SCHEDA **ASE2304**	Data: **03 Aprile 2023**

Il testo deve essere riconsegnato nella cartellina. Non è ammessa la consultazione degli appunti e dei compiti precedenti. Si possono consultare i data sheet. Non usare il colore rosso nello svolgimento.

ESERCIZIO N°1

5 punti

Siano dati i 2 valori razionali 19/11 e 175/131.

a) Determinare la rappresentazione arrotondata in notazione 1.7 e valutare l'errore di rappresentazione (assoluto e relativo) commesso nei 2 casi.

b) Valutare la rappresentazione del risultato del prodotto tra i due numeri così come ottenuta usando l'operazione FMUL di un microcontrollore AVR in [R1:R0]. Indicare anche il valore del flag C.

ESERCIZIO N°2

8 punti

Realizzare un sottoprogramma per il microcontrollore AVR XMEGA256A3BU, in grado di concatenare 2 stringhe poste in memoria rispettivamente agli indirizzi 0x2000 e 0x2100; la stringa risultante andrà posta in memoria all'indirizzo 0x2400. Una stringa, la cui massima lunghezza può essere di 255 caratteri (codici ASCII su 8 bit), è composta da un primo byte che contiene l'indicazione della lunghezza della stringa stessa (nel caso di stringa vuota l'indicazione di lunghezza è 0) seguito dai codici dei caratteri che compongono la stringa. Se la stringa risultante superasse la massima lunghezza ammessa, andrà troncata ai primi 255 caratteri.

ESERCIZIO N°3

5 punti

Sintetizzare in forma ottima SP la funzione combinatoria non completamente determinata delle 5 variabili X_4, X_3, X_2, X_1 e X_0, individuata dalla seguente tabella di verità
1, 0, 1, -, 1, 0, 1, 0, 1, 0, 1, -, 0, 1, 0, 1, 0, -, 1, 0, 0, -, -, -, 1, -, 0, 1, 1, 1, 0, -.
Indicare gli implicanti essenziali, evidenziando uno dei mintermini che giustificano tale proprietà.

ESERCIZIO N°4

5 punti

Con riferimento al set delle istruzioni dei microcontrollore AVR XMEGA256A3BU, elencare tutte le istruzioni che modificano (o potrebbero modificare) il valore del registro “stack pointer”, indicando il tipo di modifica.

ESERCIZIO N°5

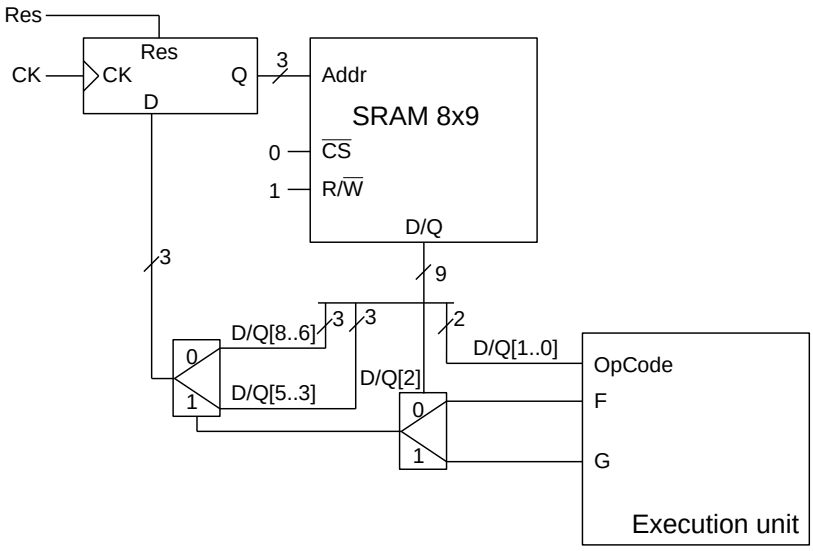
5 punti

Disegnare il grafo di una macchina sequenziale sincrona secondo il modello di Moore con un ingresso e una uscita in grado di riconoscere - ponendo 1 in uscita per 1 ciclo di clock - tutte le sequenze non interallacciate di 3 valori che hanno più 0 che 1. Si preveda uno stato iniziale garantito dalla presenza di flip-flop con reset asincrono, e curare che non avvengano riconoscimenti spuri nella fase iniziale. Codificare gli stati e presentare lo schema a blocchi dell'architettura corrispondente. Non è richiesta la sintesi.

