Fault Tolerant Transputing

di Andrea de Prisco

Il 25 novembre u.s. si è svolta presso l'INMOS Business Centre di Assago (MI) una demo di un sistema fault tolerant interamente basata su una rete di transputer. L'applicazione riguardava il calcolo della FFT, effettuato in pipeline su due nodi, di un segnale digitale fornito da un AD converter. Questo era a sua volta alimentato da un comunissimo lettore di musicassette tipo walkman. Staccando un link o addirittura spegnendo completamente uno dei transputer in funzione, non solo il calcolo continuava senza interruzioni di sorta, ma ripristinando il collegamento o dando nuovamente alimentazione al transputer disattivato l'intero sistema riprendeva automaticamente a lavorare a pieno regime ossia riconfigurandosi nuovamente come sistema fault tolerant.

Ottimo spunto per proporvi questo articolo didattico sull'argomento più che mai interessante. Almeno speriamo...

Fault tolerance

La tolleranza ai guasti, si sa, non è per nulla un concetto assoluto ma può comodissimamente essere interpretato in vari modi. Partiamo cioè dall'assunto che la tolleranza ai guasti ASSOLUTA è una cosa OVVIAMENTE impossibile. Realizzare un sistema totalmente indistruttibile è impensabile oltre che, naturalmente, irrealizzabile. Diciamo quindi che la tolleranza ai guasti possiamo vederla come una specie di soglia più o meno alta che ci mette al riparo da un numero più o meno grande di inconvenienti. Una volta realizzato un sistema che «tollera» determinati tipi di guasti, non possiamo pretendere che continui a funzionare anche in seguito a guasti diversi da quelli previsti o in numero maggiore.

Un altro punto abbastanza poco «definito» è il fatto se la tolleranza implichi o meno il ripristino. Personalmente credo che un sistema in grado di continuare a funzionare anche in seguito ad un guasto sia fault tolerant ANCHE se per ripristinare il sistema sia necessario l'intervento di un tecnico che per effettuare la riparazione deve sospendere momentaneamente l'intero sistema. C'è ovviamente chi la pensa diversamente dal sottoscritto e reputa fault tolerant solo i sistemi che continuino a lavorare anche MENTRE un tecnico sostituisce il pezzo difettoso. È un modo diverso di intendere la cosa: un po' come affermava Enzo Ferrari, dicendo che un motore affidabile è un motore che ti permette di completare la gara anche se due metri dopo il traguardo scoppia in mille pezzi. Visto quindi dal punto di vista mio, un sistema in grado di essere riparato senza sospendere l'elaborazione è ancor di più di un sistema fault tolerant... liscio. E giustifico questa mia presa di posizione nel significato stesso delle parole «tolleranza ai guasti» che indentificano qualcosa che in quanto «tollerante» continua a funzionare anche in seguito ad un guasto. Punto e basta.

Detto questo

Chiaramente per ottenere una continuità di funzionamento anche a seguito della rottura di un modulo è necessario introdurre una certa ridondanza nel sistema. Un singolo processore, se si sfascia, non può far nulla per continuare il suo compito, più processori invece possono assorbire abbastanza facilmente la rottura di uno o più di essi.

