

## Tutorial: Toshiba – Controller LCD T6963

Il chip *Toshiba T6963* è un controller LSI (Large Scale Integration) per LCD (Liquid Crystal Display) monocromatici, costituiti da una matrice di pixel.

### Premessa

Questo tutorial non vuole essere assolutamente esaustivo. Il suo scopo è fornire una serie di informazioni (si spera utili) in maniera organica, in modo che chi non abbia mai utilizzato chip simili possa essere operativo già dopo una prima lettura.

In rete è disponibile il datasheet completo e un paio di application notes molto interessanti, materiale – d'altra parte – usato per la stesura di questo documento e a cui si rimanda per ulteriori e più approfondite informazioni.

Il chip in esame è un controller della Toshiba Corporation dedicato al pilotaggio di LCD: il microcontroller principale dell'applicazione comunicherà con esso, fornendo particolari comandi che seguono precise regole e un protocollo ben determinato. La logica interna del chip (su cui, d'altronde, gira un firmware dedicato) assocerà agli input provenienti dal microcontroller adeguati segnali di pilotaggio per l'architettura del display in cui il chip è inserito. In questa maniera, il microcontroller, comunicando con il controller del LCD, riesce a gestire il display.

### Note alla revisione 1

Questa nuova revisione del documento integra la precedente inserendo nuovi spunti pubblicati in un articolo molto ampio ed interessante di "Fare Elettronica" (Edizioni InWare, [www.farelettronica.com](http://www.farelettronica.com)), numero 264 del giugno 2007. L'autore dell'articolo, "Il controller T6963C", è Giuseppe Modugno, a cui vanno i miei personali complimenti.

### Generalità

Il microcontroller si interfaccia con il chip T6963, in modo da pilotare in maniera opportuna la visualizzazione delle schermate del display utilizzato. Il T6963 e il microcontroller comunicano attraverso un bus dati parallelo a 8 bit e attraverso opportune linee di controllo, utilizzate per settare operazioni di lettura e scrittura.

Il chip della Toshiba è dotato di una ROM interna in cui sono memorizzati 128 caratteri di default che esso può visualizzare. La dimensione massima di questi caratteri è 8x8 pixel.

Inoltre esso può anche interfacciarsi con una RAM esterna (di dimensioni massime 64KB). La comunicazione tra T6963 e RAM avviene indirizzando 16 bit sul bus indirizzo. Inoltre, tale RAM può essere divisa in tre blocchi logici:

- una *text area*, in cui sono allocati i caratteri;
- una *graph area*, in cui sono allocati elementi grafici;
- una *CGRAM area* (Char Generator RAM), in cui sono allocati caratteri definiti dall'utente.

La finestra di visualizzazione (quello che, cioè, appare sul display del LCD) può essere spostata da un'area di memoria all'altra facilmente, consentendo anche di visualizzare elementi della text area e della graph area insieme.

La presenza di molti pin di configurazione rende questo controller utilizzabile su diversi tipi di display grafici: dai più piccoli 128x64 ai più grandi 240x128. Inoltre, il duty-cycle del display può variare da 1/16 a 1/128.

Infine, è bene tener presente che questo chip ha solo funzioni di controllore, ma non è capace di pilotare direttamente i segmenti di un LCD; sarà necessario, quindi, usare appositi driver (come il T6A39 per le colonne e il T6A40 per le linee). Il produttore del display si occuperà dell'interfaccia tra il controller e il chip, mentre l'utilizzatore deve occuparsi dell'interfaccia tra il chip T6963 e il proprio microcontroller.

### Caratteristiche principali

1. Formato del display (configurabile attraverso opportuni pin):
  - numero di colonne: 32, 40, 64, 80;
  - numero di linee: 2, 4, 6, 8, 10, 12, 14, 16, 20, 24, 28, 32.
2. Dimensioni dei font per i caratteri (configurabile attraverso opportuni pin):
  - pin in orizzontale: 5, 6, 7, 8;
  - pin in verticale: 8 (fissi).
3. 128 caratteri di default in memoria ROM.
4. Interfacciamento con una RAM esterna di dimensioni massime 64KB. Gli indirizzi della text area, della graph area e della CGRAM sono stabiliti tramite software.
5. Le operazioni di lettura e scrittura non disturbano il display.
6. La RAM esterna deve essere statica, il chip – infatti – non può effettuare alcun refresh della RAM.
7. Le funzioni di attributo, che in seguito saranno esaminate, possono essere usate solo in text mode.

