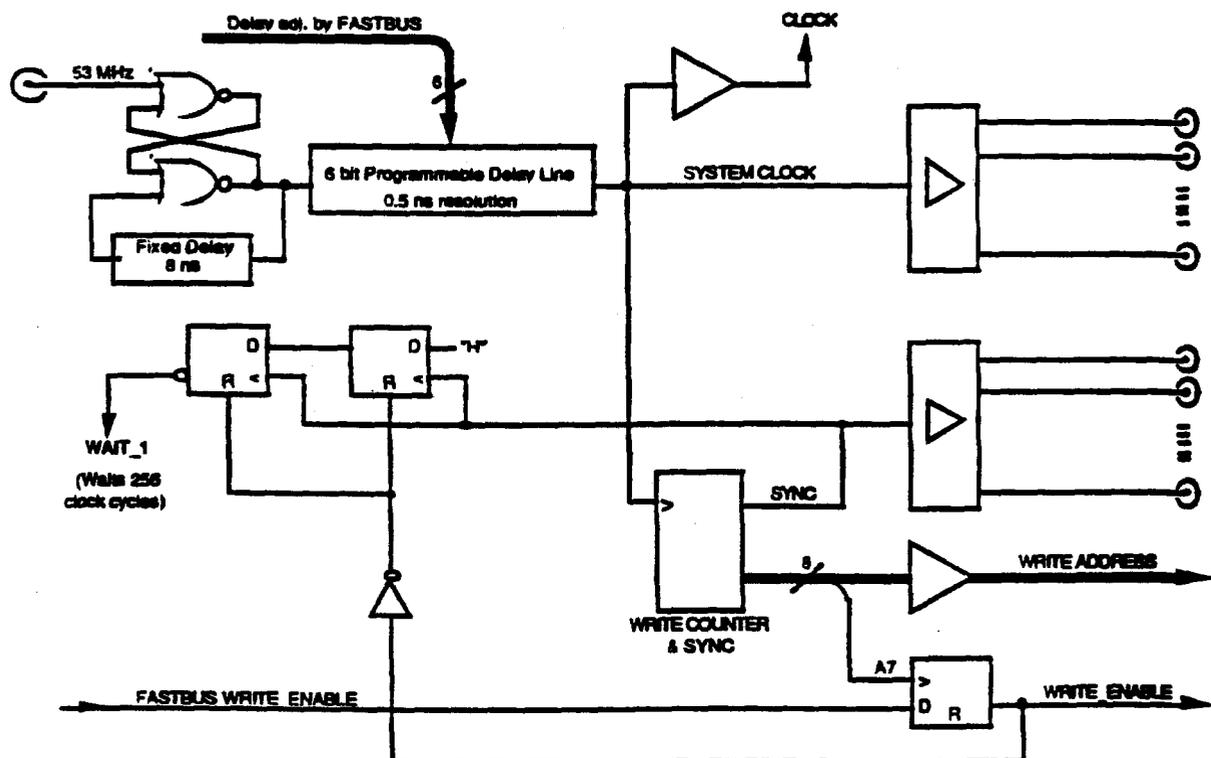
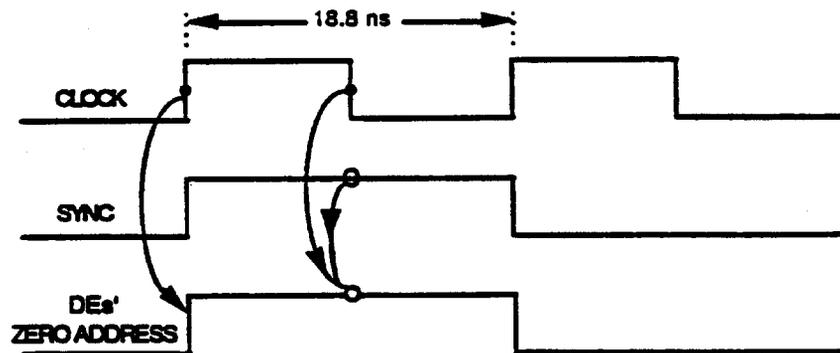


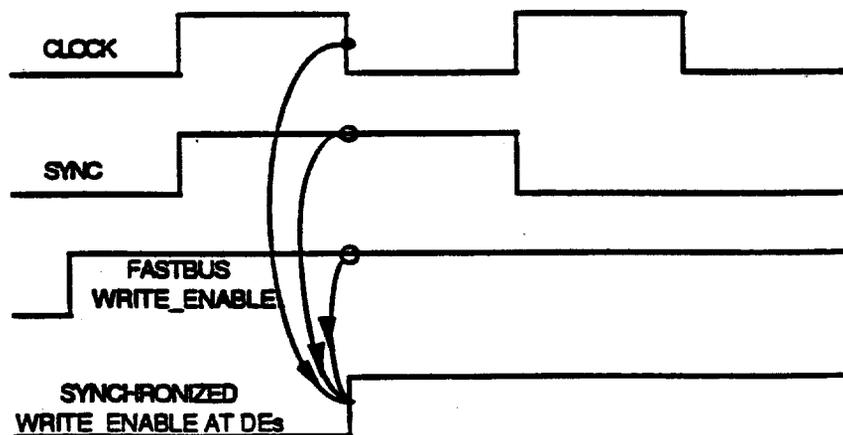
Fig. 3 - Clock Generation and System Synchronization

The MTC is thus capable of delivering a fixed phase 53 MHz clock relative to the beam. This clock is the absolute timing signal for the system and can be used as a reference throughout the system. If individual modules of the Data Acquisition System need a different phase relationship, they have to adjust the phase locally.

The MTC generates a SYNC signal with fixed phase relative to the clock on every 256 clock cycles. The MTC keeps a copy of the D/E's write counters for use as a reference to evaluate the read address (hit address). The MTC thus knows when the write counters in every D/E are expected to be at address zero. At this precise time the MTC broadcasts a 18.8 ns SYNC pulse to all Sequencers in the system. The Sequencer will then bus this signal to all the D/E's in the crate, with each D/E checking its synchronization to the SYNC pulse. If a loss of sync is detected, the faulty D/E will inform the Sequencer in the crate by asserting the SYNC_ERROR line (wired-or of all D/E's in a crate). Each faulty D/E will identify itself by latching the error and displaying it on the front panel. Although the SYNC signal has a close phase match to the system clock, it is not used as a timing signal. The SYNC signal is sampled in the D/E's on the trailing edge of the system clock, as shown in Figure 4a, and is allowed some phase skew without affecting D/E's synchronization. The Sequencers, which have control over the clock phase in each crate, have to adjust the SYNC signal phase if they change the clock phase.



(a) Synchronism checking in the D/E's



(b) WRITE_EN Synchronization

Fig. 4 - D/E and WRITE_ENABLE timing diagrams

The SYNC signal has to be calibrated internally in the MTC to phase match the clock signal. A tapped delay line is provided for this purpose, with dip switch SW3 being used to select the tap which gives the right phase. This calibration is done only once and remains fixed thereafter.

The WRITE_EN signal programmed through FASTBUS is intended to start the D/Es to acquire data, as well as allowing the MTC to accept trigger requests. The WRITE_EN is sent to the Sequencers/D/E 128 clock cycles before the SYNC pulse, so that all D/Es can see it before the next SYNC pulse.

On system start up, the WAIT signal to the trigger system is deasserted only after WRITE_EN is asserted, and 256 clock cycles after the D/Es have started taking data.

A flow chart of the system initialization is shown in Figure 5.

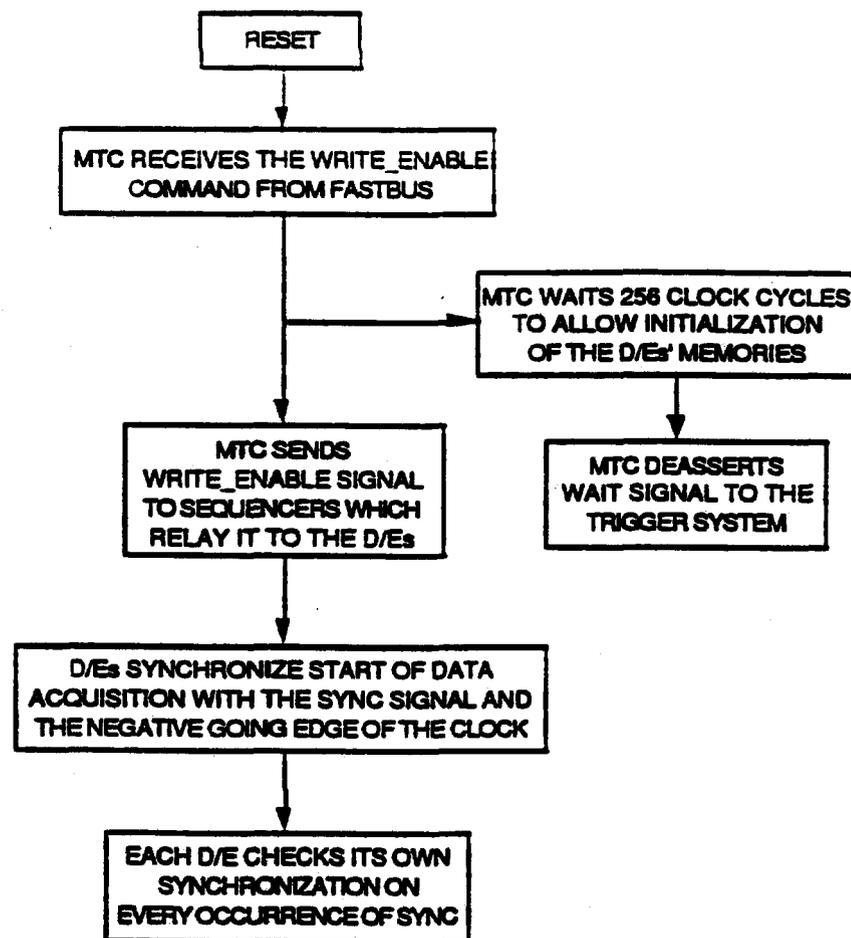


Fig. 5 - System initialization

4. READ ADDRESS

Each trigger pulse received by the MTC generates a READ_ADDRESS — address where the hit data corresponding to the trigger is stored in the D/Es memories —, obtained by subtracting an offset to the reference D/E write counter of the MTC. This offset is software programmable and its value is determined by a calibration procedure. The offset accounts for the trigger decision time, which is expected to be close to 1 μ s (for example, for a trigger decision

time of 1 μ s, the offset would be 1000 ns/18.8 ns \approx 53 clock cycles). The READ_ADDRESSES are sent asynchronously to the Sequencers (and D/Es) along with the ADDRESS_VALID signal. If the D/Es are busy, the MTC stores the trigger addresses in a FIFO for later delivery. An error is issued if the NEXT_READ_ADDR to be output from the FIFO is too close (dip switch programmable in SW2) to the current write address in the D/Es, such that a MEM_OVWR error could occur. An opened switch on SW2 means a logic one, and the quantity programmed in SW2 should be equivalent to the time (number of clock cycles) needed for the READ_ADDRESS to reach the D/Es and be processed before that memory location is overwritten with new data. The memory overwrite error monitor is inhibited when the FIFO is empty, a situation in which the hardware would flag a non-existing error.

5. TRIGGERS

Triggers can be external or internal. External triggers come from the 1st level trigger system and signal an event (external triggers are inhibited in Test mode and under error conditions in other modes). Internal triggers are software generated triggers and are intended for test purposes only. There are no hardware restrictions as to when an internal trigger is allowed. Triggers produce READ_ADDRESSES that are loaded into the trigger FIFO.

