RSX-11/IAS SIG
DECUS U.S. CHAPTER

THE MULTI-TASKER

Volume 14, Number 3 March 1981

The Newsletter of the RSX-11/IAS Special Interest Group

Contributions should be sent to: Editor, The Multi-Tasker, c/o DECUS, One Iron Way, MR2-3/E55, Marlboro, MA 01752

European members should send contributions to: Colin A. Mercer, Tennant Post, High Street, FAREHAM, PO16 7BQ,
Hants, England

Members in Australia or New Zealand should send contributions to: Clive Edington, CSIRO, Computing Research
314 Albert St., East Melbourne, VIC 3002, Australia

Letters and articles for publication are requested from members of the SIG. They may include helpful hints, inquiries to other users, reports on SIG business, summaries of SPR's submitted to Digital or other information for the members of RSX-11/IAS SIG.

All contributions should be "camera-ready copy" e.g. sharp black type in a 160x240 mm area (8 1/2" x 11" paper with 1" margins) and should not include xerox copies. If you use RUNOFF to prepare your contribution the following parameters have been found to be satisfactory:

.PAPER SIZE 60,80 .LEFT MARGIN 8 .RIGHT MARGIN 72 .SPACING 1

These parameters assume output on a lineprinter with a pitch of 10 char/inch. Adjust the parameters to maintain the same margins if another pitch is used.

TABLE OF CONTENTS

Columns

Articles

It is assumed that all articles submitted to the editor of this newsletter are with the authors' permission to publish in any DECUS publication. The articles are the responsibility of the authors and, therefore, DECUS, Digital Equipment Corporation, and the editor assume no responsibility or liability for articles or information appearing in the document. The views herein expressed are those of the authors and do not necessarily express the views of DECUS or Digital Equipment Corporation.

FROM THE EDITOR

The main article in this issue is a description from Digital software development on the RSX-11M QMG/Spooler interface. With this information it should be possible for users to write their own custom output processors. Other articles include a short synopsis of the Miami Symposium from June Baker and true RSX-11M typeahead from Eric Levy. Finally, Jim Mcglinchey and Legare Coleman have included the submission forms the the 1981 Menu ballot. If you have input for them, please fill out the form and send it to them.

The are also the traditional columns. In particular, Speak Out has a submission from Ino Better, a self-proclaimed "Wizard Supreme". I debated long about including this item, but being its about April 1, decided it would be OK.

Finally, if you are not reading your own copy of the Multi-Tasker and are tired of being last on the circulation list, it is simple to get your own copy. Write the DECUS office at One Iron Way, Marlboro, MA, 01752 and ask for a DECUS membership form. When signing up to be a member, make sure to also check the box to join the RSX/IAS SIG. You will then automatically start receiving the Multi-Tasker.

Ralph Stamerjohn
Multi-Tasker Editor

CHAIRMAN'S CORNER

The proposed amendments to the SIG's Operating Procedures to restructure the SIG Steering Committee were overwhelming approved. Of the several hundred ballots received, there were only 2 no votes. Thank you for your support.

On the subject of elections, there will be another election later this year for the seven SIG Executive Committee positions. At the Spring Symposium in Miami Beach, the Steering Committee will present a slate of at least seven nominees, officially opening nominations. Nominations will remain open until September 1st, shortly after which ballots will be mailed to all installation members. Send all nominations to the RSX-11/IAS SIG, care of the DECUS office. Be sure to include the signature of at least five DECUS members.

Note that nominations are for the Executive Committee at-large and not for a specific position. The seven persons elected will, at their first meeting, elect one of themselves as SIG chairman. The remaining six will be assigned responsibilities according to the current activities of the SIG. Results of this election and the various assignments will be published in the following issue of the Multi-Tasker.

As many of you are aware, there as been considerable reference over the past several years to possible charges for newsletters and various other DECUS services. Plans were made to start these funded activities in July (fiscal year '82). DECUS's new Executive Director, Don Frost, convinced Digital management that they should support DECUS newsletters, which after all, are just as valuable to Digital developers as to DECUS members. As a result, Digital agreed to fund newsletter costs (to the tune of about $125,000 per year) for the next year or two.

However, we still plan to continue with other funded activities. Currently they include:

- o Back issues of the Multi-Tasker either in hardcopy or microfiche. The issues would be available on a calendar year basis.
- o Symposia handouts.
- o Pre-symposia seminars (already a reality and a huge success).

More information will be made available on these in the next month or two.

