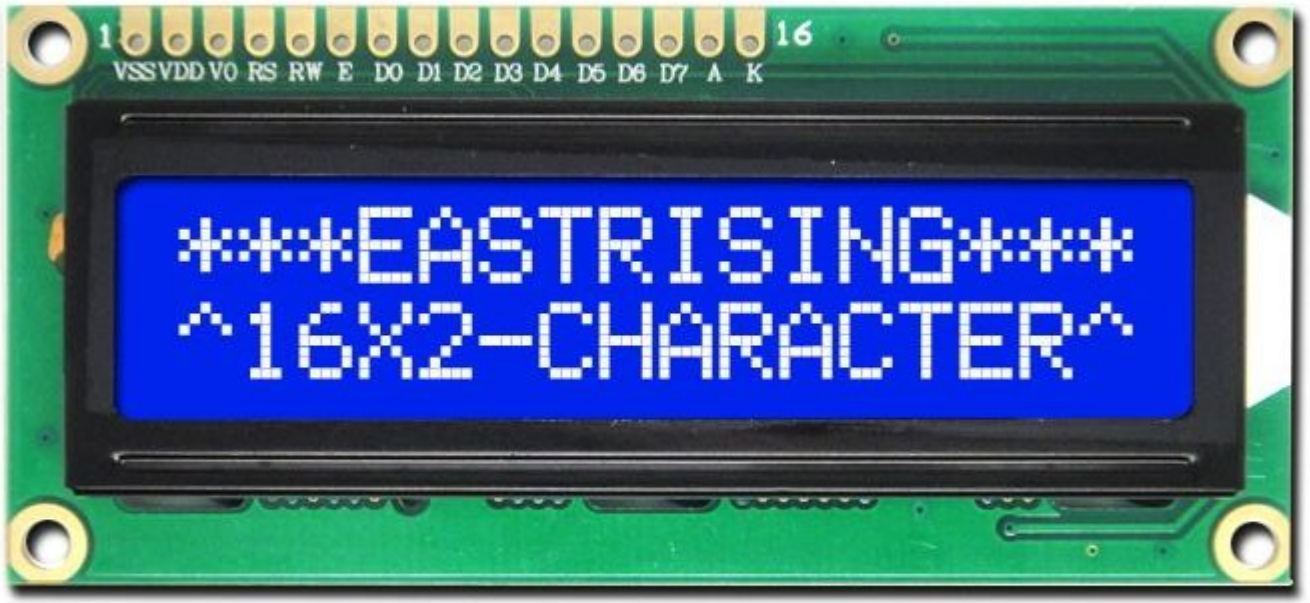
Embedded System

L7

LCD Display



Display LCD a caratteri alfanumerici

Normalmente lo troviamo in formati:

8x1, 8x2, 16x2, 16x4, 20x4

Sono display dotati di controller integrati e memorie già contenenti i caratteri.

Vengono definiti come “display intelligenti”

I display più comuni sono quelli basati sull’arcinoto controller HD44780 prodotto dalla Hitachi,

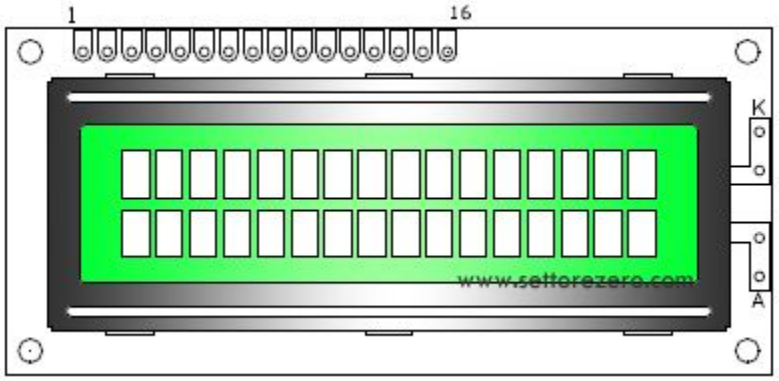
Esistono comunque in giro numerosi display aventi controller prodotti da altre marche che sono compatibili con quello della Hitachi