ESERCIZIO N°6

5 punti

Determinare il diagramma di flusso, attribuendo agli stati un nome a scelta, del seguente sequenziatore. Il contenuto della SRAM è costituito dalle 8 parole esadecimali: 0x0A3, 0x1CB, 0x100, 0x059, 0x0F3, 0x15A, 0x1E3, 0x0BB. Sarebbe stato possibile realizzare il sequenziatore con un contatore a caricamento parallelo, risparmiando sulla dimensione della SRAM (spiegare)?



1

Siano dati i 2 valori razionali 19/11 e 175/131.

a) Determinare la rappresentazione arrotondata in notazione 1.7 e valutare l'errore di rappresentazione (assoluto e relativo) commesso nei 2 casi.

b) Valutare la rappresentazione del risultato del prodotto tra i due numeri così come ottenuta usando l'operazione FMUL di un microcontrollore AVR in [R1:R0]. Indicare anche il valore del flag C.

Visto che il peso dell'LSB è 1/128, per arrotondare correttamente possiamo prima moltiplicare per 128 i valori dati e arrotondare all'intero più vicino.

$$X[1.7] = \text{round}(128 \cdot 19/11) = 221 = 0\text{b}1.1011101$$

$$Y[1.7] = \text{round}(128 \cdot 175/131) = 171 = 0\text{b}1.0101011$$

Valuto gli errori

$$\epsilon_a(X) = \frac{221}{128} - \frac{19}{11} = -7,102 \times 10^{-4} \qquad \epsilon_r(X) = \frac{\epsilon_a(X)}{X} = -4,112 \times 10^{-4}$$

$$\epsilon_a(Y) = \frac{171}{128} - \frac{175}{131} = 5,964 \times 10^{-5} \qquad \epsilon_r(Y) = \frac{\epsilon_a(Y)}{Y} = 4,464 \times 10^{-5}$$

Poi possiamo valutare il comportamento della FMUL, facendo il prodotto dei valori 221x171 e poi moltiplicando per 2.
Se il risultato in binario non entra nella coppia di registri R1:R0, si prendono i 16 bit meno significativi e C=1, altrimenti C=0.

