IL COPROCESSORE MATEMATICO 8087

- Il coprocessore 8087
- Architettura interna
- Registri e Flags
- Tipi di dato
- Istruzioni
- Esempio

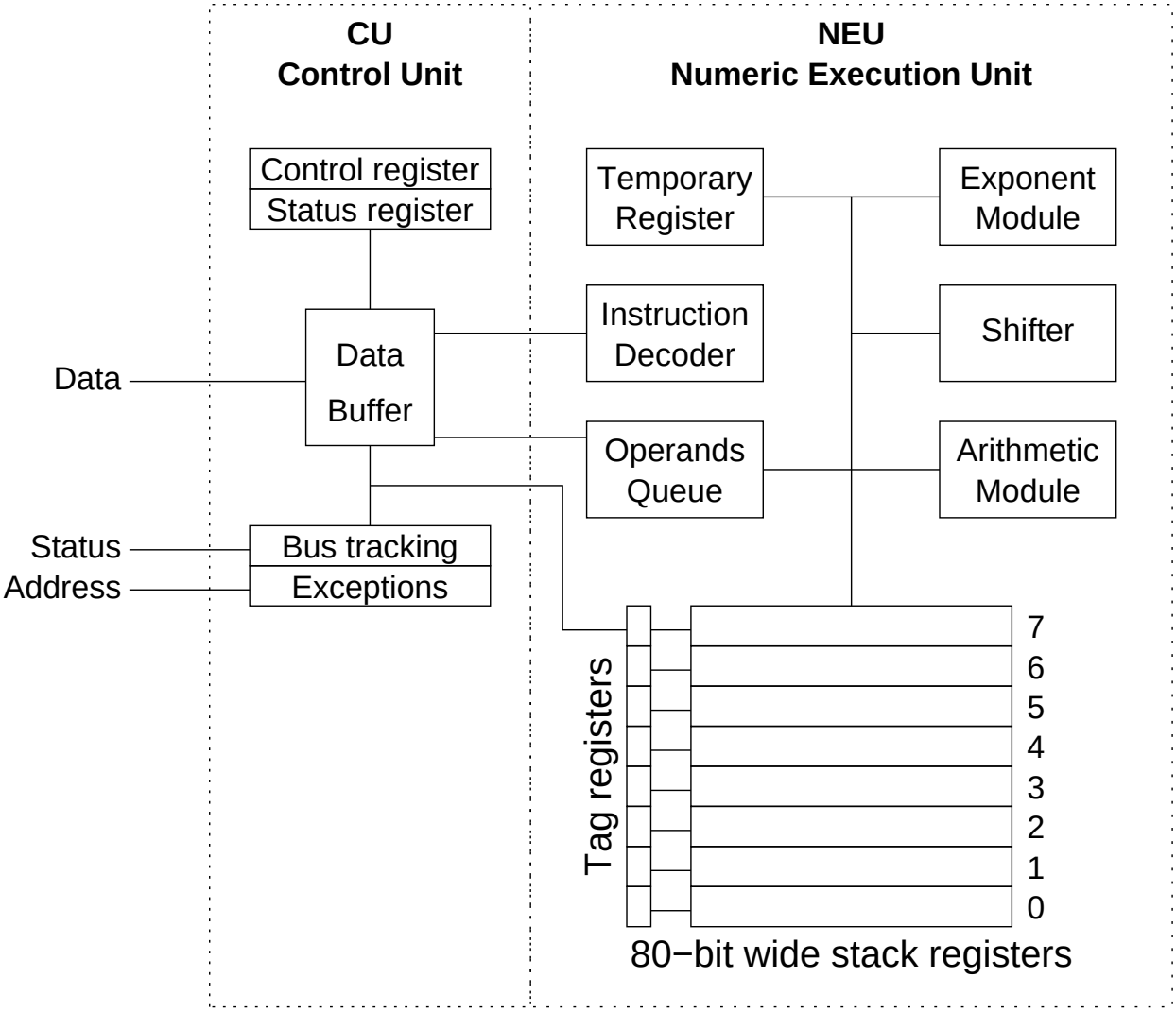
Il coprocessore 8087

L’8087 è un microprocessore dedicato che può essere opzionalmente affiancato all’8086.

