HowToCode7

COLLABORATORS			
	TITLE : HowToCode7		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 28, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

HowToCode7

1.1 HowToCode: Interrupts

```
               Legal Interrupt Handling for Amiga Demos
               ----------------------------------------

OS Legal interrupts are so easy to add and have such a very
tiny overhead that it is a real pity that so many people ignore
them...

Hardware interrupts are set with  AddIntServer
and removed with  RemIntServer .

The best way to show how to use interrupts on the Amiga is with
an example (thanks Bjorn!). I've pessimised it slightly (to make
it a little more readable) and fixed one bug!

INTB_VERTB  equ   5                ; for vblank interrupt
INTB_COPER  equ   4                ; for copper interrupt

CALL: MACRO
      jsr   _LVO\1(a6)
      ENDM


SystemOff:
      move.l   GfxBase,a6
      sub.l    a1,a1
      CALL     LoadView             ;Get default view
      CALL     WaitTOF
      CALL     WaitTOF
      CALL     OwnBlitter           ;Claim ownership of blitter
      move.l   $4.w,a6
      CALL     Forbid               ;Forbid multitasking
      moveq.l  #INTB_VERTB,d0       ; INTB_COPER for copper interrupt
      lea      VBlankServer(pc),a1
      CALL     AddIntServer         ;Add my interrupt to system list
      rts

SystemOn:
      move.l   $4.w,a6
```

```
      moveq.l  #INTB_VERTB,d0       ;Change for copper interrupt.
      lea      VBlankServer(pc),a1
      CALL     RemIntServer         ;Remove my interrupt
      CALL     Permit               ;Permit multitasking
      move.l   GfxBase,a6
      CALL     DisownBlitter        ;Give blitter back
      move.l   MyView,a1
      CALL     LoadView             ;Load original view
      rts

IntLevel3:

      movem.l  d2-d7/a2-a4,-(sp)    ; all other registers can be trashed
      ...
      movem.l  (sp)+,d2-d7/a2-a4

;     If you set your interrupt to priority 10 or higher then
;     a0 must point at $dff000 on exit.

      moveq    #0,d0                ; must set Z flag on exit!
      rts                           ;Not rte!!!

VBlankServer:
      dc.l  0,0                     ;ln_Succ,ln_Pred
      dc.b  2,0                     ;ln_Type,ln_Pri
      dc.l  IntName                 ;ln_Name
      dc.l  0,IntLevel3             ;is_Data,is_Code

IntName:dc.b "Dexion & SAE IntLevel3",0     ; :-)
      EVEN
;---------------------------------------------------------------------
```

where MyView is filled in immediately after graphics.library
have been opened:

```
   ..
   move.l   GfxBase,a1
   move.l   gb_ActiView(a1),MyView
   ...
```

1.2 exec.library/AddIntServer()

```
   NAME
      AddIntServer -- add an interrupt server to a system server chain

   SYNOPSIS
      AddIntServer(intNum, interrupt)
        -168       D0-0:4      A1

      void AddIntServer(ULONG, struct Interrupt *);

   FUNCTION
      This function adds a new interrupt server to a given server chain.
      The node is located on the chain in a priority dependent position.
      If this is the first server on a particular chain, interrupts will
```

```
   be enabled for that chain.

   Each link in the chain will be called in priority order until the
   chain ends or one of the servers returns with the 68000's Z condition
   code clear (indicating non-zero).  Servers on the chain should return
   with the Z flag clear if the interrupt was specifically for that
   server, and no one else.  VERTB servers should always return Z set.
   (Take care with High Level Language servers, the language may not
   have a mechanism for reliably setting the Z flag on exit).

   Servers are called with the following register conventions:

       D0 - scratch
       D1 - scratch

       A0 - scratch
       A1 - server is_Data pointer (scratch)

       A5 - jump vector register (scratch)
       A6 - scratch

       all other registers must be preserved

INPUTS
   intNum - the Paula interrupt bit number (0 through 14). Processor
     level seven interrupts (NMI) are encoded as intNum 15.
     The PORTS, COPER, VERTB, EXTER and NMI interrupts are
     set up as server chains.
   interrupt - pointer to an Interrupt structure.
     By convention, the LN_NAME of the interrupt structure must
     point a descriptive string so that other users may
     identify who currently has control of the interrupt.

WARNING
   Some compilers or assemblers may optimize code in unexpected ways,
   affecting the conditions codes returned from the function.  Watch
   out for a "MOVEM" instruction (which does not affect the condition
   codes) turning into "MOVE" (which does).

BUGS
   The graphics library's VBLANK server, and some user code, currently
   assume that address register A0 will contain a pointer to the custom
   chips. If you add a server at a priority of 10 or greater, you must
   compensate for this by providing the expected value ($DFF000).
```

1.3 exec.library/RemIntServer()

```
NAME
   RemIntServer -- remove an interrupt server from a server chain

SYNOPSIS
   RemIntServer(intNum, interrupt)
      -174        D0      A1

   void RemIntServer(ULONG,struct Interrupt *);
```

```
FUNCTION
   This function removes an interrupt server node from the given
   server chain.

   If this server was the last one on this chain, interrupts for this
   chain are disabled.

INPUTS
   intNum - the Paula interrupt bit (0..14)
   interrupt - pointer to an interrupt server node

BUGS
   Before V36 Kickstart, the feature that disables the interrupt
   would not function.  For most server chains this does not
   cause a problem.
```