Mappe di Karnaugh

G. MARSELLA
UNIVERSITÀ DEL SALENTO

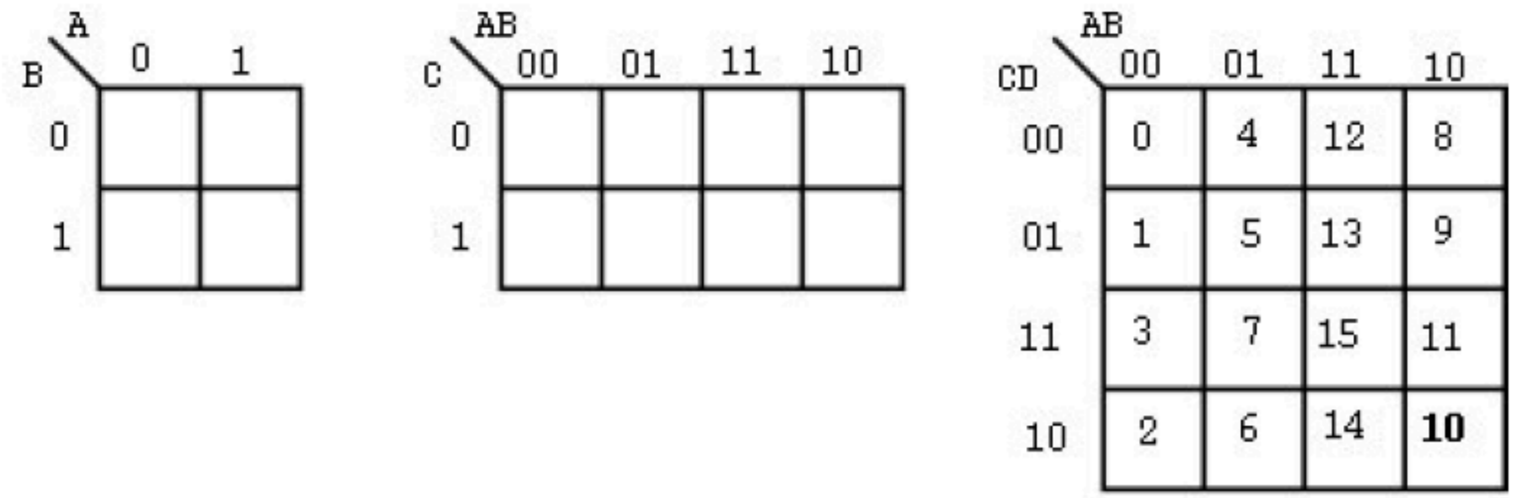
Introduzione

- Le semplificazioni di una funzione logica possono essere effettuate mediante i teoremi dell'algebra di Boole. Esiste però un metodo molto più pratico di semplificazione che quello costituito dalle mappe di Karnaugh.
- Tale metodo di facile applicazione per funzioni di poche variabili, in genere fino ad un massimo di quattro o cinque, risulta alquanto difficoltoso se le variabili diventano numerose.

Mappe di Karnaugh

- Sono riportate le mappe di Karnaugh (di forma quadra o rettangolare) per funzioni di due, tre o quattro variabili:

Figura 1. Mappe di Karnough



Mappe di Karnough

Mappe di Karnaugh

- Le mappe di Karnaugh sono una particolare forma di tabella della verità che consente immediatamente di operare alcune semplificazioni del tipo:

$$A \cdot B + A \cdot \overline{B} = A \cdot (B + \overline{B}) = A$$

$$A \cdot (\overline{B} \cdot \overline{C} + \overline{B} \cdot C + B \cdot C + B \cdot \overline{C}) = A$$

Mappe di Karnaugh

- Ad esempio la seguente tabella di verità della funzione Y = Y(A,B,C)

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

può essere ridisegnata così :

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

Mappa di Karnaugh della funzione **Y**

- Dalla tabella di verità o dalla mappa di Karnaugh è immediato ottenere l'espressione booleana della funzione Y come "somma" di "prodotti", cioè come OR di tanti termini AND quante sono le caselle in cui la funzione vale 1; ciascuno di questi termini AND (detti minterm) è costituito dall'AND delle variabili di ingresso, negate oppure no a seconda che il valore della variabile associato a quella casella sia 0 oppure 1.

$$Y = A \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot C + A \cdot \overline{B} \cdot C$$

Mappe di Karnaugh

- Nel caso di funzioni di 4 variabili, ad es. Z = Z(A,B,C,D), la mappa di Karnaugh ha 4 righe e quattro colonne:

Nelle mappe di Karnaugh i valori della funzione sono scritti dentro le caselle.

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

Mappa di Karnaugh della funzione **Z**

Mappe di Karnaugh

- I valori delle variabili A,B,C,D sono indicati come “coordinate” delle caselle. Esaminando queste “coordinate, si constata che le coppie di valori di A e B (di C e D) associate alle colonne (alle righe) sono ordinate in modo che tra due caselle adiacenti (della medesima riga o della medesima colonna)
- cambia il valore di una sola delle variabili, mentre quello di tutte le altre rimane lo stesso;
- Questa proprietà vale anche tra le caselle estreme di ciascuna riga e di ciascuna colonna (che, sotto questo aspetto, possono quindi essere considerate “adiacenti”, in senso circolare).

Mappe di Karnaugh

- Si osserva che, in virtù di questo fatto, a ciascuna coppia di caselle adiacenti contrassegnate con il valore 1 corrispondono, nella espressione booleana, due termini “prodotto” (minterm) nei quali una variabile è presente negata in uno e non negata nell’altro, mentre tutte le altre variabili hanno lo stesso valore.
- E` allora possibile semplificare l’espressione sostituendo quei due termini con un unico termine nel quale non è più presente la variabile che cambia valore.
- Ad esempio le ultime due caselle della seconda riga nella mappa della funzione Y portano alla seguente semplificazione:

$$A \cdot B \cdot C + A \cdot \overline{B} \cdot C = A \cdot C$$

Mappe di Karnaugh

- Allo stesso modo, quaterne di caselle adiacenti tutte con il valore 1 (sulla stessa riga o sulla stessa colonna) corrispondono a quattro termini che si riducono ad uno;
- ad esempio le quattro caselle della terza riga nella mappa della funzione Z portano alla seguente semplificazione:

$$C \cdot D \cdot (\overline{A} \cdot \overline{B} + \overline{A} \cdot B + A \cdot B + A \cdot \overline{B}) = C \cdot D$$

- le quattro caselle della terza colonna nella mappa della funzione Z portano alla seguente semplificazione:

$$A \cdot B \cdot (\overline{C} \cdot \overline{D} + \overline{C} \cdot D + C \cdot D + C \cdot \overline{D}) = A \cdot B$$

