di Sergio Polini

Controllo del coprocessore

Anche questa volta approfitto del corsivo iniziale per scambiare due chiacchiere con alcuni di voi. A Giacomo Spica Dinatale, di Militello in Val di Catania, sono costretto a dire che, nonostante la cura con cui ha controllato i suoi listati della unit TSR (che mi ha inviato su dischetto), vi sono alcuni errori: in SetPSP, il primo confronto tra VersioneDOS e $300 è seguito da una istruzione Dec invece che Inc; in SetDTA, si assegna $1A a Reg.AX invece che a Reg.AH (i miei sorgenti erano stati pubblicati e commentati nel numero di maggio). Per Umberto Tarozzi, di Forlì, ecco l'indirizzo della TurboPower Software: P.O.Box 66747, Scotts Valley, California 95067-0747

La volta scorsa ho illustrato un bug nella funzione Sqrt del Turbo Pascal: non tanto qualcosa che ne compromettesse il normale funzionamento, quanto piuttosto la mancanza di un MOV SP,BP che consentisse l'intercettazione e la gestione della eccezione «argomento negativo». Come vi avevo promesso, ho sottoposto il problema alla Borland. Ciò è avvenuto in occasione dell'appuntamento italiano del loro World OOP Tour (di cui vi abbiamo dato notizia nel numero scoso); ne ho parlato con David Intersimone, ricevendone l'assicurazione di un pronto controllo. Tuttavia, non ho dovuto neppure aspettare la sua risposta: di lì a poco, infatti, ho potuto mettere le mani sul nuovo Turbo Pascal 6.0 (di cui vi anticipo le principali caratteristiche in altra parte della rivista), e ho potuto così verificare che il problema era già stato individuato e risolto, appunto aggiungendo quel MOV SP,BP che mancava. Con la nuova versione, quindi, è possibile gestire anche l'errore 207, che può pertanto essere tolto dal set delle situazioni «non gestibili», sia nella procedura *Raise* della unit GESTECC, sia nel file GESTECC.ASM (sostituendo CMP AX,207 con CMP AX,208 alla riga 28).

Se si usa o si emula il coprocessore numerico con la direttiva $N, tuttavia, l'errore 207 è gestibile anche con il Turbo Pascal 5.5. Almeno a mio parere. Ma non tutti sono d'accordo. La TurboPower Software propone una ricca libreria di funzioni denominata «Object Professional», in cui si possono trovare molte delle potenti routine già presenti in un loro precedente prodotto («Turbo Professional»), ampiamente rimaneggiate e riorganizzate in una vera e propria gerarchia di classi. Un ottimo esempio di cosa si può ottenere con la programmazione orientata all'oggetto. Vi sono anche routine per la gestione delle eccezioni, vagamente simili nella interfaccia a quelle che vi ho proposto il mese scorso, ma piuttosto diverse nella implementazione. Tra le altre cose, vengono intercettati gli interrupt generati dalle eccezioni del coprocessore allo scopo di escludere ogni possibilità di gestione, se non attraverso altre routine comprese in una distinta unit. Quest'ultima è peraltro espressamente dedicata alla scrittura di programmi residenti che usino o emulino il coprocessore, e da tale punto di vista offre certamente un supporto più valido di quanto potete trovare nella nostra unit TSR (soprattutto se il programma residente ricorre all'emulazione). Mi sembra tuttavia troppo drastica la scelta di escludere completamente, anche nei programmi «normali», la possibilità di gestire le eccezioni generate dal coprocessore.

Un po' di architettura

I coprocessori della famiglia 80x87 hanno cinque aree di dati: uno stack di otto registri, una parola di stato, una

B	C3		ST		C2	C1	C0	IR		PE	UE	OE	ZE	DE	IE

```
B           = busy
C3 C2 C1 C0 = codice di condizione
ST          = registro considerato "top" dello stack
IR          = richiesta di interrupt

Flag di eccezione:
PE          = precisione
UE          = underflow
OE          = overflow
ZE          = divisione per zero
DE          = operando denormalizzato
IE          = operazione non valida
```

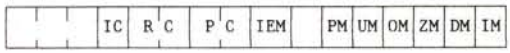
Figura 1. La parola di stato di un coprocessore 80x87.

parola di controllo, una parola di *tag* (ho visto *tag* tradotto con «etichetta», «contrassegno» o simili, ma nessuna traduzione in italiano mi sembra soddisfacente) e quattro parole destinate a contenere, per ogni istruzione che viene eseguita, il suo codice e il suo indirizzo, oltre all'indirizzo degli eventuali operandi.

