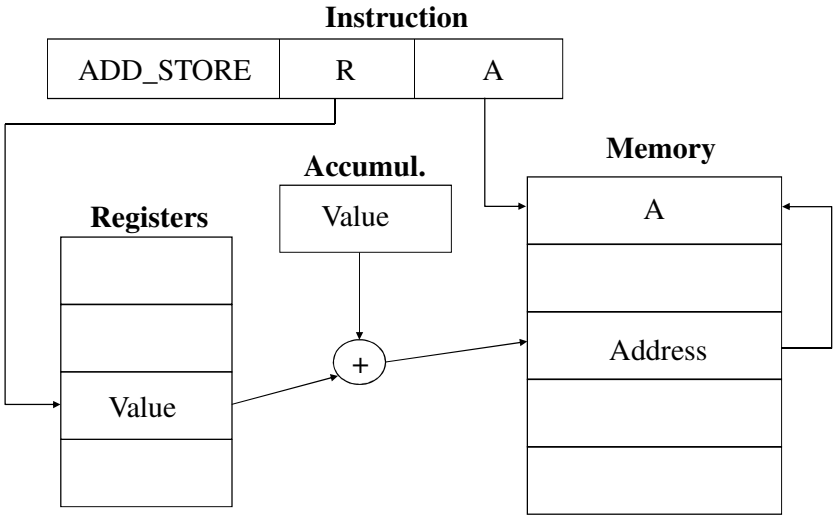
Compito di Architettura dei Calcolatori - A.A. 2010-11
Prova di esame del 21 febbraio 2012

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

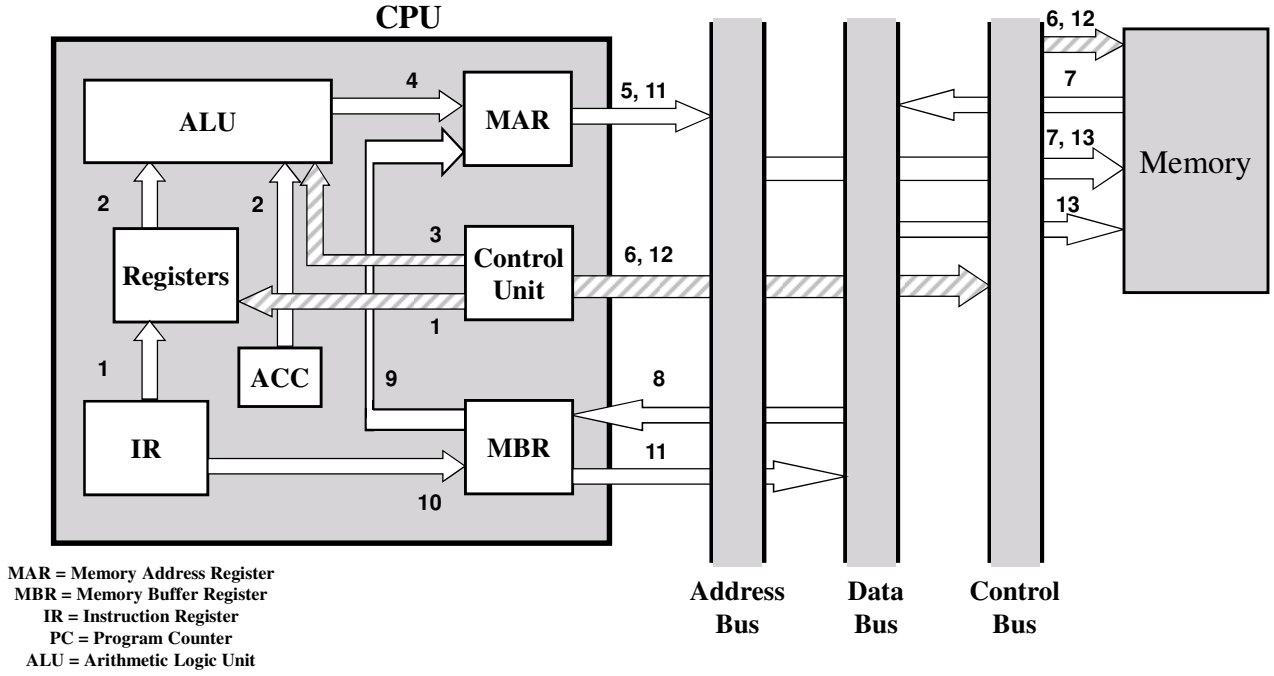
1) [10 punti] 1) [10 punti] Si assuma di avere un'istruzione ADD_STORE R A che effettua la seguente operazione: somma il valore contenuto nell'accumulatore ACC e il valore contenuto nel registro R ed il risultato così ottenuto viene successivamente usato per memorizzare A accedendo alla memoria mediante un indirizzamento indiretto. Prima disegnare il relativo diagramma di costruzione dell'indirizzo di memorizzazione. Poi disegnare e spiegare il diagramma con i flussi dei dati tra le componenti della CPU relativamente a tutte le fasi di esecuzione di tale istruzione (assumere che essa sia interamente contenuta nel Registro Istruzioni), indicando la sequenza temporale con cui i vari flussi avvengono.

