ISTITUTO NAZIONALE DI FISICA NUCLEARE
Laboratori Nazionali di Frascati

LNF-81/49(NT)
1 Luglio 1981

V. Rossi: COLLEGAMENTO SERIALE TRA CALCOLATORI DEI LABORATORI NAZIONALI DI FRASCATI.

Servizio Documentazione
dei Laboratori Nazionali di Frascati

INFN - Laboratori Nazionali di Frascati
Servizio Documentazione

LNF-81/49(NT)
1 Luglio 1981

V. Rossi: COLLEGAMENTO SERIALE TRA CALCOLATORI DIGITAL DEI LNF.

1. - INTRODUZIONE.

Descriviamo un collegamento realizzato tra il PDP 11/45 del Gruppo Camera a Bolle e il VAX dei Laboratori Nazionali di Frascati.

I due calcolatori si trovano ad una distanza di circa 300 m.

La parte "hardware" del collegamento è realizzata mediante due interfacce tipo DL 11[1] collegate fra loro da un cavo. Il cavo è terminato, dalla parte del PDP 11, da disaccoppiatori ottici per evitare problemi di masse.

Il VAX "vede" il PDP come un "terminale" standard gestito dalla propria DL 11.

La gestione della DL 11 del PDP/45 è affidata a due programmi con "devices handler" scritti appositamente. Pertanto tale "device" non è conosciuto dal sistema operativo del PDP.

Il primo programma permette di utilizzare un qualsiasi terminale del PDP, come un terminale interativo del VAX, il secondo gestisce la trasmissione nei due sensi di files RSX 11 da un qualsiasi "device" dei due calcolatori (ad esempio da nastro del PDP al disco del VAX).

I programmi sono scritti tenendo conto che il PDP 11 è gestito da un sistema RSX 11. Comunque è facile operare sotto sistemi operativi diversi (ad esempio RSX 11 M).

In questo modo la gestione non dipende dal tipo di calcolatore "host"; programmi sostanzialmente identici hanno permesso un collegamento con il PDP 11/34 dello stesso centro.

I programmi rispondono ai seguenti requisiti: una gestione veloce che impegni al minimo le risorse della CPU (il PDP 11 gestisce il PEPR[2,3] ed è quindi impegnato da questo dispositivo); inoltre devono avere una piccola occupazione di memoria.

2. - PROGRAMMA DI COLLEGAMENTO INTERATTIVO.

Tale programma di cui riportiamo la lista (allegato 1) si basa sulla seguente idea: l'attivazione del programma da parte di un qualsiasi terminale del PDP permette di riconoscere l'indirizzo "hardware" del terminale stesso. Questo consente di connettere all'indirizzo della DL 11 una "interrupt service routine" (ISR) che trasferisce un carattere dal "buffer" di quest'ultima al "buffer" del terminale identificato. Per una corretta ricezione dei caratteri trasmessi è sufficiente che la "baud rate" dei terminali dei due calcolatori sia la stessa (nel nostro caso 4500 baud).

Il terminale del PDP 11 è un loop di attesa di lettura di un carattere mediante una QIOW "no echo" con "buffer size" di un carattere.

Se quindi un carattere viene impostato sul terminale del PDP 11 esso viene letto mediante la QIOW e trasmesso al "trasmitter data buffer" della DL 11.

Il VAX ricevendo il carattere dalla sua DL 11 imposta la sua eco che compare sul "reciver data resister" della DL 11 del PDP 11. L'ISR provvederà a far comparire questo carattere-eco del VAX sul terminale del PDP.

E' necessaria una gestione particolare di alcuni caratteri di controllo. Precisamente CONTROL-C comporta una interazione con il sistema operativo del PDP. CNTRL-Q, CNTRL-S, CNTRL-O non sono accompagnati da interrupt (per il RSX-11 D e quindi non sono "letti" dalla QIOW).

