E706

NEXT GENERATION DIGITAL READOUT ELECTRONICS FOR THE E706 LIQUID ARGON CALORIMETER

By

Charles Robert Benson

Submitted in Partial Fulfillment

of the

Requirements for the Degree

MASTER OF SCIENCE

Supervised by Dr. Fred Lobkowicz

Department of Physics and Astronomy

University of Rochester

Rochester, New York

1989

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. Fred Lobkowicz for his encouragement and support of this thesis. I would also like to thank Ernest Buchanan for his unending patience in leading me into the world of electronics. My fellow members of the E706 project Eric Prebys, Chris Lirakis, George Ginther, and Lucyna DeBarbaro among others provided invaluable assistance in the understanding of the E706 experiment and the requirements of the project. Thanks and appreciation is also due to John Borders and Bob Hirosky of the D0 experiment at the University of Rochester cyclotron lab for introducing me to that facility and lending a skilled and knowledgeable hand.

This work was supported by grants from the National Science Foundation and the Department of Energy. Without their financial support research such as the E706 experiment would not be possible.

Finally but foremost I would like to thank my family in particular my parents for their moral support and encouragement during my education.

VITAE

The author was born in Syracuse New York. He graduated from Christian Brothers Academy in 1984 and received a Bachelor of Science degree with majors in Mathematics and Physics from St. Bonaventure University in 1987. The author entered the University of Rochester in 1987 and held a teaching assistantship for the first year. He joined the High Energy Particle Group's E706 experiment in 1988.

ABSTRACT

The data collection systems for the Liquid Argon Calorimeter used in the E706 direct photon experiment at the Fermi National Accelerator Lab is discussed. The need for replacement of the MX signal processor computer is shown. The design of the ICBM intelligent data boxcar is discussed in general. The design of the WOLF nonintelligent data boxcar is discussed in detail. Testing results from these systems are included.

TABLE OF CONTENTS

		Page
ACKNOWLEDGEMENTS		ii
VITAE		iii
ABSTRACT		iv
LIST OF FIGURES		vi
LIST OF TABLES		vii
CHAPTER		
1	INTRODUCTION	1
2	CURRENT LAC READOUT DESIGN	2
3	NEW LAC READOUT DESIGN	5
4	ICBM DESIGN	8
5	WOLF DESIGN	11
6	ICBM-WOLF OPERATIONS	26
7	TESTING PROCEDURES AND RESULTS	32
	BIBLIOGRAPHY	35
APPENDIX		
A	WOLF SCHEMATIC DIAGRAM	
B	WOLFTEST SCHEMATIC DIAGRAM	
C	ICBM BLOCK DIAGRAM	
D	ICBM-WOLF INTERFACE TIMING	
E	WOLF TEST SOFTWARE	

LIST OF FIGURES

Figure		Page
1	Current LAC readout system	4
2	New LAC readout system	7
3	WOLF Timing	21

LIST OF TABLES

Table		Page
1	Data line to UBUS write correspondence	15
2	UBUS to data line read correspondence	15
3	Interrupt signals	25

CHAPTER 1 - INTRODUCTION

The E706 experiment at the Fermi National Accelerator Lab is a study of direct photons produced in hadronic collisions. A fixed target experiment, it utilizes a large liquid argon calorimeter (LAC) to analyze the energy and momentum of incident particles. The experiment had its first run in 1987-88 with satisfactory results.[1] However, the readout electronics of the LAC, in particular the digital system are undergoing refit to phase out obsolete equipment and increase performance. The goals of this refit are to decrease maintenance problems and increase data throughput speed to allow more information to be readout from the LAC. This will result in more data per event and more events by increasing the acceptance rate since the readout of the LAC takes more time then any other system on the experiment.

This paper will examine the reasons for the refit and concentrate on the design of the new equipment involved.

---

1. "Preliminary Results from Fermilab Experiment E-706 - A Study of Direct Photon Production in Hadronic Collisions," The E706 Collaboration, 4 June 1988, Submitted to XXIV International Conference on High Energy Physics.

CHAPTER 2 - CURRENT LAC READOUT DESIGN

The current LAC readout electronics are shown in figure 1. The LAC is a set of parallel plate capacitors with one plate set to high voltage. Current leakage through these capacitors represents incident particles. The signals are fed into the LACAMP modules sitting in RABBIT crates located on top of the dewar in a Faraday cage. The LACAMP charge integrates the signal and utilizing trigger information compares before and after incident conditions to compute a number that is related to total energy deposited on the capacitor plates. The EWE module in the RABBIT crate contains an analog to digital converter, and given instruction over the UBUS, the EWE will convert the LACAMP analog signals to digital information and pass it down to its controller via the same UBUS.[2] These controllers are MX computers which are located in the counting house. The MXs are controled and readout by the 1821 FASTBUS controller via the MEP interface board. The 1821 is in turn controlled and readout by the VAX computer which stores the data on magnetic tape.

There are 12 MX computers and each of them can control up to eight EWEs. There can be two EWEs in each rabbit crate since the rabbit crates have two independent controlling buses, but it was found that the system when operated in this manner produced too much noise. Each Rabbit crate can have as many as 450 inputs from the LAC.

The majority of problems in operating the systems during the first run were centered around the MX computers. The MX boards are

---

2.D. Skow, "FEED for the RABBIT," 9 June 1986, E706 Notes.

large wire-wrapped ECL digital devices that are difficult to error trace. Built from discrete components to increase speed, they have a limited instruction set, thus making programming difficult. Another problem was the large number of EWEs connected to each MX that reduced the readout speed.

