REAL TIME MINICOMPUTER SIMULATION

Jacob Rootenberg
Systems Research Group
Computer Science Department
Queens College, Flushing, N.Y. 11367

and

Lester Tannenbaum
E.D.O. Corporation
College Point, N.Y. 11356

ABSTRACT

This paper describes a simulation of a real time minicomputer operating system using the GPSS language. Different priority devices and priority tasks are simultaneously run on the minicomputer operating system. The aim of the simulation is to identify the devices that bottleneck the system and how many tasks can be efficiently run on the system at the same time.

Another objective of this paper is to acquaint the minicomputer user with using the simulation tool to model his own hardware configuration using an existing operating system as well as a help in developing his own minicomputer operating system.

The conclusions of the simulation reported in this paper show that the efficiency of the operating system is highly dependent on the speed of the peripheral devices. Program documentation and detailed GPSS block diagrams as well as final results are given in the paper.

Introduction

The comparatively low cost of todays minicomputer systems has made them very attractive to a wide spectrum of users and applications. The technology of large scale integration and the improvement in minicomputer architecture has enabled these systems to effectively handle jobs customarily run on large scale computers. It is extremly important for any potential user to fully understand the operating systems of such minicomputers. This paper will focus on this problem via a GPSS simulation.

Users of various minicomputer operating systems may need to know the utilization of the CPU and peripheral devices. The user may have to cost justify the need for these devices. The minicomputer programmer may want to determine which devices called by the program are bottlenecking the system or what peripherals, if any, are seldom used. A simple solution for finding the effectiveness for any proposed system would be to simulate the system. This paper will do so.

Most minicomputer operating systems are based on priority ranking, where one program or task has priority over other programs in its class as well as between classes. Device servicing is done by priority interrupt, where high speed peripheral devices are assigned higher priority over slower devices. In a real time environment, will programs having a low priority be executed and what is their percentage of processing time? These questions will be answered by the simulation. If a device is bottlenecking the performance of the system, the simulation will detect it and, the user can try other strategies without endangering the system itself.

Program Description

In this paper no specific application program will be simulated. A "generalized program" which will indicate the logic and structure that go into a minicomputer operating system will be used in the simulation. Each program handled by the operating system is structured, having a processing period and an input/output request to a device. The time period and device selection will be random.

The inner workings of the operating system will not be simulated. A skeleton operating system which will correctly allocate computer resources will be used. To those readers familiar with minicomputer operating systems, the method of allocating resources in the GPSS block diagram is similar to minicomputer operating system allocation of resources.

The time unit in this application will be milliseconds but the user can tailor the time unit to his specific needs. If another application has a complexity that requires microsecond operations the simulation can handle this easily.

The hardware devices will be common types found in most minicomputer systems. They are teletype, card reader, line printer, magnetic tape, and disk. Peripherals can be added and deleted from the system as required by the application.

The GPSS priority structure is numbered from 0 to 127. The larger the number, the higher the priority. The devices will have a higher priority then the programs, since hardware must be serviced before program software. Higher speed devices such as the disk will have the higher priority.

Hardware Considerations

The system (Figure 1) will consist of a minicomputer (CPU will be device 1 and real time clock) and 5 peripheral devices, i.e.:

Device 2	Teletype
Device 3	Card reader
Device 4	Line Printer
Device 5	Magnetic Tape
Device 6	Disk

These devices will be given different priorities and different spooling times per character.

Device	Time	Priority
2	30 milliseconds	21
3	12 "	22
4	4 "	23
5	1000 "/4000 chars	24
6	100 "/4000 chars	25

The real time clock will tick every 100 milliseconds at priority 30.

Software Considerations

User Programs

The programs will be given priorities from 0 to n, where 1 LnL14. The programs will use the CPU from 5 to 205 milliseconds per processing period. The time period is chosen by a GPSS cariable "TIME".

Twenty five percent of the programs will wait for a clock interrupt which arrives every 100 milliseconds. The programs which are to wait for the clock are determined by a GPSS transfer block. Only one clock interrupt service request may enter and pass through the clock interrupt and execution blocks at a time. Other clock service requests are put on a "user chain" in a FIFO discipline. When a clock pulse is received, the program will process for 10 more milliseconds on a priority basis. The programs then wait for a free input/output buffer. The program must wait for a system buffer (5 to 15 buffers will be defined in this simulation) before notifying the system driver program. If a buffer is available the program sends message character length, sender's program priority, and type of device to the system driver program. Message character length is determined by the GPSS variable "CHAR" and device type by a GPSS function (FN1). The sending program waits for I/O complete before resuming processing.