5.1. Trigger Phase

The incoming trigger pulse must hold a certain phase relationship to the system clock for the MTC to work properly. Specifically, a trigger must not arrive close to the rising edge of the clock, which would cause the trigger synchronization logic to malfunction, by not honoring the setup time of the logic. If the trigger phase is not set right, the MTC will flag an error named TRIG_PHASE_ERR, which will halt the system by discarding new trigger requests and by not sending out READ_ADDRESSES. The MTC produces the signals TRG WIN (trigger window) and TRG MON (trigger monitor) to help adjust the trigger timing. By observing these signals with a scope, one should externally delay the trigger pulse input such that the TRIGGER MONITOR signal lies inside the range presented by the TRIGGER WINDOW signal. It is recommended that the TRIGGER MONITOR signal be positioned in the center of this time window, to allow for eventual timing drifts.

An alternative way of setting the trigger timing is to monitor the TRIG_PHASE_ERR LED while making the delay adjustments. By noting the range for which no errors are flagged, one can set the trigger delay to be centered to the window.

The width of the window described above can be trimmed to allow a narrower or looser phase error check for the incoming triggers. A rotary switch (SW5) is provided for setting the window width.

In test mode, triggers are generated by software and have a random phase. The trigger phase errors are disregarded in this case, and do not halt the system.

5.2. Trigger Interface

The MTC accepts pulses from the 1st level trigger system and broadcasts the addresses of the data corresponding to those triggers (READ_ADDRESSES) to the Sequencers/D/E. The MTC is capable of pipelining trigger requests if the D/Es are busy encoding the previous trigger. The pipeline depth, programmable through FASTBUS, depends on the trigger delay, and on the expected average number of hits in the detector, which in turn causes different encoding times in the D/Es. A wise selection of the number of stages in the pipeline prevents the D/Es from wrapping around their memories and consequently overwriting data; the MTC tests for this type of error as said in the previous section. When the number of events waiting to be serviced exceeds the maximum programmed number of stages in the pipeline, the MTC sends a WAIT signal to the trigger system. The MTC, however, doesn't block out new trigger requests, even though it has sent the WAIT to the trigger system. This feature is important in the case of trigger

requests that are in the process of being delivered when the WAIT signal is asserted. The trigger request FIFO can store up to 7 trigger requests.

For test purposes, triggers can be generated by software (FB_TRIGGER) and are OR'ed to external triggers.

5.3. Trigger WAIT

The way the MTC throttles the trigger system is by sending a WAIT signal. This signal is asserted under four conditions: whenever the system is in stand-by, is running in Test mode, the FIFO depth exceeds the programmed FIFO depth, or the Sequencers are not ready (the Sequencers' FIFOs used to store the encoded data from the D/Es become more than half full with encoded data from the D/Es).

5.4. Trigger FIFO

When the D/Es are busy encoding the previous trigger, the MTC stores the READ_ADDRESSES corresponding to the incoming triggers in a FIFO. Commercial FIFOs available today have limitations in speed and functionality. To meet the requirements of the SSD trigger pipelining, a discrete ECL FIFO is built in the MTC, with the following characteristics:

- 53 MHz input frequency
- depth (number of stages) programmable through FASTBUS
- generation of a WAIT signal if the current number of stages in the FIFO is greater than the FASTBUS programmed depth
- generation of status such as *empty, full, and error*
- generation of the NEXT_READ_ADDRESS

The number of stages in the FIFO was limited to 7. Figure 6 shows the FIFO connections to other MTC blocks.

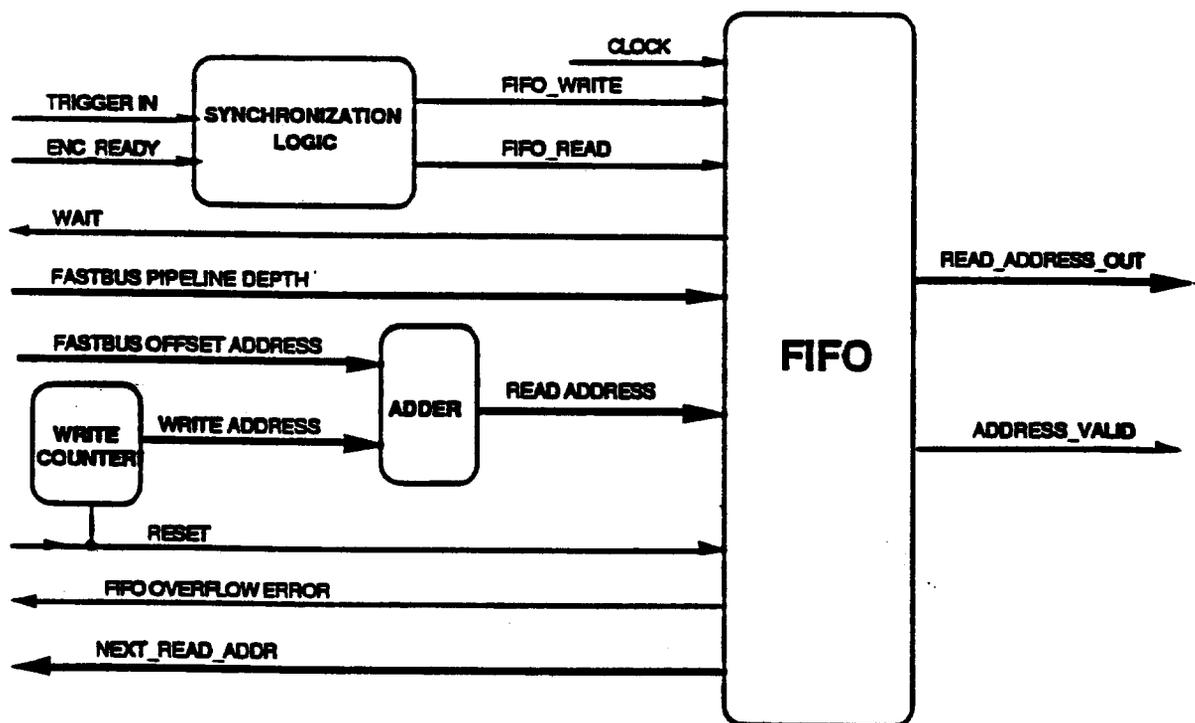


Fig. 6 - FIFO signals

The pipelining functional characteristics are:

- *fifo empty*: do nothing; wait for trigger.
- *fifo not empty*: if the D/Es are READY, retrieve a read address stored in the pipe; point to the next read address in the pipe.
- *fifo depth > programmed number of stages*: send WAIT to the trigger system. This is a very important feature that makes the FIFO more flexible, allowing the trigger pipelining to be tuned to the detector hit rate.
- *fifo depth > 7*: error.

A more detailed block diagram for the FIFO is depicted in Figure 7.

The FIFO is implemented as 8 registers, 2 pointers (write and read pointers) and a status generation logic.

The write pointer is always pointing to the next free location in the FIFO. Upon receiving a trigger pulse, the FIFO writes the hit address to the current free location and increments the pointer to the next one. The write operation is synchronized internally to the falling edge of the clock; the READ_ADDRESSES are generated after the rising edge of the clock.

The read pointer is always pointing to the register where the data in the pipe is to be read. The contents of this register is available and is called NEXT_READ_ADDR, used in the MEM_OVWR error monitor. In response to a FIFO_READ, this data is latched and becomes the READ_ADDRESS, which is sent out to the Sequencers/D/Es. The read pointer is then incremented, pointing to the NEXT_READ_ADDR. In order to allow minimum time for the error and status circuits, the FIFO_READ is synchronized internally to the rising edge of the clock. This provides one half clock cycle for the error checking circuit, since the writes occur on the falling edge of the clock. The difference between the write pointer and the read pointer is the FIFO depth.

6. MODES OF OPERATION

The MTC has 3 modes of operation: Run (Acquisition) mode, Test mode, and Calibration mode.

6.1. Run Mode

The Run mode is the normal mode for data acquisition. The MTC basically broadcasts READ_ADDRESSES in response to trigger requests, and is capable of pipelining triggers (read addresses) when the system is busy. The MTC checks for system integrity, halting the system on the occurrence of an error. A FB_WRITE_EN causes the MTC to enter Run mode.

6.2. Test (Debugging) Mode

The Test mode feature serves many purposes, the most important one being the system test. In TEST_MODE, the D/Es acquire data in the usual fashion, the data now being a known pattern generated by the Post-Amp/Comparator board. The TEST_MODE capability allows one to read the contents of the D/Es' memories and compare them to the known pattern being written into the memories.

In this mode, the write counter in the MTC is shut down (SYNC pulses are generated by an alternate counter used solely for this purpose), and the read address is controlled by the trigger offset (TOFFSET) setting alone. The actual read address that is broadcast is achieved by subtracting an offset from the current write counter contents (which is zero in this case).

$$\text{"Test" READ_ADDRESS} = 256 - \text{TOFFSET}$$

In this way, one has control over which addresses to read from the D/Es. A software program can scan over the entire D/E memory and check for correct data.

External trigger requests are disregarded in this mode, and triggers are generated by software (FB_TRIGGER). The software generated triggers are not phased to the clock, and their role is to load test READ_ADDRESS into the trigger FIFO. A WAIT condition can be read back through Fastbus, caused by either a FIFO_WAIT or by the Sequencers not being ready, and has to be observed by the software routine that generates triggers.

The procedures to execute the test are:

- . Assert the TEST_MODE signal.
- . Assert FB_WRITE_EN. This will initiate the D/Es accepting hit data.
- . Load an offset that will give the desired READ_ADDRESS:

$$\text{READ_ADDRESS} = 256 - \text{TOFFSET}$$

- . Generate the FB_TRIGGER pulse by software.
- . Change offset and wait for ENC_READY to issue another FB_TRIGGER

The Test mode can be also used to test the MTC alone. A great amount of the MTC's hardware can be checked under Test mode. The generation of a FB_ENC_READY closes the loop on the board, by allowing triggers stored in the FIFO to be read back through Fastbus. This checks the circuitry that evaluates the read address, the FIFO itself, and the Fastbus interface. The routine can be extended to further check the FIFO depth (generates Fastbus readable WAIT signals if the number of stages in the FIFO exceeds the programmed depth), the memory overwrite error monitor, and the hardware used in the calibration feature of the MTC.

6.3. Calibration Mode

This mode is used for evaluating the correct TOFFSET (number of clock cycles) necessary to accomplish for trigger decision time and other intrinsic delays. When set to Calibration mode, the MTC waits for the first external trigger pulse to arrive and generates N (switch setable) consecutive read addresses, using an estimated offset. The number of triggers produced is equal do the switch setting plus 1.

Fig. 8 shows a diagram of this switch, which is set to the number 4 in the figure (5 trigger pulses produced)

These READ_ADDRESSES are stored in the MTC's trigger FIFO, and are delivered on demand to the D/Es. Additional trigger pulses cause the MTC to generate a new burst of N trigger addresses synchronized to the arrival of the trigger pulse input. In other words, each trigger pulse will generate one burst of N consecutive trigger addresses. However, if a new trigger arrives before the N calibration triggers corresponding to the previous trigger have been stored in FIFO, the FIFO_OVFL_ERR is generated.

The procedure to perform the calibration cycle is:

- . RESET the module to clear the error latches and reset the FIFO.
- . Set trigger pipeline depth (FIFO_DEPTH) to 1, so that the FIFO_WAIT will inhibit external triggers if the first burst of N read addresses is not serviced yet.
- . Assert CALIB_MODE



Fig. 8 - Diagram for the number of calibration triggers switch

. Set **WRITE_EN**

If any error occurs, external triggers are blocked out and **ADDRESS_VALIDs** are not produced.

