OCCAM: l'esempio concreto

seconda parte

di Luciano Macera

Continuando il discorso iniziato lo scorso mese, su questo numero commenteremo la rimanente parte del software di rete di ADPnetwork scritta in OCCAM. Inutile dirvi che per comprendere il funzionamento delle varie routine descritte è necessario tenere sott'occhio il listato pubblicato sull'articolo precedente nonché la descrizione a blocchi del numero di novembre ultimo scorso. Sempre e solo, come detto, con l'unico scopo di non perdervi nei meandri dei troppi canali e processi di cui parleremo. In più, pubblichiamo questo mese il file di «include» contenente le varie costanti e il «.PGM» che, come vedremo, alloca processi e canali del processore o dei processori utilizzati

Poche chiacchiere

Visto che il mese scorso a furia di discorsetti, introduzioncine, parentesucce e affini siamo riusciti a malapena a commentare solo il listato del processo Dispatcher, questo mese partiamo subito con il commento, iniziando per l'appunto dal processo successivo, ReceiverAmiga.

Rappresentato nel pallogramma di figura 1 (il listato, come detto, dovete cercarlo sul numero scorso) ha in pratica funzioni di buffer sia per i messaggi in partenza che per quelli in transito. Per la precisione si tratta di un «multibuffer» in quanto pacchetti di natura o lunghezza diversa vengono bufferizzati al suo interno in array differenti, in modo da dimensionare quest'ultimi ad hoc e, perché no, eventualmente creare corsie preferenziali per messaggi più importanti.

Ad esempio, i pacchetti di Ack (generati dai processi dispatcher delle varie macchine in rete), sono sicuramente più urgenti di qualsiasi altro tipo di pacchetto in quanto se non arriva in tempo, il mittente del messaggio (del quale appunto il Dispatcher del destinatario ha generato il pacchetto di Ack) provvederà a rispedirlo pensando che si sia perso per strada. Con conseguente aggravio sul traffico generale dei pacchetti, con ulteriori ritardi, Ack sempre più lenti e così via come un gatto che rincorre la coda che oltre a non afferrarla mai satura ben presto tutti i buffer e tutti i link della rete.

Quindi, precedenza assoluta agli Ack, sia in transito che in partenza. Localmente, poi, ad ogni scheda, è opportuno tenere i buffer più vuoti possibile (per evitare il riempimento totale e l'uccisione da parte del dispatcher dei pacchetti in arrivo): in pratica (sempre figura 1) accogliere principalmente le richieste del processo SenderLink che prende i pacchetti bufferizzati e li spedisce sulla rete.

Da qui il motivo dell'utilizzo, nel processo ReceiverAmiga (presente sul numero scorso), di un costrutto PRI ALT che in caso di più guardie verificate dà priorità alla prima (l'ordine è quello dato dal listato stesso). In ogni comando con guardia del processo ReceiverAmiga, il controllo (nella guardia, appunto) riguarda lo stato dei buffer stessi, vuoto, pieno, «mezzo pieno». Quest'ultimo caso è stato aggiunto per lasciare sempre almeno mezzo buffer ai pacchetti (non Ack) in transito su quel nodo. Vediamo, allora, caso per caso le varie alternative del comando PRI ALT (l'ordine, come detto, implica priorità). Nel primo caso troviamo la guardia:

```
(FrameCount > 0) OR (AckCount > 0) & GiveMe ? dummy
```

che letteralmente significa: «se FrameCount è maggiore di zero (buffer pacchetti lunghi non vuoto) oppure AckCount è maggiore di zero (buffer pacchetti di Ack non vuoto) e c'è una richiesta da parte del processo SenderLink di un nuovo pacchetto da spedire esegui le linee di codice seguenti il successivo SEQ». Chiaramente FrameCount è una variabile che contiene continuamente il numero di pacchetti bufferizzati nel buffer «normale» e AckCount in quello «speciale». Per moderare, poi, la prevaricazione prioritaria dei pacchetti di tipo Ack, il corpo della sequenza di comandi associata alla prima guardia si comporta in maniera «flip-flop» pescando una volta prima dal buffer degli Ack e la volta successiva prima in quello dei pacchetti nor-

mali. Va da sé che qualora uno dei due buffer fosse vuoto comunque parte il pacchetto presente in quello non vuoto. Tutto ciò è realizzato dalla variabile booleana swap che una volta vale TRUE e la volta successiva FALSE facendo eseguire alternativamente o il primo o il secondo ramo del comando IF.