Mappe di Karnaugh

- Così pure quaterne adiacenti disposte secondo un quadrato producono un unico termine;
- Ad esempio le quattro caselle in basso a sinistra nella mappa della funzione Z portano alla seguente semplificazione:

$$\overline{A}\bullet C\bullet(\overline{B}\bullet\overline{D} + \overline{B}\bullet D + B\bullet D + B\bullet\overline{D}) = \overline{A}\bullet C$$

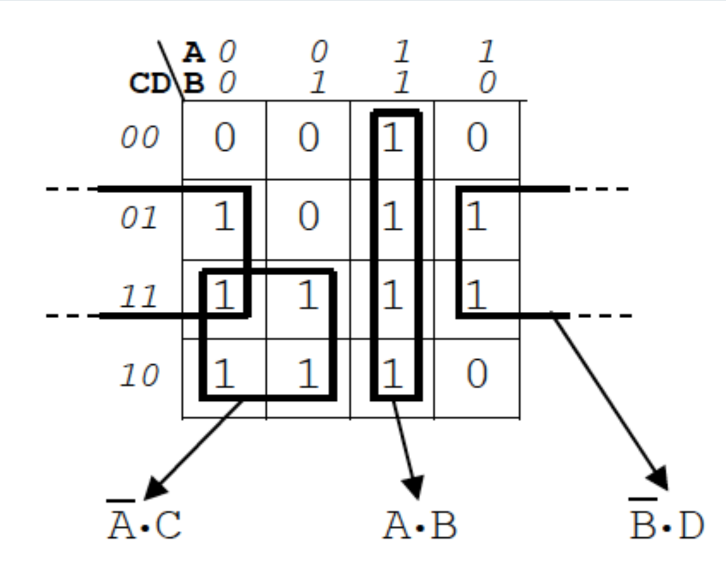
- Analogo discorso vale per gruppi di otto caselle adiacenti tutte con il valore 1.

Mappe di Karnaugh

- Per semplificare l'espressione booleana di una funzione, si tratta dunque di individuare, nella relativa mappa di Karnaugh, i gruppi di (2 o 4 o 8) caselle adiacenti con il valore 1.
- Nel far ciò conviene tenere presente la proprietà A+A=A, che consente di utilizzare più volte la stessa casella (ovvero più volte lo stesso minterm nell'espressione booleana), per formare gruppi diversi, al fine di operare il maggior numero di semplificazioni possibile.
- Individuando un insieme di gruppi (da 1, 2, 4 o 8) che copre tutte le caselle in cui compare il valore 1, si ottiene una espressione semplificata, costituita dall'OR dei termini corrispondenti a ciascun gruppo.

Mappe di Karnaugh

- Riprendendo l'esempio della funzione Z, si possono individuare i gruppi segnati in figura:



- Con questi raggruppamenti si ottiene, immediatamente, l'espressione semplificata di Z:

$$Z = \overline{A}\cdot C + A\cdot B + \overline{B}\cdot D$$

Mappe di Karnaugh

- Nell'esempio si può osservare che si sono considerate adiacenti anche le caselle estreme delle righe o delle colonne.
- Si osserva che si possono individuare diversi raggruppamenti che coprono tutte le caselle in cui Z vale 1, ciascuno dei quali porta a diverse espressioni di Z equivalenti (più o meno semplificate).

Funzioni parzialmente definite

- Una funzione booleana si dice parzialmente definita se il suo valore è specificato solo per alcune combinazioni dei valori delle variabili.
- Nella pratica si ha a che fare con funzioni booleane parzialmente definite in due casi:
  - quando le altre combinazioni dei valori delle variabili non si possono verificare mai;
  - quando, anche se si verificano, i corrispondenti valori della funzione non importano (possono essere indifferentemente 0 od 1, perché comunque non vengono usati).
- Nella tabella di verità (o nella mappa di Karnaugh) di una funzione parzialmente definita, i valori non specificati sono comunemente indicati con un trattino e corrispondono a ciò che si chiama "condizioni di indifferenza", ovvero don't care conditions (d.c.c.).
- La presenza delle d.c.c. nelle caselle di una mappa di Karnaugh può essere convenientemente sfruttata, sostituendone alcune con il valore 1, al fine di ottenere gruppi (da 2, 4, 8) che portano a semplificare l'espressione della funzione.

Funzioni parzialmente definite

- Ad esempio, considerando la funzione parzialmente definita W la cui tabella di verità è riportata qui sotto insieme con la relativa mappa di Karnaugh:

A	B	C	W
0	0	0	–
0	0	1	1
0	1	0	–
0	1	1	–
1	0	0	1
1	0	1	–
1	1	0	–
1	1	1	0

A	*0*	*0*	*1*	*1*
B	*0*	*1*	*1*	*0*
C *0*	–	–	–	1
1	1	–	0	–

Si possono sostituire due *d.c.c.* con altrettanti 1:

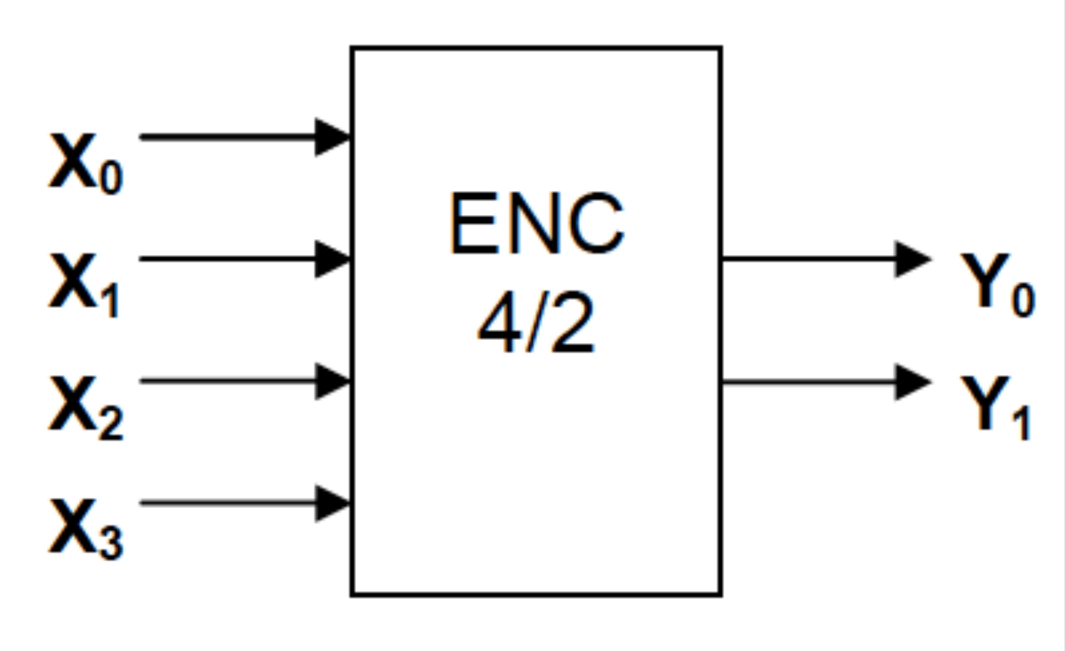
A	*0*	*0*	*1*	*1*
B	*0*	*1*	*1*	*0*
C *0*	1	–	–	1
1	1	–	0	1