For a long time members have wanted the DECUS Library to distribute more than one program per magtape (or any other media). Well, prior to the San Diego Symposium the Library added several new entries to its catalog. Each consists of a magtape of various library programs, all for a single fee. All of the tapes are 9-track and written in ANSI-format (PIP). All except one are available in 800 or 1600 BPI and include printed documentation. The tapes and their contents are:

- o A collection of the 20 most often ordered RSX/IAS programs in the Library.
- o A collection of 39 RSX/IAS programs generated as part of an experiment to come up with a standard format for all programs in the DECUS Library. This tape includes no printed documentation and is available only in 1600 BPI. It is also written with a 2048 byte blocking factor (must use PIP /BS switch).
- o A collection of the 29 most often ordered RT-11 programs in the Library.
- o A collection of 11 VAX programs.

At the San Diego Symposium, the Library accepted orders for these tapes at a reduced charge - a special offer for Symposium attendees. The response was overwhelming. The Library received orders for over 300 tapes. At this writing the Library has indicated they plan to offer the tapes again at the Miami Beach Symposium. The special rate has not been set yet, but it will probably be the same as San Diego or possibly slightly higher. Prices at the San Diego Symposium for the 4 tapes were $65, $25, $65, and $40 respectively.

Ray French
RSX/IAS SIG Chairman

SPEAK OUT

"Speak Out" is a monthly column for readers to express their opinions or to comment on a previous column. The articles published in this column are an individual's viewpoint and do not necessarily reflect the opinion of DECUS or the RSX/IAS SIG. Readers are welcome to submit articles on any subject concerning the RSX/IAS world. Submissions may be editted by the Multi-Tasker staff for space considerations and clarity.

FINAL AND DEFINITIVE MENU ITEMS

Ino Better
Wizard Supreme

I am bothered by the RSX/IAS Menu. Too much energy is spent on the Menu without the truely definitive improvements to RSX and IAS even making the ballot. It is time RSX and IAS grew up and really begin to perform for its most important user - me. I had hoped DIGITAL and the user community would have seen the light and long ago put these items into RSX and IAS. But I see that it is up to me to bring to the attention of the world the final and absolute needs of programmers everywhere.

1. CONSOLE SERVICES

   - o Add a "SWAP EVERYBODY BUT ME" command. RSX-11M on a 11/70 is a great one-user system. And with this command, I could make it a one-user system.
   - o Add a "ABORT FOREVER" command. There are some tasks which will never run right. This would normally not be a problem, but unfortunately, there are some people who will always try to run them.

2. DEBUGGING AIDS

   - o Do not provide any debugging tools, or alternatively, make the tools as primitive and obscure as possible. I cannot understand the RSX/IAS SIG's fascination with on-line, symbolic debuggers. Experience shows that debugging tools find bugs and good debugging tools find more bugs. Therefore, it follows that no debugging tools, no bugs.

3. EXECUTIVE SERVICES

- o There need to be more zero and one argument directives. I have no trouble remembering EXIT$S and I picked up STOP$S in a snap. However, QIO$ is a problem. I could live very comfortably if I had a dollar for everytime someone tried to service a QIO AST using their I/O status block. And CINT$ and CRRG$ (for novices, connect to interrupt and create region) defy even wizards to code them correctly the first time.

- o Add a CRPL$ (create pool) directive. The mapping directives are nice but users would get much more out of their systems from a create pool directive than the plain, ordinary regions they get from the mapping directives.

- o Add support for shared pool. It is a mystery why users want shared, multiuser tasks and no one has asked for the same from pool. After all, all TCB's are the same length. Think of the pool savings if PIP, MAC, and TKB all shared the same TCB. This concept could be used for all the other pool structures to which it applies.

- o Add a SCRA$ (specify crash AST address) directive. RSX and IAS systems notify user tasks about almost every exception condition except the big one - system crashing. With this directive, my task would be notified when the system is crashing and could take appropriate action, like turn off the nuclear reactor or disable logins.

4. I/O AND FILE SERVICES

- o Pitch the FCS file descriptor and filename blocks and rethink everything else. Nobody understands them anyhow.

- o Add breakthrough support to IO.ATT (attach device). I find it constantly annoying to try to use a device, issue my attach and find someone else has it attached. Now I sit, program dead, and wait for my attach. This problem could be solved quite simply by allowing attaches to breakthrough attaches.

5. SOFTWARE SUPPORT

- o Eliminate the software support services and instead use the money to throw monthly beer and pizza parties for users. In general, I suspect errors would get fixed just as fast and the service would be much more enjoyable.

6. UTILITIES

- o Keep TKB slow. If anything, make it slower. With computers getting bigger and faster, without TKB to rely on, I would never get a coffee break.

SOFTWARE PERFORMANCE REPORTS

This section contains SPR's submitted to the Multi-Tasker by users. SPR's should always be sent to DIGITAL. However, if you feel that a report should be published in the Multi-Tasker, you may send a duplicate copy to the editor at the addresses listed on the cover. Publication of an SPR in the Multi-Tasker does not imply endorsement by the SIG. Implementation of suggested fixes must be at the reader's own risk. The SPR's published in this column may be abstracts of the original submission and have not been checked for accuracy.

