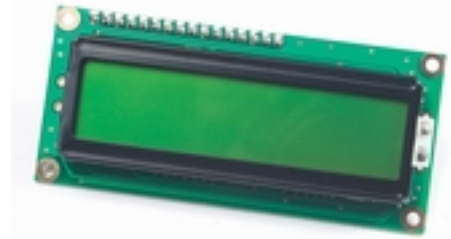


LCD 2x16A Module User's Guide

Version: V1.0

Product Overview: Innovati's LCD 2x16 A Module provides versatile display functions. Through its simple connections, it can be controlled by Innovati's BASIC Commander for a wide range of LCD applications. In this module, two display lines, each with 16 characters on each line can be displayed. By using the cursor control command, the position of the character to be displayed on the screen can be arbitrarily changed. In this module, the backlight function can be used to change the backlight to allow the message to be read easily. In addition, it can be configured to display user defined characters to display any specially required characters. Please use "LCD2x16A" as the module object name in program.



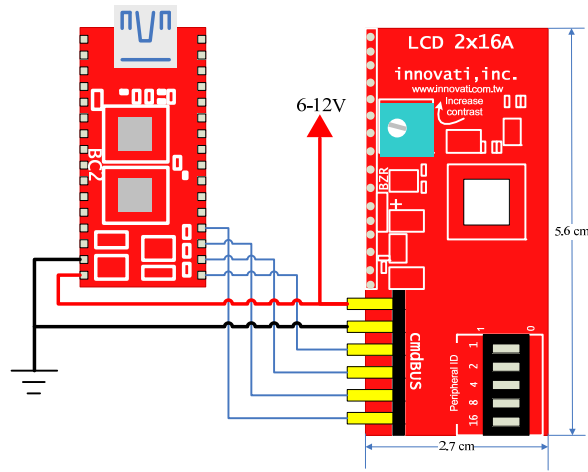
Application:

- Together with an RTC Module, it can be used to display a real time clock or a simple electronic clock.
- It can be used to display the operating status at any time for various applications.
- It can display status or error messages directly on the screen without using the PC.
- With the user-defined characters, special patterns can be created to produce creative messages.

Product Features:

- It can be used to display corresponding characters in ASCII code.
- The module will automatically convert and display the data according to its data type.
- 255 steps backlight control.
- For continuous inputs, the module will carriage return automatically
- Cursor position assignment and Tab function with configurable Tab steps and HOME function.
- Destructive backspace, clear to end of line or end of screen from the cursor position.
- Set the user defined characters to display various creative characters.
- Display off command to reduce power consumption.

Connection: Directly setup the ID switches to the required number, and then connect the cmdBUS cable to the corresponding pins on the BASIC Commander (shown in the following figure). Then the required operations can be performed through the BASIC Commander. DC power (6~12V) and ground should be connected to VIN and GND pin.



Product Specifications:

Connect these pins to the corresponding pins on the BASIC Commander with the cmdBUS cable. Then the LCD module can be controlled through the BASIC Commander. When connecting the pin, connect Vin to the Vin pin on the BASIC Commander. If the pins are incorrectly connected, the module may be damaged.

The module number setting switches. Set the module number of the LCD module in the binary format in the order from right to left. The module number is used for the BASIC Commander to determine the required module to be controlled during operations. (Refer to Appendix 2)

Contrast adjustment screw. Adjust with a Phillips screwdriver. By rotating clockwise, the contrast can be increased. By rotating counterclockwise, the contrast can be decreased.

The detailed view of the LCD 2x16A module shows the cmdBUS pins, Peripheral ID switches, and contrast adjustment screw. The module is labeled 'LCD 2x16A' and 'innovati, inc.' with the website 'www.innovati.com.tw'. The Peripheral ID switches are numbered 4, 3, 2, 1, 0 from right to left. The contrast adjustment screw is labeled 'INCREASE CONTRAST'.

