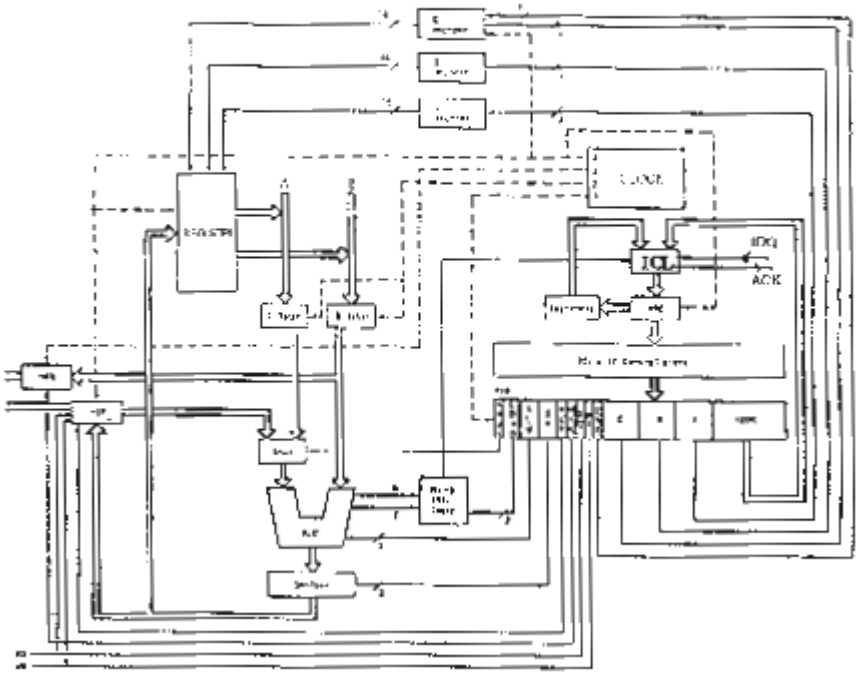
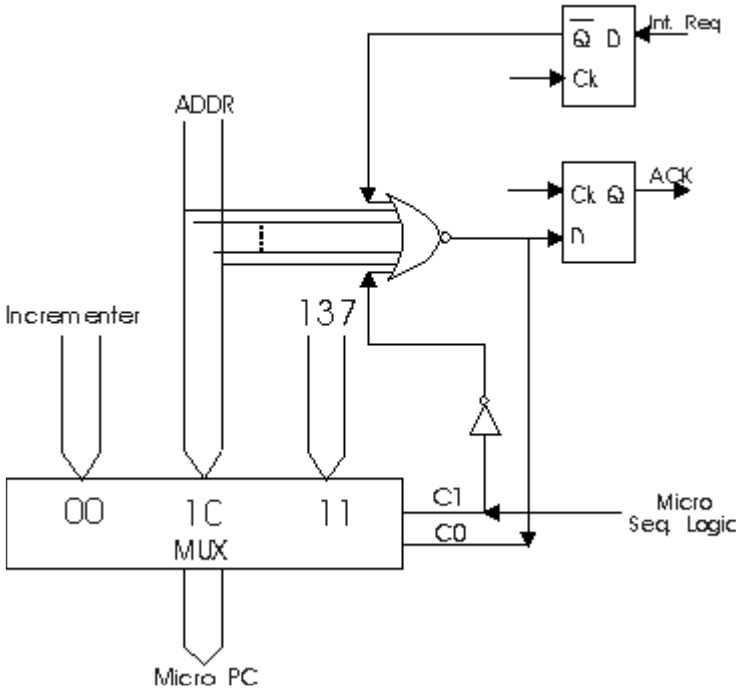
MAC1
Simulatore di Microprocessore

Architettura MAC 1



Schema I C L



INTRODUZIONE

Architettura del MAC-1

- Indirizzabilità della memoria a 16 bit (64 Kb)
- Indirizzabilità diretta a 12 bit (limitata ai primi 4 Kb)
- 16 registri a 16 bit
- Control Store a 256 x 32 bit
- Supporto Base Pointer
- Supporto Istruzioni di HALT, TRAP, RTI
- Gestione Interrupt da periferici collegati in Daisy-chain

Descrizione Registri

00 PC 04 TIR 08 Amask 12 C
01 AC 05 cost. 0 09 Smask 13 D
02 SP 06 cost. 1 10 A 14 E
03 IR 07 cost.-1 11 B 15 BP

Amask = 4095 (0000111111111111) per estrarre l'operando a 12 bit
Smask = 2048 (0000100000000000) per estrarre il segno dell'operando
D = 61440 (1111000000000000) per trasformare l'operando negativo
a 12 bit in uno negativo a 16 bit
E = 60 Che è l'indirizzo del Trap Handler