The following SPR on RSX-11M V3.2 disk drivers was submitted by Vern Cole (Georgia Pacific, 900 SW 5th, Portland, Oregon, 97204). The SPR is on the same subject as one published in the January, 1981 edition but supplies some more information.

There appears to be an error in all of the hard disk device drivers causing them to operate incorrectly in a powerfail situation. I experienced the trouble on an RP02.

The problem is that a counter in the driver is used to make several retrys if disk access is attempted while the driver is spinning back up after a powerfail. If the driver is not active before the power failure, the counter will be incorrectly initialized; resulting in an attempt to use the disk too soon.

The simplest fix is to change the timer value to a number larger than the spin-up time. The driver will then work correctly, although it may take longer to come back on-line. The timer is set a few lines after the power fail entry point (D?PWF:) in the source code. Its usual value is 15 seconds. The source line takes the form:

```
        nn$:    MOVB     15.,S.CTM(R4)  ;15 SECONDS AT A TIME
```

Change the timer to the maximum value indicated in the table below. The line number in the driver source is also indicated.

DISK	LINE	NEW TIMER VALUE
DB	525	60.
DF	281	30.
DK	308	45.
DL	401	60.
DM	461	60.
DP	334	75.
DR	560	60.
DS	341	30.

HELP YOURSELF

"Help Yourself" is a place for you to get your tough questions answered. Each month, questions from readers will be published. If you have a question, send a letter to the Multi-Tasker at one of the addresses listed on the cover.

We would also like to publish the answers to questions. If you can help someone, send a letter to the Multi-Tasker or call Ralph Stamerjohn at (314) 694-4252. Your answer will be sent directly to the person in need and published in the next edition of the Multi-Tasker.

ANSWERS TO PREVIOUS QUESTIONS

FULL-DUPLEX TERMINAL DRIVER TIMEOUT

The answer I have in the January issue to Dr. Jahn's question on how to set output timeout for the RSX-11M V3.2 terminal driver was confirmed by Jim Downward of KMS Fusion. However, the value should only be made larger, setting the timeout to zero has a bad effect.

8085 CROSS COMPILER

Bob Martin got several answers to his question in the January issue for a 8085 cross compiler. He decided on the Whitesmith "C" compiler and reported he was very satisfied with it.

WHERE IS STANFORD SYSTEMS?

Mike Neale had a answer for Richard Fisher's request on an alternate vendor for CDC 200 UT support. The Software Results Corporation, Columbus, Ohio, 32412 offers a CDC UT200 emulator package for RSTS, VAX, and RSX.

THIS MONTH'S QUESTIONS

PDP-8 EMULATION

Can anyone supply me with information about any hardware or software PDP-8 emulators for PDP-11's or VAX. We would use a PDP-8 emulator as a first step in scrapping our PDP-8's.

Warren Jacobs, OTB Corporation, 1501 Broadway, New York, New York, 10036. Phone (212) 221-5461.

IAS PRINT QUEUE RESTART

We are a business environment and often print 3,000 to 10,000 page listings on our IAS V3.0 system. When the paper jams in the middle of a listing, we have to restart printing the file from the beginning. Can anyone suggest a method or routine that would enable us to start a few pages back in the file?

M.E. Nnanwubar, Nigerian External Telecommunications Ltd. Necom House, 15, Marina, P.O. Box 173, Lagos, Nigeria. Phone 659666.

RSX PROCEDURAL TEST LANGUAGE

I need a procedural test language, for example ATLAS, for RSX-11M. Such a language would allow automated or semiautomated testing sequences, a means of on-line test procedure development, and a means of supplying a hard-copy log of all test operations. If anyone knows of such a language, please contact me.

Karl Hogquist, 5117 Woodwind Lane, Anaheim, California, 92807. Phone (714) 779-6065.

SPRING '81 DECUS SYMPOSIUM

June Baker
RSX/IAS Symposia Coordinator

Bring your software problems to the Spring '81 DECUS Symposium to be held in Miami Beach, 18 - 21 May and have them diagnosed and treated by out staff of specialist physicians available at our Software Clinic. We were overwhelmed by the success of out two hour trial clinic at the Fall '80 Symposium in San Diego, so we have quadrupled the time available to you. Experienced users and DIGITAL personnel can help you with your individual problems, so bring available documentation with you.

At this symposium we're emphasizing RSX/IAS operating system tutorials rather than introducing many new products or releases of software. Several IAS technical sessions will emphasize version 3.1 features, customizing queue management, internals of PDS and timesharing control primitives. We will have more IAS sessions than ever before, so we hope to see many IAS users. RSX-11M sessions include queue management and despooler, a FILES-11 tutorial, system tuning and performance workshop, and an introduction to DIGITAL's software performance monitor. We've included an RSX-11M+ technical session with emphasis on its unique features.