Figure 1: Pin assignment and module switches

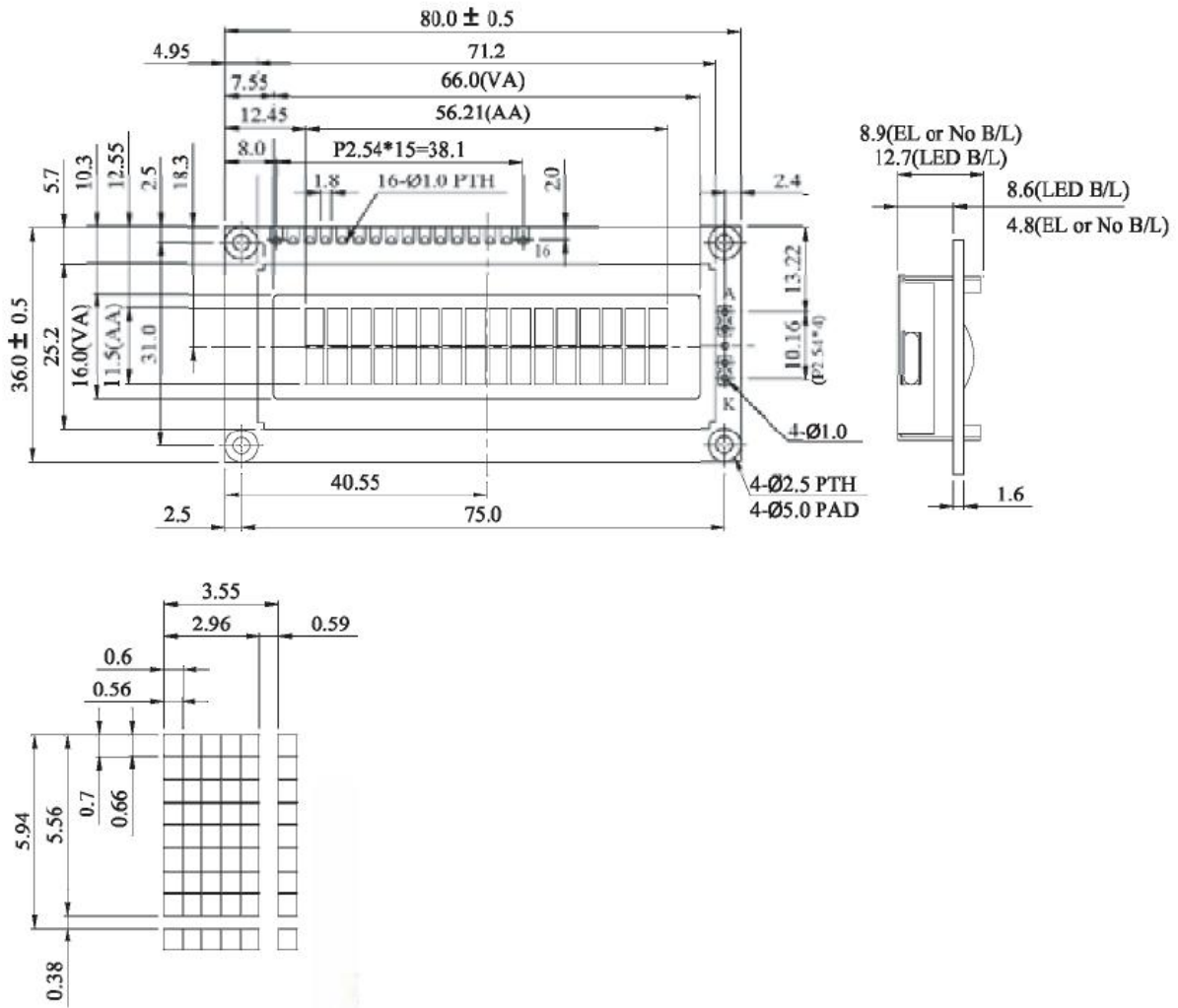


Figure 2: LCD panel specifications (unit : mm)

Item	Standard Value	Unit
Display type	16 characters x 2 Lines	—
Module dimension (L x W x H)	80.0 x 36.0 x 12.7 (Max) - LED array B/L STN Positive / 6 o'clock / Transflective	mm
Viewing Area	66.0 x 16.0	mm
Active Area	56.21 x 11.5	mm
Dot Size	0.56 x 0.66	mm
Dot Pitch	0.60 x 0.70	mm
Character size (L x W)	2.96 x 5.56	mm
Character pitch (L x W)	3.55 x 5.94	mm

