DECUS
PROGRAM LIBRARY

DECUS NO.	8-146
TITLE	HIGH SPEED EXECUTIVE FOR THE PDP-8, PDP-8/I
AUTHOR	R.L. Steel
COMPANY	University of Saskatchewan Saskatoon, Saskatchewan CANADA
DATE	July 5, 1968
SOURCE LANGUAGE	

HIGH SPEED EXECUTIVE FOR THE PDP-8, PDP-8/I

DECUS Program Library Write-up DECUS No. 8-146

There has arisen a need, both in programs being written at the moment and in future programs, for a high speed interrupt executive.

A high speed interrupt executive (HSIE) is a routine that will handle the priority scheduling of a number of different I/O devices.

Basically, the function to be performed by an interrupt executive are:

(a) Determine which device is interrupting.

(b) Determine the priority of the interrupting device.

(c) Check to see if machine was servicing an interrupt previously to the current interrupt.

(d) If machine was not servicing an interrupt before the current interrupt, hand control over to the service routine that services the current interrupting device.

(e) If the machine was servicing an interrupt prior to the current interrupt determine whether the priority of the current interrupting device is higher than the other device previously being serviced.

(f) If the priority of the current interrupting device is higher than the device previously being serviced, save the contents of the registers (AC, LK, MQ, PC and priority level). Then branch to the routine which will service the current interrupting device.

(g) If the priority of the current interrupting device is lower than the device previously being serviced, queue the current device and clear its flag. Then resume servicing the higher priority device.

(h) After completion of a service routine, restore the status of the machine to its state prior to that interrupt.

(i) Check the queue to see if any device is waiting to be serviced.

(j) If no devices are queued, resume operation under the present status.

(k) If there are devices waiting in the queue, check the priority of the highest device waiting against the present priority status of the machine.

(l) If the priority of the queued device is less than the priority of the present status, resume operation under the present status.

(m) If the priority of the queued device is greater than the priority of the present status, save the contents of registers. Then branch to the routine which will handle the queued device.

The routine available at the moment (HSIE I) for interrupt scheduling will handle a maximum of twelve devices. The number and type of devices which are scheduled is decied upon by the user, also the priority of these devices is assigned by the user.

HSIE I requires three memory pages and can be loaded in any of three pages the user decides, except the first and last memory pages which are required for special purposes.

Since these routines are designed to be as flexible as possible a certain amount of data is required from the user about his program and where certain parts of his program are stored.

The initial set up procedures for the use of these routines are as follows:

(a) At the beginning of the user's actual program the locations called PSTW, QUEUE and PROHIB must be cleared. The user must declare the location PROHIB on page 0.

(b) When an interrupt occurs and control is passed to location 0001 with the PC stored in location 0000 there must be an instruction to jump to a location called FIND. This may be done by the following:

```
*1
    JMP I .+1
    FIND
```

(c) After each of the users (or systems) interrupt service routines have completed servicing an interrupt for their particular I/O devices a jump to a location called RESTART must be executed.

(d) The coded instruction in HSIE that tests the flags from the I/O devices have been coded as SKP x where x is a hexadecimal number from 1 through to 12 designating priority. The user must explicitly define these symbolic instructions according to the machine instructions required for the I/O device of that particular priority he is using. Any SKP x instructions that are not being used must be assigned the value of 7000_8.

If a user wrote a program employing four interruptable I/O devices, eg. DEC TAPE, Pencil Follower, Keyboard and Teletype, these SKP x instructions would appear as:

```
SKP 1 = 6771 (DTSF)
SKP 2 = 6131 (SPF)
SKP 3 = 6031 (KSF)
SKP 4 = 6041 (TSF)
SKP 5 = 7000 (NOP)

SKP C = 7000
```

The user will have to prepare the tape containing the above data himself and insert it into the assembler before the HSIE routines.

(e) For each I/O device the user must have a service routine, the entry point of which must be known. These entry points, absolute addresses or addresses of instructions which enable entry to these service routines, must also be supplied to HSIE. This is done by inserting the appropriate addresses, in descending order of the priority of the device their routines service, in a table called NPSW.