Il resto del listato del processo ReceiverAmiga si commenta da sé: seguono le guardie d'ingresso per il buffer degli Ack, per il buffer normale con accesso per i pacchetti in transito e in ultimo (quindi con priorità più bassa e per di più con il buffer «dimezzato») sempre per il buffer normale, ma questa volta per i pacchetti in partenza. In quest'ultimo caso, il pacchetto viene completato di CRC calcolato sull'effettivo corpo (body) del messaggio trasportato dal pacchetto in partenza. Questo per sgravare quanto più possibile il processore dell'Amiga che oltre a implementare parzialmente la rete deve anche continuare a funzionare come computer per l'utente.

L'altro buffer

Il processo SenderAmiga (sempre in figura 1) bufferizza i messaggi in arrivo su quel nodo e li spedisce all'Amiga. Anche in questo caso troviamo una PRI ALT apparentemente funzionante al contrario: viene data priorità al riempimento del buffer invece che al suo svuotamento. Il motivo è molto semplice: è assolutamente necessario non bloccare mai il processo Dispatcher (eseguito sul transputer il quale è MOLTO più veloce del 68000 di cui è dotato l'Amiga) il quale svolge funzioni tanto per il nodo in questione quanto per tutti i nodi della rete (reinoltrando pacchetti in transito). Al punto che, l'abbiamo detto lo scorso mese, nel caso in cui il buffer di SenderAmiga fosse malauguratamente pieno il Dispatcher butta il pacchetto non bufferizzato e si rimette in attesa sul link esterno (verso il nodo precedente nell'architettura di ADPnetwork). In pratica il processo SenderAmiga preleva comunque la richiesta di inserimento da parte del Dispatcher rispondendogli sul canale OK con un valore TRUE se il pacchetto è stato bufferizzato, FALSE se lo spazio non c'era. Riallacciandoci brevemente al commento dello scorso mese, nel primo caso il Dispatcher genera se necessario l'Ack del messaggio effettivamente arrivato (si presume che una volta bufferizzato, prima o poi l'Amiga

