UNITELMA SAPIENZA

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Architettura degli Elaboratori
Lez. 1 – Introduzione al corso

Prof. Andrea Sterbini – sterbini@di.uniroma1.it

Introduzione al corso

UNITELMA SAPIENZA

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Libro di testo:

David A. Patterson, John L. Hennessy, "STRUTTURA E PROGETTO DEI CALCOLATORI", Zanichelli

Esame:

- Prova scritta di teoria
- Prova pratica di Assembler (in laboratorio)
- Orale (ammessi se le altre prove sono superate)

Argomenti del corso:

- Introduzione storica e struttura di una CPU
- L'assembler MIPS
- Progetto della CPU MIPS ad un colpo di clock
- Introduzione alla Pipeline
- Progetto della CPU MIPS con pipeline
- Gestione degli hazard ed eccezioni
- Parallelismo
- Gerarchia di memoria e Cache
- Memoria Virtuale e protezione

GENERAZIONE “-100”

I primi esempi di macchina **meccanica**

150-100 AC:
macchina di **Antikythera** per predire le eclissi

1200 DC: Automi di **Al-Jazari**

Meccanismi “programmabili” a camme e piston
(p.es. barca con banda musicale)

Medio Evo: Automi umanoidi

il “Moro” giocatore di scacchi (in realtà conteneva un nano)

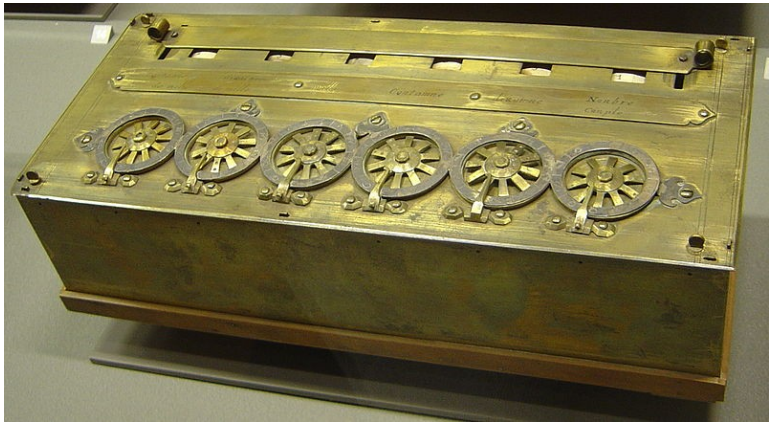
Lo scrivano, il disegnatore ed il musicista di Drosz

Il cavaliere meccanico in armatura (Leonardo da Vinci)

… molti altri

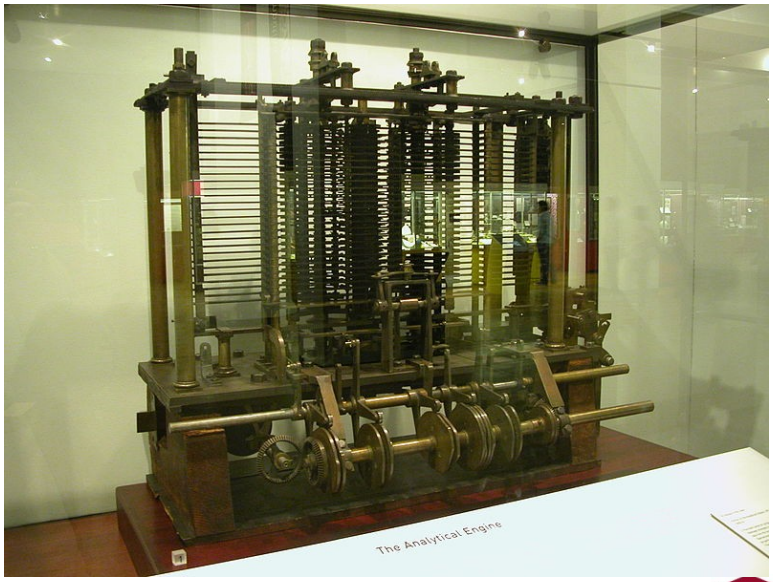
GENERAZIONE “0”

1642: **Blaise Pascal**: la Pascalina calcolatrice meccanica



1833: **Charles Babbage**
Analytical Engine

1839: Telaio automatico **Jacquard**
(«programmabile» a schede perforate!)