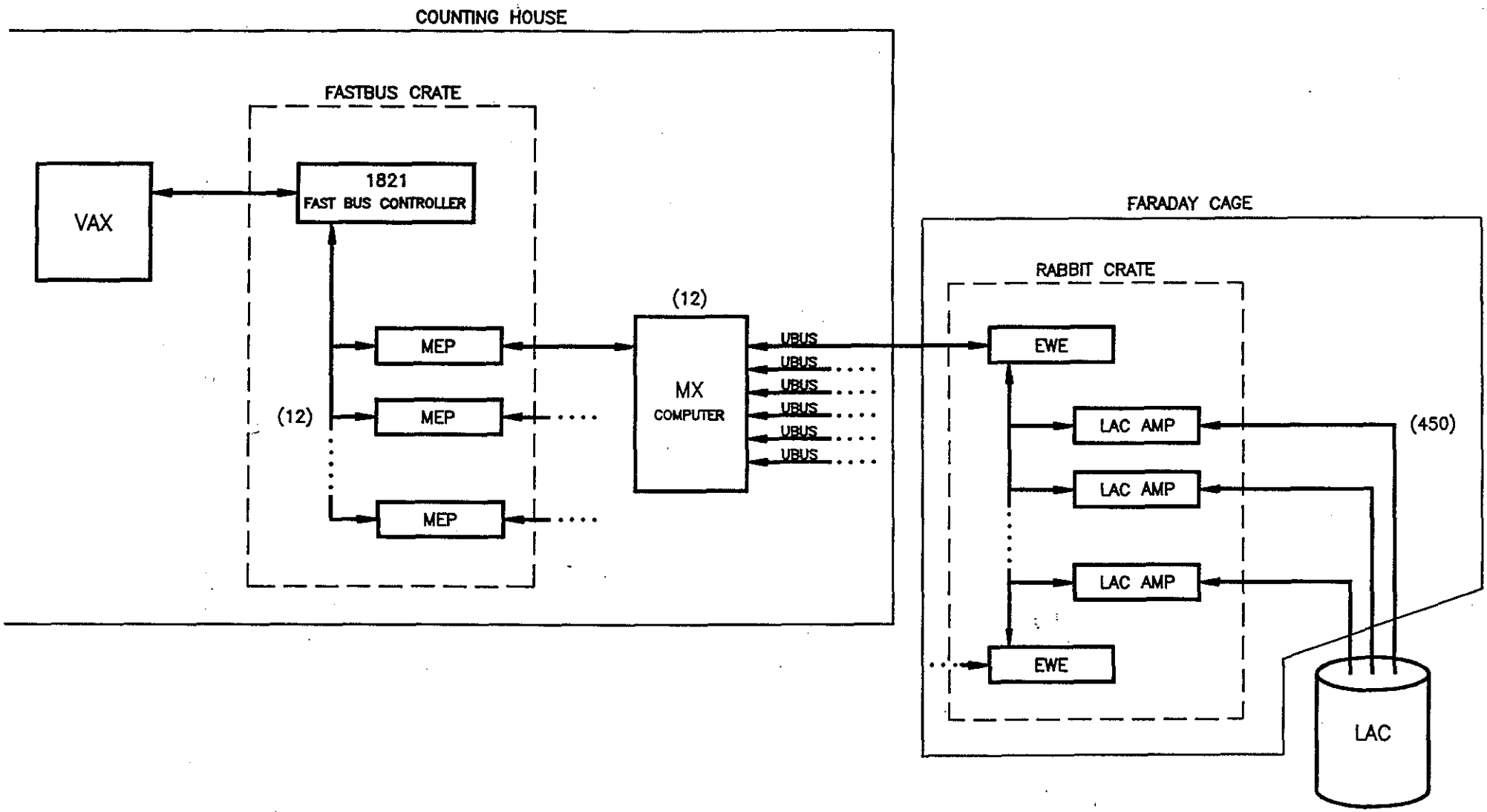


Figure 1 – CURRENT LAC READOUT DESIGN

CHAPTER 3 - NEW LAC READOUT DESIGN

In light of these problems associated with the MX computers it has been decided to replace them by a new system. There now exists general purpose high speed digital signal processor microcomputers on a single chip. These chips are generally fast TTL; thus memory and interface electronics can use readily available TTL devices instead of the more difficult ECL technology. Due to the low cost of these digital signal processors a larger number of them can be used, reducing the number of EWEs connected to a controller and increasing throughput speed.

Because the MX board contained more then just the processor, replacements of much of the interface circuitry would also be necessary. A new connection to the VAX system would be needed while the UBUS system would still be utilized and interface circuitry to it would have to be developed.

Figure 2 shows the new LAC readout design. The signals from the LAC are read into the RABBIT crates in the same manner as before. The EWEs are read out over the existing UBUS systems by new ICBM - WOLF computers. The ICBM and WOLF cards sit in a FASTBUS crate. The ICBM contains the digital signal microprocessor and related circuitry while the WOLF card contains the interface between the ICBM and two UBUS cable systems. The 1821 FASTBUS controller reads the ICBM cards and passes the information as before to the VAX computer. The advantages of this arrangement are the elimination of the troublesome MX computers (and associated hardware: a large cabinet, power supplies ...), the decrease in the number of EWEs connected to a controller from 8:1 to

2:1, and the utilization of a general purpose digital signal processor with larger memory and more versatile instruction set compared to the limited MX system.

The prototype ICBM card was designed and constructed by a group at FERMILAB headed by Eric Prebys. The prototype WOLF card was designed and constructed at the University of Rochester Cyclotron Lab by the author.

