How to Use a Graphic LCD

Winstar Display describes here how to use Graphic LCD modules.

COURTESY BY WINSTAR DISPLAY

As we enter the technological era of the world, many gadgets, devices, appliance, equipments etc. are evolving. And beyond that evolution, as a person, as a human; the one who controls this computer world of us, must know how to employ it of course.

That's why Winstar Display Co., Ltd. is here to give knowledge of what technology is all about to sustain life. As the vision of our company goes: WIN your life; STAR your eyes. So with that, it is a great privilege for us to give information about our product, especially on how to use graphic LCD modules for the benefits of everybody.

So let's see the Winstar product WG240128B as an example.

WG240128A

Control Signals

The whole picture is composing of 240 dots/pixel/character column and 128 dots row as shown to the above photo. But before the whole picture shows, it takes some process first. And that is the responsibility of the controller of every module. This controller will be the one to give signals to the whole module. The different signals are below:

- CP Signal

From the picture, this is the foundation of everything, of the whole image. This represents the single dot or single pixel or single character of the picture.

- LP Signal

This is the one that responsible to change another line if CP Signal occupies the whole row. This is the one that tells the CP to take another line.

- FLM Signal

As the LP Signal finishes its responsibility to tell CP to change the line, FLM will be the one to tell LP to change its page if the line is almost done. So this is the one responsible for the whole page.

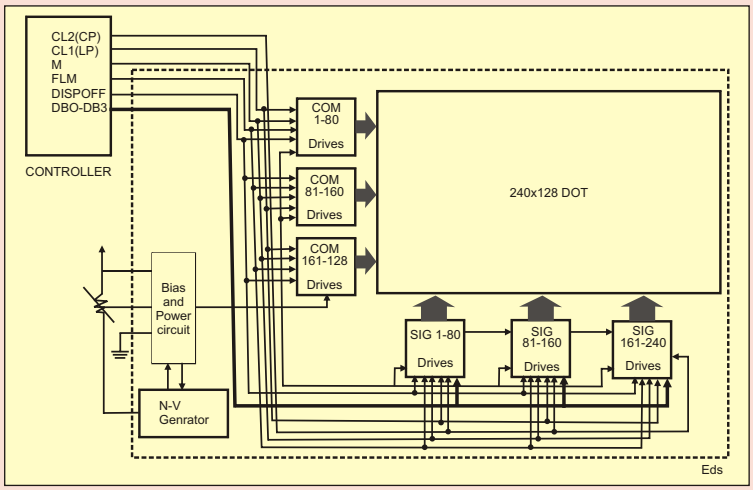
- M Signal

As the character, the line and the page are almost done, M Signal takes place by telling the BIAS to change its voltage so that the LCD (the nematic itself) will turn or change.

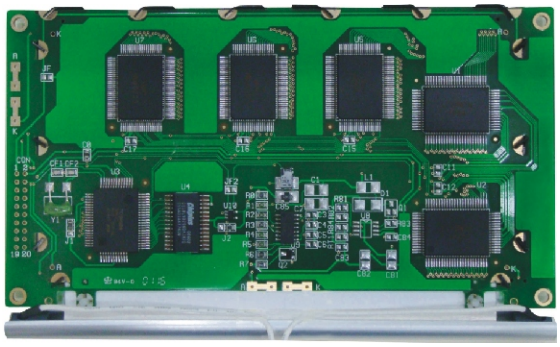
Block Diagram of LCD Module

As we know now how each dot in every line and in every page occur, let us see now what is behind on every graphic LCD.

So as you can see the illustration, before all the dots in every line and in every page takes place, there is different process that happens first. The whole picture itself comes from the signal of its COMMON and SEGMENT IC (Integrated Chip) that builds the row and column of every character.



Where in fact, asks their signal from its CONTROLLER and its proper voltages from the BIAS Circuit that will be acted upon by the Negative Voltage Generator to avoid some shortages and burning of other components IC.



Printed Circuit Board of WG240128A

Controller and 8051 IC Application

Aside from the circuitry of the LCD Module, we need also to learn about on how we are going to make real the picture that we want to pop up on the screen itself. How a single character can be put on the actual figure.

First thing we need to find out is the data bus IC (microprocessor) that we are going to use to make the program that will be use on the module. This will be the foundation of all the data that we want to put on the module that we want to utilize and of course we will base our program on the type of data bus IC that we use.

Winstar Display Co., Ltd. is commonly using the 8051 microprocessor of Intel and uses the C Language for its program. This is where we format the entire program that we are using for the different applications of our modules, depending on the capacity of its application.