GENERAZIONE 1

I primi computer **digitali**

1937 ABC (US)

Atanasoff-Berry Computer

X risolvere sistemi di equazioni linear

1941 Z3 (Konrad Zuse)

2200 relè (elettromeccanico)

Programmato con nastro perforato

1944 Colossus (UK)

2400 Valvole

Cablato (Programmato collegando tra loro le parti a seconda del compito da svolgere

Per decodificare i messaggi nazisti

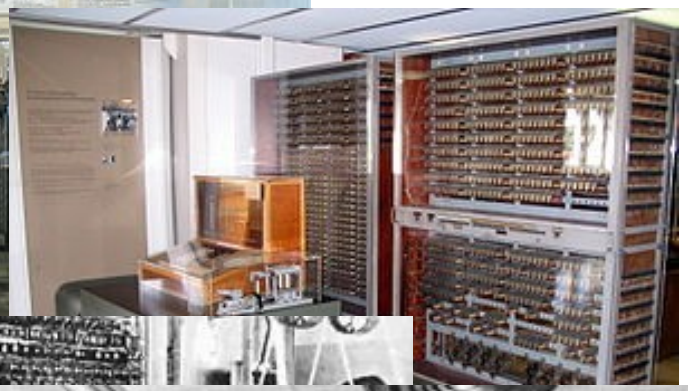
1946 ENIAC (US)

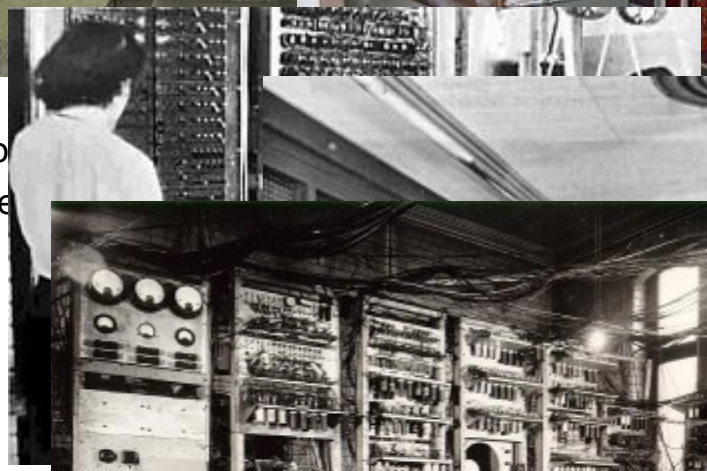
18000 Valvole, Cablato, General purpose

1948 Mark 1 (UK)

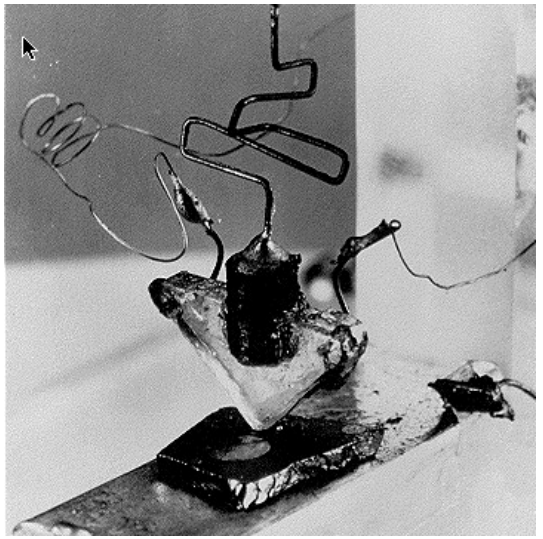
Il primo computer a programma memorizzato

Programmato in **Autocode**





GENERAZIONE 2



1947: Introduzione dei **transistor**

Piccoli, robusti, veloci, consumano poco

(le valvole si rompevano o bruciavano)

