Hourglass

Introduction and Overview

The Hourglass module will change the pointer shape to that of an hourglass. You can optionally also display:

- a percentage figure
- 'LED' indicators for status information (one above the hourglass, and one below).

Note that cursor shapes 3 and 4 are used (and hence corrupted) by the hourglass. You should not use these shapes in your programs.

Normally the Hourglass module is used to display an hourglass on the screen whenever there is prolonged activity on the Econet. The calls to do so are made by the NetStatus module, which claims the EconetV vector. See the section entitled *SOFTVECS.HTML#79772* and the *../filesystems/netstatus (on page 0)* for further details.

The hourglass should also be used by any software that may take some time to do a particular job, especially when:

- there is no other indication of activity
- the processing time is file size dependent (some users may have files much bigger than you expect)
- the processing time is processor speed dependent (some users may be in a screen mode that is hungry for memory bandwidth).

Software using the hourglass should, whenever possible, use the percentage feature; see the section entitled *Example programs (on page 12)* for an example of this.

The rest of this chapter details the SWIs used to control the hourglass.

SWI Calls

Hourglass_On (SWI &406C0)

Turns on the hourglass

On entry

None

On exit

None

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This turns on the hourglass. Although control returns immediately there is a delay of $^1/_3$ of a second before the hourglass becomes visible. Thus you can bracket an operation by Hourglass_On/Hourglass_Off so that the hourglass will only be displayed if the operation takes longer than $^1/_3$ of a second.

You can set a different delay using *SWI Hourglass_Start (on page 6).*

Hourglass_On's are nestable. If the hourglass is already visible then a count is incremented and the hourglass will remain visible until an equivalent number of Hourglass_Off's are done. The LEDs and percentage indicators remain unchanged.

Examples

The example below illustrates the use of bracketing calls to Hourglass_On / Hourglass_Off:

```
DoLoadAndProcess
        STMFD   r13!, {r0-r5, r14}
        MOV     r0, #OSFile_Load
        ADR     r2, Buffer
        MOV     r3, #0
        SWI     XOS_File
        BVS     ExitLoadAndProcess
        CMP     r4, #0
        BEQ     ExitLoadAndProcess
        SWI     XHourglass_On
        BVS     ExitLoadAndProcess
        ADR     r1, Buffer
ProcessLoop
        LDRB    r0, [r1], #1
        BL      ProcessByte
        BVS     FinishProcess
        SUBS    r4, r4, #1
        BNE     ProcessLoop
FinishProcess
        SWI     XHourglass_Off
ExitLoadAndProcess
        STRVS   r0, [r13, #0]
        LDMFD   r13!, {r0-r5, pc}
```

Related SWIs

SWI Hourglass_Off (on page 4)
SWI Hourglass_Start (on page 6)

Hourglass_Off (SWI &406C1)

Turns off the hourglass

On entry

None

On exit

None

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call decreases the count of the number of times that the hourglass has been turned on. If this makes the count zero, it turns off the hourglass.

When the hourglass is removed the pointer number and colours are restored to those in use at the first Hourglass_On.

From RISC OS 3 onwards, the system also turns the percentage display off if leaving the level that turned it on, even if the hourglass itself is not turned off. See the *SWI Hourglass_On (on page 2)* for an example of this.

Related SWIs

SWI Hourglass_On (on page 2)
SWI Hourglass_Smash (on page 5)

Hourglass_Smash (SWI &406C2)

Turns off the hourglass immediately

On entry

None

On exit

None

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call turns off the hourglass immediately, taking no notice of the count of nested Hourglass_On's. If you use this call you must be sure neither you, nor anyone else, should be displaying an hourglass.

When the hourglass is removed the pointer number and colours are restored to those in use at the first Hourglass_On, except under RISC OS 2.

Related SWIs

SWI Hourglass_Off (on page 4)

Hourglass_Start (SWI &406C3)

Turns on the hourglass after a given delay

On entry

R0 = delay before start-up (in centiseconds), or 0 to suppress the hourglass

On exit

None

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call works in the same way as Hourglass_On, except you can specify your own start-up delay.

If you specify a delay of zero and the hourglass is currently off, then future Hourglass_On and Hourglass_Start calls have no effect. The condition is terminated by the matching Hourglass_Off, or by an Hourglass_Smash.

Related SWIs

SWI Hourglass_On (on page 2)
SWI Hourglass_Off (on page 4)

Hourglass_Percentage (SWI &406C4)

Displays a percentage below the hourglass

On entry

R0 = percentage to display (if in range 0 - 99), else turns off percentage

On exit

None

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call controls the display of a percentage below the hourglass. If R0 is in the range 0 - 99 the value is displayed; if it is outside this range, the percentage display is turned off.

The default condition of an hourglass is not to display percentages.