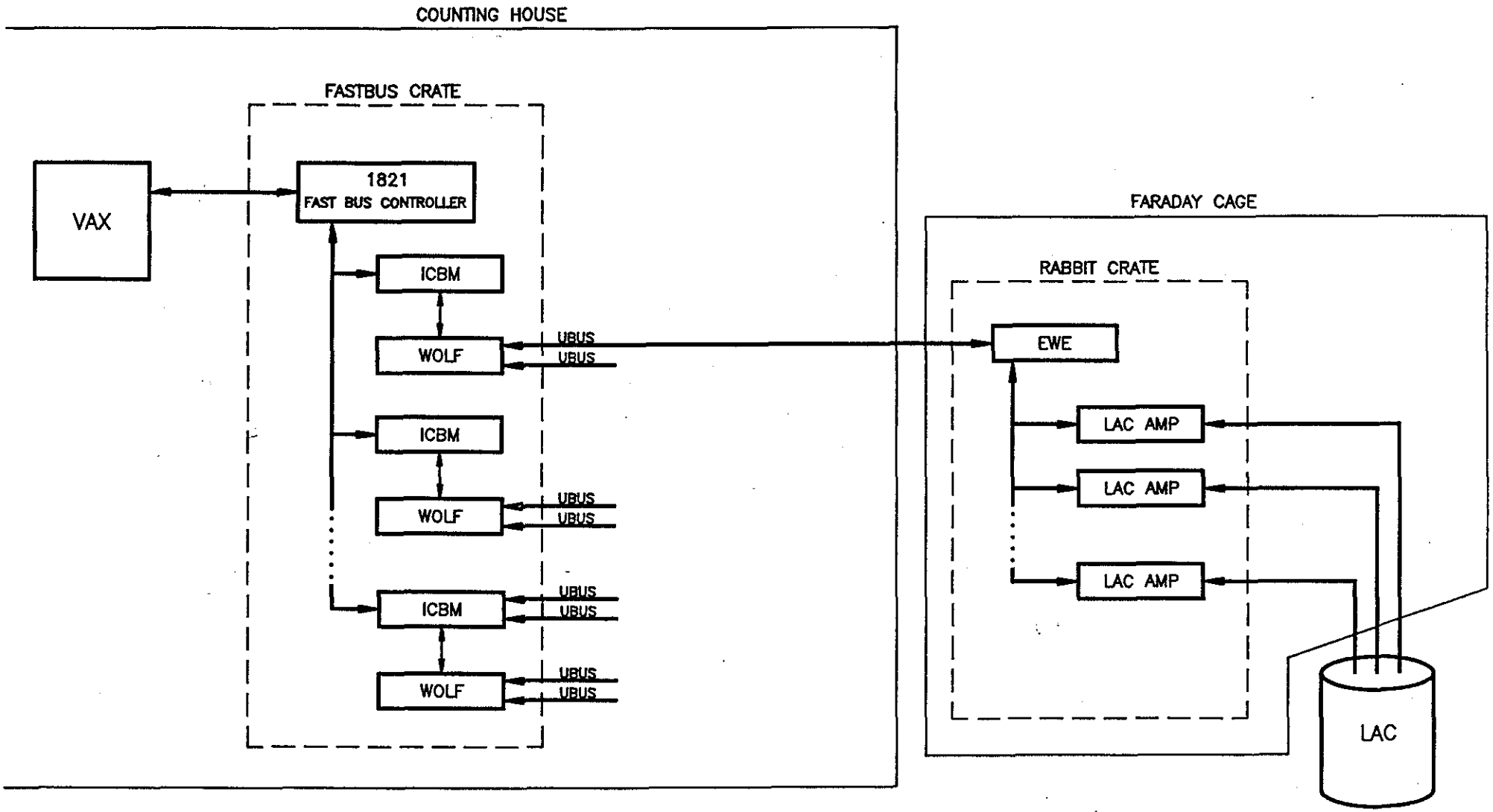


Figure 2 — NEW LAC READOUT DESIGN

CHAPTER 4 - ICBM DESIGN

The ICBM card containing the Digital Signal Processor (a Motorola 56001 DSP) was designed as a general purpose FASTBUS slave computer. The ICBM is an intelligent data boxcar with one end tied to FASTBUS and its controller and the other end tied to a general purpose I/O bus system. As far as the DSP is concerned this bus represents 64 locations in memory that it can read from and write to. All of the DSP's memory is directly accessible to FASTBUS to facilitate programming and data transfer. Because all of the ICBM memory space is accessible to the 1821 FASTBUS controller, the 1821 in principle can bypass the ICBM card and communicate with the WOLF card directly. The normal mode of operation, however, is for the ICBM card to be independently reading in the EWEs and storing data in its memory for the 1821 to access and read out. Being the work of others, the interface between the DSP and FASTBUS on the ICBM card is beyond the scope of this paper. We shall concentrate the rest of this chapter on the interface between the ICBM and the WOLF card.

The interface between the ICBM and the WOLF system is a general purpose I/O bus system. It has 49 lines and uses two 34 pin ribbon connectors on the front panels of both the ICBM and Wolf Cards. Besides an addressable 24 bit bi-directional data bus, the I/O system contains an 8 bit function control mask which allows the DSP to set operating characteristics on the WOLF card. There are also 8 interrupt control lines that allow the WOLF card to signal the DSP. These interrupts are used to notify the DSP that some task has been accomplished by the WOLF card or the EWE system. Below is a discussion

on each line of the I/O bus. The PT or PB designate connector and pin number for the line. PT refers to the top connector and PB refers to the bottom connector.

A0-5, PT3-8: Address lines. These lines allow the DSP to address data to certain registers on the Wolf card.

WR\, PT10: Write. When PER_SEL\ is low and WR\ is low then the DSP is sending data to the register specified by the address lines.

RD\, PT11: Read. When PER_SEL\ is low and RD\ is also low then the DSP wants to read data from the register specified by the address lines.

PER_SEL\, PT13: PER_SEL\ is a low going enable line that specifies that a peripheral operation in in effect as opposed to a read or write operation internal to the ICBM.

INT0-7, PT16-23: INT0-7 are negative edge interrupts. When the WOLF card causes a high to low transition to occur on one of these lines an interrupt is triggered in the DSP. The ICBM card contains circuitry to tell the DSP which line caused the interrupt and also allows the DSP to mask the interrupt, hence disabling it.

MASK0-7, PT24-31: MASK0-7 are outputs from the ICBM and can be used as general TTL control signals from the DSP to the Wolf.

DATA0-23, PB3-26: DATA0-23 is the Bi-directional data bus active during a read or write operation.

The timing of the bus has been designed to be the same as the read or write timing of the motorola 56001 DSP. Refer to appendix E for timing diagrams.

CHAPTER 5 - WOLF DESIGN

The WOLF system is a nonintelligent data boxcar connected at one end to the ICBM system and to two EWEs on the other. The WOLF card has one primary function, to execute read or write commands from the ICBM on the UBUS. To do this it must change the signals from TTL to ECL logic levels and produce accompaning timing signals so that the EWE, another non intelligent data boxcar, will accept the data. The WOLF card has numerous secondary functions such as providing error checking on the UBUS, properly terminating the UBUS and providing interrupt signals to the ICBM based on EWE conditions.

Before studying the WOLF system in detail let's examine the UBUS and the EWE. The EWE communicates via eight registers to which its controller writes instructions and from which it reads information. Reading and writing to these registers is done over the UBUS which utilizes differential ECL signaling. This indicates that for every signal there are two lines, one showing ECL positive logic and the other showing ECL negative logic. This arrangement is essential because of the great cable lengths between the WOLF and EWE. Below is a list of channels on the UBUS and a description of their functions. The UBUS is split into two cables, a 40 conductor cable labeled U40 and a 26 conductor cable labeled U26. Numbers after the cable label represent pin numbers. Odd pins are positive ECL logic while even pins are negative ECL logic.[3]

3.K. Turner and S. Littleton, "MX, EWE-II and UNIBOX Registers, PIN 88," 3 October 1986, E706 Notes.

UDAT15-UDAT0, U40 1-37: The UBUS differential bi-directional data channels.

URS2, U40 33-34: A differential EWE status channel indicating a parity error on the UBUS. also known as the EWE ERROR.

URS1, U40 35-36: A differential EWE status channel indicating that the EWE's analog signal is greater then a set threshold value. Also known as EWE DATA.

URS0, U40 37-38: A differential EWE status channel indicating that the EWE has finished the last command given it. Also known as EWE DONE.

UADR1-UADR0, U26 1-4: The differential EWE address lines. It is possible to connect up to four EWE's on one UBUS by addressing them with these lines. However, E706 will utilize one EWE per UBUS in the upcoming run.

USA2-US0, U26 5-10: The differential EWE register address lines. These lines indicate which register the controller has selected for an operation.

URD, U26 13-14: A differential signal from the controller indicating a read operation is taking place.

USTRB, U26 15-16: A differential signal indicating valid data is on the UBUS and should be strobed into the EWE.

UXQT, U26 11-12: A differential signal from the controller telling the EWE to execute the current instruction in its command register.

UWPAR, U26 17-18: A differential signal from the controller indicating the parity of the current UBUS signal from the controller.

URPAR, U26 19-20 A differential signal from the EWE indicating the parity of the current UBUS signal from the EWE.

CERR26-, U26 26, CERR40-, U40 40: Non differential signals always logically on to indicate that the UBUS is plugged into the the WOLF card.

The plan behind the WOLF system is to have the ICBM write to latches whose outputs would be converted to differential ECL signals. This is done not only to produce the necessary ECL levels on the UBUS but also to match the timing requirements of the UBUS. The speed of the latches is sufficient to allow a match between the fast ICBM timing and the relatively slow UBUS timing. The output of these latches will represent all the UBUS signals that originate from the controller side of the UBUS. Table 1 is a table showing which data lines produce which UBUS signals. An ICBM read from the WOLF card will yield data from latches whose inputs would be signals on the UBUS that originated from

the EWE. Table 2 is a table showing which UBUS lines correspond to which data lines.

Operation on the UBUS will be as follows. To write to a register in a EWE one would construct in software a 24 bit word that contains the EWE address, the register address, the URD bit low, and the UDAT0-15 containing the appropriate data and write this word to the WOLF card. To read from a register in a EWE one would construct a 24 bit word that contains the EWE address, the register address, and the URD bit high and write this word to the WOLF card. Then one reads the WOLF card after a certain amount of time to retrieve the data from the UBUS. The DSP is notified that the data is valid by either continuously reading the WOLF card until the EWE BUSY (DATA 23) signal goes low or by setting up an interrupt to signal the ICBM card when EWE BUSY goes low. To accomplish these tasks the WOLF card handles many functions automatically including the USTRB generation, the parity signal generation, the parity error checking and the read latch clock signal. A more complete discussion of WOLF system operation is given in chapter 6.