SVOLGIMENTO:

Il diagramma di costruzione dell'indirizzo è il seguente (l'accumulatore è stato evidenziato a parte anche se in realtà non è altro che uno dei registri):



Il diagramma di flusso dei dati è il seguente



dove il significato dei flussi è il seguente:

1. La Control Unit richiede la lettura del registro selezionato in base al valore R presente in IR (flusso di *controllo* e flusso dati)
2. Il contenuto del registro selezionato viene presentato alla ALU insieme al contenuto dell'Accumulatore
3. La Control Unit richiede alla ALU di effettuare la somma (flusso di *controllo*)
4. Il risultato della ALU viene copiato in MAR
5. Il contenuto di MAR viene messo sull'Address Bus
6. La Control Unit richiede alla memoria un'operazione di lettura (flusso di *controllo*)
7. La Memoria legge l'Address Bus e mette il risultato sul Data Bus
8. Il risultato (cioè l'indirizzo dell'operando) viene copiato in MBR
9. Il contenuto di MBR viene copiato in MAR
10. Il valore A presente in IR viene copiato in MBR
11. Il contenuto di MAR viene messo sull'Address Bus e il contenuto di MBR viene messo sul Data Bus
12. La Control Unit richiede alla memoria un'operazione di scrittura (flusso di *controllo*)
13. Il contenuto di MBR viene copiato nella memoria

2) [10 punti] Siano dati due interi non negativi n_A e n_B rappresentati nei 16 bit più a destra delle celle di memoria di indirizzo A1, A2 per n_A e B1, B2 per n_B. Scrivere un programma nel linguaggio Assembly di Virtual CPU per effettuare $n_C = n_A + n_B$ e scriverlo nei 16 bit più a destra delle celle di memoria di indirizzo C1 e C2. Per tutti e 3 gli interi la cella con indice 1 rappresenta i 16 bit meno significativi dell'intero. Ricordarsi che in Virtual CPU le celle di memoria hanno 24 bit.

I valori A1, A2, B1, B2, C1, C2 sono contenuti nelle celle di memoria di indirizzo rispettivamente da 1 a 6. Commentare il programma scritto spiegandone con adeguato dettaglio la logica seguita, le scelte effettuate e le istruzioni usate

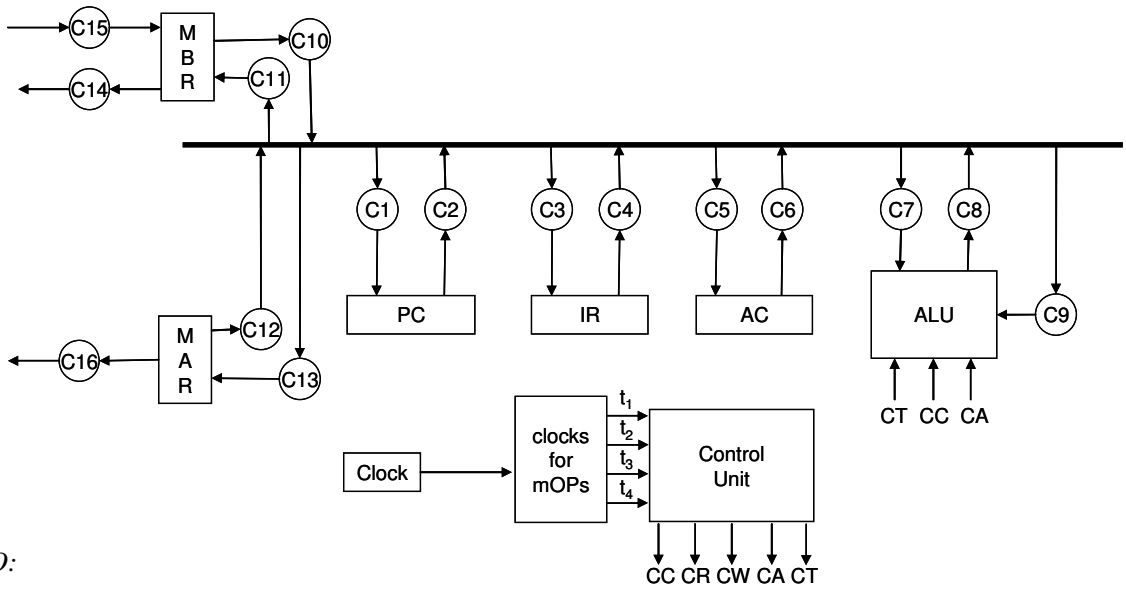
SVOLGIMENTO:

