```
*-----------------------------------------------------------
* Title      : RTS
* Written by : JNC
* Date       : Dec 23
* Description: Outline runtime system
*-----------------------------------------------------------
                                ;system call equates
sys     equ     0               ; system call trap (trap 0)
syscr   equ     1               ; create new task
sysdel  equ     2               ; delete task
syswtmx equ     3               ; wait on mutex
syssgmx equ     4               ; signal mutex
sysinmx equ     5               ; initialise mutex
syswttm equ     6               ; wait on timer
syswtio equ     8               ;wait on I/O

usrcode equ     $1000           ;address of user task 0
usrstk  equ     $8000           ;address of user stack
ftcbad  equ     $FFFB00         ;address of the first tcb

ntcblst equ     8               ;number of records in tcb list

tcb     org     0               ;tcb record
tcbd0   ds.l    1               ; D register save
tcbd1   ds.l    1
tcbd2   ds.l    1
tcbd3   ds.l    1
tcbd4   ds.l    1
tcbd5   ds.l    1
tcbd6   ds.l    1
tcbd7   ds.l    1
tcba0   ds.l    1               ; A register save
tcba1   ds.l    1
tcba2   ds.l    1
tcba3   ds.l    1
tcba4   ds.l    1
tcba5   ds.l    1
tcba6   ds.l    1
tcba7   ds.l    1
tcbsr   ds.l    1               ; SR (status reg) save
tcbpc   ds.l    1               ; PC save
tcbnext ds.l    1               ; link to next record
tcbused ds.l    1               ; record in use flag
tcbwtim ds.l    1               ; timer wait expiry time
tcbwtcd ds.l    1               ; a code to specify the reason for
being in the waiting list
tcblen  equ     *               ; length of tcb record

;*****************************************************************
**********
rts                                     ;RUNTIME SYSTEM
;*****************************************************************
**********
```

```
;*************************************************************************
***********
                                        ;INTERRUPT VECTORS
;*************************************************************************
***********

        org     0

        dc.l    usrstk                  ; initial SP
        dc.l    res                     ; reset
        ds.b    $5C
        dc.l    fltint                  ; interrupt 1 (timer)
        dc.l    fioint
        ds.b    $14
        dc.l    flsint                  ; trap 0 (system call)

;*************************************************************************
************
res                                     ;RESET
;*************************************************************************
************
        ;set all tcbs as unused and t0 as default user task
        or      #%0000011100000000,sr
        move.l  #ftcbad,a0
        move.l  #usrcode,d0
        move.l  d0,tcbpc(a0)
        move.l  a0,d0
        move.l  d0,tcbnext(a0)
        move.l  #0,tcbused(a0)
        move.l  d0,rdytcb

        move    #7,d0
        add     #$80,a0
l0:     move.l  #1,tcbused(a0)
        add     #$80,a0
        sub     #1,d0
        bne     l0

        move.l  #1,d0                   ;set mutex variable to
default available '1'
        move.l  d0,mtx

        move.l  #0,d0
        move.l  #0,a0
        move.l  d0,fwtcbad
        move.l  d0,prwtcb
        move.l  d0,wlstlen
        move.l  d0,wttcb
        move.l  d0,counter
        move.l  d0,wtcbcnt
        ;and     #%1111100011111111,sr
        jmp     usrcode

;*************************************************************************
```

```
************
flih                                    ;FIRST-LEVEL INTERRUPT
