Using a Nokia Colour gLCD with the PICAXE

1. Introduction

Nokia produces a variety of mobile (cell) phones which utilise a simple graphic LCD (gLCD) which may be used in various PICAXE projects.

Previously the author has posted information on a series of monochrome gLCD's which have been removed from Siemens A55 (102 x 65 pixel) and Nokia 1100/2280 (96x65 pixel) mobile phones.

This presentation covers the use of a relatively common 128 x 128 pixel resolution 4096 colour gLCD. Those use by the author have been obtained from Nokia 6610 mobile phones but the same gLCD modules are used in 6610i and 6100 series phones amongst others.

2. The Hardware

If you have access to a complete mobile phone there is every change that you have all of the hardware required to connect and use the gLCD.

There are Ebay sellers who can sell you the bare 128 x 128 pixel 4096 colour gLCD but then you need to find a way connect your wires to the gLCD module which uses a very small 10 pin Hi-Rose connector.

If you do not have the original mobile phone circuit board then a breakout board whether purchased or home built is essential. Unlike most of the monochrome gLCD modules from mobile phones there is almost no possibility of directly connecting wires to the 10-in Hi-Rose connector.

2.1 Breakout Boards

Unfortunately, breakout boards purchased on their own are hard to source.

2.1.1 TPS (India)

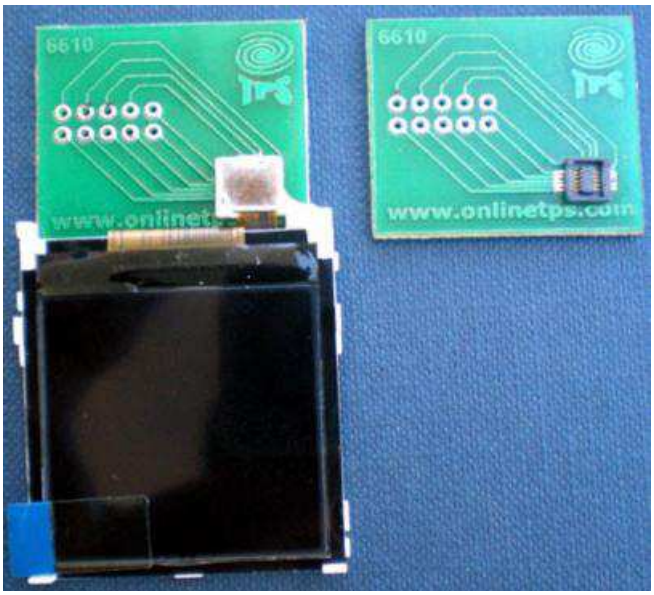
I did find an electronics store (TPS) in India who sell a breakout board for AUD$1.00 but although state that they welcome International customers, TPS seemingly are not sell/shipping to Australia and possibly other regions. This board is supposedly compatible with the displays from the following Nokia phones:

- Nokia 3100
- Nokia 3120
- Nokia 5100
- Nokia 6100
- Nokia 6610
- Nokia 7210
- Nokia 7250

The Indian website, which was current as at 4 Jan 2012, is:
http://www.onlinetps.com/shop/index.php?main_page=product_info&cPath=75&products_id=616

I have been communicating with TPS to see if I can purchase a small quantity of the breakout board they sell but while they have confirmed the connector is pre-mounted, they have not responded to requests to consider shipping to Australia. Maybe others will have greater success.

The board comprises only a breakout board with the 10 pin Hi-rose connector fitted as per the image below.



With the TPS offering, one will need to also purchase some added components to construct a 7 Vdc regulated supply for the gLCD backlight. This is not a big problem as Sparkfun on their website include an “open source” schematic for their breakout board complete with the backlight supply circuit.

2.1.1 Sparkfun

Sparkfun sell a breakout board but **only** complete with a Nokia “knock-off” (ie compatible) gCLD for around AUD$50 which is a bit steep when you already have some Nokia mobile phones to extract the displays from.

Sparkfun do however sell the Hi-Rose 10-pin connector on its own for around AUD$3 (the price via Littlebird Electronics in Australia).

An advantage of the Sparkfun board is that it has an onboard DC-DC step up converter to obtain the required 7 V dc for the gLCD backlight.

2.2 Using the Nokia Mobile Phone Board

If you have a complete Nokia mobile phone such as a 6100 or 6610 then you already have the necessary connector and voltage step up circuit.

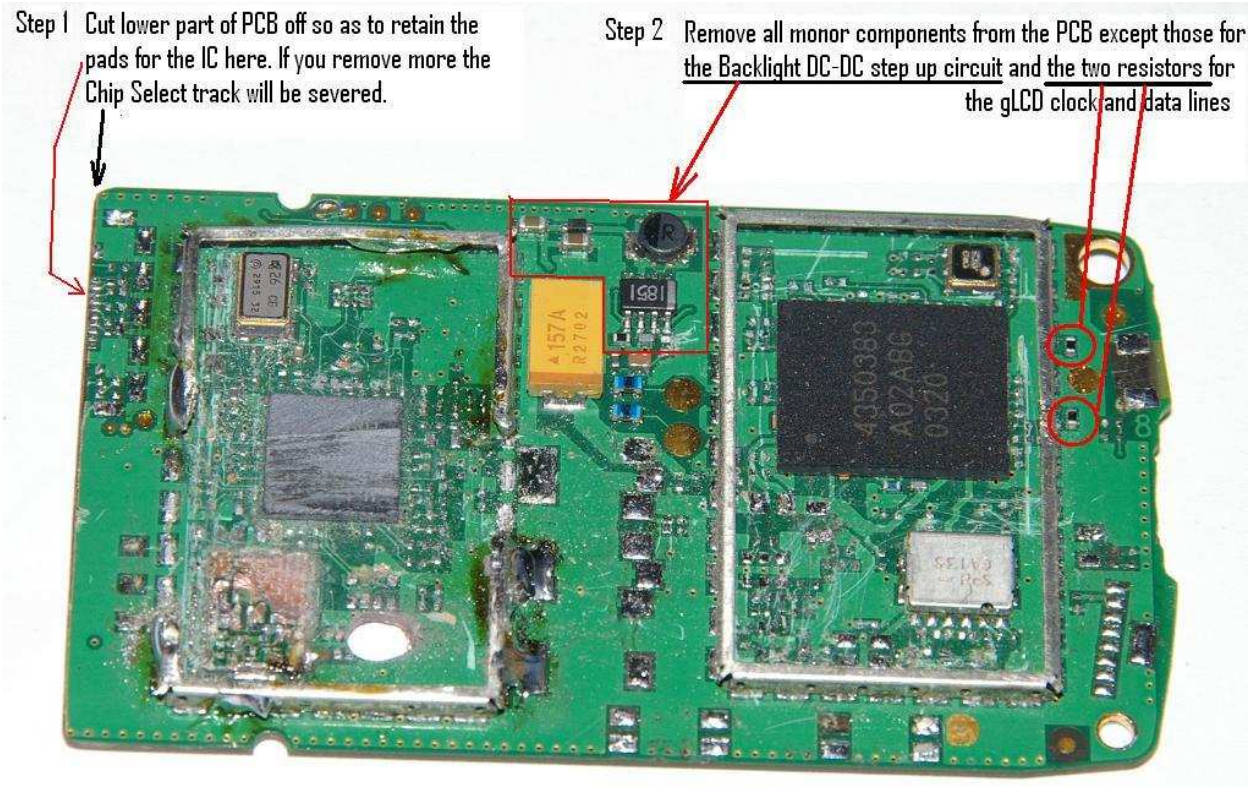
Unless you are very skilled at de-soldering, then do not try to remove the 10-pin Hi-Rose connector from the phone PCB. Instead we can strip down the PCB, cut away some of the excess board and use it as a carrier for the gLCD module.

The following sequence of Photos and commentary should be sufficient to allow you to in effect create you own breakout board with backlight supply.

Step 1 is to cut away the lower part of the PCB. Here we must ensure to cut through one of the main IC’s to retain the Chip Select trace from the gLCD connector to the test pads we will solder wires to.

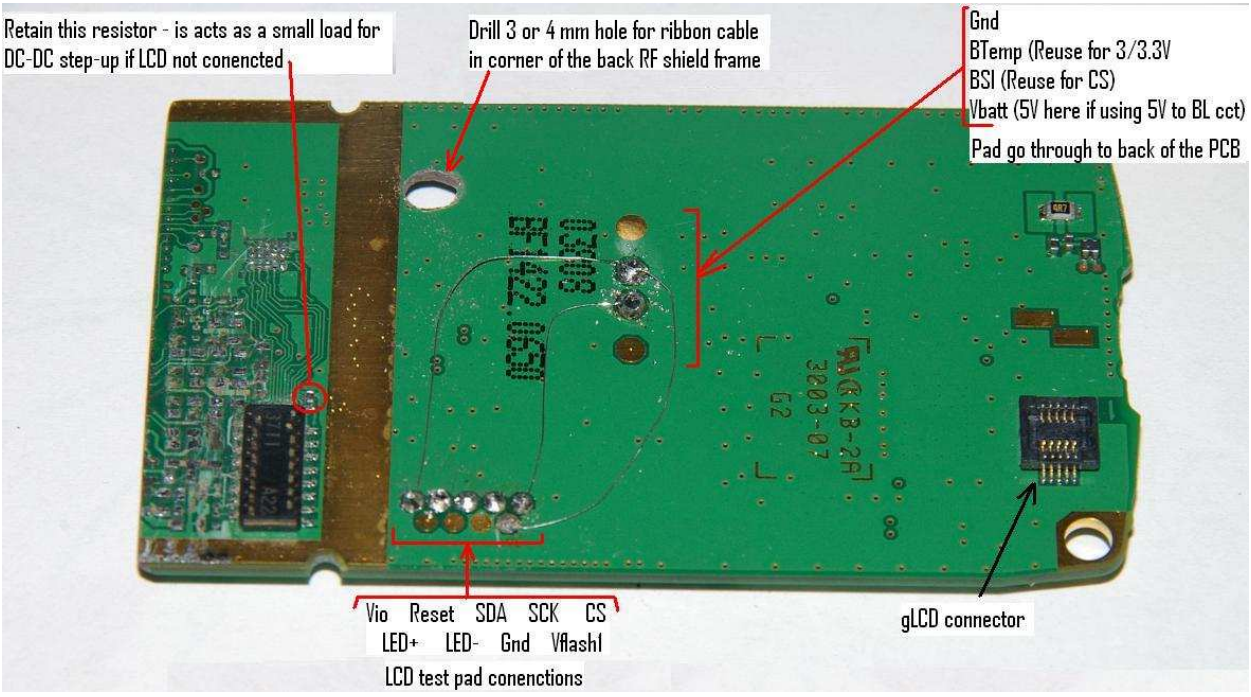
Step 2 is to remove all unnecessary minor components and the battery connector from the board which will prevent the remaining main IC’s drawing power. We do need to retain the few components in the middle part of the retained board for the DC-DC step up circuit and the two series resistors in the circuit to the 10=pin Hi-Rose connector for the gLCD clock and data signals. These areas are highlighted in the below photo.

Note also how I have retained the RF shield surrounds but cut away parts of the surround in the left side of the photo to allow our wires to pass through when added later.



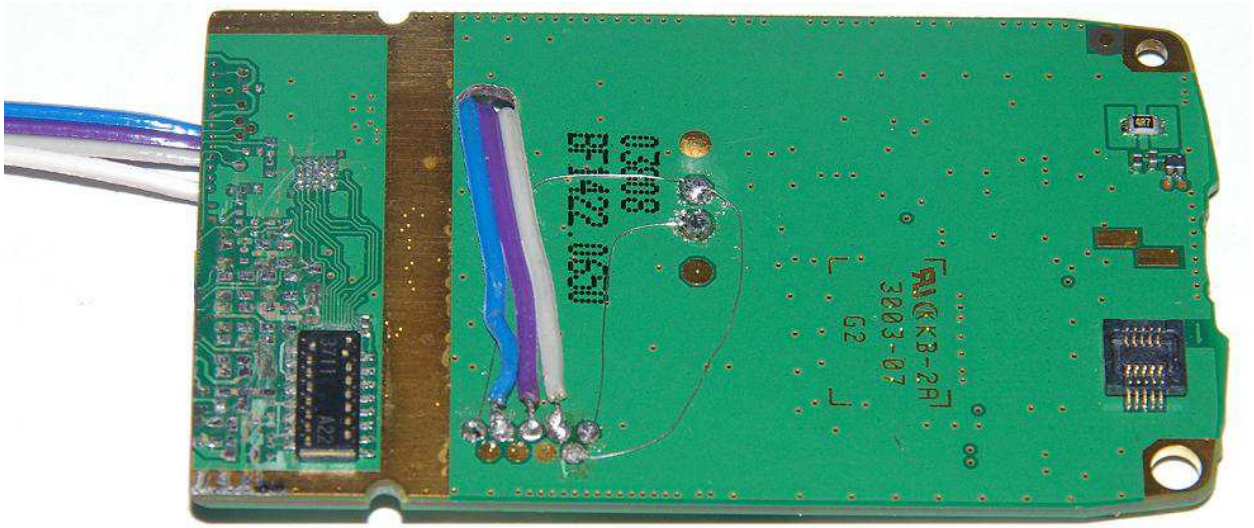
Step 3 is to drill a 3 or 4 mm diameter hole as shown this is in effect in within the RF shield frame to allow a 3-wire ribbon cable to pass through.