System Driver Program

The system driver program controls all I/O in the minicomputer. A buffer is assigned in the system for each I/O request. As many requests as system buffer space allows will be honored, others will wait for a buffer to become free.

The system driver receives information from each sender program on device type, character length, and sending program priority (Method used is given in the User Program - Driver Program communication section).

The device driver is then assigned the priority and time per character relative to that device (hardware function) by a GPSS functions (FN2,FN3). If the device is busy, it waits in the device queue.

When a device is available, it notifies the sending program that its request is being honored. It then spools to the device by using the CPU for 1 millisecond for each character and uses the device for the assigned time per character. When the last character is spooled, the system driver program notifies the sending program that the I/O is complete and the program releases the device and buffer.

Intercommunication Between Sending and Device Programs

When the user program has an input/output operation, it notifies the driver program (assuming a buffer is available) via a GPSS logic switch "Out". With this occurrence, no other user program may request a system driver program until savevalues for device type-"Kind", character length-Charn", and user priority-"Prio" are sent to the driver program by these GPSS savevalues.

When the actual operation (Savevalue "PRT") or I/O completion (Savevalue "PRII"), takes place, the driver program notifies the user program by sending the sender program's priority back to it by one of the above mentioned Savevalues. Only one notification may occur at a time, this is accomplished by logic switches "MES" and "MES1". The notification process occurrs in zero simulation time.

Clock Driver Program

A clock pulse is generated every 100 milliseconds at priority 30. The interrupt program uses the CPU for 4 milliseconds and sends a message to the awaiting program (if any).

System Operation

The complete system, programs and device buffers are generated at time zero and operate in the manner given by Software Considerations and Figure 2. (For detail, see Block Diagram).

Results and Conclusions

The program was run for two different sets of results. Type 1 was constant buffer size versus variable number of programs. Ten buffer maximum was the constant buffer size versus ten, twelve, and fifteen programs.

Type 2 was constant number of programs versus variable buffer size. Ten programs was the constant program number versus four, seven, and ten programs. An example run, including program listing and pertinent GPSS results, for ten programs and ten buffers is given in Figure 4.

Type 1 Results

In facility usage fifteen programs maximized the use of all facilities. This is shown in Table 1. The CPU was used over 50% of the time for example. For the most part, the usage of the facilities doesn't change a great deal. This shows that the system is device dependent rather than dependent on the number of programs. Notice that the teletype is used about 98% of the time in all three cases. The queue for the teletype is also quite backed up. Average number of transactions waiting for the TTY was about seven for ten programs. Output should be limited to the teletype (i.e. ten per cent of the programs instead of twenty per cent) to produce a more efficient system. In this system, programs are queued waiting to use the teletype.

In program priority, the highest priority program was processed the most by a considerable amount of times. The priorities below it were processed fairly evenly. Again program processing depended on program device selection. As programs increased the priorities at the bottom were processed less frequently. This is evident in the fifteen program run. Priorities 0,1,2 were run only 4,8, and 9 times respectively.

Type 2 Results

In facility usage (Table 2) ten buffers maximized usage while four buffers exhibited the worst usage. As buffers decreased the priorities at the bottom were processed less frequently. In the four buffer program, priorities 0 and 1 were processed only 4 and 8 times. Results were similar to those in Type 1 in system structure.

Simulation Conclusions

This system is dependent on the speed of the peripheral devices. The faster the device the more efficiently the programs will run. If the teletype was eliminated or assigned a lesser probability of being selected the system would be more efficient.

Final Remarks

The test operating system can be made more complex. The simulation program can be made to handle many programs with the same priority. To accomplish this, more attributes would have to be passed to the system driver program and a Round Robin method of processing programs of equal priority would have to be developed.

The system driver programs can also be modified to allocate buffers only when a system device is free.

Many different types of experiments can be devised for minicomputer operating systems using simulation as a tool.

REFERENCES

Schriber, Thomas J. Simulation using GPSS. John Wiley & Sons, 1974, 533 pp.

