Compito di Architettura dei Calcolatori - A.A. 2010-11
Prova di esame del 11 luglio 2011

COGNOME: NOME: MATRICOLA:

Istruzioni: Spiegare chiaramente TUTTE le assunzioni che vengono effettuate per chiarire eventuali punti che si ritengono ambigui o non specificati.

1) [10 punti] Illustrare i principi generali ed il flusso di controllo per la gestione di una "pipeline" per ottimizzare le prestazioni della CPU, discutendo sia il caso di salto incondizionato che di salto condizionato.

SVOLGIMENTO:

Si veda il cap.12 del libro di testo e dei lucidi presentati a lezione, in particolare i lucidi da 38 a 48 (compresi).

2) [10 punti] Scrivere un programma nel linguaggio Assembly di Virtual CPU che realizza il seguente comportamento:

Si legge una locazione di memoria all'indirizzo A. Se in A si trova scritto 1 allora si esegue il blocco sotto specificato altrimenti si ritorna controllare la locazione di memoria all'indirizzo A.

Il blocco da eseguire consiste nel controllare la somma dei valori scritti nelle locazioni di memoria contigue dall'indirizzo M1 all'indirizzo M2 compresi. Se tale somma è maggiore di MAX allora si scrive nella locazione di memoria OUT il valore -1 e ci si ferma. Se tale somma è minore o uguale a MAX e maggiore di MIN allora si scrive nella locazione di memoria OUT il valore della somma e si riesegue la somma ed il controllo sul suo valore. Se tale somma è minore o uguale a MIN allora si torna al test principale su A. Si assume che i numeri in queste celle di memoria tra M1 e M2 sono rappresentati *in complemento a 2* e che la somma non causa mai problemi di overflow.

I valori A, M1, M2, MAX, MIN, OUT sono contenuti nelle celle di memoria di indirizzo rispettivamente da 1 a 6.

