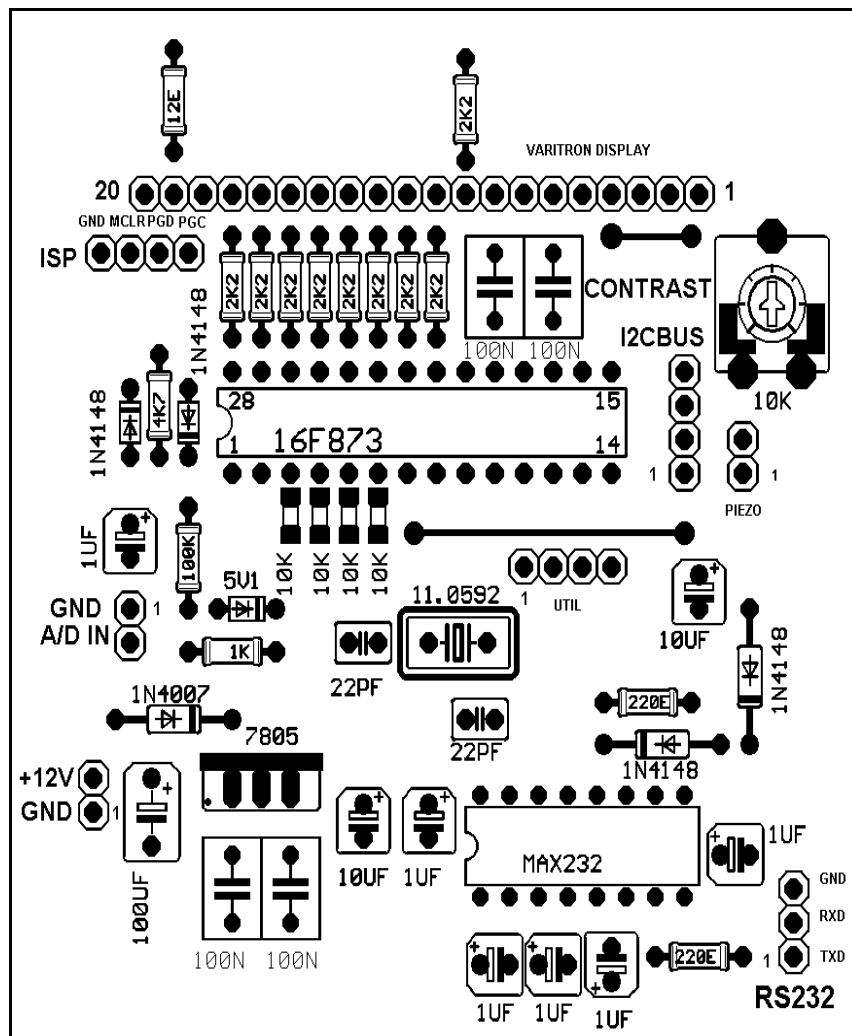


Jan van Gelder (PA3GDB) VARITRON LCD ROUTINE'S :

This is a short explanation in poor English of the C functions which can be used with a 128 x 64 varitron display which make the use of a TOSHIBA TC6963C graphic LCD driver . With those functions it is easily to make text, lines, boxes , bar graphics and signal meters . For more detailed information see the TC6963C datasheet of Toshiba.

Line numbers and pagenumber's Lcd commands etc. are defined in LCD873.h.

Code can also be used with 16F872 , 16F876 and the 16F874.



Some code examples and schematic diagram are also include and can be used to start with.



The printed board which is used to develops the LCD code.

```
void LCDPortIOInit(char portAInit, char portBInit, char portCInit);
```

Must be done before you can use LCDIniSet();

Initialisation of input/output portA, portB and portC.

Predefined values are PortAConfig, PortBConfig and PortCConfig which you can find in the header file LCD873.h.

Values used depends on which port you use for control of LCD_CE, LCD Enable, LCD_RD, LCD_WR or LCD_C_D lines of display.

```
void LCDIniSet(void);
```

Initialisation of Varitron display is done after LCDPortIOInit() is done.

As default the display is in XOR mode which means that text in graphic part of the display will be inverted displayed.

```
void LCDClearTextRAM(void);
```

Clear all pages of text ram in one time.

```
void LCDClearGraphicRAM(void);
```

Clear all pages of graphic ram in one time.

```
void LCDClearTextPage(char pageNumber);
```

Clear a selected text page.

The display text memory is divided in 8 text pages.

Values which can be used for text pages are 1 to 8.

```
void LCDClearGraphicPage(char pageNumber);
```

Clear a selected text page.

The display graphic memory is divided in 7 graphic pages.

Values which can be used for graphic pages are 1 to 7.

```
void LCDSetOrClearPixel(unsigned int lineNumber, char pixelNumber, char setOrClear);
```

Set or Clear a pixel in display or a graphic page where is pointed to.

Pixelnumber is pixel on line where is pointed to.