A number of users have submitted papers on applications which use RSX operating systems. You'll have an opportunity to listen to the developers and ask questions at the end of their talks.

Old favorites have been scheduled by the SIG. Both RSX and IAS question and answer sessions have been scheduled for evening and will undoubtedly go on late, late, late! These sessions are forums in which you can get your personal questions answered by experts -- user and DIGITAL developers -- and hear solutions to someone else's problems which may be similar to yours.

IAS and RSX Magic sessions are for wizards and apprentices to participate in open-ended sessions which traditionally deal with non-standard uses of the operating systems and anything else you can think of.

The SIG is sponsoring several pre-symposium seminars to be held the day before the symposium at the Fountainbleu Hotel. These will be announced in the Preliminary Program which will also have information on registration and cost.

Tape copy is alive and well, and the SIG plans to have a tape from this symposium.

Miami Beach presents some logistical problems. We have a headquarters hotel (the Fountainbleu) where the evening sessions will be. The receptions and other evening activities will be there, too. Several other hotels in the immediate area will be available for housing, also. During the day sessions will be at the Miami Beach Convention Center, and there will be a bus shuttle service between hotels and the convention center. We recommend that you do not rent a car, since parking is limited and expensive. The Exhibit Hall will be in the convention center as will be the Bookstore and the innovative DECUS store. Campgrounds will be the convention center, since most people will be spending the majority of their time there.

If you need solutions to you operating system problems and information about RSX and IAS, plan to come to Miami Beach to learn, meet other users, DIGITAL managers and developers, and have a good time. The SIG is looking forward to meeting you!

TRUE RSX-11M TYPEAHEAD

Eric Levy
Jet Propulsion Lab

For some time now we have been running with a patch to the RSX-11M V3.2 full duplex terminal driver to implement a flavor of typeahead as it exists on most other DIGITAL systems (VAX, IAS, RSX-11D, TOPS-10, etc.). That is, all text typed at the terminal is put into a ring buffer and removed as requested - across tasks without regard for attaching the terminal. Also, there is no such thing as unsolicited input going to MCR since all input going to MCR since all input is buffered until read by someone. The patch for this is remarkably simple - just two instructions in the entire driver need to be noop'ed out. Well, if this is so great, why keep it a secret? The answer is that once the normal MCR is gone, each terminal needs its own command line interpreter (CLI) to repeat the sequence <read, spawn>. This has a pool penalty as well as general slowdown.

However, we felt the cost was well worth the price as we do only program development on our system. In this type of system, its the number of keystrokes that count so we created a CLI that did somewhat more (editting, saving recent commands, dynamic user definition for all unused control characters, CRT functions, etc.). Now that the software tools package is here, the UNIX-like shell is also used by some of us. I believe the rest of the RSX community would be interested in this patch for general purpose typeahead.

While in theory it is a 2 instruction patch, I implemented it dynamically so not every terminal need see the patch. To do this, I required a small area where I could store a boolean for each terminal in the system (since I was not thrilled with the idea of finding room in the UCB). This table could be anywhere in the lower 20K, but I chose to have a partition at 20K-100. Even though it is not technically part of the executive or pool, all drivers map to the fill 20K. This may be unreliable in the future but based on my correspondence with others, I feel this patch may not be required in version 4.0 anyway.

True typeahead is implemented by nooping out two instructions in the terminal driver module TTICH.MAC. However, at our site, we have added logic to enable or disable the patch on a per terminal basis. The first place to patch is in TTICH around line 172. By skipping the I1CTXL call, the driver does not flush the typeahead buffers on detach. This can be done for all terminals by nooping out the CALL I1CTLX line or code to test the terminal line could be added (see later for example).

```
PPDET:: MOV     R3,-(SP)                ;SAVE R3
;
; - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
; Add code here to test if I1CTLX should be skipped for ;
; true typeahead. If so, branch to xxxx$                 ;
; - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
;
        CALL    I1CTXL          ;Flush type-ahead buffer
xxxx$:
        BIC     #S1.CTO,@R5     ;Clear CTRL-O bit
```

The second patch is also in TTICH around line 1363. This patch never lets the driver put anything into the MCR unsolicited input buffer. Instead terminal input always goes into the typeahead buffer. This can be done for all terminals by nooping out the BEQ 60$ line. Again, the line test code could be added for per terminal support.

```
40$:
;
; - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
; Add code here to test if MCR input should be skipped   ;
; for true typeahead. If so, branch to xxxx$.            ;
; - - - - - - - - - - - - - - - - - - - - - - - - - - - - ;
;
        TST     U.ATT-U.TSTA(R5)        ;;;Terminal attached?

        .IF DF  T$$CCA&A$$TRP

        BEQ     60$                     ;;;N - jump
xxxx$:
        MOV     U.TUX-U.TSTA(R5),R4     ;;;Get UCBX
        BEQ     RETCS                   ;;;see text below
```

The BEQ RETCS is added for safety reasons. It may not be necessary but it seemed the safe way to do it here. I also assume the conditional T$$CCA is true.