Step 4 is to solder some extremely fire wires (use a single bare strand from some hookup wire) on the top side of the phone PCB as shown in the photo below



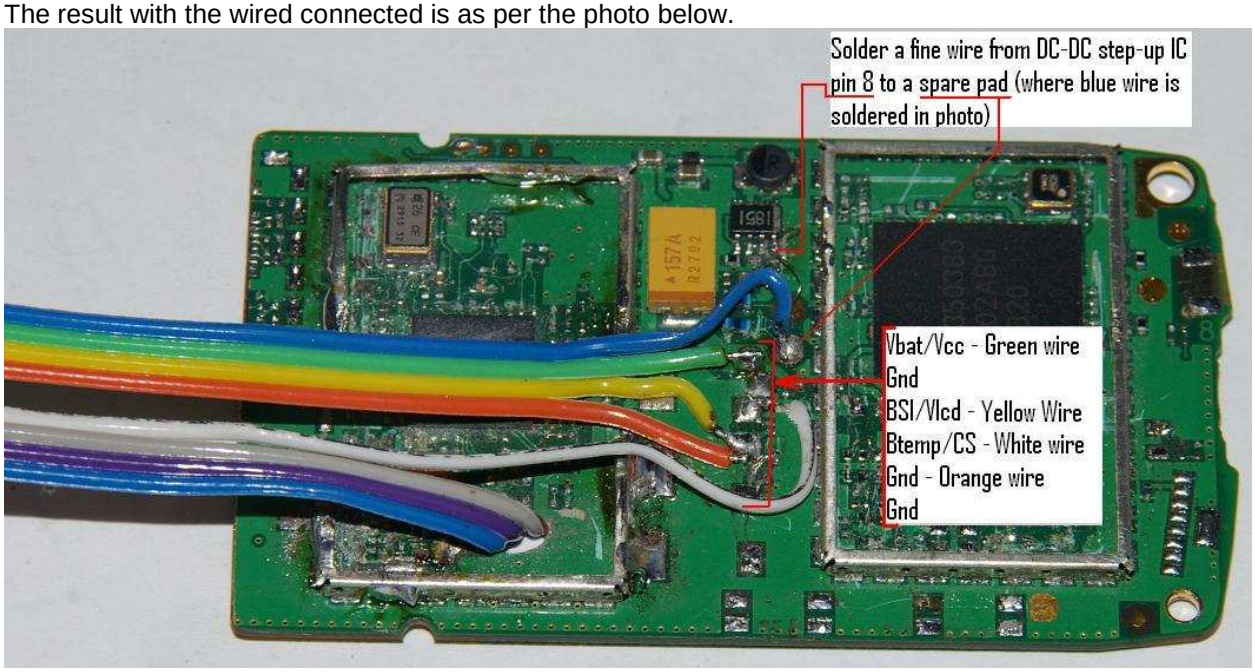
The LED+, LED-, Gnd traces already pass to the back of the PCB as do the four pads in the centre of the board which we use for the 3/3.3 V and Chip Select signals to reduce the number of larger wires we need to route to the back of the board.

Step 5 is to connect some wires to the various pads on the top (LCD) side of the board and route them through the hole already drilled. I used some ribbon cable but certainly the thinner the better as these wires will be between the PCB and the rear of the gLCD module and we should try to keep the gLCD as flat and close to the PCB as possible. I used a piece of 4 strand ribbon cable as ultimately a total of 8 wires are required (3 to the top/gLCD side and 5 to the back of the PCB. See the photo below.



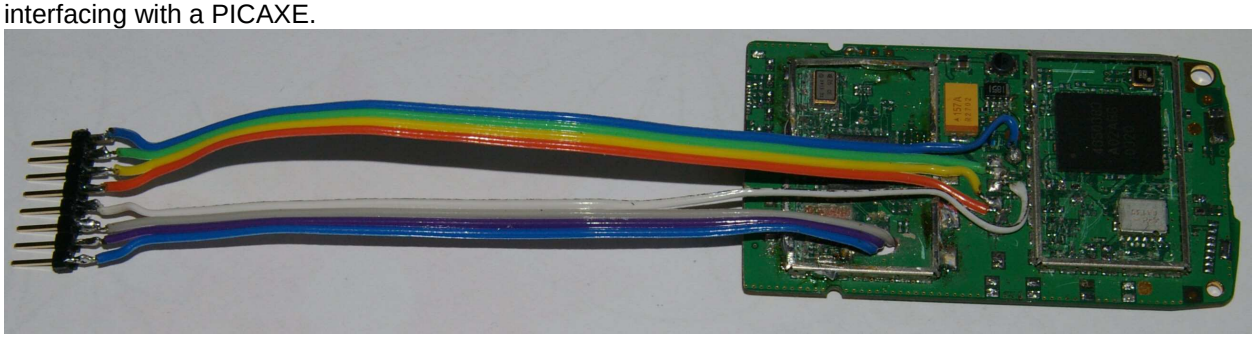
Here we add some ribbon cable via the hole drilled earlier to the test pads for the SCk (white), SDa (purple) and Reset (blue) signals.
The white wire visible to left of the PCB will be conencted to the back of the PCB next.

Step 6 is to then solder the required other 5 required wires to the back of the PCB. Four of these connect to pads where the battery connector was soldered to the PCB. The last is routed to a now spare test pad between the battery connector and the DC-DC step-up circuit.
In addition we need to solder a fine wire (another single strand from a core of hook-up wire will suffice) between the pin8 of the DC-DC step-up regulator chip and the spare test pad where the last ribbon cable core is soldered. I recommend that you do not try to solder the ribbon cable core direct to the IC pin as the contact is very small and there are several bare active earth point within a mm or two of the IC pin. The result with the wired connected is as per the photo below.

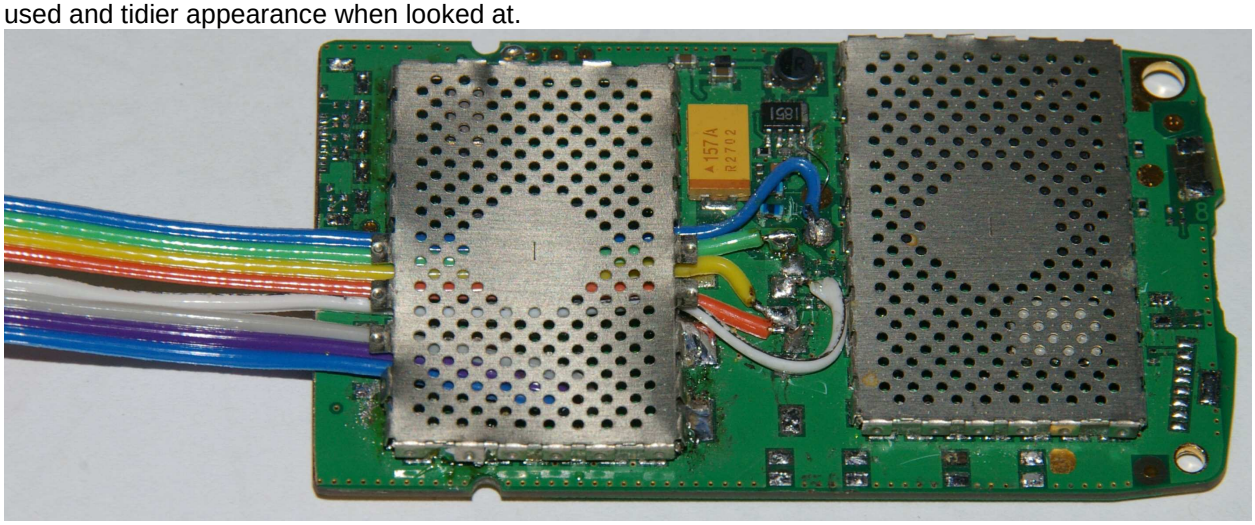


In this example I have allowed to take 5 Vdc directly to the DC-DC step-up regulator for the gLCD backlight rather than draw the extra current from the 3/3.3 Vdc 100 mA regulator used for the gLCD logic and PICAXE interface. For those who are interested, the DC-DC step-up IC is a TK11851L specifically designed as a driver for up to four white LEDs in series.

By soldering some header pins to the end of the ribbon cable we can easily connect to a breadboard for interfacing with a PICAXE.



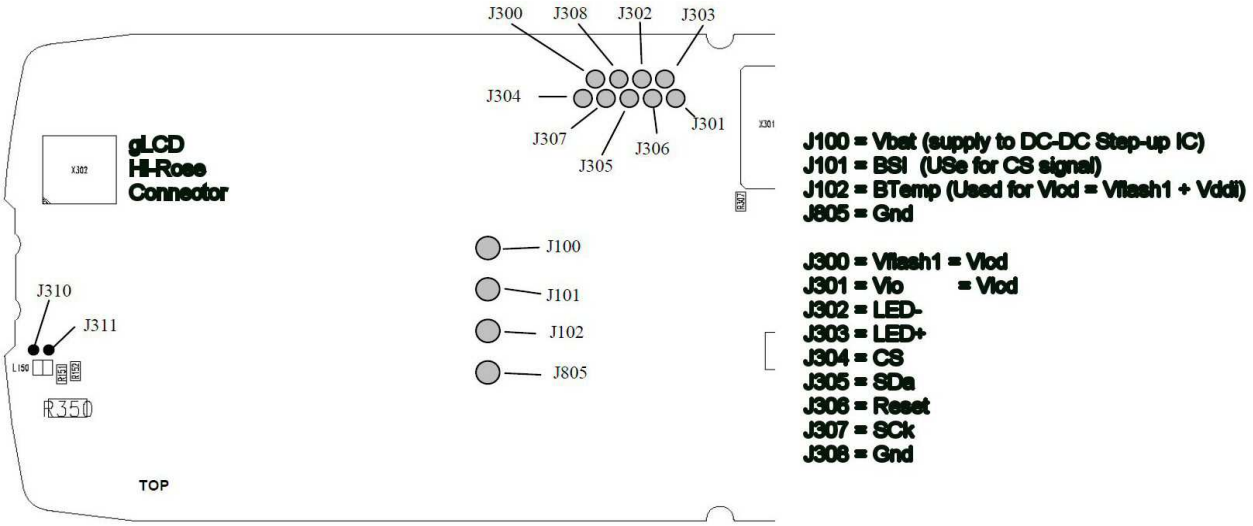
I finally also refitted some of the RF shields so the back side presents a smoother/flatter surface when used and tidier appearance when looked at.



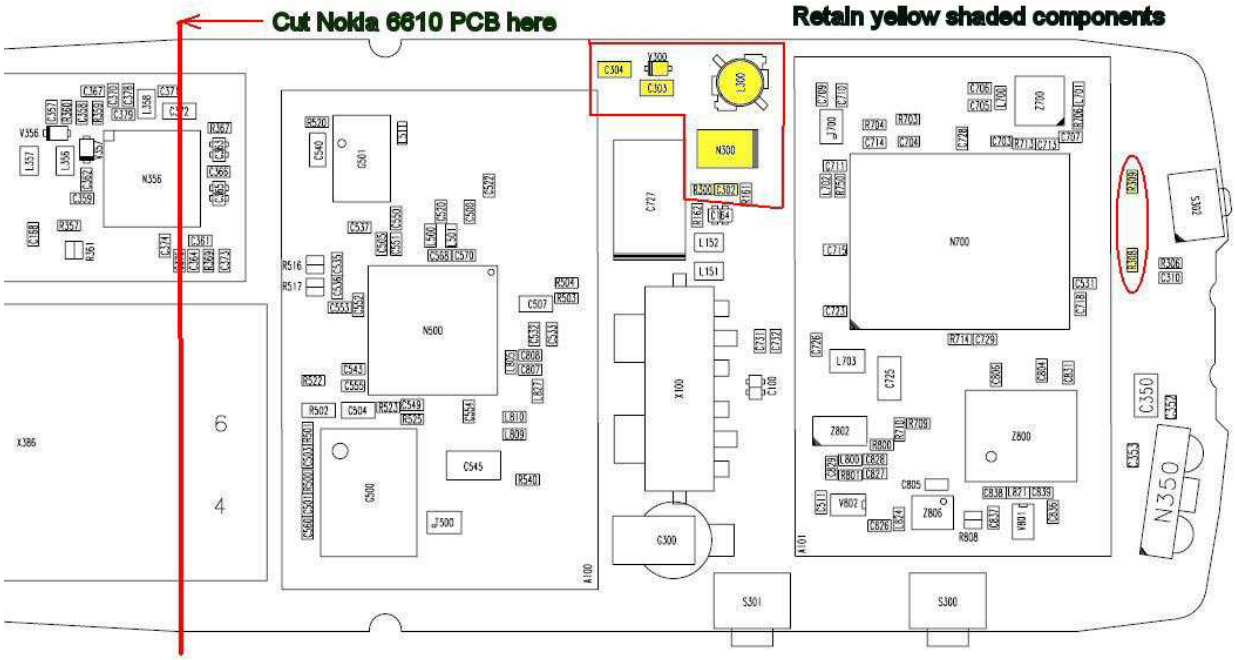
At this point, we are ready to plug in an gLCD display from the old Nokia 6610/6100 mobile phone, assemble an interface with the PICAXE if the PICAXE is operating at a voltage higher than the gLCD logic and then it is time to try some software.

As I tend to operate my PICAXE chips at 5 Vdc the majority of the time, I used a 74LVC245A octal bus transceiver chip (available as SMD SOIC20 chips from Futurlec – www.Futurlec.com.au and www.Futurlec.com)

The image below shows the Nokia 6610 mobile phone PCB layout with the various test points with the test point legend.



The following diagram reflects the bottom side of the Nokia 6610 PCB, the point to cut away the lower/excess board without affecting the Chip Select signal and the components to be retained.
All other components should be remove however if the larger chips are left the removal of resistors and inductors disconnects the power supply from these chips. Ensure you file the cut line clean to ensure no burs on inner PVCB trances than might cause a short to the retained circuitry.



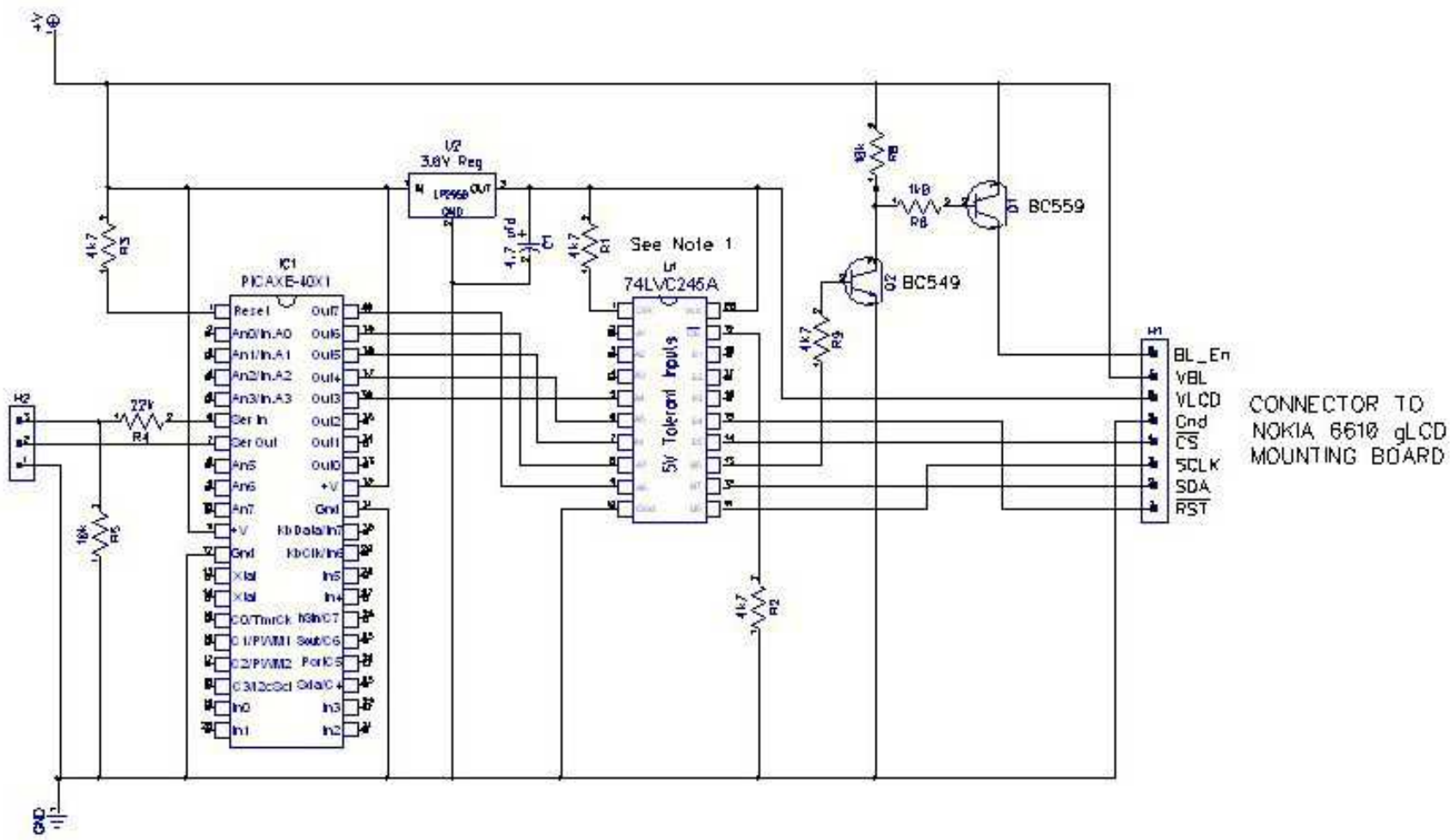
3. SCHEMATICS

The following pages provide some schematic for the circuitry required both to interface the PICAXE to the Nokia PCB or a breakout board and for the DC-DC Step-up Backlight supply circuit.