Winstar takes time also in choosing the accurate Controller IC of the module depending on the function of the LCD or the request of its customer. Programs can be created as we comprehend on the specification of the controller IC that we are using. From that specification, we can know the correct interface of it to its Data Bus and the correct way on how to program and apply the controller.

So with the Controller and Data Bus IC connection, we can now create all the thoughts that we want to be view on our screen.

Let us take a look the sample module of

Feature
1. Built-in controler TOSHIBA-(TE963C)
2. 1/128 duty cycle
3. Built -in N/V (Option)

Pin No	Symbol	Function	
1	FGND	Frame GEN (connected to bezel)	
2	VSS	GND	
3	VDD	Power Supply for logic circuit	
4	VO	Contrast Adjustment	
5	$\overline{WR}$	Data write	
6	$\overline{RD}$	Data road	
7	$\overline{CE}$	Chip enable	
8	C/D	Code/Data	
9	NC/VEE	No connect on/ Negative Voltage output	
10	$\overline{RST}$	Controller reset	
11	DB0	Data bus line	
12	DB1	Data bus line	
13	DB2	Data bus line	
14	DB3	Data bus line	
15	DB4	Data bus line	
16	DB5	Data bus line	
17	DB6	Data bus line	
18	DB7	Data bus line	
19	FS	Font Selection	FS="H", 6x8 character font FS="L", 8x8 character font
20	RV	Reverse	

Mechanical Data

Item	Standard Value	Unit
Module Dimension	170.0x103.5	mm
Viewing Area	132.0x76.0	mm
Dot Size	0.47x0.47	mm
Dot Pitch	0.5x0.5	mm
Mounting hole	162.0x85.0	mm

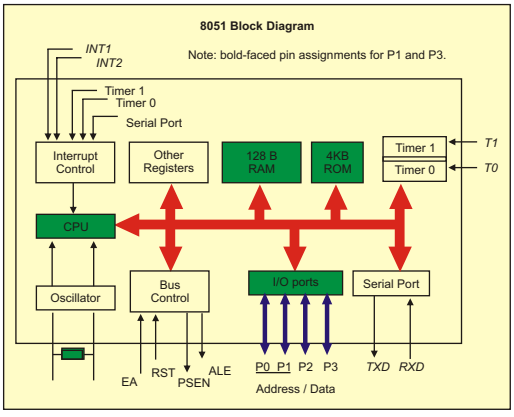
Absolute Maximum Rating

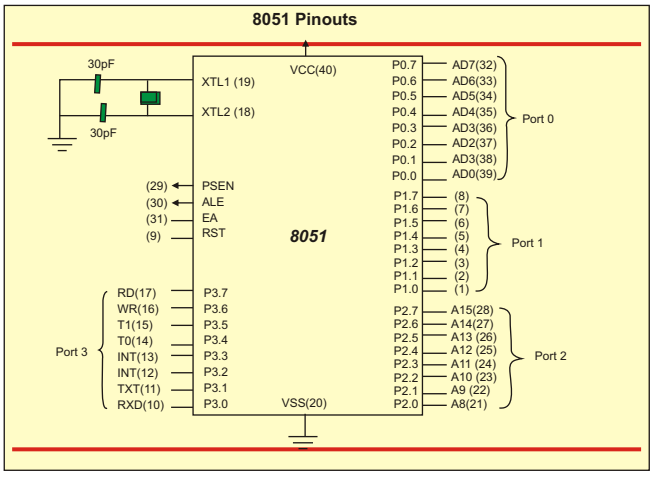
Item	Symbol	Standard Value			Unit
		min.	typ.	max.	
Power Supply	VDD-VSS	4.75	5.0	5.25	v
Input Voltage	VI	-0.3	-	VDD	V

Note:VSS=0 Volt, VDD=5.0 Volt.