NCR, RCA, IBM: i primi computer commerciali

ALU complesse

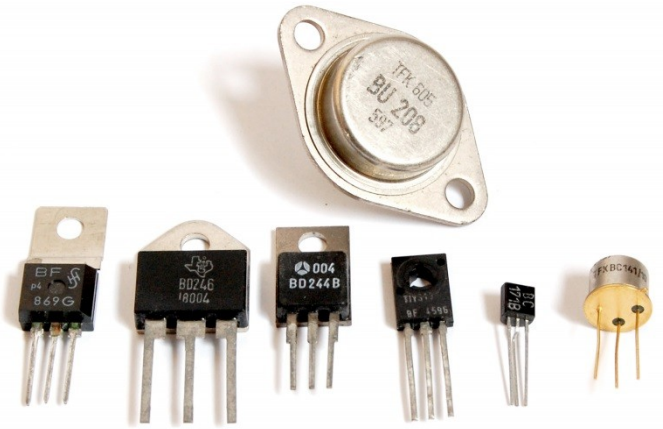
Trasferimento Diretto in Memoria (**DMA**)

Linguaggi di alto livello

FORTRAN: calcoli numerici

ALGOL: simulazioni (a oggetti!)

COBOL: calcoli finanziari



DEC sviluppa i primi Minicomputer

GENERAZIONE 3

1958: Introduzione dei **circuiti integrati**

1964:

IBM serie 360: compatibili “verso l'alto”

- Stesse istruzioni (+ istr. avanzate)
- Stesso Sistema Op. (+ funz. avanzate)
- Velocità crescente
- Memoria crescente
- Parallelismo crescente
- Molto costoso (100K$)

DEC PDP-8: poco costoso (16K$ -> 2500$)

- **Bus** di interconnessione **standardizzato**
- Facilità di espansione (produttori OEM)

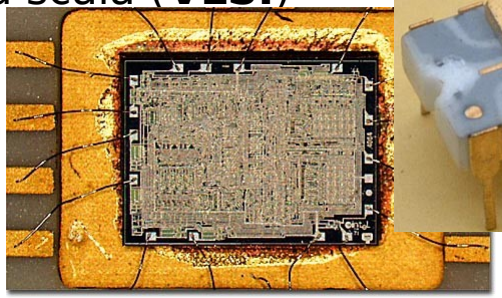
GENERAZIONE 4

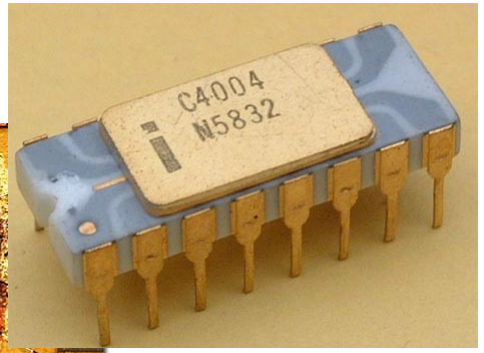
Circuiti integrati su larga e larghissima scala (**VLSI**)

1971:

Intel 4004: I° processore integrato (a

Memorie integrate





Legge (osservazione) di **Moore**:

"la densità di transistor raddoppia ogni 12-18 mesi"

Ne segue che:

