Università degli Studi di Udine

DMIF — Dipartimento di Scienze Matematiche, Informatiche e Fisiche

Esame **B** di Architetture degli Elaboratori

Soluzione

A.A. 2019-20 — I appello — 22 gennaio 2020

N.B.: il punteggio associato ad ogni domanda è solo una misura della difficoltà, e peso, di ogni domanda. Per calcolare il voto complessivo bisogna normalizzare a 32.

1. Si converta il numero periodico misto $15.0\overline{52}_8$ nella base 4.

   R: (3 pt) Essendo entrambe le basi una potenza di due, nel caso particolare abbiamo che le triplette di cifre binarie ciascuna ottenuta da una cifra appartenente al numero di partenza vengono aggregate e, successivamente, raggruppate in coppie di cifre binarie che, convertite nella nuova base, formano il numero di arrivo:

   $$15.0\overline{52}_8 = \underbrace{1}_{1}\underbrace{101}_{5}.\underbrace{000}_{0}\underbrace{\overline{101}}_{\overline{5}}\underbrace{\overline{010}}_{\overline{2}}{}_2 = \underbrace{11}_{3}\underbrace{01}_{1}.\underbrace{00}_{0}\underbrace{\overline{010101}}_{\overline{111}}{}_2 = \underbrace{11}_{3}\underbrace{01}_{1}.\underbrace{00}_{0}\underbrace{\overline{01}}_{\overline{1}}{}_2 = 31.0\overline{1}_4$$

2. Si verifichi se l'operazione $3 \cdot (-2) = -6$ può essere svolta in modo corretto da un'architettura di calcolo eseguendo l'operazione *direttamente* nella codifica in complemento a 2 a 8 bit.

   R: (3 pt) Moltiplicando le rispettive codifiche in complemento a 2 a 8 bit si ha immediatamente

   ```
     00000011
     11111110 X
     --------
     11111110
    11111110
   ----------
   1011111010
   ```

   che è proprio la codifica in complemento a due del numero -6, a 8 bit una volta che siano stati scartati i due bit di overflow.

3. [INF] Convertire il numero 9.125 in codifica *floating point* IEEE 754 a 32 bit.

   R: (3 pt) Il numero può essere covertito in base 2: $9.125 = \texttt{1001.001}_2 = \texttt{1.001001}_2\texttt{E3}$. La codifica richiesta avrà dunque bit di segno non asserito, esponente uguale a $127 + 3 = 130 = \texttt{10000010}_2$ e infine mantissa uguale a $\texttt{001001}_2$. Sistemando sui 32 bit previsti dallo standard IEEE 754 e convertendo alla base esadecimale:

   ```
   0|1 0 0 0 0 0 1 0|0 0 1 0 0 1 0 0 0 ...
      4   |   1   |   1   |   2   | 0 ...
   ```

   da cui la codifica richiesta: `0x41120000`.

4. Un chip di memoria di 32 GB negli anni assume le seguenti dimensioni in cm^2: 8×4 nel 2016, 4×4 nel 2018, 4×2 nel 2020. Assumendo la costanza nel tempo della crescita esponenziale della densità di transistor, quali saranno tipicamente le dimensioni del chip nel 2024?

   R: (3 pt) É semplice osservare che l'area del chip dimezza ogni due anni. Nel 2024 quindi il chip avrà un'area uguale a un quarto di quella posseduta nel 2020. Quindi, per esempio, misurerà 2×1 cm^2.

5. Scrivere la tabella di verità che realizza la seguente proposizione: "se Marco va a dormire, ha studiato e non è agitato allora Marco svolge correttamente l'esame; se Marco è agitato allora Marco non svolge correttamente l'esame".

   R: (3 pt) Facendo corrispondere le variabili `A` a "Marco va a dormire", `B` a "ha studiato", `C` a "è agitato" ed `E` a "Marco svolge correttamente l'esame" allora si ha subito

```
A B C   E
0 0 0   -
0 0 1   0
0 1 0   -
0 1 1   0
1 0 0   -
1 0 1   0
1 1 0   1
1 1 1   0
```

6. [INF] Si realizzi la tabella di Karnaugh che calcola la tabella di verità dell'esercizio 5.

   R: (3 pt)

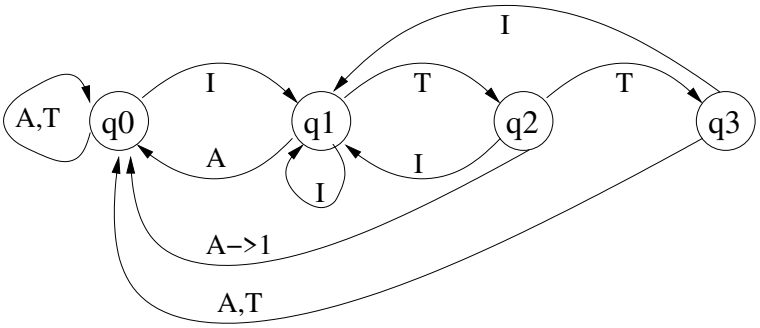
```
BC   00 01 11 10
    -------------
A 0 | -  0  0  - |
  1 | -  0  0  1 |
    -------------
```

7. Si realizzi la rete booleana che calcola la tabella di verità dell'esercizio 5.

   R: (3 pt) Ovvia, dall'espressione $\mathtt{E} = \overline{\mathtt{C}}$ ottenuta scegliendo di porre al valore uno le uscite indeterminate.

8. [INF] Si realizzi il diagramma di stato della macchina di Mealy definita sull'alfabeto $\mathcal{A} = \{A, I, T\}$, la quale produce in uscita il valore uno ogni volta che riconosce le parole `ITA` e `ITTITA`, altrimenti producendo il valore zero a ogni simbolo ricevuto.

