

# Unità di Controllo Microprogrammate

- Introduzione
- Caratteristiche
- Ottimizzazione della Memoria di Controllo
- L'Unità di Controllo dell'8088
- Nanoprogrammazione.

1

M. Sonza Reorda - a.a. 2001/02

## Introduzione

La microprogrammazione fu proposta per la prima volta da M.V. Wilkes attorno al 1950.

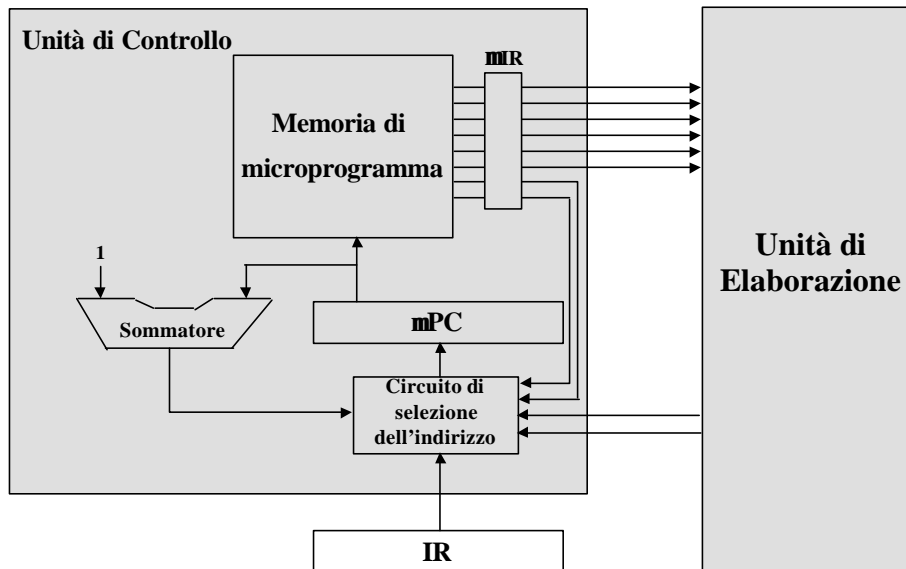
L'insieme di segnali di controllo prodotti dalla UC ad un certo istante viene denominato *parola di controllo* (Control Word, CW), o *microistruzione*.

Ciascuna istruzione di un mP corrisponde ad una sequenza di microistruzioni (*microprogramma*), memorizzate in una memoria apposita (*memoria di microprogramma*) ed aventi un formato prefissato.

2

M. Sonza Reorda - a.a. 2001/02

## Architettura



3

M. Sonza Reorda - a.a. 2001/02

## Funzionamento

- Si esegue una lettura dalla *Memoria di Microprogramma*, utilizzando il contenuto del *Micro Program Counter* come indirizzo.
- La parola corrispondente viene caricata nel *Micro Instruction Register*.
- Il contenuto del *Micro Instruction Register* pilota i segnali di controllo per l'Unità di Elaborazione e per la logica di generazione dell'indirizzo della successiva microistruzione.
- Tale logica genera un nuovo indirizzo, sulla base anche dei segnali provenienti dall'esterno (ad esempio dall'Instruction Register in una CPU).

Tutte le operazioni vengono eseguite in un solo colpo di clock.

4

M. Sonza Reorda - a.a. 2001/02

## Generazione dell'indirizzo della microistruzione successiva

L'indirizzo della microistruzione successiva può essere:

- quello successivo
- un indirizzo fornito dall'esterno (ad esempio l'inizio di un microprogramma)
- un indirizzo di salto (condizionato o incondizionato).

Nell'ultimo caso il nuovo indirizzo è contenuto nella microistruzione stessa. In particolare esso può corrispondere:

- ad un campo sempre esistente nella microistruzione
- ad un campo presente solo nelle microistruzioni di salto, riconoscibili per un codice particolare.

5

M. Sonza Reorda - a.a. 2001/02

## Caratteristiche

- + **Flessibilità:** modificare una istruzione o aggiungerne di nuove comporta semplicemente la modifica del contenuto della memoria di microprogramma.
- **Velocità:** l'esecuzione di una istruzione richiede una serie di accessi alla memoria di microprogramma, e può quindi essere più lenta che in una UC cablata.
- **Costo:** la presenza della memoria di microprogramma e della relativa logica fa crescere il costo in termini di hardware.