Performing this calibration procedure with different clock phases (adjusted in the Sequencers) and different offsets, the system is able to determine the correct offset by looking into the data collected in the D/Es. This offset then becomes a constant to the system and will be used by the MTC to determine the **READ_ADDRESSES** that are broadcasted to all Sequencers/D/Es in the system.

The Calibration mode internal hardware of the MTC can be checked by operating the MTC with both the Calibration mode and Test mode asserted. This allows one to have control of the **READ_ADDRESS** to be stored in the trigger FIFO (see Test mode) when a calibration cycle is initiated by a software generated trigger. The expected contents of the FIFO are **N** (switch setable) consecutive numbers, the first one being equal to the loaded **TOFFSET** subtracted from 256.

7. ERROR MONITORS

The MTC monitors errors in the system. Any error is fatal, and the mechanism by which the MTC informs the control system of an error in the DAS or itself is to halt the system, i.e., external triggers are disregarded and **ADDRESS_VALIDs** are not sent out, except when the MTC is in Test mode. In addition, the MTC provides a NIM output indicating the presence of an error, and front panel LEDs for visualization of the errors. All errors are latched, and can be cleared by software or by pressing the front panel switch **CLEAR ERRORS** (provided the error conditions has been corrected).

The errors can be read through **FASTBUS** and include:

DE_SEQ_ERROR : means that a D/E module has lost synchronization or that a Sequencer FIFO has overflowed. To identify the exact cause of the error, one needs to read the Sequencers' error registers.

MEM_OVWR_ERROR : indicates that a trigger address cannot be broadcast due to data being overwritten in the D/E memories. The MTC compares the D/E' write address against the **NEXT_READ_ADDR**; if they are too close to one another, as programmed in dip switch **SW2** (memory overwrite margin), the **MEM_OVWR_ERR** is flagged.

FIFO_OVFL_ERR : indicates that the trigger pipelining FIFO in the MTC has overflowed. This error may occur if the trigger system disregards the **WAIT** signal the MTC sends to it informing the pipe is almost full. Normally, the **WAIT** signal is sent out before the FIFO becomes full, whenever the programmed depth is exceeded.

TRIG_PHASE_ERR : signals that the trigger input from the first level trigger system has drifted by an amount that causes it to fall outside a pre-established time window.

CLOCK_MISSING : detects that at least one clock cycle was not received from the accelerator.

ERROR_SUM : it's the OR logic of all errors.

8. FRONT PANEL I/O AND DISPLAYS

Figure 8 shows a section of the MTC's front panel.

8.1. Front panel LEDs

MOD SEL : it is a yellow LED that flashes when module address is selected by Fastbus.

FIFO DEPTH : Is composed of 8 green LEDs numbered 0-7. They monitor the difference between the read and write pointers of the FIFO. If at 0, the read and write pointers are

pointing to the same register in the FIFO, which means the FIFO is *empty*. The maximum depth is 7, and an additional write to the FIFO will cause it to overflow, flagging an error.

- Errors -

MEM OVWR : indicates a MEM_OVWR_ERR in the D/Es.

FIFO OVF : indicates that the FIFO has overflowed.

TRG PHZ : monitors the external trigger phase to be within a pre-established time window.

DE/SEQ : signals that at least one D/Es is out of sync or that some Sequencer FIFO has overflowed.

CLK MIS : indicates the 53 MHz system clock is missing or was not present for at least one clock cycle.

- Status -

WRITE EN : status LED indicating the MTC is ready to accept triggers. The WRITE_EN is required also when the MTC is being operated in Test or Calibration modes.

CALIB : informs the MTC is in Calibration mode..

TEST : Test mode indication.

TRG IN : alongside TRG IN connector, shows when the MTC is receiving external triggers.

TRG WAIT : alongside the WAIT output, shows WAIT signals being sent to the trigger system. A persistent WAIT condition will leave the LED constantly lighted.

- At the front panel bottom -

See Fig 10.

+5 : monitors the 5 volts power in the MTC board.

-5.2 : monitors the -5.2 volts power for the ECL logic.

-2 : monitors the -2 volts power supply for the ECL terminations.

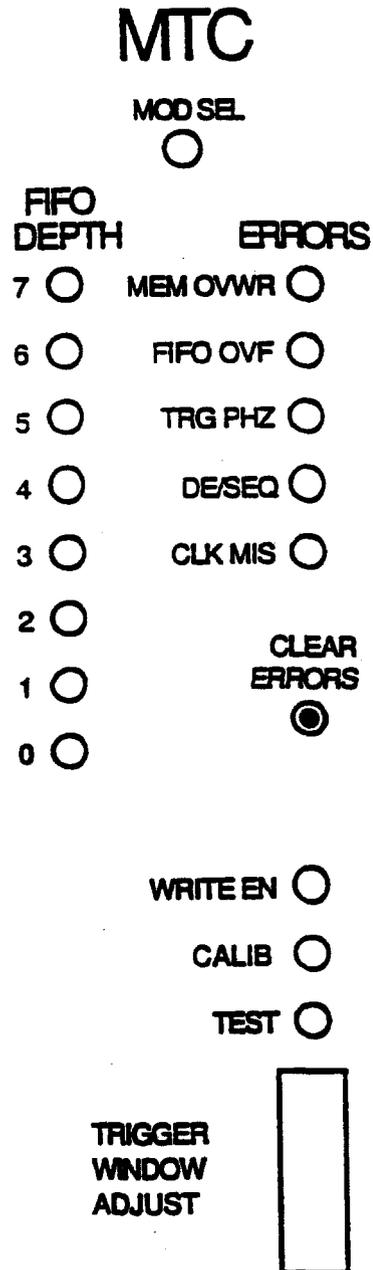


Fig 8 - MTC's front panel partial view

8.2. Coaxial connectors:

Fig 9 depicts the MTC's front panel coaxial connectors.

CLOCK IN : a NIM 53 MHz clock input which is derived from the accelerator RF via the CATV system. The only concern here is to have a signal that has a fixed phase relationship to the RF. This signal may experience slow timing drifts over long periods of time, which can be compensated by reprogramming (Fast-bus operation) the delay line internal to the MTC.

CLOCK OUT : a NIM 50% duty cycle clock output reference that has a constant phase to the beam. This signal is fanned out individually to each of the 12 crates. The 13th output is for monitoring.

SYNC OUT : NIM output pulse, synchronous to **CLOCK**, to test write counters sync at each zero count. This signal is delivered individually to each Sequencer. The 13th output is for monitoring.

ERROR : NIM output intended for immediate signalization of an error condition.

TRG IN : NIM trigger pulse input from the 1st level trigger system.

TRG MON : output used in conjunction with the **TRG WIN** signal to adjust the phase of the trigger signal. **TRG WIN** : presents the time window for phasing the trigger.

TRG WAIT : NIM output signal informing the 1st level trigger that the system is busy and cannot accept new trigger requests

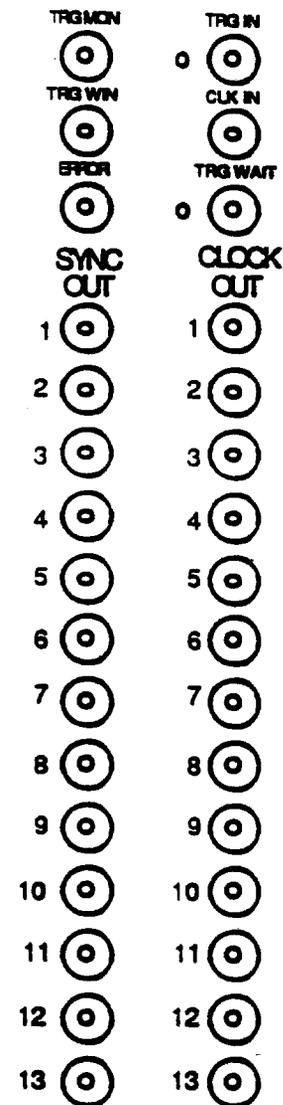


Fig 9 - Front panel Coaxial Connectors

8.3. Ribbon cable connector:

All signal in the ribbon cable connector are bussed to all crates. It is shown in Fig. 10.

RESET : initiated from Fastbus, it accomplishes system resetting and initialization. It is a differential ECL output signal, asynchronous to **CLOCK**.

WRITE_EN : output signal, differential ECL, asynchronous to **CLOCK**, for enabling the D/E modules to start data acquisition, i. e., accept hit data and increment write counters.

ENC_READY : single-ended wire OR'd ECL input signal from the Sequencers informing the D/Es status. This signal is used in determining when to deliver another

READ_ADDRESS to the Sequencers/D/Es. It is the OR logic of the encoders not being ready.

SEQ_READY : input from the Sequencers. single-ended wire OR'd ECL, it signals that the Sequencers have room in its FIFOs for encoded events. If the Sequencers are not READY, the MTC immediately sends a WAIT to the trigger system. SEQ_READY performs the OR logic of the Sequencers not being ready.

DE_SEQ_ERR : single-ended wire OR'd ECL input from the Sequencers informing that the event FIFOs in the Sequencers have been overfilled by the encoders or that a D/E has lost synchronization.

READ_ADDRESS : 8 bit differential ECL output, asynchronous to CLOCK.

ADDRESS_VALID : it is a 80 ns wide differential ECL output pulse, asynchronous to CLOCK, occurring 80 ns after the READ_ADDRESS is asserted. It signals the Sequencers that a new READ_ADDRESS is available.

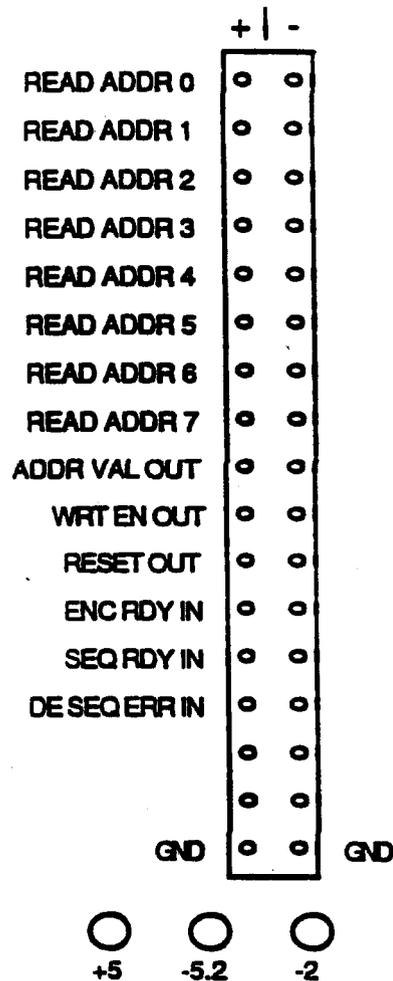


Fig 10 - Ribbon Cable connector

9. FASTBUS INTERFACE

The MTC communicates to the external world through FASTBUS. Functions executed through FASTBUS include system resetting, system calibration, system debugging, clock phase adjustment, pipeline depth setting, read address offset programming and error reporting.

The MTC is a slave device that responds to geographical address in the CSR space. The module ID is 01A2. The signals and addresses are listed below.