Per questa ragione tutti i caratteri speciali ottenuti con CNTRL-CARATTERE possono essere ottenuti con la seguenza ↑ (up-arrow) e poi CARATTERE.

La fine del programma di collegamento si ottiene impostando il carattere "bell" cioè CONTL-G oppure ↑ (up-arrow) e poi G.

3. - TRASFERIMENTO DI FILES TRA DEVICES DEI DUE CALCOLATORI.

Questo programma (allegato 2) gestisce la trasmissione di file RSX 11 tra due generici "devices" dei due calcolatori.

Quando il programma viene attivato il "default" del PDP è il terminale "TI:" e il "default host computer" è il VAX (Per definire come "host" il PDP basta scrivere la stringa "/PDP" ; per ritornare al VAX, la stringa "/VAX").

Quando viene scritta una generica altra stringa di caratteri sul terminale del PDP essa viene analizzata da una CSI$[4]. Se la stringa non contiene il simbolo "==" (che implica come vedremo, trasmissione di files) la stringa viene trasmessa al VAX.

La risposta del VAX viene "bufferizzata" da una ISR descritta in seguito e inviata al terminale del PDP sotto controllo del sistema operativo. In questo caso ovviamente la "baud rate" del terminale può essere qualsiasi.

Come si vedrà più appresso la ricezione dei caratteri "$BLANK" segnala la fine della trasmissione da parte del VAX e ciò riporta il terminale del PDP in una fase di attesa di lettura di una nuova stringa.

Impostando CNTRL-Z sul terminale si esce dal programma.

Se viene scritta una stringa che contiene il simbolo "=" il comando implica un trasferimento di files secondo lo schema " out-file = in-file ".

In questo caso il file che si riferisce al PDP viene gestito dalle "routines" di sistema definite sul "RSX-11 I/0 OPRATION MANUAL"(4).

Nello stesso tempo viene trasmessa al VAX una istruzione MCR PIP rispettivamente nella forma MCR PIP TI: = in-file per la trasmissione del VAX verso il PDP e MCR PIP out-file = TI: per la trasmissione dal PDP al VAX.

La ricezione di caratteri di eco sulla DL 11 del PDP permette la sincronizzazione e il controllo del trasferimento.

L'ISR è in questo caso più complessa ed è diversa decondo il verso del trasferimento. Precisamente se la trasmissione dei dati è verso il "VAX" l'ISR provvede ad inviare un nuovo carattere ogni volta che viene ricevuta una eco (un trattamento particolare è riservato al carattere TAB che ha una eco di n-blank).

Se la trasmissione è verso il PDP il dato ricevuto è semplicemente messo in un "buffer", la ricezione di caratteri speciali "carriage return" e "line feed" avverte il sistema del PDP che una linea (record) è stata ricevuta.

Questi caratteri sono accompagnati da un "global flag" e avvertono il sistema del PDP che deve provvedere al trasferimento della linea (record) al "device" precedentemente definito. Nello stesso tempo la ISR provvede a trasmettere un carattere CONTRL-S al VAX per interrompere la trasmissione dei dati in questa fase.

Un secondo buffer permette di ricevere eventuali caratteri trasmessi prima che il devices del VAX sia effettivamente in "stop".

Una volta avvenuto il trasferimento dal "buffer" di ricezione ad un "buffer" di trasferimento, viene trasmesso al VAX il carattere CONTL-Q e il trasferimento ricomincia.

La fine del trasferimento è identificata dalla ricezione dei caratteri "$BLANK" come ultimi caratteri di una trasmissione. Se dopo 5 millisecondi di attesa non sono stati trasmessi altri caratteri il "file" trasmesso viene considerato finito.

RINGRAZIAMENTI.

Ringrazio tutte le persone del Centro di calcolo dei LNF per i preziosi suggerimenti ed aiuti. In particolare la dott. M. L. Ferrer ed il dott. C. Serio per l'assistenza software e il sig. O. Ciaffoni per le prove hardware.