- e individuare la quaterna che consente di ottenere la seguente espressione semplificata di W:

$$W = \overline{B}$$

Sintesi di un encoder

- Si ricorda che il funzionamento di un encoder è basato sull'ipotesi che, in ogni istante, una e una sola delle variabili di ingresso abbia il valore 1.
- Si consideri il caso dell'encoder con 4 ingressi e due uscite:



Sintesi di un encoder

- Delle 16 righe della tabella di verità sono significative solo le 4 nelle quali Y_0 ed Y_1 sono definite:

X_0	X_1	X_2	X_3	Y_1	Y_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1
.	.	.	.	-	-

- La corrispondente mappa di Karnaugh per la funzione Y_0 è:

$x_2\ x_3$ \ $x_0\ x_1$	0 0	0 1	1 1	1 0
0 0	–	1	–	0
0 1	1	–	–	–
1 1	–	–	–	–
1 0	0	–	–	–

Sintesi di un encoder

- Sfruttando le condizioni di indifferenza (d.c.c.) presenti in questa mappa, si possono disegnare i due raggruppamenti da 8 caselle indicati in figura:

$x_2\ x_3$ \ $x_0\ x_1$	0 0	0 1	1 1	1 0
0 0	–	1	–	0
0 1	1	–	–	–
1 1	–	–	–	–
1 0	0	–	–	–

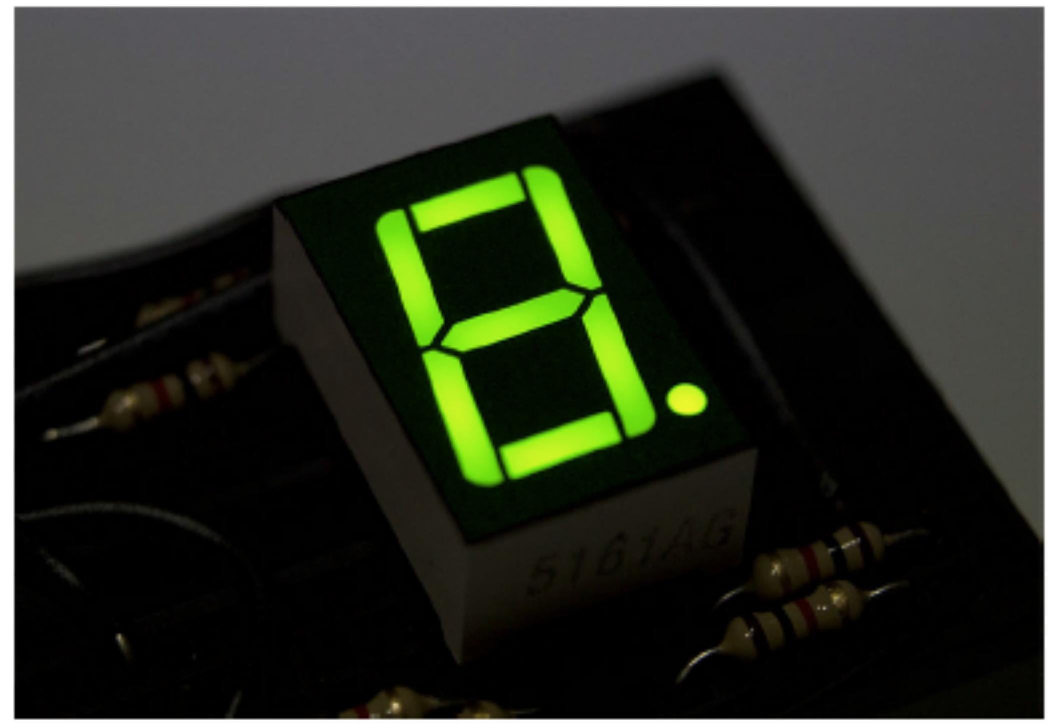
- E ottenere le espressioni semplificate:

$$Y_0 = X_1 + X_3$$

$$Y_1 = X_2 + X_3$$

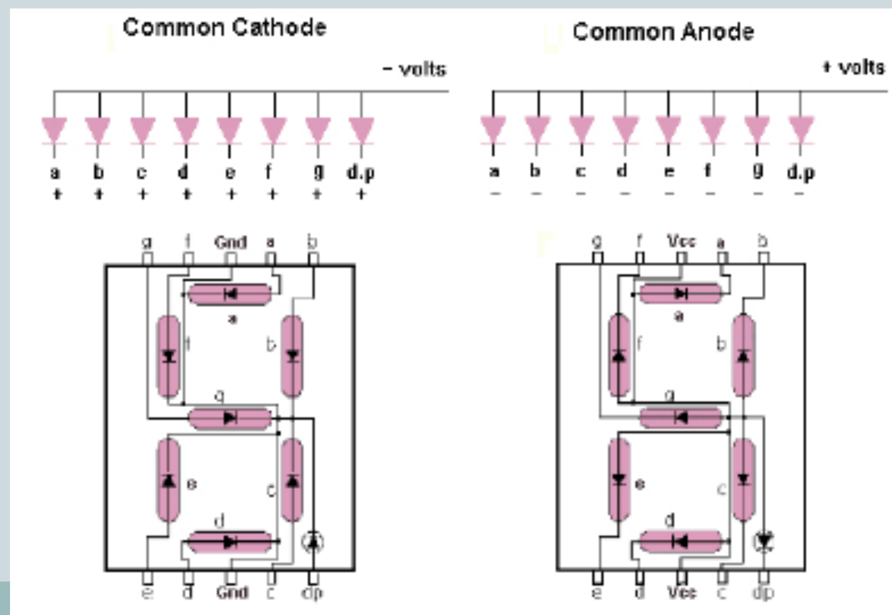
Display decoder a 7 segmenti con le mappe di Karnaugh

- Per un singolo modulo, è una rappresentazione di interi da 0 a 9 (e eventualmente di alcuni caratteri non numerici).

