CLM - P 566

CLM - P 566

UKAEA

Preprint

JOSEQ - A JOB SEQUENCER

V J CALDERBANK
E G MURPHY

CULHAM LABORATORY
Abingdon Oxfordshire

1979

CLM-P 566

JOSEQ - A JOB SEQUENCER

V J Calderbank and E G Murphy

Culham Laboratory, Abingdon, Oxon, OX14 3DB, UK
(Euratom/UKAEA Fusion Association)

ABSTRACT

During the on-line acquisition and analysis of data for physics experiments, it is important to keep to a minimum the real time required to turn around a sequence of analysis programs and return the results to the experimenter. This paper describes a job sequence scheduler which has been written for an ICL System 4-70 computer running under the MULTIJOB operating system. Six man months of effort were expended to produce the program and a six-fold reduction in job turn around time was achieved at the expense of less urgent jobs in the system.

(Submitted for publication in Software-Practice and Experience)

November 1978

INTRODUCTION

At Culham Laboratory many large experimental devices are built to carry out research into Nuclear Fusion and Plasma Physics. Data from these experiments are collected by minicomputers which are connected both to the experiment and to the laboratory's mainframe computer - a dual processor ICL System 4-70 linked to a Modular One Front End. Several such remote minicomputers are serviced by this central system which must also satisfy the other computing needs of the laboratory[1].

The ICL 4-70 runs under the MULTIJOB[2] operating system which is a multi-access and multiprogramming system. At Culham, up to 28 jobs can be run simultaneously in 9 separate streams some of which are batch and others roll-in, roll-out (RIRO) streams. Once a job has been loaded into a batch stream it remains in store until execution is completed. In a RIRO stream a job remains in store for some predefined processor time slice, after which it is rolled out, regardless of whether it has completed or not. This can only happen when the current input and output has been completed. Another job may then be rolled in for its time slice. This process is continued for all jobs in the RIRO queue and each job is rolled in and out as often as is necessary to complete it.

The MULTIJOB system maintains queues of jobs waiting for execution in each stream. If a sequence of jobs is to be run, the user must set up a procedure file containing the job control statements for every job in the sequence. The system Job Management routines read and process this procedure file and enter the first job in the queue for the required stream. There it must wait until processor time and store become available for it. It is not until this job has completely finished that the next job is entered in the queue for its stream. It, too, must wait until the necessary resources become available to it. This is continued for all jobs in the sequence. The waiting time in the queue depends on many factors including the amount of time and

store the job requires and the number of jobs already in the queue. It can be seen that when the system is busy, a sequence of jobs may be delayed for long periods thus wasting valuable experimental time.

Most fusion experiments are pulsed devices. Plasma is produced within strong magnetic fields for brief periods (varying from a few microseconds to a few seconds) by the application of an electrical discharge. These pulses are applied at regular intervals (typically ten minutes) and it is during these experimental shots that experiments are carried out on the plasma and data are collected. Experimenters write individual analysis programs for processing this data. Several such programs may be required to analyse a given set of data and different analysis programs may be used from time to time. Thus for each experimental shot, a sequence of computer programs must be run to collect and file the data, analyse it and return printed and graphical output directly to the experiment. The result of one such analysis affects the course of action taken at the next experimental shot and so it is necessary for these results to be reliably returned to the experiment in a time somewhat less than the pulsing interval. With the standard operating system the turnaround time for simple jobs has been measured to be anything between 80 seconds and 30 minutes depending on the computer load at the time. It typically averaged around 9 minutes.

It was decided that some attempt be made to remedy this so that results could be returned to the experiment more quickly and reliably. This paper describes the resulting program (JOSEQ) which avoids the delays of the standard system by acting as a self-contained Job Input and Job Scheduler. A block diagram showing the relevant features of the hardware and software system is shown in Figure 1. JOSEQ also interfaces to the MULTIJOB operating system, for loading and running programs and for error recovery, and to the communication program BEPE[3] which communicates with the remote computers at the experiment. JOSEQ may be initiated by the remote computers via BEPE and will run the user's job sequence under its own control.

