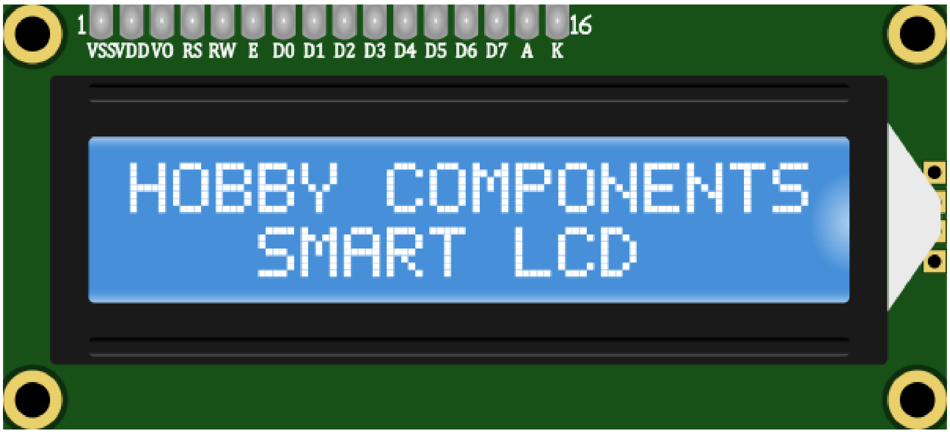
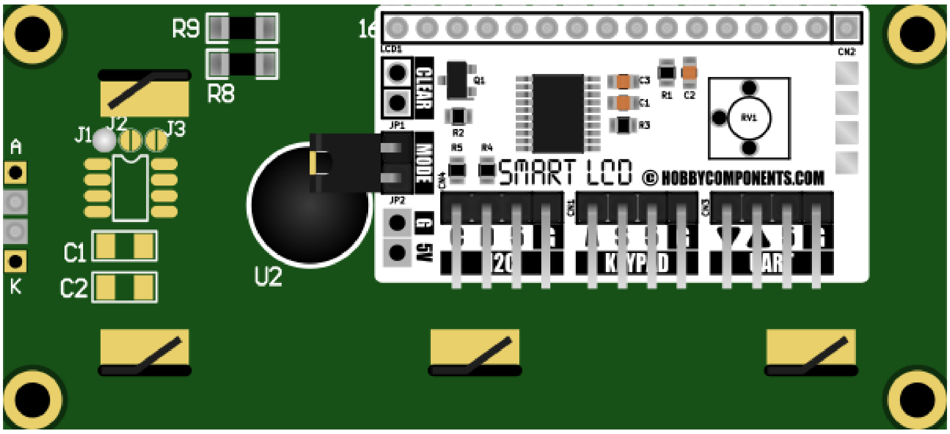
SmartLCD Backpack Module for 1602 & 2004 LCD Displays





HCMODU0162

Revision 1.1.0

DISCLAIMER

This document is provided "as is". Hobby Components Ltd makes no warranties, whether express, implied or statutory, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose, accuracy or lack of negligence. Hobby Components Ltd shall not, in any circumstances, be liable for any damages, including, but not limited to, special, incidental or consequential damages for any reason whatsoever.

The Arduino name is a protected trademark of Arduino and its partners. Our products are not directly connected to Arduino and are derivatives of the Arduino open source (Creative Commons CC-SA-BY License). Any help or technical support should be directed to Hobby Components Ltd.
(www.hobbycomponents.com).

COPYRIGHT NOTICE

This manual, including content and artwork is copyright of Hobby Components Ltd and may not be reproduced without written permission. If you paid for or received a copy of this manual from a source other than Hobby Components Ltd, please contact us at sales@hobbycomponents.com

Table of Contents

Description

The SmartLCD backpack module is designed to be interfaced directly to the parallel interface of commonly available 16x02 and 20x04 LCD alphanumeric displays using the Hitachi HD44780 controller (see hobby components items HCMODU0013, HCMODU0038, & HCMODU0039 for compatible displays). Once soldered to the rear of one of these displays the SmartLCD backpack module turns these displays into intelligent serial devices which can then be controlled via either a serial UART or I2C interface. It also requires no software or libraries to initialise the display, output text to the screen or control the features of the module, which reduces development time and frees up much needed resources in your microcontroller or development board.

For customers wishing to purchase the SmartLCD backpack already fitted to a suitable display these are available from Hobby Components in both 16x02 and 20x04 configurations:

Hobby Components 1602 SmartLCD (SKU: HCMODU0122)
Hobby Components 2004 SmartLCD (SKU: HCMODU0163)

The SmartLCD module adds three types of interface to the parallel LCD:

Terminal Mode:

This is the out-of-the-box state (mode jumper fitted) and with no configuration required will turn a 16x2 or 20x4 LCD into a serial UART terminal display. Just connect any device with a serial UART interface (9600 default BAUD rate) to the SmartLCDs UART header and the SmartLCD will automatically output any ASCII printable text it receives to the LCD. As well as printable text the SmartLCD also supports several ASCII control codes such as line feed, carriage return, backspace, delete etc.

AT Command Mode:

In AT command mode (mode jumper removed) the SmartLCD module can be controlled using simple text based ‘AT’ commands via its serial UART header. AT command mode adds a significantly greater level of control of the LCD, including precise positioning of text, backlight and contrast controls, custom characters, etc. As they are simple serial text commands they can be sent to the module using a microcontroller's serial print function such as an Arduinos built-in Serial.println() function.

I2C Mode:

Serial I2C mode gives the same level of control as AT command mode but also takes advantage of the I2C two-wire serial interface protocol. In this mode the module is addressable, allowing for multiple modules to be daisy-chained from a microcontrollers I2C interface.

For Arduino users wanting to take advantage of this mode we have written a library (SmartLCD) and example sketches to make controlling the SmartLCD module from your own sketches as simple as possible. The library will give full control of the SmartLCD using a set of straightforward library commands. The SmartLCD library can be downloaded from the software section of our support forum here:

https://forum.hobbycomponents.com/viewtopic.php?f=58&t=2206

Features

Turns a 16x2 or 20x4 alphanumeric character display into a smart serial device.
In terminal mode no software or drivers are required to initialise or control the module.
Software backlight control with 11 brightness levels.
Software contrast control.
External keypad with buzzer support (optional extra, see item HCKITS0058).
Single 5V operation.

Terminal mode

Compatible with 5V serial UART devices (9600 baud default) – no microcontroller required.
Automatically displays any printable ASCII text.
Responds to a number of additional extended ASCII control codes.

Serial AT command mode

Software control of all module features via simple AT text commands.
Software configurable baud with 8 different speeds (9600 BAUD default).
Access to external keypad and buzzer.

I2C Mode

Control multiple displays from one I2C bus.
Software control of all module features via simple I2C commands.
I2C address configurable via software from 0x08 to 0x77 (default 0x27).
Built-in 10K pull-up resistors for clock SCL and data SDA lines.
Access to external keypad and buzzer.