HANDLER
;*****************************************************************
************

fltint                                  ;ENTRY FROM TIMER INTERRUPT
        or      #%0000011100000000,sr
        move.l  d0,d0sav                ;save D0
        move.l  #$0,d0                  ;set id = 0
        move.l  d0,id
        move.l  counter,d0              ;update the clock counter
        add     #$1,d0
        move.l  d0,counter

        move.l  wtcbcnt,d0              ;code for wait timer logic
        cmp     #0,d0
        beq     flit1
        move.l  #7,d0
        move.l  d0,id
flit1:  move.l  d0sav,d0                ;restore D0
        bra     fl1

fioint                                  ;ENTRY FROM I/O INTERRUPT
        or      #%0000011100000000,sr
        move.l  #11,d0
        move.l  d0,ioid
        bra     fl1

flsint                                  ;ENTRY FROM TRAP (SOFTWARE
INTERRUPT)
        or      #%0000011100000000,sr   ;disable hardware interrupts
        move.l  d0,id                   ;store id
        bra     fl1

fl1     move.l  a0,a0sav                ;save working reg

        move.l  rdytcb,a0               ;A0 ^ 1st ready tcb (ie
running tcb)

        move.l  d0,tcbd0(a0)            ;store registers
        move.l  d1,tcbd1(a0)
        move.l  d2,tcbd2(a0)
        move.l  d3,tcbd3(a0)
        move.l  d4,tcbd4(a0)
        move.l  d5,tcbd5(a0)
        move.l  d6,tcbd6(a0)
        move.l  d7,tcbd7(a0)
        move.l  a0sav,d0
        move.l  d0,tcba0(a0)
        move.l  a1,tcba1(a0)
        move.l  a2,tcba2(a0)
        move.l  a3,tcba3(a0)
        move.l  a4,tcba4(a0)
```

```
        move.l  a5,tcba5(a0)
        move.l  a6,tcba6(a0)

        move    (sp),d0                 ;pop and store SR
        add.l   #2,sp
        move.l  d0,tcbsr(a0)

        move.l  (sp),d0                 ;pop and store PC
        add.l   #4,sp
        move.l  d0,tcbpc(a0)

        move.l  a7,tcba7(a0)            ;store SP           //28

;*************************************************************************
***********
serv                                    ;SERVICE ROUTINES
;*************************************************************************
***********
        move.l  id,d0
        move.l  ioid,d5
        move.l  rdytcb,a0

        cmp     #11,d5
        beq     iohdlr
        cmp     #1,d0                   ;check id and choose
appropriate function
        beq     crttsk
        cmp     #2,d0
        beq     dlttsk
        cmp     #3,d0
        beq     mtxwt
        cmp     #4,d0
        beq     sglmtx
        cmp     #5,d0
        beq     intmtx
        cmp     #6,d0
        beq     wttmr
        cmp     #7,d0
        beq     wttmr1
        cmp     #8,d0
        beq     wtio
        bra     sched

;***************************CREATE TASK
FUNCTION***********************************
crttsk: add     #$80,a0
        move.l  a0,d0
        sub     mtcbad,d0
        bmi     ct0
        move.l  #ftcbad,a0
ct0:    move.l  tcbused(a0),d0
        and.l   #1,d0
        beq     crttsk
```

```
        move.l  a0,d2
        move.l  rdytcb,a0
        move.l  tcbnext(a0),d0          ;create task logic ;logic to
add new item to ready list
        move.l  d2,tcbnext(a0)          ;d2 contains the address of
the new task tcb
        move.l  d2,a0
        move.l  d1,tcbpc(a0)
        move.l  d0,tcbnext(a0)

        ;move.l  #0,tcbd0(a0)            ;store registers
        ;move.l  #0,tcbd1(a0)
        ;move.l  #0,tcbd2(a0)
        ;move.l  #0,tcbd3(a0)
        ;move.l  #0,tcbd4(a0)
        ;move.l  #0,tcbd5(a0)
        ;move.l  #0,tcbd6(a0)