Table 1: LCD panel mechanical dimensions

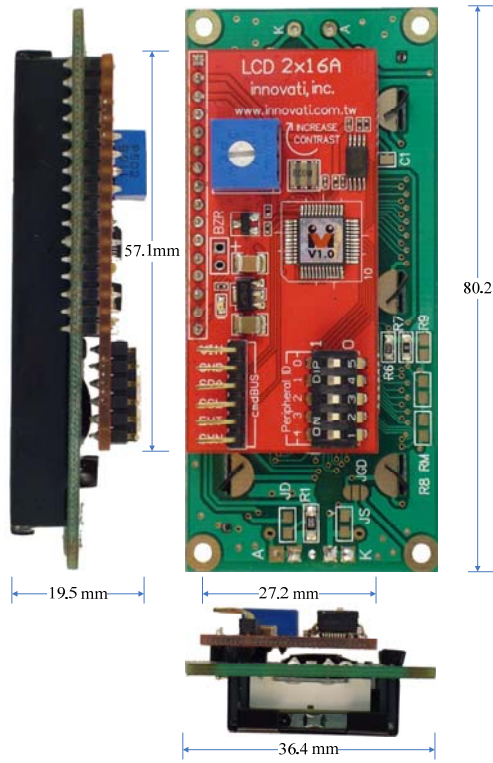


Figure 3: LCD panel mechanical dimensions

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
I _{DD}	Operating Current	7.5	Backlight On	—	180	—	mA
			Backlight Off	—	5	—	mA

Table 1: Operating current characteristic (ambient temperature 25 °C)

Absolute Maximum Ratings:

Operating Temperature : 0 °C~70 °C

Storage Temperature : -30 °C~ 80 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
View Angle	(V) θ	CR \geq 2	10		45	deg
	(H) φ	CR \geq 2	-30		30	deg
Contrast Ratio	CR	—		3		—
Response Time 25°C	T rise	—		100	150	ms
	T fall	—		150	200	ms

Table 2: LCD panel viewing angle and contrast

Command Table:

The following table lists all the unique commands provided with the LCD 2x16A Module.

Note that essential words in the commands will be written in **bold** type and *italics* in bold type.

The bold type word must be written exactly as shown, whereas the italic bold type words must be replaced with the user values. Note that the innoBASIC language is case-insensitive.

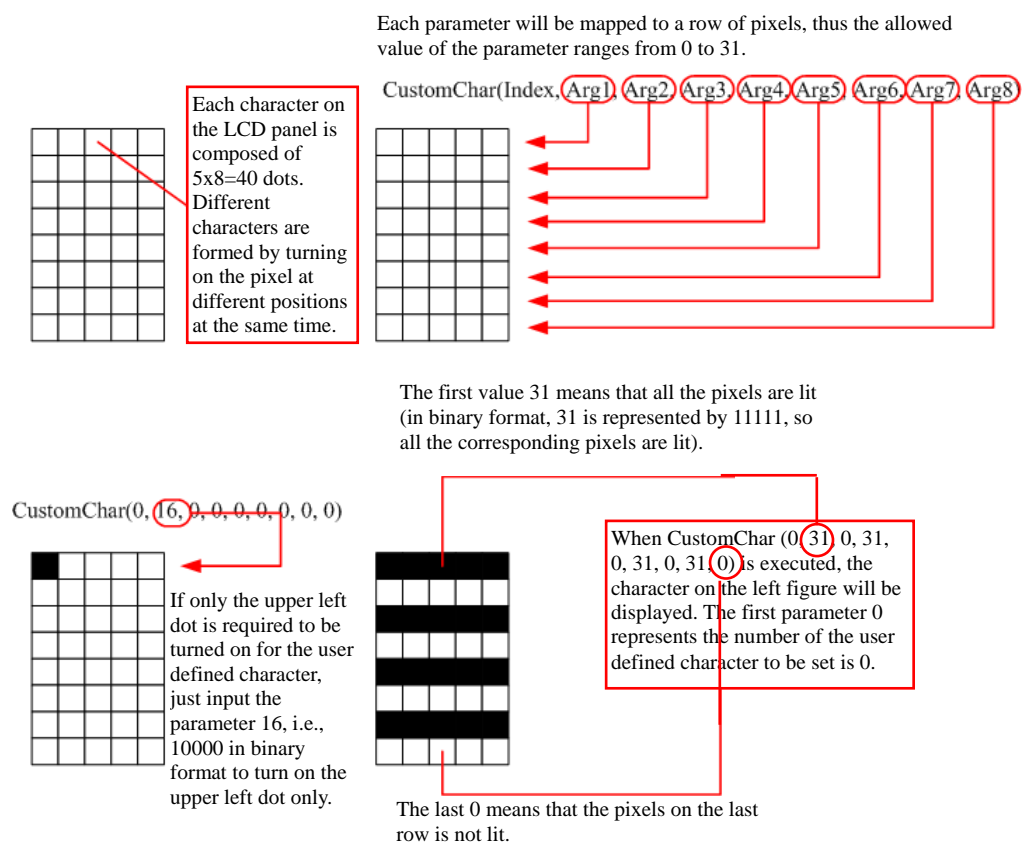
Command Format	Description
Cursor Control Commands	
CR()	Move the cursor to the beginning of the next line as carriage return.
CursorCol(<i>Col</i>)	Move the cursor to the column specified by the byte variable <i>Col</i> .
CursorDown()	Move the cursor down to the line below.
CursorLeft()	Move the cursor to the left by one character.
CursorRC(<i>Row, Col</i>)	Move the cursor to the row and the column specified by the byte variable <i>Row</i> and <i>Col</i> .
CursorRight()	Move the cursor to the right by one character.
CursorRow(<i>Row</i>)	Move the cursor to the row specified by the byte variable <i>Row</i> .
CursorUp()	Move the cursor up to the line above.
Home()	Move the cursor to the first column of the first row.
Tab()	Move the cursor to the right by the number of characters specified by SetTab command.
Clear Characters Commands	
BackSpace()	Move the cursor backward by one character, and then clear the character originally displayed at the position.
Clear()	Clear all the characters shown on the display.
ClearEOL()	Clear all the characters from the cursor position to the end of the line.
ClearEOS()	Clear all the characters from the cursor position to the end of the screen.
Displaying Characters Commands	
Display(<i>Parameter</i>)	The <i>Parameter</i> will be displayed according to its type. If the type is string, the string will be displayed directly; if it is a floating point number, it will be displayed in scientific notation; other numeric values will be displayed in decimal format.
DisplayBin(<i>IntegerNum</i>)	The <i>IntegerNum</i> will be displayed in binary format. The <i>IntegerNum</i> can be one of the BYTE, SHORT, WORD, INTEGER, DWORD or LONG. type.
DisplayChar(<i>Chr, ...</i>)	Display the characters specified by the byte variable <i>Chr</i> . The values 0~7 can be used to specify the corresponding user-defined custom characters. Multiple characters separated by a comma are allowed. The input value will be displayed as characters according to its ASCII code. Please refer to