Referring back to (d), if the user had placed his DEC TAPE routines on Page 10, Pencil Follower routines on Page 11, Keyboard routines on Page 12, and Teleprinter on Page 13, the NPSW table would be as follows:

```
Page 0                                          NPSW, DECTAP
DECTAP, JMP 1 DTLOC                                  2200
DTLOC, 0/actual address of service routine            2400
                                                      2600
```

(f) If an interrupt occurs from a device during the servicing of an interrupt for a higher priority device, that current interrupt must be held (queued) and its flag cleared. This means that the user must supply the instruction to clear the flags of his I/O devices. These instructions are stored in a table called FLAGOFF in descending order of priority for the device they are associated with. If a device requires more than one instruction to clear its flag, eg. the Pencil Follower, the user will have to write a subroutine to clear it. Then place a JMS to this subroutine in the FLAGOFF table. The instructions from this table are pulled out and located elsewhere before they are executed, therefore any subroutine to clear flags must be located on page 0.

Interrupt Prohibition

There is a feature written into HSIE where a user may prohibit certain interrupts at certain times. The number of prohibited interrupts and the number and length of times they are prohibited is completely variable. Therefore, a user may have all levels of interrupts prohibited throughout the whole duration of his program, and this will have the effect of blanking out HSIE altogether.

To use the interrupt prohibition feature the users merely sets a bit "on" in location PROHIB to prohibit interrupts for a particular level, then clears that bit to allow interrupts again on that level. Each bit in PROHIB is assigned to a particular priority level, with the highest priority being the L.S. bit through to the lowest priority being the M.S. bit. Therefore if interrupts on priority level 9 are to be prohibited, the user must execute an inclusive OR between the contents of PROHIB and 400_8. To allow level 9 to interrupt again the user will have to AND PROHIB with 7377.

In writing these routines it has been assumed that the user will turn on the interrupt line before this program goes into the wait state. Also the interrupt service routine should turn the interrupt line back on again after clearing the device flag associated with that interrupt.

Following is a listing of HSIE together with a small program to clarify the use of it.

MODIFICATION TO THE HIGH SPEED INTERRUPT EXECUTIVE ROUTINES

TO

ACCOMMODATE THE POWER FAILURE OPTION

R.L. Steel

The following addition to HSIE will allow automatic shutdown and restart of programs during a power failure.

An additional address (FSET) in location 3 of page zero must be defined.

```
*1      JMP  I  .+1
        FIND
        FSET
```

The additional instructions in HSIE occur in the routine labelled SEARCH, the new instructions are:

```
SEARCH,  SPL
         JMP     .+5
         TAD     .+3
         DCA  2  0
         HLT
         JMP  I  Z 3
```

The above instructions are to be inserted before the existing instructions and the label SEARCH is to be relocated.