Specification

Model number:	HCMODU0162
Supply voltage:	5V
Display type:	Supports 16 x 2 or 20 x 4 alphanumeric HD44780 LCDs
Supply current:	9.8mA
Interfaces:	I2C, serial UART and external keypad
Module dimensions:	42mm x 23.5mm

General Operating Conditions

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Supply voltage	Vdd-Vss	---	4.5	5	5.5	V
Supply current	Idd	Backlight 100%	26	27	28	mA
	Idd	Backlight 0%	6	6.5	7	mA
Digital output pins low level voltage	IOOVl	Vdd = 5V IOIdd = 1mA	---	Vdd – 0.2	---	V
		Vdd = 5V IOIdd = 10mA	---	Vdd – 1.1	---	V
Digital output pins sink/source current	IOOIdd	Vdd = 5V	---	---	10	mA
Digital input pins low level voltage	IOIVl	Vdd = 5V	-0.3V	---	0.3 x Vdd	V
Digital input pins high level voltage	IOIVh	Vdd = 5V	0.7 x Vdd	---	Vdd + 0.3	V
SCL clock low time	tw(SCLL)	---	4.7	---	---	us
SCL clock high time	tw(SCLH)	---	4.0	---	---	us

SDA and SCL rise time	tr(SDA)
	tr(SCL)
		---	---	---	1000	ns

SDA and SCL fall time	tf(SDA)
	tf(SCL)
		---	---	---	300	ns

Capacitive load for each I2C bus line	Cb	Vdd = 5V	---	---	400	pF
Keypad (analogue in) conversion voltage range	Vain	---	Vss	---	Vdd	V
Keypad (analogue in) internal sampling capacitor	CADC	---	---	3	---	pF
Keypad (analogue in) conversion error	EtADC	Vdd = 5V RAIN < 10KΩ	---	---	1	LSB
Operating temperature	Topr	---	0	---	50	°C

Attaching the SmartLCD to a display

The SmartLCD backpack module can be attached to a suitable LCD display by soldering the module to the display's parallel interface header using the pin header strip supplied with the module. Note that the SmartLCD module is only compatible with 16 character by 2 line and 20 character by 4 line alphanumeric LCD displays using a Hitichi HD44780 (or HD44780 compatible) controller. For compatible displays please see Hobby Components items:

16x02 character transmissive parallel LCD with blue backlight (SKU: HCMODU0013)

https://hobbycomponents.com/displays/184-1602-parallel-lcd-module-blue-backlight

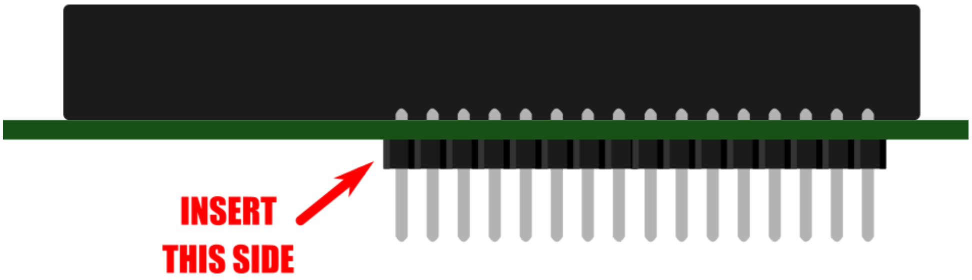
16x02 character transflective parallel LCD with yellow backlight (SKU: HCMODU0038):

https://hobbycomponents.com/displays/328-1602-parallel-lcd-module-yellow-backlight

20x04 character transmissive parallel LCD with blue backlight (SKU: HCMODU0039):

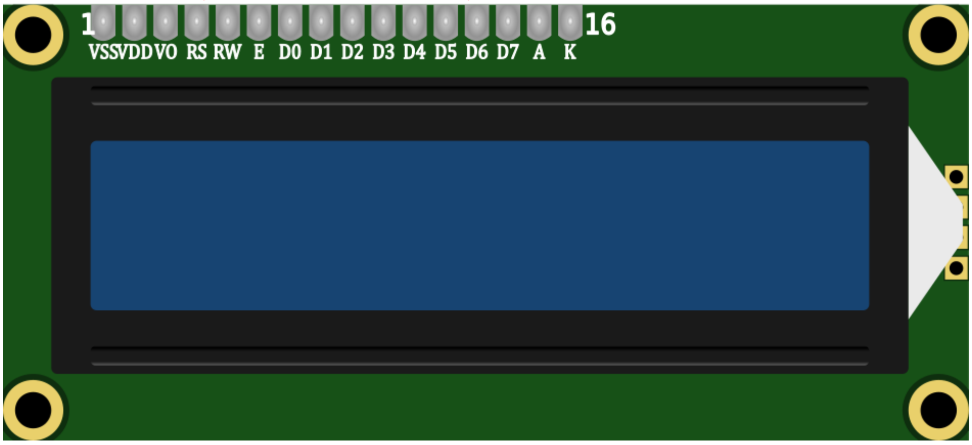
https://hobbycomponents.com/displays/478-2004-parallel-lcd-module

To attached the SmartLCD module to the LCD display, first insert the supplied header strip in to the reverse side of the LCDs parallel interface by inserting the header strip through the LCD displays pads from the reverse side of the LCD as shown:



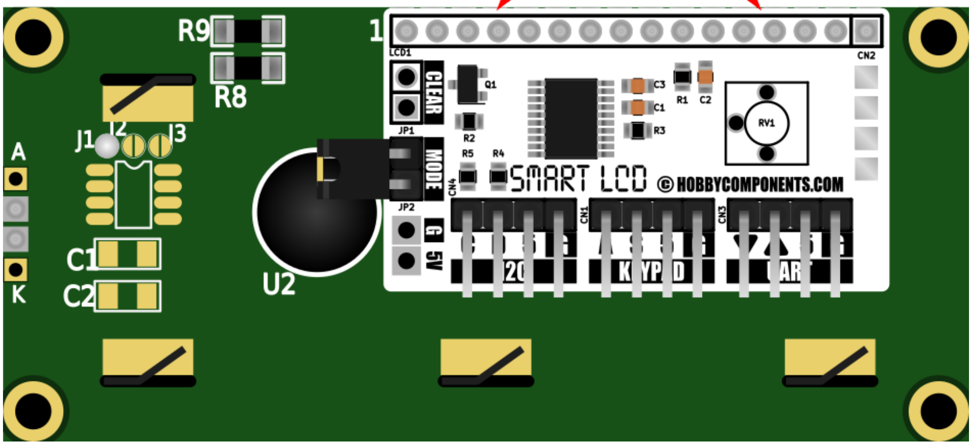
Then solder the header strip to the LCDs pads on the display side of the LCD making sure the header strip remains flat against the displays PCB on the rear side:

SOLDER PADS THIS SIDE



Next, flip the display back over to the rear side and insert the SmartLCD modules parallel interface header onto the header pins with the component side of the SmartLCD module facing away from the back of the LCD and towards you making sure that the SmartLCDs parallel interface is correctly aligned with the header pins:

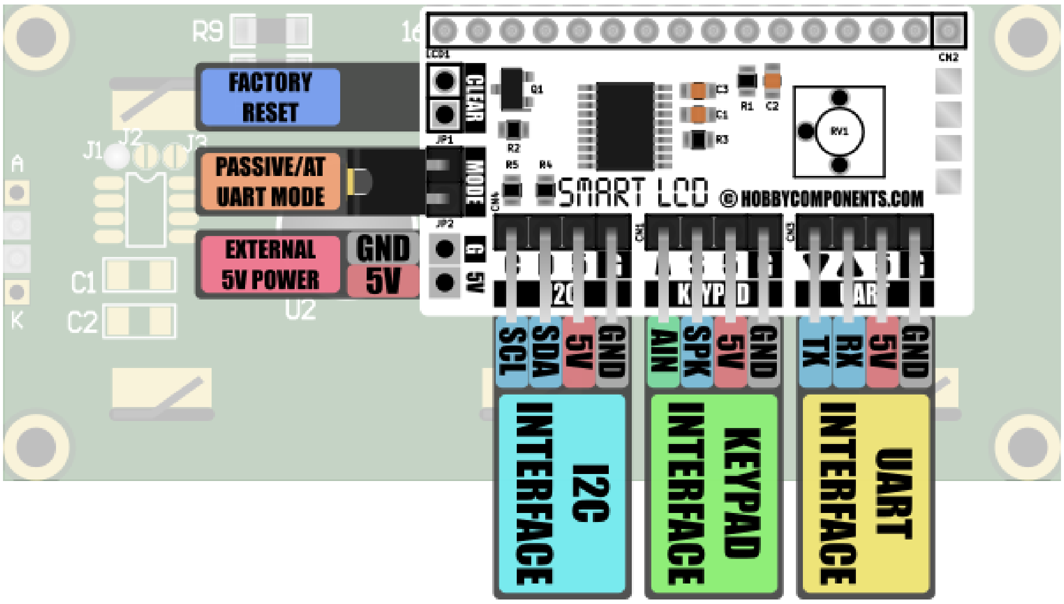
SOLDER PADS THIS SIDE



Any excess pins protruding from either the LCD or SmartLCD module headers can be optionally trimmed usings a pair of wire snips after soldering.

