Calcolatori elettronici: esempio di esame

M. Rebaudengo - M. Sonza Reorda

Politecnico di Torino
Dip. di Automatica e Informatica

Tipologia d’esame

Parte A: 10 domande a risposta chiusa

Parte B: 4 domande a risposta aperta

Parte C: 1 esercizio di programmazione in assembly

Parte A
Domande a risposta chiusa

E’ necessario rispondere correttamente ad almeno 6 domande.

Non è possibile consultare alcun tipo di materiale.

Tempo: 15 minuti.

Valutazione: passa / non passa.

Domanda 1

Si consideri il seguente frammento di codice:

MOV AH, 1

INT 21H

Se l'utente preme il tasto 0 sulla tastiera, quale valore viene scritto in AL?

A	**0**
B	**2**
C	**0H**
D	**30H**

Domanda 2

Quale delle seguenti istruzioni in linguaggio assembler 8086 è corretta?

A	**MOV DX, [AX]**
B	**MOV AX, [BX+2]**
C	**MOV [BX], CX+2**
D	**MOV [BX], [SI]**

Domanda 3

Si consideri l'istruzione assembler 8086

MUL AL

Si assuma che all'atto dell'esecuzione i registri del processore abbiamo i seguenti valori decimali: AL: 10, AH, 1, BL: 5, BH: 3, CL: 5, CH: 8, DL: 0, DH: 1; qual è il valore di AX dopo l'esecuzione dell'istruzione?

A	**10**
B	**50**
C	**100**
D	**2560**

Domanda 4

Si consideri il seguente frammento di codice assembler 8086

LEA SI, VETT

MOV AX, [SI+4]

Si assuma che VETT sia un vettore di 10 word che prima dell'esecuzione del frammento contiene i valori 100, 101, 102, 103, 104, 106, 107, 108, 109, 110. Quale valore è presente in AX dopo l'esecuzione dell'istruzione MOV?

A	**0**
B	**102**
C	**103**
D	**104**

Domanda 5

Si desidera dividere il valore contenuto in CX per il valore contenuto in DX. Ambedue i valori corrispondono a interi con segno. Quale dei frammenti di codice a lato esegue correttamente l'operazione?

A	**MOV BX, DX** **MOV AX, CX** **CWD** **DIV BX**
B	**MOV AX, CX** **IDIV DX**
C	**MOV BX, DX** **MOV AX, CX** **CWD** **IDIV BX**
D	**MOV BX, DX** **MOV AX, CX** **MOV DX, 0** **IDIV BX**

Domanda 6

Si consideri un sistema 8086 e si assuma che ad un certo istante i registri di segmento abbiano il seguente contenuto: DS=5A00, SS=AA00, CS=AB80, ES=5530. Qual è l'indirizzo fisico della cella di memoria che viene azzerata dall'istruzione MOV [BX], 0 assumendo che BX=0100?

Tutti i valori riportati sono esadecimali.

A	**5A100**
B	**AA100**
C	**AB900**
D	**55400**

Domanda 7

Si assuma che AX contenga il seguente valore

0010 1011 1001 1101

Quale valore sarà presente in AX dopo l'esecuzione dell'istruzione CBW?

Domanda 8

Si desidera eseguire la somma di due variabili V1 e V2 che contengono due numeri interi con segno su 16 bit, scrivendo il risultato nella variabile V3, pure su 16 bit. Nel caso il risultato non sia rappresentabile, si desidera saltare all'etichetta ERRORE. Quale dei seguenti frammenti di codice esegue correttamente quanto richiesto?

A	ADD V1, V2
	JO ERRORE
	MOV V3, V1

B	MOV AX, V1
	ADD AX, V2
	JO ERRORE
	MOV V3, AX

C	MOV AX, V1
	ADC AX, V2
	JC ERRORE
	MOV V3, AX

D	MOV AX, V1
	ADD AX, V2
	JC ERRORE
	MOV V3, AX

Domanda 9

A fianco è riportato il contenuto di alcune celle di memoria facenti parte di un sistema x86. Quale sarà il contenuto di AX dopo l'esecuzione dell'istruzione POP AX, assumendo che SS=15A0 e SP=A? Tutti i valori sono esadecimali.

Indirizzo	
15A00	2
15A02	4
15A04	6
15A06	8
15A08	A
15A0A	C
15A0C	E

A	2
B	A
C	C
D	E

Parte B
Domande a risposta aperta

Sino a 5 punti per ogni domanda.

Non è possibile consultare alcun tipo di materiale.

Tempo: 40 minuti.