9.1. CSR0

CSR0 is used to control the MTC. Its signals are listed below:

bit	write	read
CSR0(00)	Set Error Flag	Error Flag
CSR0(16)	Clear Error Flag	ID
CSR0(02)	WRITE_EN Set	WRITE_EN
CSR0(18)	WRITE_EN Clear	ID
CSR0(06)	Calibration Mode set	Calibration Mode status
CSR0(07)	START_CALIB	ENC_READY

CSR0(22)	Calibration mode clear	ID
CSR0(08)	Test Mode set	Test Mode status
CSR0(09)	not used	CLOCK_MISSING
CSR0(24)	Test Mode clear	ID
CSR0(10)	not used	DE_SEQ_ERR
CSR0(11)	not used	MEM_OVWR_ERR
CSR0(12)	not used	FIFO_OVFL_ERR
CSR0(13)	not used	TRIG_PHASE_ERR
CSR0(30)	Reset	ID
CSR0(16:31)		MODULE ID (01A2)

RESET is not latched internally, and the RESET pulse is the result of writing a one to bit 30 of CSR0. A hardware reset is performed on system power up, and a software reset is issued in system initialization or after a fatal error had halted the system. RESET causes the MTC to go to the following state:

- . WRITE_EN is reset
- . Calibration mode is reset.
- . Test mode is reset.
- . All errors are cleared.
- . TOFFSET, CKPHASE, and PRDPATH (programmable FIFO depth) are left unchanged.

The FB_WRITE_EN command signal to start data acquisition is set by CSR0(02) and reset by CSR0(18).

The Calibration mode is set by CSR0(06) and reset by CSR0(22).

The Test mode is set by CSR0(08) and reset by CSR0(24).

9.2. CSR10 : READ_ADDRESS

CSR10 is used in Test mode to read back the READ_ADDRESSES and to generate FB_TRIGGERS and FB_ENC_READYs.

bit	write	read
CSR10(0:7)	don't care	FIFO_RA(0:7)
CSR10(08)	FB_TRIGGER	not used
CSR10(09)	FB_ENC_READY	not used

FB_TRIGGERS and FB_ENC_READYs are not latched internally. Pulses on these lines are produced by writing a one to bit 8 and 9 of CSR10, respectively.

9.3. CSR11 : PRDPATH

CSR11 programs the trigger pipeline depth (PRDPATH). Legitimate values are 1 to 7. CSR11 is also used to monitor status used in Test mode.

bit	write	read
CSR11(0:2)	PRDPATH(0:2)	same
CSR11(4)	don't care	TRIG_WAIT
CSR11(5)	don't care	FIFO_NOT_EMPTY

The status of WAIT output on the front panel can be read through CSR11(4) which reads 1 whenever WAIT is asserted. This is intended for module diagnostic purposes.

The NOT EMPTY/EMPTY status of the FIFO can be read in CSR11(5) which reads 1 when the FIFO is NOT empty (the FIFO has READ_ADDRESSES stored in it) and 0 when the FIFO is EMPTY. This bit is intended for module diagnostic purposes.

9.4. CSR12 : TOFFSET

CSR12 holds the 8-bit trigger address offset (TOFFSET).

bit	write	read
CSR12	TOFFSET(0:7)	same

9.5. CSR13 : CKPHASE

CSR13 is used to adjust the the internal clock delay. Six bits are used to program the ELMEC PDH 6500 delay line with a .5 ns resolution.

bit	write	read
CSR13	CKPHASE(0:5)	same

9.6. Fastbus Error Responses

SS=7 Bad NTA R/W

SS=6 R/W to invalid address

SS=2 End of Block (Although not normally used, the MTC is capable of Block Transfers)

10. MODULE CALIBRATION

As said in section 3, the MTC requires the phase of the SYNC signal to phase match the clock phase. This is accomplished in SW3 (only one switch should be closed at a time) and needs to be performed only once.

11. MODULE INTERNAL SETTINGS

In addition to SW3 (SYNC phase) adjustment which is fixed for a particular module, two other dip switch adjustments are required. They are dependant on the experiment, which means that they have to be tuned to a particular system. These dip switches are SW6 for adjusting the number of triggers for Calibration mode, and SW2, used for the memory over-write margin adjustment.

SW6 is shown in Fig. 8 and explained in section 6.3 above.

SW2 setting depends on the number of clock cycles required for a READ_ADDRESS to be recognized by the D/Es, after a ADDRESS_VALID signal was issued. This number compensates for all delays due to cables, Sequencer processing and D/E acknowledging. In other words, this number should be set to some safe number (margin) that would guarantee that the D/Es' memories will retrieve the stored data before being overwritten by new data. The MTC monitors this situation by knowing the current D/Es' write address and the NEXT_READ_ADDRESS that is to be sent out.

Another switch, SW5, is provided for adjusting a time window for incoming triggers. SW5 is a rotary switch and is located in the front panel. Positions 1 to 5 are used. SW5 setting depends on how tight the external trigger phase is to be monitored.

12. POWER REQUIREMENTS

5 Volts @ 2 A

-5.2 Volts @ 8 A

-2 Volts @ 3.5 A

13. MTC PRELIMINARY TESTING

After assembling the MTC module, there are several tests that are required before automated software tests can be performed.

A Test Module (Appendix D) was built to provide the clock and trigger inputs to the MTC, so that the board can be tested. By providing a 53 MHz clock input to the MTC, the signals CLOCK OUT, SYNC, and TRG WIN should be present at the front panel coaxial connectors. It is important to check that all the 13 CLOCK and SYNC outputs deliver a nice NIM output. The CLOCK outputs should have approximately 50% duty cycle; the TRG_WIN width can be adjusted by a front panel rotary switch. This is the right moment to calibrate the SYNC phase delay, as explained in section 10.

14. AUTOMATED TESTS

The hardware requirements for testing the MTC are a standard Fastbus crate, with a Fastbus Smart Crate Controller, and the MTC Test Module, which provides the 53 MHz clock and the external trigger input to the MTC. Fig. 11 shows the connections between the MTC and the Test Module. The circuit diagram of the MTC Test Module is found in Appendix E.

The trigger phase, as observed in the trigger monitor output, has to be adjusted to reside within the time window presented by the TRIG_WINDOW signal (the leading edge of the window is dependent on the leading edge of the clock, and the width is set by the SW5 front panel rotary switch). See section 5.1 for a more detailed description. If the trigger phase is not correct, an error is produced and some tests will not run, since the MTC was designed to stop at an error condition.

The normal situation is when the external trigger produced by the 1st level trigger system is delayed externally in order to have the right phase. For the tests, however, the absolute phase is not important and the phase adjustment can be done by adjusting the clock phase, which changes the clock delay and, consequently, the window delay with respect to the trigger input. The trigger could also be delayed externally, using a delay line module or cable.

The front panel TRIG PHZ LED goes on if the trigger phase is not correct. By providing the MTC with an external clock and an appropriate trigger input, the front panel error LEDs should go off by pressing the CLEAR ERRORS front panel push-button switch.

A program, called MTC Test Software, was developed to test the MTC. This program tests all MTC features, being able to perform system tests as well.

14.1. Test Software

The MTC Test Software is resident in the FSCC (Fastbus Smart Crate Controller), burned in EPROM. Below is presented a brief summary of what is expected from the hardware and software to accomplish the tests on the MTC.

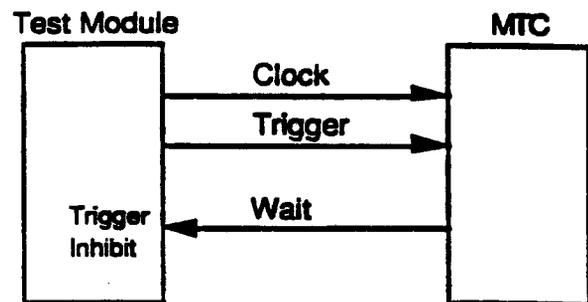


Fig. 11 - The MTC Test Module

Test Mode: External trigger requests are disregarded, triggers are generated by software (FB triggers).

Write counter is set to zero (256) and the READ_ADDRESS is controlled by the trigger offset ($READ_ADDRESS = 256 - TOFFSET$)

WAIT is always set in this mode.

FB_ENC_READY reads back the READ_ADDRESSES.

ADDRESS_VALID continues to be generated in the presence of errors.

Calibration Mode: Each external trigger pulse generates N (switch setable) consecutive READ_ADDRESSES. They are stored in the Trigger FIFO and send to the Sequencers/D/Es on demand (upon receiving of a ENC_READY signal).

ADDRESS_VALID will not be generated in the presence of an error.

Calibration Mode may also be used in combination with Test Mode. This is useful primarily in checking if the MTC is generating the correct READ_ADDRESSES. The first READ_ADDRESS, in this way, can be determine by software.

Run Mode: Each external trigger generates a READ_ADDRESS; if the D/Es are not ready, the READ_ADDRESS is stored in the trigger FIFO.

The FIFO depth (number of stages in the FIFO) is programmable and WAIT is asserted if the number of FIFO stages is greater than the programmed depth. If the number of triggers exceed the maximum FIFO capacity (equal to 7), then the FIFO OVFL error will be generated.

If the read address (obtained by subtracting the offset — determined by the calibration procedure — to the reference D/E write counter of the MTC) is close by a switch setable amount to the current write address in the D/Es, a MEM_OVWR error will be issued.

Any of the above errors, or TRIGGER PHASE and D/E errors, will stop external triggers.

ADDRESS_VALID is inhibited until all errors are cleared.

MTC features that are tested

- 1) Fastbus interface
- 2) Fifo read address
- 3) Fifo overflow
- 4) Clock phase adjustment
- 5) Memory overwrite
- 6) Wait
- 7) Calibration

Description of tests

Pseudo code for each of the tests are:

Fastbus interface : Write and read to (from) CSR10 - CSR13;
 set and reset flags (bits) and read status of CSRO.

Fifo read address : Reset
 Error flag reset
 Test mode
 Write enable
 Trigger address offset to 256 - N
 FB trigger
 FB encoder ready
 Read CSR10
 Compare
 Reset

The above test checks the arithmetic unit that evaluates the READ_ADDRESS, the trigger FIFO where this address is stored, and the control pulses FB_TRIGGER and FB_ENC_READY.

Fifo overflow : Reset
 Error flag reset
 Test mode
 Write enable
 Set trigger pipeline depth to 7
 While not overflow
 Set trigger address offset
 FB trigger
 End While
 Check number of triggers generated
 Loop on trigger number
 Read back with Encoder Ready and
 check address
 End Loop
 Reset

The current FIFO depth can be found at any time by writing to the programmable depth register and observing the WAIT signal. The WAIT signal is asserted whenever the FIFO depth exceeds the programmed depth.

Clock phase adjustment : Change the clock phase over a range of values, checking for trigger phase error. The change in the clock phase causes the trigger window to move, throwing the trigger input outside the window, producing the error. For automatic tests make sure that trigger remains in phase (inside window).

Memory Overwrite : Reset
 Error flag Reset
 Test mode
 Write Enable
 Loop on the 255 possible read addresses
 Error flag Reset
 Set trigger address offset
 Assert FB trigger
 Check Memory Overflow
 If not when address agrees with the Dip Switch, report it
 Assert FB trigger
 Read and compare address

In the Memory Overwrite test, the memory overwrite margin switch SW2 has to be hardcoded to the value 8, or one can change the default during the initialization time to agree with the switch setting.

```

Wait : Reset
      Error flag reset
      Loop over fifo depth
        Set fifo depth
        Set write enable
      Loop until wait or error or timeout
      When wait or error, external triggers are inhibited
      Set test mode
      Assert Fb encoder ready until fifo is empty
      Compare with depth or if error report
End Loop
Reset

```

```

Calibrate : Reset
           Error flag reset
           Set trigger pipeline to 1 so wait will inhibit all but the first external trigger
           Set calibration
           Set Write enable
           Loop until Wait
             Set Test mode
             Loop
               Assert Fb encoder ready
               Fill array
               Break if fifo empty
           End Loop
           Compare and check array, report errors           ; the FIFO contents should
                                                         ; read consecutive numbers
                                                         ; representing the read addresses

Reset

```

Notes:

Test mode is required for reading back READ_ADDRESSES in order to prevent the blocking of ADDRESS_VALIDs due to MEM_OVWR error. For every FB write to a register, the corresponding read is performed in order to check the FB interface.