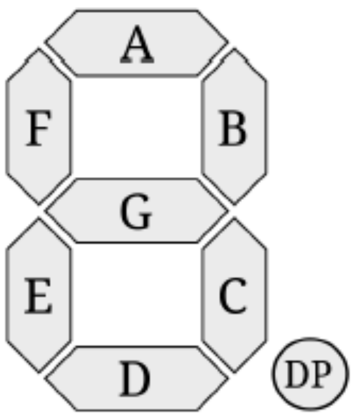


Display decoder a 7 segmenti con le mappe di Karnaugh

- Catodo comune: i catodi dei LED sono connessi alla massa del circuito. La corrente passa se agli anodi dei LED . impostato il valore logico 1 (pilotaggio degli anodi).
- Anodo comune: l'anodo . comune a tutti i LED e viene alimentato a tensione costante (Vcc). Il catodo . impostato a 0 per l'accensione del singolo LED (pilotaggio del catodo) .



Analisi del Problema



– Di quanti bit ho bisogno per rappresentare caratteri numerici da 0 a 9 (ovvero 10 valori)?

3 bit → 2^3 troppo pochi

4 bit → 2^4 è un numero adeguato

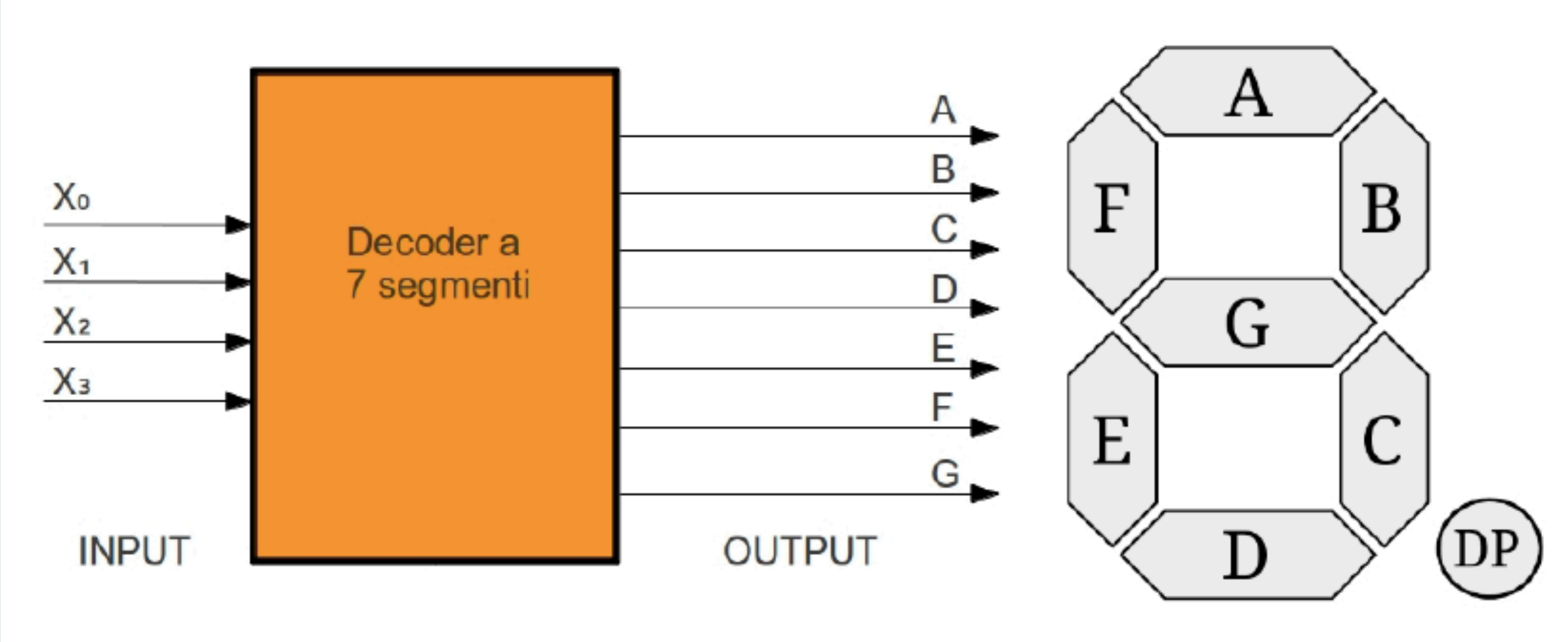
– Posso nominare ogni singolo LED e controllarlo separatamente attraverso un'opportuna funzione logica.

– Necessito di un dispositivo che realizzi ogni singola funzione logica, gestisca i bit in entrata e l'accensione dei LED (output): impiego di un **decoder**.

– Per la scrittura e minimizzazione delle funzioni logiche ricorro invece alle **mappe di Karnaugh**.

Decoder

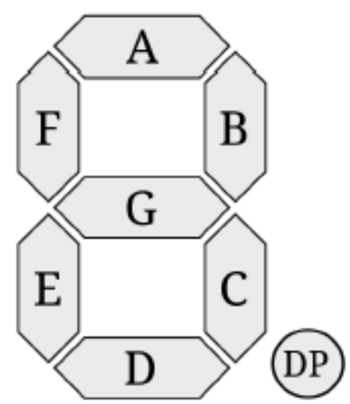
- E’ un circuito combinatorio che decodifica le informazioni di n input in entrata fornendo fino a un massimo di 2^n output.
- Nel caso in esame vi sono 4 bit di ingresso e 7 in uscita.



Mappa di Karnaugh

- Indicando con X = [X_0,X_1,X_2,X_3] l'input e S = [A, B, C, D, E, F, G] l'output, è possibile esprimere la relazione tra queste variabili attraverso la sottostante tabella della verità:

X_0	X_1	X_2	X_3	n	A	B	C	D	E	F	G
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
Other				d	d	d	d	d	d	d	d



Mappe di Karnaugh

- Considerando una disposizione $X_{2,3} \backslash X_{0,1}$

A	00	01	11	10
00	1	0	d	1
01	0	1	d	1
11	1	1	d	d
10	1	1	d	d

B	00	01	11	10
00	1	1	d	1
01	1	0	d	1
11	1	1	d	d
10	1	0	d	d

C	00	01	11	10
00	1	1	d	1
01	1	1	d	1
11	1	1	d	d
10	0	1	d	d

D	00	01	11	10
00	1	0	d	1
01	0	1	d	1
11	1	0	d	d
10	1	1	d	d

E	00	01	11	10
00	1	0	d	1
01	0	0	d	0
11	0	0	d	d
10	1	1	d	d

F	00	01	11	10
00	1	1	d	1
01	0	1	d	1
11	0	0	d	d
10	0	1	d	d

G	00	01	11	10
00	0	1	d	1
01	0	1	d	1
11	1	0	d	d
10	1	1	d	d

Mappe di Karnaugh

- Ponendo d = 0 (minimo rischio)