Sono costruiti con varie caratteristiche “esterne”, per cui in fase di acquisto possiamo scegliere il numero di righe e il numero di caratteri che il display è in grado di visualizzare

altra caratteristica da scegliere è la colorazione della retroilluminazione

Funzionamento dei display basati su controller HD44780

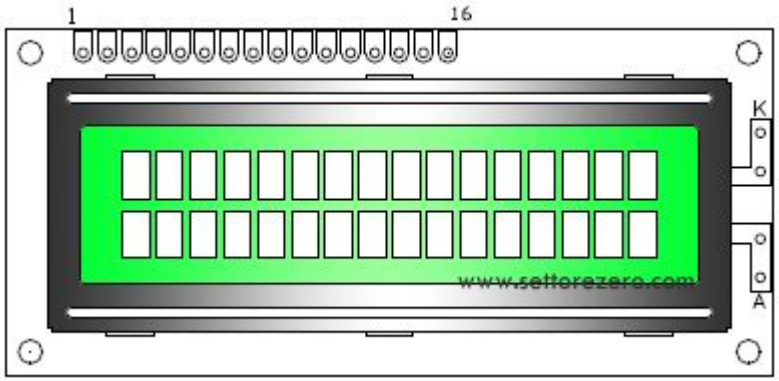
Vediamo innanzitutto come si presenta esternamente un classico display lcd 16×2:



Partiamo innanzitutto col dire che non tutti i display presentano la piedinatura disposta in questo modo. Questa è la più comune, ma alcuni hanno due file da 8 contatti disposte sulla sinistra del display

La raccomandazione quindi è sempre la stessa: scaricate il datasheet del display che avete sottomano per evitare di compiere errori fatali che possono portare al guasto del display

Di questi 16 pin, 14 sono per il funzionamento del display e i rimanenti 2 servono per la retroilluminazione



Di questi 16 pin, 14 sono per il funzionamento del display e i rimanenti 2 servono per la retroilluminazione.

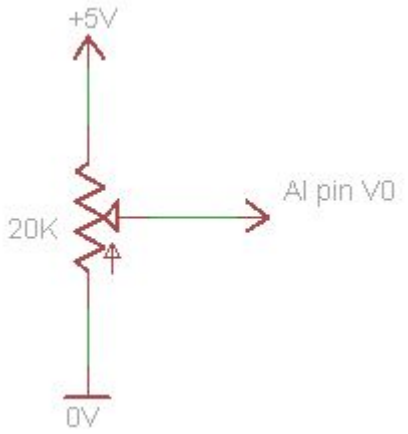
Generalmente i due pin destinati alla retroilluminazione sono gli ultimi 2: il 15 è collegato all’anodo, e richiede 5V, e il 16 è il catodo, e quindi va a massa

La piedinatura “classica”

Pin	Nome	Descr.
1	Vss o Gnd	Massa
2	Vcc	+5V
3	Vo	Contrasto
4	RS	Register Selector
5	R/W	Read/Write
6	E	Enable
7	D0	Dato 0
8	D1	Dato 1

Pin	Nome	Descr.
9	D2	Dato 2
10	D3	Dato 3
11	D4	Dato 4
12	D5	Dato 5
13	D6	Dato 6
14	D7	Dato 7
15	A	Anodo
16	K	Catodo

3 – V0 : regolazione contrasto, su tale pin deve giungere un segnale compreso tra 0 e 5 Volt per regolare il contrasto: ovvero per rendere più o meno scuri i pixel



4 – RS : Register Select. Questo pin serve per indicare al display che tipo di informazione stiamo inviando sulla linea dati: se RS viene posto a 1 (livello logico alto) vuol dire che, tramite la linea dati, stiamo inviando un carattere da visualizzare sul display, se invece poniamo RS a livello logico basso, vuol dire che il dato che stiamo inviando è un comando da eseguire (es.: cancellazione display, seleziona una riga in cui scrivere ecc.)