## Descrizione dei pin

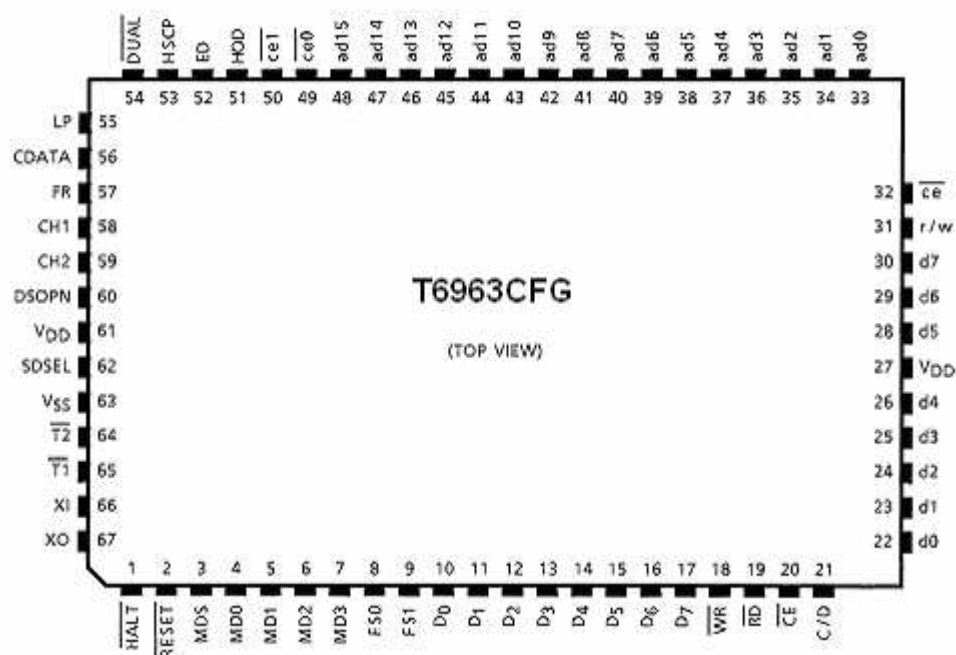


Fig. 1 – Pin-out del chip T6963CFG

Pin Name	I / O	Functions																																																																																																																															
MDS MD0 MD1	Input	<div>Pins for selection of LCD size</div> <table><tr><td>DUAL</td><td>H</td><td>H</td><td>H</td><td>H</td><td>H</td><td>H</td><td>H</td><td>H</td><td>L</td><td>L</td><td>L</td><td>L</td><td>L</td><td>L</td><td>L</td><td>L</td><td>L</td></tr><tr><td>MDS</td><td>L</td><td>L</td><td>L</td><td>L</td><td>H</td><td>H</td><td>H</td><td>H</td><td>L</td><td>L</td><td>L</td><td>L</td><td>H</td><td>H</td><td>H</td><td>H</td><td>H</td></tr><tr><td>MD1</td><td>H</td><td>H</td><td>L</td><td>L</td><td>H</td><td>H</td><td>L</td><td>L</td><td>L</td><td>H</td><td>H</td><td>L</td><td>L</td><td>H</td><td>H</td><td>L</td><td>L</td></tr><tr><td>MD0</td><td>H</td><td>L</td><td>H</td><td>L</td><td>H</td><td>L</td><td>H</td><td>L</td><td>L</td><td>H</td><td>L</td><td>H</td><td>L</td><td>H</td><td>L</td><td>H</td><td>L</td></tr><tr><td>LINES</td><td>2</td><td>4</td><td>6</td><td>8</td><td>10</td><td>12</td><td>14</td><td>16</td><td>4</td><td>8</td><td>12</td><td>16</td><td>20</td><td>24</td><td>28</td><td>32</td><td></td></tr><tr><td>V-DOTS</td><td>16</td><td>32</td><td>48</td><td>64</td><td>80</td><td>96</td><td>112</td><td>128</td><td>32</td><td>64</td><td>96</td><td>128</td><td>160</td><td>192</td><td>224</td><td>256</td><td></td></tr><tr><td></td><td colspan="8">1 SCREEN</td><td colspan="9">2 SCREENS</td><td></td></tr></table>	DUAL	H	H	H	H	H	H	H	H	L	L	L	L	L	L	L	L	L	MDS	L	L	L	L	H	H	H	H	L	L	L	L	H	H	H	H	H	MD1	H	H	L	L	H	H	L	L	L	H	H	L	L	H	H	L	L	MD0	H	L	H	L	H	L	H	L	L	H	L	H	L	H	L	H	L	LINES	2	4	6	8	10	12	14	16	4	8	12	16	20	24	28	32		V-DOTS	16	32	48	64	80	96	112	128	32	64	96	128	160	192	224	256			1 SCREEN								2 SCREENS									
DUAL	H	H	H	H	H	H	H	H	L	L	L	L	L	L	L	L	L																																																																																																																
MDS	L	L	L	L	H	H	H	H	L	L	L	L	H	H	H	H	H																																																																																																																
MD1	H	H	L	L	H	H	L	L	L	H	H	L	L	H	H	L	L																																																																																																																
MD0	H	L	H	L	H	L	H	L	L	H	L	H	L	H	L	H	L																																																																																																																
LINES	2	4	6	8	10	12	14	16	4	8	12	16	20	24	28	32																																																																																																																	
V-DOTS	16	32	48	64	80	96	112	128	32	64	96	128	160	192	224	256																																																																																																																	
	1 SCREEN								2 SCREENS																																																																																																																								
MD2 MD3	Input	<div>Pins for selection of number of columns</div> <table><tr><td>MD2</td><td>H</td><td>L</td><td>H</td><td>L</td></tr><tr><td>MD3</td><td>H</td><td>H</td><td>L</td><td>L</td></tr><tr><td>Columns</td><td>32</td><td>40</td><td>64</td><td>80</td></tr></table>	MD2	H	L	H	L	MD3	H	H	L	L	Columns	32	40	64	80																																																																																																																
MD2	H	L	H	L																																																																																																																													
MD3	H	H	L	L																																																																																																																													
Columns	32	40	64	80																																																																																																																													
FS0 FS1	Input	<div>Pins for selection of font</div> <table><tr><td>FS0</td><td>H</td><td>L</td><td>H</td><td>L</td></tr><tr><td>FS1</td><td>H</td><td>H</td><td>L</td><td>L</td></tr><tr><td>Font</td><td>5 × 8</td><td>6 × 8</td><td>7 × 8</td><td>8 × 8</td></tr></table>	FS0	H	L	H	L	FS1	H	H	L	L	Font	5 × 8	6 × 8	7 × 8	8 × 8																																																																																																																
FS0	H	L	H	L																																																																																																																													
FS1	H	H	L	L																																																																																																																													
Font	5 × 8	6 × 8	7 × 8	8 × 8																																																																																																																													
D0 to D7	I / O	Data I / O pins between CPU and T6963CFG (D7 is MSB)																																																																																																																															
WR	Input	Data Write. Write data into T6963CFG when WR = L.																																																																																																																															
RD	Input	Data Read. Read data from T6963CFG when RD = L.																																																																																																																															
CE	Input	Chip Enable for T6963CFG. CE must be L when CPU communicates with T6963CFG.																																																																																																																															

Pin Name	I / O	Functions										
C / D	Input	WR = L ..... C / D = H: Command Write    C / D = L: Data Write RD = L ..... C / D = H: Status Read    C / D = L: Data Read										
HALT	Input	H ..... Normal, L ..... Stops the oscillation of the clock										
RESET	Input	H ..... Normal (T6963CFG has internal pull-up resistor) L ..... Initialize T6963CFG. Text and graphic have addresses and text and graphic area settings are retained.										
DSPON	Output	Control pin for external DC / DC. DSPON is L when HALT is L or RESET is L. (When DSPON goes H, the column drivers are cleared.)										
DUAL	Input	H ..... Single-Scan L ..... Dual-Scan <table><tr><td>DUAL</td><td>H</td><td>H</td><td>L</td><td>L</td></tr><tr><td>SDSEL</td><td>H</td><td>L</td><td>H</td><td>L</td></tr></table>	DUAL	H	H	L	L	SDSEL	H	L	H	L
DUAL	H	H	L	L								
SDSEL	H	L	H	L								
SDSEL	Input	H ..... Sending data by odd / even separation L ..... Sending data by simple serial method <table><tr><td>Upper screen</td><td>HOD, ED</td><td>ED</td><td>HOD, ED</td><td>ED</td></tr><tr><td>Lower screen</td><td>—</td><td>—</td><td>LOD, ED</td><td>ED</td></tr></table>	Upper screen	HOD, ED	ED	HOD, ED	ED	Lower screen	—	—	LOD, ED	ED
Upper screen	HOD, ED	ED	HOD, ED	ED								
Lower screen	—	—	LOD, ED	ED								
ce0 (LOD)	Output	ce0 at DUAL = H Chip enable pin for display memory in the address range 0000H to 07FFH LOD at DUAL = L Serial data output for odd columns in lower area of LCD										
ce0 (LOD)	Output	ce0 at DUAL = H Chip enable pin for display memory in the address range 0000H to 07FFH LOD at DUAL = L Serial data output for odd columns in lower area of LCD										
ce1 (LSCP)	Output	ce1 at DUAL = H Chip enable pin for display memory in the address range 0800H to 0FFFH LSCP at DUAL = L Shift clock pulse output for column drivers in lower area of LCD										
ce	Output	Chip enable pin for display memory of any address										
d0 to d7	I / O	Data I / O pins for display memory										
ad0 to ad15	Output	Address outputs for display memory (ad15 = L: for upper area of LCD, ad15 = H: for lower area of LCD)										
R / W	Output	Read / Write signal for display memory										
ED	Output	SDSEL = H: Data output for even columns in both upper and lower areas of LCD SDSEL = L: Data output for columns in both upper and lower areas of LCD										
HOD	Output	Data output for odd columns in upper area of LCD										
CDATA	Output	Synchronous signal for row driver										
HSCP	Output	Shift clock pulse for column driver of upper area of LCD										
LP	Output	Latch pulse for column driver. Shift clock pulse for row driver										
FR	Output	Frame signal										
XI	Input	Crystal oscillator input										
XO	Output	Crystal oscillator output										
CH1, CH2	Output	Check signal										
T1, T2	Input	Test input. Usually open										
VDD	—	Power supply (5.0 V)										

Fig. 2 – Descrizione delle funzionalità dei pin

A seconda del tipo di display che si usa, diversi pin tra i presentati sono configurati in hardware dal costruttore del display (ad esempio, quelli che definiscono le dimensioni del display o le dimensioni delle linee e delle colonne).