Ringrazio, inoltre, i sigg. D. Fabbri e L. Sangiorgio del Gruppo PEPR che hanno curato la parte hardware del PDP 11/45.

BIBLIOGRAFIA.

(1) - Digital PDP 11 peripherals handbook. Programming. Interfacing.

(2) - P. Allen et al., A set of on-line routines to guide PEPR in measuring bubble chamber film, Frascati report LNF-79/13 (1979).

(3) - P. Allen et al., Basic data acquisitive routines for PEPR, Frascati report LNF-79/18 (1979).

(4) - RSX-11 I/0 operations reference manual, DEC-11, OMFSA-B-D.

APPENDICE 1.

```
.TITLE TQI
.MCALL  CALL,QIOW$S,EXIT$S
LF=12
CR=15
SL=57
CC=136
INT=360
DEV=176060
TQI::   MOV     .CRTSK,R4 ;TI IDENTIFICATION
        MOV     A.TI(4),R4 ;PUD ADRS
        MOV     U.DA(4),R4 ;TERM ADRS ON R4
        MOV     R4,TER
        ADD     #6,TER
        MOV     #INT,R0 ;CONNECT DEV INTERRUP
        MOV     #SUB,R1
        CLR     R2
        MOV     #250,R3
        CALL    @#..CINT
        BCC     S1
        QIOW$S  #IO.WLB,#5,#1,,,,<#EMS,#NEM,#40>
        BR      EXI
SUB:    MOVB    @#DEV+2,@TER ;ISR
        JMP     @#..INTX
S1:     QIOW$S  #IO.WLB,#5,#1,,,,<#IMS,#NIM,#40>
        BIS     #100,@#DEV      ; DEVICES INTERRUPT ABILITATION
        CLR     @#DEV+2         ; CLEAR DEV. REC. BUFFER
S2:     QIOW$S  #IO.RLB+24.,#5,#1,,,,<#BY,#1>
        CMPB    #CC,BY
        BNE     S3
        COM     ISW
        TST     ISW
        BNE     S2
S3:     TST     ISW
        BEQ     S4
        BICB    #340,BY
        CLR     ISW
S4:     CMPB    #7,BY ;TEST FOR EXIT
        BEQ     EXT
        MOVB    BY,@#DEV+6 ;TRASMIT
;       MOVB    BY,@#TAD+6 ;ECHO
        BR      S2
EXT:    BIC     #100,@#DEV ;DEVICE INTERRUPT DEABILTATION
EXI:    MOV     #INT,R0
        CALL    @#..DINT
        QIOW$S  #IO.WLB,#5,#1,,,,<#FMS,#NFM,#40>
S6:     EXIT$S
TER:    .BLKW   1
ISW:    .BLKW   1
EMS:    .ASCII  /INTERRUPT CONNESSO. RIPROVA !/
NEM=.-EMS
IMS:    .ASCII  /BUON COLLEGAMENTO CON IL VAX !/
        .ASCII  <CR><LF>/ PREFERIRE ^ E POI CARATTERE/
        .ASCII  <CR><LF>/ INVECE DI CNTRL + CARATTERE/
        .ASCII  <CR><LF>/PER SCOLLEGARSI DARE ^ E POI G/
NIM=.-IMS
FMS:    .ASCII  /FINE DEL COLLEGAMENTO. CIAO!/<CR>
NFM=.-FMS
BY:     .BLKB   1
        .EVEN
.END    TQI
```

