```
title   8088/6509 interprocess communications rom
.xall
;      ROM on PCB                                  1
;------------------------------------------
;      6525 equates     Tri Port
;------------------------------------------
cia_pra  equ 20h ;port register A (data)
cia_prb  equ 21h ;port register B (com/control)
cia_prc  equ 22h ;port register C
cia_ddra equ 23h ;data direction register A
cia_ddrb equ 24h ;data direction register B
cia_ddrc equ 25h ;data direction register C

pbbus1    equ 01h ;prb0
pbbus2    equ 02h ;prb1
pbsem88   equ 04h ;prb2
pbsem65   equ 08h ;prb3
pbrtn     equ 40h ;prb6

semlo     equ pbrtn ;acknowledge low
semhi     equ pbrtn+pbsem88 ;acknowlegde hi val

pcbus     equ 20h ;bit to free/claim bus pc5


;------------------------------------------
;     8259A equates
;------------------------------------------
inta00    equ 00h
inta01    equ 01h


;--------------------------------------------
;    interrupt definitions
;--------------------------------------------
iwait equ 40h ;100h, go rom, free bus, wait


page
;------------------------------------------
;     Macro Definitions
;
;          note: al is clobbered
;------------------------------------------

acklo macro
  mov  al,semlo
  out  cia_prb,al
  nop
endm


ackhi macro
  mov  al,semhi
  out  cia_prb,al
  nop
endm


waithi macro wwww
wwww:
  in   al,cia_prb
  test al,pbsem65
  jz   wwww
endm
```

```
waitlo macro llll
llll:                               2
  in   al,cia_prb
  test al,pbsem65
  jnz  llll
endm


dirin macro
  mov  al,0
  out  cia_ddra,al
endm


dirout macro
  mov  al,0ffh
  out  cia_ddra,al
endm


getbus macro gggg
  in   al,cia_prb
  test al,pbbus1
  jz   gggg
endm


frebus macro
  lock nop
  lock mov  al,0ffh-pcbus
  lock out  cia_prc,al
  lock nop
  lock nop
  lock nop
  lock or   al,pcbus
  lock out  cia_prc,al
endm
page
;------------------------------------------
;     ram data definitions
;------------------------------------------


;------------------------------------------
; offsets to data/code in ram
; eliminates relocation
;------------------------------------------


start_code equ  0000
warm       equ  0008h
warmh      equ  0009h
ipcbufr    equ  000ah
ipctab     equ  001ah
int7       equ  0007h
int7seg    equ  int7*4+2

page
;--------------------------------------------
;     rom code and data
;--------------------------------------------

thisseg equ 0f000h   ;hard value for this segno

progseg segment
```

```
assume cs:progseg
 org 0f000h
  jmp rqst900            ;return job after dma
```

```
;------------------------------------------------
; Issue IRQ request to 6509
;
;   enter:  cl = command byte
;           ipcbufr = holds input param bytes
;
;   exit:   ipcbufr = holds output param bytes
;------------------------------------------------
public rqster
rqster proc far
  push ax                ;save regs
  push bx
  push cx
  push dx
  push si
  push es
  push ds
  mov  ax,0000h           ;get segment of ipctab from int7
  mov  ds,ax
  mov  ax,ds:int7seg      ;get segment of ipctab from int7
  mov  ds,ax                                         vector
;
  mov  al,cl            ;set dl=#bytes to send
  and  al,7fh           ;    dh=#bytes to receive
  mov  bl,06
  mul  bl
  mov  si,ax
  mov  dx,ds:ipctab[si]
;
;   initiate IRQ to 6509, sending cmd byte
;
rqst000:
  in   al,cia_prb        ;test sem6509
  test al,pbsem65
  jnz  rqst000           ;locked out....
  or   al,pbsem88        ;lock sem8088
  out  cia_prb,al
  nop
  in   al,cia_prb        ;check for collision
  test al,pbsem65
  jz   rqst010           ;none!
  acklo                  ;locked out, clear sem80
  jmp  rqst000           ;and try again....