5 – R/W : Read/Write. Se posto a 0, vogliamo scrivere un dato, se posto a 1 vogliamo leggere un dato (il display può difatti funzionare in modalità lettura/scrittura: normalmente viene sempre utilizzato in modalità scrittura, ma se vogliamo realizzare effetti particolari o sfruttare la memoria dell'LCD, allora è necessario utilizzarlo anche in lettura).

6 – E : Enable. E' in pratica il pin sul quale viene inviato il segnale di sincronismo: quando viene posto a 1, il controller esegue la lettura del dato sulla linea dati, lo interpreta e quindi scrive sul display (o esegue il comando): quando dovremo inviare un carattere da visualizzare, dovremo prima predisporre opportunamente la linea dati, quindi porteremo il pin E a livello logico alto, in maniera tale che il controller esegua la lettura del dato, dopodichè riporteremo E a livello logico basso. E' fondamentalmente il clock

7 ÷ 14 – D0 ÷ D7 : Linea dati. Sulla linea dati (8 bit) vengono inviati i comandi da eseguire o il codice del carattere da visualizzare (oppure servono a ricevere i dati quando il display viene utilizzato anche in lettura). Questi display, fortunatamente, possono essere utilizzati anche in modalità 4 bit: ovvero utilizzeremo soltanto le linee dati D4,D5,D6 e D7 e gli 8 bit che identificano il carattere (continua a leggere) li invieremo in 2 gruppi da 4 (nibble): prima i 4 bit di ordine superiore (7-4) e quindi i 4 bit di ordine inferiore (3-0).

La modalità a 4 bit è vantaggiosa perché ci permette di risparmiare preziosi pin sul microcontrollore che gestirà il display, ma per contro dovranno essere scritte delle routine più complicate per poterlo gestire, soprattutto in lettura.

Il controller di questi display possiede due banchi di memoria: una CGROM (Character Generator Read Only Memory) che contiene in pratica i “font” utilizzati per i caratteri e una DDRAM (Display Data Random Access Memory) che ha il compito di memorizzare “cosa” va scritto sul display. Quando invieremo dei dati (codici dei caratteri) sulla linea dati, questi saranno memorizzati nella DDRAM, un contatore si occuperà di scrivere nella DDRAM i caratteri uno dopo l’altro. La DDRAM ha una capacità massima di 80 caratteri

Ogni posizione (carattere) sul display è quindi mappato da una locazione nella DDRAM.

Nei display 16×2, ad esempio, la prima riga è mappata dalle posizioni di memoria

0x00 ÷ 0x0F (0÷15 in decimale)

mentre la seconda riga è mappata dalle posizioni

0x40 ÷ 0x4F (64 ÷ 79 in decimale)

per cui potete capire che scrivendo, ad esempio, nella locazione 16 (decimale) un carattere, non produce nulla sullo schermo.

La CGROM, abbiamo detto, contiene i font dei caratteri.

Guardiamo sul datasheet del controller la tabella dei font presenti sulla CGROM del HD44780

Il funzionamento è molto semplice: quando vorremo, ad esempio, visualizzare la lettera A, invieremo sulla linea dati il valore binario:

0100 0001

(si incrocia la colonna con la riga).

i caratteri che hanno il codice dal n°32 (carattere spazio: 0010 0000 in binario) al n° 125 (parentesi graffa aperta a sinistra, 0111 1101) rispecchiano i codici ascii. A parte il simbolo dello Yen