APPENDICE 2.

```
.TITLE  TRV
.MCALL  QIOW$,DIR$,CLEF$S,SETF$S,WTSE$S,RDEF$S,MRKT$S,EXIT$S
.MCALL  FSRSZ$,FDBDF$,FDAT$A,FDOP$A,OPEN$R,OPEN$W,CLOSE$
.MCALL  CSI$1,CSI$2,GET$,PUT$
        FSRSZ$  1
FOU:    FDBDF$
        FDAT$A  R.VAR,FD.CR!FD.BLK,,-1,-1
        FDOP$A  2,CBL+C.DSDS
.MACRO  RED
        MOV     #IO.RLB,QIA+2
        MOV     #IBT,QIA+14
        MOV     #100,QIA+16
        DIR$    #QIA
.ENDM
.MACRO  WRI     X,Y,Z
        MOV     #IO.WLB,QIA+2
        MOV     #X,QIA+14
        MOV     Y,QIA+16
        MOV     #Z,QIA+20
        DIR$    #QIA
.ENDM
.MACRO  MFI     Y
        .IRPC   X,Y
        MOVB    #''X,(0)+
.ENDM
.ENDM
.MACRO  WLC     X,?A,?B
A:      CMPB    X,LST
        BEQ     B
        CLEF$S  #FG
        WTSE$S  #FG
        BR      A
B:
.ENDM
TRV::   BIS     #100,@#DEV
        MOV     #INT,R0
        MOV     #ISR,R1
        CLR     R2
        MOV     #250,R3
        CALL    @#..CINT
        BCC     S0
        WRI     EMS,#NEM,40
        JMP     EXT+6
S0:     MRKT$S  #1,#100,#1,
        MOVB    #CZ,IBT
        MOVB    IBT,@#DEV+6
        WTSE$S  #1
        MOVB    #CQ,@#DEV+6
        CMPB    #ES,IBR
        BEQ     S01
        WRI     OMS,#NOM,40
        JMP     EXT
S01:    MOV     #2,CBL+C.DSDS
        MOV     #PIP+10,CBL+C.DSDS+2
        CLR     CBL+C.DIRD
        CLR     CBL+C.FILD
        OPEN$W  #FOU
        BCC     S1
        WRI     EOM,#NEO,40
        JMP     EXT
S1:     MOVB    #CQ,@#DEV+6
```

```
        MOV     #204,NCM
        WRI     IMS,#NIM,"$
        RED
        CMPB    #IE.EOF,IO
        BNE     S2
        JMP     EXT
S2:     TST     IO+2
        BEQ     S4
        CMP     #"/P,IBT
        BNE     S3
        MOV     #" P,PDP
        MOV     #"DP,PDP+2
        BR      S4
S3:     CMP     #"/V,IBT
        BNE     S5
        MOV     #" V,PDP
        MOV     #"AX,PDP+2
S4:     WRI     MSP,#NSP,40
        BR      S1
S5:     CLR     SWT
        MOV     #IBF,R0
        MOV     #IBT,R1
        MOV     IO+2,R2
        CALL    FILL
        CSI$1   #CBL,#IBF,IO+2
        BITB    #CS.EQU,CBL+C.STAT
        BEQ     S13
        COM     SWT
        CLOSE$  #FOU
        CSI$2   #CBL,INPUT
        MOV     CBL+C.FILD,IFI
        MOV     CBL+C.FILD+2,IFI+2
        CMP     #" V,PDP
        BEQ     S10
        JMP     S30
S10:    MOV     #IBT,R0
        MOV     #PIP,R1
        MOV     #14,R2
        CALL    FILL
        CALL    CMLD
        SUB     #IBT,R0
        MOV     R0,NCB
        CSI$2   #CBL,OUTPUT
        TST     CBL+C.FILD
        BNE     S11
        MOV     IFI,CBL+C.FILD
        MOV     IFI+2,CBL+C.FILD+2
S11:    OPEN$W  #FOU
        BCC     S12
        WRI     EOM,#NED,40
        JMP     S1
S12:    WRI     IBT,NCB,40