During the initialization, the default values that correspond to dip switches can be changed (main menu), or the hardcoded values can be tested to set the defaults.

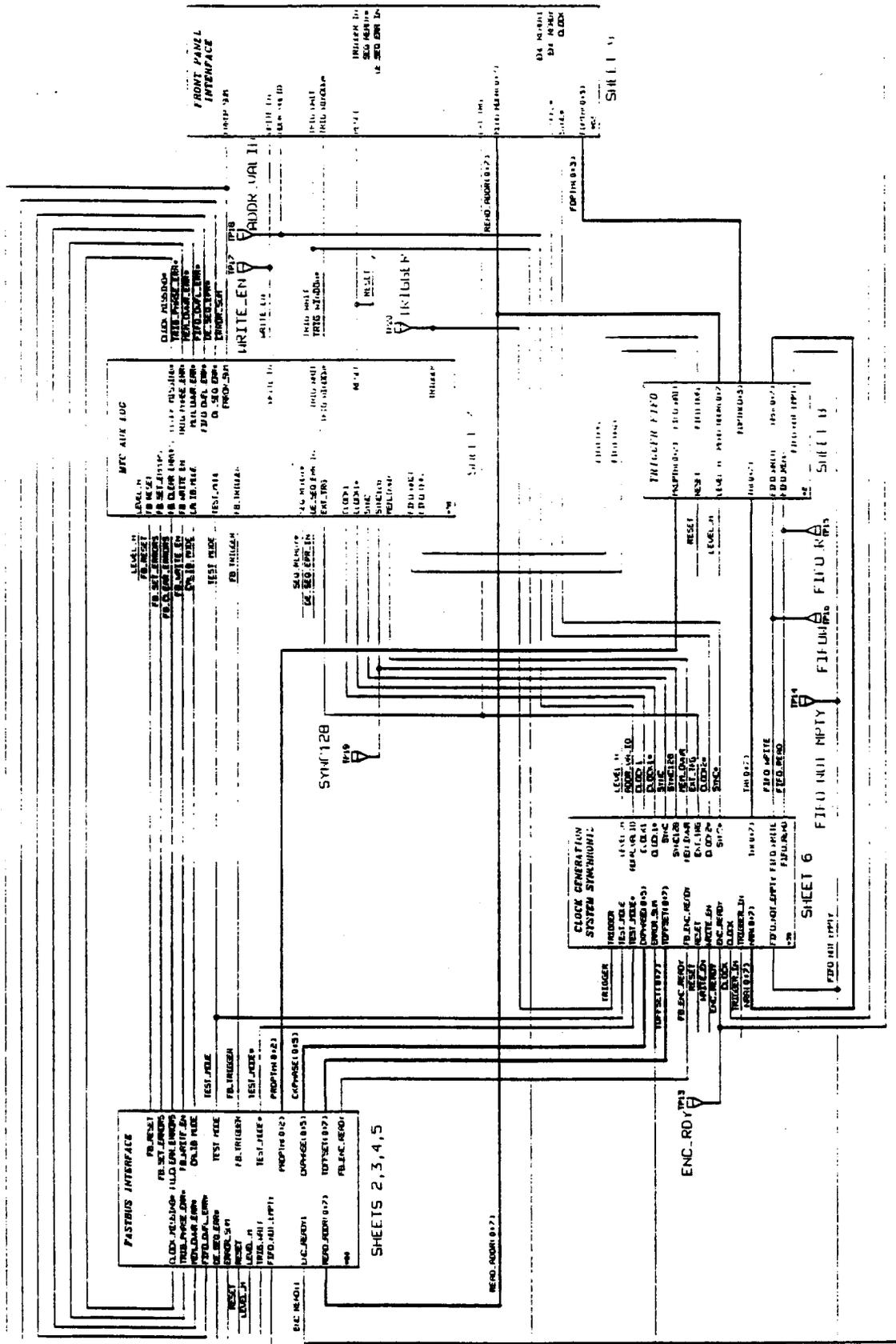
The default values (i.e trigger offset, pipeline depth,...) can be modified for each test if the program is being runned in the interactive mode.

Refer to document _____ for a complete description of the MTC Test Software.

APPENDIX A

CIRCUIT DIAGRAMS

B 7 6 5 4 3 2 1



FRONT PANEL INTERFACE

MTC AIX LOG

LEVEL IN

TEST MODE

TEST FILE

PROPTIM 0123

ENC_RDY

SYM128

CLOCK GENERATION SYSTEM SYNCHRONIZ.