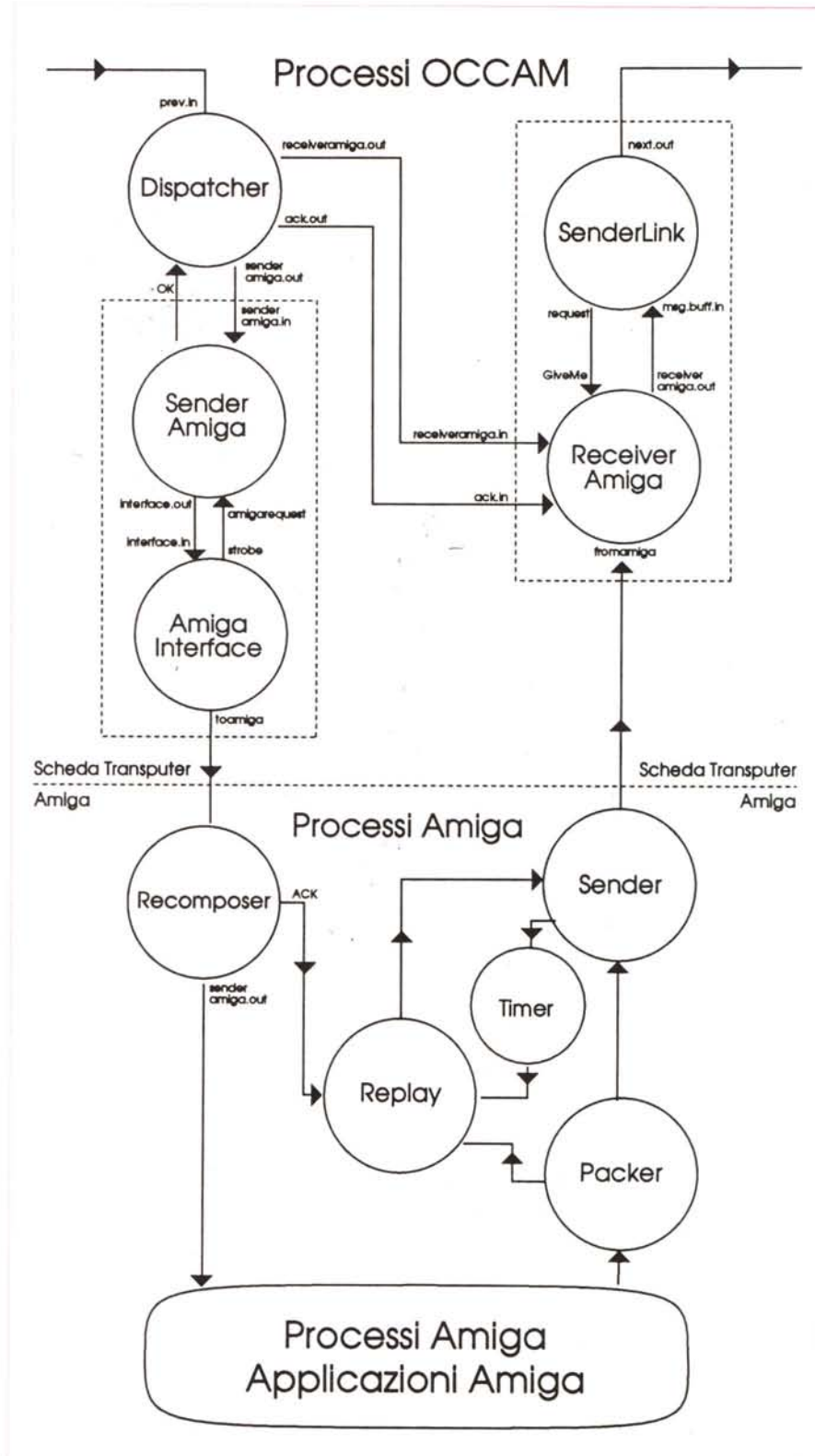

Figura 1 - Tutti i processi di ADPnetwork.

se lo legga...) nel secondo non fa nulla, uccidendo così di fatto il pacchetto «sfortunato».

La seconda guardia d'ingresso del processo SenderAmiga effettua l'interfacciamento col processo «minore» AmigaInterface, inviando ad esso un pacchetto bufferizzato ad ogni sua richiesta sul canale «strobe». Sempreché, naturalmente, il buffer non sia vuoto (condizione «totale > 0»).

I processi «minori»

AmigaInterface e SenderLink sono due processi di interfaccia che non fanno altro che richiedere al corrispondente processo buffer (SenderAmiga per il primo e ReceiverAmiga per il secondo) un elemento bufferizzato ed effettuare rispettivamente la spedizione di quest'ultimo verso l'Amiga o verso il nodo successivo.

La loro «esistenza» è dovuta al fatto che un processo buffer, per come è strutturato l'OCCAM (e in generale la comunicazione inter process ad ambiente locale), è di solito realizzato attraverso un comando alternativo (ripetitivo), con o senza priorità, sul quale è possibile inserire solo guardie d'ingresso. In generale, un processo buffer, attende da due (o più) canali d'ingresso comandi che possono essere di inserimento o di estrazione elemento nel/dal buffer. La coppia di processi «buffer-interfaccia» può semmai essere vista dal punto di vista logico come un'unica entità buffer (box tratteggiati sempre in figura 1) in cui quello che entra dai canali d'ingresso del processo buffer automaticamente esce dal canale d'uscita del processo di interfaccia appena c'è qualcosa nel buffer e il canale d'uscita è libero (è terminata, cioè, la spedizione precedente). Inutile commentarvi le tre (identiche) linee dei due processi che si autospiegano al primo colpo d'occhio.