Si ottiene:
R1=0b00100111 R0=0b00111110 C=1

2

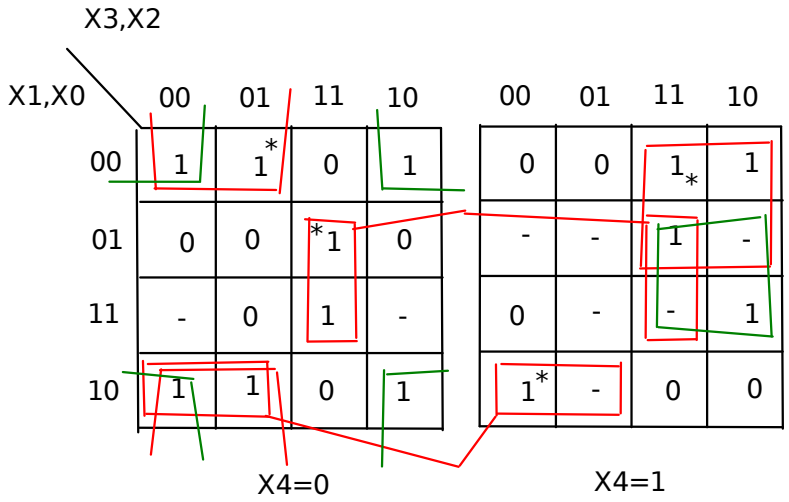
/* Realizzare un sottoprogramma per il microcontrollore AVR XMEGA256A3BU, in grado di concatenare
2 stringhe poste in memoria rispettivamente agli indirizzi 0x2000 e 0x2100;
la stringa risultante andrà posta in memoria all'indirizzo 0x2400.
Una stringa - max 255 caratteri, codici ASCII su 8 bit - è composta da un primo byte che contiene
l'indicazione della lunghezza della stringa stessa (stringa vuota: lunghezza 0)
seguito dai codici dei caratteri che compongono la stringa.
Se la stringa supera la massima lunghezza ammessa, troncare ai primi 255 caratteri. */

```
concatenate:
  push R16      //L1
  push R17      //L2
  push R18      //Ltot
  push R19      //ausiliario
  push XL       //puntatore a stringa1
  push XH
  push YL       //puntatore a stringa2
  push YH
  push ZL       //puntatore al risultato
  push ZH
  ldi XL,low(0x2000)
  ldi XH,high(0x2000)
  ldi YL,low(0x2100)
  ldi YH,high(0x2100)
  ldi ZL,low(0x2400)
  ldi ZH,high(0x2400)
  ld R16,X+           //lunghezza della prima stringa L1
  ld R17,Y+           //lunghezza della seconda stringa L2
  mov R18,R16
  add R18,R17         //somma delle lunghezze Ltot=L1+L2
  brcc no_ov
ov:                   //Ltot>255
  ldi R18,255         //lunghezza della stringa risultato Ltot
  mov R17,R16         //modifica L2
  com R17             //il numero dei caratteri da copiare dalla seconda stringa e` L2=255-L1
no_ov:
  st Z+,R18           //scrive Ltot
  tst R16             //controlla se L1=0
  breq step2
  s1: ld R19,X+
    st Z+,R19
    dec R16
    brne s1
step2:
  tst R17             //controlla se L2=0
  breq end
  s2: ld R19,Y+
    st Z+,R19
    dec R17
    brne s2
end:
  pop ZH              //ripristina i registri modificati
  pop ZL
  pop YH
  pop YL
  pop XH
  pop XL
  pop R19
  pop R18
  pop R17
  pop R16
  ret
```

3

Sintetizzare in forma ottima SP la funzione combinatoria
non completamente determinata, delle 5 variabili X4, X3, X2, X1 e X0,
individuata dalla seguente tabella di verità
1, 0, 1, -, 1, 0, 1, 0, 1, 0, 1, -, 0, 1, 0, 1, 0, -, 1, 0, 0, -, -, -, 1, -, 0, 1, 1, 1, 0, -.
Indicare gli implicanti essenziali, evidenziando uno dei mintermini
che giustificano tale proprietà.