Commentare con adeguato dettaglio la logica seguita e le istruzioni usate

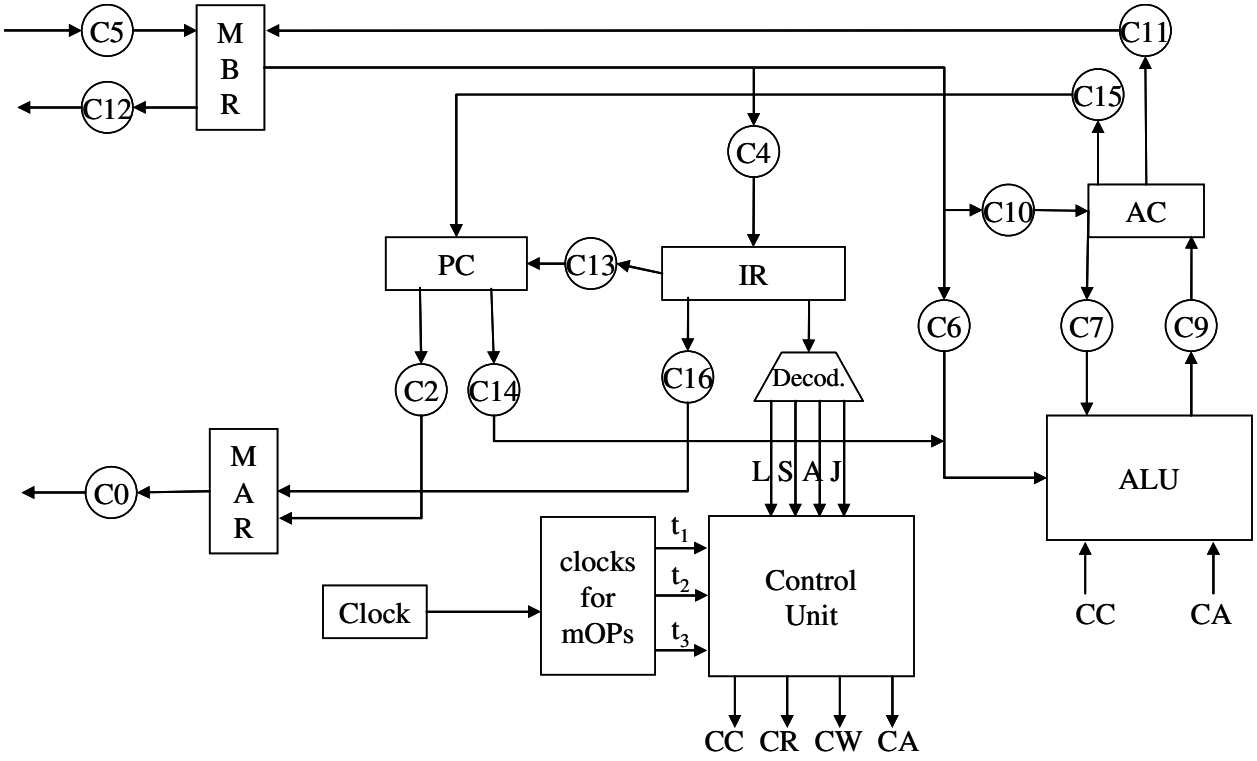
SVOLGIMENTO:

```
0   JMP 7        ; salta alla cella d'inizio del programma
; I DATI del programma, memorizzati in celle che sono usate come variabili di ingresso al programma
1                ; contiene A, cioè l'indirizzo della locazione di memoria da verificare
2                ; contiene M1, cioè l'indirizzo della prima cella della serie di valori da sommare
3                ; contiene M2, cioè l'indirizzo dell'ultima cella della serie di valori da sommare
4                ; contiene MAX, cioè il massimo valore che può assumere la somma
5                ; contiene MIN, cioè il minimo valore che può assumere la somma
6                ; contiene OUT, cioè l'indirizzo della locazione di memoria in cui scrivere il risultato
; IL PROGRAMMA
; test iniziale
7   LOAD @@1     , si carica nell'accumulatore il valore della cella all'indirizzo A
8   DEC AX       ; gli si sottrae 1
9   JNZ 7        ; se all'indirizzo A c'è un valore diverso da 1 si riesegue il test
; inizializzazione del contatore CX per il calcolo della somma
10  LOAD @3      ; si carica nell'accumulatore il valore dell'indirizzo dell'ultima cella da sommare
11  SUB @2       ; gli si sottrae il valore dell'indirizzo della prima cella da sommare
12  INC AX       ; gli si aggiunge 1
13  STORE CX     ; adesso in CX c'è il numero di celle che devono essere sommate, si assume almeno 1
; inizializzazione del risultato parziale della somma memorizzato in BX
14  LOAD 0       ; si carica 0 nell'accumulatore
15  STORE BX     ; si inizializza a 0 il risultato parziale in BX
; inizializzazione del puntatore DX alla cella corrente da sommare
16  LOAD @2      ; si carica nell'accumulatore l'indirizzo della prima cella da sommare
17  STORE DX     , si inizializza DX con l'indirizzo della prima cella da sommare
; ciclo per il calcolo della somma
18  LOAD @DX     ; si carica nell'accumulatore il valore della prima cella da sommare
19  ADD BX       ; gli si aggiunge il precedente risultato parziale
20  STORE BX     ; si salva il nuovo risultato parziale
21  INC DX       ; si aggiorna il puntatore per la prossima iterazione
22  DEC CX       ; si decrementa il contatore
23  JG 18        ; se il contatore è ancora positivo si itera il ciclo di calcolo della somma
; qui sono stati sommati tutti i valori e si fanno i vari test
24  LOAD BX      ; si carica nell'accumulatore il valore complessivo della somma
25  SUB @4       ; gli si sottrae il massimo
```

```
26   JG 34        ; se è superiore al massimo si gestisce il relativo caso
27   ADD @4       ; altrimenti si ripristina il valore complessivo della somma
28   DEC AX       ; lo si decrementa di 1 (vedere il NB)
29   SUB @5       ; gli si sottrae il minimo
30   JL 7         ; se è inferiore al minimo si ricomincia dal test sulla locazione di indirizzo A
31   ADD @5       ; altrimenti (MIN<somma<=MAX)si ripristina il valore complessivo della somma
32   STORE @@6    ; e lo si scrive nella cella di indirizzo OUT
33   JMP 10       ; e si effettua un nuovo calcolo della somma
; qui si gestisce il caso di somma > MAX
34   LOAD -1      ; si carica -1 nell'accumulatore
35   STORE @@6    ; e lo si scrive nella cella di indirizzo OUT
36   HLT          ; e si termina
```

NB: Nel simulatore ENIAC l'implementazione di JLE non è corretta. Per questo motivo nella soluzione su esposta non è stato usato JLE (che nel simulatore ENIAC si comporta erroneamente come JL) ma si è ulteriormente decrementato di 1 il valore nell'accumulatore (istruzione #28) e si è poi usato JL. Ovviamente in sede di esame si poteva benissimo usare JLE (senza l'ulteriore decremento!).

3) [10 punti] Dato lo schema della semplicissima CPU (VS0) sotto disegnato, fornire e spiegare le micro-istruzioni della micro-procedura usata da un'unità di controllo micro-programmata per l'effettuazione della fase di prelevamento della prossima istruzione (fetch), nell'ipotesi che la struttura della parola di controllo (Control Word) contenga UN SOLO flag di salto e contenga due campi con l'indirizzo della successiva micro-istruzione. Inoltre, descrivere e spiegare la circuiteria presente nell'unità di controllo..