Si riporta lo svolgimento con riferimento all'emulatore di vCPU (ENIAC). Si assume per semplicità di trattamento che nei 16 bit meno significativi delle celle di memoria gli interi non negativi sono rappresentati con una codifica **non complementata**. Allora non ci sono problemi di overflow della rappresentazione e l'unica difficoltà è gestire correttamente l'eventuale riporto derivante dalla somma delle due parti meno significative dei valori. In generale, infatti, la somma di due numeri a 16 bit potrebbe utilizzare 17 bit per la sua rappresentazione. Ma poiché l'esercizio richiede di usare solo i 16 bit più destra, tale caso va opportunamente gestito. In ogni caso, ricordiamo che nella somma di due cifre con l'eventuale riporto proveniente dalla somma delle due cifre immediatamente meno significative ***o*** viene generato un riporto dalla sola somma delle due cifre ***o*** viene generato un riporto dalla somma dell'eventuale riporto precedente con la somma delle due cifre date, ma ***non*** possono accadere entrambe le cose. Esempio del primo caso: 9+9 col riporto di 0 dà 8 col riporto di 1, ma 9+9 col riporto di 1 dà 9 sempre col riporto di 1. Esempi del secondo caso: 5+4 col riporto di 0 dà 9 col riporto di 0, mentre 5+4 col riporto di 1 dà 0 col riporto di 1; 9+0 col riporto di 0 dà 9 col riporto di 0, mentre 9+0 col riporto di 1 dà 0 col riporto di 1.

```
0    JMP 8         ; 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 l'indirizzo della cella che contiene A1 (la parte MENO significativa di A)
2                  ; contiene l'indirizzo della cella che contiene A2 (la parte PIU' significativa di A)
3                  ; contiene l'indirizzo della cella che contiene B1 (la parte MENO significativa di B)
4                  ; contiene l'indirizzo della cella che contiene B2 (la parte PIU' significativa di B)
5                  ; contiene l'indirizzo della cella che contiene C1 (la parte MENO significativa di C)
6                  ; contiene l'indirizzo della cella che contiene C2 (la parte PIU' significativa di C)
;                      a meno dell'eventuale bit più significativo derivante da un riporto
7                  ; contiene l'indirizzo della cella che contiene l'eventuale bit più significativo di C
; IL PROGRAMMA
; ci si assicura che le celle usate per gestire gli eventuali riporti siano inizializzate a 0
8    LOAD 0        ; si carica il valore 0 nell'accumulatore
9    STORE @@7     ; lo si scrive nella cella il cui indirizzo è nella cella 7
10   STORE BX      ; lo si scrive nel registro BX
; si predispone il meccanismo per il test sul riporto
11   LOAD 2048     ; 2048=2^11 è la max potenza di 2 rappresentabile nei 13 bit dell'operando come positivo
12   MUL 32        ; moltiplicando per 32=2^5 si ottiene 2^16 che ha solo il 17-mo bit meno significativo a 1
13   STORE DX      ; DX verrà usato per il test sulla presenza del riporto
; si sommano le parti MENO significative di A e B, cioè A1 e B1
14   LOAD @@1      ; si carica A1 nell'accumulatore
15   ADD @@3       ; si addiziona all'accumulatore B1
; adesso si verifica se il 17-mo bit meno significativo dell'accumulatore è 1 (=riporto) oppure 0
16   STORE CX      ; si salva A1+B1 (che può usare 17 bit) perché il test è distruttivo
17   AND DX        ; DX ha solo il 17-bit meno significativo (cioè il 17-mo contando da dx) pari a 1
```

```
18   JZ 20        ; se il 17-bit meno significativo dell'accumulatore era 0 l'AND dà 0 e si prosegue
19   INC BX       ; altrimenti si scrive il riporto di A1+B1 in BX (che era inizializzato a 0)
20   LOAD CX      ; si ripristina A1+B1 nell'accumulatore
21   DEC DX       ; ora DX=65535=2^16-1 ha tutti e soli i 16 bit meno significativi pari a 1
22   AND DX       ; si mantengono nell'accumulatore i 16 bit meno significativi, ponendo gli altri a 0
23   STORE @@5    ; si scrive A1+B1 (che adesso usa solo 16 bit) nella cella che contiene C1
; si sommano le parti PIU' significative di A e B, cioè A2 e B2
24   LOAD @@2     ; si carica A2 nell'accumulatore
25   ADD @@4      ; si addiziona all'accumulatore B2
; adesso si verifica se il 17-mo bit meno significativo dell'accumulatore è 1 (=riporto) oppure 0
26   STORE CX     ; si salva A2+B2 (che può usare 17 bit) perché il test è distruttivo
27   INC DX       ; si riporta DX a 65536=2^16
28   AND DX       ; DX ha solo il 17-bit meno significativo (cioè il 17-mo contando da dx) pari a 1
29   JZ 31        ; se il 17-bit meno significativo dell'accumulatore era 0 l'AND dà 0 e si prosegue
30   INC @@7      ; altrimenti si scrive il riporto di A2+B2 nella cella del bit più significativo di C
31   LOAD CX      ; si ripristina A2+B2 nell'accumulatore
32   DEC DX       ; ora DX=65535=2^16-1 ha tutti e soli i 16 bit meno significativi pari a 1
33   AND DX       ; si mantengono nell'accumulatore i 16 bit meno significativi, ponendo gli altri a 0
34   STORE @@6    ; si scrive A2+B2 (che adesso usa solo 16 bit) nella cella che contiene C2
; adesso si somma l'eventuale riporto di A1+B1 a A2+B2
35   LOAD BX      ; BX contiene 1 se c'era riporto altrimenti è rimasto a 0
36   ADD 0        ; predispone i valori dei flag lasciando inalterato l'accumulatore
37   JZ 39        ; se non c'era riporto si salta alla fine
38   INC @@6      ; altrimenti si incrementa A2+B2 (e sicuramente non si crea un riporto)
39   HLT
```