User's Manual, Introduction to the Real Time Disc Operating System. Data General Corp., 1974

Table 1
10 Buffers Constant versus Variable Programs
Facility Usage

Programs / Facility	10	12	15
CPU	.498	.486	.527
TTY	.978	.979	.978
Card reader	.468	.434	.572
Line printer	.237	.239	.231
Magnetic tape	.507	.477	.541
Disk	.071	.071	.075

Table 2
10 Programs Constant versus Variable Buffers
Facility Usage

Buffers / Facility	4	7	10
CPU	.466	.494	.498
TTY	.948	.978	.978
Card reader	.517	.464	.468
Line printer	.204	.241	.237
Magnetic Tape	.450	.492	.507
Disk	.065	.071	.071

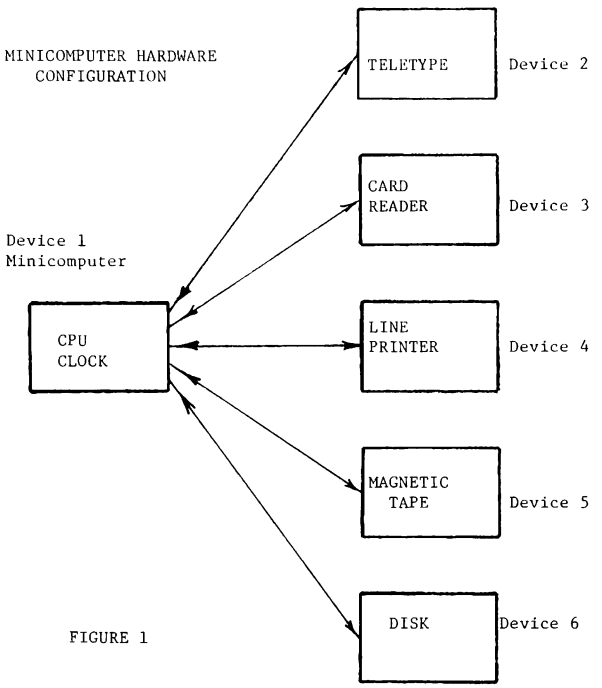
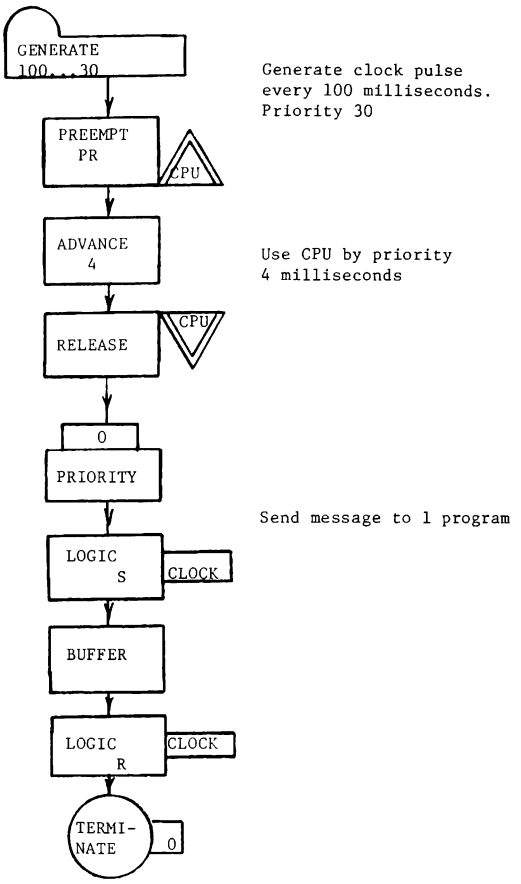


