Università degli Studi di Udine

DMIF — Dipartimento di Scienze Matematiche, Informatiche e Fisiche

Esame **A** di **Architetture degli Elaboratori**

Soluzione

A.A. 2018-19 — V appello — 17 settembre 2019

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 valore -347.0017_8 nella base 2.

   R: (3 pt) Ricordando che ogni cifra di un numero positivo o negativo in base $8 = 2^3$ viene mappata in 3 cifre binarie, si ha subito:

   $$-347.0017_8 = -\underbrace{011}_{3}\underbrace{100}_{4}\underbrace{111}_{7}.\underbrace{000}_{0}\underbrace{000}_{0}\underbrace{001}_{1}\underbrace{111}_{7}{}_2.$$

2. Qual è l'intervallo di valori rappresentabili da un'aritmetica in complemento a due a nove bit? Quali sono le codifiche nella stessa aritmetica degli estremi dell'intervallo?

   R: (3 pt) L'intervallo in questione è $[-2^8, 2^8 - 1]$. Gli estremi sono rispettivamente $100000000 = -2^8$ e $011111111 = 2^8 - 1$.

3. [INF] Convertire il numero 2 in codifica *floating point* IEEE 754 a 32 bit.

   R: (3 pt) Il numero può essere subito messo in forma esponenziale: $1.0_2\texttt{E1}$. La codifica richiesta avrà dunque bit di segno non asserito, esponente uguale a $127+1 = 128 = 10000000_2$ e infine mantissa uguale a 0_2. Sistemando sui 32 bit previsti dallo standard IEEE 754 e convertendo alla base esadecimale:

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

   da cui la codifica richiesta: `0x40000000`.

4. Si supponga che ogni anno l'area di un chip di memoria diminuisca di un fattore costante a parità di capacità. Un chip di 3 cm^2 due anni dopo misura 1 cm^2. Di quanto è aumentata la densità di memoria ogni anno?

   R: (3 pt) La densità deve aumentare di un fattore $\sqrt{3}$, in modo che dopo due anni la riduzione dell'area sia di un fattore $\sqrt{3} \cdot \sqrt{3} = 3$.

5. Scrivere le espressioni booleane che dimostrano che una porta NOR può essere realizzata con sole porte NAND a due ingressi.

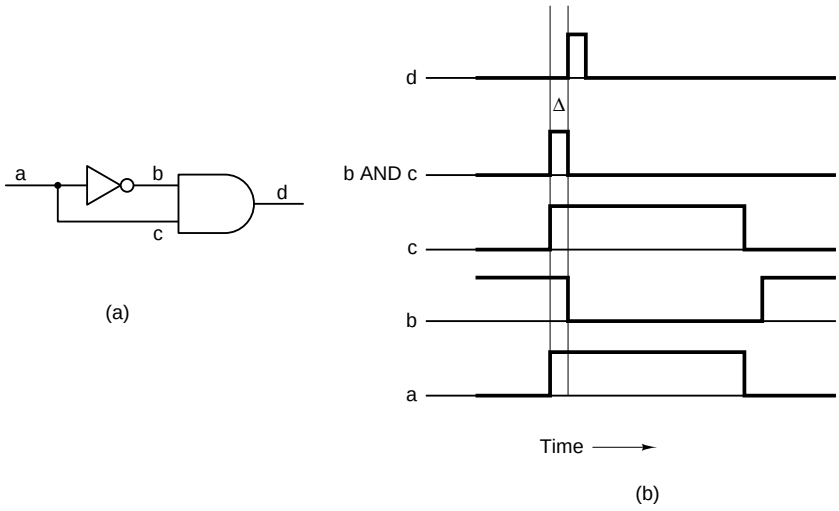
   R: (3 pt) $\overline{A + B} = (\text{De Morgan}) = \overline{A}\,\overline{B} = (\text{A NAND A } = \text{ NOT A}) = \overline{A\overline{A}}\,\overline{B\overline{B}} = (\text{idem}) = \overline{\overline{A\overline{A}}\,\overline{B\overline{B}}}$.

6. [INF] Di quanti transistor necessita la realizzazione di una porta NOR? Di quanti transistor necessita la realizzazione della stessa porta svolta all'esercizio precedente adoperando porte NAND?

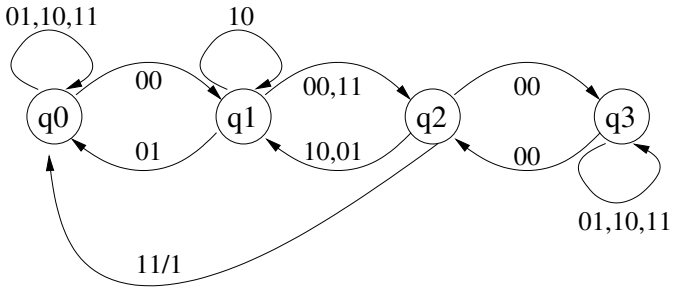
   R: (3 pt) Una porta NOR necessita di 2 transistor. La realizzazione della stessa porta svolta all'esercizio precedente adoperando porte NAND necessita di 4 porte NAND, ovvero di 8 transistor.

7. Ricordando che ogni porta logica introduce un brevissimo ma apprezzabile ritardo temporale, si disegni una semplice rete booleana in grado di trasformare un segnale di clock d'onda quadra temporalmente simmetrica in un nuovo segnale costituito da una sequenza periodica di impulsi temporalmente brevissimi.