Vediamo l'elenco dei comandi che si possono impartire al display

Command	Binary								Hex
	D7	D6	D5	D4	D3	D2	D1	D0	
Clear Display	0	0	0	0	0	0	0	1	01
Display & Cursor Home	0	0	0	0	0	0	1	x	02 or 03
Character Entry Mode	0	0	0	0	0	1	I/D	S	04 to 07
Display On/Off & Cursor	0	0	0	0	1	D	U	B	08 to 0F
Display/Cursor Shift	0	0	0	1	D/C	R/L	x	x	10 to 1F
Function Set	0	0	1	8/4	2/1	10/7	x	x	20 to 3F
Set CGRAM Address	0	1	A	A	A	A	A	A	40 to 7F
Set Display Address	1	A	A	A	A	A	A	A	80 to FF

I/D: 1=Increment*, 0=Decrement

S: 1=Display shift on, 0=Display shift off*

D: 1=Display On, 0=Display Off*

U: 1=Cursor underline on, 0=Underline off*

B: 1=Cursor blink on, 0=Cursor blink off*

D/C: 1=Display shift, 0=Cursor move

R/L: 1=Right shift, 0=Left shift

8/4: 1=8 bit interface*, 0=4 bit interface

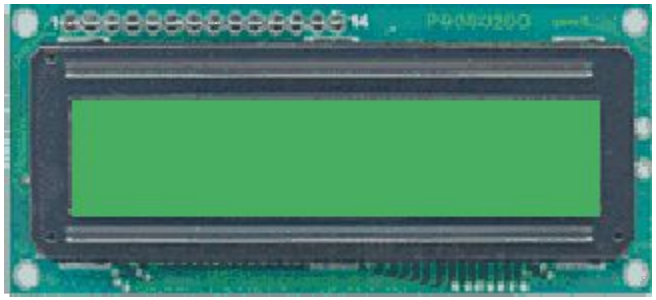
2/1: 1=2 line mode, 0=1 line mode*

10/7: 1=5x10 dot format, 0=5x7 dot format*

x = Don't care

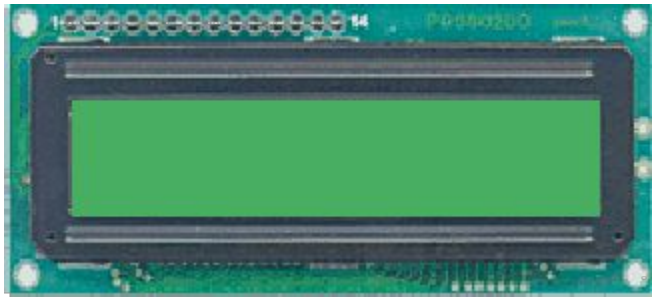
* = Initialisation settings

Entry Mode Set



Entry Mode = 0x04 (ovvero I/D=0 e S=0)

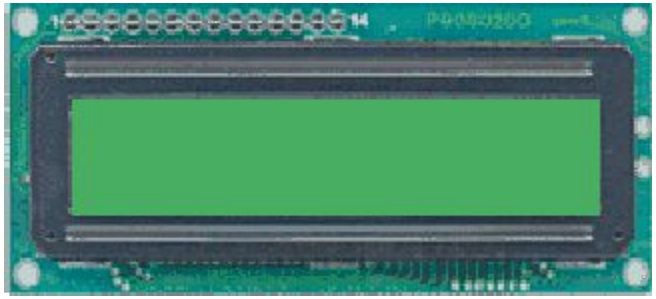
Stringa inviata all'LCD: DerelilI cursore si sposta verso sinistra (la scritta si sviluppa alla rovescia) e il testo è fermo(I/D=0 la posizione decrementa, S=0 il testo rimane fermo)



Entry Mode = 0x05 (I/D=0 e S=1)

Stringa inviata all'LCD: HakanII cursore rimane fermo e si sposta invece la scritta (la quale si sviluppa sempre alla rovescia)(I/D=0 la posizione decrementa, S=1 il testo si sposta)

