COLLÈGE DE FRANCE

1530

Laboratoire de Physique Corpusculaire

11, Place Marcelin-Berthelot, 75231 Paris CEDEX 05 - 325 62 11

M.U.M.M.

A Micro-controlled Universal Message Multiplexer

G. FONTAINE, L. GUGLIELMI, J.J. JAEGER and S. SZAFRAN

Collège de France, Paris.

LPC/81-11

Contributed paper to the "Topical Conference on the Application of Microprocessors to High-Energy Physics Experiments", CERN, 4 - 6 May 1981.

M.U.M.M.

A Micro-controlled Universal Message Multiplexer.

G. FONTAINE, L. GUGLIELMI, J.J. JAEGER and S. SZAFRAN

(Laboratoire de Physique Corpusculaire, Collège de France, PARIS).

ABSTRACT

Based on the Motorola 6800, this multiplexer is designed to provide a microprocessor development tool in the specific environment of a high energy physics laboratory. The basic philosophy of this device is to allow communication of a target (prototype) processor with a host computer under control of a human operator. The host can be an experimental on-line computer or any remote machine with a time-sharing network.

It is thus possible to speed up design and debugging of a physics application program by taking advantage of the sophisticated resources usually available in a computer centre (powerful editor, large disk space, source management via "Patchy" etc...). In addition to the classical cross-macroassembler, a loader is available on the host for down-line loading binary code, via the multiplexer, into the prototype memory.

Such a scheme is easily extended to the communication of any host interactive processing program with a data acquisition microprocessor, and provides the latter with a convenient and easily portable extension of its computing power. A typical application of this mode is described in a separate paper.

1. M.U.M.M. ORIGINS.

1.1 Working with microprocessors (μ P.) requires the development of hardware and software as well as the integration of these two components in the final application set up. This procedure is usually carried out with the help of a development system, which leads to the question of the choice of this system :

The simplest monitor dealing with hexadecimal coding, keyboard and display is soon insufficient for any sizable application which requires source code edition, storage and listing, assembly and loading, EPROM recording, and debugging facilities. Hence the real choice is usually between a cross-software running on a time sharing system(or on a dedicated mini) and a highly specialised device allowing purely local processing of the relevant information. This last solution seems to prevail in the industrial world for the following reasons :

- It allows good hardware development and emulation. Integration is thus made easier by progressive replacement of hardware pieces of the emulator by those of the tested prototype.

- It can be bought "off the shelf", and immediatly put into operation. Beeing a small "human-size" system, it is well accepted by the users, who are usually electronics engineers not very familiar with large programs.

- It is pushed forward by μ. P. manufacturers since this choice represents a good occasion for them, to sell additional hardware (large memories, numerous peripherals...).

1.2 In a high energy physics (H.E.P.) laboratory, the solution of buying commercial development tools may raise some financial difficulties since the cost of a single user system is about 200 KFF !

The alternative of building one's own local system already reduces the bill, but one still needs a rather large configuration in order to run resident software, for which a lot of manpower is required, even if the emulation possibilities of the tool are frequently left out. However, the situation in H.E.P. is quite specific, since it is one of the most computerised environments ! One is never far from any computer or at least from one of its terminals, (be it a remote central computer or an experiment-control machine), and the use of its ressources can save about 80% of the hardware cost of a development system. This solution also allows the shift of a large fraction of the software effort from the μ. P. used in local tools to a large computer where high level languages, ready to use products and a lot of experience are available (see discussion below).

1.3 To avoid the difficulties of a purely off-line approach, the target processor is connected to the host-computer where the cross software is running, and an interactive program on the host can control the μ. P. by a dialogue with its monitor or with a specific application program. It is thus possible to execute standard procedures such as down-line loading of programs or to provide some help at the debugging stage (e.g., controled step-by-step execution with disk recording of successive statuses). Beyond the development stage, this link can be used at run time for data exchange and can use the host as a powerfull processing extension of the application processor. However, most time sharing systems limit the number of connections of an interactive program to a single console. This is the reason why we have developed an "intelligent" multiplexer using a M6800 μ.P.[1)2)] to dispatch messages between a host-computer, a target processor, a user video display unit (V D U) and a local back-up memory (e.g. audio cassette). It soon became the key factor of our system, and this Micro-controlled Universal Message Multiplexer (MUMM) has since been used for quite a number of different applications for which it was not initialy designed.

2. PROS AND CONS OF MUMM AS A DEVELOPMENT TOOL.

Beyond the very low hardware investment needed, MUMM gives acces to very powerfull devices and sophisticated utilities :

- Edition by a high-level editor with all its associated facilities (string manipulations, logical conditions etc...)