You can try the patch by modifing the terminal driver on the fly with noops. Once finished, type control-C and @TI: so you have something which does a <read, spawn>.

For completness, the following is the code we use to dynamically use the patch on a per terminal basis. JPLPAR is the beginning of a 100 byte boolean table. Each byte is a terminal, starting with TT0. A non-zero value causes the patches to take effect. I define JPLPAR with a GBLDEF task builder option.

```
        MOVB    U.UNIT-U.TSTA(R5),R4    ;Get TT# = index in JPLCOM
        BIC     #177400,R4              ;Clear for safety
        MOV     R4,-(SP)                ;Save on stack
        MOV     U.DCB-U.TSTA(R5),R4     ;Get the DCB pointer
        MOVB    D.UNIT(R4),R4           ;Get beginning unit #
        BIC     #177400,R4              ;Clear for safety
        ADD     (SP)+,R4                ;Compute TT#
        TSTB    JPLPAR(R4)              ;See if JPLMCR desired
        BNE     xxxx$                   ; If ne - yes, skip
```

If you use this patch and like it, please write Digital and DEMAND full typeahead in V4.0. I know of at least one person who has an excellent way of implementing this if Digital would permit.

Eric Levy
Jet Propulsion Lab
4800 Oak Grove Drive
Pasadena, CA, 91103
Building 264/125

(213) 354-2969

QMG/OUTPUT PROCESSOR INTERFACE

This document describes the interface between the Queue Manager (QMG) and an output processor, or despooler. The DIGITAL-supplied line printer processor, LPP, is used as an example. The information is intended to help those who wish to write their own custom output processor.

1.0 **RSX-11M SPOOLING SYSTEM**

The interface described in this document is only one part of the RSX-11M spooling system.

The RSX-11M spooling system consists of the following:

- Queue Manager (QMG)
- Command line interpreters
- Output processor (despooling tasks)
- The queue file (SY:[1,7]QUEUE.SYS)

SPOOL is an acronym for Shared Peripheral Operations On Line. It refers to the orderly collection, storage, and processing of output jobs. Spooling is inserting a job into an output queue. Despooling is removing the job from the queue to process it for output.

A job is a collection of one or more files treated as a single output entity. For example, a print job is a collection of one or more files to be printed. Jobs are created when you issue PRINT commands or when tasks such as MAC and TKB issue programmed requests.

Jobs are stored in a queue file. A queue file is a disk file containing the complete description of each spooled job. QMG controls the creation of the jobs by inserting an entry for each job into the queue file. Then QMG processes the jobs by sending data packets to appropriate processors.

QMG... is the default name for the Queue Manager task. It is a privileged task that maintains the overall control of print job processing. It controls the print job queues and output processors and all their activity. QMG communicates with the command line interpreter (QMGCLI) for the creation and queuing of print jobs. It communicates with the output processor (LPP) for the orderly despooling and printing of the print jobs.

A command line interpreter is a task that takes system-level command lines, determines what functions they specify, and sends requests to QMG. QMGCLI is the RSX-11M Output Spooling command line interpreter. It interprets both PRINT and QUEUE command lines and sends packets to QMG. (See the RSX-11 Utilities Manual, Chapters 6 and 7.)

A processor is a task that despools (processes for output) a spooled job. The Queue Manager notifies the processor which job to process and which files belong to that job. The basic function of a processor is to handle the output processing (printing, punching, or whatever device-specific action is required) of the files; scheduling and control are handled by QMG. You can write an output processor to handle any kind of job. Although a line printer processor is used as the example here, your processor could be for any kind of low-speed device, such as for a card punch, or for a batch system.

The privileged task, LPP, is an output processor that prints the flag pages and controls the printing of files as directed by QMG. An output processor requires two things:

- The presence of a specific physical device in the system configuration
- A task name of the form ddPn, where dd is the device class and n is the unit number

For example, the default task name of the processor for device LP0: is LPP0.

The two tasks, QMG and LPP, are always active when the Queue Manager is running on the system. Both tasks are stopped when not in use. Therefore they are checkpointable, which allows more efficient use of memory.

Part of the standard protocol between QMG and other tasks is that QMG remains stopped until started by some other task, either an interpreter (by user commands) or a processor (by responses to previous QMG commands), or by completion of a file or job.

