

Version 2 of RSXMULTI

By

P. Heinicke, P. Constanta-Fanourakis

Software by:

P. Constanta-Fanourakis, P. Heinicke

E.K. Quigg, D. Berg

Fermi National Accelerator Laboratory

Batavia, IL. 60510

Abstract: MULTI is a general purpose, high speed, high energy physics interface to data acquisition and data investigation system that runs on PDP-11 and VAX architecture. This paper describes the latest version of MULTI, which runs under RSX-11M version 4.1 and supports a modular approach to the separate tasks that interface to it, allowing the same system to be used in single CPU test beam experiments as well as multiple interconnected CPU, large scale experiments. MULTI uses CAMAC (IEE-583) for control and monitoring of an experiment, and is written in FORTRAN-77 and assembler. The design of this version, which simplified the interface between tasks, and eliminated the need for a hard to maintain homegrown I/O system is also discussed.

Introduction

RSXMULTI [1] is a software system used for over six years by various high energy and nuclear physics experiments. MULTI can perform dynamically defined calculations on data, display data in many different forms, including histograms and scatter plots, monitor critical parameters to ensure that they stay within certain predefined limits and perform run and tape control. MULTI obtains data from a data acquisition package or from a tape. The term "MULTI" derives from the many functions available.

Extensive documentation describing the internal parts of the software (e.g. the common blocks), the user interface, the I/O subsystem, the generation procedure, and the way that various experiments have customized MULTI is available.

Developed originally by the Fermilab Caltech HEP group, and enhanced by the Fermilab Computing Department, the software is in the public domain, and is available from NESC (National Energy Software Center).

System Characteristics and Requirements

MULTI was designed with the intent to be as operating system independent as possible, and transportable across a wide variety of computer architectures. MULTI was written in FORTRAN IV but has been converted to FORTRAN-77. MULTI has been successfully ported from XEROX SIGMA computers to PDP-11's under RT(SJ) and RSX, to DECpro 3XX computers running RT(XM) and POS, and to VAX computers running VMS. The main requirements for the computer system are a FORTRAN compiler, a graphics terminal, and hardcopy facilities.

The RSX implementation requires RSX-11M version 3.2 or higher, and FORTRAN-77. It also runs under VAX-11 RSX (compatibility mode). MULTI has not been tried under RSX-11M+, but it obeys the restrictions for a non-privileged task under M+, so in principle should operate properly.

RSXMULTI requires a PDP-11 with at least 248Kb of memory, and FPU hardware.

Operating System Independent Features

MULTI, as implemented under various operating systems, contains a common set of features, such as modular organization, simple user interface language, dynamic definition of histograms and variables, and interface routines, which may be modified by the user to extend the functionality of the core routines.

The system is organized into function and subroutine modules with well defined and specific functionality. When a TRUE/FALSE status can be returned, the module is implemented as a function; when status is not required, or when the status is too complex to be indicated with a single logical variable, the module is implemented as a subroutine. All command processors are written as subroutines.

Commands have the following syntax:

Verb, argument1, ... argument16.

As many as 16 arguments may be specified. A verb refers to a command such as DISPLAY, DEFINE, and so forth. The arguments serve as input parameters to the command, and may take on different meanings for each verb. If an argument is omitted, then a default value is substituted. If a verb is one of MULTI's predefined set of commands, then the appropriate command processor is invoked. If the verb is not predefined, then MULTI searches for a command file named "VERB".MCM. If this file does not exist then the message "COMMAND NOT FOUND" is displayed.

Mnemonic references to variables contained in predefined regions of memory, (e.g. FORTRAN common blocks), may be defined by using the "DEFINE" command. For example, the fifth data word may be defined to be "MASS" and subsequently referred to as "MASS" instead of "IV5" by typing:

```
DEFINE MASS=IV5
```

In this example, IV5 is the fifth array element of the array IVARS in the common block ICOMN.

To perform simple mathematical and logical operations on event data, the EVAL command may be used. This command uses a "BASIC"-like expression syntax to define a list of operations to be applied to each event. The following example calculates the square root of DEDX times ETOT and loads the result into the variable MASS. (Each variable corresponds to a location in a common block defined using the "DEFINE" command.) Then the logical variable YESPI is loaded with the result of the logical "AND" of the two tests MASS > 120.5 and MASS < 158.5

```

>EVAL
  NEXT EXPRESSION?
>10:MASS=SQRT(DEDX*ETOT)
  NEXT EXPRESSION?
>20:YESPI=(MASS>120.5)&(MASS<158.5)
  NEXT EXPRESSION?
>END

```

Online, interactive data examination by various techniques like histograms, scatter plots, and raw data displays, can be conveniently specified using MULTI commands. Histograms and scatter plots can be defined at run time, using the HIST and DISPLAY commands. They can also be defined in a faster but less flexible way by writing FORTRAN subroutines called by EUSERA in MULTI. Histogram and scatter plot displays are continuously updated after every new event, and can be controlled by conditional variables. The following example histograms the variable MASS in the histogram MSPEC, giving it 100 bins, and histogramming the values falling between the range of 120 and 160:

```
>HIST MASS,MSPEC,100,120,160
```

User subroutine calls provide the ability for each user to customize certain parts of the system, without changing the basic structure of MULTI. These routines are implemented as skeleton routines (dummies). All the modifications contained in the user subroutines may be disabled by setting the variable "SYSTEM" to "YES". These subroutines are called when certain commands are invoked; for example the DISPLAY command calls the DIUSER subroutine to do any user defined displays.