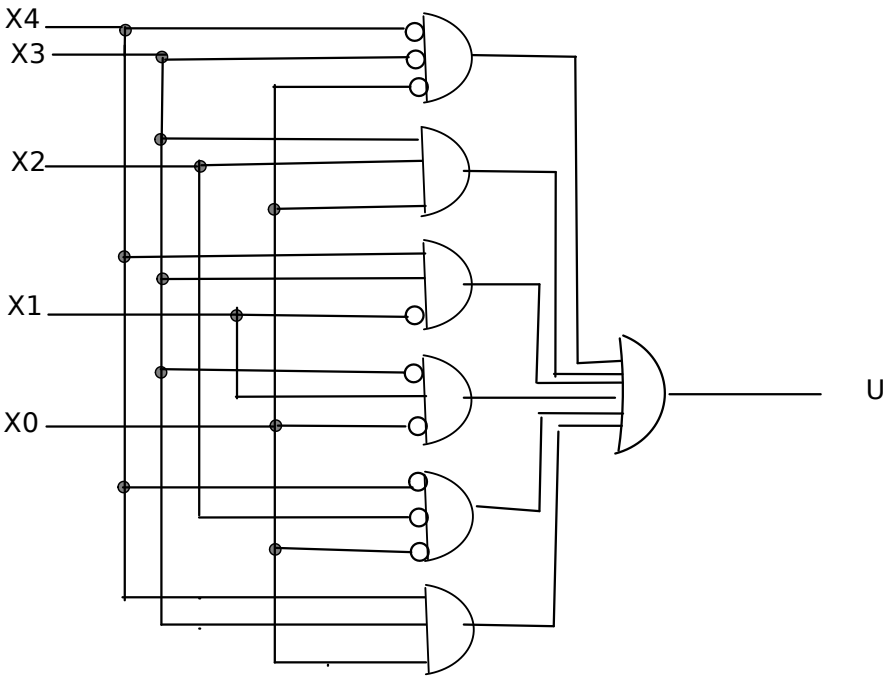
Trasferiamo la tabella di verità in mappa, seguendo l'ordine naturale rispetto al peso delle variabili.



Ci sono 4 implicanti essenziali.
Copro gli altri tre 1 usando
due implicanti principali di ordine
massimo (2).

$$U = \overline{X_4}\ \overline{X_3}\ \overline{X_0} + X_3X_2X_0 + X_4X_3\overline{X_1} + \overline{X_3}X_1\overline{X_0} +$$

$$+ \overline{X_4}\ \overline{X_2}\ \overline{X_0} + X_4X_3X_0$$



4

Con riferimento al set delle istruzioni dei microcontrollore AVR XMEGA256A3BU,
elencare tutte le istruzioni che modificano (o possono modificare)
il valore del registro “stack pointer”, indicando il tipo di modifica.

Usando il manuale del linguaggio assembly, si evidenziano le seguenti istruzioni che alterano o potenzialmente possono alterare (con opportuni valori dei loro argomenti) il registro SP a 16 bit.
Si ricorda che SP è collocato nello spazio di I/O ed è composto dalle due parti
SPH (address 0x003E) e SPL (address 0x003D).

Tutte le chiamate a subroutine salvano nello stack il valore del PC di ritorno. Le istruzioni conclusive della subroutine ripristinano il valore di PC prelevandolo dallo stack. In entrambi i casi viene fatto uso del puntatore SP.

CALL
RCALL
ICALL
EICALL
RET
RETI

Ci sono poi le due istruzioni specifiche che salvano oppure recuperano registri nello stack, modificando SP in modo automatico

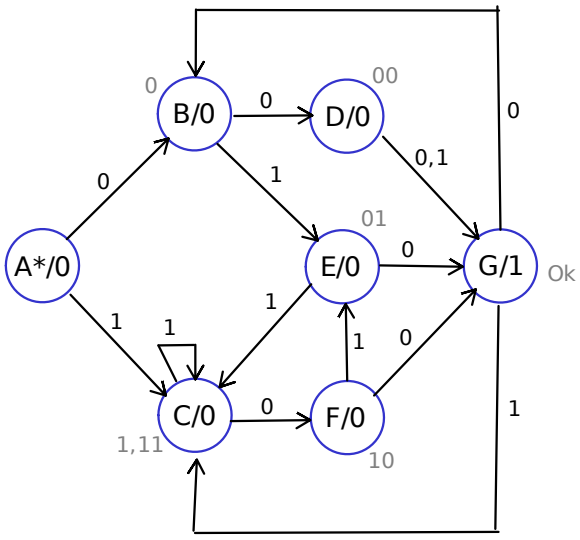
```
PUSH Rr         ;postdecrementa SP
POP Rd          ;preincrementa SP
```

