cia

	COLLABORATORS		
	TITLE : cia		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		March 14, 2022	

	REVISION HISTORY		
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

cia

1.1 cia.doc

```
                    AbleICR()

                    AddICRVector()

                    RemICRVector()

                    SetICR()
```

1.2 cia.resource/AbleICR

```
                    NAME
 AbleICR -- Enable/disable ICR interrupts.

  SYNOPSIS
 oldMask = AbleICR( Resource, mask )
 D0                       A6          D0

 WORD AbleICR( struct Library *, WORD );

  FUNCTION
 This function provides a means of enabling and disabling 8520
 CIA interrupt control registers. In addition it returns the
 previous enable mask.

  INPUTS
 mask             A bit mask indicating which interrupts to be
                     modified. If bit 7 is clear the mask
                     indicates interrupts to be disabled. If
                     bit 7 is set, the mask indicates
                     interrupts to be enabled. Bit positions
                     are identical to those in 8520 ICR.

  RESULTS
 oldMask          The previous enable mask before the requested
```

changes. To get the current mask without
making changes, call the function with a
null parameter.

 EXAMPLES
Get the current mask:
    mask = AbleICR(0)
Enable both timer interrupts:
    AbleICR(0x83)
Disable serial port interrupt:
    AbleICR(0x08)

 EXCEPTIONS
Enabling the mask for a pending interrupt will cause an
immediate processor interrupt (that is if everything else is
enabled). You may want to clear the pending interrupts with

            SetICR()
             prior to enabling them.

 NOTE
The CIA resources are special in that there is more than one
of them in the system. Because of this, the C language stubs
in amiga.lib for the CIA resources require an extra parameter
to specify which CIA resource to use. The synopsys for the
amiga.lib stubs is as follows:

oldMask = AbleICR(Resource, mask)
D0   A6   D0

WORD AbleICR(struct Library *, WORD);

 SEE ALSO

            cia.resource/SetICR()

1.3   cia.resource/AddICRVector

                NAME
AddICRVector -- attach an interrupt handler to a CIA bit.

 SYNOPSIS
interrupt = AddICRVector(Resource, iCRBit, interrupt)
D0   A6   D0   A1

struct Interrupt *AddICRVector(struct Library *, WORD,
  struct Interrupt *);

 FUNCTION
Assign interrupt processing code to a particular interrupt bit
of the CIA ICR. If the interrupt bit has already been
assigned, this function will fail, and return a pointer to the
owner interrupt. If it succeeds, a null is returned.

This function will also enable the CIA interrupt for the given
ICR bit.

 INPUTS
iCRBit   Bit number to set (0..4).
interrupt   Pointer to interrupt structure.

 RESULT
interrupt   Zero if successful, otherwise returns a
                    pointer to the current owner interrupt
                    structure.

 NOTE
A processor interrupt may be generated immediatly if this call
is successful.

In general, it is probably best to only call this function
while DISABLED so that the resource to which the interrupt
handler is being attached may be set to a known state before
the handler is called. You MUST NOT change the state of the
resource before attaching your handler to it.

The CIA resources are special in that there is more than one
of them in the system. Because of this, the C language stubs
in amiga.lib for the CIA resources require an extra parameter
to specify which CIA resource to use. The synopsys for the
amiga.lib stubs is as follows:

```
interrupt = AddICRVector( Resource, iCRBit, interrupt )
D0                        A6        D0      A1

struct Interrupt *AddICRVector( struct Library *, WORD,
  struct Interrupt *);
```

WARNING

Never assume that any of the CIA hardware is free for use.
Always use the AddICRVector() function to obtain ownership
of the CIA hardware registers your code will use.

Note that there are two (2) interval timers per CIA. If
your application needs one of the interval timers, you
can try to obtain any one of the four (4) until AddICRVector()
succeeds. If all four interval timers are in-use, your
application should exit cleanly.

If you just want ownership of a CIA hardware timer, or register,
but do not want interrupts generated, use the AddICRVector()
function to obtain ownership, and use the
              AbleICR()
               function
to turn off (or on) interrupts as needed.

Note that CIA-B generates level 6 interrupts (which can degrade
system performance by blocking lower priority interrupts). As
usual, interrupt handling code should be optimized for speed.

```
Always call
                RemICRVector()
                 when your code exits to release
ownership of any CIA hardware obtained with AddICRVector().
```

  SEE ALSO

                cia.resource/RemICRVector()
                ,
                cia.resource/AbleICR()

1.4 cia.resource/RemICRVector

```
                    NAME
RemICRVector -- Detach an interrupt handler from a CIA bit.

 SYNOPSIS
RemICRVector( Resource, iCRBit, interrupt )
              A6         D0       A1

void RemICRVector( struct Library *, WORD, struct Interrupt *);

 FUNCTION
Disconnect interrupt processing code for a particular
interrupt bit of the CIA ICR.

This function will also disable the CIA interrupt for the
given ICR bit.

 INPUTS
iCRBit          Bit number to set (0..4).
interrupt       Pointer to interrupt structure.

 RESULT

 NOTE
The CIA resources are special in that there is more than one
of them in the system. Because of this, the C language stubs
in amiga.lib for the CIA resources require an extra parameter
to specify which CIA resource to use. The synopsys for the
amiga.lib stubs is as follows:

RemICRVector( Resource, iCRBit, interrupt )
              A6         D0       A1

void RemICRVector( struct Library *, WORD, struct Interrupt *);

 SEE ALSO

                cia.resource/AddICRVector()
```

1.5 cia.resource/SetICR

```
                  NAME
SetICR -- Cause, clear, and sample ICR interrupts.

 SYNOPSIS
oldMask = SetICR( Resource, mask )
D0                 A6        D0

WORD SetICR( struct Library *, WORD );

 FUNCTION
This function provides a means of reseting, causing, and
sampling 8520 CIA interrupt control registers.

 INPUTS
mask            A bit mask indicating which interrupts to be
                    effected. If bit 7 is clear the mask
                    indicates interrupts to be reset.  If bit
                    7 is set, the mask indicates interrupts to
                    be caused. Bit positions are identical to
                    those in 8520 ICR.

 RESULTS
oldMask         The previous interrupt register status before
                    making the requested changes.  To sample
                    current status without making changes,
                    call the function with a null parameter.

 EXAMPLES
Get the interrupt mask:
    mask = SetICR(0)
Clear serial port interrupt:
    SetICR(0x08)

 NOTE
The CIA resources are special in that there is more than one
of them in the system. Because of this, the C language stubs
in amiga.lib for the CIA resources require an extra parameter
to specify which CIA resource to use. The synopsys for the
amiga.lib stubs is as follows:

oldMask = SetICR( Resource, mask )
D0                 A6        D0

WORD SetICR( struct Library *, WORD );

***WARNING***

Never read the contents of the CIA interrupt control registers
directly.  Reading the contents of one of the CIA interrupt
control registers clears the register.  This can result in
interrupts being missed by critical operating system code, and
other applications.

 EXCEPTIONS
```

Setting an interrupt bit for an enabled interrupt will cause
an immediate interrupt.

 SEE ALSO

            cia.resource/AbleICR()