SCHEDA **ASE1705**	Data: **26 Giugno 2017**
Cognome	Nome

ESERCIZIO N°1
8 punti

Realizzare un sottoprogramma per il microcontrollore AVR XMEGA256A3BU, in grado di verificare se una stringa posta in memoria a partire dall'indirizzo contenuto in X, contiene esattamente (compreso il caso di uguaglianza tra le due stringhe) una seconda stringa posta in memoria a partire dall'indirizzo contenuto in Y; il risultato dovrà essere espresso dal valore di ritorno del flag C (C = 1 se la condizione è verificata). 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.
È disponibile il sottoprogramma v_comp che confronta due vettori di N caratteri ($N > 0$ in R17) puntati da X e Y e pone il risultato nel flag Z (Z vero, vettori uguali).
Esempi con C = 1: "accasato-casa", "architetto-tetto", "elettrone-elettrone".
Esempi con C = 0: "accasato-caso", "architetto-archetto", "elettrone-protrone".

ESERCIZIO N°2
4 punti

Disegnare un flip-flop JK, facendo uso di un T-FF e di multiplexer 2:1. Si cerchi di minimizzare l'uso di multiplexer.

ESERCIZIO N°3
6 punti

Disegnare il grafo di una macchina sequenziale sincrona secondo il modello di Mealy sincronizzato con 3 ingressi (che rappresentano una cifra binaria) e una uscita che viene posta a 1 quando un ingresso è pari al quadruplo del precedente modulo 3; semplificare se possibile il grafo e sintetizzare la macchina in accordo al grafo stesso.

ESERCIZIO N°4
4 punti

Determinare, tra tutte le forme normali, quella a minimo numero di letterali che realizza la funzione logica data dall'espressione seguente (priorità: AND, OR, XOR)

$$Y = (A + D)(\bar{A} + \bar{B} + \bar{C} + \bar{D}) \oplus (A + \bar{B} + C)$$

a) Disegnare lo schema a porte logiche della forma trovata.

b) Realizzare la stessa funzione logica usando un decoder 4:16 e una porta OR (oppure NOR) col minimo numero di ingressi.

ESERCIZIO N°5

5 punti

Disegnare lo schema logico di un comparatore digitale (in grado di fornire l'indicazione A < B, A > B oppure A = B) tra numeri a 8 bit con segno rappresentati in C2.

ESERCIZIO N°6

6 punti

Siano dati i 3 valori positivi $1{,}44 \cdot 10^{-2}$; $1{,}37 \cdot 10^{-2}$; $4{,}51 \cdot 10^{5}$

a) Determinare la rappresentazione in virgola mobile IEEE754-2008 (binary32) dei 3 numeri (con arrotondamento al numero di macchina più vicino).

b) Valutare la rappresentazione della somma tra i 3 numeri, eseguita in tutti i modi possibili, eseguita usando 2 volte un unico sommatore a 2 ingressi, con rappresentazione dell'uscita in formato binary32 (usare l'arrotondamento classico se necessario).

c) L'operazione di somma tra numeri "standard" in virgola mobile rispetta la proprietà commutativa? E quella associativa? Motivare le risposte.

1

Realizzare un sottoprogramma per il microcontrollore AVR XMEGA256A3BU, in grado di verificare se una stringa posta in memoria a partire dall'indirizzo contenuto in X, contiene esattamente (compreso il caso di uguaglianza tra le due stringhe) una seconda stringa posta in memoria a partire dall'indirizzo contenuto in Y; il risultato dovrà essere espresso dal valore di ritorno del flag C (C = 1 se la condizione è verificata). 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.
È disponibile il sottoprogramma v_comp che confronta due vettori di N caratteri ($N > 0$ in R17) puntati da X e Y e pone il risultato nel flag Z (Z vero, vettori uguali).