Si precisa, a proposito del pin  $\overline{\text{MD1}}$ , che esso viene usato per impostare la modalità a schermo singolo o doppio. Nel secondo caso il display viene suddiviso in una sezione superiore ed in una sezione inferiore, per le quali il controller esegue il refresh contemporaneamente.

La scelta per i pin MD2 e MD3 è legata alla dimensione del carattere: con un font 8x8 si possono visualizzare fino a 30 colonne (si può scegliere qualsiasi configurazione per MD2 e MD3), con un font 6x8, il numero di colonne visualizzabili sarà 40 (e tale o maggiore deve essere il valore scelto con MD2 e MD3).

In realtà, queste configurazioni sono settate dal produttore del display. L'utilizzatore può analizzarle per meglio capire le possibilità del proprio display.

Il dato utilizza il bus D0-D7, mentre i 4 segnali di controllo sono:  $\overline{\text{CS}}$ ,  $\overline{\text{RD}}$ ,  $\overline{\text{WR}}$ , C/D. Se gli 8 bit del dato sono usati soltanto per la comunicazione col display (il bus, quindi, non è condiviso), il pin  $\overline{\text{CS}}$  può essere forzato permanentemente a livello basso (tramite hardware, ad esempio).

## Oscillatore

Il T6963 è un circuito digitale integrato di tipo sequenziale e necessita, quindi, di un clock che temporizzi tutte le operazioni da effettuare. Internamente è presente un oscillatore che deve essere collegato ad un quarzo e ai due condensatori di fuga. La frequenza di oscillazione ( $f_{osc}$ ), cioè il quarzo da usare, dipende dal numero di linee e dal numero di colonne, secondo la formula:

$$f_{osc} = f_R \times 64 \times 2 \times M \times N,$$

dove  $f_R$  è 60Hz, M è il numero di colonne, N è il numero di linee.

Nel caso di display pilotati in modalità dual scan, questa frequenza va raddoppiata. In ogni caso, tale frequenza non deve superare i 5.5MHz.

Anche questo parametro, comunque, è fissato dal produttore del display.

## Regolazione del contrasto

La regolazione del contrasto non è strettamente legata al controller, ma si realizza fornendo a altri chip presenti sul display una particolare tensione.

Tipicamente questa tensione è negativa (in genere si arriva anche a -12V o -15V) e, quindi, per l'utilizzo di questo display non è sufficiente un'alimentazione singola a +5V. In genere, però, i display non necessitano di una corrente molto elevata sul pin di contrasto (massimo 4mA), cosa che permette di utilizzare dei semplici convertitori DC/DC per ottenere la tensione negativa a partire dalla tensione positiva.

## Operazioni essenziali per il corretto funzionamento

Dopo aver alimentato il chip, il pin di  $\overline{\text{RESET}}$  deve essere portato a 0 in 5 colpi di clock, altrimenti il T6963 non inizia la comunicazione col microcontroller.

Lo stato del chip deve essere controllato dal microcontroller prima dell'invio di ogni comando e ogni dato deve essere inviato prima del comando a cui si riferisce.

I codici caratteri non sono quelli ASCII, ma differenti.

## Interfaccia con la RAM esterna

La RAM esterna (statica, in quanto non è possibile usare RAM di tipo dinamico con questo controller) è usata per allocare i dati che devono essere visualizzati sul display. L'indirizzamento dei dati in RAM avviene su un address bus a 16 bit (ad0-ad15), mentre un bus dati (d0-d7) consente di inviare dati alla RAM e di riceverli. Si ricorda, poi, che un altro bus a 8 bit (D0-D7) consente la comunicazione tra T6963 e microcontroller.

L'immagine visualizzata sul display è, quindi, il risultato del contenuto memorizzato nella RAM esterna.

La RAM può essere configurata in due modalità diverse: single scan e dual scan.

In modalità *single scan*, text area, graph area e CGRAM area possono essere liberamente allocate nello spazio messo a disposizione (la dimensione massima, ricordiamo, è 64KB).

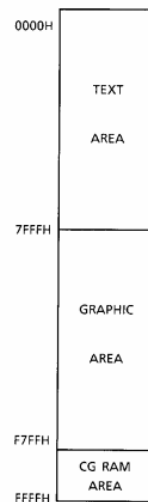


Fig. 3 – Esempio di allocazione single scan della memoria RAM nel caso di uno spazio disponibile di 64KB

Con il *dual scan*, le due pagine di memoria (ciascuna di dimensione pari alla metà della dimensione totale, ad esempio 32KB per pagina) hanno partizioni analoghe. Il bit più significativo dei 16 usati per l'indirizzamento (ad15) indica a quale delle due pagine l'indirizzo fa riferimento.

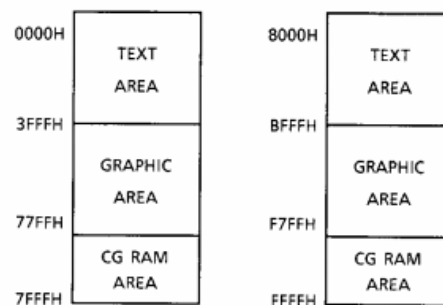


Fig. 4 – Esempio di allocazione dual scan della memoria RAM nel caso di uno spazio disponibile di 64KB

## Setting dello stato

Lo stato del chip può essere letto dal bus dati a 8 bit, dopo un opportuna sequenza eseguita dai pin di controllo. In particolare il protocollo per la lettura dello stato è (si ricordi la simbologia: L per low e H per high):

- $\overline{RD}$  =L;
- $\overline{WR}$  =H;
- $\overline{CE}$  =L;
- C/D=H;
- D0-D7=status word.

Lo stato, contenuto negli 8 bit della *status word*, è così codificato:

MSB				LSB			
STA7 D7	STA6 D6	STA5 D5	STA4 D4	STA3 D3	STA2 D2	STA1 D1	STA0 D0
STA0	Check command execution capability					0: Disable 1: Enable	
STA1	Check data read / write capability					0: Disable 1: Enable	
STA2	Check Auto mode data read capability					0: Disable 1: Enable	
STA3	Check Auto mode data write capability					0: Disable 1: Enable	
STA4	Not used						
STA5	Check controller operation capability					0: Disable 1: Enable	
STA6	Error flag. Used for Screen Peek and Screen copy commands.					0: No error 1: Error	
STA7	Check the blink condition					0: Display off 1: Normal display	

Fig. 5 – Codifica della status word

I bit STA0 e STA1 devono essere sempre controllati insieme come azione generica per il controllo del corretto funzionamento, mentre i bit STA2 e STA3 hanno la stessa funzione, ma in modalità auto mode.

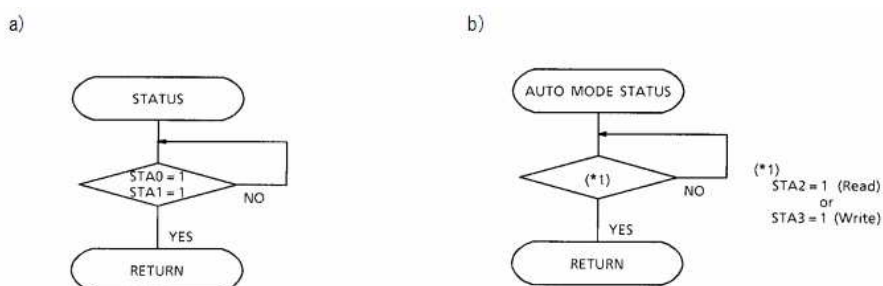


Fig. 6 – Algoritmo che il microcontroller deve seguire per il controllo dello stato

## Setting del dato

Bisogna prima settare il dato, poi inviare il comando a cui il dato si riferisce (ovviamente, come poi si vedrà, esistono anche comandi che non hanno bisogno di operandi e che sono inviati al T6963 senza nessun dato associato).