FIGURE 1

GPSS BLOCK DIAGRAM
CLOCK PULSE PROGRAM



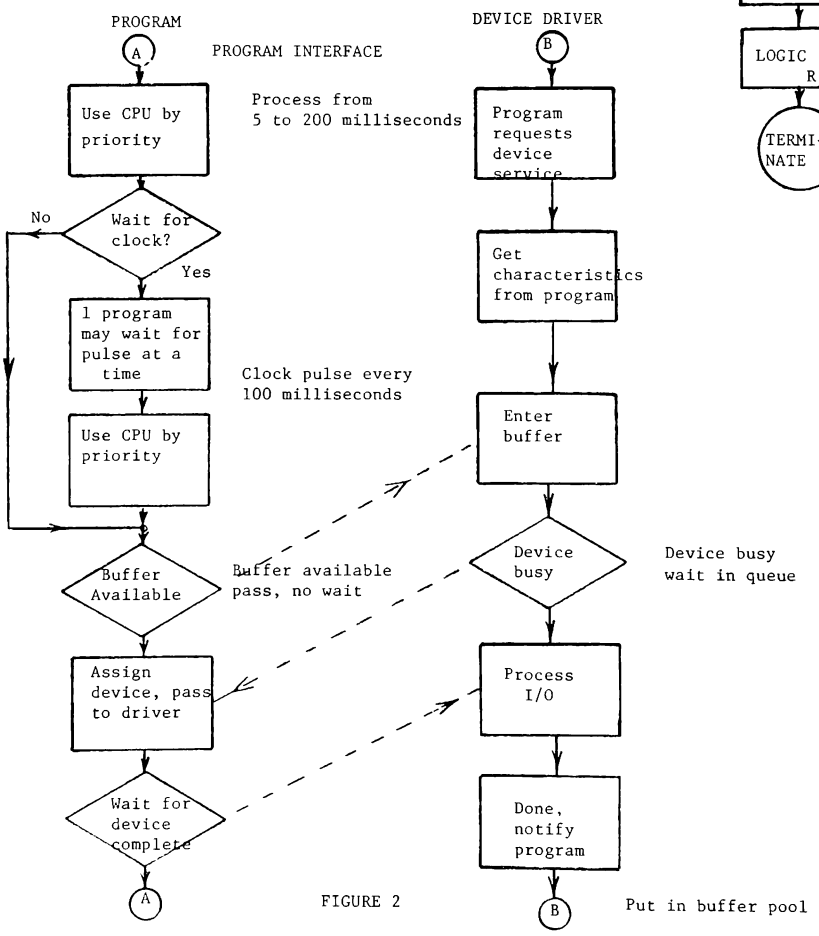
