SCHEDA **ASE2208**	Data: **12 Settembre 2022**

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

Determinare la rappresentazione delle 5 cifre BCD meno significative dei seguenti numeri interi assoluti, espressi in varie basi:
$|333777|_8$
$|2120211012|_3$
$|1001101010100010 1010|_2$
$|ABBA1AF0CACC1A|_{16}$

ESERCIZIO N°2

8 punti

Realizzare una subroutine per un microcontrollore della famiglia XMEGA AVR che calcola il numero di sequenze di 16 valori consecutivi uguali (non interallacciati) eventualmente presenti in memoria nell'intervallo di indirizzi da 0x2000 a 0x27FF, compresi gli estremi. Il risultato deve essere lasciato in R16.

ESERCIZIO N°3

5 punti

Realizzare in forma SP ottima una rete combinatoria a 5 ingressi (X_4, X_3, X_2, X_1 e X_0) e una uscita non completamente definita, che ha la tabella di verità seguente:
1,0,−,1,1,0,−,1,1,0,−,−,−,1,0,−,0,1,1,−,1,1,0,−,1,1,1,1,−,1,1,0.
Indicare tutti gli **implicanti essenziali** della funzione, evidenziando un mintermine che giustifica l'indicazione di essenziale.

ESERCIZIO N°4

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 relativi a 6 bit rappresentati in modulo e segno.

ESERCIZIO N°5

6 punti

Progettare una macchina di Moore con un ingresso e una uscita, in grado di riconoscere una qualsiasi delle 3 sequenze (comunque interallacciate) 010, 111, 1011. Quando la macchina riconosce una qualsiasi delle sequenze, pone l'uscita a 1 per 1 ciclo di clock.

ESERCIZIO N°6

4 punti

Disegnare lo schema logico di un full-adder usando esclusivamente half-adder.

1

Dovendo rappresentare solo 5 cifre decimali (in BCD 8421), i calcoli possono essere eseguiti modulo 100000.
Per la valutazione dei numeri nella relativa base conviene ricorrere all'algoritmo di Horner ed eliminare dai risultati tutte le cifre decimali sopra la sesta.

base 8
0
3 3
3 27
3 219
7 1759
7 14079
7 12639 (valore=precedente*8+attuale, scartando le cifre dalla sesta in poi)
BCD: 0001 0010 0110 0011 1001

base 3
0
2 2
1 7
2 23
0 69
2 209
1 628
1 1885
0 5655
1 16966
2 50900 (valore=precedente*3+attuale, scartando le cifre dalla sesta in poi)
BCD: 0101 0000 1001 0000 0000

base 2->16 (si fa prima a passare subito in esadecimale)
0
9 9
10 154
10 2474
2 39586
10 33386 (valore=precedente*16+attuale, scartando le cifre dalla sesta in poi)
BCD: 0011 0011 0011 1000 0110

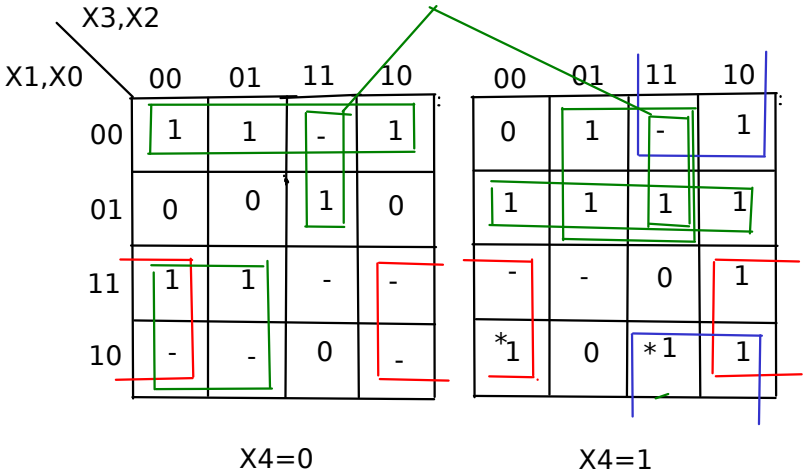
base 16
0
10 10
11 171
11 2747
10 43962
1 3393
10 54298
15 68783
0 528
12 8460
10 35370
12 65932
12 54924
1 78785
10 60570 (valore=precedente*16+attuale, scartando le cifre dalla sesta in poi)
BCD: 0110 0000 0101 0111 0000