In figura sono presenti i flow chart per il setting del dato, per comandi che lo richiedono:

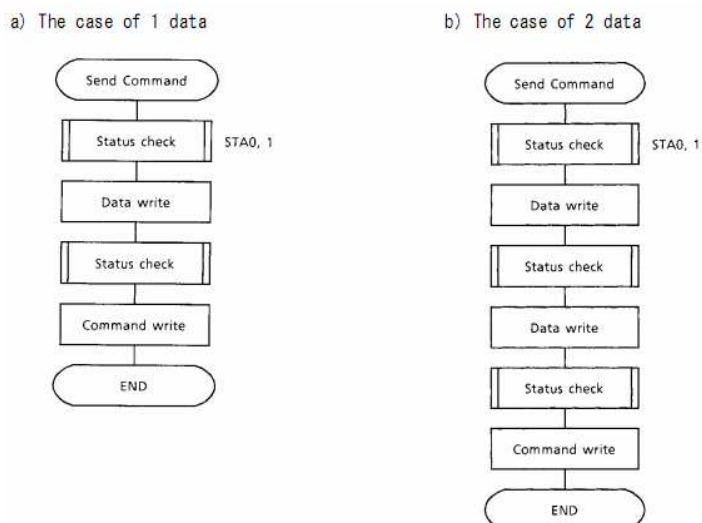


Fig. 7 – Algoritmo che il microcontroller deve seguire per la scrittura di un dato

Se si mandano più dati, l'ultimo (o gli ultimi due, a seconda del comando) è quello valido.  
L'operazione di lettura di un dato dal T6963 è analoga.

### Invio del comando

Esistono tre tipi di comandi:

- comandi che non hanno bisogno di nessun operando;
- comandi che hanno bisogno di un operando a 8 bit;
- comandi che hanno bisogno di due operandi, ciascuno di 8 bit.

Ecco, in figura, i 3 algoritmi corrispondenti:

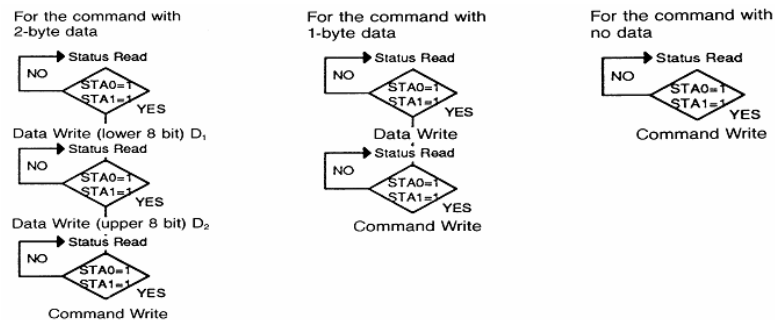


Fig. 8 – Algoritmi del microcontroller per l'invio di comandi al controller T6963

### Definizione dei comandi per la gestione della RAM esterna

Command	Code	D1	D2	Function
REGISTERS SETTING	00100001 00100010 00100100	X address Data Low address	Y address 00H High address	Set Cursor Pointer Set Offset Register Set Address Pointer
SET CONTROL WORD	01000000 01000001 01000010 01000011	Low address Columns Low address Columns	High address 00H High address 00H	Set Text Home Address Set Text Area Set Graphic Home Address Set Graphic Area
MODE SET	1000X000 1000X001 1000X011 1000X100 10000XXX 10001XXX	— — — — — —	— — — — — —	OR mode EXOR mode AND mode Text Attribute mode Internal CG ROM mode External CG RAM mode
DISPLAY MODE	10010000 1001XX10 1001XX11 100101XX 100110XX 100111XX	— — — — — —	— — — — — —	Display off Cursor on, blink off Cursor on, blink on Text on, graphic off Text off, graphic on Text on, graphic on
CURSOR PATTERN SELECT	10100000 10100001 10100010 10100011 10100100 10100101 10100110 10100111	— — — — — — — —	— — — — — — — —	1-line cursor 2-line cursor 3-line cursor 4-line cursor 5-line cursor 6-line cursor 7-line cursor 8-line cursor
DATA AUTO READ / WRITE	10110000 10110001 10110010	— — —	— — —	Set Data Auto Write Set Data Auto Read Auto Reset
DATA READ / WRITE	11000000 11000001 11000010 11000011 11000100 11000101	Data — Data — Data —	— — — — — —	Data Write and Increment ADP Data Read and Increment ADP Data Write and Decrement ADP Data Read and Decrement ADP Data Write and Nonvariable ADP Data Read and Nonvariable ADP
SCREEN PEEK	11100000	—	—	Screen Peek
SCREEN COPY	11101000			Screen Copy



Command	Code	D1	D2	Function
BIT SET / RESET	11110XXX	—	—	Bit Reset
	11111XXX	—	—	Bit Set
	1111X000	—	—	Bit 0 (LSB)
	1111X001	—	—	Bit 1
	1111X010	—	—	Bit 2
	1111X011	—	—	Bit 3
	1111X100	—	—	Bit 4
	1111X101	—	—	Bit 5
	1111X110	—	—	Bit 6
	1111X111	—	—	Bit 7 (MSB)

Fig. 9 – Lista dei comandi disponibili per il pilotaggio del T6963 (X rappresenta bit non validi)

## Analisi dei singoli gruppi di comandi

### 1. Register setting

Code	Hex.	Function	D1	D2
00100001	21H	SET CURSOR POINTER	X ADRS	Y ADRS
00100010	22H	SET OFFSET REGISTER	DATA	00H
00100100	24H	SET ADDRESS POINTER	LOW ADRS	HIGH ADRS

Fig. 10 – Famiglia register setting

#### 1.1. Set cursor pointer

XADRS e YADRS specificano la posizione del cursore sul display e sono da intendersi in termini di linee e colonne. Si ricordi che i valori ammissibili sono:

- numero di colonne: 32, 40, 64, 80;
- numero di linee: 2, 4, 6, 8, 10, 12, 14, 16, 20, 24, 28, 32.

Quindi, i range di variazione dei due operandi di questo comando sono (il prefisso 0x indica che il numero che segue è da intendersi esadecimale, mentre il prefisso 0b indica che il numero che segue è da intendersi in binario):

- 0x00-0x4F (0-79) per XADRS;
- 0x00-0x1F (0-31) per YADRS.

Inoltre, operazioni di lettura e scrittura non cambiano la posizione del cursore sul display.

#### 1.2. Set offset register

Il *registro di offset* è usato per determinare quali caratteri usare nella CGRAM.