Il processo MASTER

Tutti i processi finora descritti assomigliano, come listato, a delle comuni subroutine con tanto di passaggio di parametri. Chi è, dunque, che li fa partire come processi ? In fondo al listato pubblicato sul numero scorso c'è un ultimo processo: Netputer. La sua funzione è quella sì di lanciare come tali i processi, ma soprattutto di collegare i loro canali. Far sì, ad esempio (figura 1), che il canale strobe di AmigaInterface sia lo stesso canale amigarequest di SenderAmiga sul quale il primo invia messaggi, il secondo riceve. Ciò si realizza passando ai due come parametro lo stesso canale, dichiarato dal processo Netputer all'inizio. Lì il canale si chiama ancora strobe, ma nulla vietava di dichiararlo con nome Pippo e sempre con tale nome passarlo ai due processi. Che poi essi al loro interno lo chiamino rispettivamente strobe ed amigarequest è solo una questione di nomi, che in quanto tali a tempo di esecuzione spariscono. Né più né meno di quanto succede coi parametri attuali e formali delle normali subroutine degli altrettanto normali (nel senso di classici) linguaggi di programmazione.

Come sempre visibile nel listato pubblicato lo scorso mese, Netputer ha bisogno a sua volta di alcuni parametri esterni (come i link fisici e l'indirizzo di rete) che gli saranno passati all'interno del file di configurazione come vedremo ora.

L'eseguibile

In figura 2 è mostrato il file di configurazione per ottenere l'eseguibile per il transputer. O, meglio, per ottenere un vero e proprio file di boot che inoltrato attraverso un link su un transputer appena resettato (e con il piedino di BootFromRom settato opportunamente, diversamente il boot sarà effettuato da più classiche rom) permette a questo di partire con il codice da eseguire. Lo stesso file di configurazione permette anche di mappare processi su reti di transputer fornendo comunque sempre e solo un unico eseguibile da mandare al primo transputer della rete che provvederà a prendersi la parte di codice di sua competenza e a rigirare ai rimanenti nodi il codice per gli altri transputer.

Commentiamolo brevemente. A parte i due include iniziali necessari per le costanti di I/O e per gli indirizzi dei link fisici, il tutto si svolge dichiarando alcuni canali, piazzando questi sui link fisici dei processori disponibili, e chiamando i processi o il processo da eseguire passando gli eventuali parametri. Nel caso mostrato in figura 1 abbiamo un solo transputer (T2, basta il nome della famiglia), quattro canali (uno verso il nodo successivo, uno verso il nodo precedente, uno bidirezionale da/verso l'Amiga) e solo codice («netputt2.c2h») generato dal linker dopo la compilazione. Il processo Netputer è naturalmente contenuto in questo codice così come sono in esso contenuti tutti gli altri processi lanciati in parallelo da questo.

```
-- ***********************************************
-- *                                             *
-- *                ------------                 *
-- *                NETPUTER.PGM                 *
-- *                ------------                 *
-- *                                             *
-- *         (C) 1989,90 -  ADPsoftware          *
-- *                                             *
-- *    Questo programma permette di ottenere    *
-- *                                             *
-- *    il file eseguibile  per il transputer    *
-- *                                             *
-- ***********************************************


#INCLUDE "hostio.inc"      -- host i/o constants
#INCLUDE "linkaddr.inc"    -- link address constants


-- External channels to be mapped to links

CHAN OF BYTE next,prev:
CHAN OF INT::[]BYTE ToAmiga,FromAmiga:

PLACED PAR
  PROCESSOR 1 T2
    #USE "netputt2.c2h"
    PLACE prev AT link1.in:
    PLACE next AT link2.out:
    PLACE ToAmiga AT link0.out:
    PLACE FromAmiga AT link0.in:
    Netputer(next,prev,ToAmiga,FromAmiga,12345)
:
```

Figura 2.