- Large and cheap disk space to store files with a very high reliability (e.g. total disk space of 2400 Megaoctets(Mo) ; out of which 200 Mo can be used by a single job, and about 10 Mo can be kept permanently for a single group, compared with a floppy of 250 to 800 Ko or even with a mini-floppy three times smaller).

- Possibility of using standard magnetic tapes as disk back-up or extension, or as an exchange medium. The exchange of code or of data can also take place through the rapidly expanding interconnection network between computers (such as CERNET), or between HEP computer centres (such as the CCPN-CERN link).

- Fast line printers using normal paper.

- Source management via a source code maintenance program such as the CERN Patchy[3]. Although unusual in the μ. P. field, this method has proven to be very efficient to deal with multi-version software, or to provide for sequences or patches of code of general interest, ready to be used and assembled in an application program.

- Powerfull cross assemblers[4][5] with all possible facilities such as macro-operations, insertion of external files, conditionnal assembly, etc... These possibilities are enhanced to their maximum by the source preparation via Patchy.
  Up to now we do not have a cross compiler for a high level langage but expect to install one in the near future.

- It is also straightforward to provide debugging help by a direct dialogue between a host program and the application monitor. This facility has not been implemented up to now due to lack of time and of real need to do so.

There are also a number of distinct advantages related to this method of working :

- A majority of local development systems are mono-user, either by design or because multi-user systems are even more expensive. In fact a large fraction of their time is used for source code edition or assembly, and in our case this can just be done on any console of the time sharing system by many users at the same time. The full task can also be made multi-user just by duplicating the MUMM units since their hardware cost is low.

- Portability is also an important parameter for a H.E.P. lab. outside CERN. We have to use our system both at CERN in the experiment and at our home institution, and hence a conventional approach would require two development systems, one for each site. Since host computers are available everywhere, our 2 kg module is easily transported and only requires access to a modem or even just a standard telephone line by the use of an acoustic coupler.

- Finally the system is easily adaptable to different types of microprocessors or even to mini computers. The only requirement beeing that they have a monitor, even a simple one which supports only a few very basic operations. The initial design was oriented towards a M6800 target processor with the Minibug 3 E monitor, and a CDC Cyber 172 host with Intercom and NOS-BE as available in the Paris H.E.P. computer centre (C.C.P.N.). It has since been used also with a DEC PDP 15 as a target machine, or with a NORD 100 or the CERN CDC front-ends as a host partner.

However one has also to say honestly what are the inconveniences of working in this way :

- It offers no real time emulation possibility. This has not been a problem up to now, even for MUMM debugging. Let us hope it will continue to not do so.

- It is dependant on the availability of the host connection and is sensitive to its load-dependant response time. This should not be a problem in normal circumstances, but since the Paris CCPN machines are highly overloaded, this makes life not so easy on some days.

3. MUMM FUNCTIONNAL DESCRIPTION.

As said above, MUMM is a four port "intelligent" controller which establishes communication between the four following devices with standard serial interfaces (RS 232 C) :

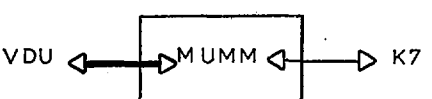
- Operator video console (VDU)
- Minicassette (K7) for audio recording of digital data
- Host Computer (HC)
- Target processor (or User Module UM)

MUMM itself beeing considered as a fifth virtual port (see fig. 1).

For sake of simplicity, not all communications are permanently allowed. The open channels depend on which of three modes of operation has been selected by the user through his VDU.

3.1 Local mode.

MUMM then acts as a stand-alone processor dealing only with two peripherals : VDU and K7, the other two ports beeing disabled.



This mode is mainly intended for

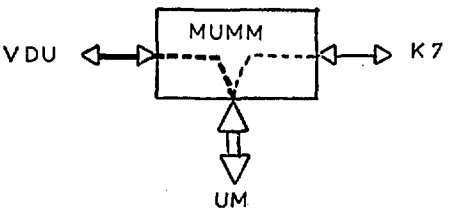
- system parameters definition : transmission speeds, control characters, echo, spy option setting etc...

- EPROM recording.

Whenever needed, MUMM can also be used like a monitor similar to MINIBUG 3 with some additional commands, and the cassette is available to dump or reload part of the memory.