6

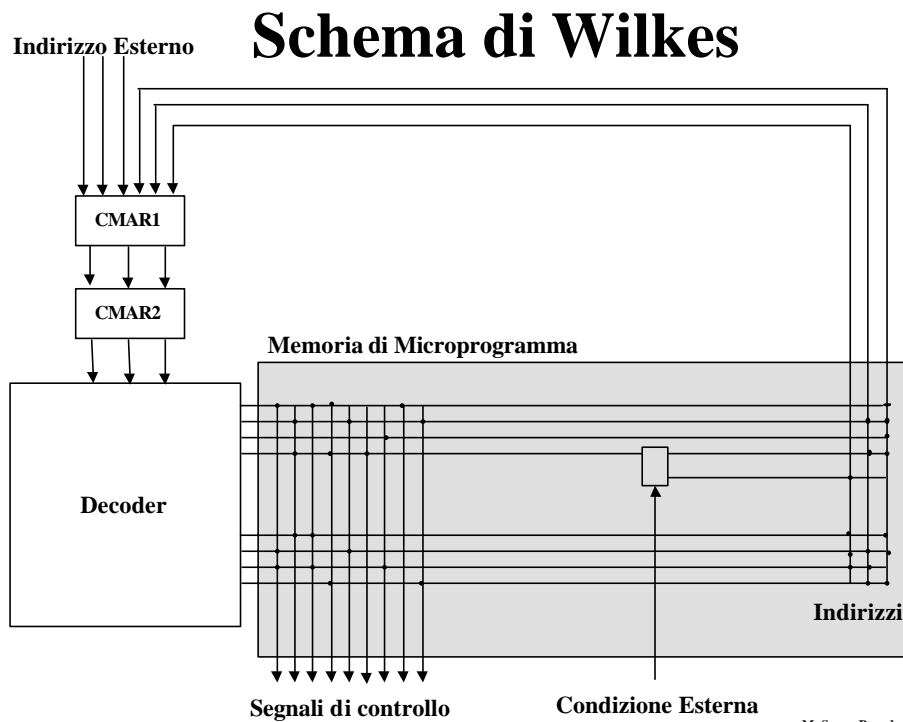
M. Sonza Reorda - a.a. 2001/02

# Firmware

Il progetto di un processore microprogrammato richiede lo sviluppo del relativo microcodice, detto *firmware*.

7

M. Sonza Reorda - a.a. 2001/02



8

M. Sonza Reorda - a.a. 2001/02

## Formato delle Microistruzioni

Dipende da come si risolvono 2 problemi:

- la determinazione dell'indirizzo della successiva microistruzione
- la codifica dei segnali di controllo.

9

M. Sonza Reorda - a.a. 2001/02

## Indirizzo della Microistruzione Successiva

Se non si hanno istruzioni di salto è dato da quello dell'istruzione precedente, opportunamente incrementato.

Può essere specificato in vari modi alternativi:

- 1 ogni microistruzione ha 2 campi aggiuntivi, che vengono utilizzati dalle microistruzioni di salto condizionato per contenere i 2 possibili indirizzi della microistruzione successiva
- 2 ogni microistruzione ha un campo aggiuntivo, che viene utilizzato dalle microistruzioni di salto condizionato per contenere il possibile indirizzo di salto
- 3 le microistruzioni di salto hanno un formato diverso da quello delle microistruzioni normali.