;
rqst010:
  dirout                 ;port dir=out
  mov  al,cl             ;write cmd to port
  out  cia_pra,al

rqst020:
  in   al,cia_prb
  and  al,0ffh-pbrtn
  out  cia_prb,al        ;CAUSE IRQ (lo->hi trans)
  nop
  nop
  nop
  or   al,pbrtn
  out  cia_prb,al
  waithi rqst030         ;sem6509->hi (irq recvd)
  dirin                  ;port dir=in
  mov  si,0
```

```
  inc  dl
  jmp  rqst120
;
;  send parameter bytes to 6509
```

```
;
rqst100:
  dirout                    ;port dir=out
  mov  al,ipcbufr[si]
  out  cia_pra,al           ;write data to port
  ackhi                     ;sem8088->hi (data ready)
  waithi rqst110            ;sem6509->hi (data rcvd)
  dirin                     ;port dir=in
  inc  si
rqst120:
  dec  dl                   ;decrease count
  jz   rqst200              ;no more to send...
  acklo                     ;sem8088->lo (ack)
  waitlo rqst130
  jmp  rqst100              ;and repeat...
;
rqst200:
  test cl,80h               ;need to give up data bus?
  jz  rqst210               ;no....
  frebus                     ;give up bus
  acklo                      ;signal to do cmd
 rqstlp:
  jmp  rqstlp               ;wait....
;
;   receive data bytes from 6509
;
rqst210:
  acklo                     ;signal to do cmd
  waitlo rqst220            ;sem6509->hi (data avail)
  mov  si,0
  inc  dh                   ;get return bytes, if any
  jmp  rqst350
rqst310:
  ackhi                     ;sem8088->hi (rdy to recv)
  waithi rqst320            ;sem6509->hi (data available)
  in   al,cia_pra           ;read data from port
  mov  ipcbufr[si],al
  acklo                     ;sem8088->lo (data recvd)
  waitlo rqst330            ;sem6509->lo (ack ack)
  inc  si
rqst350:
  dec  dh                   ;decrease count
  jnz  rqst310              ;more...
;
rqst400:
  pop  ds                   ;restore ds,es
  pop  es
  pop  si
  pop  dx
  pop  cx
  pop  bx
  pop  ax
  ret

rqst900:
  pop dx          ;pop ret adr
  pop dx
  pop dx          ;pop # bytes returned
  acklo           ;terminating acknowledge
  pop si
  pop ds
  pop es
  pop dx
  pop cx
  pop bx
  pop ax
  pop ax        ;pop uneeded irq ret
  pop ax
  popf
  jmp  rqst400        ;done
rqster endp
page
;-------------------------------------------
;  Service an IRQ from the 6509
;
;-------------------------------------------
server proc near
  push ax             ;save regs
  push bx
  push cx
  push dx
  push es
  push ds
  push si
  mov  ax,0000
  mov  ds,ax          ;set up ds to int area
  mov  ax,ds:int7seg   ;get segment of ipctab
  mov  ds,ax
;
;   decode the command
;
  in   al,cia_pra     ;read cmd byte
  and  al,7fh
  mov  bl,06          ;calculate index to entry
  mul  bl
  mov  si,ax          ;get jumpaddr,param counts
  mov  cx,ds:ipctab+2[si]  ;cx=jump address offset
  mov ds:start_code+4,cx ;move offset to ram jump vector
  mov cx,ds:ipctab+4[si] ;cx=jump address segment
  mov ds:start_code+6,cx ;move segment to ram jumpvector
  mov dx,ds:ipctab[si]  ;dl=#ins, dh=#outs
  ackhi               ;sem8088-> hi
  waitlo serv050      ;sem6509->lo (ack)
  mov  si,0
  inc  dl
  jmp  serv130
;
;    get input parameter bytes
;
serv100:
  acklo               ;sem8088->lo (ack ack)
  waithi serv110      ;sem6509->hi (data available)
  in   al,cia_pra     ;read data
  mov  ipcbufr[si],al ;and store it
  ackhi               ;sem8088->hi (data recvd)
  waitlo serv120      ;sem6509->lo (ack)
  inc  si
serv130:
  dec  dl             ;decrease count
  jnz  serv100        ;more...