        MOV     NCB,IO+2
S13:    MOV     IO+2,R3
        CLR     NCM
        CLEF$S  #FG
        CALL    TRSM
        WTSE$S  #FG
        MOV     #204,NCM
        CLR     NCR
        MOV     #IBR,R1
```

```
        CLEF$S  #FG
S14:    WTSE$S  #FG
;       RED     ;CR WAIT
        MOV     #IBF,R0
        MOV     NLF,R3
        BLE     S17
S16:    MOVB    #LF,(0)+
        SOB     R3,S16
S17:    MOV     NCP,R2
        CALL    FILL
        ADD     NLF,R3
        CLR     NLF
        CLEF$S  #FG
        MOVB    #CQ,@#DEV+6
        TST     SWR
        BNE     S18
        MOV     #IBR,R1
S18:    CMPB    #ES,IBF-1(3)
        BNE     S20
        CLEF$S  #FG
        MRKT$S  #1,#10,#1,
        WTSE$S  #1
        CMP     #2,NCR
        BNE     S14
        TST     SWT
        BEQ     S19
        CLOSE$  #FOU
        JMP     S01
S19:    JMP     S1
S20:    TST     R3
        BGT     S21
        MOV     #1,R3
S21:    PUT$    #FOU,#IBF,R3
;       RED     ;CR WAIT
        JMP     S14
S30:    OPEN$R  #FOU
        BCC     S31
        WRI     EIM,#NEI,40
        JMP     S1
S31:    CSI$2   #CBL,OUTPUT
        MOV     #IBT,R0
        MOV     #PIP,R1
        MOV     #10,R2
        CALL    FILL
        CALL    CMLD
        MOVB    #'=,(0)+
        MOV     #PIP+10,R1
        MOV     #3,R2
        CALL    FILL
        SUB     #IBT,R0
        MOV     R0,R3
        WRI     IBT,R3,40
        CLEF$S  #FG
        CLR     NCM
S32:    CALL    TRSM
        CLEF$S  #FG
        GET$    #FOU,#IBF;#N1B
        WTSE$S  #FG
        CLEF$S  #FG
        CMPB    #IE.EOF,FOU+F.ERR
        BEQ     S33
```

```
        MOV     FOU+F.NRBD,R2
        MOV     #IBT,R0
        MOV     #IBF,R1
        CALL    FILL
        TST     R3
        BR      S32
S33:    MOVB    #CZ,IBT
        MOVB    IBT,@#DEV+6
        CLOSE$  #FOU
        JMP     S01
FILL:   CLR     R3
        TST     R2
        BEQ     SF2
        MOV     R0,-(SP)
SF0:    MOVB    (1)+,(0)
        CMPB    #BK,(0)+
        BEQ     SF1
        MOV     R0,R3
SF1:    SOB     R2,SF0
        SUB     (SP)+,R3
SF2:    RTS     PC
TRSM:   TST     R3
        BGT     ST2
ST1:    MOV     #1,R3
        MOVB    #BK,IBT
        BR      ST3
ST2:    TSTB    IBT-1(3)
        BNE     ST3
        DEC     R3
        BGT     ST2
        BR      ST1
ST3:    MOVB    #CR,IBT(3)
        MOVB    IBT,@#DEV+6
        RTS     PC
CMLD:   MOV     CBL+C.DEVD,R2
        BEQ     C01
        MOV     CBL+C.DEVD+2,R1
        CALL    FILL
        MOVB    #':,(0)+
C01:    MOV     CBL+C.DIRD,R2
        MOV     CBL+C.DIRD+2,R1
        CALL    FILL
        TST     CBL+C.FILD
        BNE     C02
        MOV     IFI,CBL+C.FILD
        MOV     IFI+2,CBL+C.FILD+2
C02:    MOV     CBL+C.FILD,R2