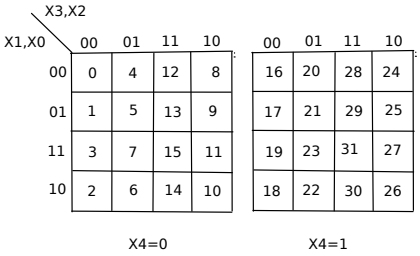
2

```
nseq: //sequenze di 16 valori consecutivi uguali n.i.
  push R17
  push R20
  push R21
  push XL
  push XH
  clr R16 //inizializza risultato
  ldi XL,low(0x2000) //inizializza puntatore
  ldi XH,high(0x2000)
  clr R17 //inizializza contatore di valori
  loop:
    tst R17
    brne newread //se 0 deve ripartire
      ld R20,X+
      inc R17 //primo valore di una nuova sequenza
      rjmp loopend
    newread:
      ld R21,X+ //eventuale prosecuzione sequenza
      cp R20,R21
      brne restart
      inc R17 //valori consecutivi uguali
      cpi R17,16
    brne loopend
    inc R16 //sequenza completa
    clr R17 //riparte da capo
    rjmp loopend
  restart:
  mv R20,R21
  ldi R17,1 //riparte col nuovo valore
  loopend:
  cpi XL,low(0x27FF+1)
  brne loop
  cpi XH,high(0x27FF+1)
  brne loop
  pop R17
  pop R20
  pop R21
  pop XL
  pop XH
  ret
```

3

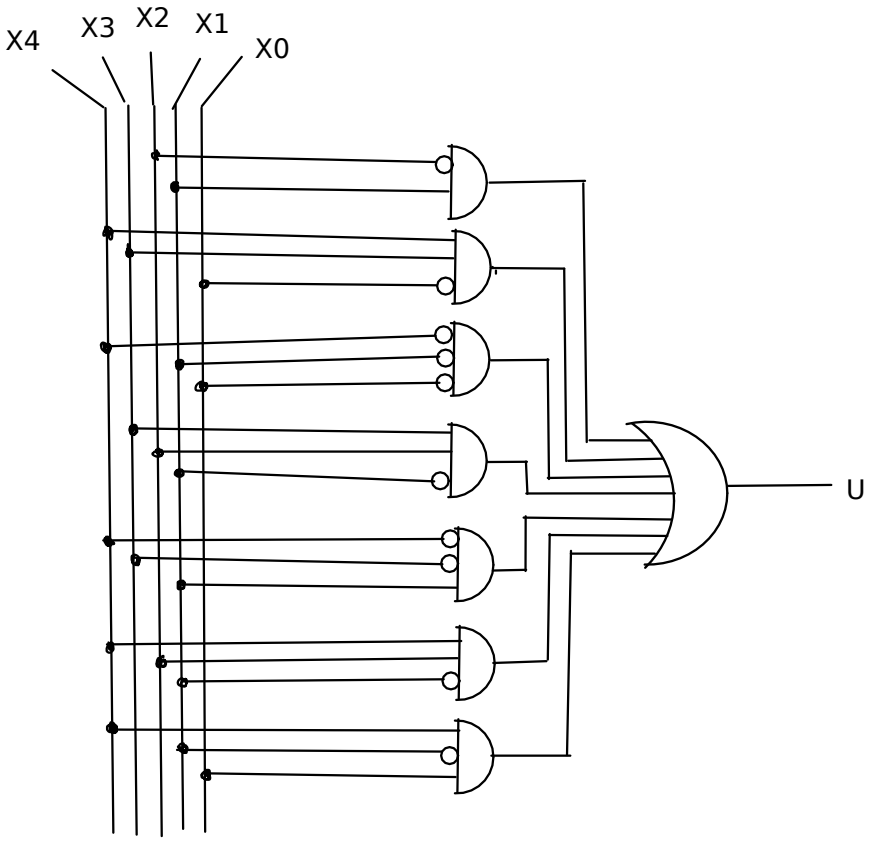
Tabella di verità
1,0,−,1,1,0,−,1,1,0,−,−,−,1,0,−,0,1,1,−,1,1,0,−,1,1,1,1,−,1,1,0