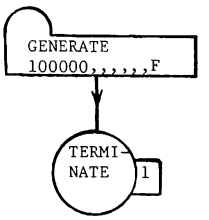
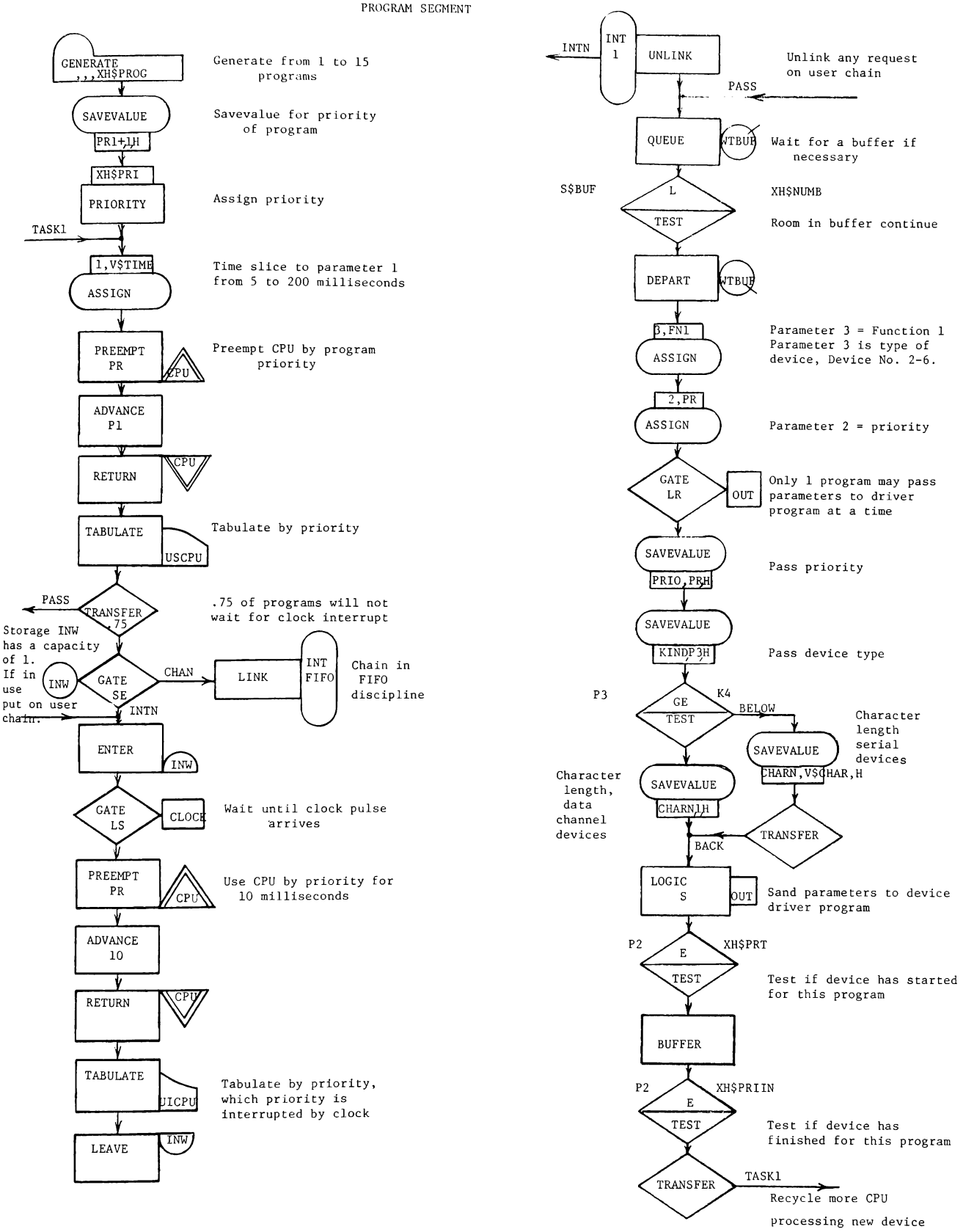