Altra cosa da mettere in chiaro è che

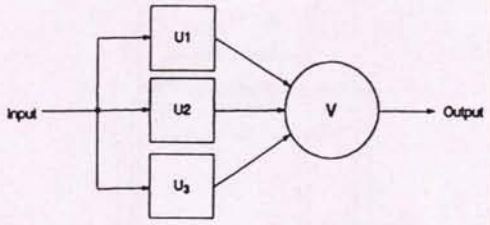


Figura 1

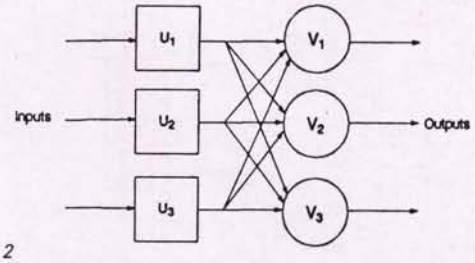
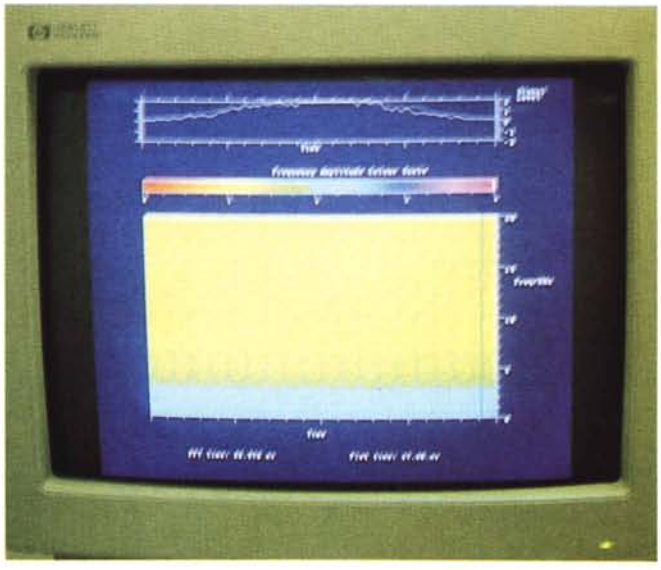


Figura 2

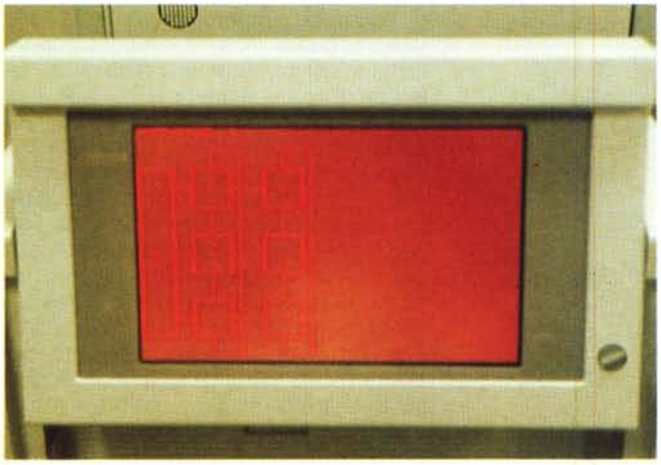


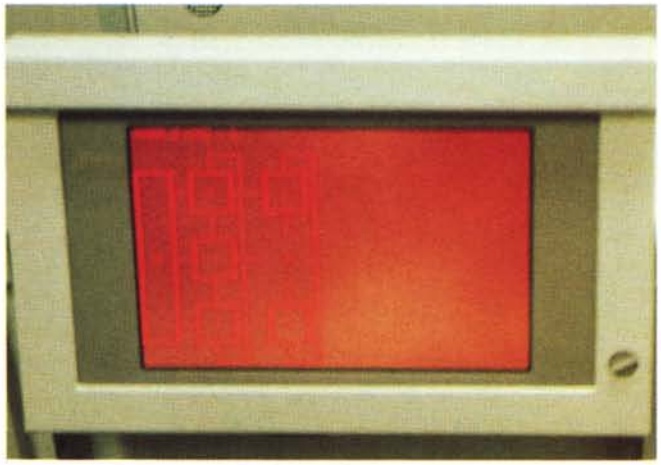
Output grafico del programma di FFT.

di solito un sistema fault tolerant oltre ad essere, banalmente, molto più costoso di un sistema classico, molto probabilmente avrà performance inferiori a causa dell'overhead indotto dalla tolleranza stessa: ci saranno molti più controlli da fare poiché ogni processore in generale deve, oltre che eseguire il suo lavoro, essere sempre al corrente di quello che sta succedendo nell'intero sistema o in parte di esso.