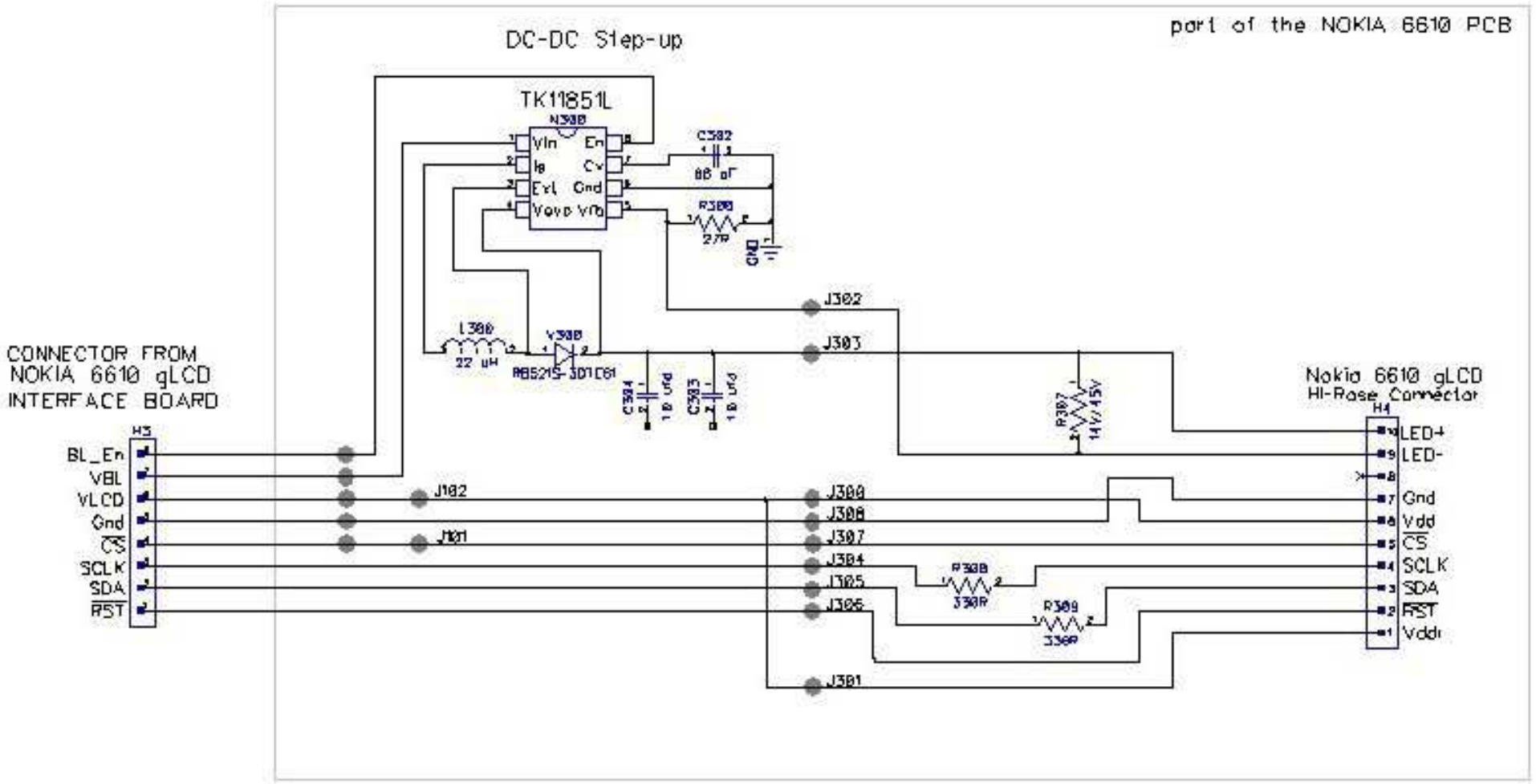
The first schematic below shows the interface circuit which enable the PICAXE to operate at 5 Vdc while the gLCD is powered at 3/3.3 Vdc. If the PICAXE is powered at the same voltage as the gLCD logic circuits then the 74LVC245A octal transceiver chip is not essential.

The second schematic below shows the DC-DC step-up supply for the gLCD backlight and reflects the test points as utilised for our connections on the Nokia 6610 PCB. Note that the DC-DC Step-up IC will operate from a 3/3.3 Vdc input but I have electec to bypass the 3 Vdc voltage regulator to reduce the current drawn through the 100 mA regulator IC.

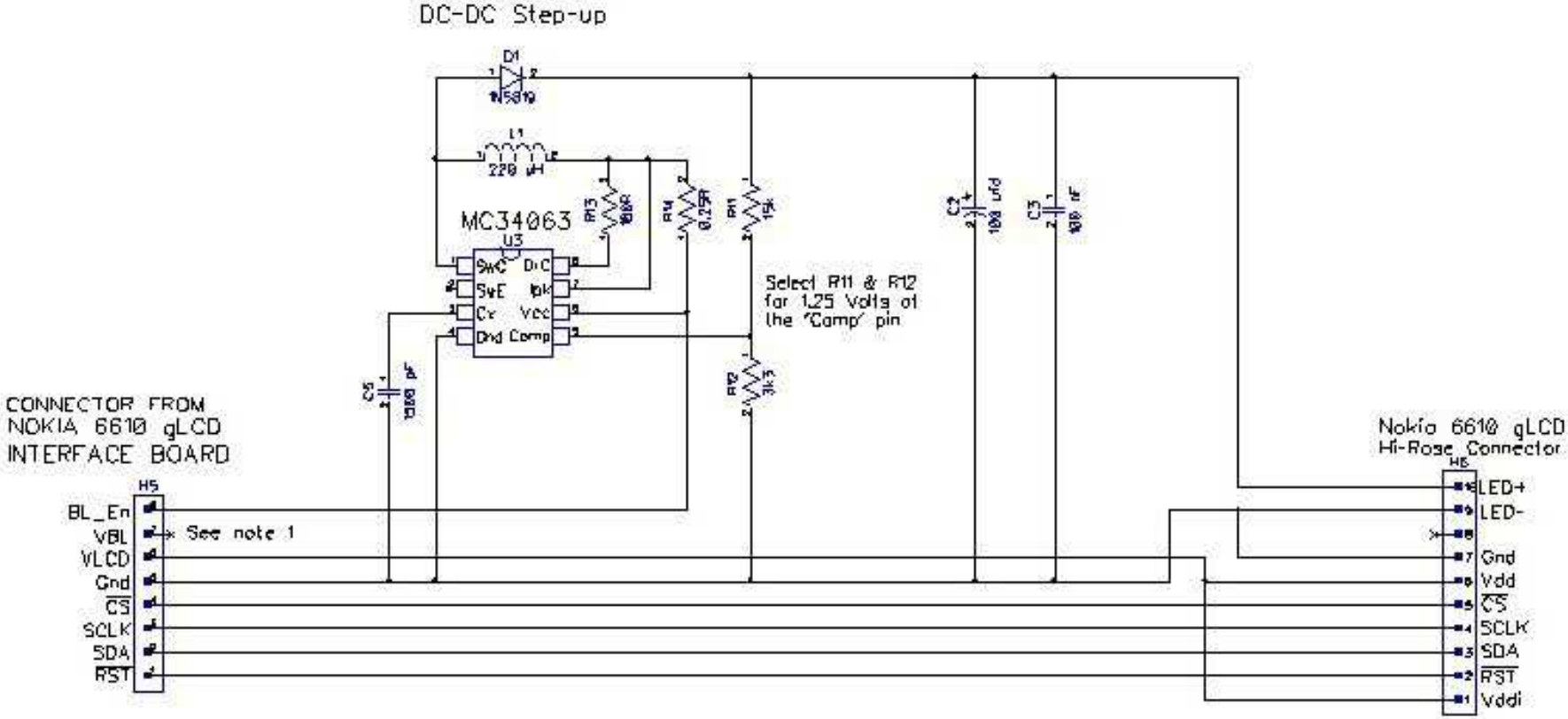
The third schematic is an alternate DC-DC step-up based on the MC34063 1.5 Amp switching regulator.



Notes: 1. If a 74HC245 is used in place of the specified 74LVC245A, then a voltage divider comprising a 6k8 and 10k Ohm resistor (or equal ratio) MUST be used to reduce the PICAXE output to 3 Volt on a High output to protect the gLCD.



● = NOKIA PCB TEST POINT



Notes:

1. By using the BL_En line rather than VBL the PICAXE can be used to turn the gLCD backlight on and off.
2. It is better to operate the DC-DC step-up input from 5 Vdc rather than 3/3.3 Vdc to reduce current draw on the 3/3.3 V reg
3. Indicative CUrrent drawn by backlight circuit:
   @3V = 380 mA @3.3V = 325 mA
   @5V = 120 mA @6.0V = 108 mA

4. Software

4.1 LCD Controller/Driver Chips

There are three different controller chip types used by Nokia for these 128 x 128 pixel 4096 colour gLCD modules.

One is the EPSON S1D15G00 series chip, the second is the Philips PCF8833 chip and the third is the Leadis LDS176 chip. There are apparently some other chips also used in the "Knock-off" replacement displays and reading available information suggests these are compatible with one or other of the chips originally used by Nokia.

While many sources suggest that the controller for the displays can be identified by the colour of the flexible circuit board from the 10-pin connector to the gLCD display I have found that that this is not 100% reliable.

The EPSON controller can have a green or brown flexible circuit at the rear of the display from the actual gLCD to the 10-pin Hi-Rose connector. While many internet sites and other documents suggest only the green flexible circuit is used, I have two gLCD modules taken from Nokia 6610i mobile phones. A seemingly better way based upon the quantity of gLCD's I have obtained directly from Nokia mobiles is:

(a) The flex circuit at the rear is covered with white tape,
(b) The rear of the LCD has a reflective silver covering,
(c) The first 2 characters on the top row of text on the sticker commences with "6P",
(d) There is a red circle on the label with the two lines of text, and
(e) The controller as visible from the front is is a bluish colour and approx 24 x 2.8 mm in area.

The displays with the Philips controller can be distinguished

(a) A brown flexible circuit at the rear of the display from the actual gLCD to the 10-pin Hi-Rose connector,
(b) The rear of the LCD has a white plastic cover,
(c) The first 2 characters on the top row of text printed directly on the rear commences with "7P",
(d) There is a smaller red circle to the right of the two lines of text, and
(e) The controller as visible from the front is a silver colour and approx 19 x 2.0 mm in area.

The displays with the Leadis LDS176 controller can be distinguished by:

(a) A green flexible circuit with very pronounced/darker brown wire traces compared with EPSON based modules,
(b) The flex circuit at the rear is covered with white tape,
(c) The rear of the LCD has a reflective silver covering,
(d) The characters on the top row of text on the sticker commences with "UG13D004. . .",
(e) The number on the second row of text is "4850835 P" and may be followed by " C"
(f) There is a green filled circle of variable size to the right of the two lines of text, and
(g) The controller chip on the front is covered by black tape. If the tape is removed, the controller chip is dark grey in colour and approx 19 x 2.8 mm in area.

Of concern when writing an initialisation code for these Nokia 6610 gLCD modules is that the initialisation commands and parameters are different to get each display type into a visible state and "format" such as 256 colour or 4096 colour.

A note here also relating to the resolution of these gLCD modules. All of the controllers are designed for 132 x 132 RGB resolution and Nokia apparently only used them as 128 x 128 resolution displays. However in fact the gLCD modules can display 130 x 130 pixels and one row/column of pixels around the outer edge is not displayed or visible.

From reading information on the Internet, it would appear that there are also some clone gLCD controllers used in the various available "knock-off" type gLCD displays. Some of these do not have the NOP command (if fact it seemingly may cause the driver to "lock up" until reset) and others may require the use of the display invert and x and Y axis mirroring commands so the display orientation is correct.

The following photo shows some samples of genuine Nokia gLCD modules with each of the three type types of controller chip.

The following sections cover the initialisation and example code for the gLCD's with each of the three gLCD controller chips mentioned above,

4.2 EPSON S1D15G00 Controller

4.2.1 Command Set

The following 4-page table provides a summary of the commands and where required the parameters associated with each command for the EPSON S1D15G00 controller chip. While I have endeavoured to incorporate a lot of data for the more likely parameters, the reader should still download and read the datasheet for the EPSON S1D15G00 controller chip which is readily found by searching the Internet.

In reality, the last 3 commands on the last page are for reading back information and factory testing. As such they would generally never be used, so can be ignored for the majority of cases.