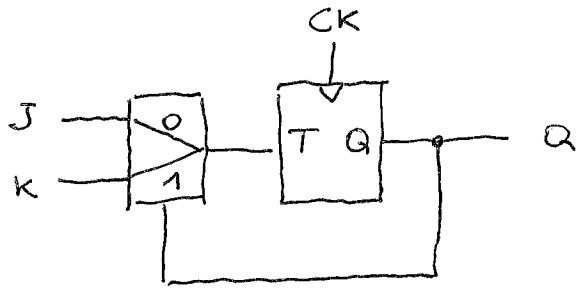
```
string_XY_include:
 push R16 //L1
 push R17 //L2
 push XL
 push XH
 ld R16,X+ //L1, lunghezza di Str1
 ld R17,Y+ //L2, lunghezza di Str2
 tst R17 //se L2 nulla, Str2 è sicuramente inclusa in Str1
 breq cs //carry set
 tst R16 //se nulla Str1 non puo' includere stringhe non nulle
 breq cc //carry clear
 sub R16,R17
 brcs cc //se Str1 è più corta di Str2 non la può includere
 inc R16 //numero di test da eseguire (L1-L2+1), sempre minore di 256
test_eq:
 rcall v_comp //vede se c'è uguaglianza tra L2 caratteri
 breq cs //se uguali Str2 è inclusa in Str1
 adiw XH:XL,1 //prova alla posizione successiva di Str1
 dec R16
 brne test_eq
cc:
 clc
 rjmp restore
cs:
 sec
restore:
 ld R16,-Y //ripristina Y senza alterare i flag
 pop XH
 pop XL
 pop R17
 pop R16
 ret

//questa subroutine è data e non fa parte della soluzione
v_comp:
 push R17
 push R18
 push R19
 push XL
 push XH
 push YL
 push YH
lp:
 ld R18,X+
 ld R19,Y+
 cp R18,R19
 brne fi
 dec R17 //valore iniziale non nullo
 brne lp
fi:
 pop YH
 pop YL
 pop XH
 pop XL
 pop R19
 pop R18
 pop R17
 ret
```

(2) Tabella di transizione del JK (e di eccitazione del T)

Q	J	K	Q^+	T (valore di T per avere il cambiamento di stato desiderato)
0	0	0	0	0
	0	1	0	0
	1	0	1	1
	1	1	1	1
1	0	0	1	0
	0	1	0	1
	1	0	1	0
	1	1	0	1

Dalla tabella si vede che $\begin{cases} T = J \text{ se } Q = 0 \\ T = K \text{ se } Q = 1 \end{cases}$

Questa è proprio la funzionalità di un MUX 2:1.
Lo schema è quindi



(3) condizioni per cui l'uscita deve valere 1

VALORE ATTUALE \ VALORE PRECEDENTE	0	1	2	3	4	5	6	7
0	x			x			x	
1		x			x			x
2			x			x		
≥3								

Alla luce della tabella i valori precedenti:

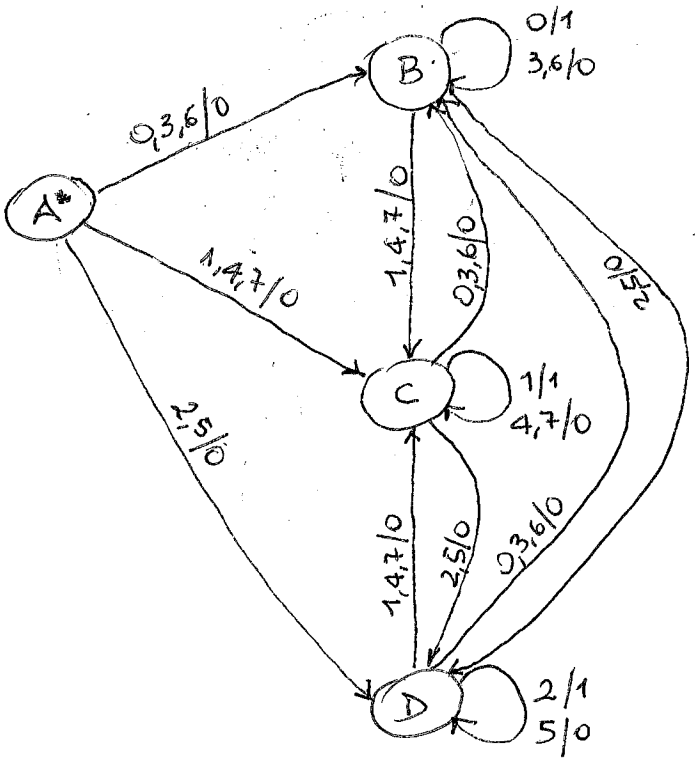
(0, 3, 6)

(1, 4, 7)

(2, 5)

si comportano allo stesso modo.