Una metodologia abbastanza semplice per implementare sistemi fault tolerant si basa sullo schema mostrato in figura 1. L'input da processare viene passato non ad un singolo processore ma contemporaneamente a tre unità indipendenti. I risultati da loro processati saranno inviati ad un processo «Voter» (presumibilmente in esecuzione su un quarto processore) che li confronta. Se i risultati sono tutt'e tre identici il sistema sta funzionando perfettamente e quindi l'output può essere rilasciato. Se uno solo dei tre risultati differisce dagli altri due vuol dire che un processore è rotto e l'output giusto (per questioni di «maggioranza») è quello fornito dai due processori concordanti. Se i risultati sono tutt'e tre differenti il Voter non potrà scegliere quello giusto (che magari è uno dei tre ma non ha nessun metodo per riconoscerlo) e il sistema fallisce. Nel discorso appena fatto abbiamo intrinsecamente posizionato la soglia di fault tolerance alla quale prima ci riferivamo: il sistema di figura 1 «tollera» la rottura di un solo processore di calcolo (U1, U2, U3) ma non quella di due. Aumentando il numero di unità (sempre in mumero dispari, possibilmente) alziamo la soglia di tolleranza. In ogni caso il sistema funziona correttamente se funzionano la metà più uno dei processori disponibili inizialmente.

Altrettanto ovviamente cessa di funzionare se «zompa» il processore del Voter indipendentemente dal funzionamento dei processori di calcolo. Per ovviare a questa sorta di collo di bottiglia (preferirei chiamarlo anello debole della catena) in figura 2 è mostrato lo schema comprendente la ridondanza anche per i processori Voter. Ognuno di questi riceve i risultati da tutti i processori di calcolo e sceglie col medesimo meccanismo visto prima l'output da espellere. Naturalmente occorre prendere ulteriori precauzioni onde evitare che i Voter diano a loro volta risultati





Su un display era mostrato sempre lo stato della rete di transputer funzionanti.