FIGURE 2

GPSS BLOCK DIAGRAM
TIMER



GPSS BLOCK DIAGRAM
PROGRAM SEGMENT



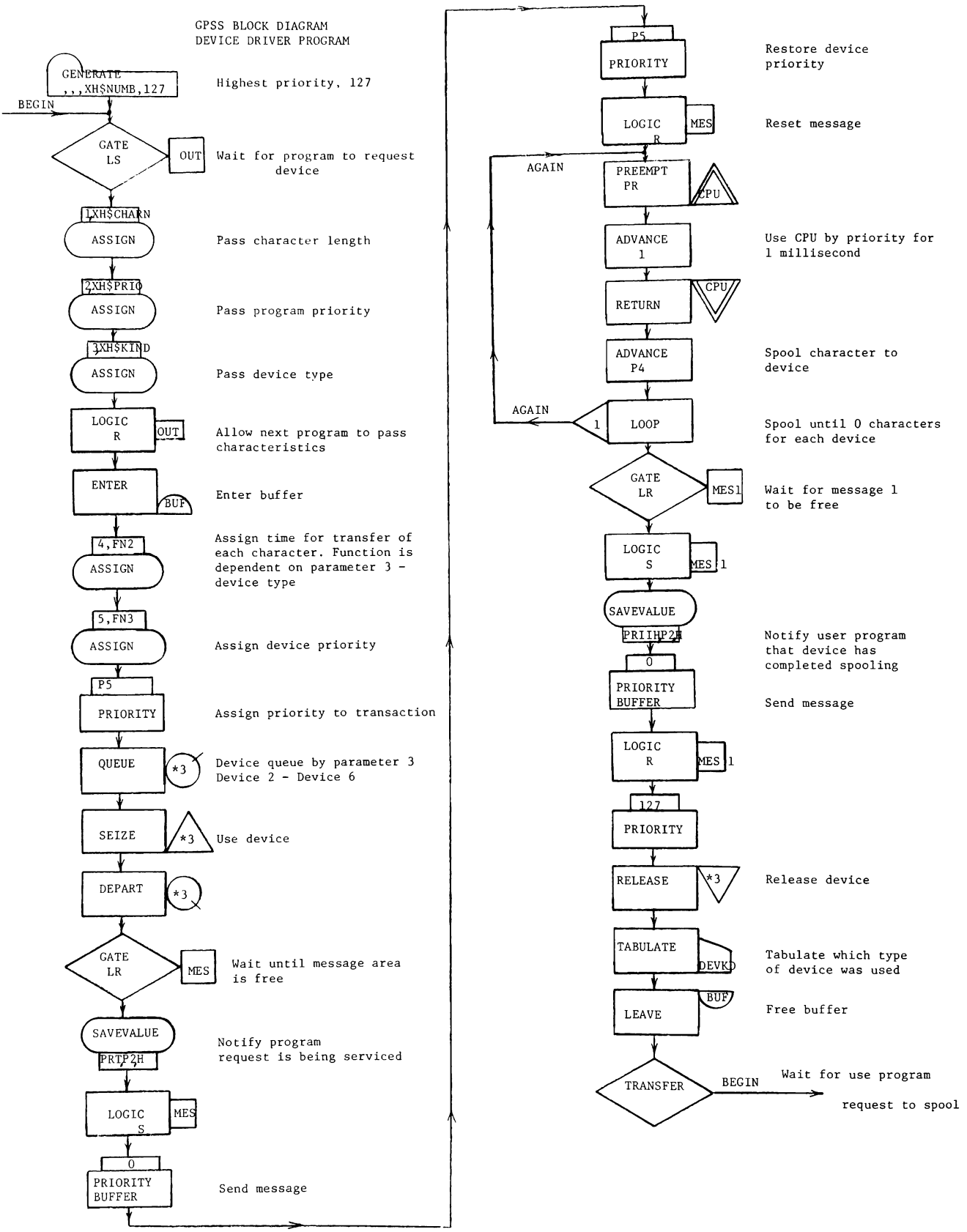

GPSS BLOCK DIAGRAM
DEVICE DRIVER PROGRAM
GENERATE
,,,XH$NUMB,127
Highest priority, 127
BEGIN
GATE
LS
OUT
Wait for program to request device
1,XH$CHARN
ASSIGN
Pass character length
2,XH$PRIO
ASSIGN
Pass program priority
3,XH$KIND
ASSIGN
Pass device type
LOGIC
R
OUT
Allow next program to pass characteristics
ENTER
BUF
Enter buffer
4,FN2
ASSIGN
Assign time for transfer of each character. Function is dependent on parameter 3 - device type
5,FN3
ASSIGN
Assign device priority
P5
PRIORITY
Assign priority to transaction
QUEUE
*3
Device queue by parameter 3 Device 2 - Device 6
SEIZE
*3
Use device
DEPART
*3
GATE
LR
MES
Wait until message area is free
SAVEVALUE
PRTP2H
Notify program request is being serviced
LOGIC
S
MES
0
PRIORITY
BUFFER
Send message
P5
PRIORITY
Restore device priority
LOGIC
R
MES
Reset message
AGAIN
PREEMPT
PR
CPU
ADVANCE
1
Use CPU by priority for 1 millisecond
RETURN
CPU
ADVANCE
P4
Spool character to device
AGAIN
1
LOOP
Spool until 0 characters for each device
GATE
LR
MES1
Wait for message 1 to be free
LOGIC
S
MES
1
SAVEVALUE
PRIIHP2H
Notify user program that device has completed spooling
0
PRIORITY
BUFFER
Send message
LOGIC
R
MES
1
127
PRIORITY
RELEASE
*3
Release device
TABULATE
DEVKD
Tabulate which type of device was used
LEAVE
BUF
Free buffer
TRANSFER
BEGIN
Wait for use program request to spool