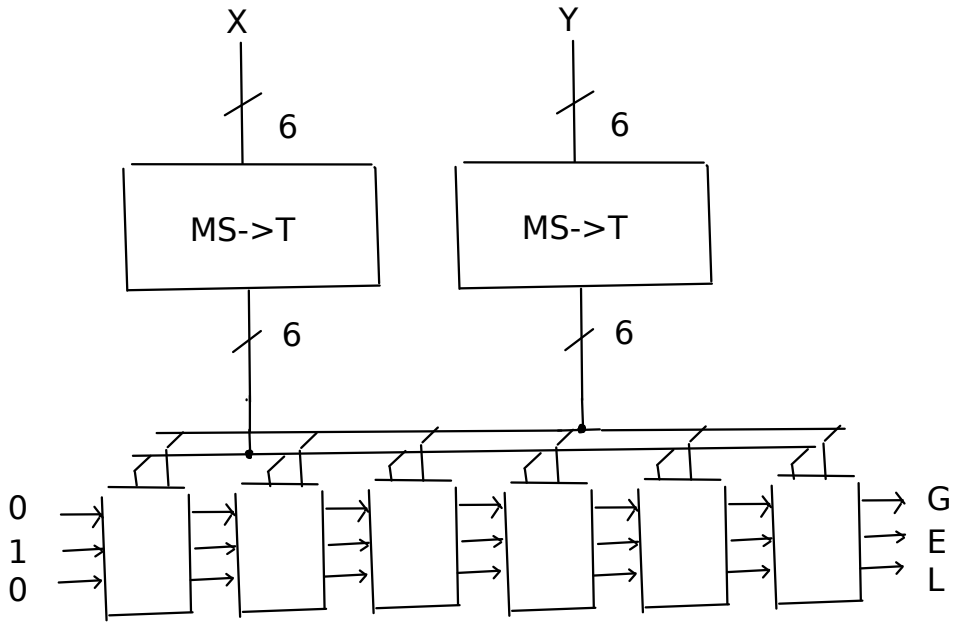
Ci sono 2 implicanti essenziali; gli 1 rimanenti possono essere coperti con 5 implicanti principali (non essenziali) di ordine 2, per un totale di 20 letterali
(quello proposto non è l'unico modo).

e e
U= X2! X1 + X4 X3 X0! + X4! X1! X0! + X3 X2 X1! + X4! X3! X1 + X4 X2 X1! + X4 X1! X0



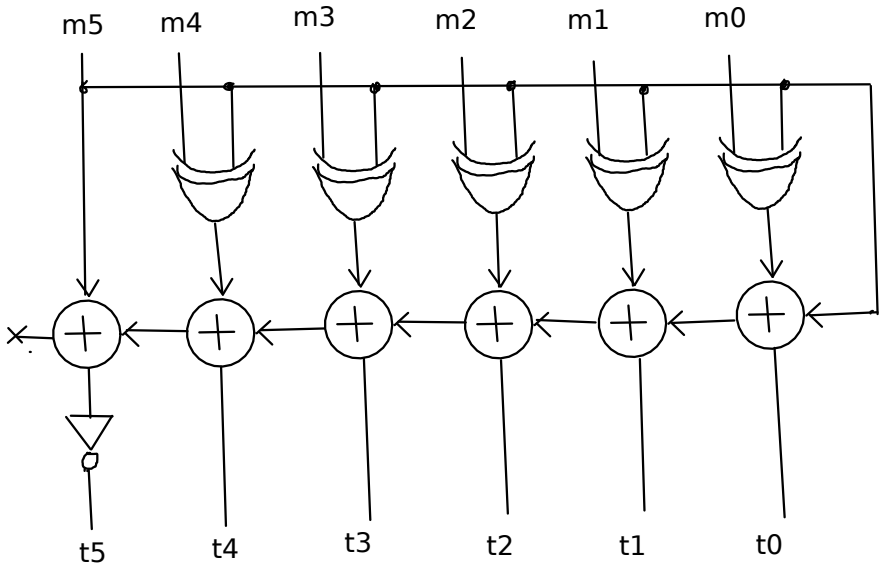
4

Comparatore digitale per numeri in modulo e segno.
Conviene passare in traslazione e usare poi un normale comparatore per interi assoluti.