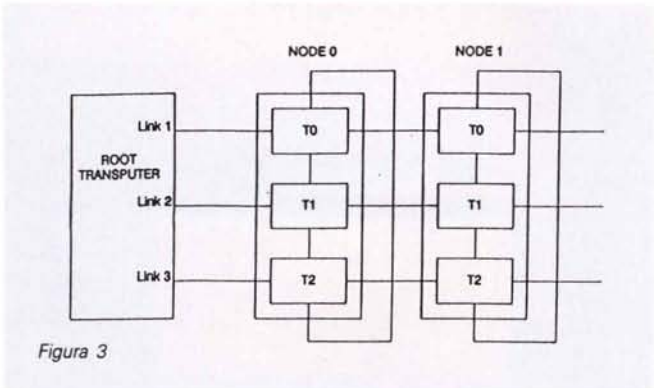


Figura 3

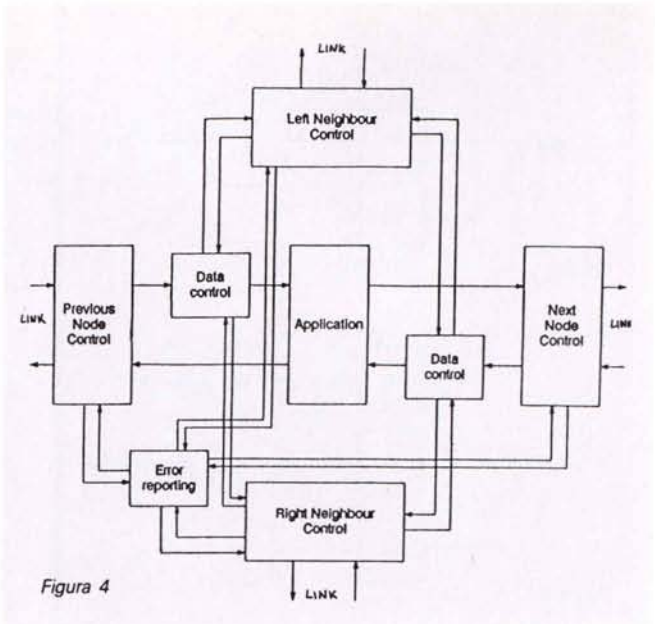
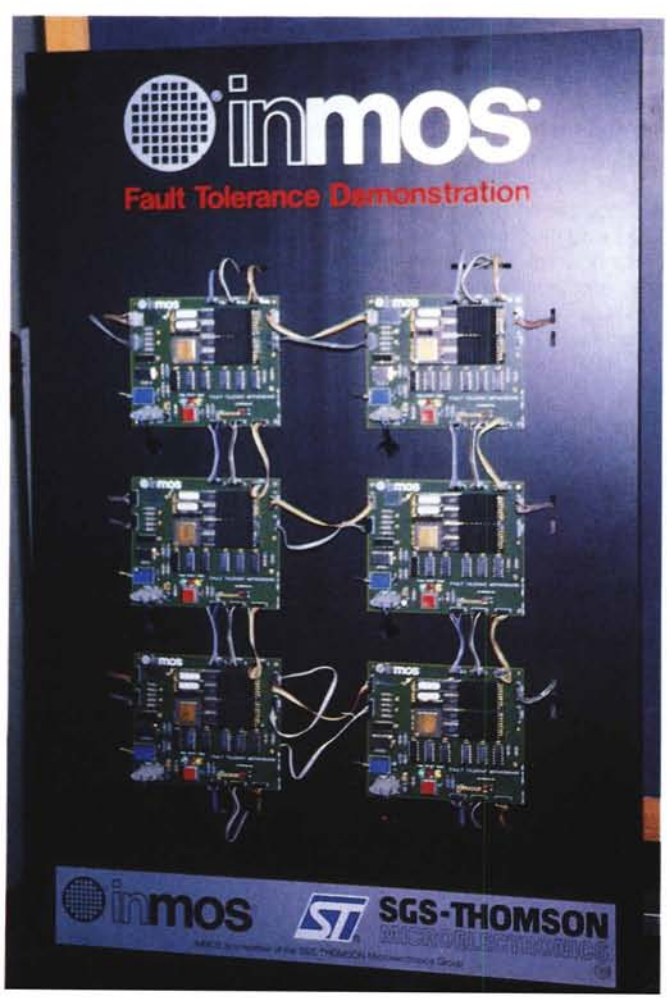


Figura 4

incorretti su dati corretti, ad esempio lasciandoli dialogare tra loro (autocontrollandosi vicendevolmente) in modo da mettere a tacere il Voter mal funzionante.

Il sistema della INMOS

In figura 3 è mostrata l'implementazione INMOS della loro demo fault tolerant che, vista dal punto di vista mio, è ancor di più di un sistema tollerante ai guasti. Come detto in apertura tale sistema INMOS (che ribadiamo non essere un prodotto commerciale ma solo una demo) è in grado di autoripristinarsi non appena viene eliminato il guasto, operazione effettuabile senza interrompere il normale funzionamento. Tra l'altro, non essendo prevista alcuna ROM sui moduli, un nuovo transputer installato in seguito ad una riparazione on line è automaticamente resettato dagli altri transputer dello stesso nodo ed esegue il suo bootstrap ricevendo via link dal suo fratello «destro» (o «sinistro», non ricordo) l'immagine della sua memoria in quel momento. Quindi un nuovo transputer installato riprende a funzionare esattamente dallo stesso stato in cui si trovano in quel momento gli altri chip di quel nodo, che nulla ha a che vedere con lo stato che aveva quando è accaduto il fault.

