Nome, cognome, matricola .....

Calcolatori Elettronici (12AGA) – esame del 11.2.2020

Domande a risposta chiusa (è necessario rispondere correttamente ad almeno 6 domande).
Non è possibile consultare alcun tipo di materiale. Tempo: 15 minuti.
!!!! Attenzione: il compito è su 2 facciate !!!!

1	Si consideri un sistema sequenziale caratterizzato da • 24 segnali di ingresso • 18 segnali di uscita • 76 stati. Se si utilizza il modello di Huffman per la sua implementazione, quanti sono i segnali di ingresso della logica combinatoria?	

2	Si consideri un contatore binario in avanti su 5 bit. Assumendo che il contatore abbia il valore 31, che cosa succede all'arrivo di un nuovo fronte sul segnale di conteggio?	Il contatore assume il valore 32	A	
		Il contatore assume il valore 0	B	
		Il contatore si ferma al valore 31 e smette di contare sino a che non viene resettato	C	
		Il contatore assume il valore 30	D	

3	Quante porte NOR sono necessarie per realizzare un flip-flop SR asincrono?	1	A	
		2	B	
		6	C	
		10	D	

4	Quale vantaggio presenta una cache set associative rispetto ad una cache direct mapped?	Maggiore semplicità di progettazione	A	
		Minor costo hardware	B	
		Minore tempo di accesso	C	
		Maggior valore della hit ratio media	D	

5	Si consideri la Memory Management Unit (MMU), utilizzata per supportare il meccanismo della Memoria Virtuale. Quale delle seguenti affermazioni è <u>falsa</u>?	La MMU è un blocco di logica che traduce gli indirizzi logici in indirizzi fisici	A	
		La MMU contiene il Translation Lookaside Buffer (TLB)	B	
		La MMU contiene la Memory Address Table (MAT)	C	
		La MMU è interposta tra CPU e cache	D	

6	Si consideri una cache con le seguenti caratteristiche · 64 linee da 16 byte · direct mapping · write-through. Assumendo che gli indirizzi emessi dal processore siano su 32 bit, in quale linea è memorizzata la parola con indirizzo esadecimale 6692 0A12?	

7	Quale delle seguenti tecnologie (o combinazione di tecnologie) di memoria viene normalmente utilizzata per realizzare la memoria secondaria in un sistema general-purpose?	RAM	A	
		Flash/dischi magnetici	B	
		DVD/CD-ROM	C	
		ROM	D	

8	Qual è il principale vantaggio della micro-programmazione verticale rispetto a quella orizzontale?	Maggiore velocità	A	
		Maggiore semplicità nel progetto dell'unità di controllo	B	
		Maggior numero di istruzioni supportate	C	
		Minore dimensione della memoria di microcodice	D	

9	Quando un processore MIPS esegue la generica istruzione `jal lab`, dove viene salvato l'indirizzo di ritorno?	Nello stack	A	
		Nel registro `$v0`	B	
		Nella variabile `lab`	C	
		Nel registro `$ra`	D	

10	Si scriva un frammento di codice in Assembler MIPS che, date due variabili con segno su 32 bit VAR1 e VAR2, scriva in una terza variabile VAR3 (sempre con segno e su 32 bit) il valore 1 se VAR1 > VAR2, -1 altrimenti.	

Risposte corrette

1	2	3	4	5	6	7	8	9	10
31	B	B	D	C	33	B	D	D	

Esempio di soluzione per la domanda #10:

```
li      $t3, -1
li      $t4, 1
lw      $t0, VAR1
lw      $t1, VAR2
bgt     $t0, $t1, salta
sw      $t3, VAR3
j       fine
salta:
sw      $t4, VAR3
fine:
…
```

Nome, cognome, matricola ….....…….....…….....

Compito A

Domande a risposta aperta (sino a 5 punti per ogni domanda) – Non è possibile consultare alcun materiale - Tempo: 40 minuti.

11	Si consideri un sistema a processore che utilizza un DMA Controller per gestire la comunicazione con un dispositivo di output. Si disegni lo schema di connessione tra CPU, DMA Controller, memoria e dispositivo periferico, riportando i relativi segnali di interconnessione. Si descrivano le operazioni eseguite dal sistema a partire dal momento in cui esso programma il DMA Controller per eseguire il trasferimento di un blocco di dati dalla memoria al periferico e fino al termine dell'operazione di trasferimento, assumendo che venga utilizzato il meccanismo del trasferimento a blocchi (burst transfer).
12	Si descrivano le caratteristiche di un processore RISC.

