A GENERALIZED REAL-TIME EXECUTIVE ROUTINE FOR THE UNIVAC 1230 C-ETC(U)
M AR 70  D POTTER,  G BOTSEAS,  C BECKER
UNCLASSIFIED
NUSL-TH-2211-63-70
A GENERALIZED REAL-TIME EXECUTIVE ROUTINE FOR THE UNIVAC 1230 COMPUTER

by

David/Potter, George/Botseas and Clair/Becker

NUSL Technical Memorandum No. 2211-63-70

INTRODUCTION

Use of a computer to perform repetitive real-time tasks such as equipment control, monitoring, data gathering and data reduction requires an executive routine which will ensure that the proper actions are initiated at the appropriate times.

This memorandum describes GPEXEC1, a generalized real-time executive routine for the Univac 1230 computer. Written in assembly language, it can be assembled and run on other Univac computers which are similar to the 1230.

Included is a flow chart, a sample timing chart, a program listing and a timing test routine.

GENERAL DESCRIPTION

GPEXEC1 is a program which calls on a sequence of routines in a predetermined order. The time at which each routine in the sequence is called is determined by preset table entries and the computer's real-time clock.

THE REAL-TIME CLOCK

The real-time clock is, for programming purposes, a memory cell whose contents are increased by one every 1024th of a second. Since the clock will overflow into the sign bit and turn negative after approximately six days, it is necessary to reset it periodically. Therefore, a check on the clock is made prior to the execution of each
routine. If a reset is necessary, a flag is set and the clock is cleared following the execution of the upcoming routine.

CAPABILITIES OF GPEXEC1

GPEXEC1 is capable of repeatedly executing a sequence of N routines in a specified order. The time of execution for each routine, relative to the start time of the exec, is specified by the user. Also specified by the user is REPTIME, the time interval between successive executions of the sequence of routines.

It is assumed that each routine will be completed before it is time to execute the next. However, should a routine fail to return control to the exec prior to the execution time of the next, the next routine will be executed immediately upon return of control. The manner in which the timing is handled will ignore this tardiness and the routine will be scheduled for its next execution at the proper time.

The time of execution of any routine may be varied relative to the preceding and following routines. Also, the entire sequence may be easily shifted forward or backward in time.

GPEXEC1 can be set up so that one or more routines are executed only occasionally. A routine can also be deleted entirely from the sequence of events. These capabilities will be discussed in detail later in this memorandum.

SETTING UP GPEXEC1 FOR A SPECIFIC TASK

To set up GPEXEC1 for a specific task, it is necessary to supply entries to two tables and to set two parameters.

The first parameter is the number punched on the card labeled EXEC 05 (see program listing). N should be set to the number of routines which the exec will control.

The second parameter is the number punched on the REPTIME card, labeled EXEC 54. REPTIME should be set to the number of clock cycles from the start of one sequence to the next.

The first table to be filled is the INITLTIME table, which must contain N entries. The jth entry is the number of clock cycles after the start of the exec that the jth routine is to be executed.
The other required table is called JUMPTABLE. It also must contain N entries, each of which must be a return jump (RJP) to the desired routine. Each entry in the JUMPTABLE may be coded with a keyset condition.

THEORY OF OPERATION

Once initiated, the exec transfers the contents of INITLTIME to a working area, TIMETABLE, and waits for a command to start. When this is received, the real-time clock is zeroed and a comparison is made between the clock and the first entry in TIMETABLE. When the clock reaches or exceeds this value, the table value is increased by REPTIME and the new value is checked for overflow. The corresponding routine in JUMPTABLE is executed, provided that the keyset condition is met or the entry has not been cleared.

If the overflow test indicated that a reset of the clock was necessary, the clock reading is subtracted from all entries in TIMETABLE and the clock is zeroed.

A check is then made to determine whether the program should be terminated. If so, the exec exits. If not, the routine index is incremented by one and the exec awaits the execution time of the next routine. Following the execution of the last routine in the sequence, the routine index is cleared so that the exec now waits for the proper time to re-execute the first routine in the sequence.

METHODS OF DELETING A ROUTINE

There are two ways in which a routine can be deleted. The most flexible is by including a keyset condition in the coding of the JUMPTABLE. Thus the execution would be deleted if the keyset condition were not met.

Another method is to simply zero out the desired cell in JUMPTABLE. The exec performs a test on the entries prior to any attempted execution and aborts if a zero is found.

CHANGING THE TIMING

The execution time of any or all routines may be advanced or retarded by increasing or decreasing the contents of the associated TIMETABLE cell(s) by the desired number of clock cycles.
GENERAL COMMENTS

The enclosed program listing is coded so that the exec may be started and terminated by either a key setting or the setting of a memory cell. The memory cell flag seems to be preferable since it can be set by interrupt from a keyboard or an external piece of equipment.