3.2 User Module mode (Target oriented).

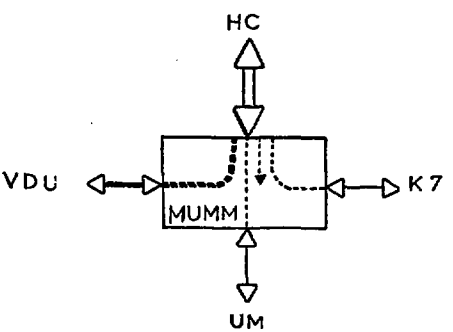
In this mode, dedicated to debugging and running of an application program, MUMM aims at being transparent so that VDU and K7 could be considered as target peripherals.



However, since our philosophy is to be compatible with target standard monitors (e.g. MINIBUG 3 E) which do know of only one peripheral, there exist a prefered (default) connection which is VDU ⟷ UM and for which MUMM is fully transparent. The link is switched to K7 ⟷ UM by MUMM itself by recognition, trapping and modification of specific commands sent from VDU to UM. If needed this trapping may be inhibited.

3.3 Connected mode (Host oriented).

This mode, which is the origin of MUMM design, allows connection of HC to any of the other devices, including MUMM itself. Link switching is controlled from HC via the use of control characters, the default connection being HC ⟷ VDU.



A special procedure is needed for the link HC—UM to allow for the (usually incompati-

ble) specificities of the target monitor and of the time-sharing system (e.g. for end of messages concept).

The goal of this mode is to give access to the host, mainly for binary preparation and down-line loading either into MUMM for EPROM recording, or into the target processor for execution, or into K7 for magnetic back-up.

Moreover, this mode can be used at execution time for data exchange between host computer and target processor[6] :

- Target sending data for processing on the host
- Host controlling the target by sending to it a series of commands retrieved from an "execution disk file".

4. MUMM IMPLEMENTATION.

4.1 Hardware.

MUMM is based on a Motorola 6800 microprocessor and is housed in a CAMAC mechanics to use the crate power supplies, which is very convenient in a H.E.P. environment, but has no relation with the CAMAC dataway. Due to the presence of the RS 232 C connectors on the back, and of the display front panel, its width is 6/25 of the rack.

Two versions have been successively built :

- one entirely with "home made" cards[2)7)] transversely implanted and providing 4 Kbytes of RAM, 4 K of EPROM and the 4 ACIA.

- One using a commercial CPU card[8)] and a specific interface card longitudinaly situated, with a standard Motorola Exorciser connector and providing 4 K of RAM, up to 16 K of EPROM, 4 ACIA and 2 PIA.

The serial interfaces are RS 232 C normalised with rates from 110 up to 19 200 bauds. The VDU port can be changed to a 20 mA current loop and the audio connection follows a Kansas city 300 bauds FSK modulation scheme. The EPROM programmer can process 2708, 2716 and 2732 memories.

4.2 Firmware structure.

The structure of MUMM firmware is shown in fig. 2. It consists mainly of two parts :

- a monitor providing general facilities for interupt processing, I/O handling and bufferisation. The monitor is also responsible for keeping track of the current mode of operation and for switching this mode according to special characters sent by the VDU. In addition, the monitor includes a scheduler for real-time-task activation and servicing.

- 4 "independant" programs called supervisors, each of them dealing with one of the 4 ports, and processing the incoming characters, usually one at a time except for a few exceptions like trapping etc... To avoid the single processor units being kept busy by one of these supervisors, they are normally idle, one of them being activated by the scheduler when it can satisfy a pending request.

MUMM firmware has been written in assembly langage and requires less than 3 K bytes of EPROM for the code and 1 K byte of RAM for buffers and working memories. In addition EPROM programming needs some RAM space in which to load data to be recorded. The scenario of operation is about the following :

A character coming from any port triggers the interrupt handler which calls one of the port input handlers. The character is read from the port ACIA and stored into the circular port input buffer. Then the interrupted activity resumes.

The monitor scheduler keeps looping over the supervisor requests (such as, give me next n characters), looking to see if they can be satisfied (buffer non-empty) and if their turn has come. If so, the scheduler fetches the characters from the related input buffer, stores them in the supervisor mini buffer and finally gives control to the supervisor itself.

The supervisor then processes the character(s) if needed, and according to a destination flag calls the appropriate output handler when it has to output a character to a port. When its task is over, the supervisor sets a new request to the monitor and returns control.

The apparent symmetry of this description in fact hides a fundamental asymmetry of the setup : the four devices connected to MUMM have drastically different behaviours :

- The cassette is a pure slave device which cannot take any decision.
- The target should be able to run under very simple standard monitors (such as Minibug 3) which know of only one kind of peripheral.
- The host and the terminal are in fact the only devices able to decide which links are to be established or destroyed.