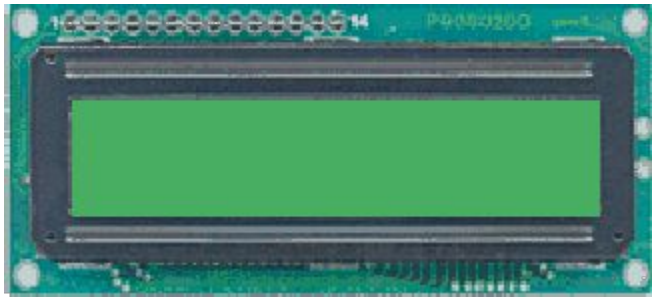
Entry Mode Set



Entry Mode = 0x06 (I/D=1 e S=0)

Stringa inviata all'LCD: DerelilI cursore si sposta verso destra (la scritta si sviluppa "dritta") e il testo rimane fermo.

(I/D=1 la posizione incrementa, S=0 il testo è fermo)Questa è l'impostazione utilizzata nella maggior parte dei casi)



Entry Mode = 0x07 (I/D=1 e S=1)

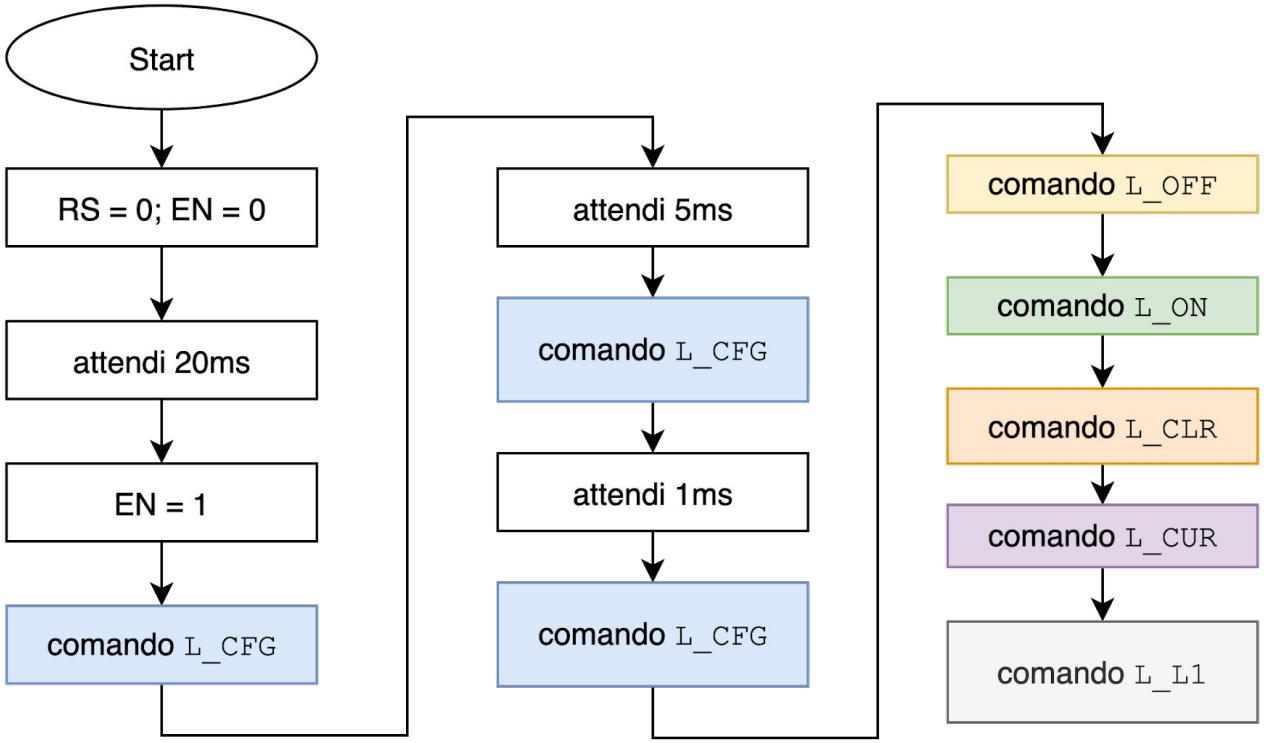
Stringa inviata all'LCD: HakanlI cursore rimane fermo e si sposta invece la scritta (la quale si sviluppa "dritta")(I/D=1 la posizione incrementa, S=1 il testo si sposta)

Ovviamente, programmando, può **non** verificarsi l'effetto "macchina da scrivere" dal momento che l'invio dei dati è solitamente molto veloce, per cui visualizzeremo unicamente la scritta completa.