3) [10 punti] Dato lo schema della semplicissima CPU (VS0) sotto disegnato nella versione a singolo bus disegnare un nuovo schema (se necessario ed utilizzando sempre una struttura interna della CPU basata su singolo bus) per poter eseguire le istruzioni LOAD AC R e LOAD AC (R). Esse caricano l'accumulatore AC, rispettivamente, con il contenuto di R o con il contenuto della cella di memoria il cui indirizzo è contenuto in R, dove R è uno tra 4 possibili registri. Descrivere inoltre, con riferimento a tale nuovo schema, il flusso dei dati tra le varie componenti della CPU per l'esecuzione completa di tali due istruzioni evidenziandone il raggruppamento in micro-operazioni.

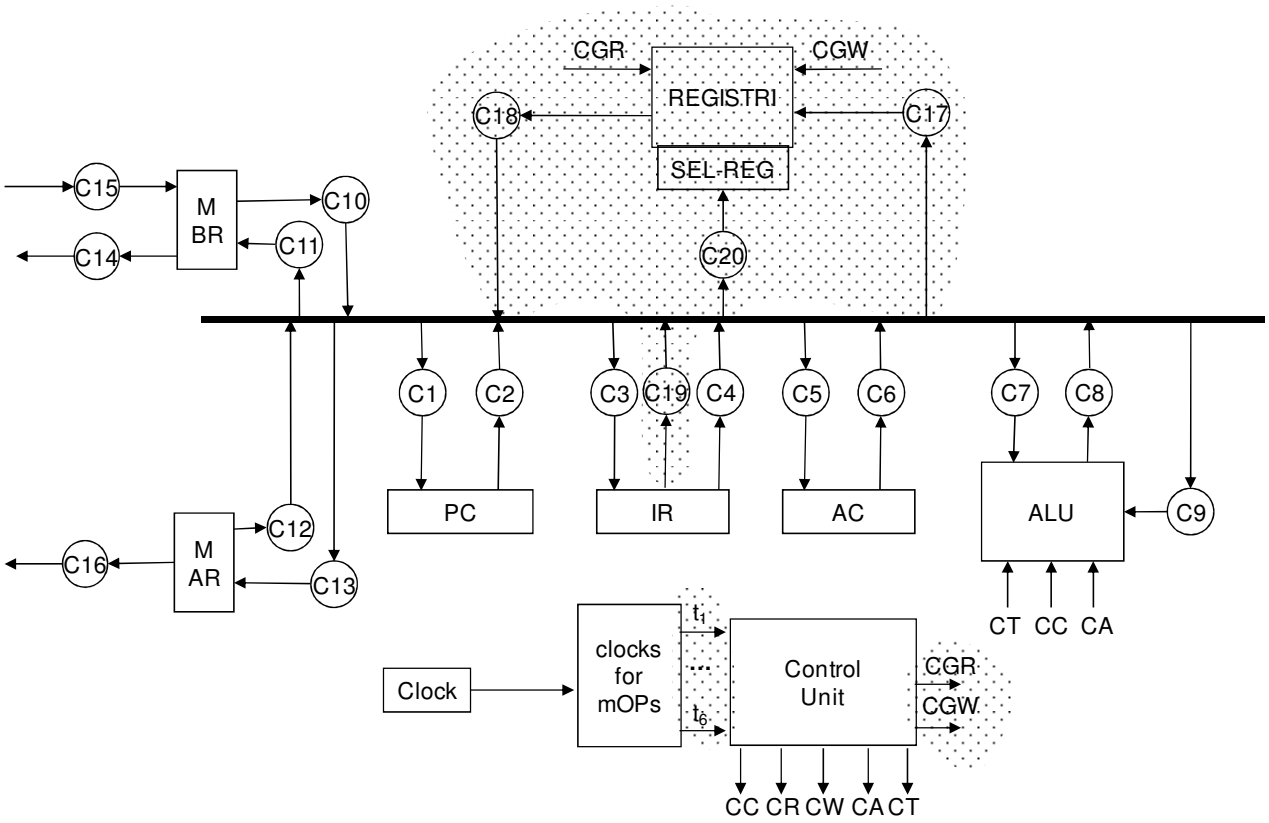


SVOLGIMENTO:

Il nuovo schema della CPU, con evidenziate le parti aggiunte, è disegnato più sotto.

Sono stati aggiunti allo schema 4 registri che vengono acceduti come se fossero una memoria. Serve quindi un registro di selezione (SEL-REG nel nuovo schema sotto riportato) cui si fa arrivare il valore del registro desiderato tra i quattro, come se fosse il Memory Address Register per l'accesso alla RAM. Il valore del registro desiderato è un numero a 2 bit (proveniente da IR) che viene fatto arrivare a SEL-REG dal bus mediante il nuovo segnale di controllo C20. Serve inoltre un nuovo segnale di controllo per attivare la lettura (CGR) dal registro che viene selezionato mediante il valore in SEL-REG.