Note: By default the SmartLCD is configured for a 16 character by 2 line display. If you have fitted the SmartLCD to a 20 character by line display you will need to initialise the display to 20x4 display mode to take advantage of the 20x2 LCDs additional character resolution. To do this see the ‘Clear (Factory Reset)’ section.

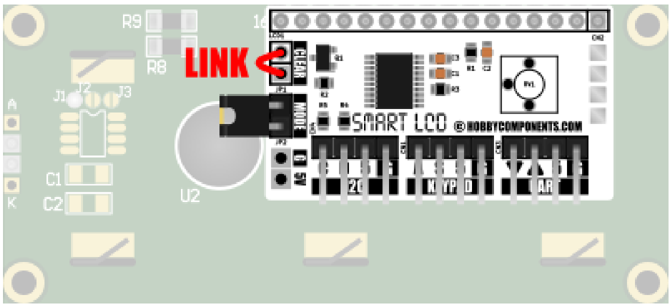
Interface



Interfacing to the Smart LCD can be done via the right-angled header pin or solder pads found around the edges of the Smart LCD module. The following sections provide descriptions and examples for each interface.

Clear (Factory Reset)

The SmartLCD is capable of storing a number of user configuration settings in the non-volatile memory. This allows the module to retain these settings even after power is removed. The two pads marked 'CLEAR' provide a means of restoring the module back to its factory default settings. Should you wish to reset the display simply connect power to the module whilst shorting the two pads together.

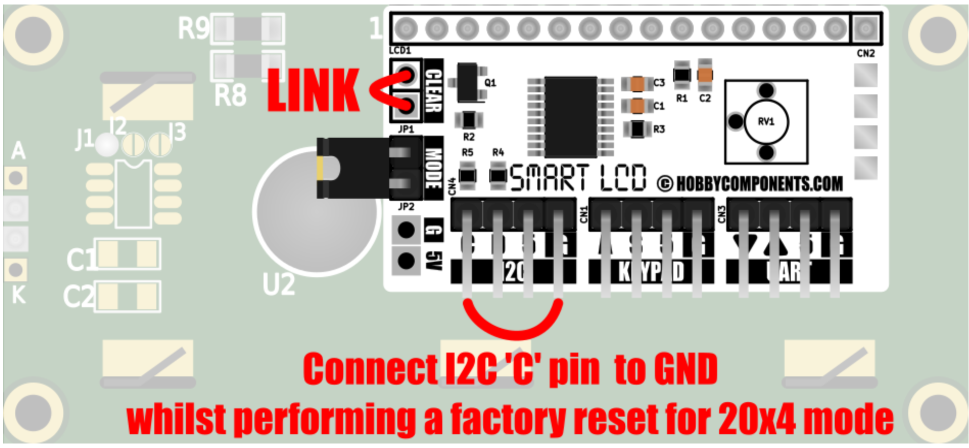


After power is applied to the module it will perform a simple factory test and reset its configuration settings to their default values. While the display is performing this process the screen will show various test information. At the end of the test the display will show confirmation of the default I2C address and serial baud rates. Once the reset process is complete you can then remove the link and restart the module.

Setting the SmartLCD to 16x2 or 20x4 character mode

Whilst performing a factory reset the SmartLCD can also be configured to either 16x2 or 20x4 display modes. By default the SmartLCD will be initialised into 16x2 mode when a factory reset is made.

If you have fitted the SmartLCD to a 20x4 display the SmartLCD can be set to 20x4 mode by connecting the 'C' pin on the SmartLCDs I2C interface header to any GND pin whilst performing a factory reset as described above:


LINK
CLEAR
MODE
SMART LCD
©HOBBYCOMPONENTS.COM
Connect I2C 'C' pin to GND
whilst performing a factory reset for 20x4 mode

Whilst performing the initialisation process the SmartLCD will confirm that it is in 20x4 mode by displaying '20x04 Mode' on the display.

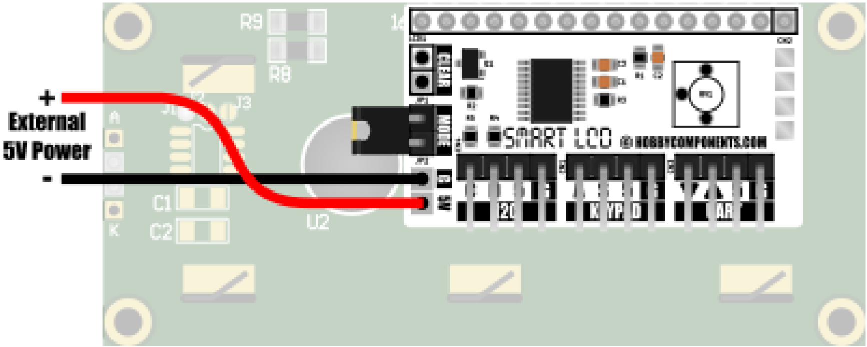
Once the factory reset process is complete both the 'Clear' and GND connection to the I2C 'C' pin can be removed. The SmartLCD will remain in 20x4 mode even after power is removed from the display and can only be changed back to 16x2 mode by performing a factory reset with the I2C 'C' pin left unconnected.

Mode (UART Terminal/AT mode)

The mode header allows selection of the serial interface operating mode. Using the supplied jumper allows for selection between passive terminal mode (jumper fitted) and AT command mode (jumper removed). See the UART Interface section below for more information about these modes.

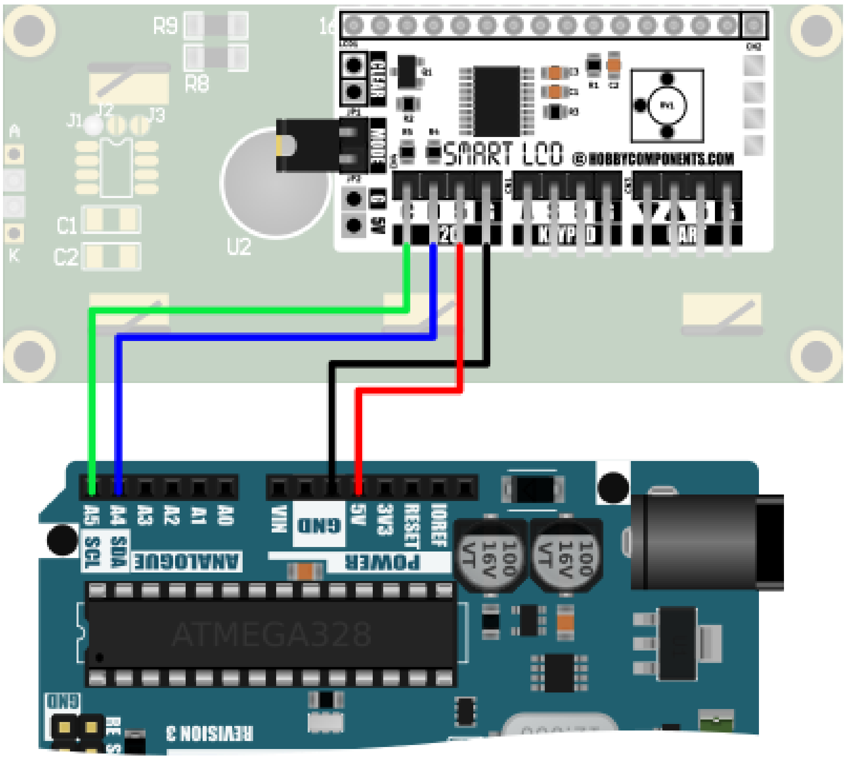
External 5V Power

Besides powering the SmartLCD module from any of the interface headers, the module can also be powered via a GND (0V) and 5V connection to solder pads marked ‘G’ and ‘5V’. This provides a convenient way to power the module from an external 5V power source.