Per concludere

In figura 3 è mostrato l'include «ADPnet.inc» utilizzato dal software di rete. Lì dentro troverete tutte le costanti usate, la tabella delle costanti per il calcolo del CRC a 32 bit e due array per il controllo veloce della lunghezza e del tipo del pacchetto ricevuto da parte del processo Dispatcher che, tra le tante cose, deve anche riconoscere pacchetti validi da eventuale «monnezza» erroneamente in circolazione sulla rete (a causa ad esempio di «morte violenta» di un nodo).

Possiamo, anche per questo mese, mettere il nostro «punto e basta» con la soddisfazione però di non aver lasciato discorsi a metà: non crediate che ci divertiamo a spezzare gli articoli! È solo che stringendo troppo si rischia di non riuscire a farsi comprendere, pubblicando articoli di 10-12 pagine si ha quasi la certezza di non essere affatto letti. Ci spiace solo per chi ha perso il numero precedente o perderà questo. Pazienza! MC

Luciano Macera è raggiungibile tramite MC-link alla casella MC2833.

```
-- /************************* 06/08/90 **
-- *                                    *
-- *                                    *
-- *      A D P n e t w o r k   3.1     *
-- *          (#include file)           *
-- *                                    *
-- *       --------------------         *
-- *       (c) 89,90 ADPsoftware        *
-- *       --------------------         *
-- *                                    *
-- ************************************/

VAL   HEADER         IS     28 :
VAL   DEFAULT.LEN    IS   1024 :
VAL   FRAME.BUFFER   IS   1200 :
VAL   MAX.LEN        IS   1024 :
VAL   MIN.LEN        IS     32 :
VAL   ACK.LEN        IS     32 :

VAL   BUFFER.LEN     IS      32: -- Deve essere potenza di 2 !!!
VAL   AMBUFFER.LEN   IS      16: -- Deve essere potenza di 2 !!!
VAL   ACKBUFFER.LEN  IS      64: -- Deve essere potenza di 2 !!!

VAL   ACK.FRAME      IS    'A' :
VAL   CON.FRAME      IS    'C' :
VAL   HAN.FRAME      IS    'H' :
VAL   SER.FRAME      IS    'S' :
VAL   NEX.FRAME      IS    'N' :
VAL   POR.FRAME      IS    'P' :
VAL   POL.FRAME      IS    'O' :
VAL   CHA.FRAME      IS    'T' :
VAL   CCH.FRAME      IS    't' :
VAL   ICS.FRAME      IS    'X' :
VAL   BOO.FRAME      IS    'B' :
VAL   MAI.FRAME      IS    'M' :

VAL   Dest.pos       IS      0 :
VAL   Mitt.pos       IS      2 :
VAL   EffLen.pos     IS      4 :
VAL   MsgID.pos      IS      8 :
VAL   Offset.pos     IS     12 :
VAL   Stamp1.pos     IS     16 :
VAL   Stamp2.pos     IS     18 :
VAL   Len.pos        IS     20 :
VAL   Type.pos       IS     21 :
VAL   Chop.pos       IS     22 :
VAL   CkSum.pos      IS     23 :
VAL   CRC.pos        IS     24 :
VAL   Body.pos       IS     28 :

VAL crc.32.tab IS [ #00000000 (INT32), #77073096 (INT32), #EE0E612C (INT32), #990951BA (INT32),
                    #076DC419 (INT32), #706AF48F (INT32), #E963A535 (INT32), #9E6495A3 (INT32),
                    #0EDB8832 (INT32), #79DCB8A4 (INT32), #E0D5E91E (INT32), #97D2D988 (INT32),