	Appendix 3.
DisplayFloat(<i>FloatNum</i>, <i>Digits</i>)	Set the <i>FloatNum</i> of effective digits specified by the byte variable <i>Digits</i> and displayed in scientific format.
DisplayHex(<i>IntegerNum</i>)	Display the <i>IntegerNum</i> in hexadecimal format. The <i>IntegerNum</i> can be one of the BYTE, SHORT, WORD, INTEGER, DWORD or LONG type.
DisplayLeft(<i>Parameter</i>, <i>Width</i>)	According to the width specified by the byte value <i>Width</i> , <i>Parameter</i> is displayed in decimal format aligned to the left. If the <i>Parameter</i> length exceeds the width, it will be automatically adjusted to a proper width. The Parameter cannot be a string.
DisplayReal(<i>FloatNum</i>, <i>Digits</i>)	Set the <i>FloatNum</i> of effective digits specified by the byte variable <i>Digits</i> and displayed in Real number format.
DisplayRight(<i>Parameter</i>, <i>Width</i>)	According to the width specified by the byte value <i>Width</i> , <i>Parameter</i> is displayed in decimal format aligned to the right. If the <i>Parameter</i> length exceeds the width, it will be automatically adjusted to a proper width. The Parameter cannot be a string.
Miscellaneous Commands	
BacklightOff()	Turn off the backlight.
BacklightOn(<i>Time</i>)	Set the time interval specified by the byte value <i>Time</i> to turn on the backlight using the value of Time. If it is 0, the backlight will be constantly turned on.
CursorBlinkOff()	Stop the cursor blinking.
CursorBlinkOn()	Start the cursor blinking.
CursorOff()	Disable the cursor display.
CursorOn()	Enable the cursor display.
CustomChar(<i>Index</i>, <i>Arg1</i>, <i>Arg2</i>, <i>Arg3</i>, <i>Arg4</i>, <i>Arg5</i>, <i>Arg6</i>, <i>Arg7</i>, <i>Arg8</i>)	Set the character with ASCII code <i>Index</i> , ranging from 0 to 7. The byte values <i>Arg1</i> ~ <i>Arg8</i> represent the patterns to be displayed on each row of the user-defined character and the corresponding dots will be lit by the specified values in binary format on the display. (See Note 1.)
DisplayOff()	Turn off the display.
DisplayOn()	Turn on the display.
GetTab(<i>TabCount</i>)	Get the preset Tab value and store it in the byte variable <i>TabCount</i> .
RotateLeft(<i>Line</i>, <i>Speed</i>)	Move the characters on the line specified by the byte value <i>Line</i> to the left cyclically. The leftmost characters will be displayed at the rightmost position in the next period. The speed of the movement is specified by the byte value <i>Speed</i> . A smaller <i>Speed</i> value represents a faster speed.
RotateOff()	Stop the automatic rotation movement to the left or to the right.