Note

Modifications listed above are not contained in the tape presently distributed with the program.

```
*1
          JMP I    .+1
          FIND
PROHIB,    0
*20
   CDT,    0
          CLA CLL
          TAD      .+5
          DTXA
          TAD      .+3
          DTXA
          JMP I    CDT
          200
   CPF,    0
          RPL
          RPM
          RPL
          RPM
          CLA CLL
          JMP I    CPF
  REST,   RESTART
*200
          CLA CLL
          DCA I    WSPT
          DCA I    KQUE
          DCA    Z PROHIB
          RRB
          PCF
          KCC
          TCF
          DCMA
          JMS    Z CDT
          JMS    Z CPF
          ION
          JMP      .+0
  WSPT,   PSTW
  KQUE,   QUEUE
*400
          KRB
          ION
          TLS
          JMP I Z REST
*600
          TCF
          ION
          JMP I Z REST

PAUSE
```

```
SKP1=6031
SKP2=6041
SKP3=7000
SKP4=7000
SKP5=7000
SKP6=7000
SKP7=7000
SKP8=7000
SKP9=7000
SKPA=7000
SKPB=7000
SKPC=7000
```

```
/                HIGH SPEED INTERRUPT EXECUTIVE
/WRITTEN BY              BOB STEEL                    5TH JULY 1968
/
*4000
FIND,     DCA     SAVEAC      /INITIAL PSW STORAGE
          RAL
          DCA     SAVELK
          MQA
          DCA     SAVEMQ
          TAD Z   0
          DCA     SAVEPC
          TAD     PSTW        /INITIALLY SET PSTW TO ZERO
          DCA     PSTS
          JMP I   .+1
          SEARCH
FSET,     TAD     PSTS         /RESTORE PSW AND RESUME
          DCA     PSTW
          TAD     SAVEPC
          DCA Z   0
          TAD     SAVEMQ
          MQL
          TAD     SAVELK
          RAR
          TAD     SAVEAC
          ION
          JMP I Z 0
SAVEAC,   0
SAVELK,   0
SAVEMQ,   0
SAVEPC,   0
  PSTS,   0
  PSTW,   0
 QUEUE,   0
RESTART,  IOF
          CLA CLL
          TAD     PSTW       / DETERMINE PRESENT STATUS OF
          RAR                 /MACHINE TO RETRIEVE PSW'S
          SZL
          JMP     .+3
          ISZ     RSTATW
          JMP     .-4
          CLA     CLL
          TAD     RSTATW
          DCA     .+3         /SET UP INSTRUCTIONS FOR RETRIEVING
          TAD     RSTATW+1    /OLD PSW'S
          DCA     RSTATW
          NOP
          DCA     RSTATW+2
          TAD I   RSTATW+2    /RETRIEVE OLD PSW'S
          DCA     PSTS
          ISZ     RSTATW+2
          TAD I   RSTATW+2
          DCA     SAVEPC
          ISZ     RSTATW+2
          TAD I   RSTATW+2
          DCA     SAVEMQ
          ISZ     RSTATW+2
          TAD I   RSTATW+2
          DCA     SAVELK
```

```
            ISZ     RSTATW+2
            TAD I   RSTATW+2
            DCA     SAVEAC
            DCA     QUECON      /CLEAR QUEUE COUNTER
            TAD     QUEUE       /ARE THERE ANY QUEUED INTERRUPTS?
            SNA
            JMP     FSET        /NO, RESUME WITH LAST STATUS
            RAR                 /YES, DETERMINE HIGHEST LEVEL OF
            SZL                 /QUEUED INTERRUPT
            JMP     .+3
            ISZ     QUECON
            JMP     .-4
            CLA CLL
            DCA     PSTC        /CLEAR PROGRAM STATUS COUNTER
            TAD     PSTS        /ARE THERE ANY WAITING INTERRUPTS?
            SNA
            JMP     RPROG       /NO, INITIATE HIGHEST QUEUED INTERRUPT
            RAR                 /YES, DETERMINE WHETHER HIGHEST QUEUED
            SZL                 /INTERRUPT IS AT A HIGHER LEVEL THAN
            JMP     .+3         /THE WAITING INTERRUPTS
            ISZ     PSTC
            JMP     .-4
            TAD     QUECON
            CMA     IAC
            TAD     PSTC
            SMA     CLA
            JMP     RPROG       /QUEUED INTERRUPT > WAITING INTERRUPT,
            JMP     FSET        /INITIATE QUEUED INTERRUPT
                                /WAITING INTERRUPT > QUEUED INTERRUPT
                                /RESUME WAITING INTERRUPT
RPROG,      TAD     QUECON      /SET UP JUMP INSTRUCTION TO APPROPRIATE
            TAD     QUECON+1    /PLACE TO SERVICE QUEUED INTERRUPT
            DCA     .+13
            TAD     QUECON      /MASK OUT BIT IN QUEUE
            CMA
            DCA     QUECON
            CLL CML
            RAL
            ISZ     QUECON
            JMP     .-2
            CMA
            AND     QUEUE
            DCA     QUEUE
            NOP
RSTATW,     TAD     PSW0S
            TAD     PSW0S
            0
  PSTC,     0
QUECON,     0
            JMP I   SIMINT
SIMINT,     TEST1
            TEST2
            TEST3
            TEST4
            TEST5
            TEST6
            TEST7
            TEST8
            TEST9
            TESTA
```

```
         TCSIC
PSWDS,   PSW1
         PSW2
         PSW3
         PSW4
         PSW5
         PSW6
         PSW7
         PSW8
         PSW9
         PSWA
         PSWB
         PSWC
```

```
*4200
STOLD,    0
          TAD I   STOLD          /GET PSW STORAGE ADDRESS