Communication between the tasks QMG and an output processor, for example LPP (or between QMG and QMGCLI), is accomplished by the use of SEND/RECEIVE DATA directives, EXIT WITH STATUS directives, and, in the case of LPP, RECEIVE DATA AST. (QMG may also send data packets to QMGCLI to return status or to provide the file identifier of the active queue file for a QUE /LIS command.) The data packets, referred to here simply as packets, contain all the information needed by the receiving task to determine the action desired and the parameters needed for the action.

Once it is started, QMG takes appropriate action, which may include sending more packets to the interpreter or processor, and then stops itself.

Similarly, the processor remains in a stopped state waiting for a RECEIVE DATA AST. After receiving a packet, it takes appropriate action, responds to QMG, and stops itself.

(See Chapters 6 and 7 of the RSX-11M Utilities Manual for complete information on how to set up and use the spooling system.)

2.0 OPERATIONS

There are a limited number of dialogues between QMG and an output processor. The basic operations are:

- Processor startup
- Processor exit
- Processing a print job

- Stopping an active processor
- Continuing a processor

An operation on a job being processed requires a dialogue between the Queue Manager and the despooling task. For example, to delete a print job being processed, the basic operations are processing a print job and stopping an active processor. The dialogue between QMG and LPP looks like this:

```
     QMG to LPP:    STOP          LPP to QMG:    SOLICITED PAUSE

     QMG to LPP:    END OF JOB    LPP to QMG:    DONE
```

All dialogues take place by means of SEND/RECEIVE DATA directives which transfer data packets back and forth between QMG and the LPP processor. Each of the data packets is described in detail in Section 3.

2.1 Processor Startup

To start up a processor, the processor task must be previously installed with the proper task name, for example, LPP0 for LP0:. After QMG is directed to initialize a processor, it attaches the device (if the processor is device-specific), requests the processor, and issues a CONNECT to the processor. QMG then sends the processor the STARTUP packet. The processor issues an ALUN$ directive (Assign Logical Unit Number) to assign LP: (or the card punch, or whatever device you are spooling to). Then it stops itself and waits for a RECEIVE DATA AST.

There is only a single message in the dialogue when starting a processor. For example, to start LPP in response to the user command

```
     >QUE LP0:  /SPOOL
```

the communication is simply

```
     QMG to LPP:  STARTUP
```

2.2 Processor Exit

To terminate a processor, QMG unstops the processor and sends it an EXIT packet. The processor exits, returning status via the EXIT WITH STATUS directive. QMG then detaches the device if there are no other processors assigned to it.

2.3 Job Processing

To despool a job, QMG sends two packets to the output processor (LPP, for example) the START OF JOB packet and the continuation of the START packet. Then the process loops as QMG sends a PROCESS FILE packet for each file, and LPP responds with a DONE packet after successfully printing it. Finally QMG sends and END OF JOB packet, and LPP responds with one more DONE packet for end of job.

For example, to process a job in response to a command such as

```
     >PRINT FILE.DAT
```

the dialogue between QMG and LPP is

```
     QMG to LPP:  START OF JOB
     QMG to LPP:  second START OF JOB packet
     QMG to LPP:  PROCESS FILE  LPP to QMG:  DONE
     QMG to LPP:  END OF JOB    LPP to QMG:  DONE
```

When you write your own despooler, this part of the dialogue will be specific to your application.

2.4 **Stop Processor**

To stop the output processor, QMG sends it the STOP packet. The output processor responds with a SOLICITED PAUSE packet and stops.

For example, to stop LPP in response to the command

```
     >QUE LP0:  /STOP
```

the dialogue is

```
     QMG to LPP:  STOP           LPP to QMG:  SOLICITED PAUSE
```

2.5 **Continue Processor**

QMG directs a processor to resume activity by issuing a CONTINUE packet to the stopped processor.

For example, if you want to stop the processor and continue after you have fixed the printer, the dialogue would be

```
     QMG to LPP:  STOP
     LPP to QMG:  SOLICITED PAUSE
     (Then, when you issue the command
     >QUE LP0:  /START)
     QMG to LPP:  CONTINUE
```

3.0 **DATA PACKETS**

The paragraphs below describe each of the packets. The first byte of each packet is the function code, Q.PFUN for messages from QMG to LPP and Q.MFUN for messages from LPP to QMG. These codes are shown in Table 1.

Table 1
QMG/LPP Function Codes

Function	Symbol	Value
START OF JOB	QP.SOJ	0
PROCESS FILE	QP.FIL	1
STOP	QP.STP	2
CONTINUE	QP.CON	3
END OF JOB	QP.EOJ	4
STARTUP	QP.STA	5
EXIT	QP.EXT	6
DONE	QM.DUN	100
SOLICITED PAUSE	QM.SPA	101
UNSOLICITED PAUSE	QM.UPA	102

All of the global symbols are defined in QMGSYM.MAC, which is included as an object module in QMG.OLB.