I2C Interface

Allows two way communication with the Smart LCD from an I2C master device. When connected to the I2C interface the Smart LCD will behave as a standard I2C slave device with a default 7 bit I2C address of 0x27. This default address can be changed via a software I2C command (see command section). The module also includes built-in 10K pull-up resistors on the SDA and SCL bus communication lines. Using the I2C interface gives full control of all the Smart LCD’s display and control features.



Example: Interfacing to the I2C interface of an Uno development board

Serial UART Interface

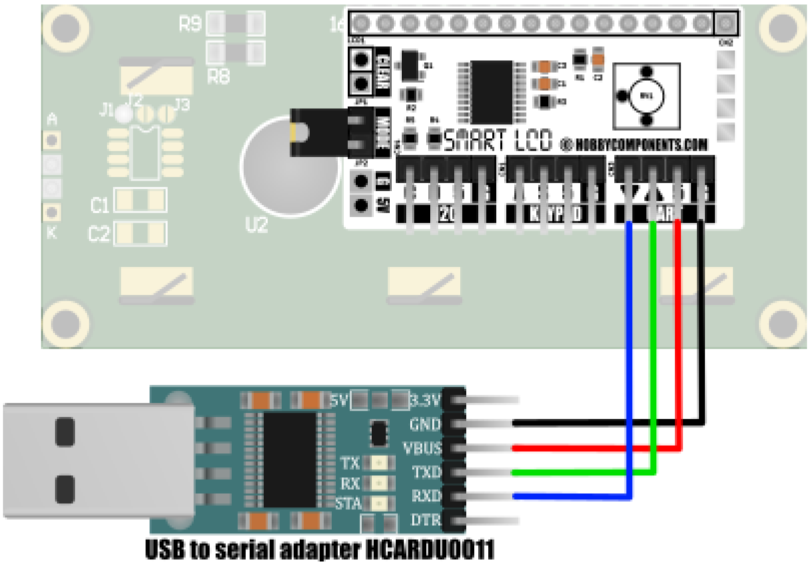
Allows two way communication with the SmartLCD from a standard serial UART device using 5V TTL logic levels. When using the UART to access the device there are two selectable modes of operation:

Terminal mode

Passive mode can be selected by connecting the supplied jumper across both MODE header pins. In this mode the SmartLCD will automatically display any printable ASCII text received via its UART Rx pin. This allows the module to be used as a simple terminal display. In this mode access to most of the Smart LCD's control features are not available, however the module will respond to some extended ASCII control codes providing limited screen control. See the software section of this manual for a list of ASCII extended control characters compatible with this module.

AT command mode

In AT command mode full access to the SmartLCD's features is available by simple AT text based commands. This includes outputting text and control of the screen, plus full access to the SmartLCD's features. For a list of available AT commands and how to use them please see the software section of this manual.



Example: Connecting the SmartLCD to a computer via a USB to serial adapter.

Software

Control of the SmartLCD can be made via the serial I2C and UART interfaces. In both cases full access to all the SmartLCD's features are available. I2C interface control and printing to the screen can be done via simple multi-byte I2C commands from a I2C master device. Serial UART interface control and printing to the screen is done by text based AT commands. The next sections of the document will describe how to communicate and control the SmartLCD via these interfaces.

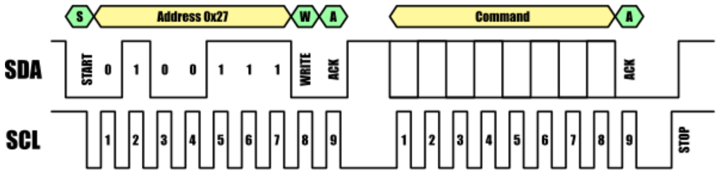
I2C

The SmartLCD will respond to data sent from an I2C bus master device. By default the SmartLCD has a slave address of 0x27 and will ignore data sent to any other I2C address. It supports both read and write operations that allow the bus master to send predefined commands to the SmartLCD, and also to read back the state of the I2C and status registers.

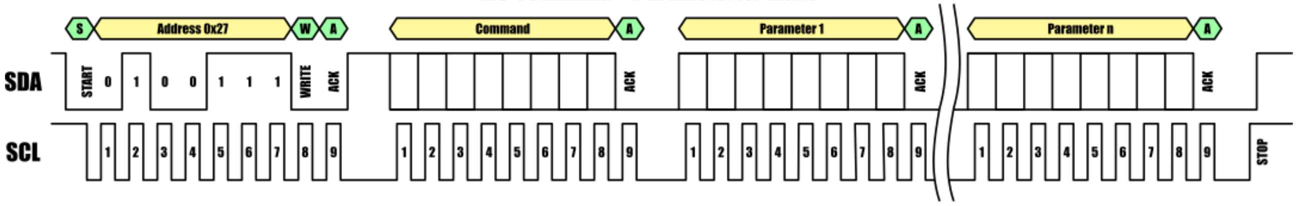
Master to slave write format

When writing to the SmartLCD from an I2C master, the format of the data consists of a 7 bit address byte (plus 1 R/W bit), a command byte. Following the command byte there may be one or more parameter bytes depending on the type of command sent in the command byte. Both the command and parameter bytes must be sent in the same write session.

I2C Command Write



I2C Command + Parameter(s) Write



I2C commands

Command	Code (hex)	Parameter Number	Parameter	Param Min	Param Max
Print character	0x00	1	ASCII code for character to be printed	0x20	0x7E
Clear screen	0x01	NA	NA	NA	NA
Set cursor position	0x02	1	Row index	0x00	0x03
		2	Column index	0x00	0x4F
Display state	0x03	1	0x00 (LCD OFF) or 0x01 (LCD ON)	0x00	0x01
Display cursor	0x04	1	0x00 (Cursor OFF) or 0x01 (Cursor ON)	0x00	0x01
Cursor blink	0x05	1	0x00 (Blink disabled) or 0x01 (Blink enabled)	0x00	0x01
Backlight state	0x06	1	On time in 10% increments	0x00	0x0A
Print custom character	0x07	1	Custom character index	0x00	0x07
Define custom character	0x08	1	Custom character index	0x00	0x07
		2	Character row 0 bitmap	0x00	0x1F
		3	Character row 1 bitmap	0x00	0x1F
		4	Character row 2 bitmap	0x00	0x1F
		5	Character row 3 bitmap	0x00	0x1F
		6	Character row 4 bitmap	0x00	0x1F
		7	Character row 5 bitmap	0x00	0x1F
		8	Character row 6 bitmap	0x00	0x1F
		9	Character row 7 bitmap	0x00	0x1F
Cursor direction	0x09	1	0x00 = Left to right or 0x01 = Right to left	0x00	0x01
Initialise the display	0x0A	NA	NA	NA	NA
Clear button state	0x0B	NA	NA	NA	NA
Play note	0x0C	1	Note	0x00	0x0B
		2	Duration in 10ms increments	0x00	0xFF
I2C address	0x0D	1	I2C address	0x08	0x77
Contrast	0x0E	1	Contrast level (0 = max, 255 = min)	0x00	0xFF
Display firmware version	0x0F	NA	NA	NA	NA

Table2: I2C command lookup table

Print character (0x00)

Address	Command	Parameter 1
0x27	0x00	ASCII Code (0x20 to 0x7E)

Writes a single printable ASCII character to the screen at the current cursor position. Once the character is printed the cursor is then automatically moved to the next character position.