Comparatore MSb first
Eo=E(xy+x'y')
Go=G+Exy'
Lo=L+Ex'y

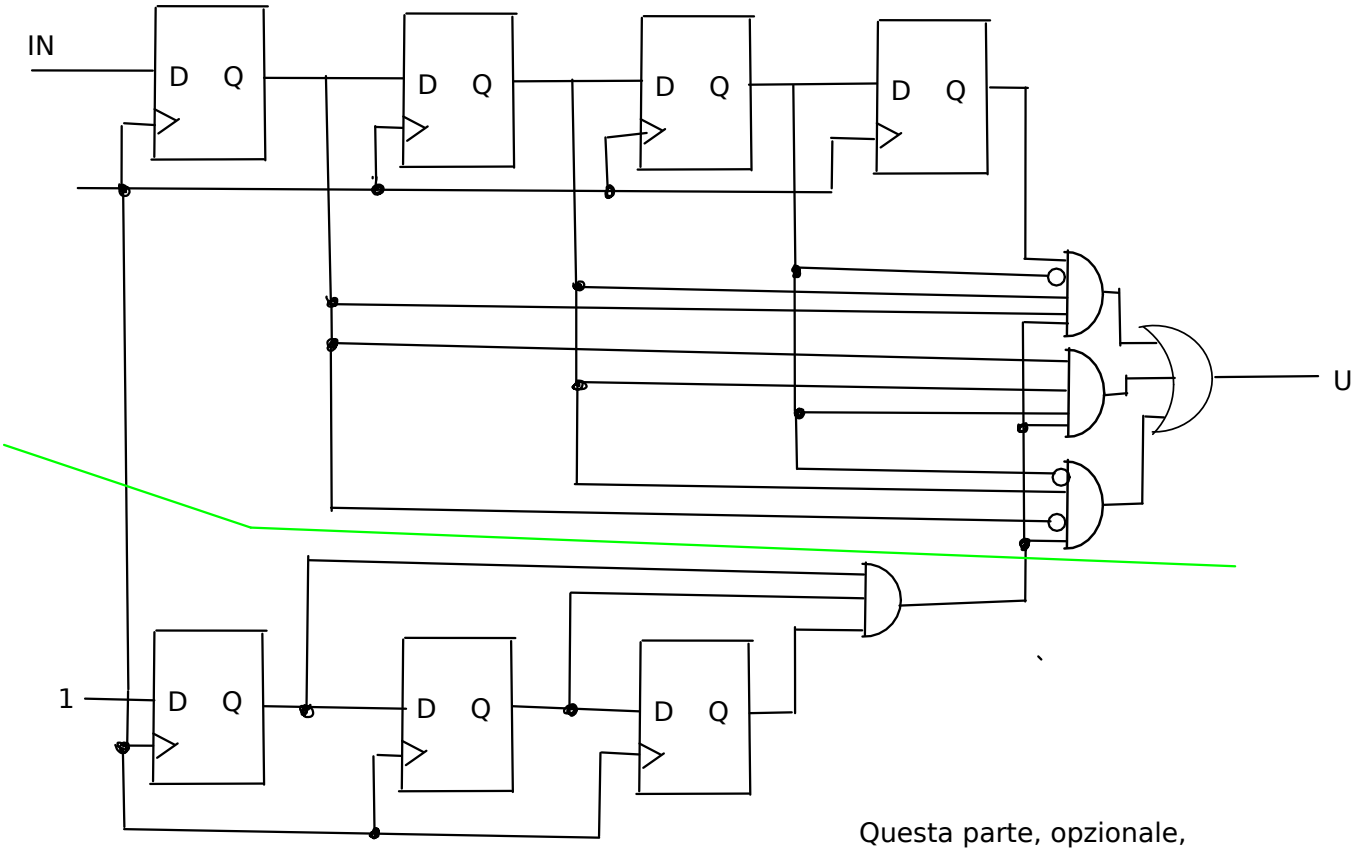
Convertitore MS->T



5

Riconoscitore delle sequenze 010, 111, 1011 (interallacciate)
Soluzione con shift register

D-FF con garanzia dello 0 al power-on



Questa parte, opzionale, impedisce riconoscimenti spuri nei primi 2 cicli di clock. Dal 3° ciclo non ha più alcun effetto.

		U	
0000	000	0	Condizione iniziale
X000	100	0	
XX00	110	0	
XXX0	111	R	sequenze da 3
XXXX	111	R	completo

6

Full-adder fatto con half-adder (sfruttando il fatto che non ci può essere contemporaneamente carry nelle 2 semisomme)