Per ottenere questo effetto si può inserire una pausa tra l'invio di un carattere e l'altro.

Inizializzazione del display

Il display deve essere inizializzato prima di poter essere utilizzato, in realtà l'inizializzazione di base viene effettuata dalla circuiteria interna del display, ma ciò non sempre soddisfa i nostri requisiti, per cui l'avvio del display viene sempre effettuato via software impartendo al display la giusta sequenza di istruzioni come indicato dal datasheet

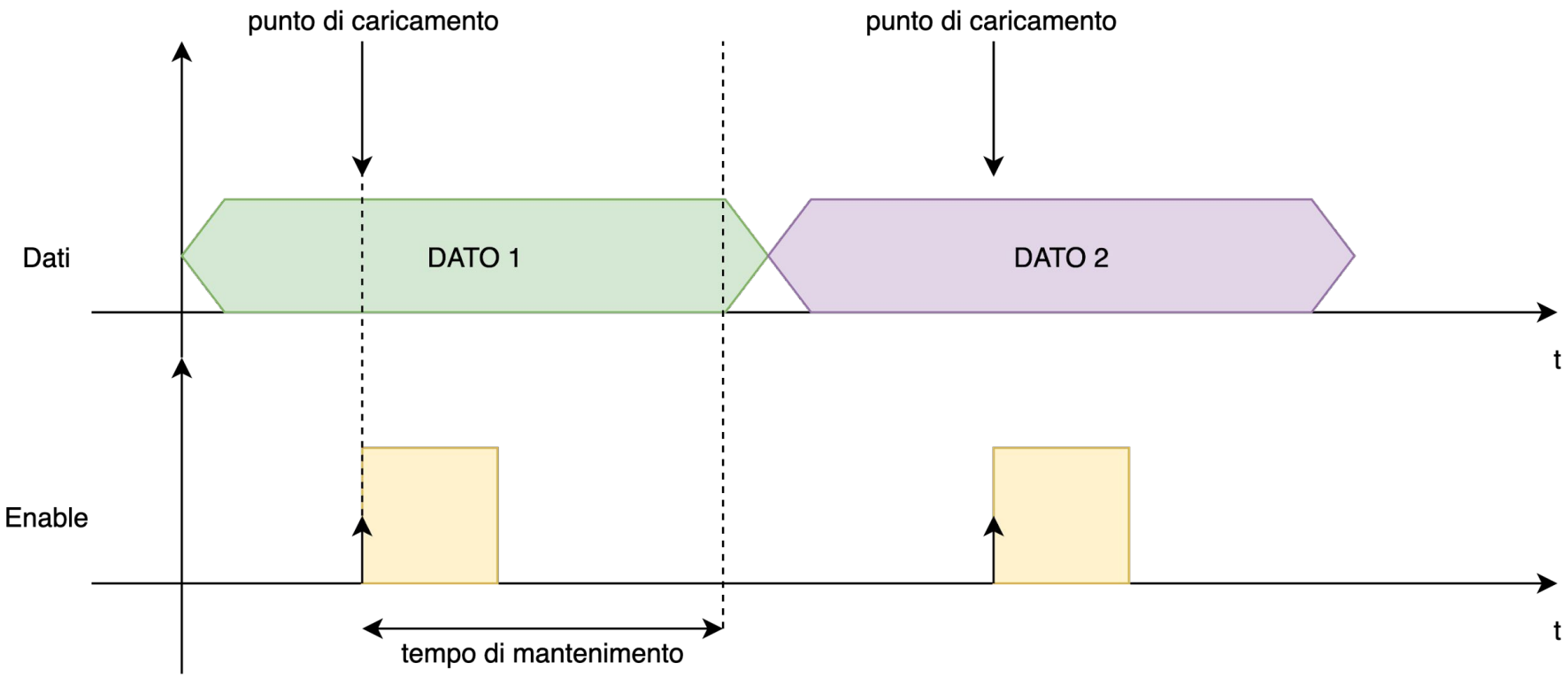


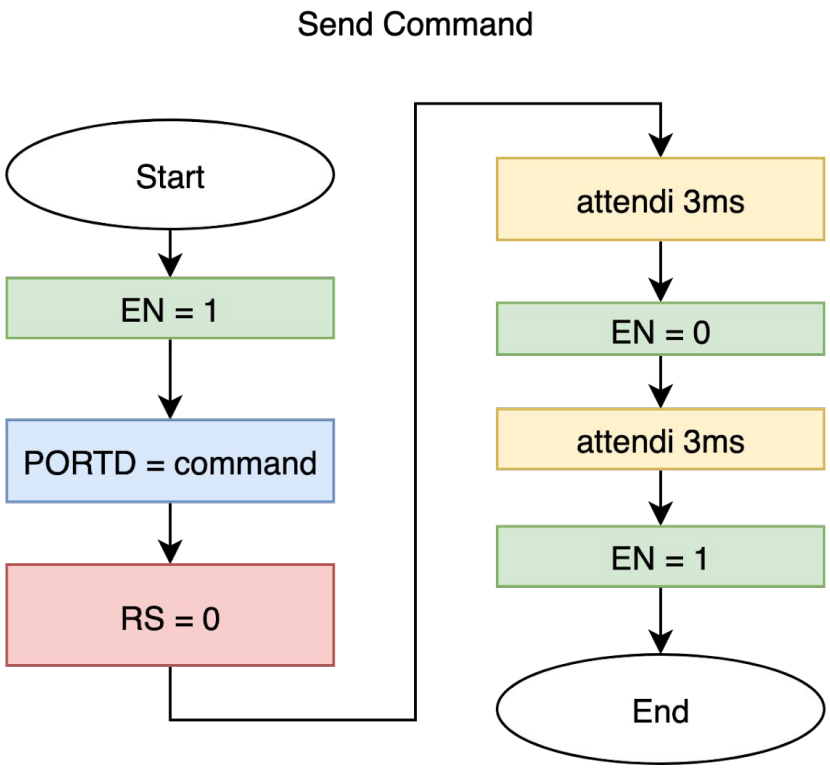

```
#define L_ON     0x0F
#define L_OFF    0x08
#define L_CLR    0x01
#define L_L1     0x80
#define L_L2     0xC0
#define L_CR     0x0F
#define L_NCR    0x0C
#define L_CFG    0x38
#define L_CUR    0x0E
```