FIGURE 4

```
*LOC   OPERATION  A,B,C,D,E,F,G                    COMMENTS

       SIMULATE
*      VARIABLES
       INITIAL    XH$PRT,128/XH$PRIO,128/XH$CHARN,0/XH$PRI,=1
       INITIAL    XH$PRIIN,128/XH$NUMB,10/XH$PKOG,10
CHAR   VARIABLE   RN1*120/1000+1
TIME   VARIABLE   RN3*200/1000+5
1      FUNCTION   RN2,D5
.2,2/.4,3/.6,4/.8,5/1,6
2      FUNCTION   P3,D5
2,30/3,12/4,4/5,1000/6,100
3      FUNCTION   P3,D5
2,21/3,22/4,23/5,24/6,25
INW    STORAGE    1
BUF    STORAGE    10
UICPU  TABLE      PR,0,1,11
USCPU  TABLE      PR,0,1,11
DEVKD  TABLE      P3,2,1,6
       GENERATE   ,,,XH$PROG
       SAVEVALUE  PRI+,1,H     INCREMENT SAVEVALUE FOR PRIO ITY
       PRIORITY   XH$PRI       ASSIGN PRIOTITY
TASK1  ASSIGH     1,V$TIME     ASSIGN TIME
       PREEMPT    CPU,PR
       ADVANCE    P1           USE CPU
       RETURN     CPU
       TABULATE   USCPU
       TRANSFER   .75,,PASS
       GATE SE    INW,CHAN     FACILITY EMPTY
INTV   ENTER      INW          ENTER WAIT FOR CLOCK
       GATE LS    CLOCK        WAIT FOR CLOCK PULSE
       PREEMPT    CPU,PR       GET CPU AGAIN
       ADVANCE    10           USE CPU FOR 10 MILLISECONDS
       RETURN     CPU

       TABULATE   UICPU
       LEAVE      INW
       UNLINK     INT,INTV,1           UNLINK IF ANY
PASS   QUEUE      WTBUF                WAIT FOR BUFFER TO FREE
       TEST LE    S$BUF,XH$NUMB        ENTRIES LESS THEN 10
       DEPART     WTBUF
       ASSIGN     3,FN1                ASSIGN TYPE OF FUNCTION
       ASSIGN     2,PR                 ASSIGN PRIO
       GATE LR    BUT                  WAIT UNTIL FREE
       SAVEVALUE  PRIO,PR,H            SAVE AND PASS PRIORITY
       SAVEVALUE  KIND,P3,H            TRANSFER TYPE
       TEST GE    P3,K5,BELOW          P9 .GT. 4YES
       SAVEVALUE  CHARN,1,H            HIGH SPPED 1=4000 WORDS
BACK   LOGIC S    BUT                  SET LOGIC BUT
       TEST E     P2,XH$PRT
       BUFFER
       TEST E     P2,XH$PRIIN          WAIT FOR PROPER PRIORITY
       TRANSFER   ,TAS<1
       TERMINATE
*
CHAN   LINK       INT,FIFO             LINK ON USER CHAIN
BELOW  SAVEVALUE  CHARN,V$CHAR,H       GEET 38A-ACTER LENGTH
       TRANSFER   ,BACK
*
```

```
*LOC   OPERATION  A,B,C,D,E,F,G                    COMMENTS

*      DEVICE DRIVERS
       GENERATE   ,,,XH$NUMB,127
BEGIN  GATE LS    BUT                  TASK WANTS TO TYPE
       ASSIGN     1,XH$CHARN           PASS CHARACTER LENGTH
       ASSIGN     2,XH$PRIO            ASSIGN PRIO
       ASSIGN     3,XH$KIND            ASSIGN TYPE OF DEVICE
       LOGIC R    BUT                  RESET BUT
       ENTER      BUF                  ENTER BUFFER AREA
       ASSIGN     4,FN2                ASSIGN CHAR TRANSFER TIME
       ASSIGN     5,FN3                ASSIGN PRIORITY OF DEVICE
       PRIORITY   P5
       QUEUE      *3                   ENTER QUEUE GIVEN BY P3
       SEIZE      *3                   SEIZE FACILITY
       DEPART     *3                   LEAVE QUEJ
       GATE LR    MES                  TEST IF MESSAGE ALREADY SENT
       SAVEVALUE  PRT,P2,H
       LOGIC S    MES                  MEASSAGE IN USE
       PRIORITY   0,BUFFER
       PRIORITY   P5
       LOGIC R    MES                  RELEASE MES
AGAIN  PREEMPT    CPU,PR               SEIZE CPU IF AVAILABLE
       ADVANCE    1
       RETURN     CPU
       ADVANCE    P4                   TIME TO SEND CHARACTER
       LOOP       1,AGAIN              REPEAT UNTIL NO CHARACTERS
       GATE LR    MES1                 MES1 NOT IN USE PROCEES
       LOGIC S    MES1
       SAVEVALUE  PRIIN,P2,H
       PRIORITY   0,BUFFER
       LOGIC R    MES1
       PRIORITY   127                  GIVE HIGHEST PRIORITY
       RELEASE    *3                   RELEASE FACILITY
       TABULATE   DEVKD
       LEAVE      BUF                  LEAVE BUFFER
       TRANSFER   ,BEGIN
       TERMINATE

*      CLOCK TANSFER

       GENERATE   100,,,,30            GENERATE CLOCK PULSE EVERT 1 0 MI
       PREEMPT    CPU,PR               ADVANCE
       ADVANCE    4
       RETURN     CPU
       PRIORITY   0                    CHANGE PRIORITY TO 0
       LOGIC S    CLOCK                SEND MEASSGE TO WAITING TASK
       BUFFER
       LOGIC R    CLOCK                RESET LOGIC
       TERMINATE  0
       GENERATE   100000,,,,,,F
       TERMINATE  1
       START      1
       CLEAR
       INITIAL    XH$PRT,128/XH$PRIO,128/XH$CHARN,0/XH$PRI,=1
       INITIAL    XH$PRIIN,128/XH$NUMB,10/XH$PKOG,12
       START      1
       CLEAR
       INITIAL    XH$PRT,128/XH$PRIO,128/XH$CHARN,0/XH$PRI,=1
       INITIAL    XH$PRIIN,128/XH$NUMB,10/XH$PKOG,15
       START      1
       END
```