SetOrClear is a predefined value SETPIXEL for setting a pixel and CLEARPIXEL for clearing a pixel.

```
void LCDSetAdresPointer(unsigned int lineNumber);
```

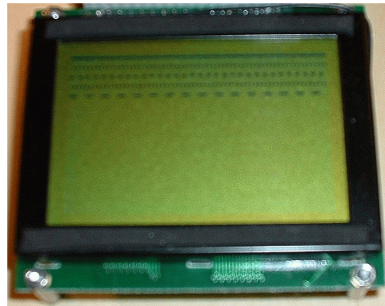
Set de address pointer to the next line where the next writing is done to.

```
void LCDputchar(char data);
```

Put a character on display or in text page where the address pointer is pointed to.
A automatic increment of the address pointer is done after each writing.

```
void LCDIncAdresPoint1Line(void);
```

Increment the address pointer exactly one line.



```
void LCDDrawHorLine(char lineShape, char lineSize);
```

Draws a horizontal line on display or in page where is pointed to.
lineShape is a predefined value.

Predefined values which can be used for lineShape are :

SOLID _____

DOTS1

DOTS2

DASH1 _ _ _ _ _

DASH2 _ _ _ _

CLEAR for clearing of a line.

lineSize give the size of the line which is a multiple of 8pixels.
So the maximum value of a complete line is 16.

```
void LCDDrawVerLine(char lineShape, char lineSize, char bitPosition);
```

Draws a vertical line with difference predefined line shapes.

Line shape is a predefined value.

Value which can be used for line shape are:

Predefined values which can be used for line shape are:

SOLID

DOTS

DOTS

DASH

DASH

CLEAR for clearing of a line.

Line-size is the vertical size of the line in pixels.

Bit position is the pixel position of the vertical line from the left of the display which can have a value between 0 and 127 .

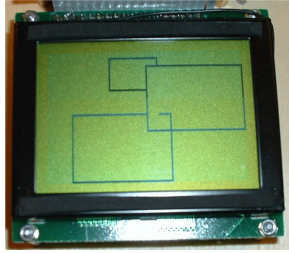
Example :

First you must define on which the line must start
(line starts drawing start from top to bottom)

```
LCDDSetAdresPointer(LINE4_PAGE1G);
```

The vertical line will start on Line 4 on graphic page1.

When bitPosition is 63 the line will be drawn in the middle of the Display.



```
void LCDDrawSquare(unsigned int lineNumber, char sizeHor, char sizeVer);
```

Draws a square on the display kept in mind that size may not be bigger then a line is or that the distance from the sides is big enough .

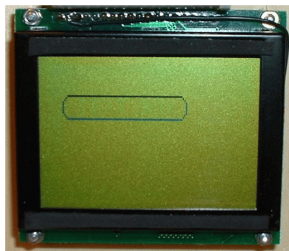
lineNumber is the line where the top of the square is drawn from.

sizeHor = horizontal size in pixels.

sizeVer = vertical size in pixels.

```
void LCDEraseSquare(unsigned int lineNumber , char sizeHor, char sizeVer);
```

To erase a square from the display the lineNumber , sizeHor and sizeVer must be the same a given by drawing of the square.



```
void LCDDrawTextBox(unsigned int lineNumber , char sizeHor, char sizeVer);
```

Draws a square with rounded edges on the display kept in mind that size may not be bigger then a line is or that the distance from the sides is big enough .

lineNumber is the line where the top of the square is drawn from.

sizeHor = horizontal size in pixels.

sizeVer = vertical size in pixels.

```
void LCDEraseTextBox(unsigned int lineNumber , char sizeHor, char sizeVer);
```

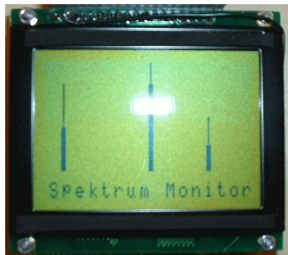
To erase a text box from the display the lineNumber , sizeHor and sizeVer must be the same a given by drawing of the textbox.

```
void LCDSelectGraphicPage(char pageNumber);
```

The graphic page which will be displayed on the LCD.
Values which can be used for pageNumber are 1 to 7.

```
void LCDSelectTextPage(char pageNumber);
```

The graphic page which will be displayed on the LCD.
Values which can be used for pageNumber are 1 to 8.



```
void LCDDrawVerticalBar(unsigned int lineNumber, char pixelNumber, char valueToDraw);
```

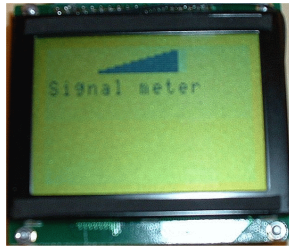
lineNumber is the top of the vertical bar.

From which pixelNumber from the left the bar picture is made on the display.

valueToDraw is the value which the bar displays.

valueToDraw will be divided by 5 .

The maximum value to display is 255 and is $255/5 = 51$ pixels high.



```
void LCDDrawSignalMeter(unsigned int lineNumber, char pixelNumber, char valueToDraw);
```

LineNumber is the top of the Signalmeter

From which pixelNumber from the left the Signalmeter picture is made on the display.

valueToDraw is the value which the signal meter display's.