Set Istruzioni

```
00000 0000xxxxxxxxxxx LODD Load Direct
04096 0001xxxxxxxxxxx STOD Store direct
08192 0010xxxxxxxxxxx ADDD Add Direct
12288 0011xxxxxxxxxxx SUBD Sub Direct
16384 0100xxxxxxxxxxx JPOS Jump Positive
20480 0101xxxxxxxxxxx JZER Jump Zero
24576 0110xxxxxxxxxxx JUMP Jump
28672 0111xxxxxxxxxxx LOCO Load Constant
32768 1000xxxxxxxxxxx LODL Load Local
36864 1001xxxxxxxxxxx STOL Store Local
40960 1010xxxxxxxxxxx ADDL Add Local
45056 1011xxxxxxxxxxx SUBL Sub Local
49152 1100xxxxxxxxxxx JNEG Jump Negative
53248 1101xxxxxxxxxxx JNZE Jump Non Zero
57344 1110xxxxxxxxxxx CALL Call Procedure
      yyyyyyyyyyyyyyy
61440 1111000000000000 PSHI Push Indirect
61952 1111001000000000 POPI Pop Indirect
62464 1111010000000000 PUSH Push
62976 1111011000000000 POP Pop
63488 1111100000000000 RETN Return
64000 1111101000000000 SWSP Swap AC,SP
64512 1111110000000000 SWBP Swap AC,BP
65024 1111111000000000 HALT Halt
65280 1111111100000000 LJMP Long Jump
65344 1111111101000000 TRAP Call Kernel
65408 1111111110000000 RTI Return from Interrpt
65472 1111111111000000 LLCO Load Long Constant
      yyyyyyyyyyyyyyy
```

La TRAP salva sullo stack il PC e l' AC (Per Interrupt annidato).
La RTI ripristina l' AC e il PC.
La LLCO carica in AC una costante a 16 bit (per indirizzare 64 Kb).
La HALT esegue un salto alla linea 255 del microprogramma.

Organizzazione della Memoria

Le prime 1024 locazioni della RAM sono riservate al Kernel.
In questo blocco si trova anche un'area Flag di utilità generale e i buffer di lettura e scrittura.
Le locazioni da 1024 a 4095 sono riservate al Programma Utente.
Le locazioni da 4096 a 65531 sono destinate allo Stack.
I periferici sono gestiti con il sistema Memory Mapped
Tramite la loc. 65532 si scrive nel buffer della tastiera.
Tramite la loc. 65533 si scrive nel CSR della tastiera.
Tramite la loc. 65534 si scrive nel buffer del video.
Tramite la loc. 65535 si scrive nel CSR del video.
I bit 15, 2 e 1 dei CSR rappresentano rispettivamente:
- Ready
- Interrupt Enable
- Interrupt Pending

Control Store

La CS è stato implementato come un array di 256 x 13 interi, perchè 13 sono i campi del MIR.
La CS viene caricata e salvata come una sequenza di interi.

Periferici

Il periferico di lettura (Tastiera) è rappresentato dalla procedura [perif_TASTI()] che si avvale di una procedura ausiliaria [key_to_int(unsigned int)] che converte un carattere Ascii in un intero.
Il periferico di scrittura (Video) è rappresentato dalla procedura [perif_VIDEO()] che equivale alla scheda video di un Personal Computer.
Il Monitor di 5 righe x 8 colonne dotato di scrolling verso l'alto è rappresentato da un array di interi dichiarato e gestito nella procedura [Out_Video(unsigned int)].

KERNEL

Inizializzazione
- Setta CSR_Tastiera con Ready=1, IE=0, IP=0
- Setta CSR_Video con Ready=1, IE=0, IP=0
- Setta Buffer_Tastiera a 0
- Setta Buffer_Video a 0
- Setta flag "End Print" a 0
- Setta flag "End Read" a 0
- Setta Puntatore del Buff. Lett. a inizio Buff. Lett.
- Setta Puntatore del Buff. Scrit. a inizio Buff. Scrit.
- Setta Costante ENTER (10)
- Setta Costante BACKSPACE (11)
- Setta Costante 1
- Setta Costante 2

Trap Handler
- Se AC=0 chiama la procedura Read_Int (Lettura Intero)
- Se AC=1 chiama la procedura Print_Int (Scrittura Intero)

Area Flag
- Locazione 89 : End Print
- Locazione 90 : End Read
- Locazione 91 : Puntatore del Buffer di Lettura
- Locazione 92 : Puntatore del Buffer di Scrittura
- Locazione 93 : Codice ENTER
- Locazione 94 : Codice BACKSPACE
- Locazione 95 : Costante 1
- Locazione 96 : Costante 2
- Locazioni da 97 a 99 libere ed utilizzabili

Driver Tastiera
Controlla se il Buffer di lettura è pieno, in tal caso setta il flag di End Read a 1 ed esegue una RTI perchè la procedura microprogrammata Serve Interrupt salva PC e AC.
Inserisce il dato letto nel Buffer Video.
Controlla se il dato letto è il codice di ENTER, nel qual caso setta il flag di End Read a 1 ed esce.
Controlla se il dato letto è il codice di BACKSPACE, nel qual caso decrementa il puntatore del Buffer di lettura, setta il bit di IE del CSR della tastiera a 1 ed esce.
Se il dato è una cifra tra 0 e 9 la inserisce nel Buffer di lettura, incrementa il puntatore del Buffer di lettura, setta il bit di IE del CSR della tastiera a 1

Driver Video
Controlla se il Buffer di scrittura è vuoto, in tal caso setta il flag di End Print a 1 ed esce.
Setta il bit di IE del CSR del video a 1 e decrementa il puntatore del Buffer di scrittura.
Inserisce il carattere nel Buffer_Video ed esce (RTI).