10

M. Sonza Reorda - a.a. 2001/02

## Formato delle Microistruzioni di Salto (modo 3)

Codice mIstruz.	
-----------------	--

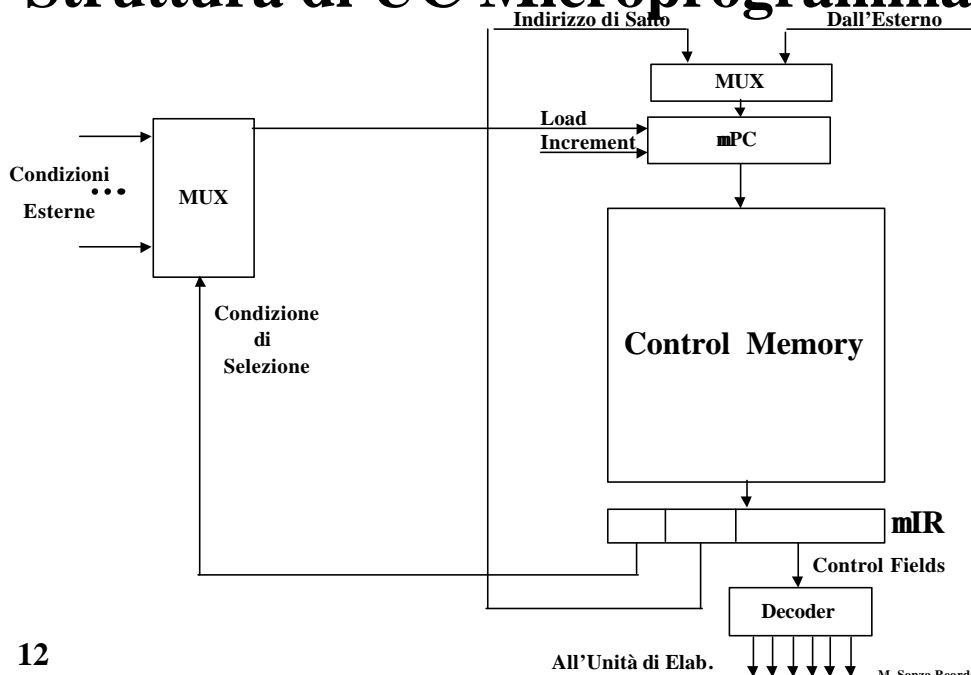
Possibilità:

mIstruz. Normale	Segnali di controllo	
mIstruz. Salto Incondiz.	Indirizzo a cui saltare	
mIstruz. Salto Condiz.	Condizioni	Indirizzo a cui saltare

11

M. Sonza Reorda - a.a. 2001/02

## Struttura di UC Microprogrammata



12

All'Unità di Elab.

M. Sonza Reorda - a.a. 2001/02

## Parallelismo delle Microistruzioni

Nel caso più semplice (microistruzioni *orizzontali*), le microistruzioni contengono un bit per ogni segnale di controllo; in tal caso si ottiene:

- massimo parallelismo nell'esecuzione: si possono eseguire contemporaneamente molte microoperazioni
- massima velocità di esecuzione: la microistruzione può essere trasformata in segnali di controllo senza bisogno di manipolazioni.

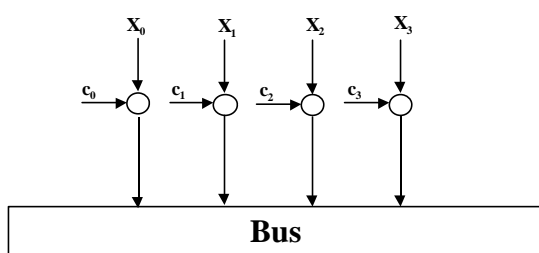
Esistono tuttavia alcune controindicazioni:

- la lunghezza delle microistruzioni può divenire eccessiva
- talune combinazioni di valori dei segnali di controllo possono non verificarsi mai.

13

M. Sonza Reorda - a.a. 2001/02

## Esempio



Configurazioni possibili:

0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

14

M. Sonza Reorda - a.a. 2001/02

# Microistruzioni Verticali

Contengono le informazioni da inviare ai segnali di controllo in maniera codificata.

Hanno una lunghezza più ridotta.

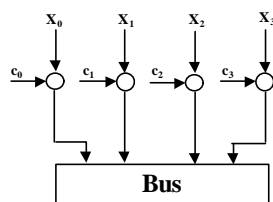
Sono meno adatte a descrivere microoperazioni contemporanee.

Sono anche possibili soluzioni intermedie, in cui ogni gruppo di segnali di controllo corrisponde ad un gruppo di bit (*Control Field*) della singola microistruzione, il cui valore è codificato.

15

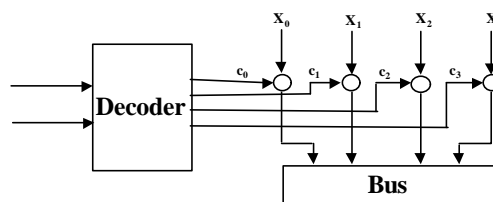
M. Sonza Reorda - a.a. 2001/02

## Esempio



Configurazioni possibili:

- 0 0 0 1
- 0 0 1 0
- 0 1 0 0
- 1 0 0 0



16

M. Sonza Reorda - a.a. 2001/02



## Compatibilità

Due segnali di controllo sono *compatibili* se non sono mai attivati nella stessa microistruzione.

Essi possono dunque essere associati allo stesso control field.

Una *classe di compatibilità* è un insieme di segnali di controllo i quali sono a 2 a 2 compatibili.

I segnali di controllo di una classe di compatibilità sono associabili ad uno stesso insieme di control field.