Hence the supervisors are split into 2 classes :

- the master class, for which supervisors can change their own destination flag as well as destination flags of other supervisors.
- The slave class which are not allowed to do so.

In the connected mode, the host supervisor is the master, in the U-M mode the VDU supervisor is the master.

Also connected to this asymmetry is the handshake problem. Cassette and U-M monitor cannot support such a situation and hence no systematic handshaking has been implemented between MUMM and other devices, this making MUMM more transparent. It is however possible to have handshaking between the host and the application programs, but MUMM is not explicitely concerned by it.

4.3 Firmware specifics.

Most of supervisors are split into independant parts corresponding to different modes of operation. These differences are explained below :

Connected mode.

The host supervisor switches the different possible links on reception of special characters. The link between the Host and the User module is a special case : seen from the Host the MUMM is considered as an ordinary interactive terminal. The messages received by the Host must be terminated by a carriage return (C.R.) character. But the monitor of the UM has no idea of this necessity. To avoid troubles caused by this incompatibility, the Host supervisor, when it opens the link with UM, tells the UM supervisor to add one C.R. after a given number of characters transmitted from UM to Host. This number is encoded as a 7 bit pattern in the character following the one opening the link. If this number is 0, no transfer is allowed from UM to HC, if this number is 7F, the C.R. will not be added. It is thus possible to link a Host computer with any Target Processor, whether it sends C.R. or not. One has only to modify the interactive programm running on the Host computer.

Once the link from UM to Host has been initialised, the link from Host to UM can be closed, and the link from Host to VDU opened. In this way one can receive some general messages coming from Host such as HOST DOWN or something else more pleasant, but not necessarily useful for the U.M.

The others links with the Host are completely transparent.

U.M. mode.

In this mode the VDU is the Master. The link between UM and K7 is established by the VDU supervisor on recognition of commands destined to the the UM (Loading or Dumping commands). After this trapping the received command is changed to one recognisable by UM and sent to it. This is very U.M.-dependant, but future versions of U.M. will make this more general.

It is also possible to make the VDU supervisor completely transparent. But in this case one cannot use the K7.

5. STATUS AND CONCLUSION.

MUMM has proved to be an efficient and cost-effective tool to work with microprocessors. Two units have been built with first hardware, its firmware has been adapted to two units of an other group, three units with new electronics are presently under construction, and four more are foreseen for educational purposes. It has been used since september 79 with a connection to Paris C.C.P.N. and later on to CERN front ends and CAI experiment Nord 100, in evolutive application set ups.

Its role has extended from μ.P. development support, to a means for providing CAMAC acquisition on large time-sharing systems. Miscellanous applications were also its use as an RS 232 C ASCII spy, an interface between two computers (each of them having only one serial port), or "gearbox" to adapt different transmission speeds.

As a future evolution for it, we foresee two distinct possibilities :

- Development of "Super MUMM" with enhanced firmware allowing for simultaneous enabling of all connections.

- Systematic presence on each application μ. P. of two serial ports supported by the target monitor (i.e. as for M68000 MACSBUG). This would allow a connection of both VDU and Host to the target processor and with some part of MUMM firmware added in it, would allow for most of MUMM functions without it.

Whatever it will be, the path to processor interconnection is wide open.

REFERENCES.

1) G. FONTAINE, "Proposition d'un système de développement pour microprocesseurs bati autour d'un ordinateur hote".
Note CDF mP-03, 26 juin 1978.

2) S. EJZMAN, L. GUGLIELMI, J.J. JAEGER, "Multiplexeur de messages à microprocesseur".
Rapport LPC 80-26, 16 juillet 1980.

3) H. KLEIN and J. ZOLL, "Patchy reference manual", CERN.
See also simplified guide ref. CERN-DD/US/60.

4) C. ADAMS and I. WILLERS, "CERN Motorola 6800 microprocessor support", CERN-DD/US/48 June 1978.

5) H. Von EICKEN, "M68MIL a cross macro assembler for the Motorola 68 000". CERN-DD/80/26 Sept. 1980.

6) G. FONTAINE, L. GUGLIELMI, J.J. JAEGER, S. SZAFRAN, "Equipment calibration with a microprocessor connected to a time-sharing system".
Contributed paper to the Topical Conference on the Application of Microprocessors to High-Energy Physics Experiments, CERN 4-6 May 1980.

7) Electronic cards designed by P. COURTY from our Laboratory.

8) "Notice d'utilisation de la carte multifonction CMF 6800 C/E",
Ets CROS SA, 5 rue Pascal, F-94800 Villejuif.