TRIGGER FIFO

FIFO NPT

WRITE

READ

TEST

PROPTIM 0123

ENC_RDY

TEST MODE

TEST FILE

PROPTIM 0123

ENC_RDY

TEST MODE

TEST FILE

PROPTIM 0123

ENC_RDY

GENERAL WARNING: MILLER ELECTRONICS CORPORATION
 DETROIT ELECTRONICS DIVISION GROUP

MTC BLOCK 01100001

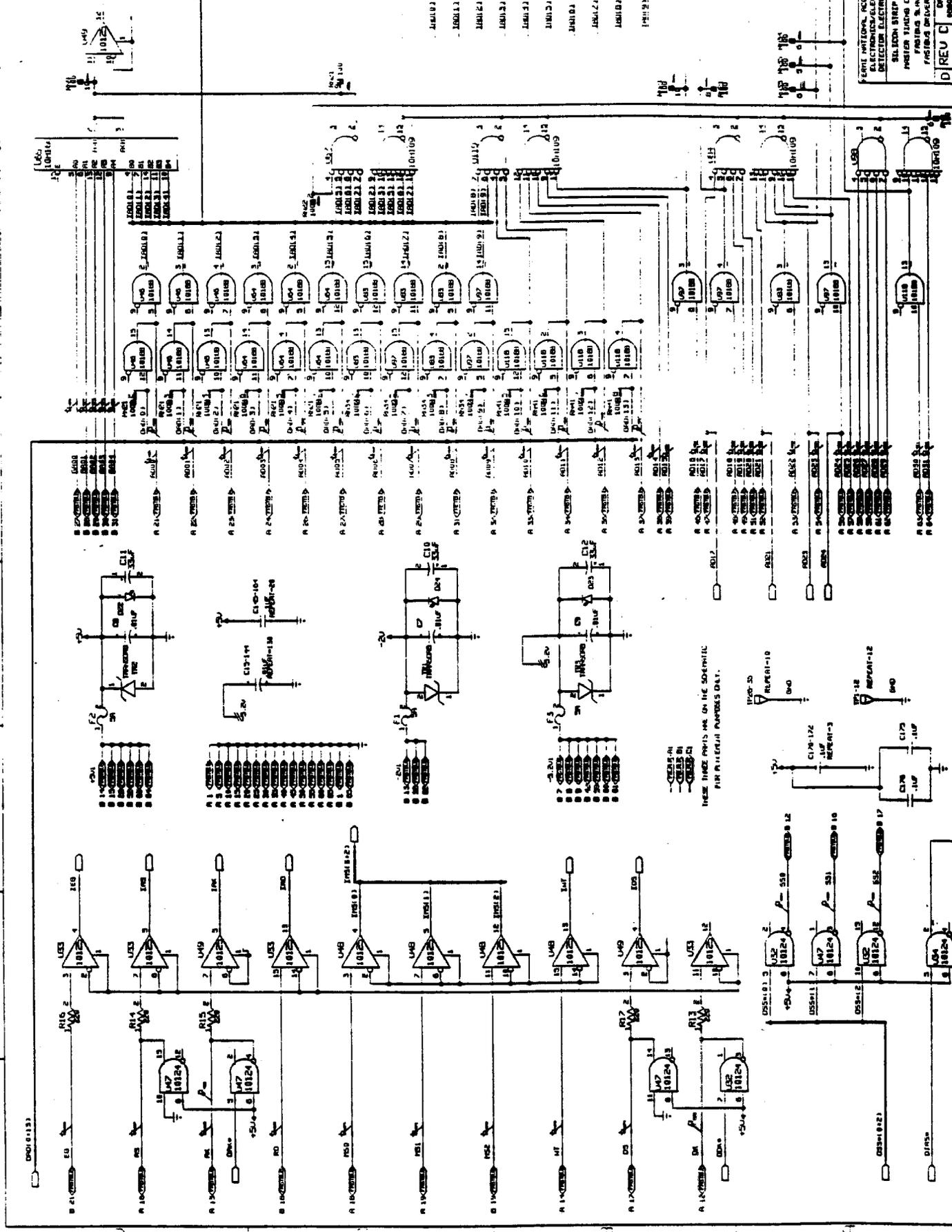
USED ON: MTC FILES: MTC

REV D

DESIGNED BY: J. J. JONES
 DRAWN BY: J. J. JONES
 CHECKED BY: J. J. JONES

00000
 00000
 00000

1 2 3 4 5 6 7 8

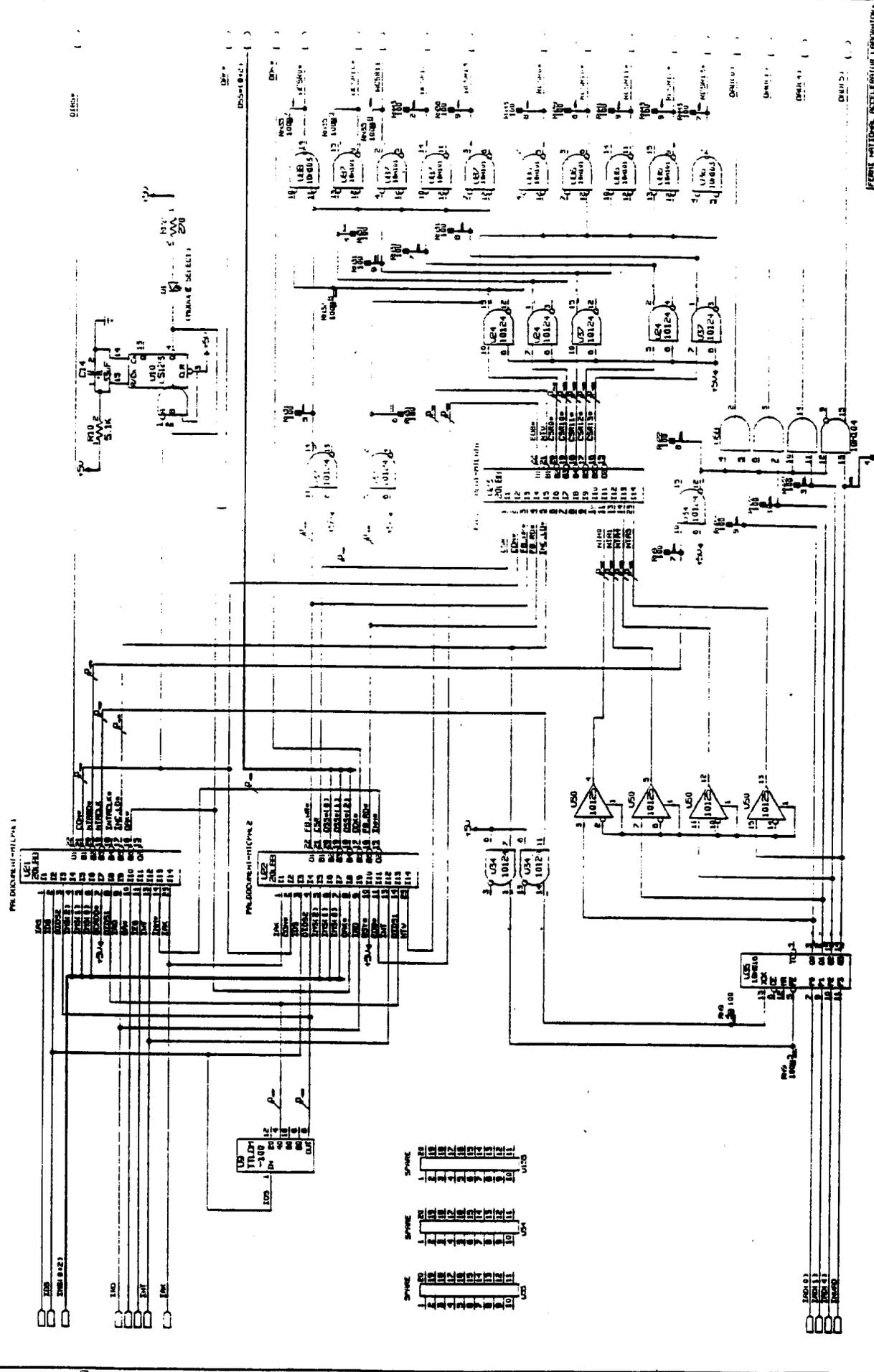


U101 10111
 U102 10111
 U103 10111
 U104 10111
 U105 10111
 U106 10111
 U107 10111
 U108 10111
 U109 10111
 U110 10111
 U111 10111
 U112 10111
 U113 10111
 U114 10111
 U115 10111
 U116 10111
 U117 10111
 U118 10111
 U119 10111
 U120 10111
 U121 10111
 U122 10111
 U123 10111
 U124 10111
 U125 10111
 U126 10111
 U127 10111
 U128 10111
 U129 10111
 U130 10111
 U131 10111
 U132 10111
 U133 10111
 U134 10111
 U135 10111
 U136 10111
 U137 10111
 U138 10111
 U139 10111
 U140 10111
 U141 10111
 U142 10111
 U143 10111
 U144 10111
 U145 10111
 U146 10111
 U147 10111
 U148 10111
 U149 10111
 U150 10111
 U151 10111
 U152 10111
 U153 10111
 U154 10111
 U155 10111
 U156 10111
 U157 10111
 U158 10111
 U159 10111
 U160 10111
 U161 10111
 U162 10111
 U163 10111
 U164 10111
 U165 10111
 U166 10111
 U167 10111
 U168 10111
 U169 10111
 U170 10111
 U171 10111
 U172 10111
 U173 10111
 U174 10111
 U175 10111
 U176 10111
 U177 10111
 U178 10111
 U179 10111
 U180 10111
 U181 10111
 U182 10111
 U183 10111
 U184 10111
 U185 10111
 U186 10111
 U187 10111
 U188 10111
 U189 10111
 U190 10111
 U191 10111
 U192 10111
 U193 10111
 U194 10111
 U195 10111
 U196 10111
 U197 10111
 U198 10111
 U199 10111
 U200 10111

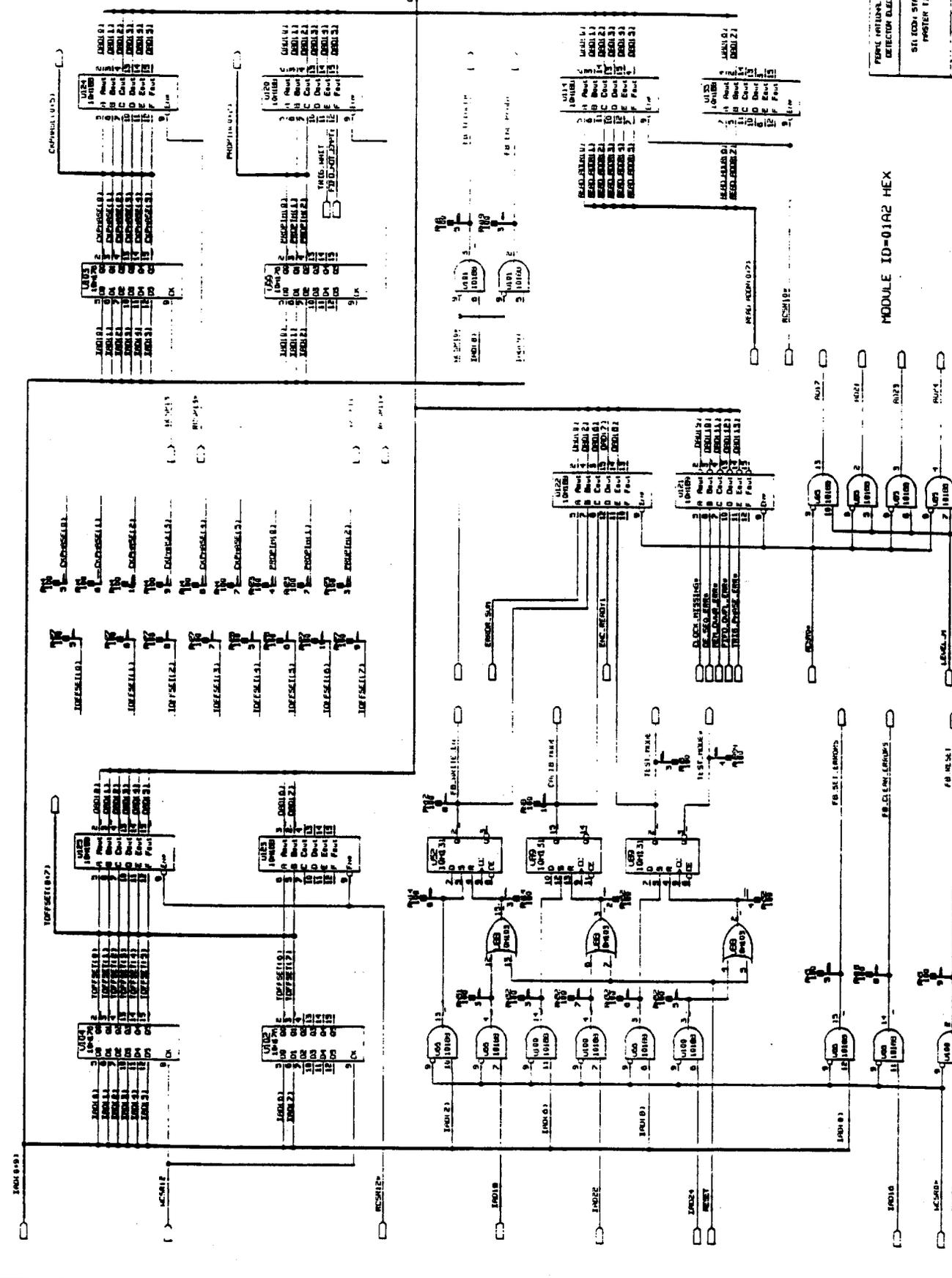
REVISIONS
 1. ORIGINAL DESIGN
 2. REVISED FOR...
 3. REVISED FOR...
 4. REVISED FOR...
 5. REVISED FOR...
 6. REVISED FOR...
 7. REVISED FOR...
 8. REVISED FOR...
 9. REVISED FOR...
 10. REVISED FOR...

DATE: 10/10/70
 DRAWN BY: J. SMITH
 CHECKED BY: M. JONES
 APPROVED BY: R. BROWN

1 2 3 4 5 6 7 8



8 7 6 5 4 3 2 1

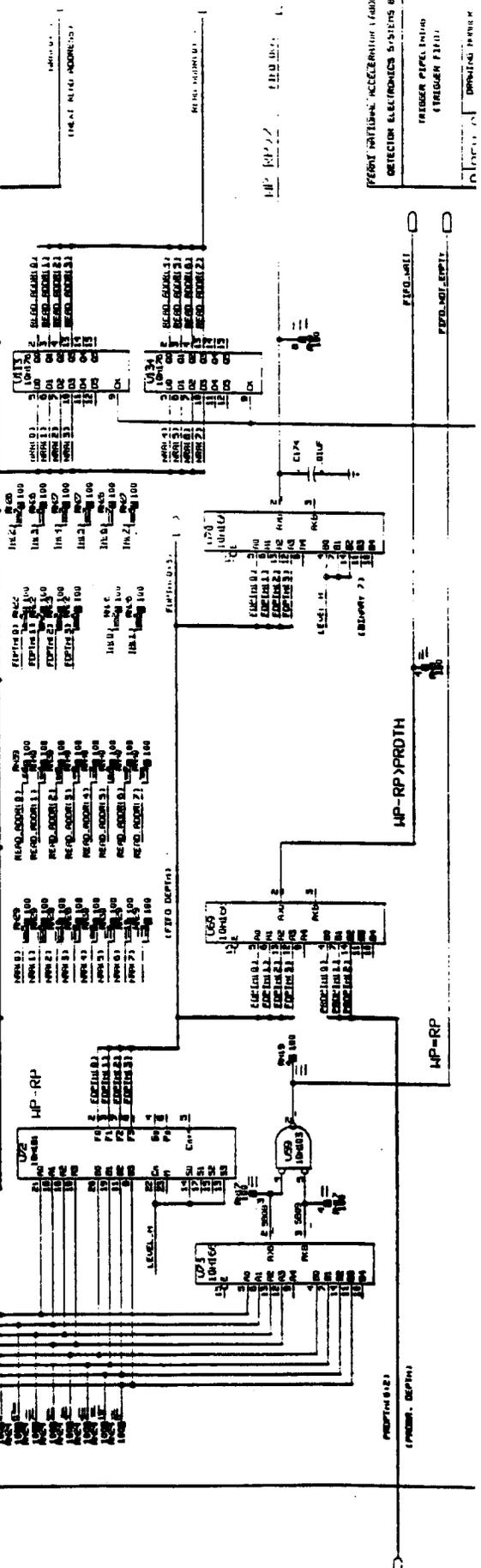
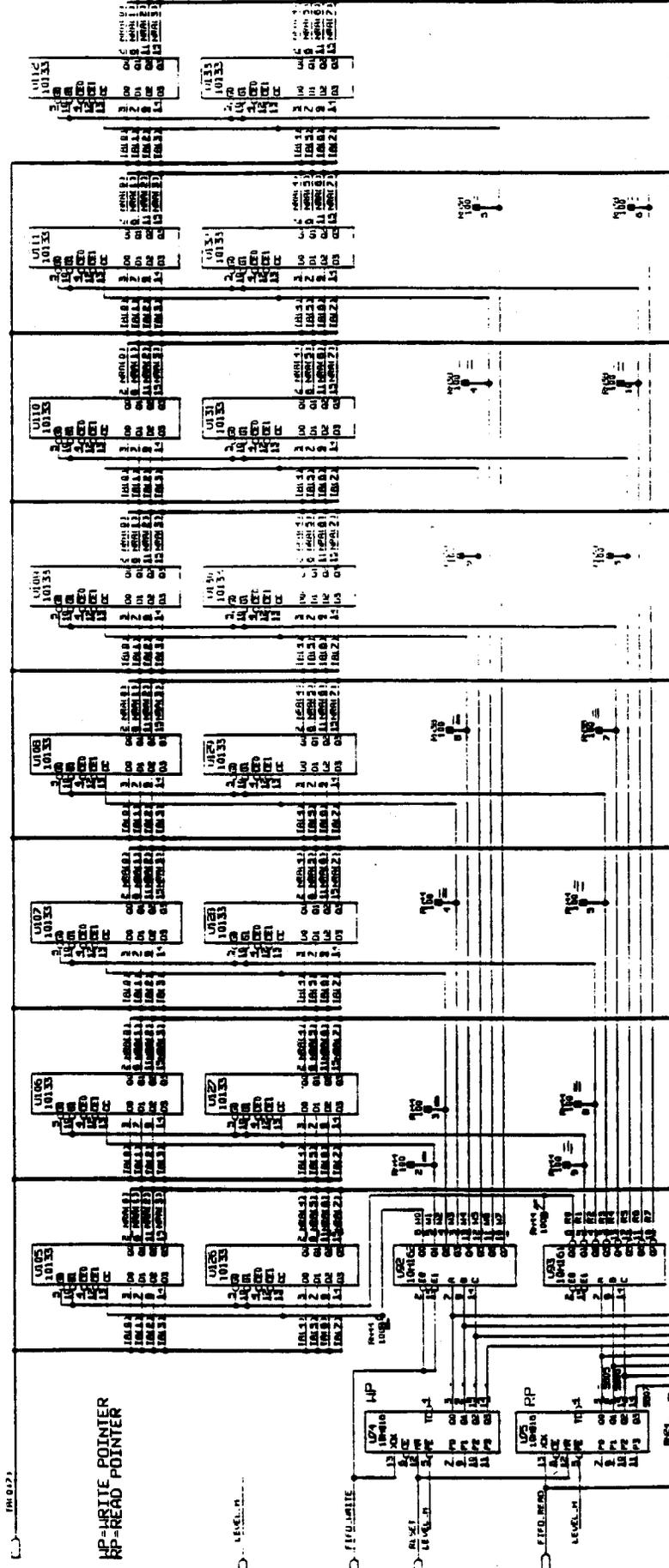