For a full example of the use of Hourglass_Percentage, see the section entitled *Example programs (on page 12).*

From RISC OS 3 onwards, lower levels of calls cannot alter the hourglass percentage once a higher level call is using it. Furthermore, Hourglass_Off automatically turns the percentage display off when leaving the level that turned it on, even if the hourglass itself is not turned off. For example:

```
SYS "Hourglass_On"
```

```
  SYS "Hourglass_On"
  SYS "Hourglass_Percentage",10       :REM sets to 10%
  SYS "Hourglass_Percentage",20       :REM sets to 20%
    SYS "Hourglass_On"
    SYS "Hourglass_Percentage",50     :REM DOESN'T set to 50%
    SYS "Hourglass_Off"
  SYS "Hourglass_Percentage",30       :REM sets to 30%
  SYS "Hourglass_Off"                 :REM turns off percentages
SYS "Hourglass_Off"                   :REM turns off hourglass
```

Related APIs

None

Hourglass_LEDs (SWI &406C5)

Controls the display indicators above and below the hourglass

On entry

R0 = AND value for LEDs
R1 = EOR value for LEDs

On exit

R0 = old value of LEDs' word

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call controls the two display indicators above and below the hourglass, which can be used to display status information. These are controlled by bits 0 and 1 respectively of the LEDs' word. The indicator is on if the bit is set, and off if the bit is clear. The new value of the word is set as follows:

New value = (Old value AND R1) EOR R0

The default condition is all indicators off.

Related APIs

None

Hourglass_Colours (SWI &406C6)

Sets the colours used to display the hourglass

On entry

R0 = new colour to use as colour 1 (&00BBGGRR, or -1 for no change)
R1 = new colour to use as colour 3 (&00BBGGRR, or -1 for no change)

On exit

R0 = old colour being used as colour 1
R1 = old colour being used as colour 3

Interrupts

Interrupts are undefined
Fast interrupts are enabled

Processor mode

Processor is in SVC mode

Re-entrancy

Not defined

Use

This call sets the colours used to display the hourglass. Alternatively you can use this call to read the current hourglass colours by passing parameters of -1.

The default colours are:

Value	Meaning
Colour 1	cyan
Colour 3	blue

This call is not available in RISC OS 2.

Related APIs

None

Example programs

The examples below illustrate the use of Hourglass_Percentage.

```
DoLoadAndProcess
        STMFD   r13!, {r0-r5, r14}
        MOV     r0, #OSFile_Load
        ADR     r2, Buffer
        MOV     r3, #0
        SWI     XOS_File
        BVS     ExitLoadAndProcess
        CMP     r4, #0
        BEQ     ExitLoadAndProcess
        SWI     XHourglass_On
        BVS     ExitLoadAndProcess
        ADR     r1, Buffer
        MOV     r2, #0
        ; Compute a constant, in R3, such that as the index
        ; in R2 goes from 0 to the maximum value, in R4, the
        ; result of (R2 * R3) DIV 2^24 goes from 0 to 100.
        ; R3 = (100 * 2^24) DIV R4.
        MOV     r5, #100 :SHL: 24       ; So we get a percentage
        MOV     r14, r4                 ; R3 := R5 DIV R4
        CMP     r14, r5, LSR #1
DivisionLoop1
        MOVLS   r14, r14, LSL #1
        CMPLS   r14, r5, LSR #1
        BLS     DivisionLoop1
        MOV     r3, #0
DivisionLoop2
        CMP     r5, r14
        SUBCS   r5, r5, r14
        ADC     r3, r3, r3
        MOV     r14, r14, LSR #1
        CMP     r14, r4
        BCS     DivisionLoop2
        ; R3 is now a simple constant
ProcessLoop
        MUL     r0, r2, r3
        MOV     r0, r0, ASR #24
        SWI     XHourglass_Percentage   ; Call with result
        LDRVCB  r0, [r1], #1
        BLVC    ProcessByte             ; May also return V set
        BVS     InternalError
        ADD     r2, r2, #1              ; Move the index
        TEQ     r2, r4
        BNE     ProcessLoop
FinishProcess
        SWI     XHourglass_Off
ExitLoadAndProcess
        STRVS   r0, [r13, #0]
        LDMFD   r13!, {r0-r5, pc}
InternalError
        MOV     r1, r0                  ; Preserve the actual error
        SWI     XHourglass_Off          ; Ignore possible error
        MOV     r0, r1                  ; Retore real error
        CMP     pc, #&80000000          ; Set V, to indicate an error
        B       ExitLoadAndProcess
```

Or in BBC BASIC V:

```
DEF PROCLoadAndProcess(Name$)
LOCAL Length%,Index%
LOCAL ERROR
SYS "OS_File",255,Name$,Buffer%,0 TO ,,,,Length%
IF Length%<>0 THEN
  SYS "Hourglass_On"
  ON ERROR LOCAL RESTORE ERROR:SYS "Hourglass_Off":ERROR ERR,REPORT$
  FOR Index%=0 TO Length%
    SYS "Hourglass_Percentage",(100*Index%) DIV Length%
    PROCProcessByte(Buffer%?Index%)
  NEXT Index%
  SYS "Hourglass_Off"
ENDIF
ENDPROC
```

Document information

Maintainer(s):	RISCOS Ltd <developer@riscos.com>

History:

Revision	Date	Author	Changes
1		ROL	Initial version