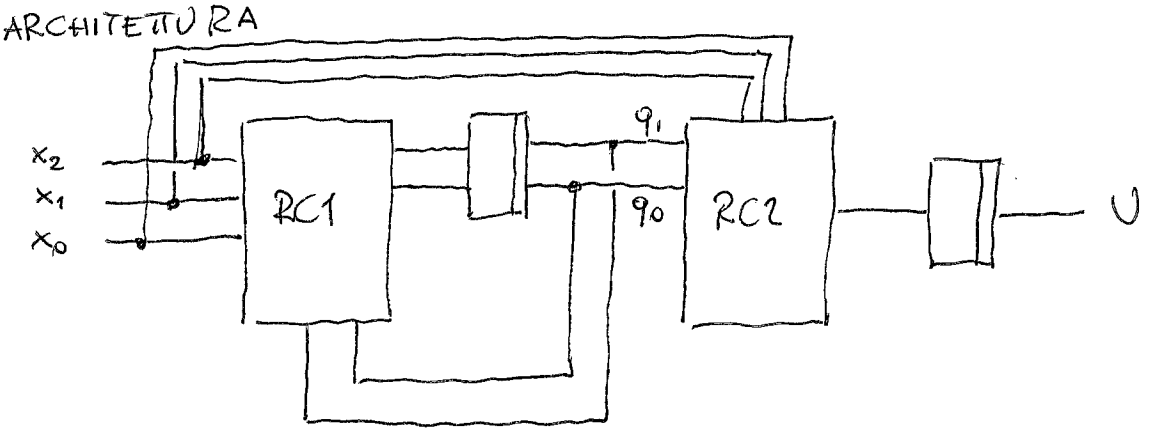
GRAFO (con A* stato iniziale)



codifica stati

	q_1 q_0
A	0 0
B	0 1
C	1 0
D	1 1

ARCHITETTURA



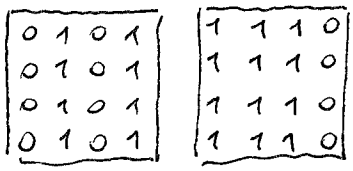
Sintesi di RC_1 e RC_2

$x_2 = 0$

$q_1 q_0$ \ $x_1 x_0$	00	01	11	10
(A) 00	01/0	10/0	01/0	11/0
(B) 01	01/1	10/0	01/0	11/0
(D) 11	01/0	10/0	01/0	11/1
(C) 10	01/0	10/1	01/0	11/0
	(0)	(1)	(3)	(2)

$x_2 = 1$

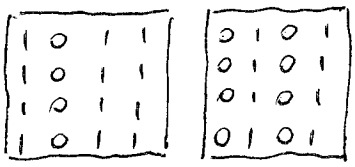
00	01	11	10
10/0	11/0	10/0	01/0
10/0	11/0	10/0	01/0
10/0	11/0	10/0	01/0
10/0	11/0	10/0	01/0
(4)	(5)	(7)	(6)



$$q_1^+ = \bar{x}_2 x_1 \bar{x}_0 + \bar{x}_1 x_0 + x_2 \bar{x}_1 + x_2 x_0$$

oppure

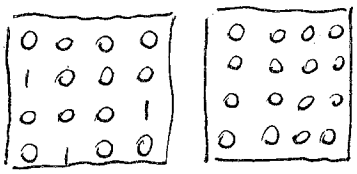
$$q_1^+ = (x_2 + x_1 + x_0)(x_2 + \bar{x}_1 + \bar{x}_0)(\bar{x}_2 + \bar{x}_1 + x_0)$$



$$q_0^+ = \bar{x}_2 \bar{x}_1 \bar{x}_0 + \bar{x}_2 x_1 + x_2 \bar{x}_1 x_0 + x_1 \bar{x}_0$$

oppure

$$q_0^+ = (x_2 + x_1 + \bar{x}_0)(\bar{x}_2 + x_1 + x_0)(\bar{x}_2 + \bar{x}_1 + \bar{x}_0)$$



$$U = \bar{x}_2 \bar{x}_1 \bar{x}_0 \bar{q}_1 q_0 + \bar{x}_2 x_1 \bar{x}_0 q_1 q_0 + \bar{x}_2 \bar{x}_1 x_0 q_1 \bar{q}_0$$

④ Sintesi

$$Y = (A+D)(\bar{A}+\bar{B}+\bar{C}+\bar{D}) \oplus (A+\bar{B}+C) =$$

Mappe dei due termini della XOR

CD \ AB	00	01	11	10
00	0	0	1	1
01	1	1	1	1
11	1	1	0	1
10	0	0	1	1

⊕

CD \ AB	00	01	11	10
00	1	0	1	1
01	1	0	1	1
11	1	1	1	1
10	1	1	1	1

=

S_1S_0 \ S_3S_2 (CD \ AB)	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	1	1	0	0

Sintesi ottima SP

$$Y = \bar{A}\bar{B}\bar{D} + \bar{A}B\bar{C}D + ABCD + \bar{A}C\bar{D} \qquad (14 \text{ letterali})$$