The value will be divided by a defined value SIGNAL_METER_STEPSIZE
which is default 22 .

So a value with a maximum value of 255 will have a 11 pixels high picture with 11 steps.

```
void LCDBlinkingText(int blinkOn_OffTime, char howManyBlinks,char pageNumber);
```

Let the text on the display blinks on and off.

blinkOn_OffTime = time in msec and is the $\frac{1}{2}$ duty cycle of one on/off period.

howManyBlinks = how many times must the text go on and off.

pageNumber = pagenumber which is displayed on the display.

```

/*****
* File name : lcd_examples
* Description : examples how to make use of the LCD functions in lcd873.c
*
*
*
*****/
#include "Lcd873.c"

void main(void);

//////////////////////////////////////
//      TEST ROUTINE' S to test LCD  code      //
//////////////////////////////////////

void main(void)
{

char i;
char j;
const char *text = "Graphic LCD      128 x 64      T6963C  ";
const char *text3 = "Signal meter";
const char *text4 = "Spektrum Monitor";
const char *text5 = "TEXT BOX";

LCDPortIOInit(PortAConfig,PortBConfig,PortCConfig);
LCDIniSet();

LCDClearTextRAM();
LCDClearGraphicRAM();

AGAIN:

// example how to write a predefined string to display

LCDSetAdresPointer(LINE1_PAGE1T);

for(i = 0; text[i] != 0 ;i++)
{
    LCDputchar(text[i]);
}

delay_s(2);

```

```

// let text blinks on display

LCDBlinkingText(1600,20,1);

// example how to clear text page1

LCDClearTextPage(1);

// write the string "Signal meter" to display

LCDSetAdresPointer(LINE3_PAGE1T);

for(i = 0; text3[i] != 0 ;i++)
{
    LCDputchar(text3[i]);
}

// example of a signalmeter with different value' s

for( i = 0 ; i < 250; i ++)
{
    LCDDrawSignalMeter(LINE3_PAGE1G + 4 ,1,i);
    i ++;
}

for( i = 250 ; i > 3; i --)
{
    LCDDrawSignalMeter(LINE2_PAGE1G + 4,1,i);
    i --;
}

// clear all text ram

LCDClearTextRAM();

// set address pointer to graphic page 1 line number 2

LCDSetAdresPointer(LINE2_PAGE1G);

// draw a horizontal solid line on the display 16 chars wide

LCDDrawHorLine(SOLID,16);

```

```

// set address pointer to graphic page 1 line number 6

LCDSetAdresPointer(LINE6_PAGE1G);

// draw a horizontal line with dots on the display 16 chars wide

LCDDrawHorLine(DOTS1,16);

LCDSetAdresPointer(LINE10_PAGE1G);

LCDDrawHorLine(DASH1,16);

LCDSetAdresPointer(LINE14_PAGE1G);

LCDDrawHorLine(DOTS2,16);

LCDSetAdresPointer(LINE18_PAGE1G);

LCDDrawHorLine(DASH,16);

// wait 2 seconds before the next command

delay_s(2);

// example of how to draw a spectrum display with vertical bars
// set address pointer where to write the string Spectrum Monitor

LCDSetAdresPointer(LINE8_PAGE1T);

for(i = 0; text4[i] != 0 ;i++)
{
    LCDputchar(text4[i]);
}

// example which draws bars with different values

j = 2;

for( i = 0; i < 128 ;i++)
{
    LCDDrawVerticalBar(LINE1_PAGE1G,i,j);
    j = j + 2;
}

```

```
delay_s(2);

// clear all graphic Ram

LCDClearGraphicRAM();

LCDDrawVerticalBar(LINE1_PAGE1G,64,32);
LCDDrawVerticalBar(LINE1_PAGE1G,65,48);
LCDDrawVerticalBar(LINE1_PAGE1G,66,32);

delay_s(1);

LCDDrawVerticalBar(LINE1_PAGE1G,100,64);
LCDDrawVerticalBar(LINE1_PAGE1G,101,128);
LCDDrawVerticalBar(LINE1_PAGE1G,102,64);

delay_s(1);

LCDDrawVerticalBar(LINE1_PAGE1G,10,100);
LCDDrawVerticalBar(LINE1_PAGE1G,11,200);
LCDDrawVerticalBar(LINE1_PAGE1G,12,100);

delay_s(1);

LCDDrawVerticalBar(LINE1_PAGE1G,64,200);
LCDDrawVerticalBar(LINE1_PAGE1G,65,250);
LCDDrawVerticalBar(LINE1_PAGE1G,66,200);

delay_s(1);

LCDDrawVerticalBar(LINE1_PAGE1G,64,32);
LCDDrawVerticalBar(LINE1_PAGE1G,65,48);
LCDDrawVerticalBar(LINE1_PAGE1G,66,32);

delay_s(2);

LCDBlinkingText(800,20,1);

delay_s(4);

LCDClearGraphicRAM();
LCDClearTextRAM();

LCDDrawTextBox(LINE14_PAGE1G + 1,10,12);

LCDSetAdresPointer(LINE3_PAGE1T+2);
```