13	Si progetti un circuito combinatorio minimo avente 4 ingressi a, b, c e d e un'uscita u che va a 1 se e solo se è vera l'espressione (a=c) OR (b!=d). Nella risposta si richiede di riportare 1. La tavola di verità 2. La mappa di Karnaugh 3. La relativa copertura 4. Il circuito minimo.
14	Si disegni l'architettura di un'unità di controllo microprogrammata che utilizza la microprogrammazione orizzontale e se ne descriva il funzionamento. Assumendo che la memoria di microcodice sia composta da 250 parole da 180 bit ciascuna e che non esistano microistruzioni di salto, si dimensionino il μPC e il μIR.

Nome, Cognome, Matricola:……

Esercizio di programmazione

sino a 12 punti – è possibile consultare solamente il foglio consegnato con l'instruction set MIPS - tempo: 60 minuti – scrivere in stampatello

Si vuole realizzare un programma in linguaggio MIPS per comprimere un vettore di byte contenente numeri interi senza segno, utilizzando il seguente algoritmo di compressione:

- gli elementi del vettore di partenza sono quantizzati
- una sequenza di elementi uguali (dopo la quantizzazione) è sostituita con un unico elemento.

Esempio: si supponga che l'intervallo di quantizzazione sia 10, ossia il vettore quantizzato contenga solo multipli di 10. I vettori dopo la quantizzazione e finale, rispettivamente, sono riportati di seguito.

vettore iniziale	14	16	18	134	24	22	23	149	140	141	145	146
vettore quantizzato	10	10	10	130	20	20	20	140	140	140	140	140
vettore finale	10	130	20	140								

Si richiede di implementare la procedura `comprimi` che implementa l'algoritmo di compressione descritto. La procedura riceve in input:

1. indirizzo del vettore di byte senza segno da comprimere
2. numero di elementi nel vettore da comprimere
3. indirizzo di un vettore di byte non inizializzato, che conterrà i valori compressi.

La procedura restituisce il numero di elementi nel vettore di byte compresso.
La procedura scandisce ogni elemento del vettore. Ad ogni iterazione del ciclo, richiama la procedura `quantizza` (il cui codice è fornito sotto) per ottenere il valore quantizzato dell'elemento corrente. Dopo aver ottenuto il valore quantizzato, la procedura `comprimi` lo confronta con l'ultimo elemento inserito nel vettore compresso. Il nuovo valore quantizzato è inserito nel vettore compresso solo se diverso.

La procedura `quantizza` riceve come unico argomento l'elemento corrente (l'intervallo di quantizzazione è definito tramite costante) e restituisce il valore quantizzato; la sua implementazione è fornita sotto.
Di seguito un esempio di programma chiamante e il codice della procedura `quantizza`:

```
        .data
vettore:            .byte 14, 16, 18, 134, 24, 22, 23, 149, 140, 141, 145, 146
vettoreCompresso:   .space 12
INTERVALLO_QUANT = 10

        .text
        .globl main
        .ent main
main:
        subu $sp, $sp, 4
        sw $ra, ($sp)
        la $a0, vettore
        li $a1, 12
        la $a2, vettoreCompresso
        jal comprimi

        lw $ra, ($sp)
        addu $sp, $sp, 4
        jr $ra
        .end main

quantizza:
        divu $t0, $a0, INTERVALLO_QUANT
        mulou $v0, $t0, INTERVALLO_QUANT
        jr $ra
```

Soluzione proposta

```
comprimi:         subu $sp, $sp, 24       # salvataggio registri
                  sw $ra, ($sp)
                  sw $s0, 4($sp)
                  sw $s1, 8($sp)
                  sw $s2, 12($sp)
                  sw $s3, 16($sp)
                  sw $s4, 20($sp)

                  move $s0, $a0
                  move $s1, $a1
                  move $s2, $a2
                  li $s3, 0         # numero di elementi nel vettore compresso

inizioCiclo:      lbu $a0, ($s0)
                  jal quantizza
                  beqz $s3, nuovoValore   # prima iterazione
                  beq $s4, $v0, fineCiclo

nuovoValore:      move $s4, $v0
                  sb $s4, ($s2)
                 addi $s2, $s2, 1
                  addi $s3, $s3, 1

fineCiclo:        addi $s0, $s0, 1
                  subu $s1, $s1, 1
                  bnez $s1, inizioCiclo

                  move $v0, $s3     # valore di ritorno e ripristino registri
                  lw $ra, ($sp)
                  lw $s0, 4($sp)
                  lw $s1, 8($sp)
                  lw $s2, 12($sp)
                  lw $s3, 16($sp)
                  lw $s4, 20($sp)
                  addu $sp, $sp, 24
                  jr $ra
```