=> Il costo di produzione resta uguale

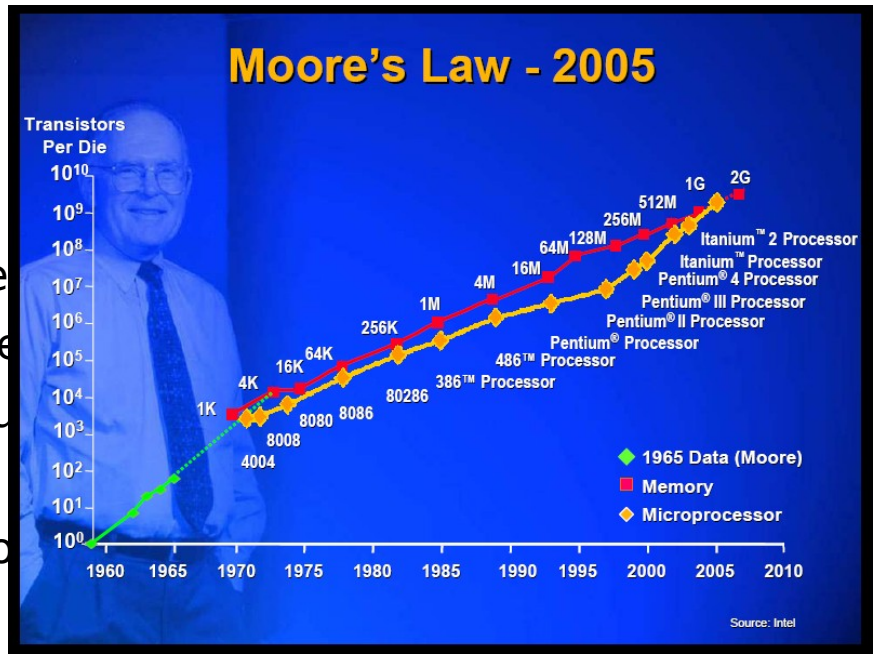
=> Il prezzo per componente decresce

=> La distanza tra componenti diminu

=> La velocità aumenta

=> L'energia ed il calore diminuiscono

=> L'affidabilità aumenta



Architettura di Von Neumann a programma memorizzato

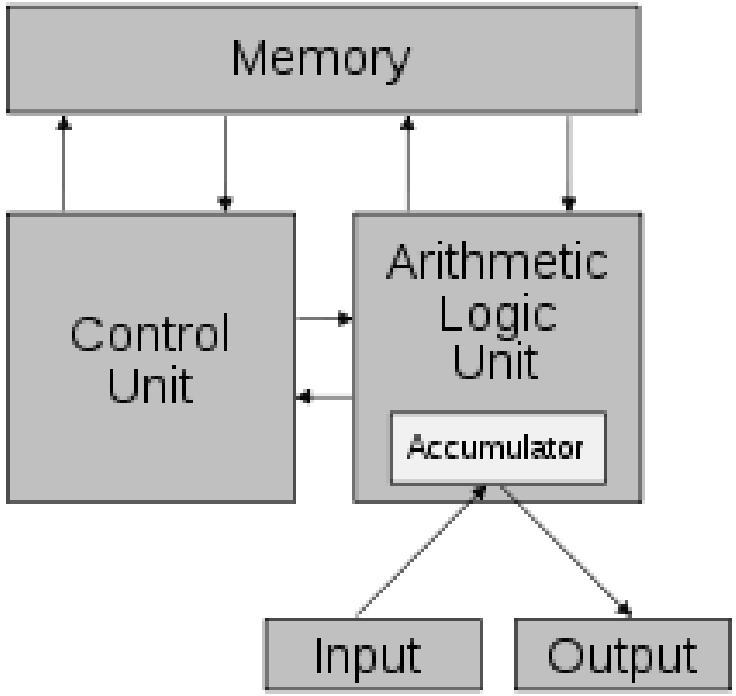
Un computer è diviso in:

Memoria
dati e **programmi**

CPU (CU + ALU + registri)

Bus di comunicazione tra le diverse parti

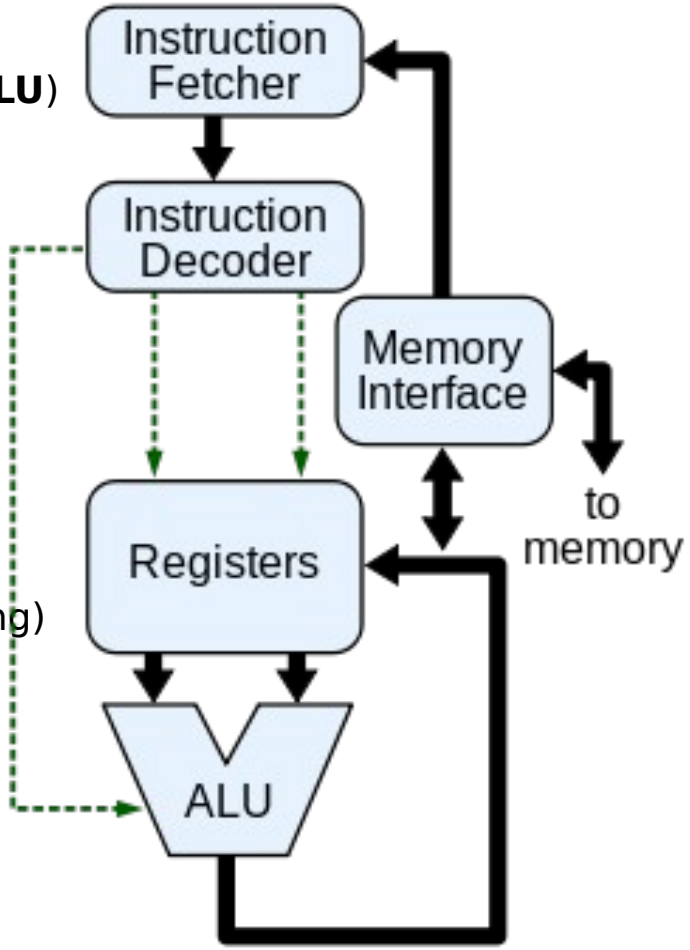
Periferiche di I/O (tastiera, schermo, stampante, scheda di rete eccetera)