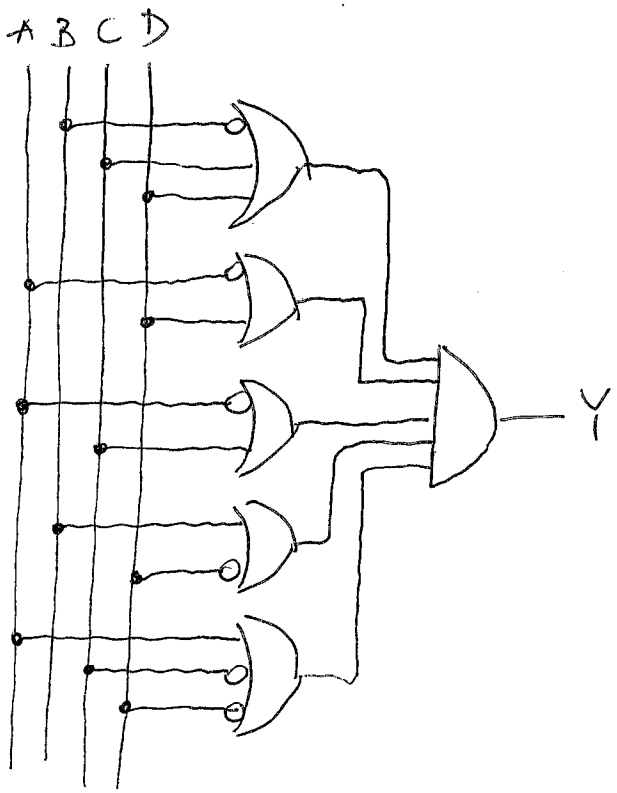
Sintesi ottima PS

$$Y = (\bar{B}+C+D)(\bar{A}+D)(\bar{A}+C)(B+\bar{D})(A+\bar{C}+\bar{D})$$

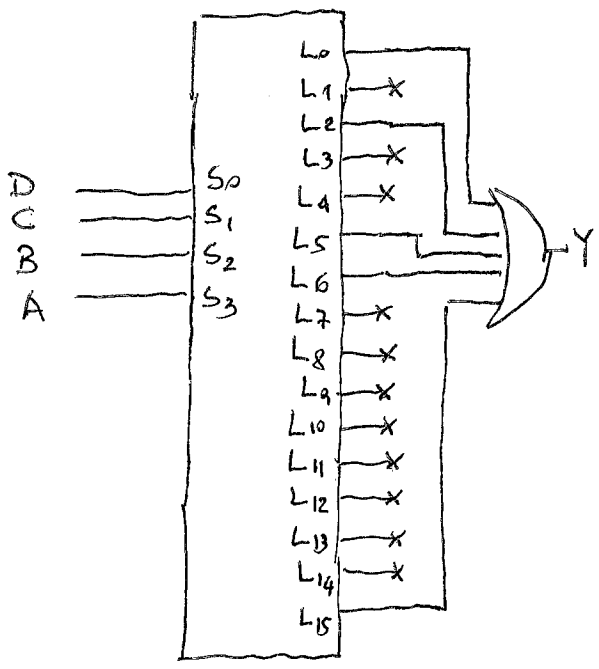
(12 letterali)

1	0	0	0
0	1	0	0
0	0	1	0
1	1	0	0

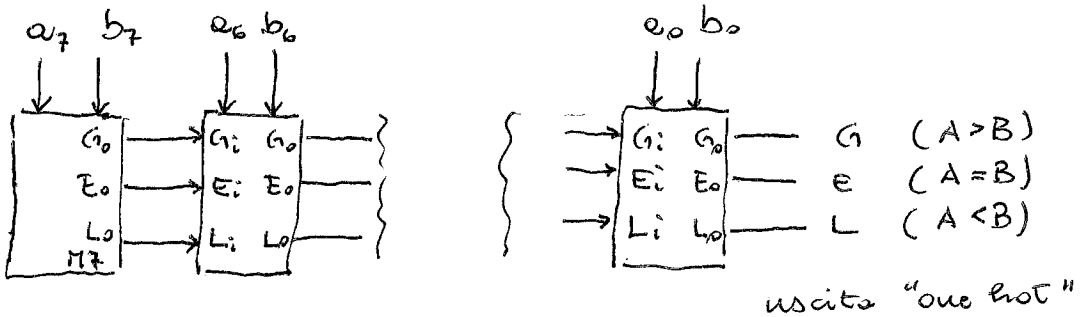
col pallino i MAXTERM. che rendono gli implicati ESSENZIALI



Sintesi con Decoder e porte OR (che meno 1)



(5) Comparatore: schema modulare MSB first



Realizziamo il circuito con 8 moduli: quello iniziale M7 gestisce il segno, gli altri uguali tra loro perfezionano il confronto.