Il T6963 ha un bus indirizzo (*address bus*) verso la RAM di 16 bit, schematizzato in figura:

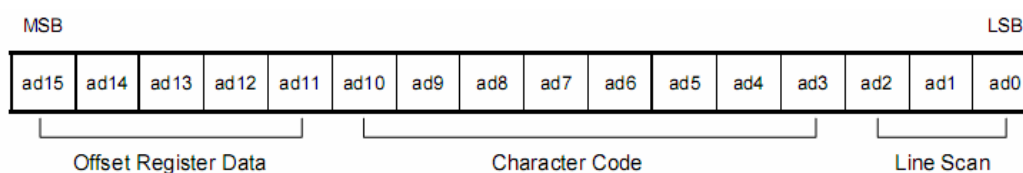


Fig. 11 – Bus indirizzo per la comunicazione con la RAM esterna

Si può usare la CGRAM in due modalità diverse:

- internal CGROM mode;
- external CGRAM mode.

In modalità *internal CGROM*, si discriminano gli 8 bit del *character code* in due grosse famiglie: valori compresi nel range 0x00-0x7F (0-127) e valori compresi nel range 0x80-0xFF (128-255). Per valori del character code compresi in 0x00-0x7F (0-127) verranno utilizzati i 128 caratteri della ROM interna del chip, per valori del character code compresi in 0x80-0xFF (128-255) verranno utilizzati i 128 caratteri inseriti opportunamente nella RAM esterna (nell'area, appunto, della CGRAM).

Ogni carattere è definito da 8 byte (8 bit fissi per l'altezza e al massimo 8 bit per la larghezza).



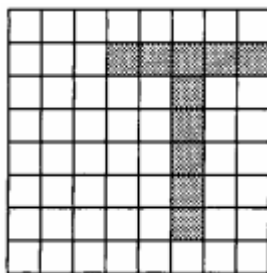


Fig. 12 – Esempio di un carattere, ogni quadratino è un pixel del LCD

In modalità *external CGRAM*, si inseriscono nella RAM esterna (nell'area CGRAM) 256 caratteri. Quindi, anche character code compresi in 0x00-0x7F (0-127) faranno riferimenti a caratteri nella RAM.

In entrambi i casi, la CGRAM ha dimensioni:

$$\begin{aligned}
 &128 \text{ locazioni per character code in } 0x00-0x7F \text{ (0-127)} + \\
 &128 \text{ locazioni per character code in } 0x80-0xFF \text{ (128-255)} = \\
 &128 \times 8 \text{ byte} + 128 \times 8 \text{ byte} = \\
 &1024 \text{ byte} + 1024 \text{ byte} = \\
 &2048 \text{ byte} = \\
 &0x800 \text{ byte.}
 \end{aligned}$$

Ovviamente, in modalità interna i primi 1024 byte non contengono caratteri utili (in quanto i 128 caratteri sono in quel caso presi dalla ROM del chip), ma comunque il T6963 li alloca, in modo tale da avere una gestione della memoria più comoda.

L'offset è codificato su 5 bit (ad15-ad11), quindi si hanno 32 valori possibili. Una RAM di 64KB, dunque, è divisa in 32 zone di 2048 byte ciascuna. La corrispondenza tra offset e zona della RAM è mostrata in figura:

Offset register data	CG RAM hex. address (start to end)
00000	0000 to 07FFH
00001	0800 to 0FFFH
00010	1000 to 17FFH
11100	E000 to E7FFH
11101	E800 to EFFFH
11110	F000 to F7FFH
11111	F800 to FFFFH

Fig. 13 – Corrispondenza tra offset e RAM

La scelta di un offset (da 0 a 31) stabilisce la zona in cui è allocata la CGRAM, il character code (da 0 a 255) stabilisce l'inizio del carattere di interesse, il line scan (da 0 a 7) identifica quale tra le 8 linee di un carattere si sta indirizzando.

Ad esempio, si ipotizzi di voler indirizzare la prima linea del carattere 0x80 (0b10000000) in una CGRAM che inizia da 0x1800, relativo quindi all'offset 0b00011 (3). I 16 bit della stringa di indirizzo sono:

ad15	ad14	ad13	ad12	ad11	ad10	ad9	ad8	ad7	ad6	ad5	ad4	ad3	ad2	ad1	ad0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
1				C				0				0			

Quindi, l'indirizzo dell'inizio del carattere è 0x1C00, che è l'indirizzo assoluto nella RAM per il dato che interessa.

In figura è presentato un esempio chiarificatore nel caso in cui l'indirizzo impostato sul bus non sia 0x1C00, ma 0x1400:

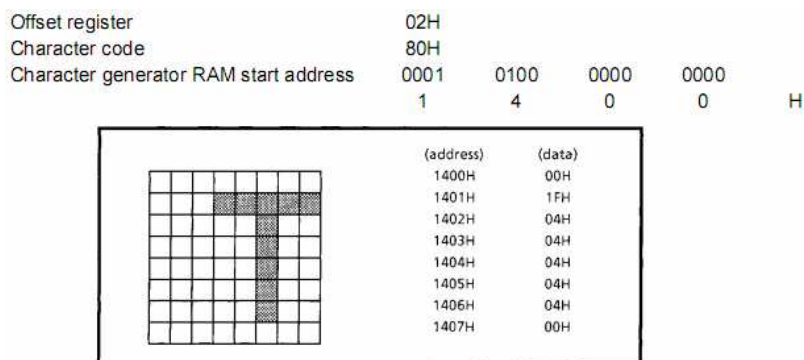


Fig. 14 – Relazione tra offset e indirizzo in RAM

### 1.3. Set address pointer

Questo comando è usato per impostare l'indirizzo iniziale da cui iniziare a leggere o in cui scrivere (*address pointer*).

Per settare l'address pointer, il microcontroller deve seguire il seguente flow chart:

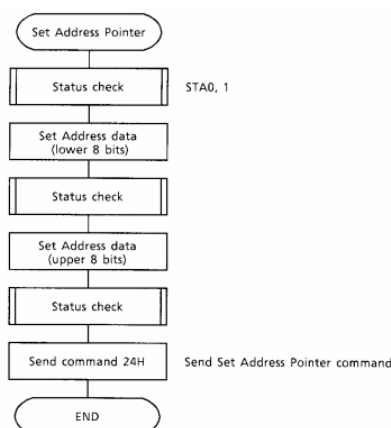


Fig. 15 – Algoritmo per la determinazione dell'indirizzo

## 2. Set control word

Code	Hex.	Function	D1	D2
01000000	40H	Set Text Home Address	Low address	High address
01000001	41H	Set Text Area	Columns	00H
01000010	42H	Set Graphic Home Address	Low address	High address
01000011	43H	Set Graphic Area	Columns	00H

Fig. 16 – Famiglia set control word

### 2.1. Set text home address

Questo comando definisce l'indirizzo iniziale per la text area della RAM, vale a dire che ciò che è allocato dall'indirizzo TH è interpretato dal controller T6963 come testo e viene visualizzato a partire dall'angolo in alto a sinistra e di lì poi a scalare. Ogni indirizzo indica un byte, che rappresenta il character code della lettera che si vuole visualizzare.

Il chip, quindi, usa il character code presente alla locazione indicata, per prelevare il carattere desiderato dalla CGRAM (se si è in modalità internal CGROM e il character code è in 0x00-0x7F, il carattere visualizzato non è prelevato dalla RAM, ma dalla ROM).

Esiste ovviamente una corrispondenza tra l'indirizzo in RAM e la posizione sul display. Tale corrispondenza è evidenziata in figura:

TH		TH + CL
TH + TA		TH + TA + CL
(TH + TA) + TA		TH + 2TA + CL
(TH + 2TA) + TA		TH + 3TA + CL
TH + (n - 1) TA		TH + (n - 1) TA + CL

TH: Text home address

TA: Text area number (columns)

CL: Columns are fixed by hardware (pin-programmable).