La CPU

La **C**entral **P**rocessing **U**nit è formata da:

- La **U**nità di elaborazione **A**ritmetico/**L**ogica (**ALU**)
  - fa solo calcoli
  - NON ha uno stato interno
- I **registri**
  - A uso generale e speciale,
  - per manipolare istruzioni, indirizzi, dati, risultati
- Il **bus** di comunicazione con la **M**emoria
- Il **bus** di comunicazione con le periferiche
  - (non necessario col memory mapping)
- La **U**nità di **C**ontrollo che coordina il tutto
  - (Instruction Decoder nella figura)



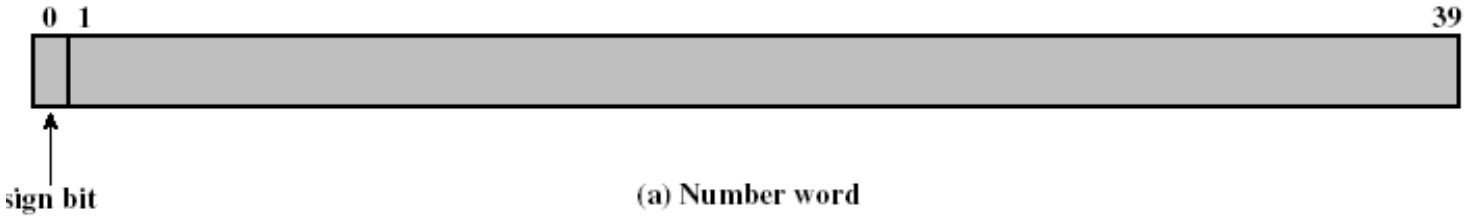
Esempio: la IAS machine

1951 : IAS machine – Institute for Advanced Studies – Princeton

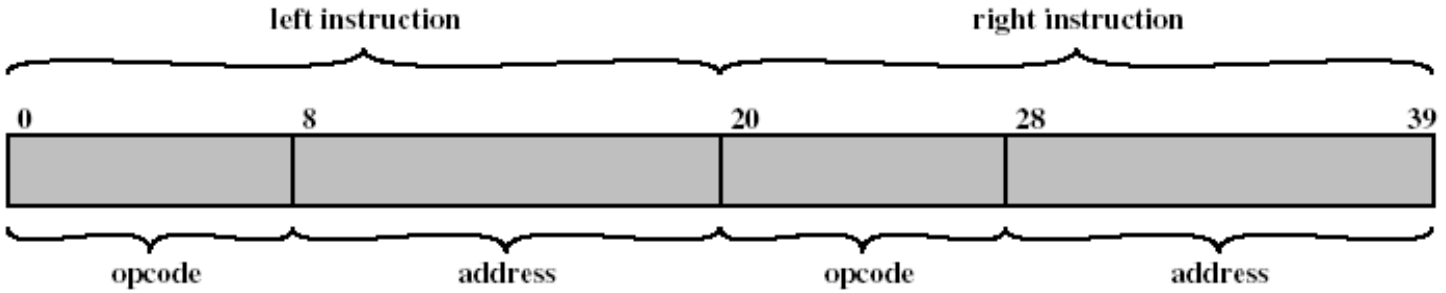
Aveva una memoria di 1000 word da 40 bit ciascuna (ovvero 5 Kbyte totali!!!)