TABLE 1

Data line to UBUS correspondences during an ICBM write operation.

```
DATA 23   --------------------------- (Soft parity)
DATA 22   --------------------------- URD
DATA 21   --------------------------- UXQT
DATA 20   --------------------------- USA2
DATA 19   --------------------------- USA1
DATA 18   --------------------------- USA0
DATA 17   --------------------------- UADR1
DATA 16   --------------------------- UADR0
DATA 15-0 --------------------------- UDAT15-0
```

TABLE 2

UBUS to data line correspondence during an ICBM read operation.

```
(EWE BUSY) -------------------------- DATA 23
(READ parity error) ----------------- DATA 22
CERR26 ------------------------------ DATA 21
CERR40 ------------------------------ DATA 20
URPAR ------------------------------- DATA 19
(WRITE parity error) ---------------- DATA 18
URS1 -------------------------------- DATA 17
URS0 -------------------------------- DATA 16
UDAT15-0 ---------------------------- DATA 15-0
```

Note: Signals lines in parentheses are locally generated signals on the Wolf card or otherwise not directly connected to the UBUS.

We will now examine the WOLF system design in detail. For discussion's sake, the WOLF system is divided up into a number of subsystems. For the rest of this chapter each subsystem's function will be described. Throughout this discussion reference will be made to the WOLF schematic diagrams found in appendix A.

Decode Subsystem.

The Decode subsystem is responsible for taking the control lines from the ICBM interface and producing read/write signals for the input and output latches of the two EWE system. It must accomplish this task quickly due to the high speed the ICBM expects to read or write (see appendix D). To accomplish this speed the decode ignores address lines A2-A5, thus causing redundancy in the WOLF memory map. There are only four truly different memory locations on the WOLF card, but each one can be selected by 16 different addresses by changing the value of these higher order bits. Since the decoder ignores them they have the same effects. The decoder allows simultaneous write operations to both EWE systems. By sending a write command with address 0 both sets of output latches will accept data. This feature can speed up operations if both EWE systems require identical setup parameters.

The Decode system also includes a disconnection protection circuit. If the ICBM and WOLF card become disconnected, the pullup resistors at the inputs of the decode system will send the system into a non-operating mode; because to produce action on the board WR\, RD\ or PER_SEL\ would need to be low.

Output Latch Subsystem.

The output latches, found on page 3 of the WOLF schematics, hold data from the ICBM. When EWE WR, generated from the decode subsystem, goes high, the data is clocked into these latches. They are all high speed 74ALS373 latches and are commensurate with the speed requirements found for a DSP write in appendix D. The outputs of these latches are always enabled and the data proceeds directly to the TTL to ECL converter subsystem, and to the parity generation subsystem.

Parity Generation Subsystem.

Parity error checking is used bi-directionally on the UBUS. A parity signal is generated on the Wolf card and sent out on the UBUS line UWPAR. This signal is then compared with the rest of the UBUS by a parity checker and, if the answer is inconsistent, the EWE generates an error signal. The EWE also generates a parity signal itself and sends it down to the Wolf on the URPAR line. There it is compared with the rest of the UBUS and generates an error signal if the information is inconsistent.

Parity generation on the WOLF card can be done in two ways, determined by a mask signal from the ICBM. There is the hard parity generated by U16, U15, and U14 74ALS280 chips, and the soft parity generated by software from the ICBM sent out on data23 with any write operation. U17 determines, using the EWE parity select signal, whether hard parity or soft parity will be utilized. The parity signal is the

sent to the TTL to ECL subsystem to be converted and sent along the UBUS.

Hard parity checks the UBUS only, but it does its job automatically with no need for program intervention. Soft parity is used to check the signal all the way from the VAX host system to the EWE. Software instructions generated on the VAX and handed down though the FASTBUS controller, FASTBUS, and the ICBM without change will be sent out over the UBUS and if a mistake happened anywhere from the VAX to the EWE an error signal will be generated.

In actual use the parity system is likely to be masked in routine use and utilized in a off line diagnostic role only. It was found during the first run that the UBUS was stable after initial checkout and that any communication errors were of the catastrophic type, such as the removal of a UBUS cable which is easily detected. Another reason for not using parity checking online is the time a parity error interrupt handling routine would use. Such a program would slow down the data throughput speed.

TTL to ECL Subsystem.

The TTL to ECL Subsystem converts data from the output latches and other signals from TTL logic levels to tristate ECL logic levels. The subsystem contains U18 through U31. U18, U20, U22, U24, U26, U28 and U30 are MC10124 quad TTL to ECL converters. U19, U21, U23, U25, U27, U29, and U31 are MC10192 quad ECL tristate drivers. The output of U31, U29, and U27 MC10192 are set permanently asserted; these lines are onedirectional UBUS control lines and need not be tristate. The output

of the rest of the MC10192s are controlled by URD. URD is a signal in the data corresponding to DATA22 which controls the direction of the UBUS. If URD is high then the system is reading data from the EWE and the data from the output Latches should not go to the UBUS. If URD is low then the system is writing to the EWE and the output data should be connected to the UBUS. Refer to Wolf schematics page 3 for a detailed diagram of this subsystem.

Timing Subsystem.

The timing subsystem is responsible for the generation of a number of signals. WOLF schematics page 2 shows its circuits and figure 3 is a timing diagram, both of which are referred to in the discussion below.

Once the data has been put on the UBUS by the TTL to ECL subsystem one must send a signal to the EWE system to clock in the data and/or instruction. The signal is called USTRB for EWE strobe. Because the cable length between the WOLF and the EWE is approximately 200ft and the existence of logic delays in the EWE itself one must delay the USTRB signal by some 500 to 1000ns relative to the data. This delay is not constant, however. When ever the data direction is reversed from going out to the EWE to coming back from it or visa versa the USTRB delay must be increased by approximately 200ns to allow for data line settling. The WOLF schematic page 2 shows the circuit to accomplish this. The URD TTL signal is compared to the latched value in the flip flop U8. If the values are the same, then the USTRB delay starting with a EWE write signal from the decode subsystem will be the delay from the

one shot in U4a plus the delay from U4b. If the URD TTL signal differs from that in the U8 flip flop then the USTRB delay will be U4a plus the delay in U5a. The width of the USTRB signal will be determined by U5b. All these one shot delays are generated by 74LS123 chips and are variable by the pots on the front panel of the Wolf card. The flip flop in U8 is clocked with the URD data by the USTRB signal to allow a fresh comparison with the next instruction.