Ai nostri fini sono soprattutto interessanti le parole di stato e di controllo. La parola di stato, illustrata nella figura 1, contiene in ogni momento lo stato del coprocessore: il bit *busy* consente la sincronizzazione con il processore principale (8088 o 80x86), i tre bit ST indicano quale degli otto registri è in quel momento considerato «top» dello stack, i quattro bit di condizione vengono usati per le comparazioni, il bit IR viene settato ogni volta che si genera una eccezione, quale che sia (una sorta di «indicatore generico di errore»), i flag di eccezione consentono di determinare il tipo dell'eccezione. I tipi in totale sono sei: «precisione» indica che il risultato di un'operazione non è rappresentabile esattamente ed è stato quindi arrotondato, «underflow» e «overflow» indicano che il risultato è troppo piccolo o troppo grande, «divisione per zero» indica che si è tentato di dividere per zero un numero diverso da zero, «operando denormalizzato» scatta ogni volta che si tenta di operare su un numero denormale, «operazione non valida» denota situazioni quali la radice quadrata di un numero negativo o una divisione di zero per zero.

Per i non iniziati, cercherò di spiegare in due parole cosa vuol dire «numero denormale». La rappresentazione interna di un numero in virgola mobile prevede un bit di segno, un esponente e una parte significativa (o «mantissa»); quest'ultima è in genere un numero compreso tra 1 e 2, in modo da consentire la rappresentazione del maggior numero possibile di cifre significative. Potremmo fare un esempio con la notazione decimale, supponendo di avere a disposizione una mantissa di sole quattro cifre: il numero 0,1234 può essere espresso con «0123 * 10e−4», ma così si perde una cifra significativa; osservando la convenzione che la prima cifra della mantissa deve essere diversa da

Figura 2 — La parola di controllo di un coprocessore 80x87. Va notato che il bit IEM ha senso solo nell'8087, mentre in un 80387 l'infinito è sempre affine e gli operandi denormali sono automaticamente normalizzati.




```
IC  = controllo di infinito:
      0 : chiusura proiettiva
      1 : chiudura affine
RC  = controllo di arrotondamento:
      00: al piu' vicino o pari
      01: verso meno infinito
      10: verso piu' infinito
      11: verso zero
PC  = controllo di precisione:
      00: mantissa di 24 bit
      10: mantissa di 53 bit
      11: mantissa di 64 bit
IEM = abilitazione degli interrupt (solo 8087)
      0 : interrupt abilitati
      1 : interrupt disabilitati

Maschere delle singole eccezioni:
 PM          = precisione
 UM          = underflow
 OM          = overflow
 ZM          = divisione per zero
 DM          = operando denormalizzato
 IM          = operazione non valida
```

Figura 3 — Un breve sorgente in Assembler che implementa una procedura Control87 con la quale è possibile impostare come si desidera la parola di controllo di un coprocessore installato o emulato.

```
; CWORD87.ASM
; Consente di impostare una parola di controllo diversa da quella di default
; per il coprocessore installato o emulato.
; Da assemblare con il Turbo Assembler 1.0 o 2.0.
; Se il programma Pascal viene compilato con N+,E- (vero 80x87):
;       tasm cword87
; se il programma Pascal viene compilato con N+,E+ (80x87 emulato):
;       tasm /dEMUL87 cword87

        .MODEL TPASCAL

IFDEF EMUL87
        EMUL
ENDIF
        .DATA

EXTRN   Test8087 : BYTE

        .CODE

PUBLIC  Control87

Control87 PROC FAR
        ARG CWord : WORD
        cmp     Test8087, 3     ; se e' presente un 80387 ...
        jne     No80387
        or      CWord, 0002h    ; ... maschera l'eccezione "denormale"
No80387:
        fldcw   CWord
        ret
Control87 ENDP
        END
```

zero (che è appunto il formato *normale*), abbiamo «1234 * 10e−5», senza perdita di precisione. Abbiamo però dovuto «stiracchiare» l'esponente, cosa non sempre possibile. In alcuni casi (pensate alla differenza tra due numeri molto vicini tra loro), il risultato di un'operazione è talmente prossimo allo zero che, rispettando sia il formato normale che i limiti minimo e massimo dell'esponente, non potrebbe essere rappresentato che come zero, con totale perdita di significatività; ad evitare questo, l'80x87 adotta un formato *denormale*, in cui cioè le prime cifre della mantissa possono essere zero. Ciò tuttavia consente la normale prosecuzione solo dei calcoli che avvengono all'interno del coprocessore, per i quali viene adottata una rappresentazione che comprende tutti i bit della parte significativa di un numero; negli altri casi si assume invece sempre il formato normale, si assume cioè che il primo bit della mantissa sia «uno», e quindi quel primo bit non viene rappresentato. È questo il motivo per cui viene generata una eccezione ogni volta che si tenta di operare direttamente su un numero denormale: si tratta di un tentativo di intervenire su un numero che ha una rappresentazione diversa da quella assunta.