DESIGN OBJECTIVES

JOSEQ is a self-contained Job Input and Job Scheduler which has been designed with the following main objectives in mind:

(a) it must load and execute any number of independent programs in a sequence without incurring the usual system overheads and delays;

(b) it must operate on a job control procedure file identical to that submitted to MULTIJOB and, unlike MULTIJOB, it must handle file deletion and replacement immediately;

(c) user programs must be capable of running without modification both under MULTIJOB and under JOSEQ. JOSEQ must be able to run segmented programs;

(d) it should provide all the diagnostic facilities and program error traps provided by MULTIJOB such as recovery of user programs from endless loops. The user program must be forced to return control to JOSEQ whether it runs to normal completion via FORTRAN STOP (or usercode end-of-job) or receives an error interrupt. Neither the user program nor JOSEQ must ever terminate except under well defined conditions;

(e) it should keep statistics to provide user program monitoring information which will help in the spotlighting of inefficient programs.

DESIGN AND IMPLEMENTATION DETAILS

JOSEQ is a System 4 Usercode program consisting of 12 K bytes of code. It runs as a user job under the MULTIJOB system but carries out some privileged operations. The program consists of two modules - the Job Input module and the Job Loader. Job Input is an interpreter which reads and checks job control statements in a procedure file, deals with instant file deletion and replacement and stores information required subsequently by the Job Loader. In particular, it saves the names of the files containing the composed programs

to be run, the time limit for the run, the names of input and output files required by the user program and a copy of the run time parameters if any. It then calls Job Loader to run the job. This loads the user job into a large area of store reserved at the beginning of JOSEQ and uses the information passed to it by Job Input to set up file parameter and run time parameter tables for the user job in the highest store addresses of the stream. It then starts its execution. A diagram of the layout of the store is shown in Figure 2. FORTRAN STOP (or a special EOJ macro in the case of usercode programs) returns control to Job Loader and thence to Job Input to process the next job in the sequence. On completion of the entire sequence of jobs JOSEQ suspends itself and awaits re-activation by the communication program (BEPE) which is the link with the remote computers controlling the experiment.

JOSEQ saves time over the standard system in several ways. Firstly, and most importantly, it eliminates all queueing time except roll-in, roll-out time if run in a RIRO stream. The only job which is queued and loaded by the MULTIJOB system is JOSEQ itself once per session. Secondly JOSEQ avoids writing job description blocks into the system stream queue files and thus saves disc transfer times. Thirdly it recognises only a subset of the entire job control commands in the system and provides only a minimum of input validation. Since the Job Control statements for JOSEQ are standard JCL, submission to MULTIJOB JOB INPUT can be used to generate error messages if the JCL statement file is suspect. Fourthly a system adjustment allows JOSEQ itself to be scheduled with top priority so that it is always loaded and entered quickly. In our installation it is run in a RIRO stream capable of executing up to 4 programs simultaneously. One of the four slots in the RIRO stream is provided for each experiment using JOSEQ.

All these factors have produced a three-fold saving in the turnaround time of a job sequence, reducing what used to be the average 9 minute analysis to 3 minutes partly at the expense of less urgent jobs which could be held up

by JOSEQ and partly by eliminating queueing. Furthermore the results are always returned to the experiment reliably within this time which depends only on the number of RIRO slots in use by other experiments and the amount of computation demanded by those experiments.

USER PROGRAM OPTIMISATION

Further time savings have been achieved by providing the user with monitoring information which can help spotlight particularly inefficient jobs in a sequence thus enabling optimisation to be carried out. Inefficiency is defined as a poor ratio of elapsed time to processor time.

In the MULTIJOB system a journal file is produced for every job in a sequence and this file gives the user some timing statistics logged to the nearest elapsed time unit (where 1 etu = 3.6 seconds).