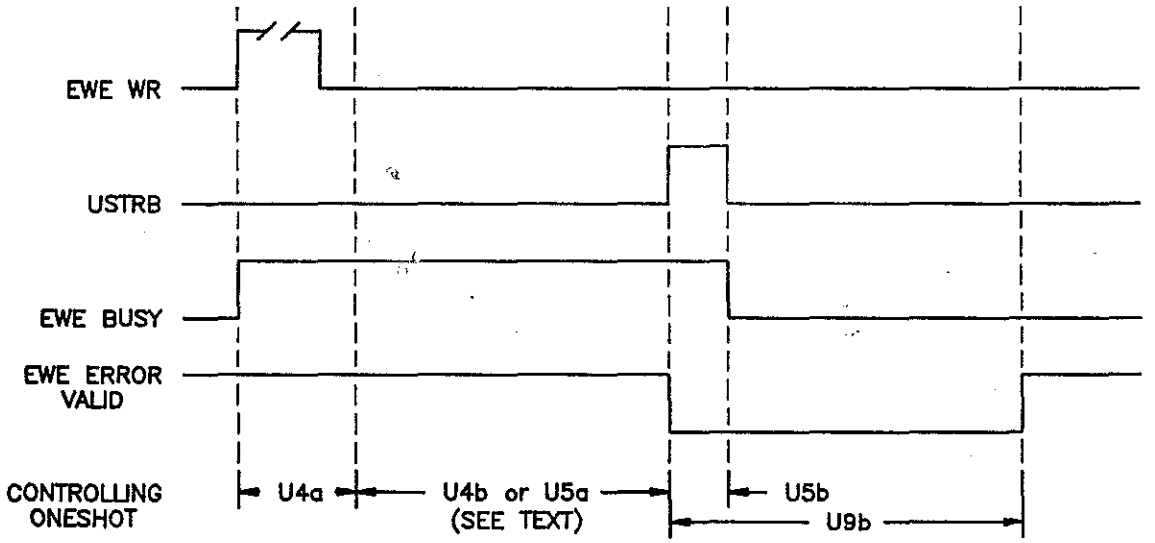
The Timing subsystem is also responsible for producing the RD CLK signal, for read clock. This signal tells the input latch subsystem when to latch in the data from the UBUS. As the WOLF schematics show, this signal is triggered by a high URD and is timed from the end of USTRB. The time delay in U9a delays the clock signal from the time the EWE receives the read instruction to when the data is produced by the EWE and travels down the 200ft cable to the WOLF and settles. This is set by a variable delay just like the USTRB delays.

The timing subsystem produces the EWE error valid signal. This is the signal that indicates enough time has passed from the USTRB signal so that the EWE error signal from the EWE is valid. This signal is timed from the USTRB signal by the one shot 9b. A low to high transition indicates a valid signal on the UBUS.

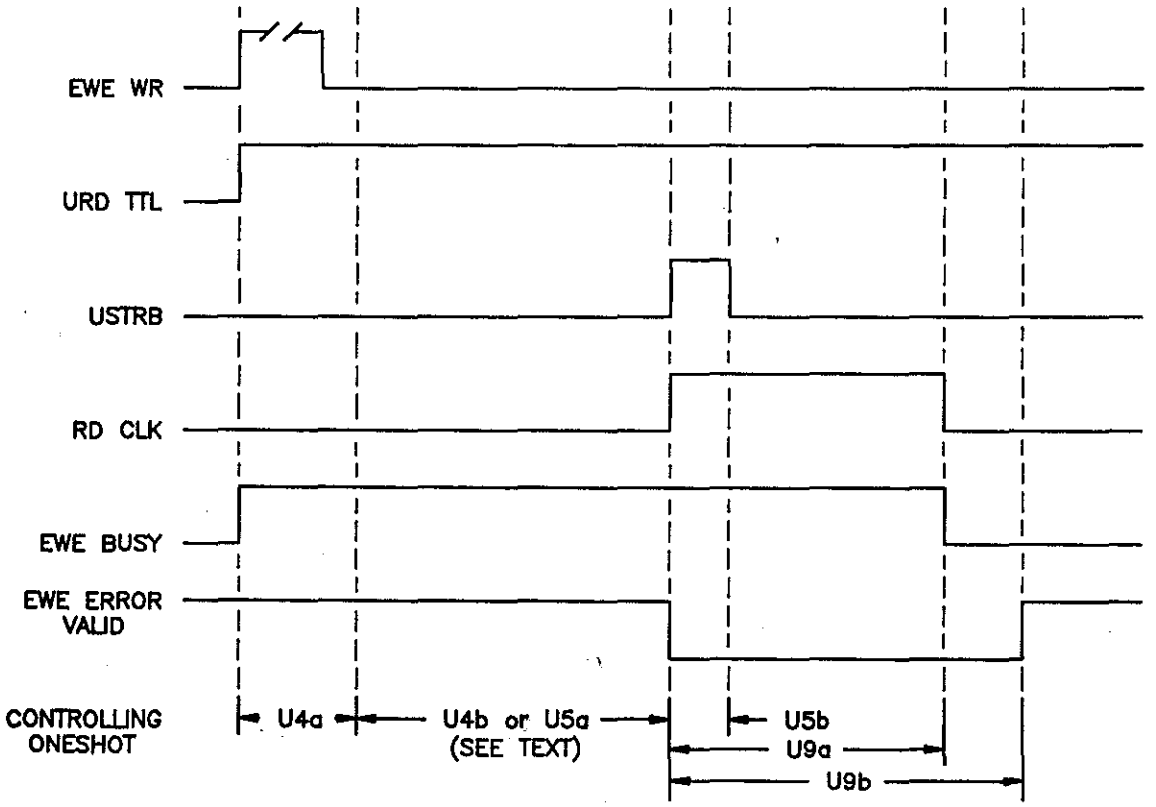
The timing subsystem is also responsible for generating the EWE BUSY signal. If the USTRB is being timed or we are waiting for the RD CLK to time out, the EWE is busy and no instructions should be sent to the Wolf card. EWE Busy is high under those situations.

FIGURE 3

WOLF TIMING WRITE ONLY



WOLF TIMING WRITE/READ



ECL to TTL Subsystem.

The ECL to TTL subsystem is composed of U32, U33, U34, U35, U36, and U37. These are MC10125 quad ECL to TTL chips that take the differential ECL signal and generate a TTL signal. These signals are always enabled with the RD CLK signal designating when the data is valid. U37, which handles the only non differential UBUS signals (CERR26 and CERR40) utilizes the bias output from itself to accompany the input to this chip.

Input Latch Subsystem.

The input latch system accepts input from the ECL to TTL converters when the RD CLK signal states they are valid. Note that U44 is not a latch but only a buffer so that the its inputs can be accessed at any time buy the ICBM. The output of the subsystem is controlled by the EWE RD\ signal generated by the decode subsystem. when EWE RD\ goes low the data in the latches U42 and U43 and the value of the inputs of U44 are dumped to the data lines.

Parity Check Subsystem.

The parity Check subsystem compares the URPAR line with the rest of the UBUS and generates an error signal if the results are inconsistent with each other. This signal is valid only when the data bus is reading information from the EWE to the WOLF card and so it is

therefor clocked into flip flop U45a when RD CLK goes low. The signal ends up at U44 where it can be examined and any time by the DSP.

The parity check subsystem also handles the UBUS parity error signal, this being clocked into flip flop U45b when the EWE error valid signal goes from low to high. The signal ends up at U44 where it can be examined by the DSP at any time. The result is also combined with the local parity error signal from the parity checker and passed along to the interrupt subsystem.

UBUS Termination Subsystem.

In order to function properly the MC10192 in the TTL to ECL converter subsystem must be properly terminated. This is accomplished by having all the MC10192 outputs tied to ECL midrange, about -0.7VDC, through a 56 ohm resistor. The circuity to generate a stable -0.7VDC and the resistors themselves form the UBUS termination subsystem and are depicted in the WOLF schematic diagram on pages 5 and 6.

Front Panel Display Subsystem.

The front panel display subsystem consists of some of the internal WOLF signals driving LED's on the front panel. This lets the operator know if the WOLF card is receiving valid signals from the ICBM and EWEs. The power status is also indicated.

Interrupt Subsystem.