COMMAND	D/C Bit	COMMAND / DATA BYTE								EQUIV HEX	FUNCTION	FOLLOWING PARAMETERS	COMMENTS
		D7	D6	D5	D4	D3	D2	D1	D0				
SLPOUT	0									**$94**	Sleep out	None	Brings the display out of sleep mode.
SLPIN	0									**$95**	Sleep in	None	Places the display in low power sleep mode.
DISNOR	0									**$A6**	Normal display	None	Used to normally highlight the display area without modifying the display data RAM.
DISINV	0									**$A7**	Inverse display	None	Used to inversely highlight the display area without modifying the display data RAM.
DISOFF	0									**$AE**	Display off	None	It is used to forcibly turn the display off.
DISON	0									**$AF**	Display on	None	It is used to turn the display on. Must cancel the sleep mode first
COMSCN	0									**$BB**	Common scan dir.	1 Byte	Used to specify the common output scan direction.
(P1)	1	x[1]	x[1]	x[1]	x[1]	x[1]	P12	P11	P10		When display duty = 1/160: %000 = 1→80 & 81→ 160 %001 = 1→80 & 160→ 81		
DISCTL	0									**$CA**	Display control	3 Bytes	Used to perform the display timing-related setups. This command **MUST** be selected before using SLPOUT. **DO NOT** change this command while display is turned on.
(P1)	1	x[1]	x[1]	x[1]	x[1]	P13	P12	P11	P10		P13-P12 for **CL** dividing ratio P11-P10 for **F1** & **F2** drive pattern. Default is %0000		
(P2)	1	x[1]	x[1]	P25	P24	P23	P22	P21	P20		Drive duty = (Numbers of display lines)/4-1 e.g. 132/4-1 = 32 ; 128/4 – 1 = 31		
(P3)	1	x[1]	x[1]	x[1]	1	P33	P32	P31	P30		FR inverse-set value specify number of lines to be inversely highlighted on LCD panel in range 2-16 lines = (Inversely highlighted lines) –1 e.g. $11 → $11-$01=$10		
PASET	0									**$75**	Page address set	2 Bytes	Used to specify the page address area.
(P1)	1	P17	P16	P15	P14	P13	P12	P11	P10		Start page As addresses are incremented from the start to the end page the column address		
(P2)	1	P27	P26	P25	P24	P23	P22	P21	P20		End page is incremented by 1 and the page address is returned to the start page The relationship "start page < end page" MUST be maintained.		
CASET	0									**$15**	Column addr. set	2 Bytes	Used to specify the column address area.
(P1)	1	P17	P16	P15	P14	P13	P12	P11	P10		Start address As addresses are incremented from the start to the end column the page addr.		
(P2)	1	P27	P26	P25	P24	P23	P22	P21	P20		End address is incremented by 1 and the column address is returned to the start column. The relationship "start column < end column "MUST be maintained.		
DATCTL	0									**$BC**	Data scan dir., etc	3 Bytes	Used to perform various setups
(P1)	1	x[1]	x[1]	x[1]	x[1]	x[1]	P12	P11	P10		Normal/inverse display of page address and page-address scan direction. P10: 0 = Normal display of the page address. 1= Inverse display P11: 0 = Normal rotation/turn of column address 1 = reverse rotation P12: 0 = Address-scan in Column direction 1 = in page direction		
(P2)	1	x[1]	x[1]	x[1]	x[1]	x[1]	P22	P21	P20		RGB arrangement: %000 = RGB,RGB,RGB… %001 = BGR,BGR,BGR…		
(P3)	1	x[1]	x[1]	x[1]	x[1]	x[1]	P32	P31	P30		Select desired display colors between the 256 colors (8 gray-scale) or 4096 colors (16 gray-scale) for the display color. %001 = 8 gray-scale display %010 = 16 gray-scale display		

VOLCTR	0									**$81**	Contrast control	2 Bytes	Used to specify the voltage regulator circuit's value α and resistance ratio
(P1)	1	x^1	x^1	P15	P14	P13	P12	P11	P10		V1 volume value 〈 **α**) Typically, **ONLY** adjust this parameter for LCD Contrast control		
(P2)	1	x^1	x^1	x^1	x^1	x^1	P22	P21	P20		1 + Rb/Ra used to specify V_2 so that V_2 <= 80 % of V_{CSL} $V_{CLS} = 3*V_{DD2}$ (ie 3* module supply voltage) V_{REG} is 1.2V constant voltage source $V_2 = (1+ Rb/Ra) * (1 – (\alpha +2)/218) * V_{REG}$		
VOLUP	0									**$D6**	Incr. Contrast by 1	None	Increments value **α** for voltage regulator circuit by 1.
VOLDWN	0									**$D7**	Dec. Contrast by 1	None	Decrements value **α** for voltage regulator circuit by 1.
RGBSET8	0									**$CE**	256-colour position set	20 Bytes	When using 256-colour mode, allows choosing colours to represent each of red, green & blue from 4096 cols.
(P1)	1	x^1	x^1	x^1	x^1	P13	P12	P11	P10		Intermediate red tone 000		
(P2)	1	x^1	x^1	x^1	x^1	P23	P22	P21	P20		Intermediate red tone 001		
(P3)	1	x^1	x^1	x^1	x^1	P33	P32	P31	P30		Intermediate red tone 010		
(P4)	1	x^1	x^1	x^1	x^1	P43	P42	P41	P40		Intermediate red tone 011		
(P5)	1	x^1	x^1	x^1	x^1	P53	P52	P51	P50		Intermediate red tone 100		
(P6)	1	x^1	x^1	x^1	x^1	P63	P62	P61	P60		Intermediate red tone 101		
(P7)	1	x^1	x^1	x^1	x^1	P73	P72	P71	P70		Intermediate red tone 110		
(P8)	1	x^1	x^1	x^1	x^1	P83	P82	P81	P80		Intermediate red tone 111		
(P9)	1	x^1	x^1	x^1	x^1	P93	P92	P91	P90		Intermediate green tone 000		
(P10)	1	x^1	x^1	x^1	x^1	P103	P102	P101	P100		Intermediate green tone 001		
(P11)	1	x^1	x^1	x^1	x^1	P113	P112	P111	P110		Intermediate green tone 010		
(P12)	1	x^1	x^1	x^1	x^1	P123	P122	P121	P120		Intermediate green tone 011		
(P13)	1	x^1	x^1	x^1	x^1	P133	P132	P131	P130		Intermediate green tone 100		
(P14)	1	x^1	x^1	x^1	x^1	P143	P142	P141	P140		Intermediate green tone 101		
(P15)	1	x^1	x^1	x^1	x^1	P153	P152	P151	P150		Intermediate green tone 110		
(P16)	1	x^1	x^1	x^1	x^1	P163	P162	P161	P160		Intermediate green tone 111		
(P17)	1	x^1	x^1	x^1	x^1	P173	P172	P171	P170		Intermediate blue tone 000		
(P18)	1	x^1	x^1	x^1	x^1	P183	P182	P181	P180		Intermediate blue tone 001		
(P19)	1	x^1	x^1	x^1	x^1	P193	P192	P191	P190		Intermediate blue tone 010		
(P20)	1	x^1	x^1	x^1	x^1	P203	P202	P201	P200		Intermediate blue tone 011		
RAMWR	0									**$5C**	Writing to memory	Data	This command turns on data entry mode.
Data	1	Data (as successive bytes) to be written to the LCD display RAM									On entering sets page & column addresses at the start address. Increments the page or column address as applicable. Mode is cancelled if any other command entered.		
RAMRD	0									**$5D**	Reading memory	Data	This command turns on data read mode.
Data	1	Data (as successive bytes) to be read from to the LCD display RAM									On entering sets page & column addresses at the start address. Increments the page or column address as applicable. Mode is cancelled if any other command entered.		
PTLIN	0									**$A8**	Partial display in	2 Bytes	This command turns on partial display mode.
(P1)	1	x^1	x^1	P15	P14	P13	P12	P11	P10		Start block address - S1D15G00 processes the LCD signals on 4-line (block) basis.		
(P2)	1	x^1	x^1	P25	P24	P23	P22	P21	P20		End block address both the display & non-display areas are both on 4-line (block) basis.		

PTLOUT	0									**$A9**	Partial display out	None	Used to exit from the partial display mode.
RMWIN	0									**$E0**	Read & modify write	None	Used when frequently modifying data to specify a specific display area (eg cursor).
RMWOUT	0									**$EE**	End	None	This command (or any other command except RMWIN) cancels the read modify write mode.
ASCSET	0									**$AA**	Area scroll set	4 Bytes	Used when scrolling only the specified portion of the screen (dividing the screen by lines).
(P1)	1	P17	P16	P15	P14	P13	P12	P11	P10		Top block address - Used to specify the TBA of the scroll + background areas. Specify the 0th block for the top screen scroll or whole screen scroll.		
(P2)	1	P27	P26	P25	P24	P23	P22	P21	P20		Bottom block address – used to specify the BBA of scroll + background areas. Specify the 41st block for the bottom or whole screen scroll. The relationship "start block < end block" must be maintained.		
(P3)	1	P37	P36	P35	P34	P33	P32	P31	P30		Number of specified blocks - {Numbers of (Top FIX area + Scroll area) blocks - 1}. When the bottom scroll or whole screen scroll, the value is identical with (P2).		
(P4)	1	x[1]	x[1]	x[1]	x[1]	x[1]	x[1]	P41	P40		Area scroll mode: %00 = centre, %01 = top, %10 = bottom, %11 = whole		
SCSTART	0									**$AB**	Scroll start set	1 Byte	Used to specify the start block address of the scroll area. Note that you MUST execute this command after executing the area scroll set command.
(P1)	1	x[1]	x[1]	P15	P14	P13	P12	P11	P10		Start block address		
OSCON	0									**$D1**	Intern. Oscillator on	None	This command turns on the internal oscillation circuit.
OSCOFF	0									**$D2**	Intern. Oscillator off	None	Turns off the internal oscillation circuit. This circuit is turned off in the reset mode.
PWRCTR	0									**$20**	Power control	1 Byte	This command is used to turn on or off the liquid crystal driving power circuit, booster/step-down circuits and the voltage follower circuit.
(P1)	1	x[1]	x[1]	x[1]	x[1]	P13	P12	P11	P10		P13 = Primary booster circuit; P12 = Secondary booster/step-down circuit; P11 = Voltage regulator + voltage follower; P10 = Reference voltage gen. circuit For each bit: 1 = On and 0 = Off		
TMPGRD	0									**$82**	Temp. gradient set	1 Byte	To specify the average temperature gradient of liquid crystal drive voltage
(P1)	1	x[1]	x[1]	x[1]	x[1]	x[1]	x[1]	P11	P10		Average temp gradient [%/℃] %00=–0.05, %01=–0. 1, %10=–0.15, %11=–0.2		
EPCTIN	0									**$CD**	Control EEPROM	1 Byte	selects S1F65170 EEPROM Control mode.
(P1)	1	x[1]	x[1]	P15	x[1]	x[1]	x[1]	x[1]	x[1]		Selects Write or Read - If P5=0: Read; if P5=1: Write		
EPCTOUT	0									**$CC**	Cancel EEPROM ctrl	None	Cancels S1F65170 EEPROM Control mode
EPMWR	0									**$FC**	Write into EEPROM	None	Writes the Electronic Control (Contrast) value and built-in resistance ratio into the EEPROM.
EPMRD	0									**$FD**	Read from EEPROM	None	This command reads the Electronic Control value and built-in resistance ratio from the EEPROM.

NOP	0									**$25**	No Operation	None	This command does not affect the operation.
EPSRRD1	0									**$7C**	Read register 1	None	Issue the EPSRRD1 and STREAD (Status Read) commands in succession to read the Electronic Control value. Also, always issue the NOP command after the STREAD (Status Read) command.
EPSRRD2	0									**$7D**	Read register 2	None	Issue the EPSRRD1 and STREAD (Status Read) commands in succession to read the built-in resistance ratio. Also, always issue the NOP command after the STREAD (Status Read) command.
STREAD	0	Status											Status read – It is the command for the IC chip test. Don't try to use this command.

4.2.2 EPSON S1D15G00 Initialisation

The following PICAXE subroutine provides the necessary commands and data to initialise the gLCD display for viewing with the 10-pin connector to the top and in 256 colour (8 bit mode).
Hopefully there are sufficient comments provided that others can use this routine.

For the very first command, the Display control command, many examples on the Internet have bit 4 of the third parameter clear (=0). My observation was that this results in a very washed out display. Correctly setting (=1) bit 4 of this parameter provided on much better and clearer display with a blacker background.