Si può infine intervenire direttamente su SP con le istruzioni di scrittura nello spazio di I/O, usando il valore opportuno di indirizzo (in modo diretto o mettendolo con indirizzamento indiretto nel puntatore).
Poiché l'indirizzo dei due registri SPH e SPL è inferiore a 64, tra queste istruzioni
si può considerare anche la OUT.
Il datasheet ci avvisa che "To prevent corruption when updating the stack pointer from software, a write to SPL will automatically disable interrupts for up to four instructions
or until the next I/O memory write", quindi, per evitare problemi con le interruzioni,
va scritta prima SPL e subito dopo (entro 4 cicli) SPH.

```
STS CPU_SPx,Rr
OUT CPU_SPx,Rr
ST X|Y|X,Rr         ;anche con modifica del puntatore
STD Y+k|Z+K,Rr
XCH Z,Rd
LAC Z,Rd
LAS Z,Rd
LAT Z,Rd
```

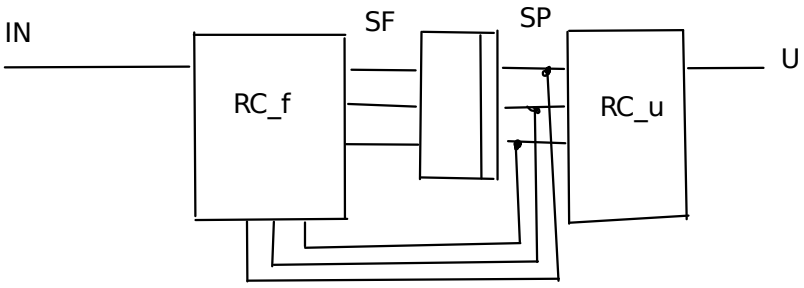
5

Disegnare il grafo di una macchina sequenziale sincrona secondo il modello di Moore
con un ingresso e una uscita in grado di riconoscere - ponendo 1 in uscita per 1 ciclo di clock -
tutte le sequenze non interallacciate di 3 valori che hanno più 0 che 1.
Si preveda uno stato iniziale garantito dalla presenza di flip-flop con reset asincrono,
e curare che non avvengano riconoscimenti spuri nella fase iniziale.
Codificare gli stati e presentare lo schema a blocchi dell'architettura corrispondente.
Non è richiesta la sintesi.

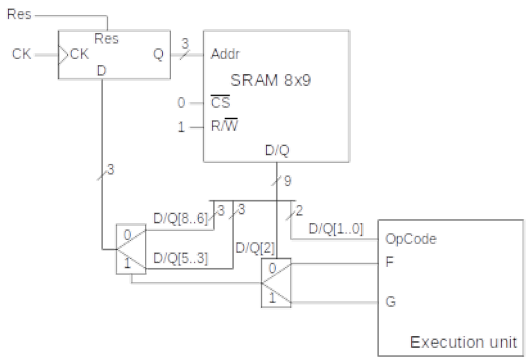


A: 000 stato iniziale (garantito al reset)
B 001
C: 010
D: 011
E: 100
F: 101
G: 111

Architettura



6



Codifica stati:
S0: 000
S1: 001
S2: 002
S3: 011
S4: 100
S5: 101
S6: 110
S7: 111

Contenuto della SRAM: 0x0A3, 0x1CB, 0x100, 0x059, 0x0F3, 0x15A, 0x1E3, 0x0BB
Riscrivo il contenuto della SRAM dividendo i campi e sostituendo il nome dello stato ai codici.

A	S_F	S_V	FL	OP
0	S2	S4	F	Op3
1	S7	S1	F	Op3
2	S4	S0	F	Op0
3	S1	S3	F	Op1
4	S3	S6	F	Op3
5	S5	S3	F	Op2
6	S7	S4	F	Op3
7	S2	S7	F	Op3

Nota: il flag G non viene mai usato. Non è possibile realizzare questo sequenziatore con un contatore perché per entrambi i valori del flag NON si può avere una sequenza ciclica e completa (in entrambe le colonne ci sono stati che si ripetono più volte).

Diagramma di flusso