Parameter(s):

The command has one parameter which specifies the ASCII character code to print. Printable characters range between 0x20 (‘ ‘) and 0x7E (‘~’).

Clear screen (0x01)

Address	Command
0x27	0x01

Clears the entire screen. Issuing a clear command will also set the position of the cursor to row 0 and column 0.

Parameter(s):

None.

Set cursor position (0x02)

Address	Command	Parameter 1	Parameter 2
0x27	0x02	Row (0x00 to 0x03)	Column (0x00 to 0x4F)

Sets the row and column position of the cursor where row 0, column 0 is the top left character position of the screen.

Parameter(s):

There are two parameters, the first sets the cursor row position. Note that on a 16 x 2 line display only rows 0 to 1 are visible.

The second parameter sets the cursors column position. Note that on a 16 x 2 line display only columns 0 to 15 are visible. On a 20 x 4 line display only columns 0 to 19 are visible.

Display State (0x03)

Address	Command	Parameter 1
0x27	0x03	LCD State (0x00 to 0x01)

Turns the screen on or off. Note that when turned off only the LCD panel will be switched off, the backlight will remain on if it is already enabled.

Parameter(s):

The command has one parameter which specifies the state of the LCD. A value of 0 will turn the LCD off, and a value of 1 will turn the LCD on.

Display Cursor (0x04)

Address	Command	Parameter 1
0x27	0x04	Cursor State (0x00 to 0x01)

Sets the visible state of the cursor. When visible, the cursor will appear as a single line of pixels below the current selected character row.

Parameter(s):

The command has one parameter which specifies the visible state of the cursor. A value of 0 will make the cursor invisible, and a value of 1 will make the cursor visible.

Cursor Blink (0x05)

Address	Command	Parameter 1
0x27	0x05	Blink State (0x00 to 0x01)

Enables or disables blinking of the cursor.

Parameter(s):

The command has one parameter which specifies the blink state. A value of 0 will disable blinking, and a value of 1 will enable cursor blinking.

Backlight State (0x06)

Address	Command	Parameter 1
0x27	0x06	Brightness (0x00 to 0x0A)

Sets the brightness of the backlight.

Parameter(s):

The command has one parameter which specifies the brightness of the backlight in 10% increments where 0 = backlight off and a value of 10 = 100% on.

Print Custom Character (0x07)

Address	Command	Parameter 1
0x27	0x07	Index (0x00 to 0x07)

Prints one of the eight custom (user defined) characters to the screen at the current cursor position. Note that the custom character to be printed must have been previously defined using the write custom character command (0x08).

Parameter(s):

The command has one parameter that specifies which of the eight custom characters (0 to 7) to print to the screen.

Define Custom Character (0x08)

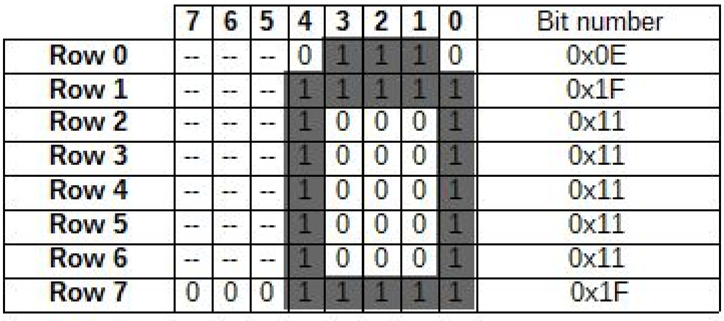
Address	Command	Parameter 1	Parameter 2	Parameter 3	Parameter 9
0x27	0x08	Index (0x00 to 0x07)	Row 0 (0x00 to 0x1F)	Row 1 (0x00 to 0x1F	Row 7 (0x00 to 0x1F)

Writes a user defined 5 x 8 pixel bitmap to one of the eight custom characters. Note that on 16x02 and 20x04 displays the eighth character row is not visible. After writing to one of the custom characters it is necessary to reset the cursor position before any further characters can be written to the screen.

Parameter(s):

The command has 9 parameters. The first specifies which of the eight (8) custom characters (0 to 7) to write the bitmap data too.

Parameters 2 to 9 contain the 5 pixel bitmap information for each of the 8 rows of the custom characters.



	7	6	5	4	3	2	1	0	Bit number
Row 0	--	--	--	0	1	1	1	0	0x0E
Row 1	--	--	--	1	1	1	1	1	0x1F
Row 2	--	--	--	1	0	0	0	1	0x11
Row 3	--	--	--	1	0	0	0	1	0x11
Row 4	--	--	--	1	0	0	0	1	0x11
Row 5	--	--	--	1	0	0	0	1	0x11
Row 6	--	--	--	1	0	0	0	1	0x11
Row 7	0	0	0	1	1	1	1	1	0x1F

Example bit pattern

Cursor Direction (0x09)

Address	Command	Parameter 1
0x27	0x09	Cursor Direction (0x00 to 0x01)

Sets the direction in which the cursor will move after a character is printed to the screen.

Parameter(s):

The command has one parameter that specifies which direction to move the cursor. A value of 0 will move the cursor from left to right, and a value of 1 will move the cursor from right to left.

Initialise The Display (0x0A)

Address	Command
0x27	0x0A

Initialises the LCD setting it to its default power on state. If the backlight was off it would also be set to its default on state.

Parameter(s):

None

Clear Button State (0x0B)

Address	Command
0x27	0x0B

Clears (resets) the button number bits in the status register. See the I2C status register section for more information about the button status and number bits.

Parameter(s):

None

Play Note (0x0C)

Address	Command	Parameter 1	Parameter 2
0x27	0x0C	Note (0x00 to 0x0B)	Duration (0x00 to 0xFF)

Plays one of 12 musical notes for a specified duration. Note that this command requires the optional keypad and speaker accessory to be connected to the SmartLCD.

Parameter(s):

This command has two parameters, the first selects the note to play. See table below for a list of available notes.

Value	Note	Frequency (Hz)
0x00	C	262.2
0x01	C#	277.8
0x02	D	294.1
0x03	D#	312.5
0x04	E	322.6
0x05	F	357.2
0x06	F#	370.4
0x07	G	384.6
0x08	G#	416.7
0x09	A	434.8
0x0A	A#	476.2
0x0B	B	500

Note that frequencies are generated by the SmartLCDs RC oscillator. As a result note frequencies shown may vary by +/- 10%

The second parameter sets the duration of the note in 10ms increments where a minimum value of 1 = 10 milliseconds and a maximum value of 0xFF = 2.55 seconds.

Set I2C address (0x0D)

Address	Command	Parameter 1
0x27	0x0D	New Address (0x08 to 0x77)

Changes the I2C slave address for the SmartLCD. Once this command is issued the display will immediately respond to commands sent to the new slave address and will no longer respond to the original address.

Parameter(s):

The command has one parameter that specifies the new I2C slave address. A slave address can be any address from 0x08 to 0x77. The new address will be stored in the SmartLCD's non-volatile memory and it will continue to use this address even after the SmartLCD's power is cycled. Performing a factory reset will restore the current I2C address to the default 0x27.

Contrast (0x0E)

Address	Command	Parameter 1
0x27	0x0E	Contrast Level (0x00 to 0xFF)

Sets the contrast level of the screen.

Parameter(s):

The command has one parameter that specifies the contrast level. A value of 0 = maximum contrast and a value of 0xFF = minimum contrast.

Display Software Version (0x0F)

Address	Command
0x27	0x0F

Shows the version of firmware flashed into the SmartLCD by displaying it to the LCD.