19

M. Sonza Reorda - a.a. 2001/02

## Esempio

microistruzione	segnali di controllo							
I <sub>1</sub>	a	b	c					g
I <sub>2</sub>	a		c		e			h
I <sub>3</sub>	a			d		f		
I <sub>4</sub>		b	c			f		

20

M. Sonza Reorda - a.a. 2001/02

## Indirizzamento delle Microistruzioni

Avviene tramite un registro denominato **Microprogram Counter (mPC)**: esso contiene l'indirizzo della successiva microistruzione nella memoria di microprogramma.

Il mPC può:

- venire caricato con il contenuto di un apposito campo, presente in ogni microistruzione
- venire incrementato dopo ogni caricamento.

21

M. Sonza Reorda - a.a. 2001/02

## Temporizzazione Polifase

Consiste nel sincronizzare le microoperazioni su segnali di temporizzazione diversi, tra loro sfasati, anzichè su un segnale solo.

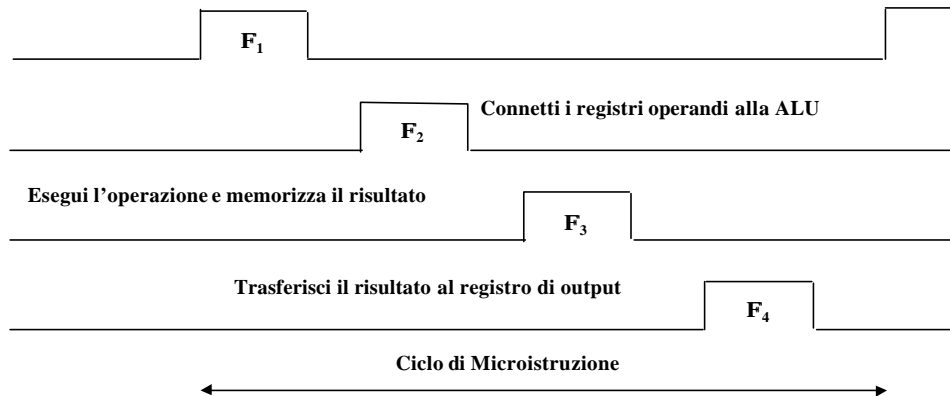
In tal modo si può accrescere il numero di microoperazioni corrispondenti a ciascuna microistruzione.

22

M. Sonza Reorda - a.a. 2001/02

## Esempio

Genera l'indirizzo della micoristruzione successiva  
ed esegui il fetch dalla memoria di microprogramma



23

M. Sonza Reorda - a.a. 2001/02

## Emulazione

**Modificando opportunamente il contenuto della memoria di microprogramma si può fare in modo che un processore riconosca il linguaggio di un altro.**

**Tale operazione permette di costruire processori più evoluti (ad esempio perché dotati di Unità di Elaborazione più efficienti), mantenendo la completa compatibilità software con le versioni precedenti.**

24

M. Sonza Reorda - a.a. 2001/02

# Applicazioni della Microprogrammazione

- **Emulazione**
- **Supporto per Sistemi Operativi**
- **Dispositivi Special-Purpose**
- **Supporto per Linguaggi ad Alto Livello**
- **Microdiagnosi**
- **Hardware Personalizzabile.**

25

M. Sonza Reorda - a.a. 2001/02

# Esempi di CPU Microprogrammate

- **IBM System 360/370**
- **Intel 80x86**
- **Motorola 680x0**

26

M. Sonza Reorda - a.a. 2001/02

## Microprogram Sequencer

Sono dispositivi atti a semplificare la realizzazione di una UC microprogrammata, qualora questa avvenga con logica sparsa.

Contengono la logica per:

- implementare il **mPC** ed il **mIR**
- gestire i salti condizionati
- fornire supporto per il debug del microcodice.