          ISZ     STOLD
          DCA     ASTPSW
          TAD I   STOLD          /GET PRIORITY LEVEL
          MQL
          MQA
          AND  Z PROHIB
          SZA
          JMP     .+14
          TAD I   IPSTW          /DETERMINE IF PRESENT STATUS > NEW STATUS
          SNA
          JMP     .+32
          CMA   IAC
          TAD I   STOLD
          SPA     KLA
          JMP     .+26
          TAD I   IQUEUE         /PRESENT STATUS > NEW STATUS
          MQA                    /QUEUE NEW STATUS AND CLEAR FLAG
          DCA I   IQUEUE
          TAD I   STOLD
          ISZ     STOLD
          CLL
          RAR
          SZL
          JMP     .+3
          ISZ     IFLAGOF
          JMP     .-4
          CLA     CLL
          TAD     ASTPSW+1
          DCA     .+1
          NOP
          DCA     .+1
          NOP
          CLA     CLL
          TAD     IFLAGOF+1
          DCA     IFLAGOF
          JMP I   IFSET          /RESUME PRESENT STATUS
          TAD I   IPSTW          /NEW STATUS > PRESENT STATUS
          DCA I   ASTPSW         /SAVE PRESENT PSW
          ISZ     ASTPSW         /INITIATE NEW STATUS
          TAD I   ISAVPC
          DCA I   ASTPSW
          ISZ     ASTPSW
          TAD I   ISAVMQ
          DCA I   ASTPSW
          ISZ     ASTPSW
          TAD I   ISAVLK
          DCA I   ASTPSW
          ISZ     ASTPSW
          TAD I   ISAVAC
          DCA I   ASTPSW
          TAD I   STOLD
          ISZ     STOLD
          DCA I   IPSTW
          JMP I   STOLD
ASTPSW,   0
          TAD I   IFLAGOF
ISAVAC,   SAVEAC
```

```
ISAVLK,   SAVELK
ISAVMQ,   SAVEMQ
ISAVPC,   SAVEPC
 IPSTW,   PSTW
 IFSET,   FSET
IQUEUE,   QUEUE
PSW1,     0
          0
          0
          0
          0
          0
PSW2,     0
          0
          0
          0
          0
          0
PSW3,     0
          0
          0
          0
          0
          0
PSW4,     0
          0
          0
          0
          0
          0
PSW5,     0
          0
          0
          0
          0
          0
PSW6,     0
          0
          0
          0
          0
          0
PSW7,     0
          0
          0
          0
          0
          0
PSW8,     0
          0
          0
          0
          0
          0
PSW9,     0
          0
          0
          0
          0
          0
PSWA,     0
          0
          0
          0
          0
          0
PSWB,     0
          0
          0
          0
```

```
            0
            0
PSWD,       0
            0
            0
            0
            0
            0
IFLAGUF,    FLAGOFF
            FLAGOFF
```

```
*4400
SEARCH,   SKP1
          JMP     .+7
TEST1,    JMS I   ISTOLD
          PSW1
          1
          TAD     NPSW
          DCA Z   Ø
          JMP I   Z Ø
          SKP2
          JMP     .+7
TEST2,    JMS I   ISTOLD
          PSW2
          2
          TAD     NPSW+1
          DCA Z   Ø
          JMP I   Z Ø
          SKP3
          JMP     .+7
TEST3,    JMS I   ISTOLD
          PSW3
          4
          TAD     NPSW+2
          DCA Z   Ø
          JMP I   Z Ø
          SKP4
          JMP     .+7
TEST4,    JMS I   ISTOLD
          PSW4
          1Ø
          TAD     NPSW+3
          DCA Z   Ø
          JMP I   Z Ø
          SKP5
          JMP     .+7
TEST5,    JMS I   ISTOLD
          PSW5
          2Ø
          TAD     NPSW+4
          DCA     Z Ø
          JMP I   Z Ø
          SKP6
          JMP     .+7
TEST6,    JMS I   ISTOLD
          PSW6
          4Ø
          TAD     NPSW+5
          DCA     Z Ø
          JMP I   Z Ø
          SKP7
          JMP     .+7
TEST7,    JMS I   ISTOLD
          PSW7
          1ØØ
          TAD     NPSW+6
          DCA Z   Ø
          JMP I   Z Ø
          SKP0
          JMP     .+7
```

```
TEST8,    JMS I  ISTOLD
          PSW8
          200
          TAD    NPSW+7
          DCA Z  0
          JMP I  Z 0
          SKP9
          JMP    .+7
TEST9,    JMS I  ISTOLD
          PSW9
          400
          TAD    NPSW+10
          DCA Z  0
          JMP I  Z 0
          SKPA
          JMP    .+7
TESTA,    JMS I  ISTOLD
          PSWA
          1000
          TAD    NPSW+11
          DCA Z  0
          JMP I  Z 0
          SKPB
          JMP    .+7
TESTB,    JMS I  ISTOLD
          PSWB
          2000
          TAD    NPSW+12
          DCA Z  0
          JMP I  Z 0
          SKPC
          JMP I  .+7
TESTC,    JMS I  ISTOLD
          PSWC
          4000
          TAD    NPSW+13
          DCA Z  0
          JMP I  Z 0
          FSET
ISTOLD,   SIOLD
NPSW,     400
          600
          0
          0
          0
          0
          0
          0
          0
          0
          0
          0
          0
FLAGOFF,  KCC
          TCF
          0
          0
          0
          0
          0
          0
```

```
0
0
0
0
```

PAUSE