Fig. 17 – Corrispondenza tra indirizzi in RAM e posizione sul display

Come esempio, si consideri il caso di una text area che inizia a TH=0x0000, di n=16 linee, di TA=40 colonne. CL è l'offset, in termini di colonne, all'interno della linea (se il numero di colonne è TA=40, tale offset assume valori da 1 a 40, a seconda di quale colonna visualizzare in quella linea). Ogni spazio carattere sul display (di dimensioni verticali fisse 8 pixel e orizzontali configurabili tramite pin) corrisponderà alle locazioni in RAM indicate nella seguente tabella, che riproduce il display dell'esempio:

	1	2	3	4	...	40=TA
0	0=0x0000=TH	1=0x0001	2=0x0002	3=0x0003	...	39=0x0027
1	40=0x0028	41=0x0029	42=0x002A	43=0x002B	...	78=0x004F
...	...	...	...	...	...	...
15=n-1	600=0x0258	601=0x0259	602=0x025A	603=0x025B	...	639=0x027F

La prima linea di caratteri, ad esempio, ha indirizzi che seguono la legge: TH+CL, con CL che va da 1 a TC=40. La seconda linea di caratteri, poi, segue la legge: TH+TA+CL=TH+40+CL, con CL che va da 1 a 40. L'ultima linea di caratteri, infine, segue la legge: TH+(n-1)TA+CL=TH+15TA+CL=TH+600+CL, con CL che va da 1 a TC=40. L'intera area in questione è grande  $n \times TA = 16 \times 40 = 640$  byte.

Si ipotizzi di voler visualizzare un carattere in una certa posizione, ad esempio seconda linea e terza colonna. Innanzi tutto, bisogna far puntare il chip alla locazione desiderata, che nell'esempio è la locazione 0x002A. Poi si scriverà in questa posizione il byte che contiene il character code del carattere da visualizzare, secondo la logica vista al punto 1.2. Il controller visualizzerà, allora, quel carattere in quella determinata posizione. Infatti, quello che si scrive nella text area (come, d'altra parte, nella graph area) è ciò che si visualizzerà sul display nella posizione corrispondente.

In definitiva, un byte scritto nella text area è la codifica numerica del carattere che si vuole visualizzare. Il disegno del carattere viene generato dal controller T6963, andando a consultare la mappa interna dei caratteri. La posizione del carattere sul display dipenderà, poi, dall'indirizzo nella RAM.

## 2.2. Set graphic home address

Il discorso è analogo a quello fatto per il text home address.

In questo caso la corrispondenza tra allocazione in RAM e posizione visualizzata sul display sarà la seguente:

GH		GH + CL
GH + GA		GH + GA + CL
(GH + GA) + GA		GH + 2GA + CL
(GH + 2GA) + GA		GH + 3GA + CL
GH + (n - 1) GA		GH + (n - 1) GA + CL

GH: Graphic home address

GA: Graphic area number (columns)

CL: Columns are fixed by hardware (pin-programmable).

Fig. 18 – Corrispondenza tra indirizzi in RAM e posizione sul display

L'unica differenza rispetto a prima è che il controller non interpreterà il byte indicato in una certa locazione come un character code, ma come 8 pixel successivi da visualizzare (se il bit corrispondente è 1) o da non visualizzare (se il bit corrispondente è 0). In pratica, ora non saranno memorizzati i codici dei caratteri, ma direttamente le informazioni di on/off per i pixel del display.

Ritornando all'esempio del caso precedente, l'unico valore diverso è n: infatti 16 linee di caratteri corrispondono a 128 linee di pixel, quindi n=128. Le coordinate orizzontali sono invece sempre interpretate in termini di colonne.

### 2.3. Set text area

Il numero di colonne del display è configurabile tramite pin. Questo comando adatta il numero di colonne configurate alle colonne per la text area del display.

È comodo, infatti, avere la stessa corrispondenza tra colonne definite dai pin (in hardware) e numero di colonne definite a software con questo comando.

### 2.4. Set graphic area

Il numero di colonne del display è configurabile tramite pin. Questo comando adatta il numero di colonne configurate alle colonne per la graph area del display.

È comodo, infatti, avere la stessa corrispondenza tra colonne definite dai pin (in hardware) e numero di colonne definite a software con questo comando.

## 3. Mode set

Code	Function	Operand
1000X000	OR Mode	—
1000X001	EXOR Mode	—
1000X011	AND Mode	—
1000X100	TEXT ATTRIBUTE Mode	—
10000XXX	Internal Character Generator Mode	—
10001XXX	External Character Generator Mode	—

Fig. 19 – Famiglia mode set

Questi comandi definiscono la modalità di visualizzazione. Tale modalità non cambia finché non è inviato un nuovo comando. Si può visualizzare un or logico, una and logica o un exor logico tra la text area e la graph area.

In figura, a partire da una graph area e da una text area fissate, sono presentati – a titolo esemplificativo – alcuni esempi di visualizzazione:

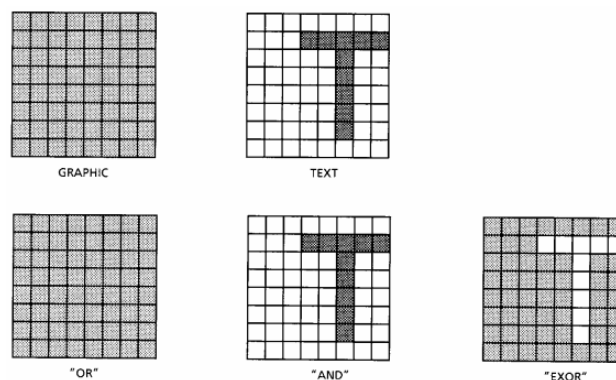


Fig. 20 – Esempi di mode setting

Le *funzioni attributo* possono essere applicate solo alla visualizzazione del testo, poiché il dato di attributo (un byte, chiamato attribute RAM) è allocato nella graph area.

Le operazioni di attributo comprendono la possibilità di mandare il display in reverse, di effettuare un blink sui caratteri o di inibire il display. La word di attributo che contiene informazioni su quale operazione effettuare è scritta nella graph area definita da set graphic home address. Dopo aver inviato il comando text attribute mode, il controller disabilita la visualizzazione della graph area e legge all'inizio della graph area il byte che associa alla attribute RAM (quindi, prima di inviare questo comando bisogna avere cura di scrivere in questo byte l'attributo che si vuole attivare). Gli attributi possibili sono così codificati:

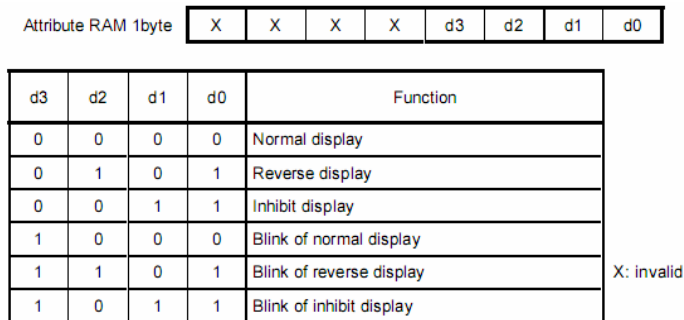


Fig. 21 – Attribute RAM e codifica degli attributi

#### 4. Display mode

Code	Function	Operand
10010000	Display off	—
1001XX10	Cursor on, blink off	—
1001XX11	Cursor on, blink on	—
100101XX	Text on, graphic off	—
100110XX	Text off, graphic on	—
100111XX	Text on, graphic on	—

X: invalid

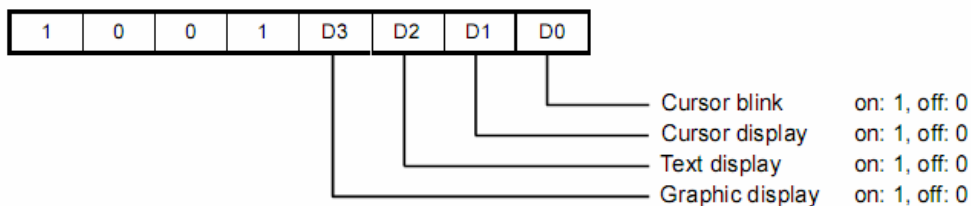


Fig. 22 – Famiglia display mode

Questi comandi gestiscono il modo di visualizzazione del display.