Parameter(s):

None

Clear Error State (0xFB)

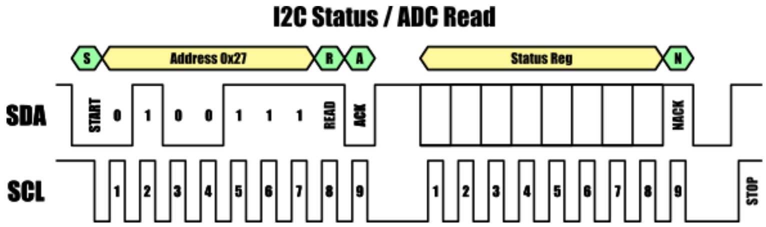
Address	Command
0x27	0xFB

Clears the error flag in the status register.

Parameter(s):

None

I2C Status Register



The contents of the status register can be obtained by performing a single byte read from an I2C master device.

7	6	5	4	3	2	1	0
BUSY	STATUS	RESERVED	RESERVED	BPS	BTN2	BTN1	BTN0

Bit 7 **BUSY:** SmartLCD busy status

0: SmartLCD is ready to accept new commands.

1: SmartLCD is busy executing the last command.

Bit 6 **STATUS:** SmartLCD error status

0: Last command executed without error.

1: There was an error executing or receiving the last command.

Bit 5:4 RESERVED

Bit 3 **BPS:** Keypad button pressed status

0: No buttons are currently pressed on the external keypad.

1: A button on the external keypad is being pressed.

Bit 2:0 **BPS[2:0]:** Last button pressed on external keypad

0: No buttons have been pressed.

1: Button 1 (BACK) has been pressed

2: Button 2 (SELECT) has been pressed

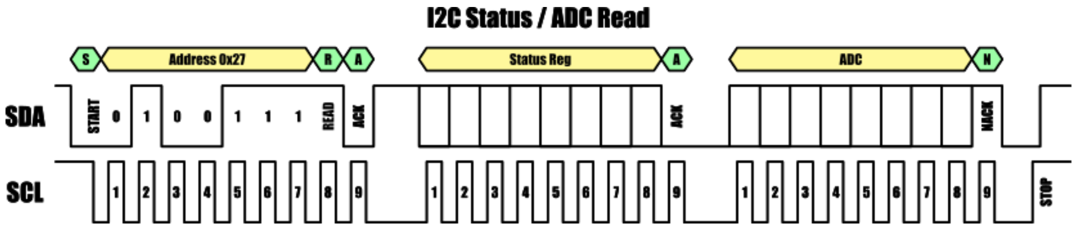
3: Button 3 (RIGHT) has been pressed

4: Button 4 (DOWN) has been pressed

5: Button 5 (UP) has been pressed

6: Button 6 (LEFT) has been pressed

I2C ADC Register



The smart LCD includes an 8-bit ADC which can be accessed via the A pin on the keypad header. When the external SmartLCD keypad is connected it is used for sensing the state of the keypad buttons. However, if the external keypad is not connected this pin can be used as a general purpose 8-bit ADC.

To obtain the current state of the ADC a 2 byte read must be made from the I2C master device. The first byte read back will contain the status register and may be ignored if not required. The second byte will contain the 8 bit data representing the current analogue voltage at the ADC pin where a value of 0 = GND and a value of 255 = VCC.

7	6	5	4	3	2	1	0
ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0

Bit 7:0 **ADC[7:0]:** Current value of the 8-bit ADC.

Serial UART – AT Command Mode

Command	Description	Parameter	Response
AT	Test connection	None	OK
AT+PR=String	Print ASCII string	String: ASCII string to be printed	OK
AT+CLR	Clear screen	None	
AT+RC=Row,Col	Cursor position	Row: Row number Col: Column number	OK
AT+DS=State	Display state	State: Specifies the display state (ON/OFF)	OK
AT+DC=State	Display cursor	State: Specifies the cursor state (ON/OFF)	OK
AT+DB=State	Cursor blink	State: Specifies the cursor blink state (ON/OFF)	OK
AT+BL=Level	Set the backlight brightness level	Level: Backlight brightness level (0 = 0%, 10 = 100%)	OK
AT+PC=Index	Print custom character	Index: Index of the special character (0 to 7) to print	OK
AT+SC=Index,Row0,Row1, Row2,Row3,Row4, Row5,Row6,Row7	Define custom character	Index: Index of the special character (0 to 7) to define Row0: Bitmap data for row 0 (0 to 31) Row1: Bitmap data for row 1 (0 to 31) Row2: Bitmap data for row 2 (0 to 31) Row3: Bitmap data for row 3 (0 to 31) Row4: Bitmap data for row 4 (0 to 31) Row5: Bitmap data for row 5 (0 to 31) Row6: Bitmap data for row 6 (0 to 31) Row7: Bitmap data for row 7 (0 to 31)	OK
AT+CD=Direction	Set cursor direction	Direction: Direction to move cursor (R = left to right, L = Right to left)	OK
AT+BR=Baud	Change the UART baud rate	Baud: Index of new baud rate (0 to 7)	OK
AT+TO=Time	Change AT command timeout	Time: Timeout time in ms (min = 1ms, max = 6.5 seconds)	OK
AT+INI	Initialise the display	None	OK
AT+BT?	Get last button pressed*	None	Button number (0 to 5)
AT+BP?	Get button pressed state*	None	Button pressed status (F = not pressed, T = pressed)
AT+BCL	Clear button state*	None	OK
AT+AD?	Read the ADC	None	ADC value (0 to 255)
AT+BE=Note,Time	Play a musical note*	Note: Index of note to play (0 to 11) Time: Duration of note in 10ms increments (10ms to 2.55 seconds)	OK
AT+CT=Level	Contrast level	Level: Contrast level (0 = max, 255 = min)	OK
AT+FV?	Get software version	None	Software version

** Requires optional SmartLCD keypad to be connected*

The SmartLCD will respond to text based commands prefixed with the AT command designator via its serial UART interface. To put the SmartLCD into AT command mode, the Mode jumper on the LCD daughter board must be removed. AT commands can then be sent to, and responses received from, the SmartLCD with a default serial baud rate of 9600bps, 8 data bits, 1 stop bit and no parity. Additionally, commands must be terminated with a carriage return (CR) + line feed (LF). By using serial AT commands full control of the SmartLCD is available, including printing text to the screen and reading of the status and ADC registers. The following is a list of available AT commands:

AT

Can be used to test the connection to the SmartLCD. Sending the ASCII text AT will cause the Smart LCD to respond with an 'OK'

Example: AT

Response: OK

AT+PR=*String*

Prints a string of text at the current cursor position to the screen, where string is the ASCII string of text to print. Maximum string length is 30 character bytes – any additional characters will be ignored.

Example: AT+PR=Hello World *Prints Hello World to the screen*

Response: OK

AT+CLR

Clears the entire screen. Issuing a clear command will also set the position of the cursor to row 0 and column 0.

Example: AT+CLR *Clears the entire screen*

Response: OK

AT+RC=*Row*,*Col*

Sets the row and column position of the cursor where row 0, column 0 is the top left character position of the screen.

There are two parameters, the first sets the cursor row position. Note that on a 16 x 2 line display only rows 0 to 1 are visible.

The second parameter sets the cursors column position. Note that on a 16 x 2 line display only columns 0 to 15 are visible. On a 20 x 4 line display only columns 0 to 19 are visible.

Example: AT+RC=0,0 *Sets the cursor to row 0, column 0*

Response: OK

AT+DS=*State*

Turns the screen on or off. Note that when turned off only the LCD panel will be switched off, the backlight will remain on if it is already enabled.

The command has one parameter which specifies the state of the LCD. A value of OFF will turn the LCD off, and a value of ON will turn the LCD on.

Example: AT+DS=OFF *Turns the screen off*

Response*: OK*

AT+DC=*State*

Sets the visible state of the cursor. When visible the cursor will appear as a single line of pixels below the currently selected character row.

The command has one parameter which specifies the visible state of the cursor. A value of OFF will make the cursor invisible, and a value of ON will make the cursor visible.