David Potter
DAVID POTTER
Mathematician

George Botseas
GEORGE BOTSEAS
Computer Specialist

Clair Becker
CLAIR BECKER
Mathematician
| TEST 01 | GPEXCL | PROGRAM*USN/USL |
| TEST 02 | GPEXCL | COMMENT*GENERAL PURPOSE REAL-TIME EXECUTIVE |
| TEST 03 | GPEXCL | COMMENT*ROUTINE FOR UNIVAC 1230 COMPUTER |
| TEST 04 | GPEXCL | ENTRY |
| TEST 05 | GPEXCL | STR#UW*(STARTEND)*INITIALIZE CONTROL FLAG |
| TEST 06 | GPEXCL | EN#7*N-1*INITIALIZE |
| TEST 07 | GPEXCL | ENTRY*#
A*#INITIALTIME#U7*#TIMETABLE |
| TEST 08 | GPEXCL | STR#N*(TIMETABLE#U7) |
| TEST 09 | GPEXCL | WIP#D#EXEC1 |
| TEST 10 | GPEXCL | COMMENT*INSERT ADDITIONAL INITIALIZATION |
| TEST 11 | GPEXCL | COMMENT*ROUTINES HERE |
| TEST 12 | GPEXCL | EN#T#*(STARTEND)*ANOT#TIME TO START |
| TEST 13 | GPEXCL | JP*EXEC2*KEY1*NO WAIT |
| TEST 14 | GPEXCL | STR#UW*(CLOCK)*YES CLEAR CLOCK |
| TEST 15 | GPEXCL | CLR#1*SUBROUTINE INDEX |
| TEST 16 | GPEXCL | EN#T#*(TIMETABLE#U1)*EXECUTION TIME |
| TEST 17 | GPEXCL | EN#T#*(CLOCK)*APOST#TIME TO EXECUTE |
| TEST 18 | GPEXCL | JP*EXEC4*NO WAIT |
| TEST 19 | GPEXCL | EN#T#*(REPTIME)*YES |
| TEST 20 | GPEXCL | REP#T#*(TIMETABLE#U1)*UPDATE TABLE |
| TEST 21 | GPEXCL | STR#UW*(CLOCKFLAG)*CLEAR RESET FLAG |
| TEST 22 | GPEXCL | LSH#A*LPOS*TEST FOR CLOCK RESET |
| TEST 23 | GPEXCL | J#EXEC2#(CLOCKFLAG)*RESET NEEDED |
| TEST 24 | GPEXCL | STR#1#L*(EXEC3)*SAVE INDEX |
| TEST 25 | GPEXCL | EN#T#*(JUMPTABLE#U1) |
| TEST 26 | GPEXCL | STR#*(EXEC3)*ZERO ROUTINE DELETED |
EXEC 31 EXEC0 0'NO EXECUTE IT
EXEC 32 EXEC0 ENT*L1*XO'RESTORE INDEX
EXEC 33 EXEC0 ENT*AX*(CLOCKFLAG)RESET CLOCK
EXEC 34 EXEC0 JP*EXEC7*AZERO*NO
EXEC 35 EXEC0 ENT*G*X*(CLOCK)*YES
EXEC 36 EXEC0 RPT*N*ADV*ADJUST
EXEC 37 EXEC0 RPL*Y*U*(TIMETABLE)*TABLE
EXEC 38 EXEC0 STR*U*X*(CLOCK)*CLEAR CLOCK
EXEC 39 EXEC7 BSK*BUL*(STARTEND)*TERMINATE PROGRAM
EXEC 40 EXEC0 EXIT*YES
EXEC 41 EXEC0 EXIT*KEY2*YES IF KEY2 SET
EXEC 42 EXEC0 BSK*BN*Ian-1*NO INCREMENT INDEX
EXEC 43 EXEC0 NO-OP
EXEC 44 EXEC0 JP*EXEC3
EXEC 45 INITTIME 20480*TT1 - 2 SECONDS
EXEC 46 INITTIME 128000*TT2 - 12.5 SECONDS
EXEC 47 INITTIME 256560*TT3 - 25.25 SECONDS
EXEC 48 INITTIME 409320*TT4 - 39.675 SECONDS
EXEC 49 TIMETABLE RESERVE*INUMBER OF ROUTINES
EXEC 50 JUMENTABLE JP*KH1
EXEC 51 JUMENTABLE JP*K2*KEY3
EXEC 52 JUMENTABLE JP*K3
EXEC 53 JUMENTABLE JP*K4
EXEC 54 KEPTIME 1024000*TIME BETWEEN RECYCLES 100 SECONDS HERE
EXEC 55 CLOCKFLAG 0*CLOCK RESET FLAG
EXEC 56 STARTEND 0*UPPER START1LOWER END
END=DATA
APPENDIX B
TIMING AND FLOW CHARTS
FIGURE 1 - DETAILED FLOWCHART OF GPEEXECI
NOTES

1. THE ENTIRE SEQUENCE OF SUBROUTINES MAY BE SHIFTED EARLIER/LATER IN TIME BY DECREMENTING/INCREMENTING ALL ITEMS IN TIMETABLE BY $\Delta t$.

2. INDIVIDUAL SUBROUTINES MAY BE SHIFTED EARLIER/LATER IN TIME BY DECREMENTING/INCREMENTING THE CORRESPONDING ITEM IN TIMETABLE BY $\Delta t$.

3. SUBROUTINE R2 WILL BE EXECUTED ONLY IF KEY 3 IS SET.

4. INDIVIDUAL SUBROUTINES MAY BE DELETED BY STORING A ZERO IN THE CORRESPONDING ITEM IN JUMPTABLE.

FIGURE 2. SAMPLE TIMING CHART AND CONTENTS OF EXEC TABLES