Electronical Characteristics

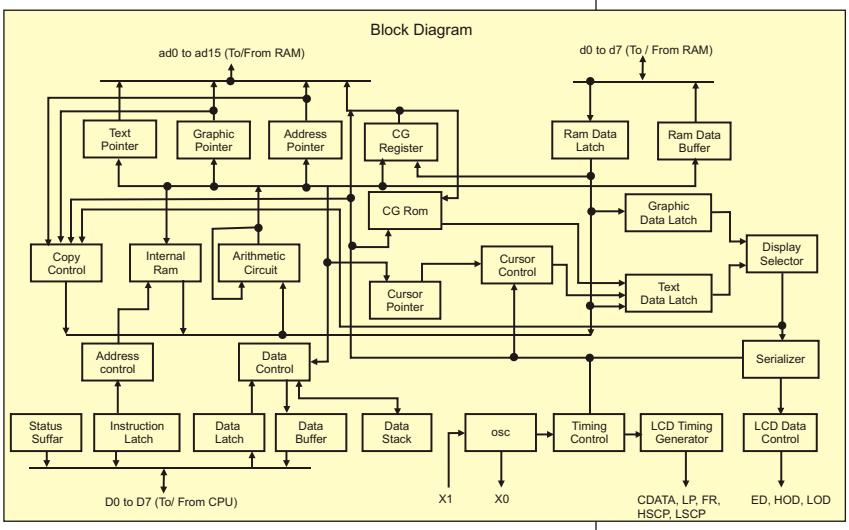
Item	Symbol	Condition	Standard Value			Unit
			min.	typ.	Max.	
Input Voltage	VDD	L Leve	0.7VDD	-	VDD	V
	VIO	H Level	-	-	0.3VDD	V
Supply Current	IDD	VDD=5.0V	-	23	-	mA
Recommended LC Driving Voltage for Normal Temp. Version module	VDD-V0	-20°C	19.1	19.5	20.1	V
		25°C	18.1	18.5	19.1	
		70°C	17.1	17.5	18.1	
LED Forward Voltage	VF	25°C	-	-	-	V
LED Forward Current	IF	25°C		-	-	mA
CCFL	VF	25°C	-	325	580	V
	IF	25°C	-	-	0.0	mA
EL	IEL	Vel=110V/AC, 400Hz	-	-	5.0	mA

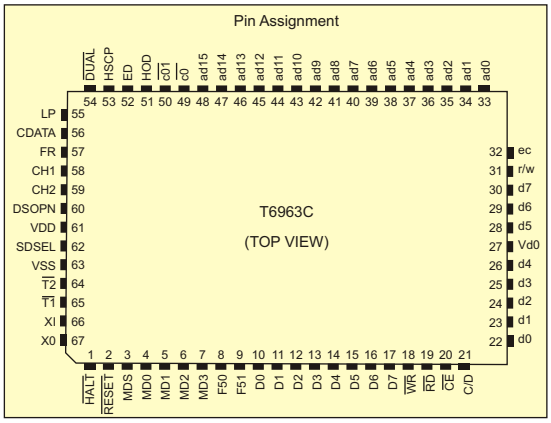




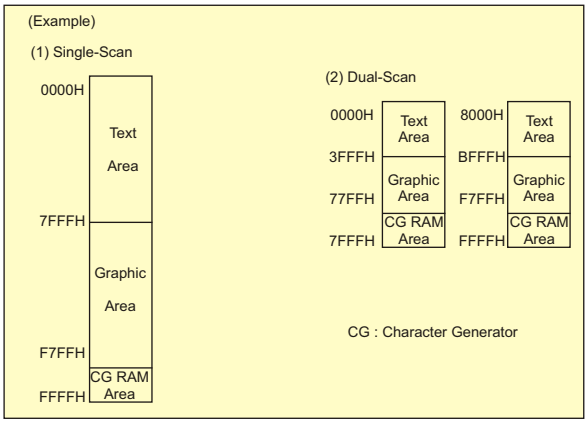
WG240128A on how we really create this one. With the following feature, we can start. As we are aware of its controller, we need to study it for the connection and programming.

TOSHIBA T6963C





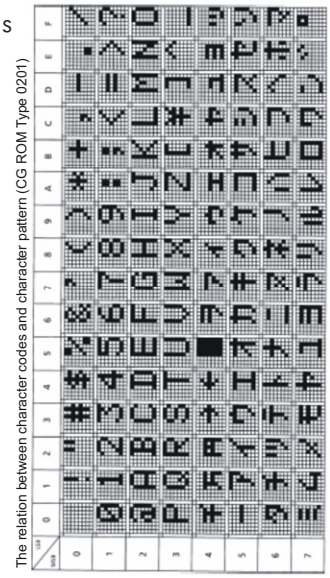
RAM Interface



The external RAM is used to store display data (text, graphic and external CG data). With single-scan, text data, graphic data and external CG data can be freely allocated to the memory

With dual scan, LCD I is allocated to 0000H to 7FFFH (32 KB max), LCD II is allocated to 8000H to FFFFH (32 KB max). Text data, graphic data and external CG data can be freely allocated in LCD I. In LCD II, the same addresses must be allocated as in LCD I, except ad15.ad15 determines selection of LCD I or LCD II. It can be use the address decoded signals ce0 (0000 to 07FFH), ce1 (0800 to 0FFFH) within 4KB. ce0 and ce1 allow decoding of addresses in the ranges (0000 to 07FFH) and (0800 to 0FFFH) respectively within a 4-KB memory space.