        ;move.l  #0,tcbd7(a0)
        ;move.l  #0,tcba0(a0)
        ;move.l  #0,tcba1(a0)
        ;move.l  #0,tcba2(a0)
        ;move.l  #0,tcba3(a0)
        ;move.l  #0,tcba4(a0)
        ;move.l  #0,tcba5(a0)
        ;move.l  #0,tcba6(a0)
        move.l  #$01000000,tcba7(a0)
        move.l  #$2000,tcbsr(a0)
        move.l  #0,tcbused(a0)
        bra     sched

;***************************DELETE TASK
FUNCTION***********************************
dlttsk: move.l  a0,d1
dtsk0:  move.l  tcbnext(a0),d0          ;delete task logic
        cmp     d1,d0
        beq     dtsk1
        move.l  d0,a0
        bra     dtsk0
dtsk1:  move.l  tcbnext(a0),a1
        move.l  tcbnext(a1),tcbnext(a0)
        move.l  #1,tcbused(a0)
        bra     sched

;***************************WAIT MUTEX
FUNCTION************************************

mtxwt:  move.l  mtx,d0
        cmp     #1,d0
        beq     mt0

        move.l  a0,d0                   ;removing the tcb from ready
list
mtl0:   move.l  tcbnext(a0),d1
        cmp     d0,d1
```

```
        beq     mtl1
        move.l  d1,a0
        bra     mtl0
mtl1:   move.l  tcbnext(a0),a1
        move.l  tcbnext(a1),tcbnext(a0)
        move.l  a0,rdytcb
        move.l  a1,a0
        move.l  #$C8,tcbwtcd(a0)        ;updating the reason for tcb
in wait list

        move.l  wttcb,d0                ;adding tcb to wait list
        cmp     #0,d0
        beq     mt1

        move.l  wttcb,a1
        move.l  a0,tcbnext(a1)
        move.l  #0,tcbnext(a0)
        move.l  a0,wttcb
        move.l  wlstlen,d0              ;update the length of wait
list
        add     #1,d0
        move.l  d0,wlstlen
        bra     sched

mt1:    move.l  a0,wttcb                ;adding a single item to
wait list
        move.l  a0,fwtcbad
        move.l  #1,wlstlen
        move.l  #0,tcbnext(a0)
        bra     sched

mt0:    move.l  #0,d0
        move.l  d0,mtx
        bra     sched

;**************************SIGNAL MUTEX
FUNCTION*********************************
sglmtx: move.l  wlstlen,d0
        cmp     #0,d0
        beq     st0

        move.l  fwtcbad,a1
stl0:   move.l  tcbwtcd(a1),d0
        cmp     #$C8,d0
        beq     st1
        move.l  tcbnext(a1),d0          ;handling a case with zero
items in wait list waiting for mutex
        cmp     #0,d0
        beq     st4
        move.l  a1,prwtcb
        move.l  d0,a1
        bra     stl0

st1:    move.l  prwtcb,d0
```

```
        cmp     #0,d0
        beq     st2
        move.l  tcbnext(a1),tcbnext(a2)
        move.l  prwtcb,a2
        bra     st3

st2:    move.l  tcbnext(a1),fwtcbad     ;removing the first item
from the wait list (case 1)

st3:    move.l  #$FF,tcbwtcd(a1)
        move.l  tcbnext(a0),tcbnext(a1)
        move.l  a1,tcbnext(a0)
        move.l  wlstlen,d0              ;updating the length of wait
list
        sub     #1,d0
        cmp     #0,d0
        bne     st5
        move.l  #0,wttcb
st5:    move.l  d0,wlstlen
        move.l  #0,prwtcb               ;erasing the holding
variable
        bra     sched

st0:    move.l  #1,d0
        move.l  d0,mtx
st4:    bra     sched

;**********************INITIALIZE MUTEX
FUNCTION*********************************
intmtx: cmp     #0,d1                  ;set mutex as specified in
the parameter
        beq     intm0
        cmp     #1,d1
        beq     intm0
        bra     disp
intm0:  move.l  d1,mtx
        bra     disp

;*************************WAIT TIMER
FUNCTION************************************
wttmr:  move.l  counter,d2
        add     d2,d1                   ;update the expiry time
        move.l  d1,tcbwtim(a0)

        move.l  a0,d0                   ;removing the tcb from ready