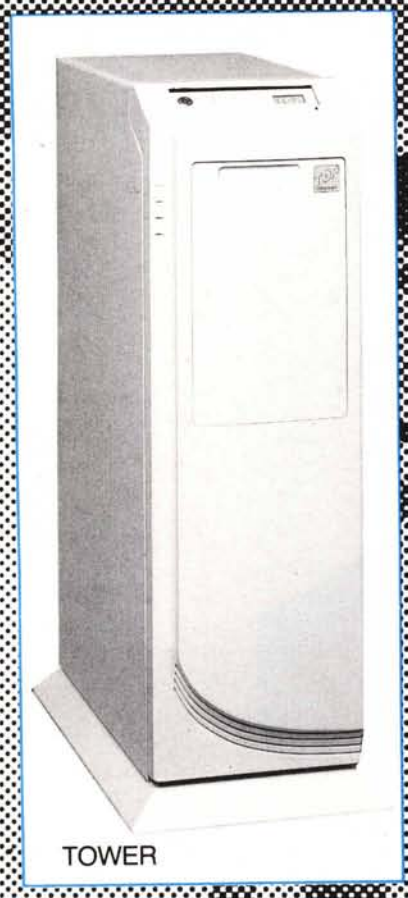


La Demo Fault Tolerant della Inmos.

Per finire, in figura 4 sono mostrati i processi in esecuzione parallela su ogni transputer presente in ogni nodo. Notare al centro l'applicazione (che in quella demo eseguiva il calcolo della FFT ma poteva trattarsi di qualsiasi altra cosa) interamente circondata da altri processi di controllo per implementare la tolleranza ai guasti. Non fatevi però impressionare dal numero di processi in più (su un analogo sistema NON fault tolerant è sufficiente il solo processo «Application») in quanto ciò che conta è la richiesta di CPU da parte di ognuno di questi, naturalmente molto maggiore per l'applicazione vera e propria. Fatti due conti, ma soprattutto cronometro alla mano, il sistema fault tolerant impiega solo il 40% in più dell'analogo sistema pipeline ad un solo transputer per nodo, ma funziona anche se giochiamo con questo al tiro a segno: basta usare un fucile a proiettile (e non a pallini), mirare bene, e colpire un solo trasputer di ogni nodo per volta. Provare per credere... MC

per presentarsi bene...

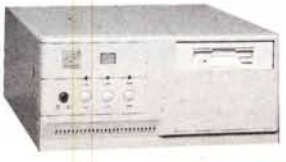
TOP computer



MINITOWER

DESK

BABY



	286/16	386/25SX	386/33	386/40	486/25SX	486/33
CPU	80286	80386SX	80386DX	80386DX	80486SX	80486DX
Clock	16	25SX	33	40	25SX	33
Cache Memory	—	—	64K	32/256K	—	32/256K
Memoria da:	1Mb	1Mb	1Mb	1Mb	1Mb	1Mb
a:	4Mb	16Mb	8Mb	32Mb	32Mb	32Mb
Drive Epson	1,2Mb 1,44Mb	1,2Mb 1,44Mb	1,2Mb 1,44Mb	1,2Mb 1,44Mb	1,2Mb 1,44Mb	1,2Mb 1,44Mb
VGA	800 x 600 1024 x 768	800 x 600 1024 x 768	800 x 600 1024 x 768	800 x 600 1024 x 768	800 x 600 1024 x 768	800 x 600 1024 x 768
Hard Disk da:	40Mb	40Mb	40Mb	40Mb	40Mb	40Mb
a:	520Mb	520Mb	520Mb	520Mb	520Mb	520Mb

Tutti i sistemi sono disponibili nelle linee Baby, Desk, Minitower e Tower.

ESSEGI informatica 00142 ROMA

VIA ALBERTO ASCARI, 172 Tel. 06·5193221-2 FAX 06·5043056

CONCESSIONARI AUTORIZZATI:

EUROSOFTWARE sas - Via Renato Fucini 40, ROMA - Tel. 06/888.39.94
FABER INFORMATICA sas - Via Olivella 1, FORMIA (LT) - Tel. 0771/771304
FUNCTION snc - Via del Casaletto 161, ROMA - Tel. 06/588.74.13
GIPA srl - Via A. La Marmora 27, FIRENZE - Tel. 055/500.06.10
MICROSIS snc - Via Paolo V 33, ROMA - Tel. 06/629.19.90
R.O. srl - Via di Donna Olimpia 140, ROMA - Tel. 06/531.56.85