



Fermi National Accelerator Laboratory

TM-1540

**FSMI - VME to Lecroy 1821 Interface Routines
Preliminary Version**

Dean Alleva
Computing Department - Evaluation Group
Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois

August 1988



Operated by Universities Research Association Inc. under contract with the United States Department of Energy

FSMI - VME to Lecroy 1821 Interface Routines
Preliminary Version
August 1988

Dean Alleva
Computing Dept.- Evaluation Group
Fermilab

1.0 Introduction

This document describes the VME/Lecroy 1821 Interface routines (FSMI). These routines were designed to meet two needs. First, the routines enable programs written in PILS running on a MVME 101 under Valet-Plus to control a Lecroy 1821 FASTBUS interface from VME [1], [2]. Secondly, the routines provide a high level language version of the FASTBUS standard routines for the 1821 which can easily be translated into other high level languages (like C). The routines fall into two general types, control and transaction. The control routines (also called the 1821 primitive routines) work directly with the 1821 interface. These routines set up and monitor operations between VME and the 1821. The control routines are usually used indirectly by the programmer through the transaction routines. The transaction routines, such as FSMI_CYCLE_ARBITRATE, use the control routines to carry out complete functions on FASTBUS.

To facilitate access to the 1821 registers from VME, a Super-VIOR DMA board was used as a set of I/O register tied to the 1821/DEC interface [3], [4]. The DMA controller was not used in the preliminary version but will be used in future versions.

The routines are written in PILS, a high-level language similar to BASIC and Pascal. PILS, a relatively simple language, is powerful and fast enough for most applications [1].

This document describes a preliminary version of the FSMI routines. All necessary control routines have been implemented. The FASTBUS standard routine library is functional up to the FASTBUS primitive routine level. As with any preliminary software, further testing and coding needs to be done before this software is truly useful.

This document is divided into four sections, the first being the introduction. The remaining sections detail the FSMI routines, suggested changes to 1821 microcode, and a reference section. It is assumed that the reader is familiar with VME, FASTBUS, and has some knowledge of the Lecroy 1821 [2]. A copy of the FSMI software is available on BitNet at Fermilab as

```
"FNAL::USR$ROOT:[ALLEVA.PUBLIC]FSMI.DEF"  
"FNAL::USR$ROOT:[ALLEVA.PUBLIC]FSMI.PRM"  
"FNAL::USR$ROOT:[ALLEVA.PUBLIC]FSMI.LIB"
```

2.0 The Routines

The FSMI routines are divided among three files. "FSMI.DEF" defines constants used by the FSMI routines. "FSMI.PRM" contains the control routines for the 1821. "FSMI.LIB" contains the FASTBUS standard routine library. With the preliminary version, only the FASTBUS primitive routines have been implemented. The more complex transaction routines can be built from these primitives.

The preliminary routines have been built for easy expansion and change. The control routines contain a set of microcode interface routines which isolate the microcode calling sequence from the rest of the the FSMI code. Microcode changes can be accommodate by changing the microcode interface routines or by adding new routines.

The preliminary FSMI routines do not use the Super-VIOR DMA features [3]. Data is transferred to or from the 1821 through a set of routines call FSMI_P_GET_DATA and FSMI_P_LOAD_DATA. The data is moved by the CPU from the 1821 to VME memory. To use the Super-VIOR DMA features, these routines should be rewritten. Note, however, that the 1821/DEC interface only allows DMA during reads.

2.1 Routine Description

1) FSMI_CYCLE_ARBITRATE

Description: Does an arbitration cycle, holding the bus when control is gained.

2) FSMI_CYCLE_RELEASE_BUS

Description: Releases the bus by lowering GK.

3) FSMI_CYCLE_PA_DAT (pradd)

Description: Does a primary address cycle to data space.

Parameters:

pradd (INT32, input): The primary address.

4) FSMI_CYCLE_PA_CSR (pradd)

Description: Does a primary address cycle to csr space.

Parameters:

pradd (INT32, input): The primary address.

5) FSMI_CYCLE_PA_DAT_MULT (pradd)

Description: Does a broadcast primary address cycle to data space.

Parameters:

pradd (INT32, input): The primary address.

6) FSMI_CYCLE_PA_CSR_MULT (pradd)

Description: Does a broadcast primary address cycle to csr space.

Parameters:

pradd (INT32, input): The primary address.

7) FSMI_CYCLE_READ_SA (rword)

Description: Does a secondary address cycle read. Bus mastership and primary address cycles must be complete before using this routine.

Parameters:

rword (INT32, output): Returned word from read.

8) FSMI_CYCLE_WRITE_SA (wword)

Description: Does a secondary address cycle write. Bus mastership and primary address cycles must be complete before using this routine.

Parameters:

wword (INT32, input): Word to be written to NTA.

9) FSMI_CYCLE_READ_WORD (rword)

Description: Does a single word read cycle. Bus mastership and primary address cycles must be completed before using this routine.

Parameters:

rword (INT32, output): Word transferred.

10) FSMI_CYCLE_WRITE_WORD (wword)

Description: Does a single word write cycle. Bus mastership and primary address cycles must be completed before using this routine.

Parameters:

wword (INT32, output): Word to transfer.

11) FSMI_CYCLE_READ_BLOCK

Description: Does a block read cycle. Bus mastership and primary address cycles must be completed before using this routine. Reads until SS=1 (end of block).

13) FSMI_INITIALIZE (sv_add, retry_cnt, arb_vec, mod_num)

Description: Initialize the FSMI routines. This should be the first FSMI library call in a program.

Parameters:

sv_add (INT32, input): VME address of Super-VIOR.

retry_cnt (INT32, input): Retry count.

arb_vec (INT32, input): Arbitration vector.

mod_num (INT32, input): 1821 module select number.

14) FSMI_RESET

Description: Resets the 1821 interface.

15) FSMI_DOWNLOAD (menu_number)

Description: Downloads a menu memory into the sequencer memory.

Parameters:

menu_number (INT32, input): Menu memory number (0-7).

3.0 Suggested 1821 Changes

After working with the 1821 for several weeks it had become obvious that some of the microcode needed to be changed. Below is a list of these needed changes.

- 1) Microcode routines should be added for clearing the bus but leaving GK(u), putting AS(d) and waiting for AK(d), and clearing the AD lines.
- 2) Primary address routines should support 32-bit addressing. At present only the broadcast primary address routines support 32 bit addressing.
- 3) Full support for the WT line should be included.
- 4) Support for FASTBUS block writes should be included. This is necessary if the full Standard Routine library is to be implemented. Better support for DMA writes from the host should also be included in hardware.
- 5) The standard routine block transfers include setting the size of the block to be transferred. At present, the microcode block read transfers until a end of block status response. A more general purpose implementation of block transfers is needed (if this is possible with the 1821).

4.0 References

- [1] Berners-Lee, T. et al. The VALET-Plus, a VMEbus Microcomputer for Physics Applications. Fifth conference on Real Time Computer Applications in Nuclear, Particle and Plasma Physics - San Fransisco, May 1987.
- [2] Lecroy 1821 User's Manual, Revised March 1987
- [3] Super-VIOR, VMEbus Dual 16-bit Input/Output Register with Full DMA, hardware description, Opifex AB publication (Version 1.1)
- [4] Lecroy 1821/DEC Manual, November 1984