list
wtl0:   move.l  tcbnext(a0),d1
        cmp     d0,d1
        beq     wtl1
        move.l  d1,a0
        bra     wtl0
wtl1:   move.l  tcbnext(a0),a1
        move.l  tcbnext(a1),tcbnext(a0)
        move.l  a0,rdytcb
```

```
        move.l  a1,a0

        move.l  #$34,d0                 ;updating the reason for tcb
in wait list
        move.l  d0,tcbwtcd(a0)

        move.l  wttcb,d0
        cmp     #0,d0
        bne     wt1
        move.l  #0,tcbnext(a0)
        move.l  a0,wttcb
        move.l  a0,fwtcbad
        move.l  #1,wlstlen
        bra     wt2

wt1:    move.l  wttcb,a1
        move.l  a0,wttcb
        move.l  a0,tcbnext(a1)
        move.l  #0,tcbnext(a0)
        move.l  wlstlen,d0
        add     #1,d0
        move.l  d0,wlstlen

wt2:    move.l  wtcbcnt,d0
        add     #1,d0
        move.l  d0,wtcbcnt
        bra     sched

wttmr1: move.l  #0,d4
        move.l  fwtcbad,a1
fltl0:  move.l  tcbwtcd(a1),d2
        cmp     #$34,d2
        bne     flt1
flt0:   move.l  tcbwtim(a1),d2
        move.l  counter,d3
        cmp     d2,d3
        beq     flt2
flt1:   move.l  tcbnext(a1),d1
        cmp     #0,d1
        beq     sched
        move.l  a1,d4
        move.l  d1,a1
        bra     fltl0
flt2:   cmp     #0,d4
        beq     flt3
        move.l  d4,a3
        move.l  tcbnext(a1),tcbnext(a3)
        bra     flt4
flt3:   move.l  tcbnext(a1),fwtcbad
flt4:   move.l  wlstlen,d1
        sub     #1,d1
        cmp     #0,d1
        bne     flt5
        move.l  #0,wttcb
```

```
flt5:   move.l  d1,wlstlen
        move.l  #$FF,tcbwtcd(a1)
        move.l  tcbnext(a0),tcbnext(a1)
        move.l  a1,tcbnext(a0)
        move.l  wtcbcnt,d1
        sub     #1,d1
        move.l  d1,wtcbcnt
        bra     sched

;*************************wait I/O
function************************************
wtio:   move.l  a0,d0                   ;removing the tcb from ready
list
wtil0:  move.l  tcbnext(a0),d1
        cmp     d0,d1
        beq     wtil1
        move.l  d1,a0
        bra     wtil0
wtil1:  move.l  tcbnext(a0),a1
        move.l  tcbnext(a1),tcbnext(a0)
        move.l  a0,rdytcb
        move.l  a1,a0
        move.l  #$3A,tcbwtcd(a0)        ;updating the reason for tcb
in wait list

        move.l  wttcb,d0                ;adding tcb to wait list
        cmp     #0,d0
        beq     wti1

        move.l  wttcb,a1
        move.l  a0,tcbnext(a1)
        move.l  #0,tcbnext(a0)
        move.l  a0,wttcb
        move.l  wlstlen,d0              ;update the length of wait
list
        add     #1,d0
        move.l  d0,wlstlen
        bra     sched

wti1:   move.l  a0,wttcb                ;adding a single item to
wait list
        move.l  a0,fwtcbad
        move.l  #1,wlstlen
        move.l  #0,tcbnext(a0)
        bra     sched

;************************I/O handler
function************************************
iohdlr: move.l  #0,d4
        move.l  fwtcbad,a1
iohl0:  move.l  tcbwtcd(a1),d2
        cmp     #$3A,d2
        beq     ioh2
        move.l  tcbnext(a1),d1
```

```
        cmp     #0,d1
        beq     sched
        move.l  a1,d4
        move.l  d1,a1
        bra     iohl0
ioh2:   cmp     #0,d4
        beq     ioh3
        move.l  d4,a3
        move.l  tcbnext(a1),tcbnext(a3)
        bra     ioh4
ioh3:   move.l  tcbnext(a1),fwtcbad
ioh4:   move.l  wlstlen,d1
        sub     #1,d1
        cmp     #0,d1
        bne     ioh5
        move.l  #0,wttcb
ioh5:   move.l  d1,wlstlen
        move.l  #$FF,tcbwtcd(a1)
        move.l  tcbnext(a0),tcbnext(a1)
        move.l  a1,tcbnext(a0)
        move.l  #0,ioid
        bra     sched

;*************************************************************************