                    #09B64C2B (INT32), #7EB17CBD (INT32), #E7B82D07 (INT32), #90BF1D91 (INT32),
                    #1DB71064 (INT32), #6AB020F2 (INT32), #F3B97148 (INT32), #84BE41DE (INT32),
                    #1ADAD47D (INT32), #6DDDE4EB (INT32), #F4D4B551 (INT32), #83D385C7 (INT32),
                    #136C9856 (INT32), #646BA8C0 (INT32), #FD62F97A (INT32), #8A65C9EC (INT32),
                    #14015C4F (INT32), #63066CD9 (INT32), #FA0F3D63 (INT32), #8D080DF5 (INT32),
                    #3B6E20C8 (INT32), #4C69105E (INT32), #D56041E4 (INT32), #A2677172 (INT32),
                    #3C03E4D1 (INT32), #4B04D447 (INT32), #D20D85FD (INT32), #A50AB56B (INT32),
                    #35B5A8FA (INT32), #42B2986C (INT32), #DBBBC9D6 (INT32), #ACBCF940 (INT32),
                    #32D86CE3 (INT32), #45DF5C75 (INT32), #DCD60DCF (INT32), #ABD13D59 (INT32),
                    #26D930AC (INT32), #51DE003A (INT32), #C8D75180 (INT32), #BFD06116 (INT32),
                    #21B4F4B5 (INT32), #56B3C423 (INT32), #CFBA9599 (INT32), #B8BDA50F (INT32),
                    #2802B89E (INT32), #5F058808 (INT32), #C60CD9B2 (INT32), #B10BE924 (INT32),
                    #2F6F7C87 (INT32), #58684C11 (INT32), #C1611DAB (INT32), #B6662D3D (INT32),
                    #76DC4190 (INT32), #01DB7106 (INT32), #98D220BC (INT32), #EFD5102A (INT32),
                    #71B18589 (INT32), #06B6B51F (INT32), #9FBFE4A5 (INT32), #E8B8D433 (INT32),
                    #7807C9A2 (INT32), #0F00F934 (INT32), #9609A88E (INT32), #E10E9818 (INT32),
                    #7F6A0DBB (INT32), #086D3D2D (INT32), #91646C97 (INT32), #E6635C01 (INT32),
                    #6B6B51F4 (INT32), #1C6C6162 (INT32), #856530D8 (INT32), #F262004E (INT32),
                    #6C0695ED (INT32), #1B01A57B (INT32), #8208F4C1 (INT32), #F50FC457 (INT32),
                    #65B0D9C6 (INT32), #12B7E950 (INT32), #8BBEB8EA (INT32), #FCB9887C (INT32),
                    #62DD1DDF (INT32), #15DA2D49 (INT32), #8CD37CF3 (INT32), #FBD44C65 (INT32),
                    #4DB26158 (INT32), #3AB551CE (INT32), #A3BC0074 (INT32), #D4BB30E2 (INT32),
                    #4ADFA541 (INT32), #3DD895D7 (INT32), #A4D1C46D (INT32), #D3D6F4FB (INT32),
                    #4369E96A (INT32), #346ED9FC (INT32), #AD678846 (INT32), #DA60B8D0 (INT32),
                    #44042D73 (INT32), #33031DE5 (INT32), #AA0A4C5F (INT32), #DD0D7CC9 (INT32),
                    #5005713C (INT32), #270241AA (INT32), #BE0B1010 (INT32), #C90C2086 (INT32),
                    #5768B525 (INT32), #206F85B3 (INT32), #B966D409 (INT32), #CE61E49F (INT32),
                    #5EDEF90E (INT32), #29D9C998 (INT32), #B0D09822 (INT32), #C7D7A8B4 (INT32),
                    #59B33D17 (INT32), #2EB40D81 (INT32), #B7BD5C3B (INT32), #C0BA6CAD (INT32),
                    #EDB88320 (INT32), #9ABFB3B6 (INT32), #03B6E20C (INT32), #74B1D29A (INT32),
                    #EAD54739 (INT32), #9DD277AF (INT32), #04DB2615 (INT32), #73DC1683 (INT32),
                    #E3630B12 (INT32), #94643B84 (INT32), #0D6D6A3E (INT32), #7A6A5AA8 (INT32),