When JOSEQ is run, one system journal is produced for JOSEQ itself and, in addition, JOSEQ creates its own separate journal file for logging information from every job in each user sequence run. This file is created in the Project Leader's filespace and there is one such file per project. The times logged in this file are the time that JOSEQ's JOB INPUT is entered, the time that the JOB LOADER is entered, the time that execution of the user job is started and the time that this job terminates. On completion of each user job the number of etus (accurate to one tenth of an etu), the elapsed time (in seconds) used by the job and the ratio of seconds to etus are output. At the end of each sequence the total number of etus and the total number of seconds used by the entire sequence are printed. At the end of the session, when JOSEQ is finally abandoned, the total etus used by that session are also logged.

This information gives the Project Leader a more accurate account of the use of the system than is provided by MULTIJOB journal files. Studies of this journal revealed that some user jobs required 30 or more seconds/etu when

running as the sole job whilst others required 10 or less. Since one processor etu is equivalent to 3.6 seconds, a job which requires 30 seconds per etu is spending approximately one-eighth of its time using the processor to do useful work and seven-eighths of its time occupying store and waiting for the supervisor to carry out I/O operations. Users were therefore advised to optimise these particularly inefficient jobs so that the ratio of seconds to etus was as small as possible.

Optimisations of this kind produced a further saving of a factor 2 in turnaround time. Thus a typical average 9 minute turnaround was reduced to about 1½ minutes. Not all of this saving, of course, was realised because users took advantage of the better facilities to carry out more complex analyses.

DIFFICULTIES ENCOUNTERED

Delving into a complicated operating system always presents difficulties. The greatest problems in implementing JOSEQ arose when changes actually had to be made to MULTIJOB supervisor tables or when some detailed knowledge of the system was required. Manuals are useless for solving specific problems such as this and one has to rely on experiment and people who are expert on appropriate aspects.

The first problems were caused by the interface between MULTIJOB, the user job and JOSEQ particularly for user job entry and for both normal and abnormal termination. Fortunately a system macro to load and enter a program is available to users on System 4 and so user job entry presented no particular problem except in the case of segmented programs (see later). However user job termination did present more difficulty since if a sequence of user jobs is to run under JOSEQ then individual jobs in that sequence must not be allowed to terminate when end-of-job is encountered. Control must be returned to JOSEQ which can then load and enter the next job in the sequence. In the MULTIJOB System the FORTRAN STOP statement results in entry to the FORTRAN MAIN PROGRAM

INITIALISER (MPI) which itself terminates the user job with an EOJ macro call. It was therefore necessary to provide a new EOJ macro which would override the system version and which would return control to JOSEQ. This macro has to be compiled with all usercode jobs and also had to be compiled into a new version of MPI which must be linked with all FORTRAN programs. To ensure that user programs will run without modification under both JOSEQ and MULTIJOB the macro tests whether the current program is running under the control of JOSEQ or not. If not, then the program is terminated normally.

Errors in user jobs must also be trapped by JOSEQ and recovered from if possible so that the next job in the sequence may be run. Under MULTIJOB, errors result in entry to error (or STXIT) routines which may be provided by the user or which may be a part of the language debugging system. In the FORTRAN system, an entry is made into STXIT routines in MPI which in turn may call error traceback facilities. It was necessary to ensure that all error paths resulted in control returning to JOSEQ. Special action had to be taken to handle time out errors. In the standard MULTIJOB system the time requested for every job scheduled is stored and checked against the actual time being used by that job. An error interrupt occurs if the user job exceeds this time allocation. But when a user job is run under JOSEQ, the MULTIJOB system has no means of knowing the time limit for that particular job - it knows only of JOSEQ's time allocation which is set to a high value at the beginning of a session. Thus if a user program loops it will not be interrupted until it has consumed the whole of JOSEQ's time allocation. This problem had to be overcome by allowing JOSEQ to enter the user job's time limit in the MULTIJOB system tables so that a time out error would occur when it exceeded its time limit. Further modifications had to be made to allow several consecutive time out errors to occur because MULTIJOB normally allows only one time out error to occur in a user sequence. All these operations are dangerous (requiring as they do modifications to the supervisor tables) and must be

carried out in privileged mode. On return from a user job JOSEQ must reset its own remaining time allocation in the supervisor tables so that it can run on to the limit of its allowed time.