MODULE ID-01A2 HEX

PERFECTIONE KELLER-ROHM LABORATORIES
 DETECTOR ELECTRIC SYSTEM GROUP
 SILICON STRAP DETECTION SYSTEM
 MASTER LINE CONTROL
 COM-3
 DRAWING NUMBER
 3-9

DREV D

1 2 3 4 5 6 7 8



APPENDIX B

PAL EQUATIONS USED IN THE FASTBUS INTERFACE

MODULE MTC_NTA

TITLE FASTBUS NTA decode PAL for SSD Master Timing Controller Module
 Ken Treptow -- FERMILAB
 Sep 25, 1989 -- Revised Sep 25, 1989

MTCNTA DEVICE 'P20V8C'; " normally P20L8

"Inputs:

CSR,!CON,!FB_WR,!FB_RD,INC PIN 1,2,3,4,5 ;
 nc6,nc7,nc8,nc9,nc10 PIN 6,7,8,9,10 ;
 NTA0,NTA1,NTA4,NTA5 PIN 11,13,14,23 ;

"Outputs:

!EOB,NTV,!CSR0,!CSR10 PIN 22,21,20,19 ;
 !CSR11,!CSR12,!CSR13,nc15 PIN 18,17,16,15 ;

"Constant declarations:

X = .X. ;
 ADDR = [NTA5,NTA4,X,X,NTA1,NTA0] ;

Equations

NTV = CON & CSR & !((ADDR == 0) " Not CSR0
 # ((ADDR >= ^h10) & (ADDR <= ^h13))); " Not CSR10-13

CSR0 = CON & CSR & (ADDR == 0) ;

CSR10 = CON & CSR & (ADDR == ^h10) ;

CSR11 = CON & CSR & (ADDR == ^h11) ;

CSR12 = CON & CSR & (ADDR == ^h12) ;

CSR13 = CON & CSR & (ADDR == ^h13) ;

EOB = CON & CSR & INC & NTV ;

END MTC_NTA

MODULE MTC_PAL1

TITLE FASTBUS Slave PAL1 forr SSD Master Timing Controller Module
 This is a modified FASTBUS Slave PAL1 from E706s ICBM Module
 Ken Treptow -- FERMILAB
 Aug 25, 1989 -- Revised Aug 25, 1989 '

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
 " when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP

"DATE 1986 OCTOBER 20
 "CHIP FB009_SCL2 PAL20L8

MTCPAL1 DEVICE 'P20V8C'; " normally P20L8

"Inputs:

IAS,IDS,DIDS2,IMS2,IMS1,IMS0 PIN 1,2,3,4,5,6 ;
 !BCADD,DIDS1,IRD,!GA PIN 7,8,9,10 ;
 IEG,IWT,!INH,IAK PIN 11,13,14,23 ;

"Outputs:

!DIAS,!CON,!NTARD,NTACLK PIN 22,21,20,19 ;
 !INTACLK,INC_LD,!OAK,nc PIN 18,17,16,15 ;

equations

DIAS = IAS ; "Delayed Input AS

"CONnected (attached)

CON = IAS & !IAK & !IRD & !IMS2 & !IMS1 & IMS0 & GA & IEG "Geographical Add CS
 # IAS & !IAK & !IRD & !IMS2 & IMS1 & BCADD "Broadcast Address
 # IAS & CON ; "latch while AS is up

"NTA ReaD

NTARD = CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2 "set on DS up
 # NTARD & CON & IRD & !IMS2 & IMS1 & !IMS0 & IDS & DIDS2 ;
 "latch until new MS or Write and DS,DK up

"NTA CLoCK

NTACLK = INTACLK
 # INC_LD & !IAS " Inc NTA when terminating Blk xfer with AS dn
 # NTACLK & IDS & !DIDS2 "Latch while DS up
 # NTACLK & !IDS & DIDS2 ; "Latch while DS down

"Internal NTA CLoCK

(cont. MTC_PAL1)...

```

INTACKL = CON & !INC_LD & !IRD & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2 & !NTACKL
          "NTA write cycle
          # CON & INC_LD & !INH & !IMS2 & IMS0 & IDS & !DIDS2 & !NTACKL
          "Block transfer DS up
          # CON & INC_LD & !INH & !IMS2 & IMS0 & !IDS & DIDS2 & !NTACKL
          "Block transfer DS dn
          # CON & INC_LD & !INH & IRD & !IMS2 & IMS1 & !IMS0
          & IDS & !DIDS2 & !NTACKL          "NTA read cycle
          # CON & INC_LD & !INH & !IMS2 & !IMS1 & !IMS0
          & IDS & !DIDS2 & !NTACKL;          "Single R/W cycle

```

"INCrement or LoaD the NTA

```

INC_LD = !((CON & !INH & IMS0 & IDS & DIDS2) & !INC_LD          "Set if blk
          # !IRD & !IMS2 & IMS1 & !IMS0 & IDS & !DIDS2          "Reset on NTA write
          # !IMS2 & !IMS0 & NTACKL          "Reset on NTA or Single & after NTACKL
          # !CON );          "Reset when disconnected

```

"Output AK

```

OAK = CON & !IAK & !IWT & !BCADD          "Send AK if not broadcast address
          # OAK & CON          "Latch until CON goes away
          # OAK & IWT          "Hold if Wait is asserted
          # OAK & DIDS2 ;          "Stretch until DK is off

```

END MTC_PAL1

MODULE MTC_PAL2

TITLE FASTBUS Slave PAL2 for SSD Sequencer Module
 This is a modified FASTBUS Slave PAL2 from E706s ICBM Module
 Ken Treptow - FERMILAB
 Aug 28, 1989 - Revised Aug 28, 1989

"Note this is a modification of FASTBUS Slave PALS done by L. PREGERNIG
 " when he was at the UNIVERSITY OF ILLINOIS HIGH ENERGY PHYSICS GROUP
 "DATE 1986 OCTOBER 20
 "CHIP FB010_SCL2 PAL20L8

MTCPAL2 DEVICE 'P20V8C'; " normally P20L8

"Inputs:

IAK,!CON,IDS,DIDS2,IMS2,IMS1,IMS0 PIN 1,2,3,4,5,6,7 ;
 !OAK,!IRD,!BSY,!EOB,!WT,DIDS1,NTV PIN 8,9,10,11,13,14,23 ;

"Outputs:

!FB_WR,CSR,!OSS0,!OSS1,!OSS2 PIN 22,21,20,19,18 ;
 !ODK,!FB_RD,!OINH PIN 17,16,15 ;

"Constant declarations

"INHibit data transfers

INH = CON & EOB & !IMS2 & IMS0 "If End Of Block is reached (block or pipeline)
 # CON & BSY "when BuSY
 # CON & NTV "when NoT Valid address is in the NTA
 # IMS2 ; "when bad MS code

Equations

"FastBus WRite strobe

FB_WR = CON & !INH & !IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
 "MS=0 random data write, DS up only
 # CON & !INH & !IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
 "MS=1 block transfer write, DS up
 "MS=3 pipeline transfer write, DS up
 # CON & !INH & !IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2 ;
 "MS=1 block transfer write, DS down
 "MS=3 pipeline transfer write, DS down

"Addressed in CSR

CSR = CON & IMS0 & !IAK "set if CSR at primary address time
 # CSR & CON ; "latch until end of CONNected (attached)

"Output Slave Status bit 0

(cont MTC_PAL2)

```

OSS0 = CON & BSY & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2      "BuSY & single xfer
# CON & BSY & !IMS2 & IMS0 & DIDS1 & !DIDS2      "BuSY and MS=1 or 3 on DS up
# CON & BSY & !IMS2 & IMS0 & !DIDS1 & DIDS2      "BuSY and MS=1 or 3 on DS dn
# CON & NTV & !IMS2 & IMS1 & !IMS0 & DIDS1 & !DIDS2  "NTV & 2nd Address
# OSS0 & IDS & DIDS2 & CON                          "Latch while DS=DK=1
# OSS0 & !IMS2 & IMS0 & !IDS & !DIDS2 & CON ;      "Latch if DS=DK=0 & MS=1or3

```

"Output Slave Status bit 1

```

OSS1 = CON & IMS2      "any MS=4-7 on DS up or down
# CON & !IMS2 & NTV & !BSY & DIDS1 & !DIDS2
  "any Not Valid address if not BuSY on DS up
# CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
  "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
# CON & !IMS2 & IMS0 & EOB & !BSY & DIDS1 & !DIDS2
  "End Of Block if not BuSY on DS up
# CON & !IMS2 & IMS0 & EOB & !BSY & !DIDS1 & DIDS2
  "End Of Block if not BuSY on DS down
# OSS1 & IDS & DIDS2 & CON                          "Latch while DS=DK=1
# OSS1 & IMS0 & !IDS & !DIDS2 & CON ;                "Latch if DS=DK=0 & MS=1,3,5,or 7

```

"Slave Status bit 2

```

OSS2 = CON & IMS2      "any MS=4-7 on DS up or down
# CON & !IMS2 & NTV & !EOB & !BSY & DIDS1 & !DIDS2
  "any Not Valid address if not BuSY and not End Of Block on DS up
# CON & !IMS2 & IMS0 & NTV & !EOB & !BSY & !DIDS1 & DIDS2
  "any Not Valid address, not BuSY, not End Of Block, & MS=1or3 on DS down
# OSS2 & IDS & DIDS2 & CON                          "Latch while DS=DK=1
# OSS2 & IMS0 & !IDS & !DIDS2 & CON ;                "Latch if DS=DK=0 & MS=1,3,5,or 7

```

"Output Data acKnowledge generates DK

```

ODK = !IWT & CON & OAK & DIDS2      "set if DS (delayed) and attached and not Wait
# IMS1 & IMS0 & CON & OAK & DIDS2
  "set if DS (delayed) and attached and MS=3 (pipeline) even if Wait
# ODK & CON & OAK & DIDS2            "transition hold while DS (delayed)
# IWT & !IMS1 & ODK                  "hold if Wait and not MS1
# IWT & !IMS0 & ODK ;                "hold if Wait and not MS0
  "i.e. hold if not MS=3 (pipeline) AND Wait, release it otherwise