                    #E40ECF0B (INT32), #9309FF9D (INT32), #0A00AE27 (INT32), #7D079EB1 (INT32),
                    #F00F9344 (INT32), #8708A3D2 (INT32), #1E01F268 (INT32), #6906C2FE (INT32),
                    #F762575D (INT32), #806567CB (INT32), #196C3671 (INT32), #6E6B06E7 (INT32),
                    #FED41B76 (INT32), #89D32BE0 (INT32), #10DA7A5A (INT32), #67DD4ACC (INT32),
                    #F9B9DF6F (INT32), #8EBEEFF9 (INT32), #17B7BE43 (INT32), #60B08ED5 (INT32),
                    #D6D6A3E8 (INT32), #A1D1937E (INT32), #38D8C2C4 (INT32), #4FDFF252 (INT32),
                    #D1BB67F1 (INT32), #A6BC5767 (INT32), #3FB506DD (INT32), #48B2364B (INT32),
                    #D80D2BDA (INT32), #AF0A1B4C (INT32), #36034AF6 (INT32), #41047A60 (INT32),
                    #DF60EFC3 (INT32), #A867DF55 (INT32), #316E8EEF (INT32), #4669BE79 (INT32),
                    #CB61B38C (INT32), #BC66831A (INT32), #256FD2A0 (INT32), #5268E236 (INT32),
                    #CC0C7795 (INT32), #BB0B4703 (INT32), #220216B9 (INT32), #5505262F (INT32),
                    #C5BA3BBE (INT32), #B2BD0B28 (INT32), #2BB45A92 (INT32), #5CB36A04 (INT32),
                    #C2D7FFA7 (INT32), #B5D0CF31 (INT32), #2CD99E8B (INT32), #5BDEAE1D (INT32),
                    #9B64C2B0 (INT32), #EC63F226 (INT32), #756AA39C (INT32), #026D930A (INT32),
                    #9C0906A9 (INT32), #EB0E363F (INT32), #72076785 (INT32), #05005713 (INT32),
                    #95BF4A82 (INT32), #E2B87A14 (INT32), #7BB12BAE (INT32), #0CB61B38 (INT32),
                    #92D28E9B (INT32), #E5D5BE0D (INT32), #7CDCEFB7 (INT32), #0BDBDF21 (INT32),
                    #86D3D2D4 (INT32), #F1D4E242 (INT32), #68DDB3F8 (INT32), #1FDA836E (INT32),
                    #81BE16CD (INT32), #F6B9265B (INT32), #6FB077E1 (INT32), #18B74777 (INT32),
                    #88085AE6 (INT32), #FF0F6A70 (INT32), #66063BCA (INT32), #11010B5C (INT32),
                    #8F659EFF (INT32), #F862AE69 (INT32), #616BFFD3 (INT32), #166CCF45 (INT32),
                    #A00AE278 (INT32), #D70DD2EE (INT32), #4E048354 (INT32), #3903B3C2 (INT32),
                    #A7672661 (INT32), #D06016F7 (INT32), #4969474D (INT32), #3E6E77DB (INT32),
                    #AED16A4A (INT32), #D9D65ADC (INT32), #40DF0B66 (INT32), #37D83BF0 (INT32),
                    #A9BCAE53 (INT32), #DEBB9EC5 (INT32), #47B2CF7F (INT32), #30B5FFE9 (INT32),
                    #BDBDF21C (INT32), #CABAC28A (INT32), #53B39330 (INT32), #24B4A3A6 (INT32),
                    #BAD03605 (INT32), #CDD70693 (INT32), #54DE5729 (INT32), #23D967BF (INT32),
                    #B3667A2E (INT32), #C4614AB8 (INT32), #5D681B02 (INT32), #2A6F2B94 (INT32),
                    #B40BBE37 (INT32), #C30C8EA1 (INT32), #5A05DF1B (INT32), #2D02EF8D (INT32)]:

VAL range IS [ 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ] :

VAL alpha IS [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
               0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ] :
```