A second group of problems was caused by the interface between JOSEQ, the user program and the communication program BEPE. The remote computers activate JOSEQ via BEPE which maintains a record of all the copies of JOSEQ currently active in the system. Thus every time JOSEQ is started up it must set a flag in BEPE to indicate this fact and it must never be allowed to terminate without clearing this flag. Thus JOSEQ must never be abandoned unless steps are taken to ensure that the communication flags and the times in the supervisor's tables have been reset first. The implementation of this necessitated changes to MPI's STXIT routines to allow operator interrupts to return control to JOSEQ.

A third group of problems was caused by particular facilities in MULTIJOB which proved difficult to implement. This resulted in some restrictions having to be imposed on the user jobs which could be run under JOSEQ. For example, the implementation of segmented programs presented difficulties. When a segmented program is run under MULTIJOB, the system reads segments from the current program file and these segments are assigned addesses relative to the start of the program. When JOSEQ is running the MULTIJOB system takes the current program file to be the JOSEQ program file and not the user program file. If JOSEQ is to run segmented programs, therefore, it must overwrite its own file name in the overlay control module by the name of the current user program file. The simplest method of doing this involves the modification of MPI so that it overwrites the name at run time. This leads to the restriction that only segmented FORTRAN programs can be run under JOSEQ. Furthermore it is necessary to ensure that relative segment addresses remain correct by always loading user programs into the beginning of store from location Ø onwards as

shown in Figure 2. It can be seen from this that if user programs become too large, JOSEQ itself will be overwritten. In particular, the FORTRAN I/O buffers, which by default are dynamically allocated in the area of store between the end of the user program and the end of the stream, may overwrite JOSEQ if they become too large. It is advised therefore that a special system facility which allows the I/O buffers to be included within the user program itself be used. This involves the linking of a special DATAD module with the user program.

It can be seen from Figure 2 that restrictions have to be imposed on the size of user program which may be run with JOSEQ ie the size of the available stream minus the size of JOSEQ itself. This was chosen to be 130 K bytes in our installation.

Despite these problems it has been found that the large savings in efficiency provided by JOSEQ more than offset the minor difficulties and restrictions it imposes. There is no standard analysis program in use at Culham which cannot run under JOSEQ and these programs (once composed in the way required by JOSEQ) still run perfectly in the standard system without any changes.

CONCLUSIONS

When we first embarked on this project we knew very little about the details of the MULTIJOB system or how feasible the proposed program would prove to be. But we have shown that by expending a relatively small amount of effort (6 man months) and by asking a lot of questions it is possible to achieve priority treatment from the system and produce a six-fold improvement in turnaround time for urgent job sequences. JOSEQ is now one of the most frequently used programs within the laboratory. We have succeeded in producing an acceptable real time response from a system not specifically designed for

that purpose.

The original design objectives have been achieved with the following restrictions:

a) User programs must be recompiled with a special end-of-job macro or linked with a special version of MPI before they can be run under JOSEQ.

b) Segmented programs must be written in FORTRAN if they are to run under JOSEQ.

c) Programs may not be compiled and composed under JOSEQ since the MULTIJOB TRIALS system is a segmented usercode program.

d) User programs may only use disc files. Magnetic tapes may not be used.

ACKNOWLEDGEMENTS

We would like to thank many of our friends and colleagues at Culham who have contributed by discussion to the design of JOSEQ. In particular we would like to thank Dr M Calderbank and Miss S J Roberts for information on the MULTIJOB system and help with specific implementation problems, and Mr C A Steed and Dr E M Jones for assistance with the running and debugging of JOSEQ in actual real life situations.