Example: AT+DC=ON *Turns the cursor on*

Response: OK

AT+DB=*State*

Enables or disables blinking of the cursor.

The command has one parameter which specifies the blink state. A value of OFF will disable cursor blinking, and a value of ON will enable cursor blinking.

Example: AT+DB=ON *Turns cursor blinking on*

Response: OK

AT+BL=*Level*

Sets the brightness of the backlight.

The command has one parameter which specifies the brightness of the backlight in 10% increments where 0 = backlight off and a value of 10 = 100% on.

Example: AT+BL=10 *Sets the backlight to 100%*

Response: OK

AT+PC=*Index*

Prints one of the 8 custom (user defined) characters to the screen at the current cursor position. Note that for the custom character to be printed it must have previously been defined using the define custom character command (AT+SC).

The command has one parameter that specifies which of the 8 custom characters (0 to 7) to print to the screen.

Example: AT+PC=0 *Prints special character 0 to the screen*

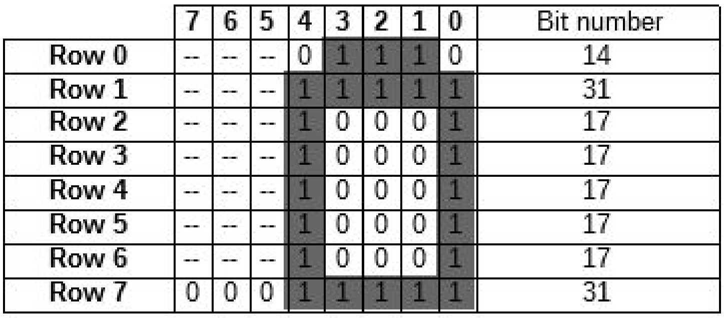
Response: OK

AT+SC=*Index,Row0,Row1,Row2,Row3,Row4,Row5,Row6,Row7*

Writes a user defined 5 x 8 pixel bitmap to one of the 8 custom characters. Note that on 16x02 and 20x04 displays the eighth character row is not visible. After writing to one of the custom characters it is necessary to reset the cursor position before any further characters can be written to the screen.

The command has 9 parameters. The first specifies which of the 8 custom characters (0 to 7) to write the bitmap data too.

Parameters 2 to 9 contain the 5 pixel bitmap information for each of the 8 rows of the custom character. Note that these parameters must be specified in ASCII decimal.



	7	6	5	4	3	2	1	0	Bit number
Row 0	--	--	--	0	1	1	1	0	14
Row 1	--	--	--	1	1	1	1	1	31
Row 2	--	--	--	1	0	0	0	1	17
Row 3	--	--	--	1	0	0	0	1	17
Row 4	--	--	--	1	0	0	0	1	17
Row 5	--	--	--	1	0	0	0	1	17
Row 6	--	--	--	1	0	0	0	1	17
Row 7	0	0	0	1	1	1	1	1	31

Example bit pattern

Example: AT+SC=0,14,31,17,17,17,17,17,31 *Writes a bit pattern to special character 0*

Response: OK

AT+CD=*Direction*

Sets the direction the cursor will move after a character is printed to the screen.

The command has one parameter that specifies which direction to move the cursor. A value of R will move the cursor from left to right, and value of L will move the cursor from right to left.

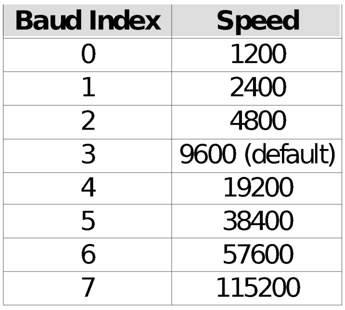
Example: AT+PC=R *Set the cursor direction to move from left to right*

Response: OK

AT+BR=*Baud*

Changes the default baud rate from 9600 baud to one of 8 different speeds. Note: any new value set will be stored in the SmartLCD's non-volatile memory.

The command has one parameter which specifies which baud to set the baud rate to.



Baud Index	Speed
0	1200
1	2400
2	4800
3	9600 (default)
4	19200
5	38400
6	57600
7	115200

Example: AT+BR=5 *Sets the baud rate to 38400 baud*

Response: OK

AT+TO=*Time*

Sets the serial communications timeout time if communication with the SmartLCD is interrupted or an error occurs. Note: default timeout time is 2 seconds, any new value set will be stored in the Smart LCD's non-volatile memory.

The command has one parameter which specifies the new timeout time in 1ms increments. Minimum time is 1ms (*Time* = 1) and maximum time is 6.5 seconds (*Time* = 6500).

Example: AT+TO=1000 *Set the serial timeout time to 1 second*

Response: OK

AT+INI

Initialises the LCD setting it to its default power on state. If the backlight is off it will also be set to the default on state.

Example: AT+INI *The Smart LCD is initialised*

Response: OK

AT+BT?

Gets the SmartLCD button number for the last button to be pressed. If no button has been pressed since the SmartLCD was powered or an AT+BCL command was sent then the command will return an index value of 0.

Example: AT+BT? *Gets the last button pressed state*

Response: 0 = No buttons have been pressed.

1 = Button 1 (BACK) has been pressed

2 = Button 2 (SELECT) has been pressed

3 = Button 3 (RIGHT) has been pressed

4 = Button 4 (DOWN) has been pressed

5 = Button 5 (UP) has been pressed

6 = Button 6 (LEFT) has been pressed

AT+BP?

Checks if any of the SmartLCD keypad buttons are currently being pressed.

Example: AT+BP? *Gets the current button pressed status*

Response: F = No buttons are currently pressed on the external keypad.

T = A button on the external keypad is being pressed.

AT+BCL

Clears (resets) the button number status in the status register. When cleared the button number (AT+BT?) and button pressed state (AT+BP?) AT commands will return 0 and F respectively. Note that if this command is sent whilst a button is currently being pressed, neither the current button pressed or button number status will not be cleared.

Example: AT+BCL *Clear the button pressed and button number status.*

Response: OK

AT+AD?

Gets the current value of the 8-bit ADC connected to pin ‘A’ on the keypad connector.

Example: AT+AD? *Gets the current ADC value*

Response: Returns a value between 0 (GND) and 255 (VCC)

AT+BE=*Note*,*Time*

Plays one of 12 musical notes for a specified duration. Note that this command requires the optional SmartLCD keypad and speaker accessory to be connected to the SmartLCD.

This command has two parameters:

Note selects the index of the note to play. See table below for a list of available notes.

Value	Note	Frequency (Hz)
0x00	C	262.2
0x01	C#	277.8
0x02	D	294.1
0x03	D#	312.5
0x04	E	322.6
0x05	F	357.2
0x06	F#	370.4
0x07	G	384.6
0x08	G#	416.7
0x09	A	434.8
0x0A	A#	476.2
0x0B	B	500

Note that frequencies are generated by the SmartLCDs RC oscillator. As a result note frequencies shown may vary by +/- 10%

Time sets the duration of the note in 10ms increments where a minimum value of 1 = 10ms and a maximum value of 255 = 2.55s

Example: AT+BE=1,100 *Plays C# for 1 second*

Response: OK

AT+CT=*Level*

Sets the contrast level of the screen.

The command has one parameter that specifies the contrast level. A value of 0 = maximum contrast and a value of 255 = minimum contrast. The SmartLCD ships with a default contrast level of 55

Example: AT+CT=0	*Set the contrast level to maximum*
Response: OK	

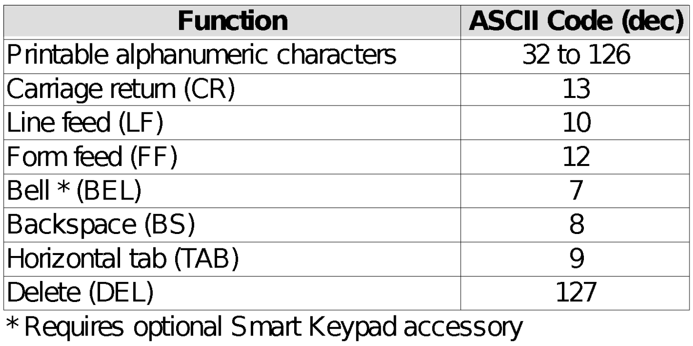
AT+FV?