Valutazione: massimo 20 punti (4 domande)

Domanda

- Si consideri un sistema composto da
  - CPU 8086
  - memoria
  - Interrupt Controller 8259
  - 6 interfacce di periferiche, connesse alle line IR0÷IR5 dell'8259.
- Le ISR delle 6 periferiche sono piazzate in memoria ai seguenti indirizzi esadecimali
  - ISR0: 10A00
  - ISR1: 10A10
  - ISR2: 10A50
  - ISR3: 10AA0
  - ISR4: 10BA0
  - ISR5: 10BB0.
- I codici esadecimali associati alle 6 periferiche sono 70, 71, 72, 73, 74, 75, rispettivamente.

Domanda (cont.)

1. Si determini il valore da scrivere nella ICW2 dell'8259
2. Si determinino i valori da scrivere nella IVT all'atto dell'inizializzazione delle 6 periferiche
3. Si fornisca l'indirizzo dell'elemento della IVT in cui è contenuto l'indirizzo della ISR0.

ICW2

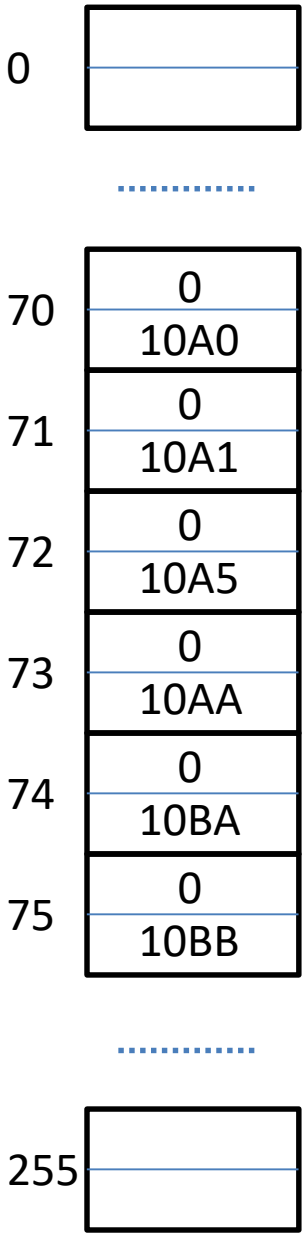
- Il byte da scrivere nella ICW2 è

D7	D6	D5	D4	D3	D2	D1	D0
0	1	1	1	0	X	X	X

IVT

- Gli elementi della IVT relativi alle 6 periferiche sono quelli con indice esadecimale da 70 a 75
- Ognuno di essi deve contenere i valori di CS e IP corrispondenti all'indirizzo della prima istruzione della corrispondente ISR
- Tali valori corrispondono ad esempio a:
  - Periferica 0: CS = 10A0, IP = 0
  - Periferica 1: CS = 10A1, IP = 0
  - Periferica 2: CS = 10A5, IP = 0
  - Periferica 3: CS = 10AA, IP = 0
  - Periferica 4: CS = 10BA, IP = 0
  - Periferica 5: CS = 10BB, IP = 0.

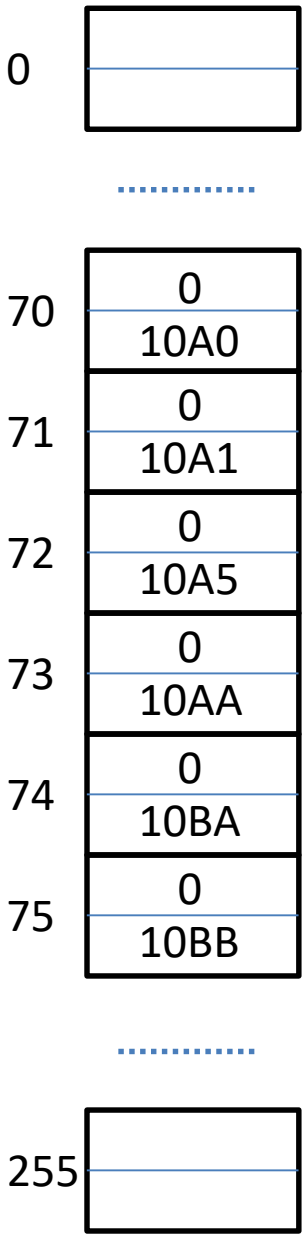
  Altre combinazioni di CS e IP sono lecite.



Elemento di IVT per ISR0

- La IVT inizia all'indirizzo 00000h
- Ogni elemento occupa 4 byte
- L'elemento che contiene l'indirizzo della ISR0 è posto all'indirizzo 70h × 4 = 1C0h

  (0111 0000 × 4 = 1 1100 0000)



Parte C
Esercizio di programmazione

- E’ possibile consultare l’instruction set dell’Intel Assembler 80186
  - fornito all’esame
  - disponibile anche fra il materiale del corso.
- Tempo: 60 minuti.
- Valutazione: massimo 12 punti.
- Sono necessari almeno 5 punti per superare l’esame.