Ad esempio, se si ha necessità di visualizzare sia elementi nella graph area che nella text area, bisogna usare il comando 0b100111XX. Un altro caso in cui è necessario avere entrambe le aree visibili è nel caso in cui si vogliano utilizzare le funzioni attributo (in quel caso, in automatico il controller disattiverà la visualizzazione della graph area).

#### 5. Cursor pattern select

Code	Function	Operand
10100000	1-line cursor	—
10100001	2-line cursor	—
10100010	3-line cursor	—
10100011	4-line cursor	—
10100100	5-line cursor	—
10100101	6-line cursor	—
10100110	7-line cursor	—
10100111	8-line cursor	—

Fig. 23 – Famiglia cursor pattern select

Per  *cursore*  si intende il carattere (che può lampeggiare o meno) visualizzato mentre si inserisce del testo alfanumerico sul display, come guida visiva alla scrittura. Quando è attiva la visualizzazione del

cursore (con i comandi visti nel paragrafo precedente), questi comandi selezionano la stringa di bit da visualizzare nel range 1-8 (gli 8 bit verticali di ogni carattere), che definiscono l'altezza del cursore. L'indirizzo del cursore (che è un elemento grafico visibile sul display) è definito dal comando cursor pointer set.

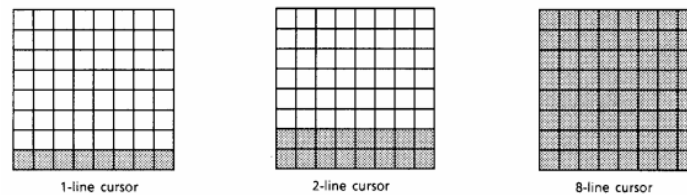


Fig. 24 – Esempio di funzionamento del cursor pattern select

## 6. Data auto read/write

Code	Hex.	Function	Operand
10110000	B0H	Set Data Auto Write	—
10110001	B1H	Set Data Auto Read	—
10110010	B2H	Auto Reset	—

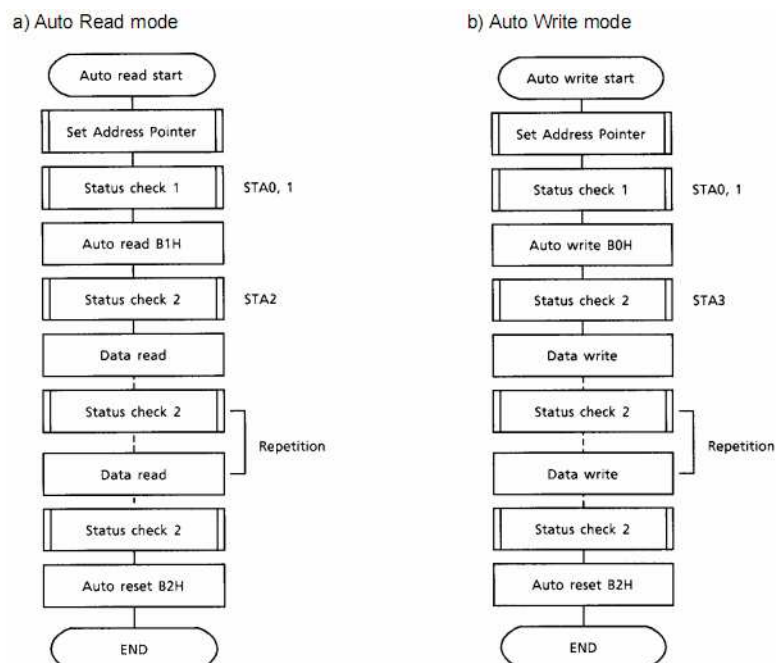
Fig. 25 – Famiglia data auto read/write

È un comando utile per visualizzare dati dalla RAM esterna. Dopo aver settato l'auto mode, non è necessario che un comando di data write o data read sia inviato dopo ogni dato, in quanto ciò è implementato in automatico, fino al reset dell'auto mode.

Prima di iniziare con l'invio di dati in successione o con la lettura di dati in successione è necessario, però, il comando set address pointer, per impostare l'indirizzo da cui iniziare l'operazione. In tal modo, l'address pointer è in automatico incrementato di 1, dopo ogni dato (inviato o ricevuto).

In auto mode, il T6963 non può accettare altri comandi.

In figura sono presentati i diagrammi di flusso per le operazioni di lettura e scrittura in auto mode:



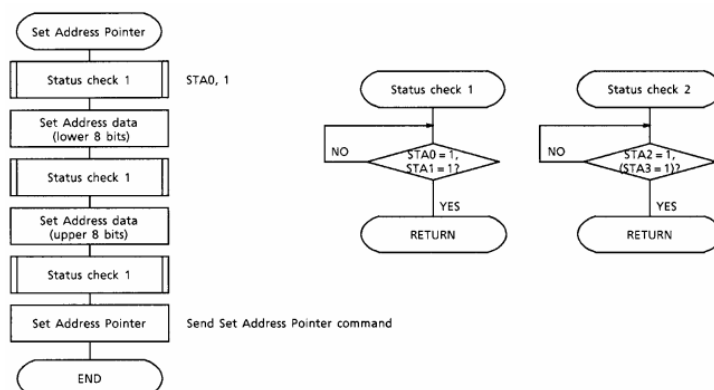


Fig. 26 – Algoritmo del microcontroller per le operazioni in auto mode

## 7. Data read/write

Code	Hex.	Function	Operand
11000000	C0H	Data Write and Increment ADP	Data
11000001	C1H	Data Read and Increment ADP	—
11000010	C2H	Data Write and Decrement ADP	Data
11000011	C3H	Data Read and Decrement ADP	—
11000100	C4H	Data Write and Nonvariable ADP	Data
11000101	C5H	Data Read and Nonvariable ADP	—

Fig. 27 – Famiglia data read/write

Questi comandi sono usati per far scrivere dati nella RAM dal microcontroller o per far leggere i dati contenuti nella RAM dal microcontroller. Questi comandi devono essere eseguiti solo dopo aver settato l'indirizzo, usando il comando set address pointer. Tale puntatore (ADP) può anche essere incrementato o decrementato automaticamente. I dati in questione sono di un byte.