The interrupt subsystem is found on page 1 of the Wolf schematics. The one shots are used to invert the signal if necessary and to limit the duration the the signals to 50ns so as not to cause repetitive interrupts. Table 3 gives a list of signals and corresponding interrupt lines. When any of the conditions are met and that interrupt line is enabled on the ICBM, the DSP will be interrupted. Which signal caused the interrupt can be determined on the ICBM by reading a particular address corresponding to the interrupt controller on the ICBM.

TABLE 3

Interrupt signals

INT0 -- EWE0 DONE, When URS1 from EWE0 goes high this indicates EWE0 has accomplished the last instruction given it and INT0 is activated.

INT1 -- EWE0 DATA, When URS0 from EWE0 goes high this indicates that its analog signal is above threshold and INT1 is activated.

INT2 -- EWE0 ERROR, When EWE0 or the WOLF card identifies that a parity error has occurred during communications INT2 is activated.

INT3 -- EWE0 BUSY, When the WOLF card is writing to or reading from EWE0 and should not be interrupted the EWE0 signal goes high. When it goes low again INT3 is activated indicating that EWE0 is ready for another operation.

INT4 -- EWE1 DONE.

INT5 -- EWE1 DATA.

INT6 -- EWE1 ERROR.

INT7 -- EWE1 BUSY.

CHAPTER 6 - ICBM-WOLF OPERATIONS

We have already discussed the read and write operations to EWE registers via the UBUS by the ICBM - WOLF combination but a number of the subsystems have program controls that require explanation. In this section we shall take the programmers point of view and examine the total interface between the DSP and the EWE. We shall discuss all the registers available to the DSP for these operations and how they might fit into a programming scheme. Some of these registers are resident on the ICBM and others on the WOLF. Recall that the WOLF system connects the ICBM to two UBUS cable systems. Number used in register names refer to which EWE system one is communicating with. .

Let us now examine the registers available to the DSP programmer to facilitate communication with the EWE's.

ewe0wr.

This is the EWE0 write register. It is connected to the UBUS to allow data to be written into a EWE0 register. Its physical location in DSP memory is Y:FFC1 and it has the following fields:

```
bit(s):   23      22    21     20-18     17-16     15-0
          +--------+-----+------+--------+---------+----------+
          | PARITY | URD | UXQT | USA2-0 | UADR1-0 | DATA15-0 |
          +--------+-----+------+--------+---------+----------+
           msb                                          lsb
```

IF the system is set up for software parity generation (see slave_mask) the PARITY bit is sent out to the EWE as the parity checksum of the rest of the word. When the URD bit is low, the data in bits 15-0 is

written to the EWE addressed by UADR1-0, register USA2-0. If the URD bit is high the WOLF card will read into its latches the data in EWE addressed by UADR1-0, register USA2-0.

ewe1wr

This is the EWE1 write register. It behaves exactly like ewe0wr but it access the EWE1 system. Its physical address is Y:FFC2.

ewe01wr

This register combines both ewe1wr and ewe0wr into a single address. Information written to this address is sent out to EWE0 and EWE1. Its physical address is Y:FFC0.

ewe0rd

This is the EWE0 read register. It has physical address Y:FFC0 and has the following fields.

```
bit(s):   23         22    21-20        19     18-16        15-0
         +----------+-----+-----------+-------+---------+----------+
         | EWE BUSY | LPE | CERR26,40 | URPAR | URS2-0  | DATA15-0 |
         +----------+-----+-----------+-------+---------+----------+
           msb                                                  lsb
```

Bits 23 through 16 are continuously updated to indicate the status of EWE0. If EWE BUSY is high, the WOLF system is reading or writing to EWE0 and no operation should be preformed until it goes low. LPE is the local parity error indicator. If this bit is high an error has occurred

in data being transmitted from the EWE to the ICBM - WOLF system. CERR26 and CERR40 indicates if the UBUS is attached to the WOLF card, if either CERR26 or CERR40 go high the UBUS is not connected. URPAR is the the raw parity signal from the EWE used to generate the LPE bit. URS2-0 are status bits from the EWE discussed earlier. URS2 is the parity error indicator, URS1 is the done indicator and URS0 is the data indicator. The data in bits 15-0 is the result of the last read operation written to ewe0wr.

ewe1rd

This register at location Y:FFC1 is identical to ewe0rd except it is for the EWE1 system.

slave_mask

This register controls the choice of parity controls of the EWE systems. It is resident on the ICBM board, has the physical address Y:FF04 and the following fields:

```
bit(s):   23-2        1         2
          +---------+---------+---------+
          |  unused | EWE1 PS | EWE0 PS |
          +---------+---------+---------+
           msb                       lsb
```

EWE# PS stands for parity select. If the bit is high that system will utilize hardware generated parity to send to its EWE, otherwise it will

send the parity bit from the ewe#wr register. The merits of these systems are discussed under the parity generation subsystem in chapter 4.

int_mask

This ICBM resident register controls which interrupt signals, generated by the WOLF card, actually cause an interrupt of the DSP. Located at address Y:FF03, it has the following form:

```
bit(s):  23-8     7     6     5     4     3     2     1     0
         +--------+-----+-----+-----+-----+-----+-----+-----+-----+
         | unused | BY1 | ER1 | DT1 | DN1 | BY0 | ER1 | DT1 | DN1 |
         +--------+-----+-----+-----+-----+-----+-----+-----+-----+
          msb                                                 lsb
```

If a bit is set, the corresponding interrupt will indeed cause an interrupt to the DSP. BY# corresponds to EWE# BUSY, ER# to EWE# ERROR, DT# to EWE# DATA and DN# to EWE# DATA.

int_rd

This register reads the status of the interrupt register. It allows the program to know which signal caused the interrupt that may have just happened. Found on the ICBM card with address Y:FF07 it has the same form as int_mask but is read only and its fields correspond to the interrupts signals.

mod_cntrl

This register controls many operations on the ICBM. Only some of these have application to EWE communications. Located at address Y:FF07 it is a write only address used to reset or clear other registers. It has the following fields:

```
bit(s): 28-13   12        11            10       9-8      7-0
        +--------+----+-------------+-----------+----+-------------+
        | unused | NA | SLAVE M CLR | INT M CLR | NA | INT CLR 7-0 |
        +--------+----+-------------+-----------+----+-------------+
         msb                                                    lsb
```

Bits labeled NA are used for other ICBM subsystems. Writing a 1 to one or more of these bits will cause action to that bit's subsystem only. A 1 in bit 11 will clear the slave mask register (see slave_mask above.) A 1 to bit 10 will clear int_mask and a 1 to bits 7-0 will clear the interrupt corresponding to the bit number.

Operation of the ICBM - WOLF combination was designed to be compatible to a certain extent with the MX. The instruction sets for the two computers are radically different but if one follows the methodology of writing EWE programs instead of MX or DSP programs the transition will be relatively easy. During the first run the MX was programmed to feed out to the EWEs a list of 24 bit words corresponding to instructions (hence a EWE program) and to watch the UBUS for return data. The DSP could be programmed for the very same task and the WOLF's write latches have the exact same format as the MX UM register so that

the same EWE programs could be utilized. This is a very beneficial feature when one takes into account that during the second run a mixed system of MXs and ICBMs will probably be utilized.