27

M. Sonza Reorda - a.a. 2001/02

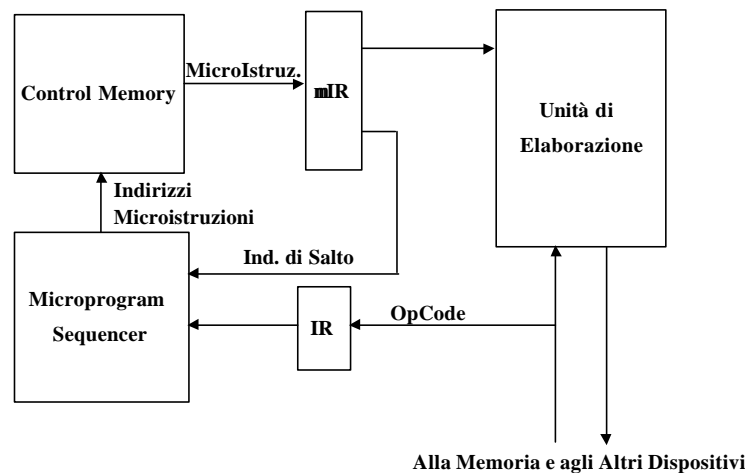
## Esempi

- AMD 2909 (bit slice, parallelismo 4 bit)
- TI 8835 (parallelismo 16 bit).

28

M. Sonza Reorda - a.a. 2001/02

## Struttura di una CPU con Microprogram Sequencer



29

M. Sonza Reorda - a.a. 2001/02

## L'Unità di Controllo dell'8088

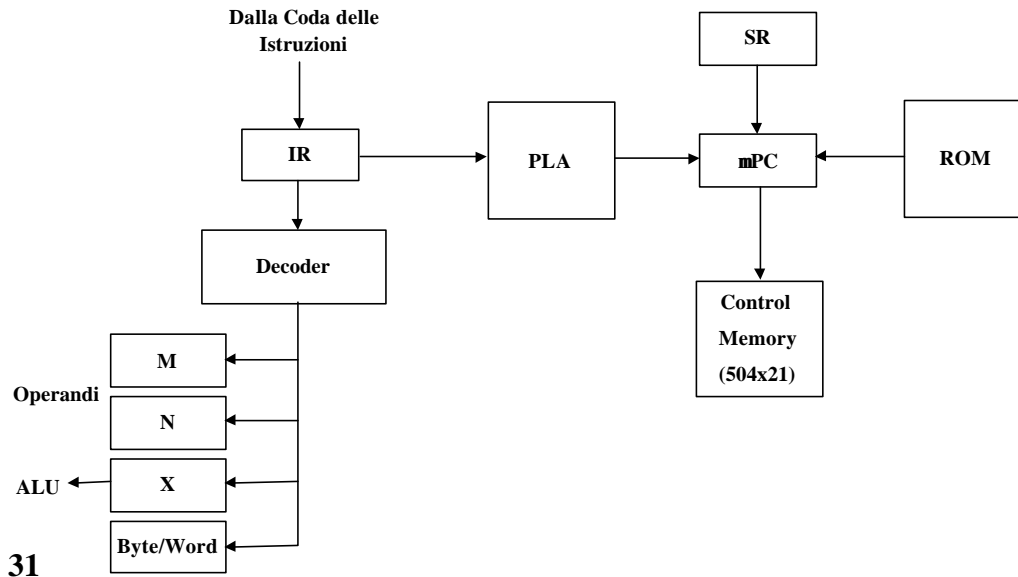
Caratteristiche generali:

- l'Unità di Controllo dell'8088 (come quella dell'8086) è in parte hardwired ed in parte microprogrammata
- le microistruzioni hanno ampiezza pari a 21 bit
- il formato delle microistruzioni è verticale
- la memoria di microcodice contiene 504 microistruzioni.

30

M. Sonza Reorda - a.a. 2001/02

## Unità di Controllo dell'8088



M. Sonza Reorda - a.a. 2001/02

## Funzionamento

- La PLA converte il codice operativo dell'istruzione nell'IR nell'indirizzo della prima microistruzione corrispondente.
- Il microcodice è diviso in *burst*, composti al più da 16 microistruzioni: ogni burst corrisponde ad una istruzione, oppure ad una operazione generale (ad esempio il calcolo degli indirizzi).
- Esistono 2 tipi di salti:
  - quello *breve*, all'interno del burst
  - quello *lungo*, per saltare ovunque.
- Sono supportate le microprocedure, il cui indirizzo di ritorno viene salvato nel registro SR (*Subroutine Return*).

32

M. Sonza Reorda - a.a. 2001/02

## Istruzioni di Calcolo

- Le istruzioni di calcolo corrispondono a 2 fasi:
  - *calcolo degli indirizzi degli operandi* (posti nei registri M e N) e *decodifica dell'operazione* che la ALU deve eseguire (nel registro X)
  - *esecuzione* del calcolo.
- La stessa microprocedura può quindi eseguire operazioni diverse, a seconda dei valori presenti nei registri M, N ed X.