A	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

B	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	1	0	0
10	1	0	0	0

C	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	0
10	0	1	0	0

Funzioni logiche corrispondenti
$A = X_1' \cdot X_2' \cdot X_3' + X_0 \cdot X_1' \cdot X_2' + X_0' \cdot X_1 \cdot X_3 + X_0' \cdot X_2$
$B = X_0' \cdot X_2' \cdot X_3' + X_1' \cdot X_2' + X_0' \cdot X_1' + X_0' \cdot X_2 \cdot X_3$
$C = X_1' \cdot X_2' + X_0' \cdot X_1 + X_0' \cdot X_3$

Mappe di Karnaugh

D	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	0	0	0
10	1	1	0	0

E	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	0	0	0
10	1	1	0	0

F	00	01	11	10
00	1	1	0	1
01	0	1	0	1
11	0	0	0	0
10	0	1	0	0

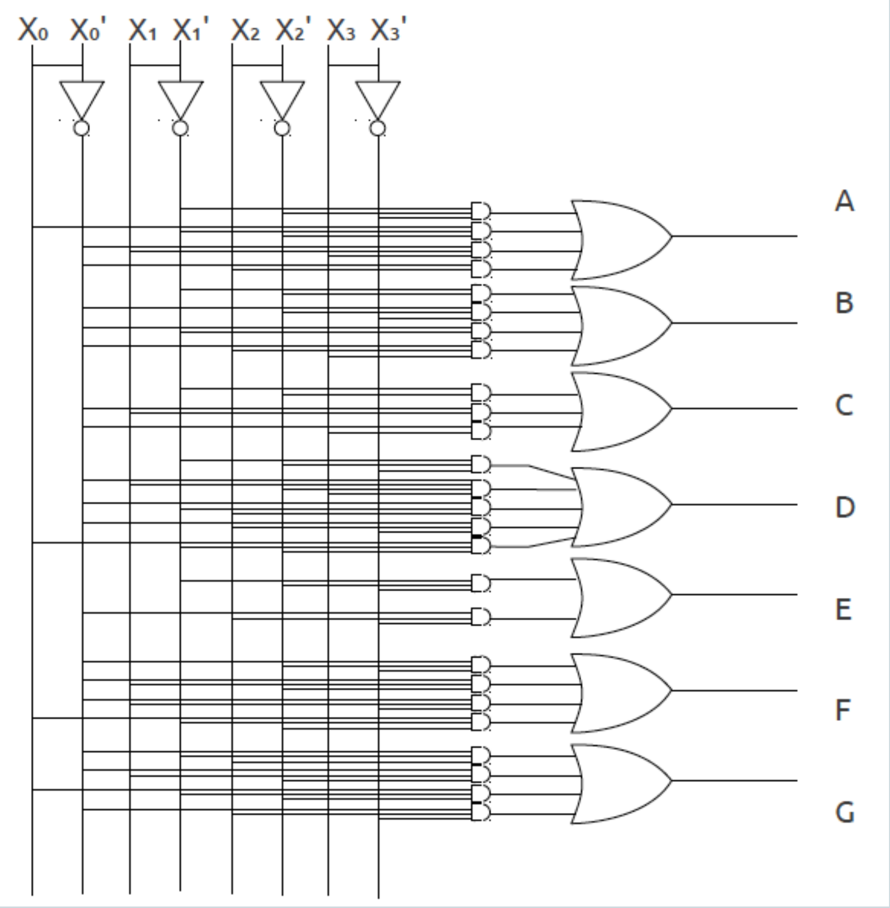
Funzioni logiche corrispondenti
$D = X_1' \cdot X_2' \cdot X_3' + X_0 \cdot X_1' \cdot X_2' + X_0' \cdot X_1 \cdot X_2' \cdot X_3 + X_0' \cdot X_1' \cdot X_2 + X_0' \cdot X_2 \cdot X_3'$
$E = X_1' \cdot X_2' \cdot X_3' + X_0' \cdot X_2 \cdot X_3'$
$F = X_0' \cdot X_2' \cdot X_3' + X_0' \cdot X_1 \cdot X_2' \; X_0 \cdot X_1' \cdot X_2' + X_0' \cdot X_1 \cdot X_3'$

Mappe di Karnaugh

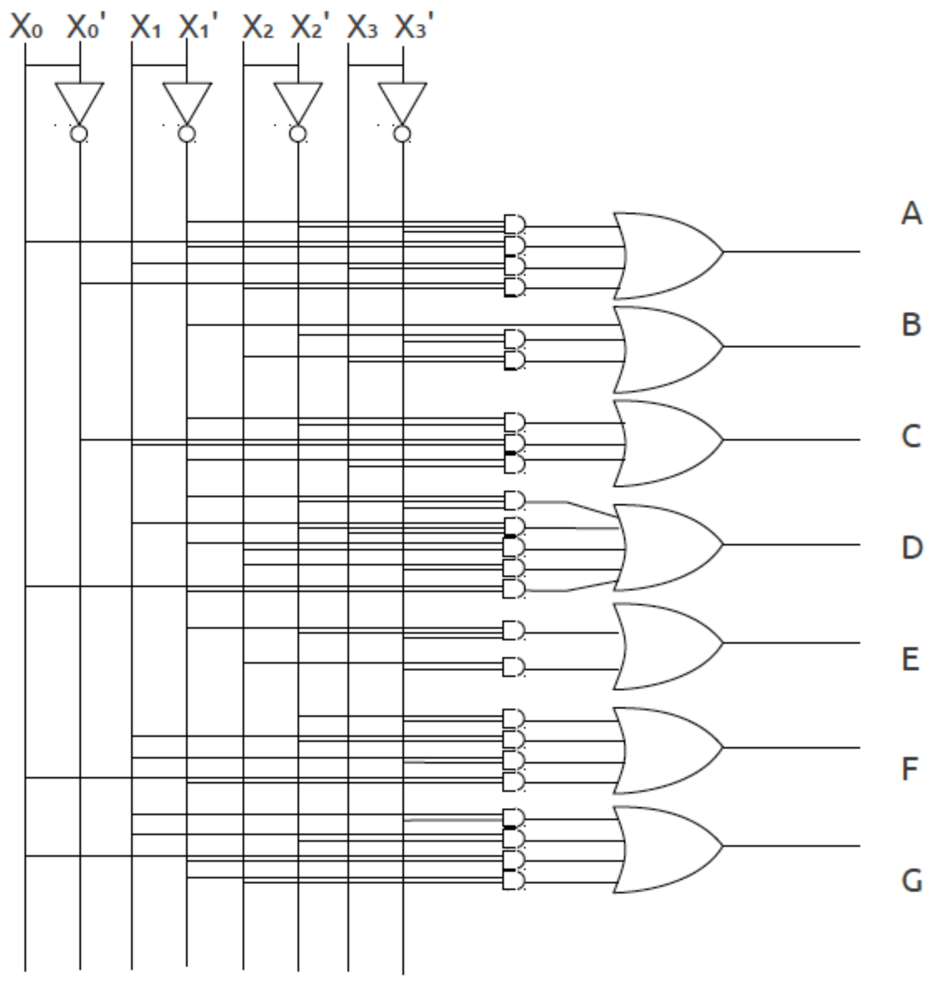
G	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	1	0	0	0
10	1	1	0	0