REFERENCES

1. C A Steed, E G Murphy, "Automated Data Acquisition on a Large Fusion Experiment", Inst.Phys.Conf. on On-Line Computers in Laboratory Use, London, September 1975. International Tracts in Computer Studies, Advanced Publications Ltd 1976.

2. The MULTIJOB Programming Manual, International Computers Limited.

3. M Calderbank. "The Use of a Front End Computer to Increase the Availability of a Central Computer Facility". The European Computing Conf. on Software Systems Engineering, London, Sept. 1976.

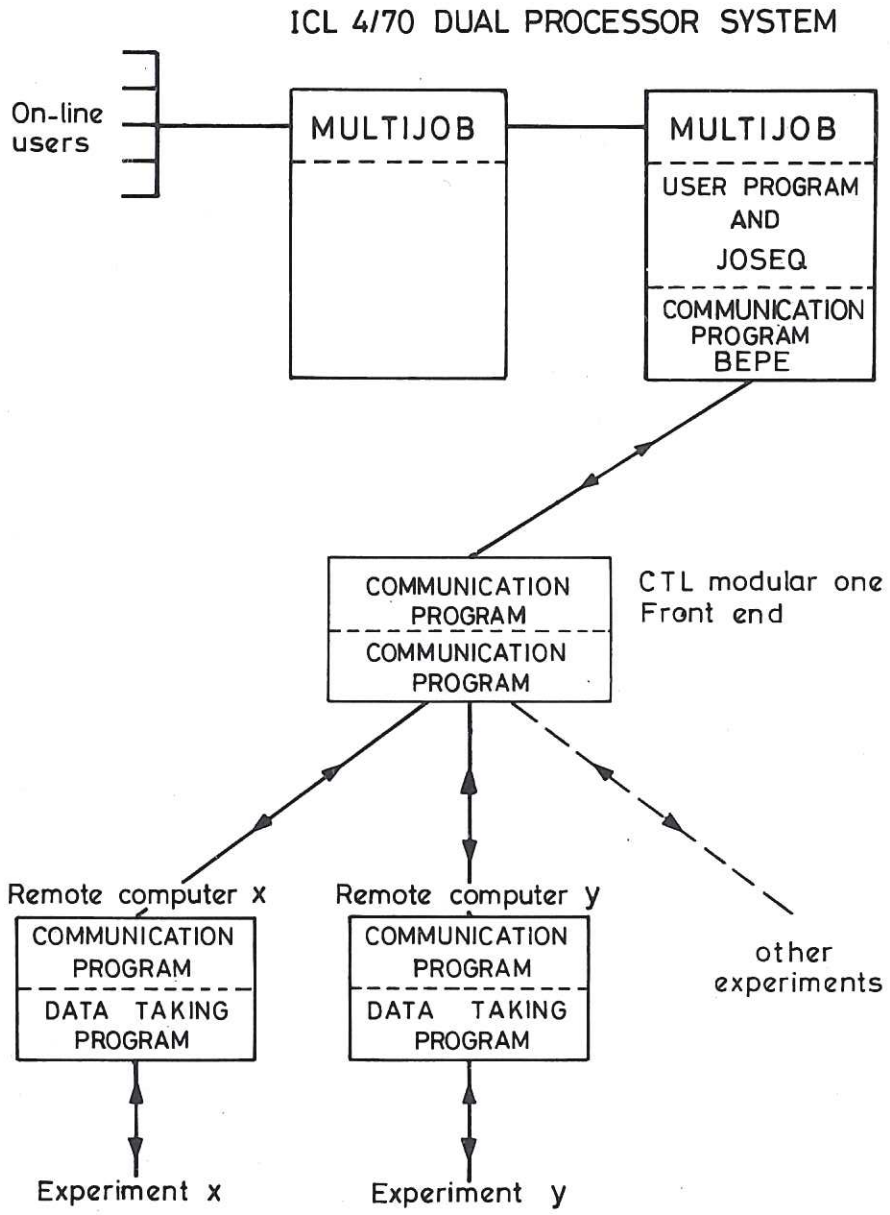

Fig.1 A Schematic Diagram of the Environment for JOSEQ.

