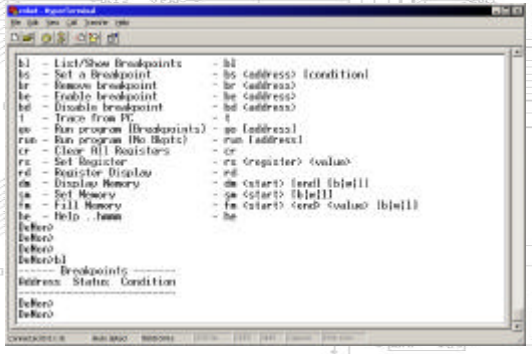
2BA4

HyperTerminal



2BA4

Setting up System Services (4)

- The Handler Code:
  - Make room on stack.

```
* Make room on stack
HL1111      clr.l       -(sp)
* Save d0 and a0
            movem.l     d0/a0,-(sp)
* Get old pc
            move.l      14(sp),a0
* Add 2 to old pc
            addq.l      #2,14(sp)
* Get instruction
            move.w      (a0),d0
* More on next page
```


2BA4

Setting up System Services (5)

```
* From last page
* Extract System Call number
* Simple version – only handles 16 calls
            andi.l      #$f,d0
* Save d0 and a0
            movem.l     d0/a0,-(sp)
* Multiply by 2
            add.l       d0,d0
* Transfer d0 to a0
            move.l      d0,a0
* Get address of system call
            move.w      CALLT(a0),10(sp)
* Restore do and a0
            movem.l     (sp)+,d0/a0
* Return to system routine
            rts
```


2BA4

A Trivial System Service Example (one)

```
* Macro to aid in calling a system service:

CALL        MACRO       CALLNO
            dc.w        $f000 + CALLNO
            ENDM
```

2BA4

A Trivial System Service Example (two)

```
* System call 4095 sets d0 to 4095
* CALLT + 2 * 4095 should point to SS4095
* Save save registers (redundant in this example)
SS4095      movem.l     d1-d7/a0-a6,-(sp)
* Set d0 equal to 4095
            move.l      #4095,d0
* Restore register
            movem.l     (sp)+,d1-d7/a0-a6
* Restore do and a0
            movem.l     (sp)+,d0/a0
* Return to caller
            rts
```


2BA4

A Trivial System Service Example (three)

*** To call SS4095**

CALL #4095

2BA4

Required System Services

- 0 **SDEV** **S**et active i/o **DEV**ice
- 1 **IACIA** Initialise **ACIA**
- 2 **STATUS** **STATUS** of active i/o device
- 3 **RACIA** Read **ACIA**
- 4 **WACIA** **W**rite **ACIA**
- 5 **TPCOM** **T**rans**P**arent **COM**unication
- 6 **WRSTR** **WR**ite **STR**ing
- 7 **WRHEX** **WR**ite **HEX**adecimal
- 8 **RHEX** Read **HEX**adecimal
- 9 **EXIT** **EXIT** to monitor

2BA4

TPCOM TransParent COMunication

- TPCOM supports transparent link maintenance
- No input parameter
- Behaviour:
  - It alternately checks both ACIA's for incoming characters.
  - Characters input on $IODev_1$ send to $IODev_2$ (except cntl-X)
  - The routine exits:
    - Cnt-x is received on $IODev_1$
    - Character received on $IODev_2$
- On Return:
  - 0 -> C if cntl-X received from $IODev_1$
  - 1 -> C
  - Char -> d0 if char. Received from $IODev_2$

2BA4

TPCOM (One)

```
TPCOM0      btst        #3, IODEV1+IOSTAT
            beq.s       TPCOM2
            btst        #4, IODEV2+IOSTAT
            beq.s       TPCOM2
            move.b       IODEV1,d0
            andi.l      #$7F,d0
* Is it a cntl-X?
            cmpi.b      #$18,d0
            bne          TPCOM1
*If so, clear carry and…
            andi.b      #$fe,1(sp)
*…return
            rte
* See next page
```


2BA4

TPCOM (Two)

```
* Continued from previous page.
TPCOM1      move.b      d0,IODEV2
TPCOM2      btst        #3,IODEV2+IOSTAT
            beq.s       TPCOM0
            btst        #4,IODEV1+IOSTAT
            beq.s       TPCOM0
            move.b       IODEV2,d0
            andi.l      #$7F,d0
* Set carry and…
            ori.b       #$01,1(sp)
*…return
            rte
```