Funzione logica corrispondente
$G = X_0' \cdot X_1 \cdot X_2' + X_0 \cdot X_1' \cdot X_2' + X_0' \cdot X_1' \cdot X_2 + X_0' \cdot X_2 \cdot X_3'$

Circuito corrispondente



Circuito corrispondente



Ponendo d = 1 (minimo costo)

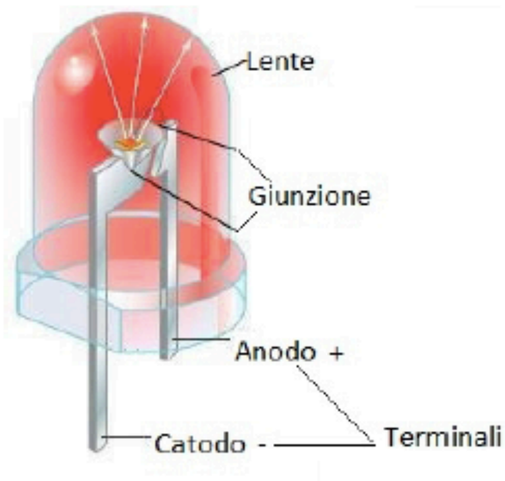
Differenze

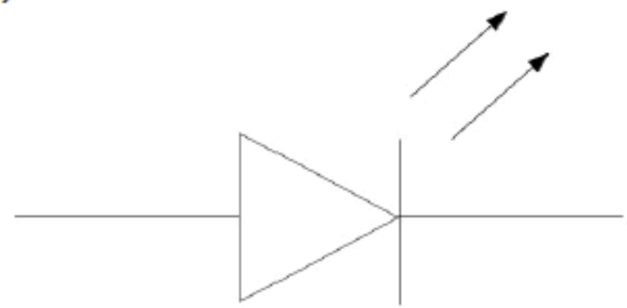
	d = 0	d = 1
Porte OR	7	7
5 ingressi	1	1
4 ingressi	4	3
3 ingressi	1	2
2 ingressi	1	1
Porte AND	26	24
4 ingressi	1	
3 ingressi	19	4
2 ingressi	6	20

- Il carico sulle porte logiche di d = 1 . inferiore rispetto a quello per d = 0.

Implementazione fisica

- LED (Light Emitting Diode)





È un diodo a giunzione p-n, formato da uno strato semiconduttore. Il colore del LED dipende da quest' ultimo.

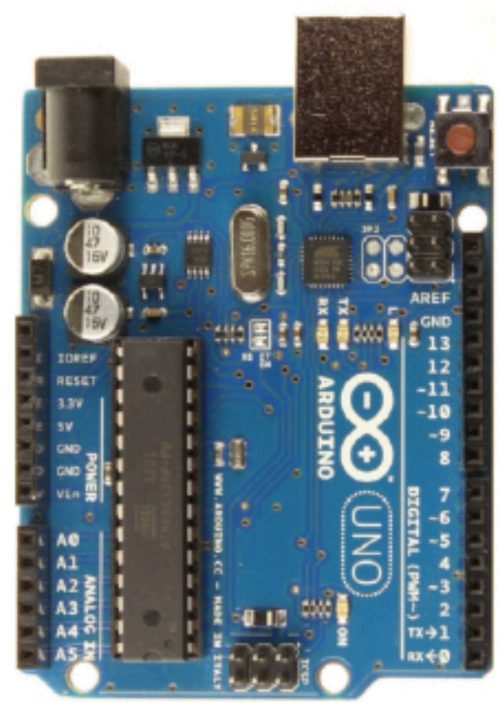
Fornendo una tensione di alimentazione (1,8 V), gli elettroni della banda di conduzione si ricombinano con le lacune della banda di valenza con il risultato dell'emissione di un fotone.

Poiché il controllo del mio dispositivo avviene attraverso il pilotaggio diretto degli anodi, il modulo implementato è a **catodo comune**.

Implementazione fisica

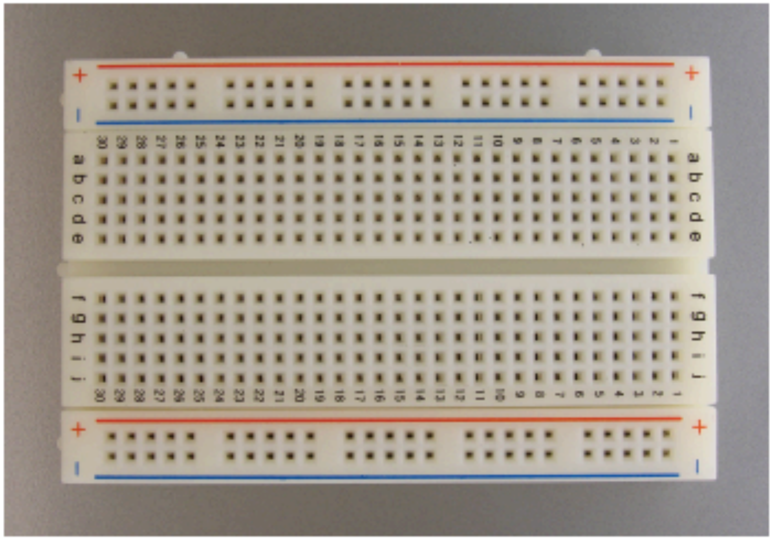
- Per la gestione dei LED ho usato una scheda di prototipazione rapida, l'Arduino UNO

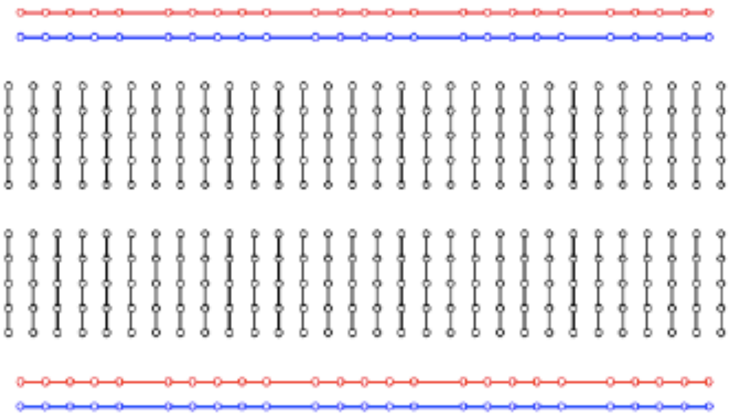
  - Microcontroller ATmega328
  - Tensione di lavoro 5 V
  - Pin digitali I/O: 14
  - Pin analogici: 6
  - DC per singolo Pin I/O: 40 mA
  - Flash memory: 32 KB
  - EEPROM: 1 KB
  - Clock: 16 MHz