CHAPTER 7 - TESTING PROCEDURES AND RESULTS

To facilitate testing of the WOLF system while the ICBM system was under development an ICBM simulator was developed utilizing a Motorola 56000 Application Development System.[4] The ADS card contains a Motorola 56000 chip, memory, an interface connection to the 56000 and a controlling host connector. The host connector is attached to an IBM AT/PC computer to allow programming of the ADS. A transition circuit to make the ADS card look like the ICBM card was produced at the Rochester Cyclotron lab by copying the relative circuity from the ICBM schematics onto a FASTBUS card. The simulator is designed to slip into the FASTBUS crate in replacement of a true ICBM card and is then attached to the ADS card via the front panel. The simulator card is attached to the WOLF card in the exact same manner as the actual ICBM card by the two 34 pin ribbon connectors on the from panels. Using the IBM AT/PC to control the ADS it is possible to test the WOLF card with the exact same software the ICBM card would be using.

Another testing procedure is a manual interface controller called WOLFTEST. This system hooks into a WOLF card in the same manner as an ICBM but has switches to for each control line in the interface and a bi-directional data switch bank with LEDs to read the bus. The schematic diagram in appendix B shows the circuity. It was designed and built in Rochester by the author and staff of the Cyclotron Lab.

---

4. DSP56000ADS Application Development System Reference Manual, (Motorola, Inc. 1987.) January 1988.

The assembly of the prototype WOLF card proceeded by the initially testing each subsystem using the WOLFTEST switch box. Problems found during this stage were layout errors but no logic design flaws. The most serious of these was a short between the +5.0 and -5.2 volt power supplies. After correcting these problems the WOLF card was connected to a EWE and using WOLFTEST successful writes and reads were accomplished to a EWE over a one hundred foot UBUS.

Testing now proceeded using the ICBM simulator. With the timing delays set to maximum on the WOLF card, and with the DSP using the maximum allowable wait states during memory operations, the ICBM simulator and DSP were connected to the WOLF. After verifying that it was possible to write to and read from the EWE under program control, a problem arose with the EWE BUSY signal. The U7 flip flop indicating a read operation under way was consistently being set even when no read operation was performed. Thus the EWE BUSY was constantly high. Normally a program would look at the status of EWE BUSY prior to the next operation on the WOLF card and wait for it to become low before executing. As long as this problem persisted, successful read-writes by the ICBM simulator had to depend on internal program delays to insure the WOLF had completed its task. The problem was first attributed to noise on the URD TTL line, but it was later discovered that the EWE WR signal was arriving before URD TTL settled, causing U7's preset to be activated. A design change in the WOLF timing subsystem corrected the problem by looking at the start of the USTRB pulse for U7's preset signal. See WOLF drawing #2 for details.

With a full working system it was time to see how fast operations on the UBUS could be done. Another hundred feet of cable and

an OPTO-COUPLER board was included in the UBUS line to simulate actual operating conditions at Fermilab. The OPTO-COUPLER board provides a groundless transition into the Faraday cage for the UBUS. First the DSP's internal clock was brought to full speed with successful EWE operations. The time required for Read and Write operations between the WOLF and ICBM simulator was brought down successfully to one wait state; but full speed operations (no wait state) did not work. It is believed that this is due to the connections between the simulator and the WOLF card being logically too long. It is hoped that during true ICBM operations full speed will be possible.

Next, the lower bound of the timing delays were found by trial. The minimum time required to write data to a EWE register from the DSP is approximately 900ns and the minimum time required for the DSP to read data from a EWE register is approximately 1200ns.

The ICBM simulator - WOLF system was then tested with an error checking program to see if any errors occurred over time. The unit was run for sixteen consecutive hours and no errors occurred during that time. The testing software is shown in Appendix E.

BIBLIOGRAPHY

1. The E706 Collaboration, "Preliminary Results from Fermilab Experiment E-706 - A Study of Direct Photon Production in Hadronic Collisions," Submitted to XXIV International Conference on High Energy Physics, (June 1988).