************
sched                                   ;SCHEDULER
;*************************************************************************
************
        move.l  rdytcb,a0               ;round robin scheduler
        move.l  tcbnext(a0),d0
        move.l  d0,rdytcb

;*************************************************************************
************
disp                                    ;DISPATCHER
;*************************************************************************
************

        move.l  rdytcb,a0               ;A0 ^ new running tcb
        move.l  tcbd1(a0),d1            ;restore registers
        move.l  tcbd2(a0),d2
        move.l  tcbd3(a0),d3
        move.l  tcbd4(a0),d4
        move.l  tcbd5(a0),d5
        move.l  tcbd6(a0),d6
        move.l  tcbd7(a0),d7
        move.l  tcba1(a0),a1
        move.l  tcba2(a0),a2
        move.l  tcba3(a0),a3
        move.l  tcba4(a0),a4
        move.l  tcba5(a0),a5
        move.l  tcba6(a0),a6
        move.l  tcba7(a0),a7
```

```
        sub.l   #4,sp                   ;push PC
        move.l  tcbpc(a0),d0
        move.l  d0,(sp)

        sub.l   #2,sp
        move.l  tcbsr(a0),d0            ;push SR
        move    d0,(sp)

        move.l  tcbd0(a0),d0            ;restore remaining registers
        move.l  tcba0(a0),a0

        rte                             ;return              //27

;*****************************************************************************
                                        ;RTS variables
;*****************************************************************************

tcblst  ds.b    tcblen*ntcblst          ;tcb list
rdytcb  ds.l    1                       ;^ ready tcb list
wttcb   ds.l    1                       ;^ waiting tcb
a0sav   ds.l    1                       ;A0 temporary save
d0sav   ds.l    1                       ;D0 temporary save
id      ds.l    1                       ;function id
ioid    ds.l    1                       ;
time    ds.l    1                       ;system time
mtx     ds.l    1                       ;mutex variable
mtcbad  dc.l    $FFFE81                 ;address limit for the last
tcb starting point
fwtcbad ds.l    1                       ;address of the first tcb in
the waiting list
wlstlen ds.l    1                       ;holds the realtime length
of the waiting list
prwtcb  ds.l    1                       ;holds the address of
previous tcb in waiting list in signal mutex logic
counter ds.l    1                       ;holds the number of timer
interupts occurred
wtcbcnt ds.l    1                       ;holds the count of tcbs'
waiting for time expiry

;*****************************************************************************
                                        ;USER APPLICATION TASKS
;*****************************************************************************

        org     usrcode
        and     #%1111100011111111,sr
;first application program [create task test]

led     equ     $e00010         ;led
sw      equ     $e00014         ;switch
sevseg  equ     $e0000e         ;seven segment display
```

```
;t0:                                ;TASK 0
;        move.l  #syscr,d0         ;start task 1
;        move.l  #t1,d1
;        move.l  #$4000,d2
;        trap    #sys
                                   ;repeat
;t00:    move.l  #$01,d1           ;  set led 0
;        move.b  d1,led

;        bra     t00

;t1:                               ;TASK 1
                                   ;repeat
;        move.l  #$02,d0           ;  set led 1
;        move.b  d0,led

;        bra     t1

;        END    res