La parola di controllo (figura 2) consente di impostare alcune caratteristiche del funzionamento del coprocessore: il bit di infinito consente di scegliere tra chiusura proiettiva, in cui più e meno infinito coincidono, e chiusura affine, in cui i due estremi sono distinti; con i due bit per il controllo di arrotondamento si può scegliere il tipo di arrotondamento che si desidera, con quelli per il controllo di precisione si può impostare una precisione operativa interna minore rispetto a quella di default (64 bit) per garantire la compatibilità con processori precedenti. Quando si verifica un'eccezione, sono possibili due comportamenti: se l'eccezione non è «mascherata» (il corrispondente bit è zero), viene generato un interrupt (ciò avviene in modo diverso nell'8087 e nei coprocessori successivi, ma ora non è il caso di scendere troppo in dettaglio), altrimenti viene intrapresa un'azione correttiva automatica; nel caso di divisione per zero, ad esempio, il risultato viene posto uguale a più o meno infinito, secondo il segno degli operandi.

```
(*$N+,E+*)
program UFlow;
uses Gestecc;
(*$L cword87.obj*)

const
  ABIL_UF = $1320;
var
  x, y: double;
  Errore: word;

(*$F+*) procedure Control87(CWord: word); external; (*$F-*)

(*$F+*)
function Gestore: boolean;
begin
  Control87(ABIL_UF);
  Gestore := TRUE;
end;
(*$F-*)

begin
  InstallaGestoreEccezioni(Gestore);
  Control87(ABIL_UF);
  repeat
    Write('x, y: '); Readln(x, y);  (* provate con: 1e-300 1e300 *)
    x := x / y;
    Errore := ErroreRT;
    Writeln('Errore: ', Errore);
  until Errore = 206;
  DisinstallaGestoreEccezioni;
end.
```

Figura 4 — Un programma che illustra sia il controllo delle eccezioni generate dal coprocessore che l'uso della procedura Control87 (figura 3) per abilitare le eccezioni di underflow.

```
procedure ParserObj.Parse;
{ Parses an input stream }
var
  FirstToken : TokenRec;
  Accepted : Boolean;
begin
  ...
  ...
  ...
  if MathError or (ErroreRT <> 0) then  (* riga da modificare *)
  begin
    ParseError := True;
    ParseValue := 0;
    Exit;
  end;
  ParseError := False;
  ParseValue := Stack[StackTop].Value;
end; { ParserObj.Parse }
```

Figura 5 — La riga da modificare nella procedura ParserObj.Parse del file TCPARSER.PAS del TurboCalc per aggiungere a questo il completo controllo delle eccezioni numeriche.

Inizializzazione e reinizializzazione

Quando parte un programma che utilizzi o emuli il coprocessore, viene eseguita una routine che, determinato il tipo del coprocessore, provvede alla sua inizializzazione. Viene utilizzata a tale scopo l'istruzione FINIT, che assegna ai vari registri del coprocessore alcuni valori di default. Non tutti tali valori sono compatibili con l'operatività di un programma Turbo Pascal; vengono ad esempio mascherate tutte le eccezioni, e ciò impedirebbe la segnalazione degli errori di esecuzione mediante un interrupt. Si apportano quindi alcuni correttivi assegnando opportuni valori alla parola di controllo: 1330h nei casi di 8087, 80287 o emulazione software, 1332h nel caso si sia rilevata la presenza di un 80387. Con l'aiuto della figura 2 possiamo vedere cosa questo comporti: infinito con chiusura affine, arrotondamento al più vicino o pari, mantissa di 64 bit

```
program TCalc;
(*$S-*)
uses Gestecc, TCRun;

(*$IFDEF UNDERFLOW*)
(*$L CWORD87.OBJ*)

const
  ABIL_UF = $1320;

(*$F+*) procedure Control87(CWord: word); external; (*$F-*)
(*$ENDIF*)

(*$F+*)
function Gestore: boolean;
begin
(*$IFDEF UNDERFLOW*)
  Control87(ABIL_UF);
(*$ENDIF*)
  Gestore := TRUE;
end;
(*$F-*)

begin
  InstallaGestoreEccezioni(Gestore);
(*$IFDEF UNDERFLOW*)
  Control87(ABIL_UF);
(*$ENDIF*)
  Run;
  DisinstallaGestoreEccezioni;
end.
```

Figura 6 — Una versione del file TCALC.PAS del TurboCalc, modificato in modo da consentire il controllo di tutte le eccezioni numeriche, underflow compreso.

per i calcoli interni, interrupt abilitati (l'azzeramento del bit IEM ha senso solo se c'è un 8087, ma non disturba negli altri casi), mascheramento delle eccezioni «precisione» e «underflow».