;
;    process command
;
serv200:
  push dx             ;save return count (dh)
  push cs              ;save seg to return to
  mov  dl,low  offset serv220
  mov  dh,high offset serv220
```

5

```
  push dx                    ;push return on stack
  int 7                      ;gone!
serv220:
  pop  dx                    ;restore dh
  cmp  dh,0
  je   serv400               ;no return params...
  mov  si,0
;
;    send return parameter bytes
;
serv300:
  acklo                      ;sem8088->lo (ack ack)
  waithi serv310             ;sem6509->hi (rdy to recv)
  dirout                ;port dir=out
  mov  al,ipcbufr[si]
  out  cia_pra,al            ;send data byte
  ackhi                      ;sem8088->hi (data rdy)
  waitlo serv320             ;sem6509->lo (data recvd)
  dirin                      ;port dir=in
  inc  si
  dec  dh                    ;decrease count
  jnz  serv300               ;more...
;
serv400:
  cli
  acklo                      ;terminating acknowledge
  pop  si                    ;restore regs
  pop  ds
  pop  es
  pop  dx
  pop  cx
  pop  bx
  pop  ax
  iret
server endp

xret:
xerr:
  ret

page
;-------------------------------------------
;  startup:  do initialization
;-------------------------------------------
startf label far
start:
  cli                        ;disable interrupts
  mov  sp,0f000h             ;initialize stack
  mov  al,40h
  out  cia_prb,al            ;prb=40h
  mov  al,0ffh
  out  cia_pra,al            ;pra=0ffh
  out  cia_prc,al            ;prc=0ffh
  inc  al
  out  cia_ddra,al           ;ddra=in
  mov  al,44h
  out  cia_ddrb,al           ;pb4,pb6=out
  mov  al,20h
  out  cia_ddrc,al           ;pc5=out
  frebus                     ;0->1 transition on pc5
;
;    8259a initialization
;
  mov  al,1bh                ;icw1: level, sngl, icw4
  out  inta00,al
  mov  al,8                  ;icw2: intrpt address
```

6

```
 out  inta01,al
 mov  al,1            ;icw4: 8086 mode
 out  inta01,al
 mov  al,0feh         ;ocw:  inhibit I7-I1
 out  inta01,al
 sti
self:
 jmp  self
```

7

```
page

;----------------------------------------------
;  6509-gen'd irq handler:
;           cold irq=>do cold start
;           warm irq=>do server, return to
;                     requester code.
;----------------------------------------------
intrpt:
  push ax             ;save ax, ds
  push ds
  mov  ax,0000
  mov  ds,ax
  mov  ax,ds:int7seg
  mov  ds,ax

  mov  al,20h         ;eoi to 8259A
  out  inta00,al

  in   al,cia_prb     ;6509 off bus?
  test al,pbbus1
  jz   quit           ;nope...
  in   al,cia_prb     ;8088 on bus?
  test al,pbbus2
  jz   nstart         ;yes!
quit:
  pop  ds
  pop  ax
  pop  ax              ;clear off uneeded iret
  pop  ax
  pop  ax
  sti
  frebus
quitl:
  jmp  quitl          ;free bus and sit


nstart:
  mov  al,0           ;bsyclk low
  out  cia_prc,al
  mov  al,0ffh        ;test for warm/cold start
  xor  al,ds:warm
  xor  al,ds:warmh
  jz   gowarm
  mov ax,0a55ah
  mov  ds:warm,al     ;cold start: set warm bits
  mov  ds:warmh,ah
  pop  ds
  pop  ax
  pop  ax              ;pop uneeded iret
  pop  ax
  popf                 ;reenable ints
  int 7         ;jmp to os entry

gowarm:
  pop  ds
  pop  ax
```

```
  jmp  server            ;server rets to finish request

progseg ends
```

8

```
page

;rstseg segment
;assume cs:rstseg
;org 0fff0h
;
; jmp startf
;rstseg ends

end
```