Implementazione fisica

- Breadboard

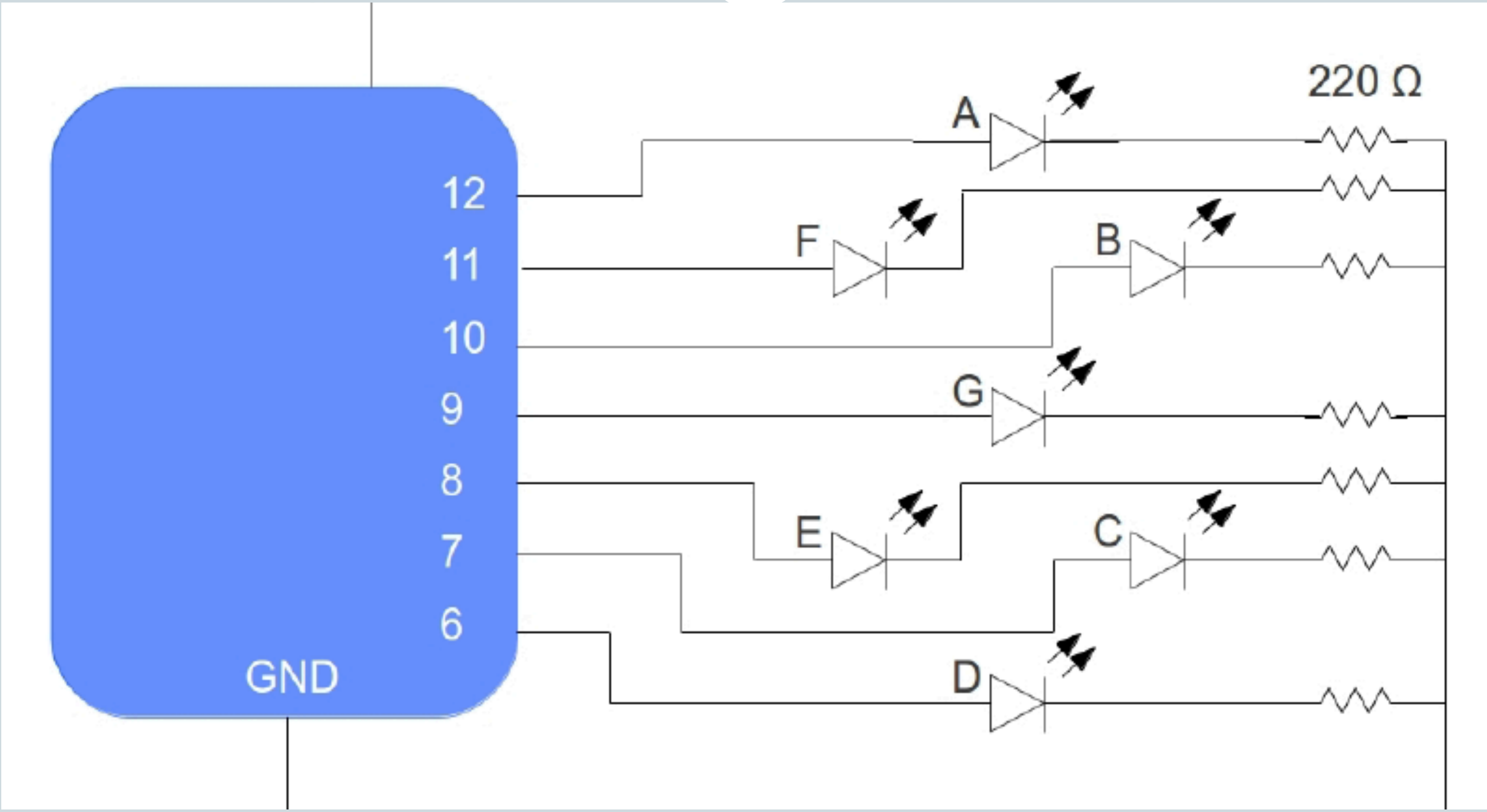




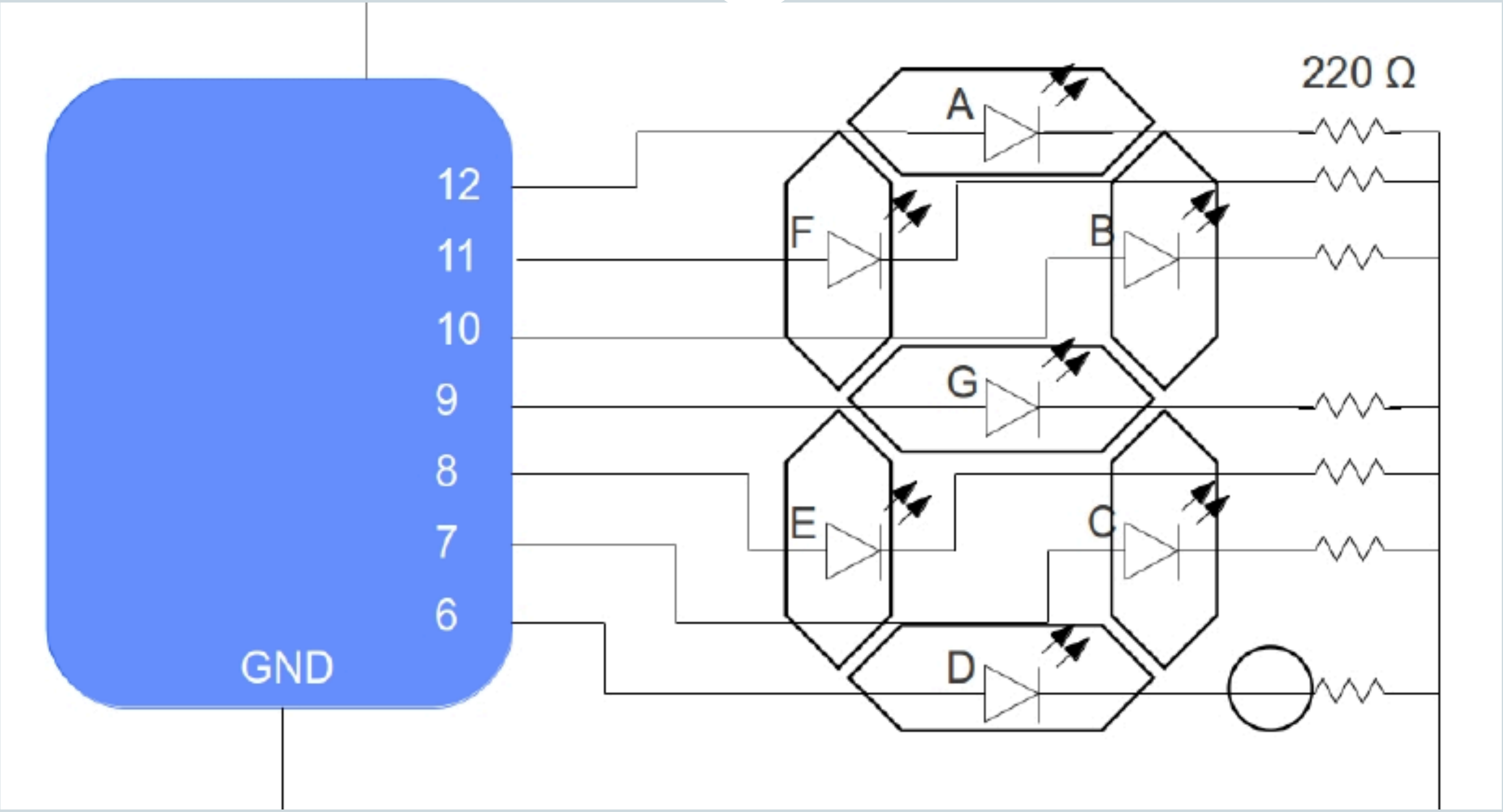
- Resistenze da 220 Ω (± 1%)