Quest'ultimo aspetto può meritare un approfondimento. Non stupisce che venga mascherata l'eccezione di precisione, che altrimenti scatterebbe, ad esempio, ogni volta che si dividesse uno per tre, o simili; si lascia quindi che il coprocessore si limiti ad arrotondare senza troppi brontolii. Può sembrare più strano, invece, che non si vogliano segnalazioni di underflow, ma, per così dire, viene riconosciuta una certa libertà di scelta all'utente: il manuale ci avverte infatti che l'errore 206 normalmente non si verifica, e che può verificarsi solo se se si usa il coprocessore o l'emulatore con una parola di controllo che non mascheri la relativa eccezione. Non viene fornito, per la verità, nessuno strumento per modificare la parola di controllo, ma a questo si rimedia facilmente con una procedura *Control87* come quella della figura 3. Vedremo tra breve un esempio.

Se si è rilevata la presenza di un 80387, viene mascherata anche l'eccezione «operando denormalizzato». Ciò avviene in quanto, quando si verifica una tale eccezione con un 8087 o 80287 (o con l'emulatore), il Turbo Pascal non fa altro che chiamare una routine che «normalizza» l'operando, cosa a cui l'80387 provvede automaticamente.

Quando si verifica un'eccezione non mascherata, scatta un interrupt al quale viene associata una routine che, tra le altre cose, salva in AL la parte «destra» della parola di stato, reinizializza il coprocessore con la parola di controllo di default, determina il codice d'errore (200 se è settato il bit ZE, 205 se OE, 206 se UE, altrimenti 207) e salta a quella routine *Terminate* su cui, come abbiamo visto la volta scorsa, si innesta il nostro meccanismo di gestione delle eccezioni. Poiché ciò avviene dopo aver posto in BX:CX l'indirizzo della istruzione successiva a quella che ha provocato l'errore, la nostra unit GESTECC può gestire le eccezioni del coprocessore proprio come le altre.

Due esempi

Nella figura 4 vi propongo un breve programma che illustra sia il controllo delle eccezioni generate dal coprocessore che l'uso della procedura *Control87* per abilitare le eccezioni di underflow: per provocarne una potete immettere, ad esempio, i due valori 1e−300 e 1e300. Vi prego di notare che la procedura *Control87* va chiamata non solo in fase di inizializzazione del programma, ma anche all'interno della funzione *Gestore*, in quanto dopo ogni eccezione generata dal coprocessore il Turbo Pascal reimposta la parola di controllo di default.

Il TurboCalc ci offre poi lo spunto per un esempio ben più sostanzioso. Si tratta di uno stupendo foglio elettronico, magistralmente implementato mediante ricorso alle tecniche della programmazione orientata all'oggetto, nettamente più potente dell'originario MicroCalc. Gli manca solo... la nostra unit GESTECC!

Provate a mettere un numero negativo in A1 e a calcolarne il logaritmo o la radice quadrata in A2. Tutto bene. Le routine in TCPARSER.PAS ben gestiscono questo tipo di situazioni. Provate a mettere 1e30 in A1 e a calcolarne il seno in A2; o a mettere 1e3000 in A1, 1e−3000 in A2 e a calcolare A1/A2 in A3. Il programma cade sotto i colpi dell'errore, rispettivamente, 207 o 205. Per rendere più «robusto» il programma bastano pochi ritocchi. In TCPARSER.PAS aggiungete GESTECC alla lista delle unit «usate» e modificate come in figura 5 le ultime istruzioni della procedura *ParserObj.Parse*. Modificate poi il file TCALC.PAS in modo da attivare la gestione delle eccezioni con la nostra unit (figura 6; se desiderate il controllo anche dell'underflow, compilate dopo aver definito il simbolo UNDERFLOW, dal menu Options/Compiler/Conditional defines, o con "tpc /DUNDERFLOW tcalc"). E il gioco è fatto.

In realtà le modifiche potrebbero (e dovrebbero) essere più consistenti, in quanto sarebbe possibile eliminare tutto il codice che si cura di «prevenire» gli errori (del tipo: se la funzione è logaritmo allora se l'argomento è minore o uguale a zero assegna TRUE ad apposita variabile altrimenti calcola pure il logaritmo; e simili), affidando al gestore di eccezioni il controllo completo delle operazioni.

Sarebbe anche necessario assicurare coerenza con il controllo di altre situazioni (memoria dinamica insufficiente, errori di I/O). Ma non possiamo riscrivere il Turbo Calc sulla rivista.

Già quei pochi ritocchi, tuttavia, vi permetterranno di verificare che, pur non offrendo espressamente un controllo delle eccezioni, il Turbo Pascal è fatto in modo tale che non è difficile provvedervi con proprie routine.