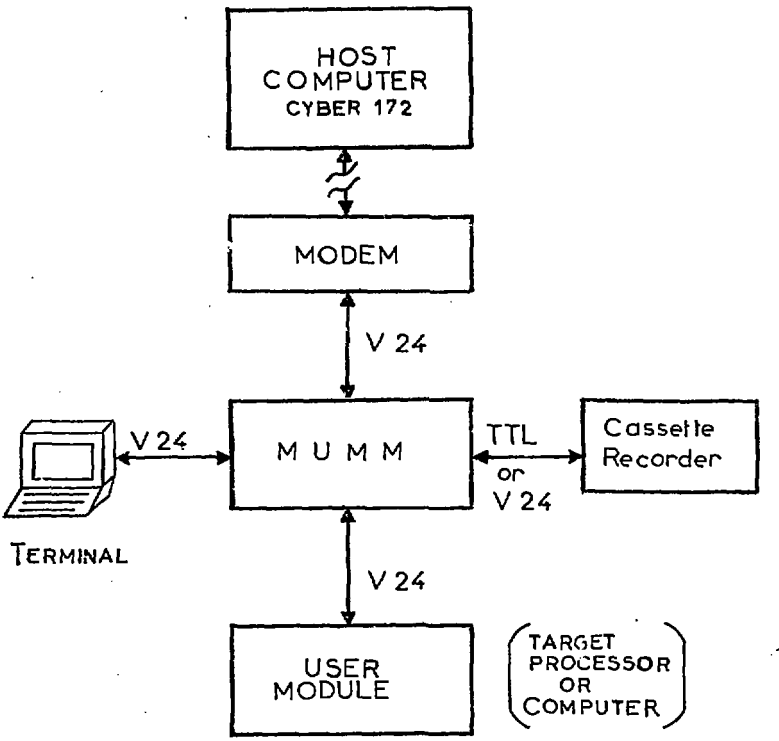
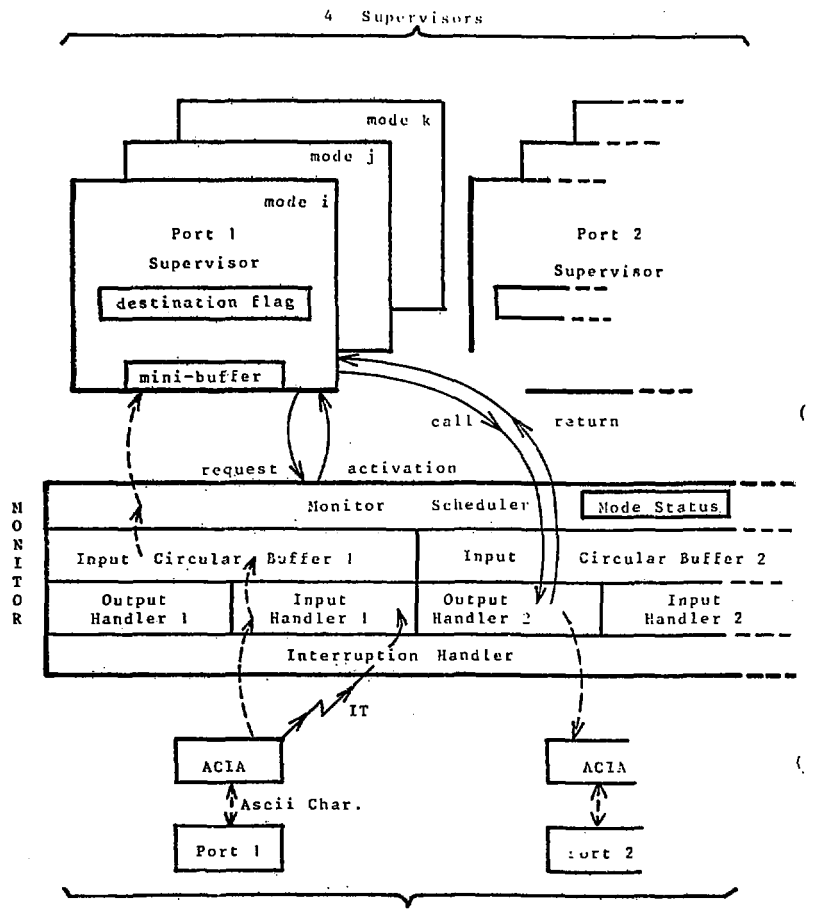


Fig 1 : M.U.M.M. External Interconnections



_____ Data (Characters) _____ Activation (Control)

(for scenario of a character transfer from Port 1 to Port 2)

Fig 2 : M.U.M.M. Firmware Structure