   R: (3 pt) Le uscite uguali a zero sono state omesse nel diagramma.



   Si noti che l'esercizio ammetteva anche una semplice soluzione a tre stati, se si notava che `ITTITA` termina con la stringa `ITA` che quindi era l'unica che bastava riconoscere.

9. Una compagnia telefonica è in possesso della tecnologia per adoperare un codificatore di simboli binari che realizza un codice di *parità dispari* a N bit, il cui circuito quindi aggiunge un N-esimo bit di parità a ciascuna $(N-1)$-upla di bit che rappresentano il simbolo da trasmettere. Al fine di rinforzare il codice, la stessa compagnia decide di accodare un ulteriore codificatore identico al precedente in modo che questo, partendo dalla codifica appena ricevuta, aggiunga un ulteriore $(N+1)$-esimo bit di parità a ciascuna N-upla di bit che rappresentano la codifica. La decisione presa dalla compagnia è utile a rinforzare il codice?

   R: (3 pt) Evidentemente no: poichè il codice di N bit contiene in ogni caso un numero dispari di simboli uno, il secondo codificatore aggiungerà un ulteriore bit di parità che risulta sempre essere uguale a zero.

10. Un bus parallelo trasmette bit su dieci canali con un periodo di clock $T_{10} = 2.5$ ns, uguale per ciascun canale. Nell'ipotesi di ridurre a quattro il numero di canali, quale sarà il periodo T_4 necessario a mantenere la stessa banda passante?

    R: (3 pt) La banda passante viene mantenuta costante se vale la relazione $10 f_{10} = 4 f_4$, in cui f_n è la frequenza del bus quando i canali sono n. Da $T_n = 1/f_n$ si ha subito $10 T_4 = 4 T_{10}$, da cui $T_4 = (4/10) T_{10} = (4/10) \cdot 2.5 = 10/10 = 1$ ns.

11. Ricordando che la scheda Arduino UNO ha un *clock* a 16 MHz, si dica dopo quanto tempo il `Timer1` (a 16 bit) va in *overflow* nel caso in cui il *prescaler* sia settato a 1 e nel caso in cui sia invece settato a 256.

    R: (3 pt) Nel caso in cui il *prescaler* sia a 1 abbiamo *overflow* dopo $1 \cdot 2^{16}/(16 \cdot 10^6) = 4096 \cdot 10^{-6}$ s. Nel caso in cui il *prescaler* sia a 256 abbiamo *overflow* dopo $256 \cdot 2^{16}/(16 \cdot 10^6) = 1048576 \cdot 10^{-6}$ s ≈ 1 s.

12. Un'architettura parallela che adopera 9 CPU esegue un algoritmo per cui la legge di Amdahl porge come risultato il rapporto 9/5. Qual è la *frazione di tempo parallelizzabile* f ammessa per l'algoritmo eseguito in quell'architettura?

    R: (3 pt) Sostituendo nella formula:
    $$\frac{\text{velocità } n \text{ CPU}}{\text{velocità 1 CPU}} = \frac{n}{1+(n-1)(1-f)} = \frac{9}{1+(9-1)(1-f)} = \frac{9}{1+8(1-f)} = \frac{9}{5},$$
    da cui subito $f = 0.5$.

13. Avendo a disposizione una memoria RAM fisica di 1 MB, in un'architettura a 32 bit fornita di sufficiente memoria di massa si vuole realizzare una memoria virtuale che sfrutta tutto l'intervallo possibile di indirizzamento sfruttando la tecnica di paginazione. Assumendo che ogni pagina abbia un'estensione di 64 kB, si calcolino le dimensioni della *page table* necessaria al funzionamento della memoria paginata.

    R: Ogni pagina occupa $64 \cdot 2^{10} = 2^{16}$ Byte. La *page table* dovrà quindi mappare $2^{32}/2^{16} = 2^{16}$ pagine. Quindi, essa sarà costituita da 2^{16} righe. Poichè la RAM ha estensione uguale a 1 MB $= 2^{20}$ Byte, ciascuna riga sarà lunga 20 bit più il bit di presenza/assenza. L'estensione totale della *page table* è quindi di $(20+1) \cdot 2^{16} = 21 \cdot 2^{13} \cdot 2^3 = 168$ KB.

14. [INF] Scrivere un programma in assembly per ARM il quale ruota verso sinistra un array di 10 elementi presente in memoria. Il numero di posizioni di cui ruotare ogni elemento dell'array è dato dal valore negativo contenuto del registro `R1`. Se lo stesso valore è positivo o nullo allora l'array non viene ruotato. É gradita la presenza di commenti al codice prodotto.

    R: (9 pt)

```
        .data
array:
        .word 1,2,3,4,5,6,7,8,9,10
        .text
main:
        mov r1, #-7             ; number of leftward rotations in r1
        mov r2, #9              ; array length-1 in r2
loop1:                          ; iterate rotations
        adds r1, r1, #1         ; increment r1 and set sign
        bgt exit                ; if r1>0 then exit
        ldr r0, =array          ; array head position in r0
        mov r3, r0              ; array head position in r3
        add r3, r3, r2, lsl #2  ; array tail position in r3
        ldr r4, [r0]            ; save leftmost element in r4
        add r0, r0, #4          ; increment array pointer
loop:                           ; iterate element shifts
        cmp r3, r0              ; if r0>r3..
        strlt r4, [r3]          ; ..write leftmost element in rightmost pos
        blt loop1               ; ..go to next rotation
        ldr r5, [r0], #-4       ; load element to shift in r5, dec r0
        str r5, [r0], #8        ; write shifted element, inc r0 by 2 words
        b loop                  ; repeat loop
exit:
        swi 0x11                ; exit

        .end
```