STORAGE	CAPACITY	AVERAGE CONTENTS	AVERAGE UTILIZATION	TOTAL ENTRIES	AVERAGE TIME/TRANS	CURRENT CONTENTS	MAXIMUM CONTENTS
INW	1	0.093	0.093	87	107.057	0	1
BUF	10	9.682	0.968	313	3093.437	10	10

FACILITY	AVERAGE UTILIZATION	NUMBER ENTRIES	AVERAGE TIME/TRAN	SEIZING TRANS. NO.	PREEMPTING TRANS. NO.
CPU	0.498	12918	3.856	0	23
2	0.978	51	1918.744	3	0
3	0.465	61	763.294	16	0
4	0.237	72	329.444	0	0
5	0.507	51	995.725	17	0
6	0.071	71	101.056	0	0

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	TOTAL ZERO ENTRIES	% ZERO ENTRIES	AVERAGE TIME/TRANS	AV EX-ZERO TIME/TRANS
ATBUF	1	0.000	314	314	100.0	0.000	0
2	10	6.926	57	13	22.8	12151.558	15741.792
3	3	0.146	62	43	69.3	236.096	770.420
4	2	0.018	72	61	84.7	25.111	164.363
5	3	0.329	51	32	62.7	646.431	1735.157
6	1	0.000	71	69	97.1	1.394	49.500

QTABLE NUMBER	CURRENT CONTENTS
0	0
0	6
0	1
0	0
0	0
0	0

TABLE DEVKO

ENTRIES IN TABLE	MEAN ARGUMENT	STD. DEVIATION	SUM OF ARGUMENTS	
303	4.105	1.398	1244.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
2	50	16.50	16.5	83.4	0.487	-1.505
3	60	19.80	36.3	63.6	0.730	-0.790
4	72	23.76	60.0	39.9	0.974	-0.075
5	50	16.50	76.5	23.4	1.217	0.639
6	71	23.43	100.0	0.0	1.461	1.354

REMAINING FREQUENCIES ARE ALL ZERO

TABLE QICPU

ENTRIES IN TABLE	MEAN ARGUMENT	STD. DEVIATION	SUM OF ARGUMENTS	
87	6.252	2.910	544.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
0	2	2.29	2.2	97.7	0.000	-2.148
1	8	9.19	11.4	88.5	0.159	-1.804
2	3	3.44	14.9	85.0	0.319	-1.461
3	7	8.04	22.9	77.0	0.479	-1.117
4	5	5.74	28.7	71.2	0.639	-0.774
5	5	5.74	34.4	65.5	0.799	-0.430
6	7	8.04	42.5	57.4	0.959	-0.086
7	8	9.19	51.7	48.2	1.119	0.256
8	12	13.79	65.5	34.4	1.279	0.600
9	30	34.48	100.0	0.0	1.439	0.943

REMAINING FREQUENCIES ARE ALL ZERO

TABLE QSCPU

ENTRIES IN TABLE	MEAN ARGUMENT	STD. DEVIATION	SUM OF ARGUMENTS	
314	5.722	2.976	1797.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PER CENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
0	15	4.77	4.7	95.2	0.000	-1.922
1	28	8.91	13.6	86.3	0.174	-1.586
2	15	4.77	18.4	81.5	0.349	-1.250
3	24	7.64	26.1	73.8	0.524	-0.914
4	28	8.91	35.0	64.9	0.698	-0.578
5	29	9.23	44.2	55.7	0.873	-0.242
6	26	8.28	52.5	47.4	1.048	0.093
7	27	8.59	61.1	38.8	1.223	0.429
8	33	10.50	71.6	28.3	1.397	0.765
9	89	28.34	100.0	0.0	1.572	1.101

REMAINING FREQUENCIES ARE ALL ZERO