In figura è riportato l'algoritmo da seguire, nel caso esemplificativo di una scrittura con incremento del puntatore:

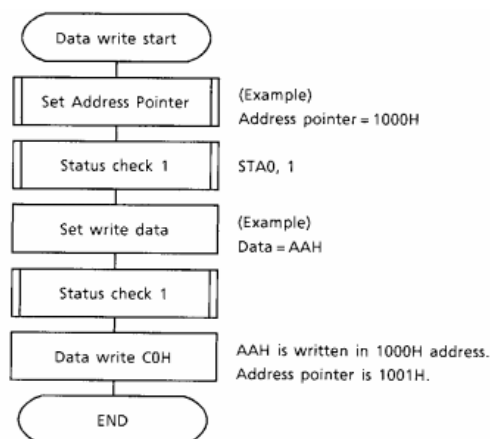


Fig. 28 – Algoritmo del microcontroller per l'operazione di write con incremento del puntatore

## 8. Screen peek

Code	Hex.	Function	Operand
11100000	E0H	Screen Peek	—

Fig. 29 – Famiglia screen peek

Questo comando permette di leggere un byte dell'area grafica, combinando insieme anche l'effetto dell'area di testo: in pratica, il contenuto di un byte dell'area grafica in RAM nulla dice sull'effettivo stato dei pixel sul display, se è attiva anche la modalità testo, in quanto dipende dal carattere visualizzato e dalla



modalità (OR, EXOR, AND) scelta. Mediante questo comando è possibile conoscere l'esatto stato di ogni pixel sul display.

L'indirizzo passato a questo comando deve ricadere nell'area grafica, altrimenti viene settato a 1 il bit del registro di stato STA6 (è bene quindi controllarlo prima del comando).

Tale comando è disponibile solo se il numero di colonne settato in hardware (con i pin MD2 e MD3) è pari a quello settato dal software (con i comandi set text area e set graphic area).

Di seguito è riportato l'algoritmo per tale comando:

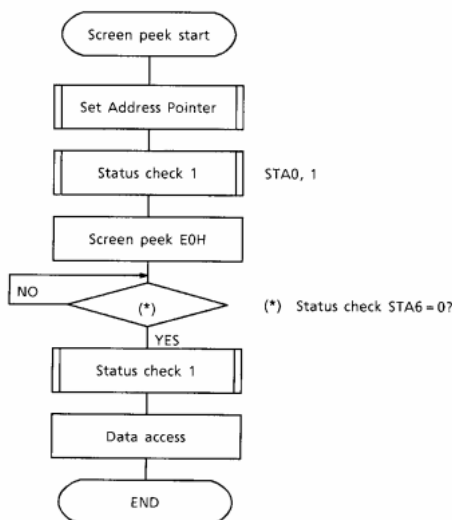


Fig. 30 – Algoritmo del microcontroller per il comando di screen peek

## 9. Screen copy

Code	Hex.	Function	Operand
11101000	E8H	Screen Copy	—

Fig. 31 – Famiglia screen copy

Questo comando copia il risultato della combinazione di testo e grafica (così come visualizzato sul display) nella graph area, a partire dall'indirizzo indicato con set address pointer.

Poiché si fa riferimento alla graph area, questo comando non è disponibile se si stanno usando le funzioni attributo. Tanto meno, questo comando non può essere usato in modalità dual scan. Inoltre, tale comando è disponibile solo se il numero di colonne settato in hardware (con i pin MD2 e MD3) è pari a quello settato dal software (con i comandi set text area e set graphic area).

È bene riferirsi al seguente flow chart:

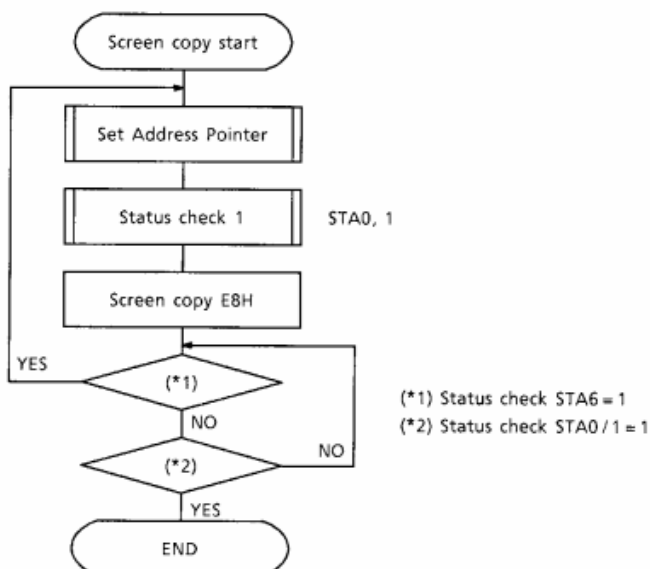


Fig. 32 – Algoritmo del microcontroller per lo screen copy

**10. Bit set/reset**

Code	Function	Operand
11110XXX	Bit Reset	—
11111XXX	Bit Set	—
1111X000	Bit 0 (LSB)	—
1111X001	Bit 1	—
1111X010	Bit 2	—
1111X011	Bit 3	—
1111X100	Bit 4	—
1111X101	Bit 5	—
1111X110	Bit 6	—
1111X111	Bit 7 (MSB)	—

Fig. 33 – Famiglia bit set/reset

Questi comandi settano a 1 o resettano a 0 uno specifico bit nel byte puntato dall'address pointer. Agiscono su un bit per volta.

Ci si riferisce al seguente flow chart:

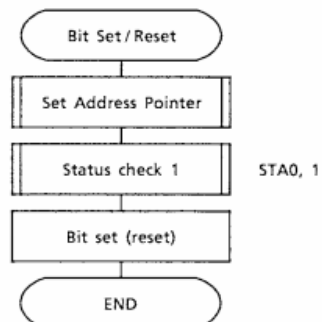


Fig. 34 – Algoritmo del microcontroller per il bit set/reset

**Mappa dei character code in ROM**

Esistono due famiglie di ROM, a seconda dei caratteri di default desiderati. Nella presente mappa sono codificati in 8 bit i 128 caratteri interni alla ROM.

The relation between character codes and character pattern (CG ROM TYPE 0101)

LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

The relation between character codes and character pattern (CG ROM TYPE 0201)

LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Fig. 35 – Mappa caratteri per la ROM del chip

Si nota che i character code non sono quelli ASCII, ma presentano delle analogie che possono aiutare nella scrittura del codice. Ad esempio, i caratteri dallo spazio (0x00) alla tilde (0x5E) sono così determinati:  
codice ASCII-0x0020.

## Riferimenti

Scrivere in un qualsiasi motore di ricerca le parole "T6963 datasheet" e "T6963 application notes" e in poco tempo si accederà a tutto il materiale reperibile on line, usato come fonte principale per la stesura di questo documento.

È stato inoltre molto utile l'articolo di Giuseppe Modugno su "Fare elettronica 264", Giugno 2007: "Il controller T6963C".

## Nota dell'autore

Il presente lavoro vuole essere una piccola guida a chi si appresta per la prima volta all'utilizzo di questo utilissimo chip.

Se il lettore nota errori, imprecisioni o altro, l'autore sarebbe felice di ricevere queste osservazioni per integrarle in una nuova revisione del documento.

Inoltre, tale materiale è reso disponibile gratuitamente e può essere distribuito in qualsiasi forma (anche cartacea) alle seguenti condizioni:

- il materiale dovrà essere distribuito senza alcuna modifica ai contenuti, salvo esplicita autorizzazione;
- niente sarà dovuto da chi lo acquisisce a chi lo distribuisce, salvo rimborso per eventuali spese per il supporto che lo contiene o su cui è stampato.

È possibile contattare l'autore all'indirizzo: [mistersnike@gmail.com](mailto:mistersnike@gmail.com).

Buon lavoro!  
Mario Buffardo