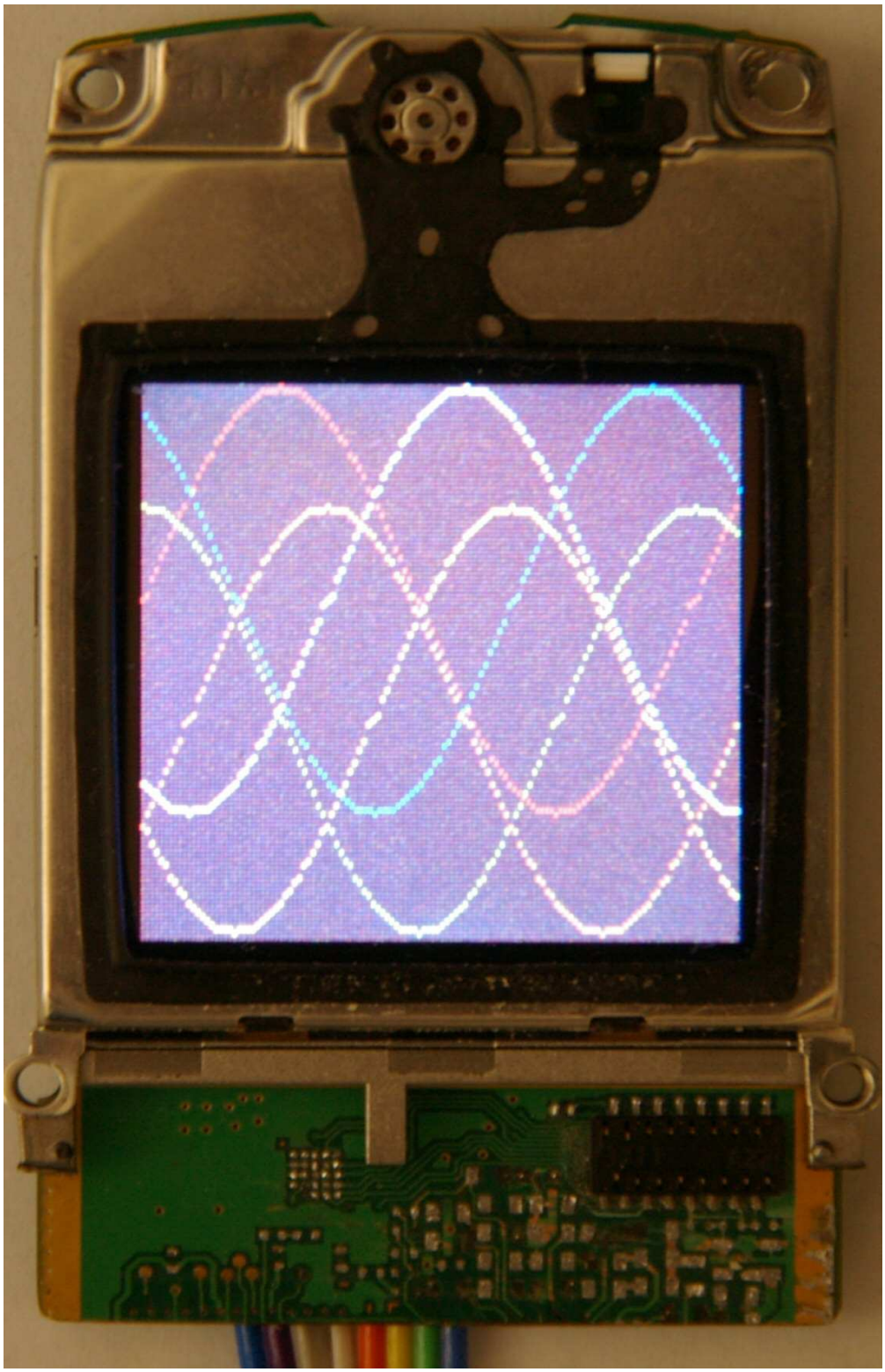
```
;---------------------------------------
InitLCD:
        ; perform signal state initialisation and gLCD reset
        HIGH CS
        PAUSE 10
        HIGH SCLK
        LOW SDA
        LOW RES
        PAUSE 1
        HIGH RES
        PAUSE 20
        LOW CS

        ; turn on the display and configure the gLCD for 8-colour mode
        value = $CA : GOSUB SendCmd     ; Display Control Command (takes 3 byte parameters)
                                        ; MUST be set before Sleepout and not adjusted when
                                        ; the display is turned on.
        value = $00 : GOSUB SendData    ; lower 4 bits used : CL diving ratio, F1 and F2 drive pattern
        value = $20 : GOSUB SendData    ; lower 6 bits used : Drive duty
        value = $12 : GOSUB SendData    ; by datasheet, bit4 MUST be 1,
                                        ;lower 4 bits used - FR inverse-set value
                                        ; can adjust by 1 or 2 from ~$10 to $14 if required


        value = $BB : GOSUB SendCmd     ; Common Scan Direction Command (takes 1 byte parameter)
        value = $01 : GOSUB SendData    ; lower 3 bits used : $00 = 1→80 & 81→160;
                                        ;                     $01 = 1→80 & 160→81
                                        ; The need for value $01 suggests Nokia reverse connections
                                        ; for COM81 pin to COM 160 pin


        value = $D1 : GOSUB SendCmd     ; Internal Oscillator On command (takes 0 byte parameters)

        value = $94 : GOSUB SendCmd     ; Sleepout command - brings the gLCD out of Sleep mode
                                        ; (takes 0 byte parameters)

        value = $81 : GOSUB SendCmd     ; Electronic Volume (Contrast) Control Command
                                        ; (takes 2 byte parameters)
        value = $10 : GOSUB SendData    ; lower 6 bits used : V1 volume/contrast value 'alpha' $00 to $3F
                                        ; is valid
        value = $03 : GOSUB SendData    ; lower 3 bits used : 1+ Rb/Ra = specifies internal resistance ratio
                                        ; may need to adjust this parameter from $01 up to #03
                                        ; depending upon settings for Display Control Command parm 3.

        value = $20 : GOSUB SendCmd     ; Power Control Command (takes 1 byte parameter)
        value = $0F : GOSUB SendData    ; lower 4 bits used: b3=prim boost circuit; b2=secondary boost cct;
                                        ; for each 1='ON'  b1=voltage regulator; b0=Ref. voltage generator

        PAUSE 200                       ; wait a while for power circuits to initialise

        value = $A7 : GOSUB SendCmd     ; Inverse Display command (takes 0 byte parameters)
```

```
        value = $BC : GOSUB SendCmd     ; Data Control Command (takes 3 byte parameters)
        value = $00 : GOSUB SendData    ; lower 3 bits used : Normal/Inverse page address & scan direction
        value = $00 : GOSUB SendData    ; lower 3 bits used : RGB arrangement
                                        ; %000=RGBRGB...RGB; $001=BGRBGR...BGR
        value = $01 : GOSUB SendData    ; lower 3 bits used : Gray-scale setup
                                        ;$01=8 gray-scale; $02 =16 gray-scale


        value = $CE : GOSUB SendCmd     ; 256-colour Position Set Command (takes 20 byte parameters)
                                        ; For each of the following 20 bytes only the lower 4 bits are used.
        value = $00 : GOSUB SendData    ; Intermediate Red   tone 000
        value = $02 : GOSUB SendData    ; Intermediate Red   tone 001
        value = $04 : GOSUB SendData    ; Intermediate Red   tone 010
        value = $06 : GOSUB SendData    ; Intermediate Red   tone 011
        value = $08 : GOSUB SendData    ; Intermediate Red   tone 100
        value = $0A : GOSUB SendData    ; Intermediate Red   tone 101
        value = $0C : GOSUB SendData    ; Intermediate Red   tone 110
        value = $0F : GOSUB SendData    ; Intermediate Red   tone 111

        value = $00 : GOSUB SendData    ; Intermediate Green tone 000
        value = $02 : GOSUB SendData    ; Intermediate Green tone 001
        value = $04 : GOSUB SendData    ; Intermediate Green tone 010
        value = $06 : GOSUB SendData    ; Intermediate Green tone 011
        value = $08 : GOSUB SendData    ; Intermediate Green tone 100
        value = $0A : GOSUB SendData    ; Intermediate Green tone 101
        value = $0C : GOSUB SendData    ; Intermediate Green tone 110
        value = $0F : GOSUB SendData    ; Intermediate Green tone 111

        value = $00 : GOSUB SendData    ; Intermediate Blue  tone  00
        value = $05 : GOSUB SendData    ; Intermediate Blue  tone  01
        value = $0A : GOSUB SendData    ; Intermediate Blue  tone  10
        value = $0F : GOSUB SendData    ; Intermediate Blue  tone  11


        value = $25 : GOSUB SendCmd     ; NOP Instruction (takes 0 byte parameters)
        value = $AF : GOSUB SendCmd     ; Display ON command (takes 0 byte parameters)

        RETURN
;---------------------------------------
```

4.**2.3 Demo with 256 colour mode**
If we now include some additional program code and routines as per the following PICAXE program snippet, we can demonstrate the initialisation and graphic operation of the Nokia 6610 gLCD using the EPSON controller chip in 256 colour mode by displaying some sine-wave curves in various colours.

A note here about the speed.
When using PICAXE 40X1 (still currently my experimental workhorse chip) at 8MHz it will take approx 18 seconds to clear the entire display. Obviously using a newer X2 PICAXE chip at say 64 Mhz will reduce that to around 2.2 seconds.

The following program will result in a display like this photo:



```
; Demonstration program for using the 128 x 128 colour gLCD from Nokia mobile/cell phones
; Such displays are available from models including the 6100, 6610, 6610i

#Terminal 9600
#PICAXE 40X1
#No_Data
#No_Table

;Define the IO pins for outputs to the gLCD
SYMBOL SCLK = 7
SYMBOL SDA  = 6
SYMBOL LED  = 5
SYMBOL CS   = 4
SYMBOL RES  = 3

; Define the user/alias names for the program variables
SYMBOL value = b2               ; value to be passed to the gLCD as command or data
SYMBOL colour= b3               ; Colour to plot
SYMBOL x                = b4    ; X-Coordinate (0 to 129)
SYMBOL y                = b5    ; Y-Coordinate (0 to 129)
SYMBOL temp             = w12   ; = b25:b24  temp word variable used for intermediate calcs in Demo
SYMBOL index = w13              ; = b27:b26 index for pointers/loops requiring values > 255

; Define constants for the gLCD controller chip D/C command
SYMBOL LCD_C = $00              ; msb = Command
SYMBOL LCD_D = $80              ; msb = Data

; Define constants for some colours in the 256 colour 8-bit RGB format (RRR-GGG-BB)
SYMBOL Black = $00              ; % 000 000 00
SYMBOL Blue             = $03           ; % 000 000 11
SYMBOL Green            = $1C           ; % 000 111 00
SYMBOL Cyan             = $1F           ; % 000 111 11 = Blue + Green
SYMBOL Brown            = $AD   ; % 101 011 01
SYMBOL Pink             = $CE           ; % 110 011 10
SYMBOL Red              = $E0           ; % 111 000 00
SYMBOL Magenta          = $E3           ; % 111 000 11 = Red  + Blue
SYMBOL Orange           = $F0           ; % 111 100 00
SYMBOL Yellow           = $FC           ; % 111 111 00 = Red  + Green
SYMBOL White = $FF              ; % 111 111 11
;---------------------------------------
Init:
        SETFREQ m8
        GOSUB InitLCD                   ; initialise the gLCD ready for use
        colour = Black                  ; set the initial "background" colour for clearing
        SERTXD ("Start clearing", cr, lf)
        GOSUB ClearLCD                  ; clear the LCD to black colour
        SERTXD ("Done clearing", cr, lf)
;---------------------------------------
; Purely a demo to now plot some waveforms in a few different colours
Main:

        SERTXD ("Plotting some demo curves", cr, lf)
        FOR x = 0 TO 129

                temp = x * 3
                y = SIN temp / 2
                IF y <64 THEN
                        y = 50 - y
                ELSE
                        y = y - 14
                ENDIF
                colour = Red
                GOSUB Plot
```

```
        temp = x * 3 + 120
        y = SIN temp / 2
        IF y <64 THEN
                y = 50 - y
        ELSE
                y = y - 14
        ENDIF
        colour = Blue
        GOSUB Plot

        temp = x * 3 + 240
        y = SIN temp / 2
        IF y <64 THEN
                y = 50 - y
        ELSE
                y = y - 14
        ENDIF
        colour = White
        GOSUB Plot


        temp = x * 3
        y = COS temp / 2
        IF y <64 THEN
                y = 78 - y
        ELSE
                y = y + 14
        ENDIF
        colour = Green
        GOSUB Plot

        temp = x * 3 + 120
        y = COS temp / 2
        IF y <64 THEN
                y = 78 - y
        ELSE
                y = y + 14
        ENDIF
        colour = Yellow
        GOSUB Plot

        temp = x * 3 + 240
        y = COS temp / 2
        IF y <64 THEN
                y = 78 - y
        ELSE
                y = y + 14
        ENDIF
        colour = Orange
        GOSUB Plot

NEXT

PAUSE 500               ; wait a while so user can see what has happened this far . . . .

; Now test the display by incrementing/looping the contrast through the entire range
; The increment and decrement volume/contrast functions only work with the 'alpha' (first) parameter
; for the full electronic volume/contrast control which only uses the lower 6 bits.
; Range is $00 to $3F and the actual value wraps around after $3F to $00.
```

```
        SERTXD ("Now changing contrast", cr, lf)
        FOR index = $00 TO $3F
                value = $D6
                GOSUB SendCmd       ; Command to increment Contrast by 1 over the full 6-bit range
                  SERTXD (#index, "  ")
                PAUSE 1000
        NEXT
        SERTXD ("All done - now going into a Do Loop", cr, lf)
        DO
        LOOP   ; loop forever
        END

;----------------------------------------
; Subroutine to rapidly clear the gLCD with a colour defined before calling
ClearLCD:
        x = 0     : width  = 130            ; set the "cursor" position to the screen origin for sequential clear
        y = 0 : height = 130                ; and the width/height for full screen
        GOSUB GotoXY                              ; move "cursor" to the desired start point
        value = $5C : GOSUB SendCmd         ; Send the command to start writing to LCD RAM
        value = colour                      ; set the colour to be used for clearing screen
        FOR index = 0 TO 3380               ; step through all 130 x 130 gLCD module visible pixel locations 5 at time
                SHIFTOUT SCLK, SDA, MSBFirst_L, (LCD_D/1, value/8, LCD_D/1, value/8, LCD_D/1, value/8,
LCD_D/1, value/8, LCD_D/1, value/8)
        NEXT
        RETURN;----------------------------------------
; Subroutine to set the colour for a pixel at a specified x,y co-ordinate
Plot:
        value = $75             : GOSUB SendCmd     ; Page Address Set command (takes 2 byte parameters)
        value = y + 2     : GOSUB SendData      ; Start page (must be lower than the 'End Page' value)
        value = 131             : GOSUB SendData    ; End Page

        value = $15             : GOSUB SendCmd     ; Column Address Set command (takes 2 byte param's)
        value = x               : GOSUB SendData    ; Start Address (must be < the 'End Address' value)
        value = 129             : GOSUB SendData    ; End Address

        value = $5C             : GOSUB SendCmd     ; Writing to memory command
        value = colour          : GOSUB SendData    ; send the required colour (256 colour value 8-bit mode)
        RETURN
;----------------------------------------
; Subroutine to send a command to the gLCD module
SendCmd:
        SHIFTOUT SCLK, SDA, MSBFirst_L, (LCD_C/1, value/8)
        RETURN
;----------------------------------------
; Subroutine to send a data value to the gLCD module
SendData:
        SHIFTOUT SCLK, SDA, MSBFirst_L, (LCD_D/1, value/8)
        RETURN
;----------------------------------------
```

After some testing with the **ClearLCD** subroutine by sending various sets of data within the one SHIFTOUT command, the optimal configuration seems to be sending 5 sets of data at one time.
Sending one data set at a time the display clear took 30 seconds.
Using four separate SHIFTOUT commands each with one set of data only reduced the time to 25 seconds.
Using four sets of data within one SHIFTOUT commands reduced to time to 20 seconds.
Using five or six sets of data in one SHIFTOUT command reduced the time to 18 seconds.
Using five sets of data in each of two SHIFTOUT command and the time was still 18 seconds.

I used this same technique in the drawing of filled rectangles. By sending lots of 5 data in one SHIFTOUT command. Even if a few more sets of data than is requires is sent, the extra data is ignored.

4.2.4 Using the controller chip Set Area and Wrap-Around feature

A feature of the controller chips is that when setting a starting position we must also define an end position.
In the case of the **ClearLCD** routine, we set the starting point to position 0,0 and the end position to 130,130 so the entire screen is selected.

For drawing a filled rectangle, the same method can be used so that when we have set all the pixels in the first row, the controller chip automatically moves to the appropriate start position in the second row and continues. This enables us to fill a rectangle quite quickly.