To completely create a program that will be use on the LCD module application, we need also to take note of the following:



Note: All of the information can be comprehend by reading all the specifications of the said controller. So if we know that, we can now go on to programming itself.

Command	Code	D1	D2	Funcation
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)

Setting registers

Code	Hex.	Funcation	D1	D1
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

X:invalid

Command Definitions

Command	Code	D1	D2	Funcation
Register 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 adress 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 Bilk 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 Read and Drecrement ADP Data Read and Drecrement ADP Data Write and Nonvariable ADP Data Read Nonvariable ADP
Screen Peek	11100000	-	-	Screen Peek
Screen Copy	11101000			Screen Copy

X:invalid

Creating A Program

Before we actually make a program, lets first take note the port of our microprocessor for connections and compatibility.

So with that, basic knowledge of C language is an important tool so we can go with the flow of the programming. Different operations must be applied such as the +, -, *, /, %, ++, --, >, <, <=, >=, &, ^, ~, >>, << , !, !=, ==. And of course some programming instructions such as the loop instructions (for, while and do while), switching instructions, array, string, etc.

VARIABLE DATA TYPE:

DATA TYPE	BITS BYTES	VALUE RANGE
bit	1	0~1
signed char	8	(-)128 ~ (+)127
unsigned char	8	0 ~ 255
enum	16	(-) 32768 ~ (+)32767
signed short	16	(-) 32768 ~ (+)32767
unsigned short	16	0 ~ 65535
signed int	16	(-) 32768 ~ (+) 32767
unsigned int	16	0 ~ 65535
signed long	32	(-)2147483648 ~ (+)2147483648
unsigned long	32	0 ~ 4294967295
float	32	0.175494E-38 ~ 0.402823E+38
sbit	1	0~1
sfr	8	0 ~ 255
sfr16	16	0 ~ 65535

This one is very important to think of.

SAMPLE GRAPHIC PROGRAM

```
#include        <reg51.h>
#include        <INTRINS.>
#define DATA_BUS        P1
sbit    LCM_FS          = P3^1;
sbit    LCM_MD2   = P3^2;
sbit    LCM_RESET       = P3^3;
sbit    LCM_CE          = P3^4;
sbit    LCM_WR          = P3^6;
sbit    LCM_RD          = P3^7;
unsigned        char    code    picture [128]
[30] = note: code of the picture can be put in here.
void write_command (unsigned char par_1)
{
   LCM_CD       =       1;
   LCM_CE       =       0;
   LCM_WR       =       0;
   _nop_();
   _nop_();
   DATA_BUS     =       par_1;
   _nop_();
   _nop_();
   LCM_CE       =       1;
   LCM_WR       =       1;
}
void write_data (unsigned char par_1)
{
   LCM_CD = 0;
   LCM_CE = 0;
   LCM_WR = 0;
   _nop_();
   _nop_();
   DATA_BUS = par_1;
   _nop_();
   _nop_();
   LCM_CE = 1;
   LCM_WR =1;
}
void init_T6963C ()
{
   LCM_RESET = 0;
   _nop_();
   _nop_();
   _nop_();
   LCM_RESET = 1;
   _nop_();
   _nop_();
   _nop_();
LCM_MD2 = 1;
   LCM_FS  = 0;
   write_data (0x00);
   write_data (0x00);
   write_command(0x40);
write_data (0x00);
   write_data (0x02);
```

```
    write_command(0x42);
    write_data (0x1E);
    write_data (0x00);
    write_command(0x41);
    write_data (0x1E);
    write_data (0x00);
    write_command(0x43);
    write_command (0x80);
    write_command (0x98);
}
void clear ()
{
    unsigned char i, j;
    write_data (0x00);
    write_data (0x20);
    write_command (0x24);
    write_command (0xB0);
          for (i=0; i<=128; i++)
          {
                for (j=0; j<=30; j++)
                {
                      write_data (0x55);
                }
          }
          write_command (0xB2);
    }
void fill_screen (picture)
char picture [128] [30];
{
    unsigned char i, j;
    write_data (0x00);
    write_data (0x20);
    write_command (0x24);
}

void main ()
{
    init_T6963C ();
    clear ();
    fill_screen (picture1);
    while ();
    {
    }
}
```

With these article, hope it would be a great help for the user who wants to be aware of using the Graphic LCD Module. If you need further information, please contact with Winstar Display Co. Ltd., Web : www.winstar.com.tw