Domanda

Il problema dello zaino è un problema di ottimizzazione combinatoria che può essere formulato come segue. Sia dato uno zaino capace di sopportare un peso W. Sono disponibili N oggetti, ciascuno caratterizzato da un peso w_i e da un valore c_i. Si vuole individuare quali oggetti inserire nello zaino al fine di massimizzare il valore totale.

Esiste un algoritmo greedy che permette di trovare una *buona* soluzione al problema dello zaino. Per ogni oggetto è calcolato il costo unitario w_i / c_i. E' valutato un oggetto alla volta, secondo l'ordine decrescente del costo unitario. L'oggetto corrente è inserito nello zaino se e solo se il suo peso non supera la capacità residua dello zaino.

Domanda (cont.)

Si scriva in linguaggio Assembly 8086 una procedura **riempiZaino** che trovi una soluzione greedy al problema dello zaino. La procedura riceve tramite stack:

- l'offset del vettore *valore*, che specifica il valore di ogni oggetto
- l'offset del vettore *peso*, che specifica il peso di ciascun oggetto
- l'offset del vettore *stato*, che indica se l'oggetto è stato già esaminato. Gli elementi del vettore possono assumere 3 valori:
    - 0: l'oggetto non è stato ancora esaminato
    - 1: l'oggetto è stato inserito nello zaino
    - -1: l'oggetto non può essere inserito nello zaino perché supera la capacità residua.
- un valore, su un byte, che indica il peso degli oggetti attualmente presenti nello zaino.

Domanda (cont.)

Tutti i vettori sono di tipo byte e hanno N elementi (N è dichiarata come costante, così come W). Non è ammesso l'uso di altre variabili.

La procedura **riempiZaino** deve:

1. trovare l'oggetto con il massimo costo unitario fra quelli il cui stato è 0. Per il calcolo del costo unitario, si consideri solo la parte intera del rapporto w_i / c_i. In caso di più oggetti con lo stesso costo unitario massimo, se ne prenda uno a scelta.
2. controllare se l'oggetto individuato sta nello zaino (ossia, il peso dell'oggetto sommato al peso degli oggetti già messi nello zaino è inferiore alla capacità dello zaino).
   - Se l'oggetto sta nello zaino, la procedura aggiorna il peso degli oggetti attualmente presenti nello zaino (modificando il valore nello stack) e pone a 1 lo stato dell'oggetto.
   - Se l'oggetto non sta nello zaino, lo stato dell'oggetto è posto a -1

Domanda (cont.)

Di seguito un esempio di programma chiamante:

```
N EQU 9          ; numero di oggetti
W EQU 27         ; capacita' dello zaino
.MODEL small
.STACK
.DATA
valore DB 200, 225, 250, 150, 125, 125,
120, 120, 75
peso   DB 10,  12,  15,  9,   7,   6,   6,   6,   4
stato  DB N DUP (0)
```

```
.CODE
.STARTUP
PUSH OFFSET valore
PUSH OFFSET peso
PUSH OFFSET stato
MOV AX, 0
PUSH AX
MOV CX, N
ciclo:
CALL riempiZaino
LOOP ciclo
ADD SP, 8
.EXIT
```

Domanda (cont.)

Al termine del programma, con i valori nell'esempio, il vettore *stato* è: 1, -1, -1, -1, -1, 1, 1, -1, 1.

Gli elementi inseriti nello zaino, così come indicato nel vettore *stato*, hanno peso complessivo 26 e valore 445.

Soluzione

```
riempiZaino PROC
MOV BP, SP
PUSH CX

MOV CX, N
MOV SI, 0
MOV DL, 0   ; memorizza il massimo corrente

cicloProc:
; controlla se l'elemento e' ancora disponibile
MOV BX, [BP + 4]
CMP [BX][SI], 0
JNE fineCiclo
```

Soluzione (cont.)

```
; calcola il rapporto valore / peso
MOV BX, [BP + 8]
MOV AL, [BX][SI]
MOV AH, 0
MOV BX, [BP + 6]
MOV BL, [BX][SI]
DIV BL
; e' il migliore rapporto trovato finora?
CMP AL, DL
JBE fineCiclo
MOV DL, AL
MOV DI, SI      ; salva la posizione del massimo corrente
```

Soluzione (cont.)

```
fineCiclo:
INC SI
LOOP cicloProc

; l'elemento sta nello zaino?
MOV AL, [BP + 2]
MOV BX, [BP + 6]
MOV AH, [BX][DI]
ADD AL, AH
CMP AL, W
JA scarta
```

Soluzione (cont.)

```
; mette l'elemento nello zaino
MOV [BP + 2], AL
MOV BX, [BP + 4]
MOV [BX][DI], 1
JMP fineProc
 ; scarta l'elemento perche' ha peso superiore al peso libero
scarta:
MOV BX, [BP + 4]
MOV [BX][DI], -1
fineProc:
POP CX
RET
riempiZaino ENDP
END
```