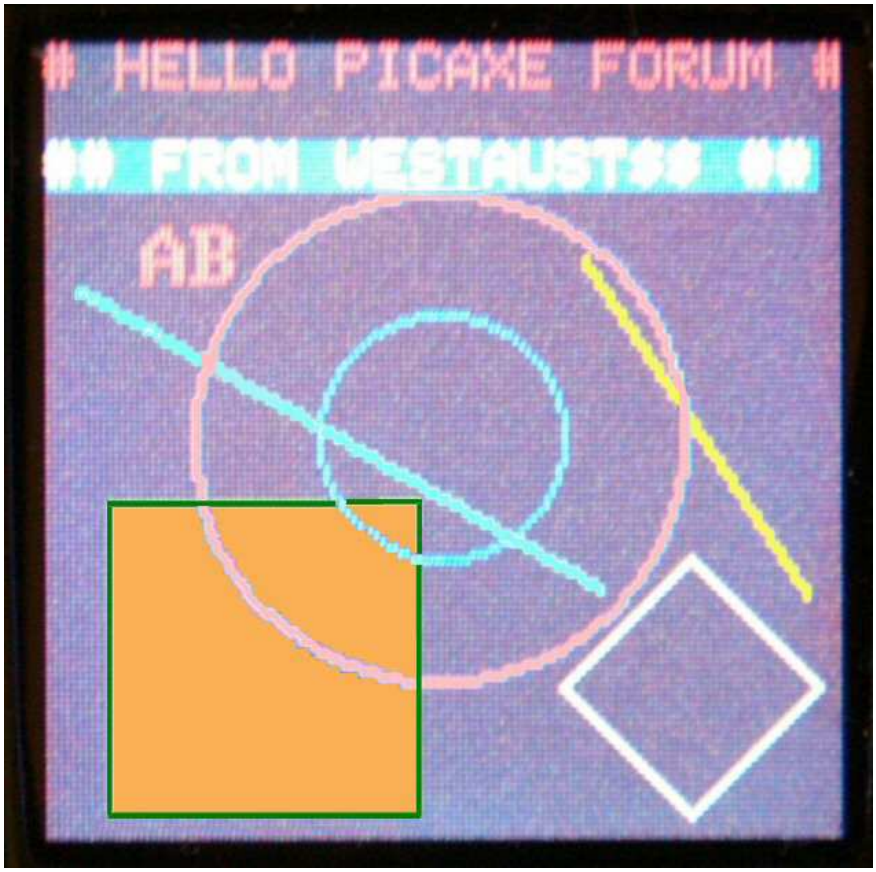
For displaying a character I use the same technique by setting the area to the size of a character and then sending out all of the data for that character in a stream. For characters we must then send the background colour for pixels which are not in the text colour. This is quite easy as the program only needs to look at the character data from EEPROM and send the text colour if the bit is a 1 and the background colour if the bit is a 0.

4.3 Extended Demonstration Program

I have provided a more extensive suite of subroutines in the form of a demonstration PICAXE driver program.
In additional to the waveforms generated in the previous program listing above, the more complete demonstration program includes:

- Initialise gLCD module
- Clear the display
- Organised shutdown of the gLCD
- display text in specified colour (routine can handle 5x7 and 7x9 pixel characters)
- plot/set a point in specified colour
- draw a line between two point in specified colour
- draw a filled or unfilled rectangle in specified colour based on two points
- draw a circle in specified colour based on centre coordinates and radius
- draw an equal sided diamond in specified colour based on centre coordinates and side length

The following photo shows the result of most of these drawing routines. The “AB” in 3rd line is in the larger font.



I have done some editorial recolouring of the green/orange boxes as it was washed out in the photos.
Due to the size of the PICAXE program listing the program is separately posted in the same thread as this pdf file.

4.4 PHILIPS PCF8833 Controller

4.4.1 Command Set

The following 4-page table provides a summary of the commands and where required the parameters associated with each command for the Philips PCF8833 controller chip. While I have endeavoured to incorporate a lot of data for the more likely parameters, the reader should still download and read the datasheet for the Philips PCF8833 controller chip which is readily found by searching the Internet.

In reality, those commands on the last page are for reading back information and factory testing. As such they would generally never be used, so can be ignored for the majority of cases.

COMMAND	D/C Bit	COMMAND / DATA BYTE								EQUIV HEX	FUNCTION	FOLLOWING PARAMETERS	COMMENTS
		D7	D6	D5	D4	D3	D2	D1	D0				
SLEEPIN	0									**$10**	Sleep in	None	Places the display in low power sleep mode.
SLEEPOUT	0									**$11**	Sleep out	None	Brings the display out of sleep mode.
INVOFF	0									**$20**	Inverse display Off	None	Used to inversely highlight the display area without modifying the display data RAM.
INVON	0									**$21**	Inverse display On	None	Used to normally highlight the display area without modifying the display data RAM.
PTLON	0									**$12**	Partial mode On	None	Use to turn on the Partial display mode
NORON	0									**$13**	Normal display mode on	None	Used to turn the display into normal mode
DISPOFF	0									**$28**	Display off	None	It is used to forcibly turn the display off.
DISPON	0									**$29**	Display on	None	It is used to turn the display on. Must cancel the sleep mode first
BSTROFF	0									**$02**	Booster voltage Off	None	Turn OFF the DC-DC converters for **V**lcd
BSTRON	0									**$03**	Booster voltage On	None	Turn ON the DC-DC converters for **V**lcd
SETMUL	0									**$C2**	Set Multiplier	1 Byte	
(P1)	1	x^1	x^1	x^1	x^1	x^1	x^1	S1	S0		Multiplication Factor of voltage multiplier %00=2x, %01=3x, %10=4x & **%11=5x (default)**		
CASET	0									**$2A**	Column addr. set	2 Bytes	Used to specify the column address area.
(P1)	1	xs7	xs6	xs5	xs4	xs3	xs2	xs1	xs0		Start address	As addresses are incremented from the start to the end column the page addr.	
(P2)	1	xe7	xe6	xe5	xe4	xe3	xe2	xe1	xe0		End address	is incremented by 1 and the column address is returned to the start column. The relationship "start column **<=** end column "MUST be maintained.	
PASET	0									**$2B**	Page address set	2 Bytes	Used to specify the page address area.
(P1)	1	ys7	ys6	ys5	ys4	ys3	ys2	ys1	ys0		Start page	As addresses are incremented from the start to the end page the column address	
(P2)	1	ye7	ye6	ye5	ye4	ye3	ye2	ye1	ye0		End page	is incremented by 1 and the page address is returned to the start page The relationship "start page **<=** end page" MUST be maintained.	
SETCON	0									**$25**	Contrast control	1 Byte	Used to specify **V**con as a part of **V**lcd
(P1)	1	x^1	V_{con6}	V_{con5}	V_{con4}	V_{con3}	V_{con2}	V_{con1}	V_{con0}		Contrast value **V**con - adjust this parameter for LCD Contrast control		
SETVOP	0									**$B0**	Set Vop	2 Bytes	Set the Vop for optimum LCD supply voltage
(P1)		x^1	x^1	x^1	x^1	V_{pr8}	V_{pr7}	V_{pr6}	V_{pr5}	**Reset Def = $08**	Vlcd = 3.6 + 0.04 * (Vpr + Vcon + MMVOPCAL) ; MMVOPCAL = 0 ?		
(P2)		x^1	x^1	x^1	V_{pr4}	V_{pr3}	V_{pr2}	V_{pr1}	V_{pr0}	**Reset Def = $01**	Rage for Vpr is 5 to 410 ($19A)		
MADCTL	0									**$36**	Memory Data access Ctrl	1 Byte	Used to set mirroring and RGB format
	1	MY	MX	V	LAO	RGB	x^1	x^1	x^1		MY,MX=1 to mirror; V=1 to write vertically; RGB=0➔ RGB, RGB=1➔BGR		
COLMOD	0									**$3A**	Interface pixel Format	1 Byte	Used to select the RGB format
(P1)	1	x^1	x^1	x^1	x^1	x^1	P12	P11	P10		$02 = 8-bit/pixel, $03 = 12-bit /pixel, $05 = 16-bit/pixel		
DALO	0									**$22**	All pixels Off	None	Switch OFF all pixels regardless of RAM data
DAL	0									**$23**	All pixels On	None	Switch ON all pixels regardless of RAM data

NOP	0									**$00**	No Operation	None	This command does not affect the operation.
SWRESET	0									**$01**	Software Reset	None	Used for same effect as hardware reset
RGBSET8	0									**$2D**	256-colour position set	20 Bytes	When using 256-colour mode, allows choosing colours to represent each of red, green & blue from 4096 cols.
(P1)	1	x[1]	x[1]	x[1]	x[1]	R13	R12	R11	R10		Intermediate red tone 000		
(P2)	1	x[1]	x[1]	x[1]	x[1]	R23	R22	R21	R20		Intermediate red tone 001		
(P3)	1	x[1]	x[1]	x[1]	x[1]	R33	R32	R31	R30		Intermediate red tone 010		
(P4)	1	x[1]	x[1]	x[1]	x[1]	R43	R42	R41	R40		Intermediate red tone 011		
(P5)	1	x[1]	x[1]	x[1]	x[1]	R53	R52	R51	R50		Intermediate red tone 100		
(P6)	1	x[1]	x[1]	x[1]	x[1]	R63	R62	R61	R60		Intermediate red tone 101		
(P7)	1	x[1]	x[1]	x[1]	x[1]	R73	R72	R71	R70		Intermediate red tone 110		
(P8)	1	x[1]	x[1]	x[1]	x[1]	R83	R82	R81	R80		Intermediate red tone 111		
(P9)	1	x[1]	x[1]	x[1]	x[1]	G13	G12	G11	G10		Intermediate green tone 000		
(P10)	1	x[1]	x[1]	x[1]	x[1]	G23	G22	G21	G20		Intermediate green tone 001		
(P11)	1	x[1]	x[1]	x[1]	x[1]	G33	G32	G31	G30		Intermediate green tone 010		
(P12)	1	x[1]	x[1]	x[1]	x[1]	G43	G42	G41	G40		Intermediate green tone 011		
(P13)	1	x[1]	x[1]	x[1]	x[1]	G53	G52	G51	G50		Intermediate green tone 100		
(P14)	1	x[1]	x[1]	x[1]	x[1]	G63	G62	G61	G60		Intermediate green tone 101		
(P15)	1	x[1]	x[1]	x[1]	x[1]	G73	G72	G71	G70		Intermediate green tone 110		
(P16)	1	x[1]	x[1]	x[1]	x[1]	G83	G82	G81	G80		Intermediate green tone 111		
(P17)	1	x[1]	x[1]	x[1]	x[1]	B13	B12	B11	B10		Intermediate blue tone 000		
(P18)	1	x[1]	x[1]	x[1]	x[1]	B23	B22	B21	B20		Intermediate blue tone 001		
(P19)	1	x[1]	x[1]	x[1]	x[1]	B33	B32	B31	B30		Intermediate blue tone 010		
(P20)	1	x[1]	x[1]	x[1]	x[1]	B43	B42	B41	B40		Intermediate blue tone 011		
RAMWR	0									**$2C**	Writing to memory	Data	This command turns on data entry mode.
Data	1	Data (as successive bytes) to be written to the LCD display RAM									On entering sets page & column addresses at the start address. Increments the page or column address as applicable. Mode is cancelled if any other command entered.		
PTLAR	0									**$30**	Partial Area Setting	2 Bytes	Used to set partial display area and display LCD RAM content of that area
(P1)	1	AAS7	AAS6	AAS5	AAS4	AAS3	AAS2	AAS1	AAS0		PTLAR active area start address/row – set in multiples of 4 **&** AAE**+**1 **–** AAS = 32		
(P2)	1	AAE7	AAE6	AAE5	AAE4	AAE3	AAS2	AAE1	AAE0		PTLAR active area end address/row – AAE **+** 1 set in multiples of 4		
VSCRDEF	0									**$33**	Vertical Scroll Def'n	3 Bytes	Used to define to vertical fixed & scrolling areas
(P1)	1	TF7	TF6	TF5	TF4	TF3	TF2	TF1	TF0		Defines the number of rows for the top fixed area. No top area when TF[7:0]=0		
(P2)	1	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0		Defines the number of rows for the scrolling area.		
(P3)	1	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0		Defines the number of rows for the bottom fixed area. No bottom area when BF[7:0]=0		
TEOFF	0									**$34**	Tearing Line OFF	None	Used to set tearing effect OFF
TEON	0									**$35**	Tearing Line ON	1 Byte	Used to set tearing effect ON
(P1)	1	x[1]	x[1]	x[1]	x[1]	x[1]	x[1]	x[1]	x[1]				

SEP	0									**$37**	Set Scroll Entry Pnt	1 Byte	Defines vertical offset down within scroll area
(P1)	1	SEP7	SEP6	SEP5	SEP4	SEP3	SEP2	SEP1	SEP0		Sets the numbers of rows offset between RAM and actual display		
IDMOFF	0									**$38**	Idle mode OFF	None	
IDMON	0									**$39**	Idle mode ON	None	Enable idle mode to reduce power consumption
BRS	0								BRS	**$B4/B5**	Bottom Row Swap	None	Used to swap/mirror the bottom rows
TRS	0								TRS	**$B6/B7**	Top Row Swap	None	Used to swap/mirror the top rows
FINV	0								FINV	**$B8/B9**	Super frame Inversion	None	Super frame Inversion is turned OFF/ON
DOR	0								DOR	**$BA/BB**	Data Order	None	Data order msb/lsb writing to RAM is normal/transposed
TCDFE	0								TCDFE	**$BC/BD**	FF Temp Compensation	None	Disable/Enable frame frequency temperature compensation
TCVOPE	0								TCVOPE	**$BE/BF**	Temp Comps'n	None	Disable/Enable Vlcd temperature compensation
EC	0								EC	**$C0/C1**	Use Int/Ext Osc	None	Use Internal/External Oscillator
TCVOPAB	0									**$C3**	Set TCVOP slopes A/B	1 Byte	Set Temp Comp slopes for regions A and B
	1	x[1]	SLB2	SLB1	SLB0	x[1]	SLA2	SLA1	SLA0		See PCF8833 datasheet for MA, MB factors and slope data		
TCVOPCD	0									**$C4**	Set TCVOP slopes C/D	1 Byte	Set Temp Comp slopes for regions C and D
	1	x[1]	SLD2	SLD1	SLD0	x[1]	SLC2	SLC1	SLC0		See PCF8833 datasheet for MC, MD factors and slope data		
TCDF	0									**$C5**	Set Divider Freq.	4 Bytes	Set temp compensation division factors
(P1)	1	x[1]	DFA6	DFA5	DFA4	DFA3	DFA2	DFA1	DFA0				
(P2)	1	x[1]	DFB6	DFB5	DFB4	DFB3	DFB2	DFB1	DFB0				
(P3)	1	x[1]	DFC6	DFC5	DFC4	DFC3	DFC2	DFC1	DFC0				
(P4)	1	x[1]	DFD6	DFD5	DFD4	DFD3	DFD2	DFD1	DFD0				
DF8colour	0									**$C6**	Set 8-col divider freq	1 Byte	Used when idle mode is selected
(P1)	1	x[1]	DF8-6	DF8-5	DF8-4	DF8-3	DF8-2	DF8-1	DF8-0		Set divider factor		
SETBIAS	0									**$C7**	Set bias system	1 Byte	Used to set the Vlcd bias system
(P1)	1	x[1]	x[1]	x[1]	x[1]	VB3	VB2	VB1	VB0		Set bias factor (a) – see PCF8833 datasheet		
NLI										**$C9**	n-line Inversion	1 Byte	
(P1)	1	NLI7	NLI6	NLI5	NLI4	NLI3	NLI2	NLI1	NLI0		After NLI slots inversion		