;second application program     [test for mutex]
;t0:     move.l  #syscr,d0
;        move.l  #t1,d1
;        move.l  #$3000,d2
;        trap    #sys

;        move.l  #syscr,d0
;        move.l  #t2,d1
;        move.l  #$4000,d2
;        trap    #sys

;t00:    move.l  #kseg,a0
;        move.l  a,d1
;        move.l  b,d2
;        add.l   d1,d2
;        move.l  #syswtmx,d0
;        trap    #sys
;        move.l  c,d3
;        sub     d3,d2
;        add.l   d2,a0
;        move.b  (a0),d3
;        move.b  d3,sevseg
;        move.l  #syssgmx,d0
;        trap    #sys
;        bra     t00

;t1:     move.l  a,d0
;        add     #1,d0
;        move.l  d0,a

;        move.l  #syswtmx,d0
;        trap    #sys
;        move.l  c,d0
```

```
;        add     #1,d0
;        move.l  d0,c
;        move.l  #syssgmx,d0
;        trap    #sys
;        bra     t1

;t2:     move.l  b,d0
;        add     #1,d0
;        move.l  d0,b

;        move.l  #syswtmx,d0
;        trap    #sys
;        move.l  c,d0
;        add     #1,d0
;        move.l  d0,c
;        move.l  #syssgmx,d0
;        trap    #sys
;        bra     t2


;a       dc.l    0
;b       dc.l    0
;c       dc.l    0

;third application program [test for wait time function]
;t0:     move.l  #syscr,d0
;        move.l  #t1,d1
;        move.l  #$4000,d2
;        trap    #sys

;t00:    move.l  #$01,d1
;        move.b  d1,led
;        bra     t00

;t1:     move.b  sw,d0
;        and.l   #1,d0
;        beq     t10
;        move.l  #$02,d1
;        move.b  d1,led
;        bra     t1
;t10:    move.l  #$00,d1
;        move.b  sw,d0
;        and.b   #1,d0
;        beq     t10

;        move.l  #syswttm,d0
;        move.l  #3,d1
;        trap    #sys
;        bra     t1
;t11:    move.l  #$02,d1
;        move.b  d1,led
;        bra     t11

;fourth application program to test delete task
```

```
;t0:     move.l  #syscr,d0
;        move.l  #t1,d1
;        move.l  #$4000,d2
;        trap    #sys
;t00:    move.l  #$01,d1
;        move.b  d1,led
;        bra     t00

;t1:     move.b  sw,d0
;        and.l   #1,d0
;        beq     t10

        ; untile Switch is pressed, perform task
;        move.l  #$02,d1
;        move.b  d1,led
;        bra     t1

;t10:    move.l  #$00,d1
;        move.b  sw,d0
;        and.b   #1,d0
;        bne     t11  ; Jump to task deletion if the button is
released

;        bra     t10  ; Continue looping if the button is still
pressed

;t11:                          ; Delete task when the button is
released
;        move.l  #sysdel,d0  ; Assuming sysdelt is the system call
for task deletion
;        trap    #sys
;        bra     t11

;fifth application program to test wait I/O
t0:     move.l  #syscr,d0
        move.l  #t1,d1
        move.l  #$4000,d2
        trap    #sys

t00:    move.l  #$01,d1
        move.b  d1,led
        bra     t00

t1:     move.b  sw,d0
        and.l   #1,d0
        beq     t10
        move.l  #$02,d1
        move.b  d1,led
        bra     t1
t10:    move.l  #$00,d1
        move.b  sw,d0
        and.b   #1,d0
        beq     t10
```

```
        move.l  #syswtio,d0
        trap    #sys
t11:    move.l  #$02,d1
        move.b  d1,led
        bra     t11
kseg                        ;7-seg display patterns
        dc.b    $3f ;0
        dc.b    $06 ;1
        dc.b    $5b ;2
        dc.b    $4f ;3
        dc.b    $66 ;4
        dc.b    $6d ;5
        dc.b    $7d ;6
        dc.b    $07 ;7
        dc.b    $7f ;8
        dc.b    $67 ;9
        dc.b    $77 ;A
        dc.b    $7c ;b
        dc.b    $39 ;C
        dc.b    $5e ;d
        dc.b    $79 ;E
        dc.b    $71 ;F
        dc.b    $80 ;.

        END     res
```