3.1 **Messages from QMG to LPP**

3.1.1 **STARTUP** - The STARTUP packet (see Figure 1) initializes a specific processor. The processor task must be installed with the appropriate task name -- for example LPP0 for LPP -- before this command is issued. If it is a device-specific processor (rather than a batch processor, which is not device-specific), then the device must not be attached by any other task.

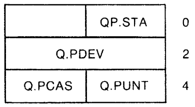

Figure 1 STARTUP Packet

Q.PFUN
The processor function. This byte determines the function that QMG is sending to LPP. The function code for STARTUP is QP.STA (5).

Q.PDEV
Two ASCII characters designating the device to be started, for example LP or TT.

Q.PUNT
The unit number of the device class named above. One binary byte.

Q.PCAS
The case designation, either:

0=upper case only
1=upper/lower case

3.1.2 **EXIT** - The EXIT packet (see Figure 2) is the complement of the STARTUP packet. The processor terminates and returns to the dormant state.

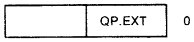

Figure 2 EXIT Packet

Q.PFUN
The function code for EXIT is QP.EXT (6).

3.1.3 **STOP** - QMG controls when to stop the processor with the STOP packet (see Figure 3). The packet may be sent while a processor is busy, to cause an immediate stop, or at the end of a job, to cause a stop for changing forms or altering the number of flag pages.

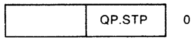

Figure 3 STOP Packet

Q.PFUN
The function code for STOP is QP.STP (2).

3.1.4 **CONTINUE** - The CONTINUE packet (see Figure 4) informs the processor to continue where it was stopped or to continue at a new place in the file, as specified by the other parameters in the packet.

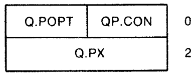

Figure 4 CONTINUE Packet

Q.PFUN
The function code for CONTINUE is QP.CON (3).

Q.POPT
Q.POPT is the processor continue option byte. The actions that may be taken on continue are:

- QP.CUR - continue at current page
- QP.AT - continue at page number in Q.PX
- QP.FWD - continue Q.PX pages forward from where stopped
- QP.BAK - continue Q.PX pages back from where stopped
- QP.ALN - align to top of form before continue

Q.PX
This word contains the value specified for the QP.AT, QR,FWD, or QP.BAK options.

3.1.5 **START OF JOB** - The START OF JOB function consists of two packets sent to the processor in succession and directs the processor to begin processing a job. The first packet contains information concerning the job name and job parameters. The second packet contains the first file specification and number of files in the job, information used for the fourth line of the job flag page only.

The format of the first packet is shown in Figure 5.

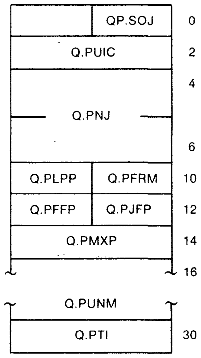

Figure 5 First START OF JOB Packet

Q.PFUN
The function code for START OF JOB is QP.SOJ (0).

Q.PUIC
The UIC the job was created under, in the form:

low order byte - member
high order byte - group

Q.PJN
The job name. Two words containing six Radix-50 characters.

Q.PFRM
The forms type.

Q.PLPP
The number of lines per page.

Q.PJFP
The number of job flag pages.

Q.PFFP
Include file flag pages (equal to the number in Q.PJFP). (0=no,1=yes)

Q.PMXP
The maximum number of pages.

The second packet for the START OF JOB function is shown in Figure 6.

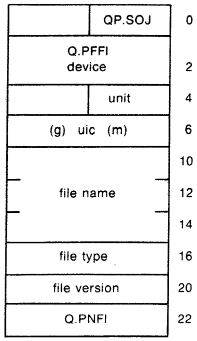

Figure 6 Second START OF JOB Packet

Q.PFUN
For the second START OF JOB packet the value is still QP.SOJ (0).

Q.PFFI
The first file specification (used for job flag page line).

2 bytes - ASCII device name
1 byte - binary unit number
1 byte - not used
1 byte - binary UFD member number
1 byte - binary UFD group number
6 bytes - 9-Radix-50-character file name
2 bytes - 3-Radix-50-character file type
2 bytes - binary file version

Q.PNFI
The number of files in the job (for the job flag page).

3.1.6 **PROCESS FILE** - QMG sends the output processor a PROCESS FILE packet (see Figure 7) for each file in the job. For example, the packet that QMG sends to LPP contains information about the file, including full file specification, number of copies desired, and the page to begin printing. The file specification can be given in either file name format (Figure 7) or file ID format (Figure 8).