Operating System Dependent Features

The current RSX-11M implementation of MULTI has relatively few system dependent attributes. These include the system generation procedures and the input output routines.

Generation and Installation

Generation of RSXMULTI proceeds with a question and answer session controlled by an indirect command file which enables the user to select from various possible configurations. RSXMULTI has two levels of complexity. Level 1.0 of RSXMULTI contains the interface to the standard FERMILAB RSX data acquisition system, RSX-DA. Level 0.5 does not contain this interface, but may be adapted using the user routines to interface to other data acquisition systems. MULTI stores its dynamic variable definitions and its histograms in the "DYNAM" array. This array may be selected during the generation procedure to be either a virtual array (slow, but allowing up to 32KW of histogram space), or an integer array (faster, but limited to about 4KW in size).

Installation of RSXMULTI involves copying of about 20 files from the distribution media, and executing the generation procedure. The complete procedure takes about 90 minutes on an 11/50.

I/O Interface Routines

MULTI has a standard set of I/O routines. This version of MULTI implements the routines in FORTRAN-77, using very few system dependent features.

The RSX-DA Interface

RSXMULTI is a generic data analysis system capable of interfacing to any event oriented data acquisition system. The modular design means that only two small subroutines need to be changed to interface to a different acquisition system. In the case of RTMULTI, this characteristic was often exploited to interface MULTI to different, nonstandard hardware. RSXMULTI is now as easy to interface to other data acquisition systems as RTMULTI.

RSXMULTI level 1.0 can interface to the RSX-DA [2]. It accesses event data by requesting it from the GTEVNT task, or by reading the data from magnetic tape directly. GTEVNT obtains its data either from the data acquisition system, (RSX-DA), or from a data tape. Level 0.5 can only read data from a magnetic tape without using GTEVNT.

GTEVNT uses a locally developed communications driver, "CD" [3], which is capable of sending blocks of data between tasks and performing inter-processor communication using DR-11W links. A set of FORTRAN callable routines, CDPACK, has been developed for use with the CD driver. Using CDPACK, RSXMULTI sends a request packet containing the type of event it requires and receives a buffer containing one or more complete events.

Since RSXMULTI finds its main area of application in high energy physics experiments, it has features to control the beginning and ending of a run. Runs are numbered consecutively, and the run number is an integral part of each event.

Timings and Size Studies

The time it takes MULTI to histogram N events was parametrized as:

$$T = N * (TRL + NH * TH + TA)$$

where:

1. N is the number of events acquired from the data source.(10000)
2. TRL is the time to do the main program loop in RMASTR (the main routine).
3. NH is the number of histograms defined.
4. TH is the time to increment a bin of a histograms.
5. TA is the time required once any histograms are defined. It is present when NH is non-zero.

Using the virtual array implementation:

```

TRL=1.35 msec,
TH=2.05 msec, and
TA=0.01 msec.

```

The same times for the integer array implementation are:

```

TRL=1.35 msec,
TH=0.86 msec, and
TA=0.07 msec.

```

RSXMULTI is extensively overlaid. The task image size is about 26KW with another 4KW taken up by the virtual array mapping window. Users have about 2KW

left for their modifications in the user routines, without reoverlying.

Conclusion

RSXMULTI is a successful implementation of the MULTI system on the RSX-11M operating system. It provides all the features available in other implementations, as well as being easy to generate and install. We anticipate further use of RSXMULTI both inside and outside the high energy physics community.

Acknowledgments

This work was supported in part by DOE contract DE-AC02-76CH03000.

The authors wish to acknowledge the work done by J.F. Bartlett in designing the original version of MULTI, and D.J. Ritchie, E.K. Quigg, T. Lagerlund, and the rest of the Fermilab Computing Department for the development of the original PDP-11 version.

References

- [1] J.F. Bartlett, et. al., "RT/RSX MULTI: Packages for Data Acquisition and Analysis in High-Energy Physics", IEEE Transactions on Nuclear Science, Vol NS-26(4), (August) 1979.
- [2] D.M. Berg, et. al., "A High-Speed CAMAC Data Acquisition System for PDP-11", paper presented to this conference.
- [3] V. White, et. al., "Multi-processor Data Acquisition and Monitoring Systems for Particle Physics", Third Biennial Conference on Real-Time Computer Applications in Nuclear and Particle Physics, May 16-19, 1983.