Figura 3 - Listato dell'Include utilizzato dal software di rete pubblicato lo scorso mese.

Ecco tutto l'hardware che serve per mettere in rete 2 Pc

ORA COMPATIBILE MS-DOS 5 E WINDOWS 3.0

Naturalmente il cavo da solo non basta: ci vuole anche U_NET99, il software che permette di collegare in rete due o tre PC MS-DOS utilizzando le porte seriali standard. Non ci sono nuovi comandi da apprendere, nuovi manuali tecnici da digerire, nuove (e strane) maniere di fare le solite cose, nuovo hardware o software applicativo da comprare. Ogni comando DOS e presumibilmente ogni programma che abbiate mai usato funzionerà esattamente come prima di installare U_NET99.

"Nient'altro sul mercato offre prestazioni confrontabili ad un prezzo così modesto". Lo ha scritto il prestigioso *PC Magazine*. In effetti, a 149.000 lire (compreso il cavo e l'IVA), il costo di U_NET99 si giustifica già solo per condividere una stampante tra due computer. Ma giudicate voi le caratteristiche tecniche:

- Supporta due o tre PC/XT/AT/386 o compatibili via porta seriale RS232, che tutti i PC già posseggono in standard;
- Massima velocità di trasferimento pari a 115.200 bit/secondo, qualcosa più di 14.000 byte/secondo;
- Ognuno dei computer può accedere alle risorse hardware e software degli altri tramite i normali comandi del DOS, quali COPY e DIR, come se si trattasse di risorse locali;
- Stampa su stampanti locali e remote;
- Usa solamente 14K di RAM ed è totalmente trasparente per l'utente e per il software applicativo.

Il concetto è veramente molto semplice: se ad esempio prima avevate 3 *drive* e una stampante su un PC e solo 2 *drive* sull'altro, con U_NET99 **entrambi** "vedranno" 5 *drive* e la stampante. Aggiungete a tutto ciò l'ottimo manuale in Italiano e un servizio di *hot line* telefonica a vostra disposizione.

La nuova versione di U_NET99, oltre ad avere qualche *utility* in più rispetto alla precedente, rende la rete perfettamente **compatibile con *Windows 3.0* e riconosce l'MS-DOS 5**. Il prezzo **rimane invariato** (L. 149.000); l'aggiornamento dalla versione 2.3j alla 2.3n costa 38.000 lire (comprende *U_NET99 Companion*!) e deve essere richiesto all'editore (Ultimobyte, tel. **02/65.97.693**).

E da oggi, fino ad esaurimento scorte, *Sidefiler* è in omaggio: un pratico contenitore per dischetti da attaccare al monitor. Affrettatevi, per non perdere questa opportunità **compilate oggi stesso il tagliando** e rispeditelo al nostro indirizzo oppure telefonateci al numero **02/65.55.306**

IN OMAGGIO *SIDEFILER*, IL PRATICO CONTENITORE PER DISCHETTI

MICROSTAR **Via Aldo Manuzio, 15 - 20124 MILANO**

SI' inviatemi con urgenza la rete locale U_NET99 al prezzo di L. 149.000, comprensivo di software, manuale in Italiano e cavo di 10 metri. **Resta inteso che riceverò in omaggio *Sidefiler*.** Formato dischetti: ☐ 3" ☐ 5"

NOME .....

VIA ..... CAP .....

CITTÀ ..... ( ) TEL. .....

P.IVA/COD. FISC. .....
(solo se si desidera fattura)

PAGAMENTO
- ☐ assegno NON TRASFERIBILE allegato
- ☐ vaglia postale (ricevuta o fotocopia allegata)
- ☐ contrassegno postale (aggiungere L. 6.000 per contributo spese)
- ☐ contrassegno corriere (spedizione in porto assegnato)