dati: solo interi nella rappresentazione in Complemento a 2

istruzioni: 2 istruzioni per ciascuna word



(a) Number word



(b) Instruction word

La CPU della IAS machine

MBR: Memory Buffer Register
riceve/manda il dato dalla/alla mer

MAR: Memory Address Register
indica l'indirizzo in memoria

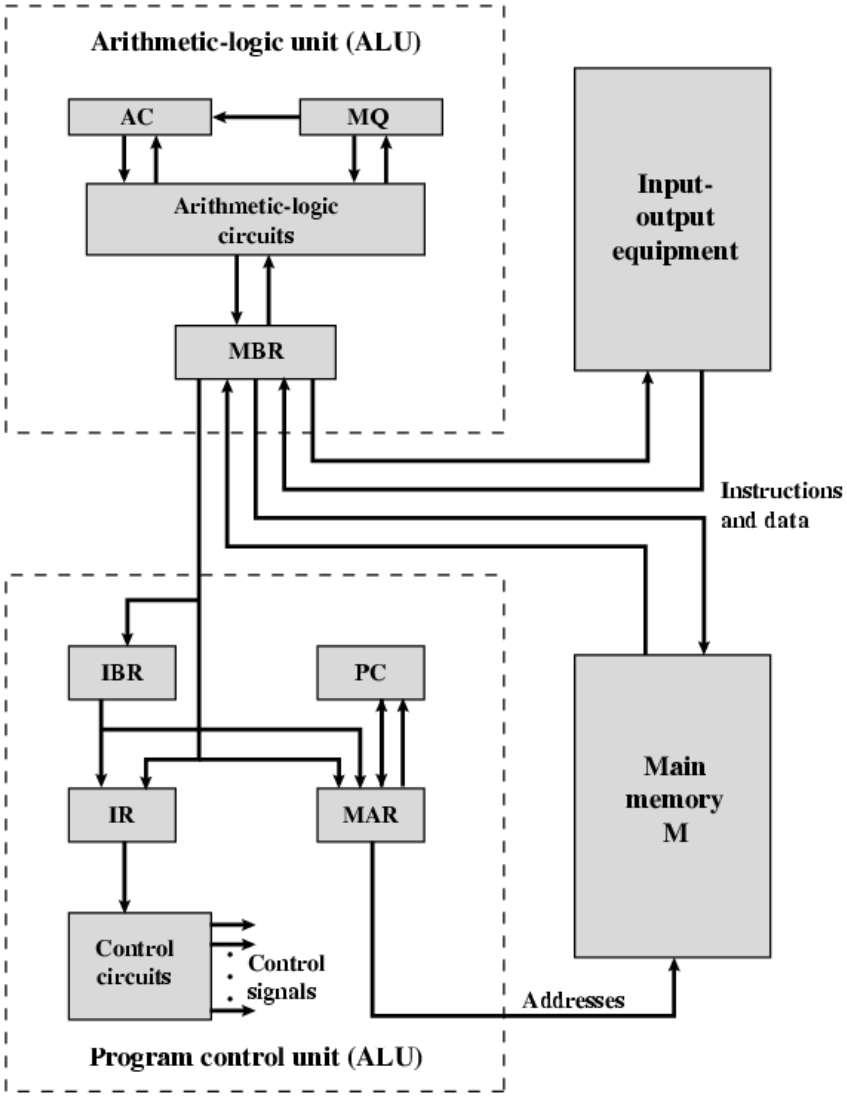
PC: Program Counter
indica l'indirizzo dell'istr. da esegui

IR: Instruction Register
riceve l'istruzione da eseguire

IBR: Instruction Buffer Register
contiene la 2° istruzione della word

AC: Accumulatore
per i risultati parziali dei calcoli

MQ: Multiplier Quotient
per i risultati parziali dei calcoli



Istruzioni di trasferimento

LOAD caricamento dalla memoria con operazioni semplici (ABS, NEG, ADD, SUB) con oppure senza il valore precedente di AC

AC <- AC <operazione> Memory(Address)

AC <- <operazione> Memory(Address)

LDMQ caricamento nel registro MQ

MQ <- Memory(Address)