```

"FastBus Read

```

FB_RD = CON & !INH & IRD & !IMS2 & !IMS1 & !IMS0 & DIDS1 & !DIDS2
  "set on DS up, MS=0 random data read
# CON & !INH & IRD & !IMS2 & IMS0 & DIDS1 & !DIDS2
  "set on DS up, MS=1 block read
  "set on DS up, MS=3 pipeline read
# CON & !INH & IRD & !IMS2 & IMS0 & !DIDS1 & DIDS2
  "set on DS dn, MS=1 block read
  "set on DS dn, MS=3 pipeline read
# CON & IRD & !IMS2 & !IMS1 & !IMS0 & IDS & DIDS2 & FB_RD
  "latch while MS=0 read and DS,DK up
# CON & IRD & !IMS2 & IMS0 & FB_RD
  "latch while MS=1,3 read
# CON & IDS & DIDS2 & FB_RD          "latch while DS,DK up
# CON & !IDS & !DIDS2 & FB_RD ;      "latch while DS,DK down
  "i.e. latch until new MS or WR cycle

```

(cont MTC_PAL2)

"Output INHibit data transfers

OINH = INH

CON & !CSR ;

"This stops NTA Incrementing for the FIFO in Data Space

END MTC_PAL2

APPENDIX C

PARTS LIST

MTC PARTS

No.	MTC Part No.	Manuf	Manuf PART No.	FNAL Stock No.	Description	Qty	Cost	Cost/brd
Resistor								
1	R3,8	AB		1487-0385	10K 1/8W Res	1	0.13	0.13
2	R6,7,9,11,12			1478-0247	51 ohm 1/8w Res	5	0.16	0.80
3	R13-R17,19,21			1487-0285	220 ohm 1/8w Res	7	0.16	1.12
4	R18,20			1487-0620	100 ohm 1/4w Res	2	0.06	0.12
5	R1,R5			1487-0590	56 ohm 1/4w Res	2	0.06	0.12
6	R10			1487-0830	5.1K ohm 1/4w Res	1	0.06	0.06
7	R2,4			1487-0720	620 ohm 1/4w Res	2	0.04	0.08
8	RN1-RN44	AB	4610X-101		100 ohm 10-pin SIP	44	0.32	14.08
9	RP1-RP3	AB	4610X-101		270 ohm 10-pin SIP	3	0.32	0.96
10	RP4-RP6	AB	4610X-101		330 ohm 10-pin SIP	3	0.32	0.96
Integrated Circuits								
11	U116-U118	Motorola	10101P	1455-5801	Quad OR/NOR Gate	1	0.31	0.31
12	U9,137	Motorola	10114P		Triple Line Receiver	1	0.76	0.76
13	U25,33,35,38,48	Motorola	10124P	1455-5824	Quad TTL to ECL Translator	5	1.52	7.60
14	U8,34,49,50,51	Motorola	10125P	1455-5825	Quad ECL to TTL Translator	5	1.22	6.10
15	U106-U113	Motorola	10133p	1455-5833	Quad latch	16		
16	U127-U134							
16	U47,65,67,84,86 U98,101,102 U119	Motorola	10188P		Hex Buffer w/enable	9	2.20	19.80
17	U20,21,31,32,46 U64,83,97	Motorola	10192P		Quad Bus Driver (ECL to Nim)	8	3.90	31.20
18	U1	Motorola	10198P		Monostable Multivibrator	1	8.85	8.85
19	U29,87,88	Motorola	10H101P		Quad OR/NOR Gate	3	0.64	1.92
20	U15,19	Motorola	10H102P		Quad 2 Input NOR Gate	2	0.64	1.28
21	U37,39,42,54,60,89	Motorola	10H103P		Quad 2 Input OR Gate	6	0.64	3.84
22	U28,45,61	Motorola	10H104P		Quad 2 Input AND	4	0.64	2.56
23	U16	Motorola	10H105P		Triple 2-3-2 Input OR/NOR	1	0.64	0.64
24	U30,68,85,99,120	Motorola	10H109P		Dual 5-4 Input OR/NOR	5	1.49	7.45
25	U75-U78 U36,44,95,96	Motorola	10H016P		4-Bit Binary counter	8	7.46	59.68
26	U3,4,26,52	Motorola	10H125P		Quad ECL to TTL Translator	4	2.00	8.00
27	U40,41,43,53,57 58,59,90,	Motorola	10H131P		Dual D type F/F	8	1.90	15.20
28	U12-U14	Motorola	10H135P		Dual JK MS Flip Flop	3	2.30	6.90
29	U94	Motorola	10H161P		Binary TO 1-8 Decoder (Low)	1	2.44	2.44
30	U72,93	Motorola	10H162P		Binary TO 1-8 Decoder (High)	2	2.44	4.88
31	U62,63,66,70,71,74	Motorola	10h166P		5-Bit magnitude comparator	6	3.24	19.44
32	U100,114,135	Motorola	10H176P		Master-slave flip-flop	6	3.54	21.24
33	U103-U105							
33	U79-U82,73	Motorola	10H181P		4-Bit a/b/function generator	5	9.95	49.75
34	U123-U126 U115,121,136	Motorola	10H188P		Hex Buffer w/enable	1	1.45	1.45
35	U122	Motorola	10H189P		Hex Inverter w/enable	1	2.20	2.20
36	U7,U11 U6	T.L	74LS123N 74LS02	1455-8123 1455-8002	Dual Retriggerable Monostable Nor gate	2 1	0.59 0.23	1.18 0.23
Switches								
37	SW2,SW3	CTS Corp.	P/N 206-8s	1455-9708	8 section dip switch	2	0.89	1.78
	SW5	C&K	3M120		10 Pos. thumbwheel switch CW811	1	4.65	4.65
38	SW6	CTS Corp.	P/N 206-4s	1455-9704	4 section dip switch	1	0.75	0.75
39	SW1	C&K	MODEL TP11		tiny pushbutton switch	1	4.05	4.05
Connectors								
40	L1-L16	KINGS	K-LOCK P/N1077-3	1435-4400	Lemo PC mont	32	4.89	156.48
41	J1	3M	P/N 3431-5302	1435-7105	34pin right angle header	1	1.61	1.61
42	FBSEG A,FBSEG B	AMP	1-102585-3		FASTBUS 130 SOCKET CONNE	1	7.87	7.87
Diodes								
43	D2,4,6,8,11,13,14 D15,22,23,24	H.P.	P/N HLMP-1503	1445-0470	green LED	11	0.24	2.64
44	D3,5,7,9,12	H.P.	P/N HLMP-1301	1445-0475	red LED	5	0.24	1.20
45	D16-D20,D1 D10,D21	H.P.	P/N HLMP-1401 1N4001	1445-0495 1445-1550	yellow LED Signal diode	6 2	0.24 0.03	1.44 0.06
Fuse								
46	F1-F3	Linifuse	type 251005	1120-0250	picofuse 5A	3	0.48	1.44
Capacitors								
47	C165,C167	ERIE	8131-100-651-334M	1415-3170	.33ufd ceramic cap	2	0.18	0.36

MTC PARTS

No.	MTC Part No.	Manuf	Manuf PART No.	FNAL Stock No.	Description	Qty	Cost	Cost/brd
48	C13	SPRAGUE	P/N 10TS-T10	1415-2110	100pfd	1	0.14	0.14
49	C173	SPRAGUE	P/N 10TS-T47	1415-2150	470pfd cap	1	0.16	0.16
50	C3	SPRAGUE	P/N 10TS-D10	1415-2170	1000pfd cap	2	0.18	0.36
51	C174	SPRAGUE		1415-2160	680pfd	1	0.06	0.06
52	C10-C12,14	MALLORY	CSR13-C336K	1425-1180	33ufd cap	4	0.45	1.80
53	C145-164,C170-172	SPRAGUE	923CZ5U104M050B		.1ufd dip cap	29	0.42	12.18
54	C4,61,68,175,176							
55	C7-C9,C15-C144	SPRAGUE	923ca7r103k050b		.01ufd dip cap	137	0.36	49.32
56	C12,169							
57			Delay Line					
58	U8	EC2	TTLDM-100T		DELAY LINE	1	15.20	15.20
59	DL1	EC2	ECLDL-80		DELAY LINE	1	13.20	13.20
60	U2,U25	EC2	ECL-100K-LDM-16		DELAY LINE	2	48.70	97.40
61	U17	ELMEC	PDH6500		DELAY LINE	1	85.00	85.00
62	U16	ELMEC	FDD9010		DELAY LINE	1	9.00	9.00
63								
64			Hardware					
65					MTC front panel 800.000-MD-269	1	40.00	40.00
66	HW				NYLON SPACER .625LONGX4-	7	0.10	0.70
								812

APPENDIX D

MTC TEST MODULE DIAGRAM

APPENDIX E

CORRECTIONS TO THE PRINTED CIRCUIT BOARD

The 2nd change was required to make the front panel TRG MON and TRG WIN signals have the right timing, as observed with a scope, for adjusting the external trigger delay to the module. In order to accomplish this, the trigger signal to the NIM driver was taken from a different point in the circuit (also an extra NAND gate was added, mainly for not messing with the ECL terminations and not to run the signal too long distances; the available gate was found in U45, inputs 12 & 13 of a 10H104, and the input signal to the gate was taken from the same U45, pin 4). The changes are shown in Fig. E-3.

The above changes in the schematics are updated in the respective Cadnetix files. The 2nd modification, however, is not handled by the Cadnetix system, since it makes use a heterogeneous gate in the 10H104 package. A note is posted in the schematic such that new PCB designs will have to find a way to bypass this problem.

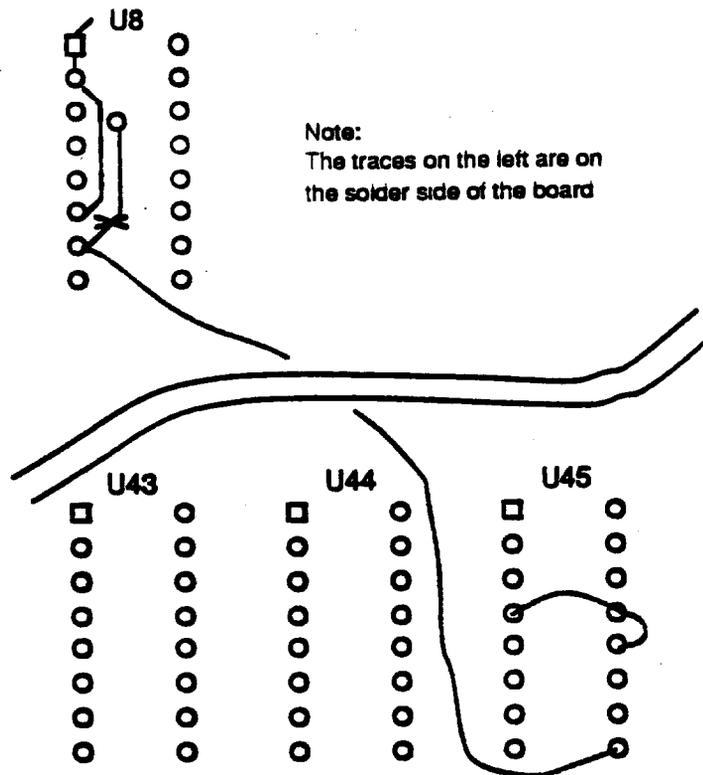


Fig E-3 - Changes to retime the TRG MON signal on the front panel coax connector