Si noti che nello schema di VS0 presentato a lezione IR è un registro a 8 bit di cui 6 (da b_5 a b_0) sono dedicati alla parte indirizzi e 2 (b_7 e b_6) sono dedicati al codice operativo. Nel nuovo schema il nuovo segnale di controllo C19 mette sul bus il valore dei 2 bit b_1b_0 di IR corrispondenti al registro usato per la nuova istruzione. (Il segnale C4 rimane inalterato e continua ad essere usato per mettere sul bus il valore dei 6 bit della parte indirizzo di IR).



Con le modifiche precedentemente descritte, riportate nel nuovo schema qua sopra, le micro-operazioni sono le seguenti (si noti quindi che il registro R specificato nelle istruzioni è un parametro generico che di volta in volta indicano quale dei quattro registri viene selezionato).

Si faccia attenzione che l'esecuzione di questa nuova istruzione richiede due tempi di clock in più che sono stati aggiunti allo schema (cioè il circuito di generazione dei segnali di clock per la Control Unit adesso genera 6 segnali invece di 4).

LOAD AC *R*

t1	SEL-REG ← IR_R	C19 C20
t2	AC ← REGISTRI	CGR C18 C5

LOAD AC (*R*)

t1	SEL-REG ← IR_R	C19 C20
t2	MAR ← REGISTRI	CGR C18 C13
t5	MBR ← memory	C16 CR C15
t6	AC ← MBR	C10 C5

Si noti che nello schema sono stati aggiunti anche i nuovi segnale di controllo C17 e CGW, il primo che mette il bus in comunicazione con il banco dei 4 nuovi registri e il secondo che abilita la scrittura sul banco stesso, segnali che sono entrambi indispensabili per caricare valori nei 4 nuovi registri, anche se non vengono usati per l'esecuzione di questa istruzione.