   R: (3 pt)



8. [INF] Qual è la tabella di verità della macchina di Mealy in figura definita sull'alfabeto $\mathcal{A} = \{0, 1\}$? Nella stessa figura s'intende che a ogni transizione dello stato l'uscita della macchina è uguale a 0 se non altrimenti specificato.



R: (3 pt) Detti S_1 e S_0 due simboli appartenenti all'alfabeto necessari per specificare lo stato, I_1 e I_0 gli ingressi, E l'uscita, la tabella di verità porge

S_1	S_0	I_1	I_0	S_1^*	S_0^*	E
0	0	0	0	0	1	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	0
0	1	1	1	1	0	0
1	0	0	0	1	1	0
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	0

9. Un codificatore binario di simboli realizza un codice di *parità pari* a 4 bit, il cui circuito quindi aggiunge un quarto bit di parità a ciascuna terna di bit che rappresentano il simbolo da trasmettere. Detti A, B e C i bit che compongono ogni terna, qual è l'espressione booleana realizzata dal circuito che genera il bit di parità P?

R: (3 pt) Poichè il circuito in questione realizza la parità pari, la sua uscita P è uguale a uno solo quando in ingresso ci sono uno oppure tre bit asseriti: $P = \overline{A}\,\overline{B}\,C + \overline{A}\,B\,\overline{C} + A\,\overline{B}\,\overline{C} + A\,B\,C$.

10. Un *multiplexer* convoglia un bus a 32 canali, ciascuno dei quali trasmette a $f = 96$ Mbit/s, su 4 canali che trasmettono a una frequenza di clock maggiore rispetto a quella dei canali in ingresso. Qual è la banda passante di ciascun canale in uscita dal multiplexer?

    R: (3 pt) Data la banda passante di ciascun canale in ingresso e il rapporto di multiplexing $32/4 = 8$, la banda passante di ciascun canale in uscita è uguale a $96 \cdot 8 = 768$ Mbit/s indipendentemente dalla frequenza con cui i bit vengono trasmessi e ricevuti.

11. Sulla scheda Arduino si può impostare il comportamento di un PIN digitale di input o di output? Se sì, come? A questo punto dove si può verificare questa impostazione?

    R: (3 pt) Sì, utilizzando il comando `pinMode(numero_PIN,CANALE)`, ad esempio `pinMode(3,INPUT)`. Il comando va ad agire su uno o più registri di memoria accessibili nel microcontrollore, associati alla porta corrispondente al PIN.

12. Un'architettura a 32 bit prevede la presenza di un'istruzione `sub R1,R2,ARG`, sempre a 32 bit, la quale scrive nel registro della CPU `R1` il risultato della sottrazione dell'argomento `ARG` al contenuto del registro della CPU `R2`. Tenendo conto del fatto che la CPU possiede 16 registri accessibili mediante istruzioni assembly e che il codice operativo che identifica l'istruzione occupa 8 bit, qual è lo spazio disponibile nell'istruzione per specificare l'argomento `ARG`? Si motivi la risposta.

    R: (3 pt) La specifica di 16 registri necessita di 4 bit di spazio. Occorrono dunque 8 bit per specificare `R1` e `R2`. Ulteriori 8 bit se ne vanno per identificare l'istruzione `sub R1,R2,ARG`, che è a 32 bit. Restano dunque disponibili 16 bit per specificare l'argomento `ARG`.

13. Adoperando la legge di Amdahl, si calcoli l'incremento di velocità percentuale rispetto a un'architettura monoprocessore di un'architettura parallela nella quale il numero di unità di calcolo è uguale a 6, contestualmente alla soluzione di un algoritmo avente una frazione di *tempo parallelizzabile* f uguale a $1/5$.

    R: Posto n il numero di unità di calcolo, dalla formula della legge di Amdahl che porge l'incremento di velocità δv rispetto a un'architettura a una unità di calcolo si ha subito

    $$\frac{n}{1+(n-1)(1-f)} = \frac{6}{1+5(1-1/5)} = \frac{6}{1+5\cdot 0.8} = \frac{6}{5} = 1.2 = 1+\delta v,$$

    corrispondente a un incremento percentuale del 20%.

14. [INF] Scrivere un programma in assembly per ARM il quale inverte l'ordine dei caratteri di una stringa presente nella memoria (direttiva assembly `.asciiz` seguita dalla stringa racchiusa tra doppi apici). La stringa risultante deve occupare la stessa area di memoria in cui inizialmente si trovava la stringa da invertire. L'informazione sulla lunghezza della stringa dev'essere estratta dalla stringa stessa. Nota bene: un carattere occupa solamente un byte di memoria. É gradita la presenza di commenti al codice prodotto.

    R: (9 pt)

```
        .data
stringa:
        .asciiz "123ciao"
        .text
main:
        ldr r1, =stringa        ; string tail pos in r1
loop1:  ldrb r2, [r1], #1       ; character in r2
        cmp r2, #0              ; if character is not NULL..
        bne loop1               ; ..then advance tail pos
        sub r1, r1, #1          ; shift tail pos to end-of-string pos
        ldr r0, =stringa        ; string head pos in r0
loop:
        ldrb r2, [r0]           ; head character in r2
        ldrb r3, [r1,#-1]!      ; tail character in r3
        cmp r0, r1              ; if r0-r1..
        bpl exit                ; ..>=0 then exit
        strb r2, [r1]           ; store head char in tail pos
```

```
        strb r3, [r0], #1       ; store tail char in head pos
        b loop                  ; repeat main loop
exit:
        swi 0x11                ; exit

        .end
```