        MOV     CBL+C.FILD+2,R1
        CALL    FILL
        RTS     PC
EXT:    BIC     #100,@#DEV
        MOV     #INT,R0
        CALL    @#..DINT
        EXIT$S
ISR:    MOV     R0,-(SP)
        MOVB    @#DEV+2,R0
        BIC     #177600,R0
;* * * DEBUG * * *
        MOV     R0,TBF
        BGT     SUM
        ADD     #20,TBF
```

```
SUM:    ADD     #40,TBF
        MOVB    TBF,@#TER+6
        TST     R0
; * END DEBUG *
        BEQ     NIN
;       MOVB    R0,@#TER+6 ;ECHO
        CMPB    #LF,R0 ;LF?
        BNE     TCR
        TST     NCM
        BEQ     RBF
        TST     NCR
        BGT     TCS
        INC     NLF
        BR      NIN
TCS:    MOV     NCR,NCP
        CLR     NCR
        MOVB    #CS,@#DEV+6
        COM     SWR
        TST     SWR
        BNE     GIN
RBF:    MOV     #IBR-MBR-4,MBR+2 ;BUFR. RESET
        CLR     NCR
        BR      GIN
TCR:    CMPB    #CR,R0 ;CR?
        BNE     MBR
        MOV     #IBT-MBT-4,MBT+2 ;BUFT. RESET
        BR      NIN
MBR:    MOVB    R0,IBR ;CAR IN BUF
        INC     MBR+2 ;INCR BUFF ADR
        INC     NCR ;INCR NCR
        TST     NCM
        BEQ     MBT
        CMPB    #ES,LST ;ES?
        BNE     NIN
        CMPB    #BK,R0
        BNE     NIN
        CMP     #2,NCR
        BNE     NIN
        MOV     NCR,NCP
        BR      GIN
MBT:    MOVB    IBT,R0
        CMPB    #CZ,R0 ;CZ?
        BEQ     NIN
        CMPB    #TB,R0 ;TB?
        BNE     MBI
        BIT     #7,NCR
        BNE     NIN
MBI:    INC     MBT+2
        MOV     MBT+2,TBM+2
        ADD     #MBT-TBM,TBM+2
TBM:    MOVB    IBT,@#DEV+6
        BR      NIN
GIN:    MOV     R4,-(SP)
        MOV     #FG,R0
        MOV     @#.CRTSK+USR,R4
        CALL    @#..SEFN
        MOV     (SP)+,R4
NIN:    MOVB    R0,LST
        MOV     (SP)+,R0
        JMP     @#..INTX
DEV=176060
```

```
TER=176040
USR=60000
INT=360
TB=11
LF=12
CR=15
BK=40
CZ=32
ES=44
CS=23
CO=21
PR=76
ES=44
SL=57
FG=35.
CBL:    .BLKB   C.SIZE
QIA:    QIOW$   2,5,1,,IO,,<14,16,20>
IO:     .BLKW   2
IFI:    .BLKW   2
PIP:    .ASCII  /MCR PIP TI:=/
.EVEN
EMS:    .ASCII  /TRV -- INTERRUPT CONNESSO /
NEM=.-EMS
.EVEN
IMS:    .ASCII  /TRV> /
NIM=.-IMS-1
.EVEN
OMS:    .ASCII  /TRV -- MANCA IL COLLEGAMENTO/
NOM=.-OMS
MSP:    .ASCII  /TRV -- L'INPUT-FILE SI TROVA SUL VAX/
NSP=.-MSP
POP=MSP+NSP-4
.EVEN
EOM:    .ASCII  /TRV -- ERRORE SU OUTPUT FILE/
NEO=.-EOM
.EVEN
EIM:    .ASCII  /TRV -- ERRORE SU INPUT-FILE /
NEI=.-EIM
.EVEN
SWT:    .WORD   0
SWR:    .WORD   0
LST:    .WORD   0
NLF:    .WORD   0
TBF:    .WORD   0
.PSECT  DATA    LCL,RW,D,CON
NCT:    .BLKW   1.
IBT:    .BLKB   204
NIB=.-IBT
NCR:    .BLKW   1
NCP:    .BLKW   1
IBR:    .BLKB   410
NCB:    .BLKW   1
IBF:    .BLKB   204
NCM:    .WORD   0
.END    TRV
```