RDTEMP	0									**$C8**	Read Temperature	None	Read the PCF8833 temperature sensor
RID1	0									**$DA**	Read ID 1	None	
RID2	0									**$DB**	Read ID 2	None	
RID3	0									**$DC**	Read ID 3	None	
SELDEF	0								SFD	**$EF**	Select Factory Defaults		
ENTCAL	0									**$F0**	Enter Calibration	1 Byte	
(P1)	1	x[1]	x[1]	ORA2	ORA1	ORA0	x[1]	OPE	CALMM		Calibration control setting		
OTPSHTIN	0									**$F1**	Shift data in OTP shift registers	1 Byte	
(P1)	1	OS7	OS6	OS5	OS4	OS3	OS2	OS1	OS0		Multiple data byte – any number of bytes allowed		
RDDIDIF	0									**$04**	Read Display ID	3 Byte	
dummy		1 dummy clock cycle, not full byte											
(P1)		P37	P36	P35	P34	P33	P32	P31	P30	**$45**	Manufacturer – hardwired as $45		
(P2)		P27	P26	P25	P24	P23	P22	P21	P20		P27 = STN B/W=0 & STN Colour=1; P26:P20 = Driver/Module version No		
(P3)		P17	P16	P15	P14	P13	P12	P11	P10		P17:P10 = Driver/Module code		
RDDST	0									**$09**	Read Display Status		
dummy		1 dummy clock cycle, not full byte											
(P1)		P37	P36	P35	P34	P33	P32	0	0		See datasheet		
(P2)		0	P26	P25	P24	P23	P22	P21	P20		See datasheet		
(P3)		P17	0	P15	P14	P13	P12	P11	0		See datasheet		
(P4)		0	0	0	0	0	0	0	0		See datasheet		

4.4.2 PHILIPS PCF8833 Initialisation

The following PICAXE subroutine provides the necessary commands and data to initialise the gLCD display for viewing with the 10-pin connector to the top and in 256 colour (8 bit mode).
Hopefully there are sufficient comments provided that others can use this routine.

Note that there are apparently some clone controller chips in “Knock-off” displays which require the display invert and X and Y-axis mirroring to be enabled for correct display orientation. I have not found this necessary with the Philips based gLCD module removed from Nokia 6610 and 6610i mobile phones.

```
;----------------------------------------
; Routine to initialise the gLCD with PHILIPS PCF8833 driver chip
InitLCD:
; perform signal state initialisation and gLCD reset
  HIGH CS
  PAUSE 10
  HIGH SCLK
  LOW SDA
  LOW RES
  PAUSE 1
  HIGH RES
  PAUSE 20
  LOW CS

  ; turn on the display and configure the gLCD for 8-colour mode
   value = $11 : GOSUB SendCmd   ; Sleepout command - brings the gLCD out of Sleep mode
                                 ;(takes 0 byte parameters)
  value = $29 : GOSUB SendCmd    ; Display ON command (takes 0 byte parameters)
  value = $03 : GOSUB SendCmd    ; Booster Circuit On (takes 0 byte parameters)

  value = $25 : GOSUB SendCmd    ; Set Contrast Command (takes 1 byte parameters)
  value = $35 : GOSUB SendData   ; lower 7 bits used for Vcon: contrast value $00 to $7F is valid
                                 ; typical good range for viewing is $30 to $38

  value = $B0 : GOSUB SendCmd    ; Set Vop  (takes 2 byte parameters)
                                 ; Vlcd = 3.6 + (Vpr + Vcon) * 0.04 Volts; assumes MMVOPCAL=0
                                 ; The default on reset state of VPR[8:0] is 257dec (13.88 V).
  value = $08 : GOSUB SendData   ; lower 4 bits used as msb for Vpr -- default =$08
  value = $01 : GOSUB SendData   ; lower 5 bits used as lsb for Vpr -- default =$01

  value = $3A : GOSUB SendCmd    ; Pixel Interface Format (takes 1 byte parameters)
  value = $02 : GOSUB SendData   ; Colour Interface Format Setting - $02 for 8-bit, $03 for 12-bit

  value = $2D : GOSUB SendCmd    ; Colour Set - for the 256-colour mode (takes 20 bytes)
                                 ; this and the following 20 data lines are not essential
                                 ; as a default set are implemented on reset (but slightly different)
                                 ; For each of the following 20 bytes only the lower 4 bits are used
  value = $00 : GOSUB SendData   ; Intermediate Red   tone 000
  value = $02 : GOSUB SendData   ; Intermediate Red   tone 001
  value = $04 : GOSUB SendData   ; Intermediate Red   tone 010
  value = $06 : GOSUB SendData   ; Intermediate Red   tone 011
  value = $08 : GOSUB SendData   ; Intermediate Red   tone 100
  value = $0A : GOSUB SendData   ; Intermediate Red   tone 101
  value = $0C : GOSUB SendData   ; Intermediate Red   tone 110
  value = $0F : GOSUB SendData   ; Intermediate Red   tone 111

  value = $00 : GOSUB SendData   ; Intermediate Green tone 000
  value = $02 : GOSUB SendData   ; Intermediate Green tone 001
  value = $04 : GOSUB SendData   ; Intermediate Green tone 010
  value = $06 : GOSUB SendData   ; Intermediate Green tone 011
  value = $08 : GOSUB SendData   ; Intermediate Green tone 100
```

```
  value = $0A : GOSUB SendData    ; Intermediate Green tone 101
  value = $0C : GOSUB SendData    ; Intermediate Green tone 110
  value = $0F : GOSUB SendData    ; Intermediate Green tone 111

  value = $00 : GOSUB SendData    ; Intermediate Blue  tone  00
  value = $05 : GOSUB SendData    ; Intermediate Blue  tone  01
  value = $0A : GOSUB SendData    ; Intermediate Blue  tone  10
  value = $0F : GOSUB SendData    ; Intermediate Blue  tone  11

  value = $20 : GOSUB SendCmd     ; Normal Display command (takes 0 byte parameters)

  value = $36 : GOSUB SendCmd     ; Memory Data Access Control Command (takes 1 byte)
  value = $00 : GOSUB SendData    ; b7=MY, b6 =MX, b5=V, b4=LAO, b3=RGB, b2:b0=xxx
                                  ; MX=1 on, MY=1 on, v=0 RAM write x axis,
                                  ; RGB 0=RGB / 1=BGR
                                  ; Seems some clone chips need $C8 instead of $00
  RETURN
```

4.4.3 Other Subroutines Requiring Changes

To enable the same demonstration code as presented for the EPSON controller chip to function there is also a need to make minor alterations to several other subroutines. These are listed below with the altered parameters in bold.

While some changes would not be required had subroutines called others as might normally be the case, embedded gLCD commands have been introduced into some subroutines for improved speed when drawing.

```
;----------------------------------------
; Routine to shut down the PHILIPS  controller and gLCD in an organised manner
Shutdown:
  value = $28 : GOSUB SendCmd        ; Display OFF command (takes 0 byte parameters)
  value = $10 : GOSUB SendCmd        ; Enter Sleep mode command (takes 0 byte parameters)
 LOW CS                              ; Chip Select OFF
  LOW BckLED                         ; Turn the gLCD Backlight OFF
 RETURN
;----------------------------------------
; Subroutine to rapidly clear the gLCD with a colour defined before calling
ClearLCD:
  x = 0  : width  = 130    ; set the "cursor" position to the screen origin for sequential clear
  y = 0 : height = 130     ; and the width/height for full screen
  GOSUB GotoXY             ; move "cursor" to the desired start point
  value = $2C : GOSUB SendCmd  ; Send the command to start writing to LCD RAM
  value = colour                    ; set the colour to be used for clearing screen
  FOR index = 0 TO 3380 ; 16900  ; step through all 130 x 130 gLCD module visible pixel locations
        SHIFTOUT SCLK, SDA, MSBFirst_L, (LCD_D/1, value/8, LCD_D/1, value/8, LCD_D/1,
        value/8, LCD_D/1, value/8, LCD_D/1, value/8)
  NEXT
  RETURN
;----------------------------------------
; Subroutine to set the start x,y and end x,y for the defined/desired screen area
GotoXY:
  value = $2A    : GOSUB SendCmd      ; Column Address Set command (takes 2 byte parameters)
  value = x + 1  : GOSUB SendData     ; Start Address (must be lower than the 'End Address' value)
  value = x+width : GOSUB SendData    ; End Address

  value = $2B    : GOSUB SendCmd      ; Page Address Set command (takes 2 byte parameters)
  value = y + 1  : GOSUB SendData     ; Start page (must be lower than the 'End Page' value)
  value = y+height : GOSUB SendData   ; End Page  (note value is in effect y + height + 2 - 1)
  RETURN
```

```
;---------------------------------------
; Subroutine to set the colour for a pixel at an already established x,y co-ordinate
Plot:
   value = $2C  : GOSUB SendCmd      ; Send command for Writing to LCD RAM memory command
  value = colour: GOSUB SendData     ; send the required colour (256 colour 8-bit mode as RRR-
GGG-BB)
  RETURN
;---------------------------------------
; Subroutine to plot one character to the gLCD at location x,y - font size is defined by calling
'FontAtXY' before calling this routine
SendChar:
  char = char - $20
  EEAddr = char * height + EEbase      ; pointer to EEPROM location for first byte of font character
using 1 bytes per char row
  value = $2C : GOSUB SendCmd          ; Send command ready for Writing to LCD RAM memory
command

  FOR indexhi = 1 to height
        HI2CIN EEAddr, (char)
        FOR indexlo = 1 TO width     ; we only use 6 bits from each byte as 5 wide for char & 1 blank
              IF char BIT 7 SET THEN    ; if the current msb is 1 then this is part of the character
                value = colour              ; so we use the text colour
              ELSE
                value = bcolour             ; otherwise we select the nominated background colour
              ENDIF
              GOSUB SendData ; write colour data for the current pixel in character area to gLCD
              char = char << 1          ; shift byte left ready to check next pixel
        NEXT indexlo                    ; loop until all pixels in current row done for this character
        INC EEAddr                      ; increment EEPROM address pointer for data for next row of
current character
  NEXT indexhi                          ; loop until all rows are plotted for the current character
  x = x + width                         ; advance the x-coord pointer ready for the next character
  GOSUB GotoXY                          ; advance the working area right one character area ready for
the next character
        RETURN
;---------------------------------------
```

4.5 LEADIS LDS176 Controller

4.5.1 Command Set

The Leadis chips uses one IO pin (ISS) to set the instruction set.
When ISS = 0 most commands have the same value and effect as the same Philips PCF8833 controller functions, however for the contrast and $\mathbf{V}_{LCD}$ functions this chip uses a setting scheme/formula similar to the EPSON controller chips with different command values.
Additionally, for the $\mathbf{V}_{LCD}$ value the parameter enters for the alpha component (α) differs from the EPSON controller in that $\alpha = 255 - EV$
where “EV” is the contrast or (in EPSON/Leadis terminology) the Electronic Volume parameter value.

When ISS=1, most commands have the same value and effect as the same EPSON S1D15G00 controller functions.

The following 4-page table provides a summary of the commands and where required the parameters associated with each command for the Leadis LDS176 controller chip with pin ISS=0 (ie pulled low) which has been found to be the case with all (7) displays taken from Nokia 6610i mobile phones available to the author at the time of writing.

While I have endeavoured to incorporate a lot of data for the more likely parameters, the reader should still download and read the datasheet for the Leadis LDS176 controller chip which is readily found by searching the Internet.
In reality, those commands on the last page are for reading back information and factory testing. As such would generally never be used, so can be ignored for the majority of cases.

COMMAND	D/C Bit	COMMAND / DATA BYTE								EQUIV HEX	FUNCTION	FOLLOWING PARAMETERS	COMMENTS
		D7	D6	D5	D4	D3	D2	D1	D0				
SLPIN	0									**$10**	Sleep in & Booster Off	None	Places the display in low power sleep mode.
SLPOUT	0									**$11**	Sleep out & Booster On	None	Brings the display out of sleep mode.
INVOFF	0									**$20**	Inverse display Off	None	Used to normally highlight the display area without modifying the display data RAM.
INVON	0									**$21**	Inverse display On	None	Used to inversely highlight the display area without modifying the display data RAM.
PTLON	0									**$12**	Partial mode On	None	Use to turn on the Partial display mode
NORON	0									**$13**	Normal display mode On	None	Used to turn the display into normal mode with partial and scroll modes off
DISPOFF	0									**$28**	Display Off	None	It is used to forcibly turn the display off.
DISPON	0									**$29**	Display On	None	It is used to turn the display on. Must cancel the sleep mode first
BSTROFF	0									**$02**	Booster voltage Off	None	Only for test purposes – see sleep in
BSTRON	0									**$03**	Booster voltage On	None	Only for test purposes – see sleep out
CASET	0									**$2A**	Column addr. set	2 Bytes	Used to specify the column address area.
(P1)	1	xs7	xs6	xs5	xs4	xs3	xs2	xs1	xs0		Start address	As addresses are incremented from the start to the end column the page addr.	
(P2)	1	xe7	xe6	xe5	xe4	xe3	xe2	xe1	xe0		End address	is incremented by 1 and the column address is returned to the start column. The relationship "start column **<=** end column "MUST be maintained.	
RASET	0									**$2B**	Row address set	2 Bytes	Used to specify the row address area.
(P1)	1	ys7	ys6	ys5	ys4	ys3	ys2	ys1	ys0		Start page	As addresses are incremented from the start to the end column the row address	
(P2)	1	ye7	ye6	ye5	ye4	ye3	ye2	ye1	ye0		End page	is incremented by 1 and the column address is returned to the start column The relationship "start row **<=** end row" MUST be maintained.	
WRCNTR	0									**$25**	Write Contrast control	1 Byte	Used to specify α as a part of **V**lcd Default=$3F
(P1)	1	x^1	EV6	EV5	EV4	EV3	EV2	EV1	EV0		Contrast value - adjust this parameter for LCD Contrast control α where α = 255-EV		
CLKINT	0									**$B0**	Internal Oscillator	None	Select and use the internal oscillator
CLKEXT	0									**$B1**	External Oscillator	None	Select and use an external oscillator
MADCTL	0									**$36**	Memory Data access Ctrl	1 Byte	Used to set mirroring and RGB format
	1	MY	MX	MV	ML	RGB	x^1	x^1	x^1		MY,MX=1 to mirror; MV=1 to write vertically; RGB=0➔ RGB, RGB=1➔BGR ML=0 Line address top down, ML=1 Line address bottom up		
COLMOD	0									**$3A**	Interface Pixel Format	1 Byte	Used to select the RGB format
(P1)	1	x^1	x^1	x^1	x^1	x^1	P12	P11	P10		$02 = 8-bit/pixel, $03 = 12-bit /pixel typeA, $05 = 16-bit/pixel, $06 = 12-bit /pixel typeB		
APOFF	0									**$22**	All pixels Off	None	Switch OFF all pixels regardless of RAM data
APON	0									**$23**	All pixels On	None	Switch ON all pixels regardless of RAM data
NOP	0									**$00**	No Operation	None	This command does not affect the display operation.
SWRESE	0									**$01**	Software Reset	None	Used for the same effect as hardware reset