- Fili metallici per il trasporto dell'informazione

Implementazione fisica



Implementazione fisica



Implementazione fisica


```
//inizializzo le variabili di
stato

int k = 0;
int x0 = 0;
int x1 = 0;
int x2 = 0;
int x3 = 0;

void setup() {

  Serial.begin(9600);

  //definisco i pin di output, i
led
  pinMode(12,OUTPUT);
  pinMode(11,OUTPUT);
  pinMode(10,OUTPUT);
  pinMode(9,OUTPUT);
  pinMode(8,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(6,OUTPUT);
}

void loop() {

if(Serial.available()>0){
   k = Serial.read();
   }

 switch (k) {

    case '0':
    x0 = 0;
    x1 = 0;
    x2 = 0;
    x3 = 0;
    break;
```

```
  case '1':
    x0 = 0;
    x1 = 0;
    x2 = 0;
    x3 = 1;
    break;

    case '2':
    x0 = 0;
    x1 = 0;
    x2 = 1;
    x3 = 0;
    break;

   case '3':
    x0 = 0;
    x1 = 0;
    x2 = 1;
    x3 = 1;
    break;

    case '4':
    x0 = 0;
    x1 = 1;
    x2 = 0;
    x3 = 0;
    break;

    case '5':
    x0 = 0;
    x1 = 1;
    x2 = 0;
    x3 = 1;
    break;
```

```
    case '6':
    x0 = 0;
    x1 = 1;
    x2 = 1;
    x3 = 0;
    break;

    case '7':
    x0 = 0;
    x1 = 1;
    x2 = 1;
    x3 = 1;
    break;

    case '8':
    x0 = 1;
    x1 = 0;
    x2 = 0;
    x3 = 0;
    break;

   case '9':
    x0 = 1;
    x1 = 0;
    x2 = 0;
    x3 = 1;
    break;

   }
```

```
if ((x1 == 0 && x2 == 0 && x3 == 0) || (x0 == 1 && x1 == 0) || (x1 == 1 && x3 == 1) || (x0 == 0 && x2
== 1)) {
  digitalWrite(12, HIGH);
  Serial.println("a");
  digitalWrite(12, LOW);
  }
if (x1 == 0 || (x2 == 0 && x3 == 0) || (x2 == 1 && x3 == 1)) {
  digitalWrite(10, HIGH);
  Serial.println("b");
  digitalWrite(10, LOW);
  }
if((x1 == 0 && x2 == 0) || (x1 == 0 && x3 == 1) || (x0 == 0 && x1 == 1)) {
  digitalWrite(7, HIGH);
  Serial.println("c");
  digitalWrite(7, LOW);
}
if((x1 == 0 && x2 == 0 && x3 == 0) || (x1 == 1 && x2 == 0 && x3 == 1) || (x1 == 0 && x2 == 1) || (x2
== 1 && x3 == 0) || (x0 == 1 && x1 == 0)) {
  digitalWrite(6, HIGH);
  Serial.println("d");
  digitalWrite(6, LOW);
}
if((x1 == 0 && x2 == 0 && x3 == 0) || (x2 == 1 && x3 == 0)) {
  digitalWrite(8, HIGH);
  Serial.println("e");
  digitalWrite(8, LOW);
}
if((x2 == 0 && x3 == 0) || (x1 == 1 && x2 == 0) || (x1 == 1 && x3 == 0) || (x0 == 1 && x1 == 0)) {
  digitalWrite(11, HIGH);
  Serial.println("f");
  digitalWrite(11, LOW);
}
  if((x1 == 1 && x3 == 0) || (x1 == 1 && x2 == 0) || (x0 == 1 && x1 == 0) || (x1 == 0 && x2 == 1)) {
  digitalWrite(9, HIGH);
  Serial.println("g");
  digitalWrite(9, LOW);
}

}
```

Considerazioni finali

- E’ modulabile? Esistono due vie: singoli decoder separati a 4 bit o un unico decoder (ma con un maggior numero di bit dipendente dal numero che si vuole rappresentare).
- Come si può comandare il LED della virgola? Linea dedicata attivabile quando il modulo successivo, dei decimali, è attivo.
- Rappresentazione di caratteri alfanumerici: alfabeto latino moderno (26 grafi) + 10 numeri. Uso di un modulo a 14 segmenti a 6 bit.

Riferimenti

- http://www.micro-digital.net/8051-to-7-segment-display-interfacing/
- http://www.ecs.umass.edu/ece/engin112/lectures/Engin112-F13-L21-modular-logic.pdf
- http://ecee.colorado.edu/~mcleod/pdfs/IADE/lectures/ECEN 1400 Lecture 15 Seven Segment Display.pdf
- http://www.arduino.cc/
- http://gorgeous-karnaugh.com/tutorials/practical-usage/7-segment-led-display.html
- http://www.elemania.altervista.org/digitale/combinatorio/comb3.html
- https://www.youtube.com/watch?v=k9lDX9_MUUM

(VHDL su Youtube)