33

M. Sonza Reorda - a.a. 2001/02

## Formato delle Microstruzioni

20	15	10	7	3	0
SRC	DEST	TYPE	ALU	REG	CC

- La parte sinistra permette il trasferimento tra registri.
- La parte destra è codificata verticalmente.
- TYPE seleziona il tipo di microistruzione:
  - operazione della ALU
  - operazione sulla memoria
  - salto breve
  - salto lungo
  - chiamata di microprocedura
  - prenotazione.

34

M. Sonza Reorda - a.a. 2001/02

## Formato delle Microstruzioni (II)

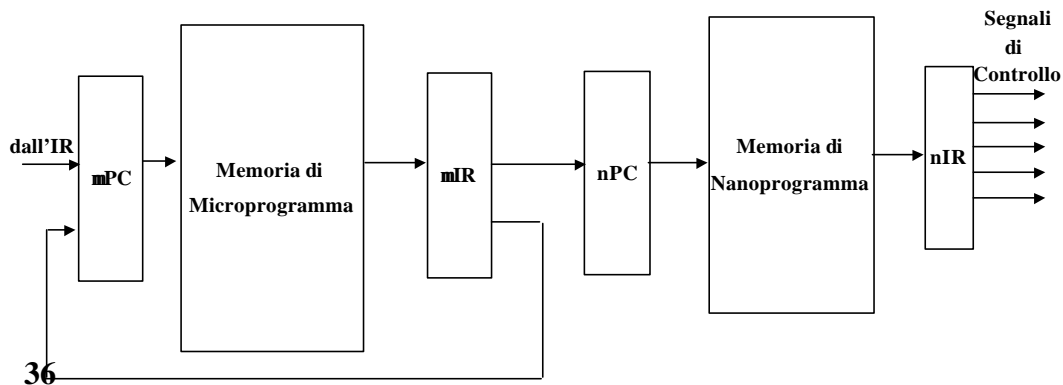
- Il campo ALU può specificare una funzione, oppure indicare quella specificata dal registro X.
- REG specifica gli operandi.
- CC specifica se si devono settare i flag.
- I salti lunghi ed i salti a procedura possono essere fatti solo ad uno tra 32 indirizzi predefiniti, identificabili tramite 5 bit: da questi la ROM produce l'indirizzo destinazione.

35

M. Sonza Reorda - a.a. 2001/02

## Nanoprogrammazione

Nella UC nanoprogrammate il contenuto della memoria di microprogramma non corrisponde ai segnali di controllo da inviare all'Unità di Elaborazione, ma ad indirizzi da inviare ad una successiva memoria (*memoria di nanoprogramma*) che contiene i segnali di controllo.



M. Sonza Reorda - a.a. 2001/02

## Caratteristiche

### Vantaggi

- la nanoprogrammazione può far diminuire la quantità di memoria necessaria, purchè il numero di microistruzioni distinte sia piccolo rispetto al numero totale

### Svantaggi

- la nanoprogrammazione è più lenta della microprogrammazione

### Applicazioni

- nei processori ad alto livello di integrazione (ad es. i Motorola 68000).

37

M. Sonza Reorda - a.a. 2001/02

## Analisi

Se si usa la sola memoria di microcodice, le sue dimensioni sono:

$$S_1 = H_m(N + \epsilon \log_2 H_m \bar{u})$$

Se si usa anche la memoria di nanoprogramma:

$$S_2 = H_m(\epsilon \log_2 H_m \bar{u} + \epsilon \log_2 H_n \bar{u}) + N H_n$$

Se il rapporto tra il numero di microistruzioni differenti rispetto al numero totale è  $r$ , allora  $H_n = r H_m$ , e si ha:

$$\begin{aligned} S_2 &= H_m(\epsilon \log_2 H_m \bar{u} + \epsilon \log_2 r H_m \bar{u} + N r) \\ &= H_m(2\epsilon \log_2 H_m \bar{u} + \epsilon \log_2 r \bar{u} + N r) \end{aligned}$$

La nanoprogrammazione è quindi conveniente se

$$N > \epsilon \log_2 H_m \bar{u} + \epsilon \log_2 r \bar{u} + N r$$

38

M. Sonza Reorda - a.a. 2001/02

## Motorola 68000

Adotta i seguenti valori:

- $N=70$
- $H_m=650$
- $r=0.4$
- $H_n=260$

Si ottiene

$$S_1=52450 \quad S_2=30550$$

con un guadagno di 21850 bit (42%) grazie alla nanoprogrammazione.