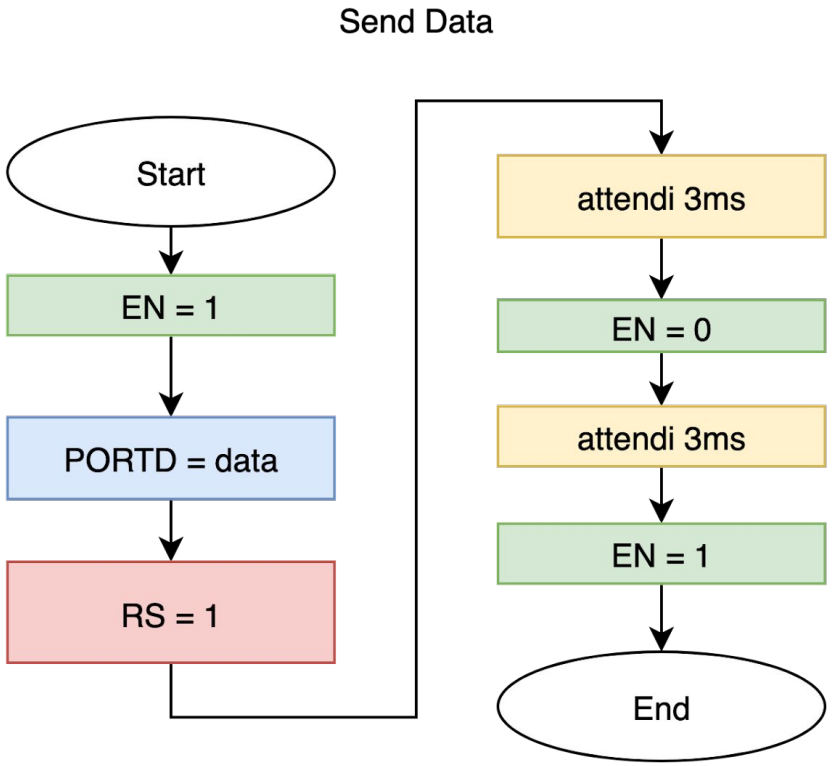
Questa la sequenza di inizializzazione del display.

Le attese vengono fatte perchè il display per eseguire particolari comandi, ha necessità di particolari tempi.

Inserire un comando o dato

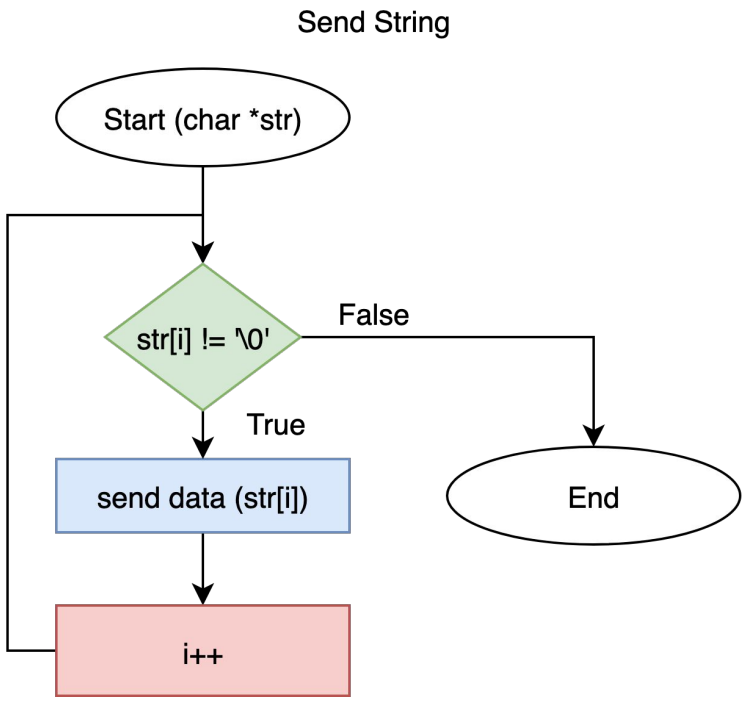






Questa la sequenza per l'invio di un comando o un dato al display

Ricordo che l'invio di comandi serve ad impostare il display, mentre l'invio di dati "scrive" i caratteri sul display



Volendo inviare una stringa al display, basterà inviare in sequenza il contenuto dell’array in modalità “data”

Esercitazione

Si richiede l’implementazione di un codice che inizializzi il display della board 4 e realizzi una scritta di benvenuto come da immagine:

Esercitazione

Implementare l’esercizio della calcolatrice usando il display LCD al fine di far comparire l’intera equazione completa di risultato

Esercitazione

Implementare un sistema provvisto di un Menu a 4 voci, selezionabili tramite due pulsanti + e -