T													
RGBSET	0									**$2D**	256-colour position set	20 Bytes	When using 256-colour mode, allows choosing colours to represent each of red, green & blue from 4096 cols.
(P1)	1	x^1	x^1	x^1	x^1	R13	R12	R11	R10		Intermediate red tone 000		
(P2)	1	x^1	x^1	x^1	x^1	R23	R22	R21	R20		Intermediate red tone 001		
(P3)	1	x^1	x^1	x^1	x^1	R33	R32	R31	R30		Intermediate red tone 010		
(P4)	1	x^1	x^1	x^1	x^1	R43	R42	R41	R40		Intermediate red tone 011		
(P5)	1	x^1	x^1	x^1	x^1	R53	R52	R51	R50		Intermediate red tone 100		
(P6)	1	x^1	x^1	x^1	x^1	R63	R62	R61	R60		Intermediate red tone 101		
(P7)	1	x^1	x^1	x^1	x^1	R73	R72	R71	R70		Intermediate red tone 110		
(P8)	1	x^1	x^1	x^1	x^1	R83	R82	R81	R80		Intermediate red tone 111		
(P9)	1	x^1	x^1	x^1	x^1	G13	G12	G11	G10		Intermediate green tone 000		
(P10)	1	x^1	x^1	x^1	x^1	G23	G22	G21	G20		Intermediate green tone 001		
(P11)	1	x^1	x^1	x^1	x^1	G33	G32	G31	G30		Intermediate green tone 010		
(P12)	1	x^1	x^1	x^1	x^1	G43	G42	G41	G40		Intermediate green tone 011		
(P13)	1	x^1	x^1	x^1	x^1	G53	G52	G51	G50		Intermediate green tone 100		
(P14)	1	x^1	x^1	x^1	x^1	G63	G62	G61	G60		Intermediate green tone 101		
(P15)	1	x^1	x^1	x^1	x^1	G73	G72	G71	G70		Intermediate green tone 110		
(P16)	1	x^1	x^1	x^1	x^1	G83	G82	G81	G80		Intermediate green tone 111		
(P17)	1	x^1	x^1	x^1	x^1	B13	B12	B11	B10		Intermediate blue tone 000		
(P18)	1	x^1	x^1	x^1	x^1	B23	B22	B21	B20		Intermediate blue tone 001		
(P19)	1	x^1	x^1	x^1	x^1	B33	B32	B31	B30		Intermediate blue tone 010		
(P20)	1	x^1	x^1	x^1	x^1	B43	B42	B41	B40		Intermediate blue tone 011		
RAMWR	0									**$2C**	Writing to memory	Data	This command turns on data entry mode.
Data	1	Data (as successive bytes) to be written to the LCD display RAM									On entering sets page & column addresses at the start address. Increments the row or column address as applicable. Mode is cancelled if any other command entered.		
PTLAR	0									**$30**	Partial Area Setting	2 Bytes	Used to set partial display area and display LCD RAM content of that area
(P1)	1	PSL7	PSL6	PSL5	PSL4	PSL3	PSL2	PSL1	PSL0		PTLAR active area start address/row – 0 to 131		
(P2)	1	PEL7	PEL6	PEL5	PEL4	PEL3	PEL2	PEL1	PEL0		PTLAR active area end address/ row – 0 to 131		
TEOFF	0									**$34**	Tearing Line OFF	None	Used to set tearing effect OFF
TEON	0									**$35**	Tearing Line ON	1 Byte	Used to set tearing effect ON
(P1)	1	x^1	x^1	x^1	x^1	x^1	x^1	x^1	M		M=0➔ mode1 – based on V-Synch info ; M=1 ➔ mode2 – based on H-Synch info		
IDMOFF	0									**$38**	Idle mode OFF	None	Disable idle mode to reduce power consumption Colour mode up to 4096 colours enabled
IDMON	0									**$39**	Idle mode ON	None	Enable idle mode to reduce power consumption Colour is reduced from 4096 to 256 mode

VSCRDEF	0									**$33**	Vertical Scroll Def'n **See comment at right**	3 Bytes	Used to define to vertical fixed & scrolling areas **Datasheet suggests this is removed from LDS176**
(P1)	1	TF7	TF6	TF5	TF4	TF3	TF2	TF1	TF0		Defines the number of rows for the top fixed area. No top area when TF[7:0]=0		
(P2)	1	VS7	VS6	VS5	VS4	VS3	VS2	VS1	VS0		Defines the number of rows for the scrolling area.		
(P3)	1	BF7	BF6	BF5	BF4	BF3	BF2	BF1	BF0		Defines the number of rows for the bottom fixed area. No bottom area when BF[7:0]=0		
VSCSAD	0									**$37**	Set Scroll Entry Pnt	1 Byte	Defines scroll area start address
(P1)	1	SSA7	SSA6	SSA5	SSA4	SSA3	SSA2	SSA1	SSA0		Sets the scroll start address	See ➔	**Datasheet suggests this is removed from LDS176**
FRMSEL	0									**$B4**	Frame Freq Select	4 Bytes	Set Frame Frequency in normal display mode
(P1)	1	x[1]	x[1]	x[1]	FA4	FA3	FA2	FA1	FA0		Frame frequency in Temp range A		
(P2)	1	x[1]	x[1]	x[1]	FB4	FB3	FB2	FB1	FB0		Frame frequency in Temp range B		
(P3)	1	x[1]	x[1]	x[1]	FC4	FC3	FC2	FC1	FC0		Frame frequency in Temp range C		
(P4)	1	x[1]	x[1]	x[1]	FD4	FD3	FD2	FD1	FD0		Frame frequency in Temp range D		
FRM8SEL	0									**$B5**	Frame Freq Select	4 Bytes	Set Frame Frequency in 8 colour mode
(P1)	1	x[1]	x[1]	x[1]	F8A4	F8A3	F8A2	F8A1	F8A0		Frame frequency in Temp range A		
(P2)	1	x[1]	x[1]	x[1]	F8B4	F8B3	F8B2	F8B1	F8B0		Frame frequency in Temp range B		
(P3)	1	x[1]	x[1]	x[1]	F8C4	F8C3	F8C2	F8C1	F8C0		Frame frequency in Temp range C		
(P4)	1	x[1]	x[1]	x[1]	F8D4	F8D3	F8D2	F8D1	F8D0		Frame frequency in Temp range D		
TMPRNG	0									**$B6**	Temp Range Set	4 Bytes	Used to set the temp ranges
(P1)	1	x[1]	TA6	TA5	TA4	TA3	TA2	TA1	TA0		Temperature Range A		
(P2)	1	x[1]	TB6	TB5	TB4	TB3	TB2	TB1	TB0		Temperature Range B		
(P3)	1	x[1]	TC6	TC5	TC4	TC3	TC2	TC1	TC0		Temperature Range C		
TMPHIS	0									**$B7**	Temp Hysteresis Range		Set the temperature hysteresis range
	1	x[1]	x[1]	x[1]	x[1]	TH3	TH2	TH1	TH0		Hysteresis Value setting		

TMPREAD	0									**$B8**	Temp Read Back	2 Bytes	Read the temperature from the LCD
	1	BF	T6	T5	T4	T3	T2	T1	T0		First a Dummy byte read followed by temperature parameter		
DISCTR	0									**$BA**	Display Control	2 Bytes	Display timing setup
(P1)	1	x^1	x^1	x^1	x^1	x^1	FS2	FS1	FS0		F1 / F2 pattern		
(P2)	1	x^1	x^1	x^1	FINV	NL3	NL2	NL1	NL0		FR inversion-set value		
EPVOL	0									**$BB**	Electronic Volume Offset	3 Bytes	Set the EV offset value to be stored in EEPROM
	1	x^1	x^1	EO5	EO4	EO3	EO2	EO1	EO0		EV offset If EO5=0 : EV_IN = EV + EO[4:0] elseif EO5=1 : EV_IN = EV – EO[4:0]		
	1	x^1	x^1	x^1	x^1	x^1	x^1	x^1	x^1		Dummy byte		
	1	x^1	x^1	x^1	x^1	x^1	RO2	RO1	RO0		RR offset If RO2=0 : RR_IN = RR + RO[1:0] elseif RR2=1 : RR_IN = RR – RO[4:0]		
EPWRIN	0									**$D1**	EEPROM Write Start	None	EEPROM write mode start
EPWROUT	0									**$D0**	EEPROM Write End	None	EEPROM write mode end
RDEV	0									**$D4**	Read Internal Contrast	2 Bytes	Read internal contrast (EV_IN) value
	1	x^1	EV6	EV5	EV4	EV3	EV2	EV1	EV0		First a Dummy byte read followed by Electronic Volume (contrast) parameter		
RDRR	0									**$D5**	Read Internal Res.	2 Bytes	Read the internal resistor ratio (RR_IN)
	1	x^1	x^1	x^1	x^1	x^1	RR2	RR1	RR0		First a Dummy byte read followed by EV (contrast) resistor ratio parameter		
RDDID	0									**$04**	Read Display ID	None	Dummy byte read, then read 3 ID bytes See commands $DA, $DB and $DC below
RDDST	0									**$09**	Read Display Status	None	Dummy byte read, then read 4 status bytes
RAMRD	0									**$2E**	Memory Read	None	Dummy byte read, then read data bytes
RID1	0									**$DA**	Read ID 1	None	Dummy byte read, then read ID byte
RID2	0									**$DB**	Read ID 2	None	Dummy byte read,, then read ID byte
RID3	0									**$DC**	Read ID 3	None	Dummy byte read, then read ID byte
TEST1	0									**$E-**	Test command 1	None	**DO NOT USED – Factory purposes only**
TEST2	0									**$F-**	Test Command 2	None	**DO NOT USED – Factory purposes only**

X^1 = don't care

4.5.2 LEADIS LDS176 Initialisation

The following PICAXE subroutine provides the necessary commands and data to initialise the gLCD display for viewing with the 10-pin connector to the top and in 256 colour (8 bit mode) when hardware pin ISS=0 for predominant compatability with the Phipips PCF8833 controller.
Hopefully there are sufficient comments provided that others can use this routine.

The initialisation is very similar to the Philips PCF8833 routine except that:

(a) there is no need for a separate Booster On command (this is now part of Sleep out)
(b) The setup for contrast differs

```
;----------------------------------------
; Routine to initialise the gLCD with LEADIS LDS176 driver chip when pin ISS=0
InitLCD:
; perform signal state initialisation and gLCD reset
    HIGH CS
    PAUSE 10
    HIGH SCLK
    LOW SDA
    LOW RES
    PAUSE 1
    HIGH RES
    PAUSE 20
    LOW CS
    value = $01 : GOSUB SendCmd       ; Software reset (takes 0 byte parameters)
    value = $B0 : GOSUB SendCmd       ; Select Internal Oscillator

    ; turn on the display and configure the gLCD for 8-colour mode
    value = $11 : GOSUB SendCmd       ; Sleepout command - brings the gLCD out of Sleep mode
                                      ; and Booster ON (takes 0 byte parameters)
    value = $29 : GOSUB SendCmd       ; Display ON command (takes 0 byte parameters)

   value = $25 : GOSUB SendCmd        ; Set Contrast Command (takes 1 byte parameters)
   value = $3F : GOSUB SendData       ; lower 7 bits used for 'EV'contrast value $00 to $7F is valid
                                      ; default is $3F, typical good range to try is $38 to $48
    PAUSE 50

    value = $3A : GOSUB SendCmd       ; Pixel Interface Format (takes 1 byte parameters)
    value = $02 : GOSUB SendData      ; Colour Interface Format Setting - $02 for 8-bit, $03 for 12-bit

    value = $2D : GOSUB SendCmd       ; Colour Set - for the 256-colour mode (takes 20 byte params)
                                      ; this and the following 20 data lines are not essential as
                                      ; a default set are implimented on reset
                                      ; For each of the following 20 bytes only the lower 4 bits used.
    value = $00 : GOSUB SendData      ; Intermediate Red   tone 000
    value = $02 : GOSUB SendData      ; Intermediate Red   tone 001
    value = $04 : GOSUB SendData      ; Intermediate Red   tone 010
    value = $06 : GOSUB SendData      ; Intermediate Red   tone 011
    value = $08 : GOSUB SendData      ; Intermediate Red   tone 100
    value = $0A : GOSUB SendData      ; Intermediate Red   tone 101
    value = $0C : GOSUB SendData      ; Intermediate Red   tone 110
    value = $0F : GOSUB SendData      ; Intermediate Red   tone 111

    value = $00 : GOSUB SendData      ; Intermediate Green tone 000
    value = $02 : GOSUB SendData      ; Intermediate Green tone 001
    value = $04 : GOSUB SendData      ; Intermediate Green tone 010
    value = $06 : GOSUB SendData      ; Intermediate Green tone 011
    value = $08 : GOSUB SendData      ; Intermediate Green tone 100
```

```
    value = $0A : GOSUB SendData      ; Intermediate Green tone 101
    value = $0C : GOSUB SendData      ; Intermediate Green tone 110
    value = $0F : GOSUB SendData      ; Intermediate Green tone 111

    value = $00 : GOSUB SendData      ; Intermediate Blue  tone  00
    value = $05 : GOSUB SendData      ; Intermediate Blue  tone  01
    value = $0A : GOSUB SendData      ; Intermediate Blue  tone  10
    value = $0F : GOSUB SendData      ; Intermediate Blue  tone  11

    value = $20 : GOSUB SendCmd       ; Normal Display command (takes 0 byte parameters)

    value = $36 : GOSUB SendCmd       ; Memory Data Access Control Command (takes 1 byte)

    value = $00 : GOSUB SendData      ; b7=MY, b6 =MX, b5=MV, b4=ML, b3=RGB, b2:b0=xxx
                                      ; MX=1 on, MY=1 on, MV=0 RAM write x axis,
                                      ; ML= 0 rows top down;  RGB 0=RGB / 1=BGR
  RETURN
;----------------------------------------
```

4.5.3 Other Subroutines Requiring Changes

Due to the high degree of compatibility with the Philips PCF8833 controller, there is no need to alter other subroutines relative to those already prepared for the Philips PCF8833 controller.
See section 4.4.3 for changes relative to the EPSON controller chip.