Tabella di M7

a_7	b_7	G_0	E_0	L_0	
0	0	0	1	0	
0	1	1	0	0	B è negativo
1	0	0	0	1	A è negativo
1	1	0	1	0	

Schema di M7

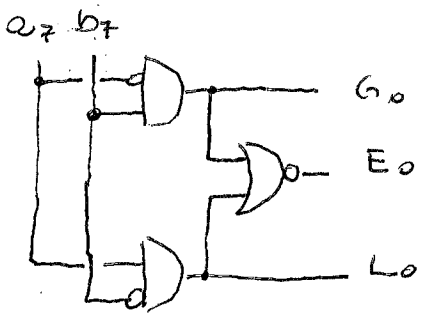
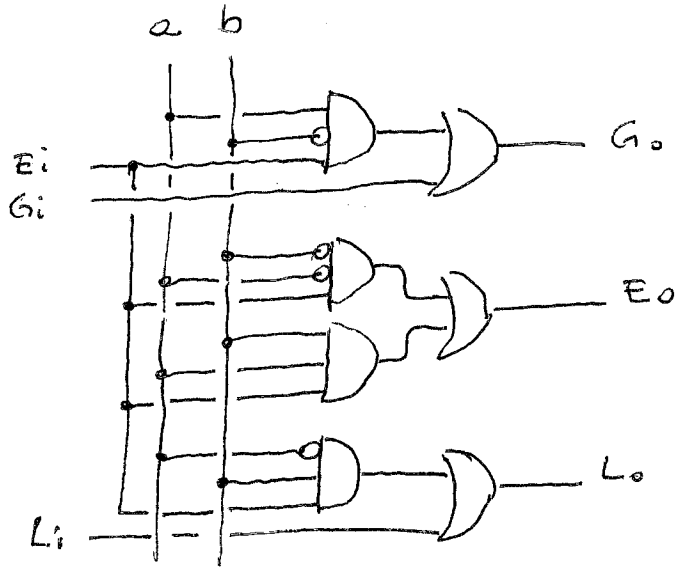


Tabella dei moduli successivi

G_i	E_i	L_i	a	b	G_0	E_0	L_0
1	0	0	–	–	1	0	0
0	0	1	–	–	0	0	1
0	1	0	0	0	0	1	0
0	1	0	0	1	0	0	1
0	1	0	1	0	1	0	0
0	1	0	1	1	0	1	0

Schema dei moduli successivi



(6) Conversione dei numeri formato norm. $x = (-1)^S \, 2^{E-127} \, (1 + T \, 2^{-23})$

$A = 1{,}44 \cdot 10^{-2} = 2^{-7} \cdot 1{,}8432 = 2^{120-127} (1 + 7073274 \cdot 2^{-23})$ con arrot.

$B = 1{,}37 \cdot 10^{-2} = 2^{-7} \cdot 1{,}7536 = 2^{120-127} (1 + 6321655 \cdot 2^{-23})$

$C = 4{,}51 \cdot 10^{5} = 2^{18} \cdot 1{,}7204\ldots = 2^{145-127} (1 + 6043392 \cdot 2^{-23})$

a) espressione binaria

	S	E	T
A:	0	01111000	11010111011101111010
B:	0	01111000	11000000011010111100111
C:	0	10010001	10111000011101110000000

L'operatore somma di 2 floating è simmetrico rispetto agli ingressi. Quindi la somma MANTIENE la proprietà COMMUTATIVA

c) A causa dell'introduzione dell'errore di arrotondamento, introdotto nelle somme parziali, non si ha la garanzia della validità della proprietà ASSOCIATIVA.

Le somme da valutare, nel nostro caso, saranno 3

$S1 = A+B+C$ uguale a $B+A+C$; $C+(A+B)$; $C+(B+A)$

$S2 = A+C+B$ uguale a $C+A+B$; $B+(A+C)$; $B+(C+A)$

$S3 = B+C+A$ uguale a $C+B+A$; $A+(B+C)$; $A+(C+B)$

L'esponente di C è superiore di 25 rispetto agli esponenti di A e B. Quindi allineando i dati nella somma si ha

b) $C+A = C$ $\quad$ $C+B = C$

Da cui $S2 = C$; $S3 = C$

Resta da valutare la somma S1 con arrotondamento classico

$A+B = 2^{121-127} (1 + 6697465 \cdot 2^{-23})$

Rispetto alla somma parziale (A+B) l'esponente di C è superiore di solo 24. Questo provoca l'arrotondamento del risultato:

	S	E	T
S1:	0	10010001	10111000111101110000001