Readint
Setta il bit di IE del CSR della tastiera a 1.

Continua a testare il flag di End Read finchè non lo trova a 1.
Setta il flag di End Read a 0.
Carica i caratteri dal Buffer di lettura sullo stack.
Calcola il numero di caratteri letti.
Trasforma le cifre singole in un intero, lo inserisce nel valore di ritorno ed esce (RETN).
Per calcolare il valore da restituire agisce nel modo seguente:

- Carica in AC il Top dello Stack (POP)
- Se non ci sono altri caratteri finisce
- Moltiplica AC per 10 (usando una var. ausiliaria)
- Somma AC con il Top dello Stack (POP)
- Mette AC sul Top dello Stack (PUSH)

Printint

Prende un intero come parametro d'ingresso.
Prepara 4 loc. di memoria ausiliarie con le costanti 10000, 1000, 100, 10 che serviranno come divisori.
Prepara 5 loc. di memoria ausiliarie azzerandole che serviranno per contenere le cifre da stampare.
Con dei cicli sottrae dall'intero i divisori preparati in precedenza in modo da ottenere le cifre da stampare.
Carica sullo stack le cifre ottenute che saranno 5 indipendentemente dall'intero. Ad esempio l'intero 123 darà origine alle cifre: 0 0 1 2 3.
Inserisce nel Buffer di scrittura il codice di ENTER.
Inserisce nel Buffer di scrittura i primi 4 caratteri presenti sullo stack.
Resta in attesa che il bit di Ready del video vada a 1.
Setta il bit di IE del CSR del Video a 1.
Inserisce nel Buffer_Video il carattere ancora sullo stack; tale azione metterà in moto il meccanismo di stampa che terminerà quando il buffer di scrittura sarà vuoto.
Resta in attesa che il flag di End Print vada a 1, lo setta a 0 ed esce (RETN).

Libreria Funzioni

A partire dalla locazione 850 inizia un'area adibita a contenere procedure di utilità generale come le funzioni matematiche di Moltiplicazione e Divisione.
Tali funzioni potrebbero formare una libreria eventualmente linkabile con un programma utente

FASI DI CLOCK

Fase 1
- Caricamento del MIR in base al microPC
- Settaggio Bus A e B con i valori dei registri selezionatidai Decoder A e B

Fase 2
- Memorizzazione dei Bus A e B nei Latch A e B
- Esecuzione Operazione richiesta da parte dell' ALU

Fase 3
- Esecuzione dell'operazione di SHIFT
- Se richiesto si setta l'Addres Bus con il valore del B_latch

Fase 4
- Se ENC=1 il registro selezionato dal Decoder_C viene caricato con il contenuto del Bus C.
- Gestione dell'incremento del microPC
- Gestione del Memory Buffer Register

MECCANISMO DI INTERRUZIONE

Serve interrupt
- Salva sullo stack il Program Counter
- Salva sullo stack l' Accumulatore
- Controlla il CSR del video
- Se il Video ha generato un interrupt resetta il bit di IP del CSR del Video e setta PC=2 (In 2 si trova un salto al Driver del Video)
- Se la Tastiera ha generato un interrupt resetta il bit di IP del CSR della Tastiera e setta PC=1 (In 1 si trova un salto al Driver della Tastiera)

ICL
La procedura ICL(int) viene richiamata dalla procedura MPC() nella quarta fase di clock.
Calcola il valore di ACK (1 se IRQ=1 e si sta eseguendo un salto alla linea 0 cioè una fase di FETCH).
Se ACK=1 pone IRQ=0 e inserisce in microPC l'indirizzo della routine di gestione dell'Interrupt (137).
Se ACK=0 e si sta eseguendo un salto, microPC=MIR[12].
Se ACK=0 e non si sta eseguendo un salto, microPC++;

Sequenza d'Interruzione
- Il periferico setta il bit di IP a 1
- Il periferico asserisce il segnale di IRQ
- La CPU nella prima fase di FETCH, successiva all'Interrupt, resetta IRQ e asserisce ACK.
- Il periferico quando riceve l'ACK resetta il bit di IE
- La CPU eseguendo la Serve Interrupt resetta il bit di IP
- Il Driver del periferico gestisce l'interruzione e setta il bit di IE a 1.