Gets the software version number for the installed firmware.

Example: AT+FV?	*Gets the firmware version*
Response:	Vx.x
	© HobbyComponents.com

Serial UART – Passive Mode

When connected to the serial UART interface the SmartLCD can also operate in a passive mode where it will automatically display any printable ASCII text to the screen. In this mode no commands are required, however, control over the LCD is limited. To put the SmartLCD into passive mode simply connect the supplied jumper across the Mode header pins. When in passive mode the SmartLCD will respond to ASCII data sent to it with a baud rate of 9600bps, 8 data bits, 1 stop bit and no parity.

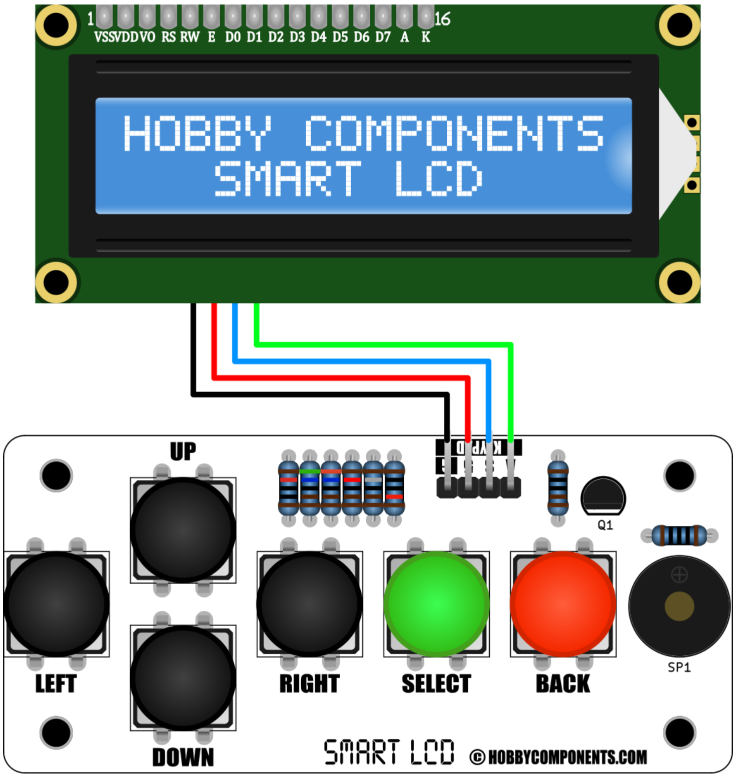


Function	ASCII Code (dec)
Printable alphanumeric characters	32 to 126
Carriage return (CR)	13
Line feed (LF)	10
Form feed (FF)	12
Bell * (BEL)	7
Backspace (BS)	8
Horizontal tab (TAB)	9
Delete (DEL)	127

* Requires optional Smart Keypad accessory

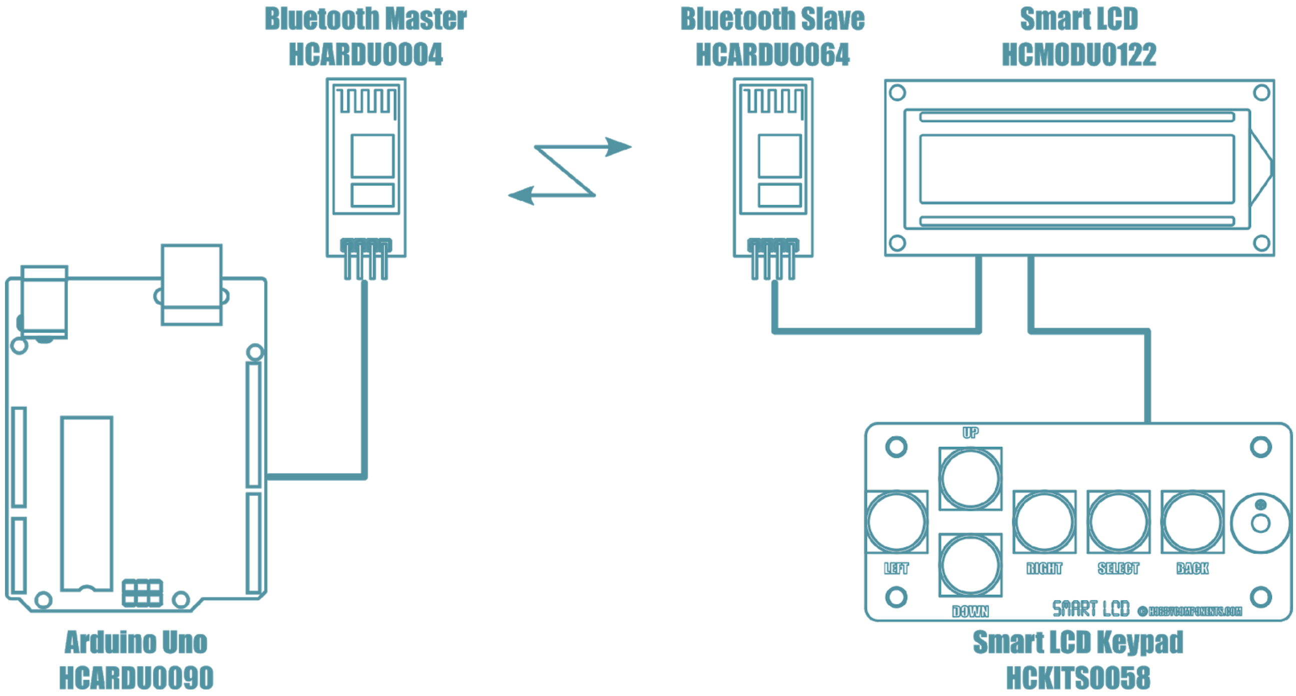
Optional Features - LCD Keypad

The SmartLCD's features can be further expanded with the addition of the SmartLCD keypad (see item SKU: HCKITS0058). This add-on board connects to the Smart LCD's keypad header using a single 4 way cable. Once connected it adds the additional features of keypad control (up, down, left, right, select, and back) and audible output to your end application. Access to button states and speaker output are then available via the Smart LCD's I2C or UART interfaces using its built-in commands (see the software section of this manual).



Optional Features – Wireless access

A bonus feature of the SmartLCD, due to it not requiring initialisation by a microcontroller, is it's ability to operate wirelessly using a suitable serial wireless pass-through module. In the example diagram below you can see a remote control application of the SmartLCD and keypad with the addition of HC-05 Bluetooth slave (see item SKU: HCARDU0004) and Bluetooth master (see items SKU: HCARDU0064) modules.



In this example a Bluetooth slave module is connected to the SmartLCD's UART interface and a Bluetooth master module to the microcontrollers serial interface. Once a wireless connection is established, AT commands can be sent to the SmartLCD when in serial AT mode, or ASCII text when in serial passive mode, just as if the SmartLCD was connected directly to the microcontroller.

Note that when using a wireless serial connection the SmartLCD is not able to handle corruption or loss of data due to connection errors. Therefore you should make sure your application is able to cater for connection issues caused by wireless interference.

FOR NEW PRODUCT ANNOUNCEMENTS AND OFFERS:

LIKE US ON FACEBOOK:

facebook

/HOBBYCOMPONENTSLTD

AND

FOLLOW US ON TWITTER:

twitter

/HOBBYCOMPONENTS

WWW.HOBBYCOMPONENTS.COM

FORUM.HOBBYCOMPONENTS.COM

BLOG.HOBBYCOMPONENTS.COM