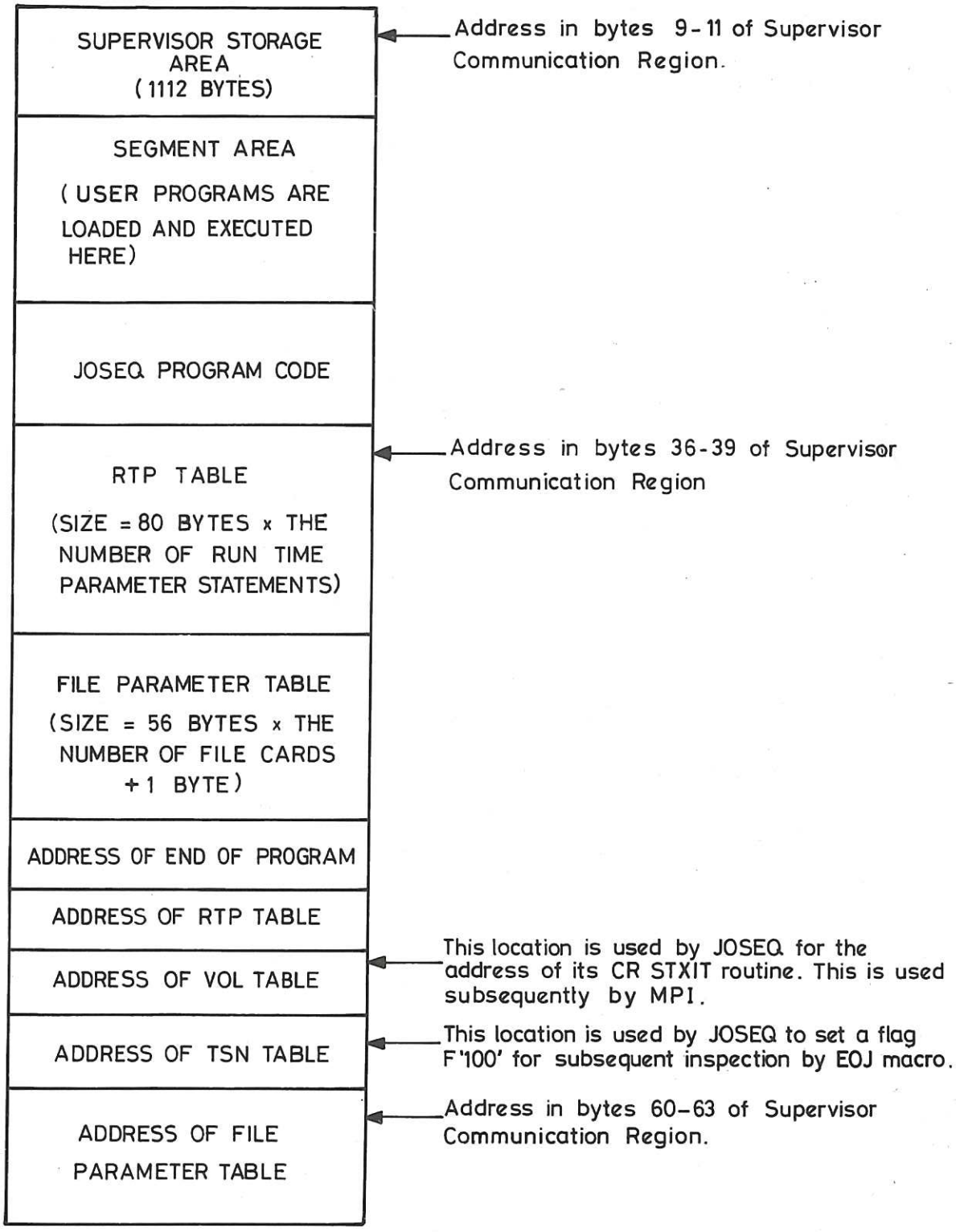

Fig.2 Core Layout for JOSEQ.