2. Motorola, Inc. DSP56000ADS Aplication Development System Reference Manual, (June 1988).

3. K. Turner and S. Littleton, "MX, EWE-II and UNIBOX Registers, PIN 88," E706 Notes, (October 1986).

4. D. Skow, "FEED for RABBIT," E706 Notes, (June 1986).

APPENDIX A

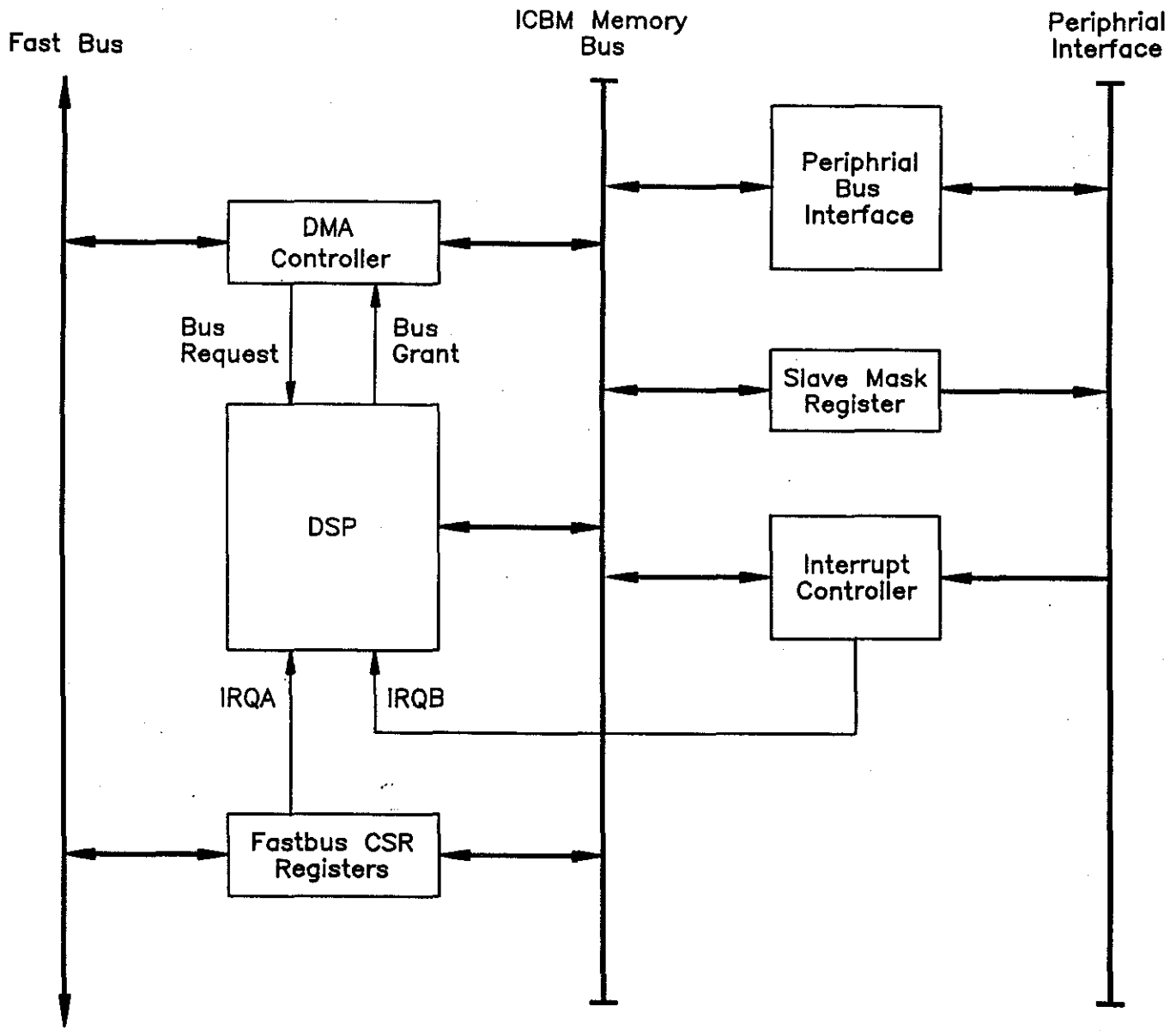
See Cover Pocket

APPENDEX B

See Cover Pocket

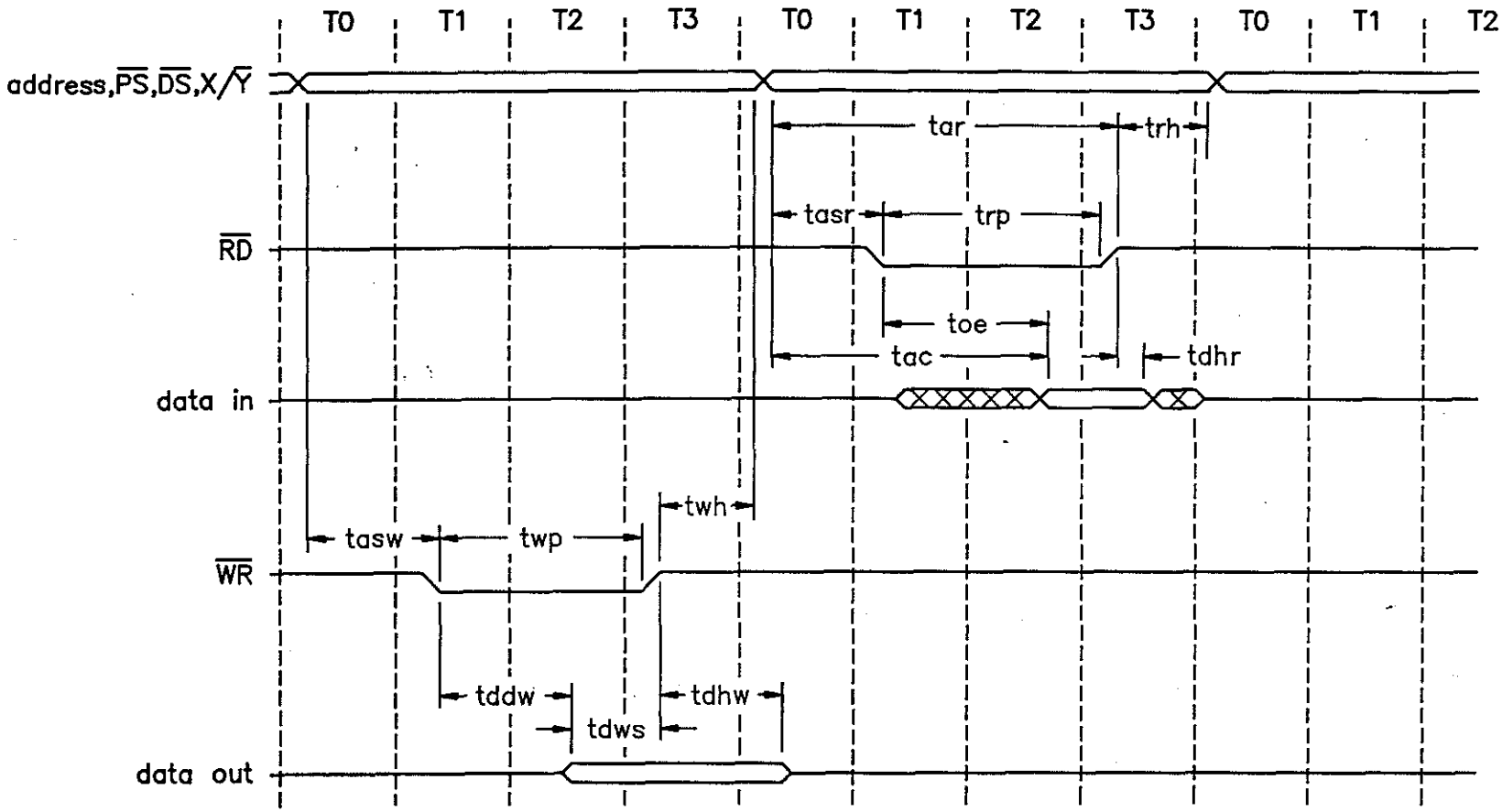
APPENDIX C

ICBM BLOCK DIAGRAM



APPENDIX D

MOTOROLA DSP56000 BUS TIMING



for Icyc = 97.5ns (f=20.5Mhz)

T = Icyc/4

No wait state memory

	MIN(ns)	MAX(ns)	MIN(ns)	MAX(ns)
tasw	T−9	T	15	24
twp	2T−9		40	
twh	T−12		12	
tddw		T+10		34
tdhw	T−9	T+7	15	31
tdws	T−4		20	
trh	T−9		15	
tar	3T−8		65	
tdhr	0		0	
trp	2T−9		40	
tac		3T−18		55
tasr	T−9	T	15	24
toe	2T−14		34	

APPENDEX E

```
;
; wolf3run.asm - WOLF-EWE test program for dsp56001
; Charles R. Benson
; 3 May 1989
;
;
start   equ $0100
ewe0rd  equ $fff0
ewe0wr  equ $fff0
;
; i/o word masks
;
uadr    equ $010000
cmd     equ $000000
stat    equ $040000
chadr   equ $080000
ped     equ $0c0000
thrsh   equ $100000
adc     equ $140000
read    equ $180000
write   equ $1c0000
urd     equ $400000
xqt     equ $200000
;
; routine constants
;
max     equ $FFFF
;
; begin compilation at start
;
        org     p:start
;
; set r1 and r2 for in/out address of wolf
;
        move    #ewe0wr,r1
        move    #ewe0rd,r2
;
; set x0 to ped register write
; set x1 to ped register read
;
        move    #uadr+ped,x0
        move    #uadr+ped+urd,x1
;
; clear wolf system with a read command
;
        move x1,y:(r1)
;
; set up b1 for master counter
;
loop1   move    #$0,b1
```

```
;
; compose write word in a1, and send it to ewe when wolf ready
;
loop2   move    b1,a1
        add     x0,a
uw0     jset    #23,y:(r2),uw0
        move    x0,y:(r1)
uw1     jset    #23,y:(r2),uw1
        move    a1,y:(r1)
;
; compose read word, send it to wolf and read responce
; when wolf ready
;
uw2     jset    #23,y:(r2),uw2
        move    x1,y:(r1)
uw3     jset    #23,y:(r2),uw3
        move    y:(r2),a1
;
; trim off most significant eight bits
;
        move    #$00ffff,y1
        and     y1,a
;
; compare the results with origional number sent there
;
        cmp     a,b
        jeq     cont
;
; spot for monitor breakpoint
;
        nop
        nop
        nop
        nop
cont
;
; bump master count
;
        move    #1,a1
        add     a,b
;
; compare master count with max and restart if nessesary
;
        move    #max+1,a1
        cmp     a,b
        jne     loop2
        jmp     loop1
;
exit    nop
        nop
        nop
;
        end
;
;
;
```