Your despooler may require different information, especially if it is not for a line printer.

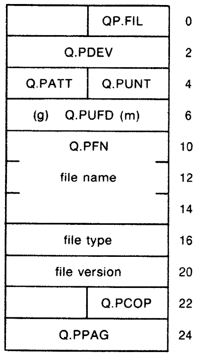

Figure 7 PROCESS FILE Packet with File Name Format

(FID Format)

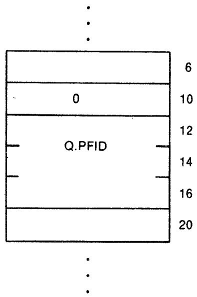

Figure 8 PROCESS FILE Packet with File ID Format

Q.PFUN
The function code for PROCESS FILE is QP.FIL (1).

Q.PDEV
The device class. Two ASCII bytes.

Q.PUNT
The device unit number. One binary byte.

Q.PATT
The attribute byte for the file. One binary byte.
QP.ADE - set to delete file after printing.

Q.PUFD
The file's UFD in binary form:

low order byte - member
high order byte - group

Q.PFN
The file specification. Ten binary bytes.

6 bytes - 9-Radix-50-character file name
2 bytes - 3-Radix-50-character file type
2 bytes - 1 word file version

Q.PCOP
The number of copies of this file. One byte.

Q.PPAG
The page number to start at. Zero means first page.

3.1.6.1 **File ID Format Fields**

Q.PFN
The first word is zero to indicate file ID format.

Q.PFID
The file entry ID.

2 bytes - File Number
2 bytes - Sequence Number
2 bytes - Relative volume number (0)

3.1.7 **END OF JOB** - The END OF JOB packet (see Figure 9) tells the processor the job is finished. It is also used to delete an active job if issued before the last file is processed.

	QP.EOJ	0

Figure 9 END OF JOB Packet

Q.PFUN
The function code for END OF JOB is QP.EOJ (4).

3.2 **Messages from LPP to QMG**

3.2.1 **DONE** - This is the packet that the output processor uses to respond to QMG after completing a file or a job (see Figure 10).

Q.MOPT	QM.DUN	0
Q.MPP		2

Figure 10 DONE Packet

Q.MFUN
The function code for DONE is QM.DUN (100)

Q.MOPT
The option byte that tells QMG which entity is done.

QM.FIL - file done
QM.JOB - job done

Q.MPP
The total number of pages printed.

3.2.2 **PAUSE** - The PAUSE packet (see Figure 11) is a response to inform QMG that the processor has stopped. A SOLICITED PAUSE is a response from a QMG-directed stop, while an UNSOLICITED PAUSE is a response from a processor-directed stop.

	QM.SPA QM.UPA	0
Q.MPP		2
	Q.MCP	4
Q.MCUR		6

Figure 11 PAUSE Packet

Q.MFUN
There are two kinds of pauses: Solicited, QM.SPA (101) and Unsolicited, QM.UPA (102).

A solicited pause is issued by the QUE/STOP command.

An unsolicited pause occurs when the processor stops itself. The DIGITAL-supplied despooler does not issue any unsolicited pauses.

Q.MPP
The number of pages printed.

Q.MCP
The number of copies printed.

Q.MCUR
The current page number (0=first).

1981 RSX/IAS MENU SUBMISSIONS

Legare Coleman
Jim McGlinchey

Menu Coordinators

The RSX/IAS Special Interest Group conducts a yearly balloting to determine the most important issues affecting DIGITAL's real-time operating systems and hardware. DIGITAL is committed to formally respond to the menu balloting and depends on the results of the voting to help set priorities and direction for product development.

The actual balloting procedure we follow consists of two phases. In the first phase, we solicit ballot items from all RSX/IAS users. These are compiled into a menu of approximately 75 items. During the second phase ballotes are mailed to all SIG members, the results are tabulated, and then formally presented to DIGITAL. The menu results, along with DIGITAL's formal response, are presented to the membership at the next DECUS Symposium.

We are currently in the first phase of the current (1981) menu cycle and have been soliciting candidate items for the next ballot. The ballot mailing will occur during the summer of 1981 with the results being presented at the Fall, 1981 DECUS Symposium.

We are asking for the participation of each SIG member in the menu process. Many members have submitted menu items on the form we distributed at the last DECUS Symposium. The following page is a copy of the same form for those who wish to submit menu items. We ask that all menu items be submitted to us by May 1, 1981.

Please, only one item per form. You may make as many copies as you need. Menu items may be mailed to either of these addresses.

Jim McGlinchey
P.O. Box 'H'
Horsham, PA, 19044

H. Legare Coleman
Phillip Morris Research
Box 26603
Richmond, VA, 23261