STANDARD SOFTWARE Routines FOR FASTBUS
R. Pordes, S. Gannon, L. Taff (1)
Fermi National Accelerator Laboratory
Batavia, Illinois 60510

Abstract

FASTBUS is a standardized modular bus system for data acquisition, data processing, and hardware control, designed for high energy physics applications. This paper introduces the proposed software standard routines for FASTBUS(2).

Introduction

FASTBUS is a standardized modular data bus system designed for high energy physics applications. It has been developed by the FASTBUS System Design Group of the U.S. NIM Committee, with representatives from many U.S. Laboratories. The work is supported by the Department of Energy(3). Experience with the standard electronic system CAMAC has shown the desirability of establishing software standards for such systems(4) which can be used in many applications.

This paper introduces the standard routines proposed for the operation of FASTBUS. It also gives examples of the defined standard routines. The aim of proposing such a standard is to make programs which access FASTBUS devices easier to write, to understand and to transport. As far as possible the names, calling sequences and parameters are defined to be independent of the type of computer and computer language that will be used. However the standard definitions include room for features that are specific to a particular implementation. Each application of the standard routines need only implement a subset of the full definition.

A preliminary proposal for Standard Portran subroutines for FASTBUS was written at Fermilab in July 1980 (5). The development of a more general software standard to access FASTBUS devices was adopted as one of the goals of the FASTBUS Software Working Group. Details of the standard are given in the FASTBUS document FSDG085, available from the Computing Department at Fermilab (2). This document was adopted in principal by the FASTBUS Software Working Group in January 1981. Development is still in progress, but it is intended that any software now being written to access FASTBUS devices will be able to make use of the already defined standards.

Fermilab is implementing the standard on the PDP-11 using RT-11 and RSX-11M operating systems. The interface from the PDP-11 to FASTBUS is through the Unibus Processor Interface currently being built at Fermilab(6).

Format of Standard Routines.

The standard does not assume a particular language for the FASTBUS routines themselves, nor for the programs which call them. The routines are therefore defined in terms of a name, an ordered set of parameters, and a description of the function of the routine. The names have a maximum of six letters. All start with the same letter (F) to make them an easily recognisable set.

The optional details of the FASTBUS operations to be performed are controlled by an array which is available to each standard routine. Options include whether to wait for operation completion, whether to abort the operation if there is a parity error, etc. Option definitions are included which change the length unit from the natural FASTBUS length of 32 bits, to the word length of the computer on which the standard is being implemented.

The typical execution times on FASTBUS are much shorter than those of the controlling processor. Interfaces between computers and FASTBUS will therefore often incorporate list processing devices. Routines to aid in the use of FASTBUS in this manner are included in the standard set. For each routine defined which results in the definition and execution of a FASTBUS operation, there is a parallel routine defined, where the operation is 'compiled' and stored in an internal list. Such lists can be constructed to contain many FASTBUS operations, and then executed as a single sequence at any time.

Examples

1. The following routine is defined to perform a FASTBUS block read from data space - for example data from a memory module. It may be implemented as a subroutine, a function, or decoded by an interpretive program.

   FROB (BUFADD, FBRDR, MAXLEN, CONTROL, STATUS )

   where

   BUFADD
   is the address of the internal buffer into which the incoming data will be written. This buffer must be long enough to contain the maximum amount of data that may be read.

   FBRDR
   is the starting 32 bit FASTBUS address from which the data will be read.

   MAXLEN
   gives the maximum number of data words that may be read by this block read. The units of this value will depend on the control options specified, but the default is in units of 32 bits.

   CONTROL
   is an array which specifies options to be used in the FASTBUS operation to be performed. The options may be selected by calling another routine where the same array is passed as a parameter.

   If parity is to be ignored for this read operation, for example, the routine FPREP (CONTROL) can be called to set the 'Continue on Parity Error' option in the array. If this array is then used in the call
to FBDB, the block read operation will be continued if a parity error occurs.

**STATUS**

is an array to contain the returned status of the FASTBUS operation. The meaning of some of the returned errors is defined in the standard definitions.

2. The following sequence of routines establishes an area to hold compiled lists of FASTBUS operations, compiles and stores a FASTBUS write operation, and then executes this stored list.

Initialize the buffer in which lists will be stored, and declare the buffer area from which the data will be read:

```plaintext
FBADR
```

is the 32 bit FASTBUS address to which the data will be written.

```plaintext
COTRL
```

is the array containing options for the write operation, such as an extended FASTBUS address to be used when establishing the connection to FASTBUS, and/or parity checking as described in the first example.

**STATUS**

is the array to contain the returned status of the FASTBUS operation.

References

1. S.Gannon is currently at the Department of Computer Science, Illinois Institute of Technology, Chicago, Illinois 60640.

L.Taff is currently at Bell Laboratories, Naperville, Illinois 60616.


5. Fermilab 5BN005, S.Gannon and L.Taff, July 1980