ST memorizzazione di AC come dato

Memory(Address) <- AC

AMODL memorizzazione di AC come indirizzo in una istruzione (parte bassa della word)

Memory(Address)[bits 0:11] <- AC[bits 0:11]

AMODH memorizzazione di AC come indirizzo in una istruzione (parte alta della word)

Memory(Address) [bits 20:31] <- AC[bits 0:11]

Istruzioni di salto

Salti NON condizionati

UBL salto incondizionato all'istruzione (parte bassa della parola)

PC <- Address; offsetPC <- 0

UBH salto incondizionato all'istruzione (parte alta della parola)

PC <- Address; offsetPC <- 1

Salti condizionati

CBL salto condizionato all'istruzione (parte alta della parola)

if AC >= 0 then PC <- Address; offsetPC <- 0

CBH salto condizionato all'istruzione (parte alta della parola)

if AC >= 0 then PC <- Address; offsetPC <- 1

Operazioni aritmetiche (e I/O)

MUL prodotto

AC,MQ <- AC * Mem(Address)

DIV divisione e resto

AC <- AC / Mem(Address); MQ <- AC % Mem(Address)

LSHIFT shift a sinistra (ovvero prodotto per 2^X)

AC,MQ <- AC,MQ << X

RSHIFT shift a destra con estensione del segno (ovvero divisione per 2^X)

AC,MQ <- AC,MQ >> X

MOVE spostamenti da MQ ad AC con le (8) operazioni semplici (ADD, SUB, ABS, NEG …)

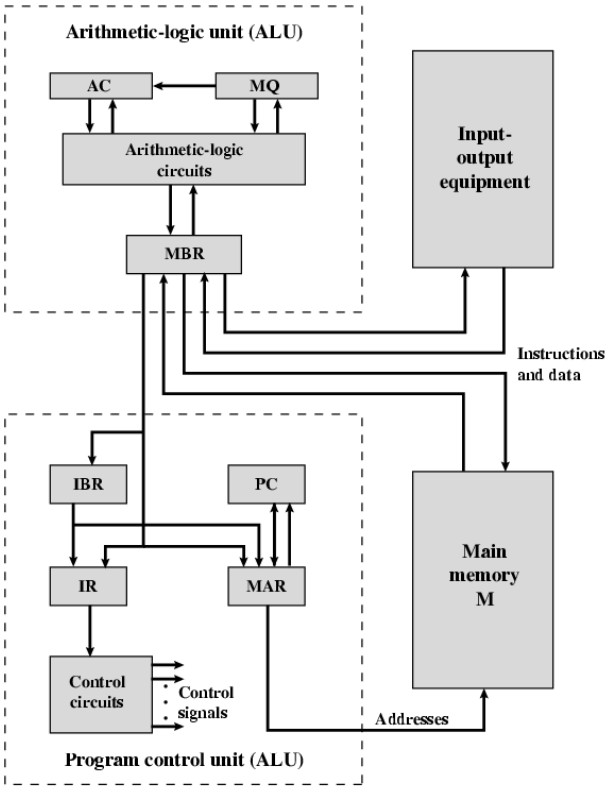
AC <- AC <operazione> MQ

IO trasferimento da e verso le periferiche

Esempio: C=A+B

Se A, B e C sono le locazioni 101, 102, 103 in memoria, il codice per calcolare C=A+B è

LD	101	AC <- Mem[101]
ADD	102	AC <-
S		] <- AC



Esecuzione delle istruzioni

Fetch della istruzione dalla memoria
MAR <- PC
IR, IBR <- MBR <- Mem[MAR]
Decodifica della istruzione (da IR)
MAR <- IR.Address; CU <- IR.Opcode
Sua **esecuzione**
AC <- MBR <- Mem[101]
Fetch della istr. successiva (da IBR)
MAR <- IBR.Address ; CU <- IBR.Op
sua **esecuzione**
AC <- AC + MBR <- Mem[102]
Aggiornamento del PC
PC <- PC + 1
Fetch della istruzione successiva
MAR <- PC
IR, IBR <- MBR <- Mem[MAR]
Decodifica della istruzione (da IR)
MAR <- IR.Address; CU <-