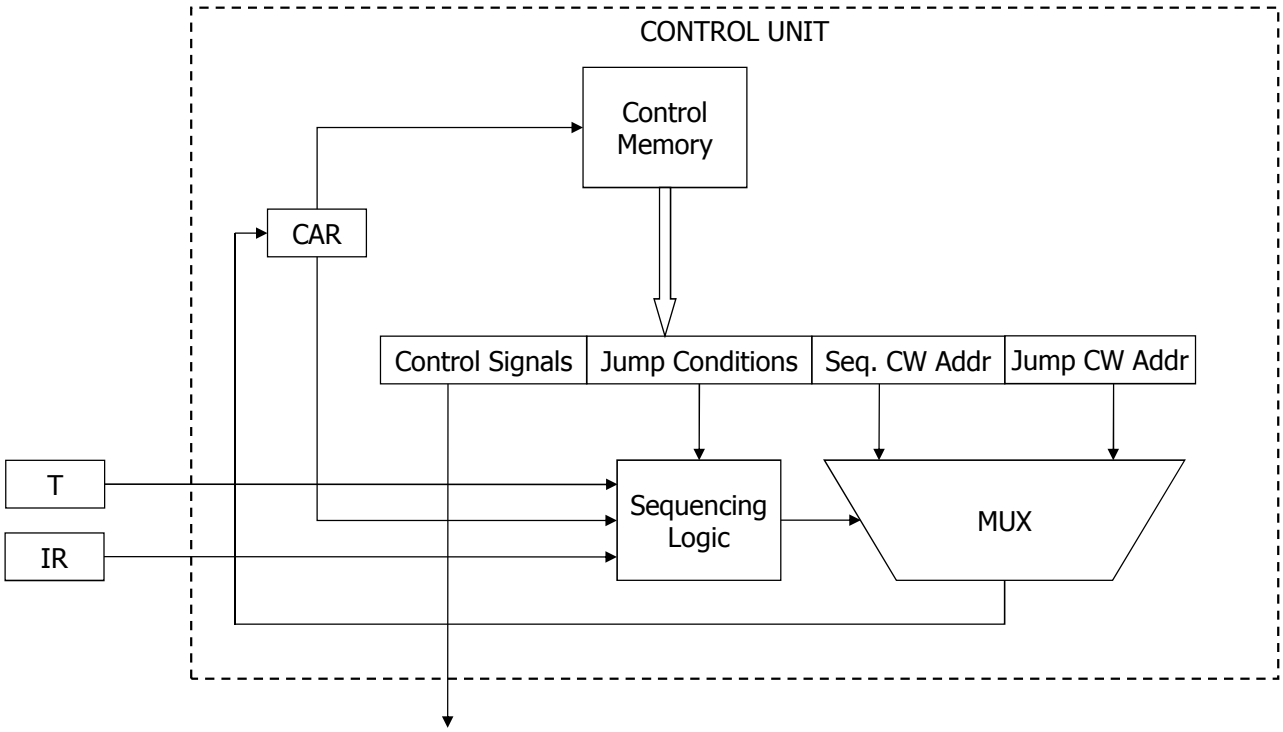
SVOLGIMENTO:

Seguendo il cap.17 del libro di testo e dei lucidi presentati a lezione, in particolare i lucidi 6 e 17 (ma si vedano anche i lucidi 11, 12, 13 e 15) si delinea nel seguito un possibile svolgimento (da completare come spiegato).

Nello schema dato per la CPU le micro-istruzioni della micro-procedura per l'effettuazione della fase di prelevamento della prossima istruzione sono le seguenti:

t1: MAR ← PC
t2: MBR ← memory; ALU ← PC; increment ALU; AC ← ALU
t3 PC ← AC; IR ← MBR

La circuiteria presente nell'unità di controllo è illustrata dalla figura seguente:



A questo punto la figura sopra presentata deve essere spiegata illustrando il funzionamento delle varie componenti e la loro interazione. Poi si prosegue:

Poiché la parte "Jump Conditions" della Control Word contiene un solo flag di salto la traduzione delle micro-istruzioni in termini di control words della Control Memory è descritta nel seguito (CWi è l'indirizzo della Control Memory a cui è memorizzata una control word):

Indir.	**Micro-operazioni**						**Jump Condit.**	**Seq.CW**	**Jump.CW**
CW1	C2						*False*	CW2	
CW2	C0	C5	C9	C14	CA	CR	*False*	CW3	
CW3	C4	C9	C15				*True*	CW4	CW6

Si è assunto che dopo la fase di prelevamento della prossima istruzione si effettui o un accesso diretto a memoria o un accesso indiretto, a seconda del valore del bit, indicato simbolicamente con D, che nell'istruzione appena prelevata (e presente in IR) segnala la necessità (D=1) di effettuare un accesso indiretto a memoria. Si è inoltre assunto che la micro-procedura per l'accesso diretto a memoria inizi all'indirizzo CW4 e che la micro-procedura per l'accesso indiretto a memoria inizi all'indirizzo CW6

La parte della formula logica implementata nella Sequencing Logic che per queste istruzioni attiva la selezione tramite il multiplexer di Seq.CW (con il segnale 0) oppure di Jump.CW (con il segnale 1) è pertanto, indicando con K l'unico bit presente nella parte "Jump Conditions" della Control Word:

KD (cioè l'AND tra il bit K e il bit D)