Le sue caratteristiche principali sono:

- Dedicato ad eseguire calcoli a virgola mobile
- 8 registri interni a 80+2 bit
- 68 istruzioni
- Possibilità di gestire numeri tra 10^{-400} e 10^{400}
- Modalità di funzionamento concorrente rispetto l’8086
- Dal 80486DX e successivi integrato nella CPU

Architettura interna del coprocessore 8087



Registri dell’8087

L’8087 possiede 8 registri a 80 bit per la gestione dei dati numerici:

S	Esponente		Mantissa	
79	78	64	63	0

Due registri aggiuntivi a 16 bit contengono i flag di controllo e di stato

I registri a 80 bit sono indicati con S(*n*) dove *n* indica la la loro posizione nello stack. Il valore di *n* è conservato nel registro di stato.

Essi sono utilizzabili in modalità LIFO, costituendo uno stack di registri (*stack register*).

La cima dello stack (**ST**) corrisponde a S(*0*).

Tipi di Dato gestibili

L’8087 può gestire i seguenti tipi di dato:

Word Integer: intero a 16 bit in complemento a 2

Short Integer: intero a 32 bit in complemento a 2

Long Integer: intero a 64 bit in complemento a 2

Short Real: valore a virgola mobile su 32 bit, 1 bit di segno, 8 bit esponente e 23 bit mantissa

Long Real: valore a virgola mobile su 64 bit, 1 bit di segno, 11 bit esponente e 52 bit mantissa

Temporary Real: valore a virgola mobile su 80 bit, 1 bit di segno, 15 bit esponente e 64 bit mantissa

Packed decimal: intero fino a 18 cifre decimali codificate (4 bit ciascuna) più un bit di segno

Internamente, solo la rappresentazione *Temporary Real* è utilizzata.

Istruzioni

L’8087 aggiunge ulteriori 68 istruzioni all’assembly dell’8086, raggruppabili nelle seguenti classi:

- Trasferimento Dati
- Aritmetiche
- Di comparazione
- Trascendenti
- Generazione costanti
- Controllo del Coprocessore

Nota: tutte le istruzioni dell’8087 iniziano per F

Istruzioni di Trasferimento Dati

	OpCode	Descrizione
A virgola mobile	FLD	Load Real onto ST
	FST	Store Real from ST
	FSTP	Store Real from ST and POP
	FXCH	Exchange Register with ST
Interi	FILD	Load Integer onto ST
	FIST	Store Integer from ST
	FISTP	Store Integer from ST and POP
Decimali codificati	FBLD	Load BCD onto ST
	FBSTP	Store BCD from ST and POP

Istruzioni Aritmetiche

	OpCode	Descrizione
Addizione	FADD	Addition
	FADDP	Addition and POP
	FIADD	Integer addition
Sottrazione	FSUB	Subtract
	FSUBP	Subtract and POP
	FISUB	Integer subtract
	FSUBR	Reverse subtract
	FSUBRP	Reverse subtract and POP
	FISUBR	Reverse integer subtract
Moltiplicazione	FMUL	Multiplication
	FMULP	Multiplication and POP
	FIMUL	Integer multiplication
Divisione	FDIV	Division
	FDIVP	Division and POP
	FIDIV	Integer division
	FDIVR	Reverse division
	FDIVRP	Reverse division and POP
	FIDIVR	Reverse integer division
Altre	FPREM	Partial remainder
	FRNDINT	Rounds ST to an integer
	FABS	Absolute value of ST
	FCHS	Change sign
	FXTRACT	Extract exponent and significand of ST
	FSCALE	Scale by factor of 2

Uso dello Stack Register

Le istruzioni aritmetiche dell'8087 che coinvolgono due operandi hanno 4 modalità di funzionamento:

Classico: non vengono indicati operandi, coinvolge ST e S(1). Lo stack viene ruotato e il risultato posto in ST.

```
FADD                ; S(1) = S(1) + ST,
                    ; pop ST(1) in ST
```

Con accesso in memoria: viene indicato un solo operando (un dato in memoria), il secondo è implicitamente ST.

```
OP1     DD      3.62

FADD    OP1         ; ST = ST + OP1
```

Registro: entrambi gli operandi sono registri, uno di questi deve essere ST.

```
FADD    ST,ST(3)    ; ST = ST + ST(3)
FIMUL   ST(4),ST    ; ST(4) = ST(4)*ST
```

Registro e POP: come il precedente, ma in questo caso ST viene rimosso e lo stack ruotato.

```
FADDP   ST(1),ST    ; ST(1) = ST(1)*ST,
                    ; pop ST out of stack
```

Istruzioni per funzioni Trascendenti

	OpCode	Descrizione
Trigonometriche	FPTAN	Partial tangent
	FPATAN	Partial arctangent
	FSIN*	Sine
	FCOS*	Cosine
	FSINCOS*	Sine and cosine
Logaritmiche	FYL2X	Compute $y \times \log_2(x)$
	FYL2XP1	Compute $y \times \log_2(x+1)$
Esponenziali	FSQRT	Square root
	F2XMI	Compute $2^x - 1$

*dall'80387 e successivi

Istruzioni di Comparazione e Costanti

	OpCode	Descrizione
Costanti		
	FLDZ	Load 0.0 on ST
	FLD1	Load 1.0 on ST
	FLDL2T	Load $\log_2 10$ on ST
	FLDL2E	Load $\log_2 e$ on ST
	FLDLG2	Load $\log_{10} 2$ on ST
	FLDLN2	Load $\log_e 2$ on ST
Comparazione		
	FCOM	Floating point compare
	FCOMP	Floating point compare and pop
	FCOMPP	Floating point compare and pop twice
	FICOM	Integer compare
	FICOMP	Integer compare and pop
	FTST	Compare ST against a 0
	FXAM	Examine ST

Istruzioni di Controllo*

	OpCode	Descrizione
Sincronia	FINIT	Initialize FPU
	FNINIT	Initialize FPU, no wait
	FWAIT	Wait while FPU is executing
	FNOP	No operation
Trasferimento stato	FLDCW	Load control register
	FSTCW	Store control register
	FSTSW	Store status register
	FSAVE	Save FPU status
	FRSTOR	Load FPU status
	FSTENV	Save FPU environment
	FLDENV	Load FPU environment
Gestione stack register	FINCST	Increment stack pointer
	FDECSTP	Decrement stack pointer
	FFREE	Free a register
Controllo	FDISI	Disable interrupts
	FENI	Enable interrupts
	FCLEX	Clear error and busy flags

*alcune istruzioni hanno due varianti

Esempio

Il frammento di codice che segue calcola la frequenza di risonanza di un circuto LC: $R = \frac{1}{2\pi\sqrt{LC}}$

```
RIS         DD    ?
C1          DD    0.000001
L1          DD    0.000001
DUE         DD    2.0

...
            FINIT       ; inizializzazione 8087
            FLD   L1    ; carica L1 in ST
            FMUL  C1    ; calcola C1*L1 in ST
            FSQRT       ; radice di C1*S1
            FMUL  DUE   ; moltiplico per due
            FLDPI       ; carico PI su ST, il valore
                        ; precedente finisce in S(1)
            FMUL        ; ST = ST*S(1)
            FLD1        ; carico 1 in ST, il valore
                        ; precedente finisce in S(1)
            FDIVR       ; ST=ST/S(1)
            FSTP RIS    ; scrivo il risultato in memoria
            FWAIT       ; sincronizzo l'8086 con l'8087
```