RotateRight (<i>Line, Speed</i>)	Move the characters on the line specified by the byte value <i>Line</i> to the right cyclically. The rightmost characters will be displayed at the leftmost position in the next period. The speed of the movement is specified by the byte value <i>Speed</i> . A smaller <i>Speed</i> value represents a faster speed.
SetBacklight (<i>Brightness</i>)	Set the backlight brightness with the byte value <i>Brightness</i> .
SetTab(<i>TabCount</i>)	Set the number of columns by the byte value <i>TabCount</i> for the cursor movement each time the Tab command is executed.

Note 1:

Refer to the following example for the display of LCD characters and user-defined characters:



Example Program:

```
Peripheral myLCD As LCD2x16A @ 0      'set module number 0

Sub Main()                            'main program
  myLCD.DisplayOn()                  'enable the display
  myLCD.SetBacklight(255)             'set LCD brightness to the maximum value
  myLCD.Backlighton(0)                'LCD backlight on constantly
  myLCD.Display("Hello World!")       'display "Hello World!" on screen
  Pause 3000
```

```
myLCD.RotateRight(1, 10)      'rotate "Hello World!" from left to right
Pause 5000
myLCD.RotateOff()            'stop rotating the message "Hello World!"
myLCD.Clear()                'clear all the characters on the display
```

'Set the number 0 user defined character to be composed of 4 horizontal lines
'on row 1, 3, 5 and 7. The number 31 is represented as 11111 in binary format,
'so the value of 31 means that all the pixels on the row will be turned on
'When this character is displayed, a pattern composed of four horizontal lines
'will be shown on the display.

```
myLCD.Customchar(0, 31, 0, 31, 0, 31, 0, 31,0)
```

'Shown as 8 repeated number 0 user defined characters on the display.

```
myLCD.DisplayChar(0, 0, 0, 0, 0, 0, 0, 0)
Pause 2000
```

'Set number 1 user defined character to be composed of 3 vertical lines on column
'1, 3 and 5. The number 21 is represented as 10101 in binary format, so the
'pixels on the 1, 3 and 5 columns will be turned on. When this character
' is displayed, a pattern composed of 3 vertical lines will be shown on the display.

```
myLCD.Customchar(1, 21, 21, 21, 21, 21, 21, 21,21)
```

































'Shown as 8 repeated number 0 user defined characters on the display

```
myLCD.DisplayChar(1, 1, 1, 1, 1, 1, 1, 1)
```

End Sub

Appendix

1. Module ID Setting Table:

	0		8		16		24
	1		9		17		25
	2		10		18		26
	3		11		19		27
	4		12		20		28
	5		13		21		29
	6		14		22		30
	7		15		23		31

2. Table of ASCII codes:

- American Standard Code for Information Interchange (ASCII) is a computer coding system based on Latin letters. The ASCII codes used here are a slightly modified version of the standard codes. The numbers inputted by the user will be converted into the corresponding character.
- The left column represents the lower four bits in binary format, and the upper column represents the higher four bits in binary format. In the Table, “L” represents 0 and “H” represents 1, so LLLL means 0000 in binary format, i.e., 0 in decimal format.
- From the upper left corner, e.g., the output character corresponds to the input value 0 (CG RAM1 means display the number 1 user defined character set by the user), to the bottom in the increasing order, e.g., the character “±” corresponds to the input value 16, In such a way, the lower right character corresponds to the input value of 255.

Upper 4 bit Lower 4 bit	LLLL	LLH	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH	HHHL	HHHH				
LLLL	CG RAM (1)	±		00P	'	P	G	E	A											
LLH	CG RAM (2)	≡	!	1	A	Q	a	9	U	a	i					J	T	Y	U	
LLHL	CG RAM (3)	7	"	2	B	R	b	r	e	b	e	°				o	o	o	o	
LLHH	CG RAM (4)	2	#	3	C	S	c	s	a	a	c	c				P	W	e	q	
LHLL	CG RAM (5)	1	\$	4	D	T	d	t	a	a	t	t				e	r	z	o	
LHLH	CG RAM (6)	1	%	5	E	U	e	u	a	a	e	e				'	s	t	a	n
LHHL	CG RAM (7)	1	&	6	F	V	f	v	a	a	v	v				u	j	e	a	
LHHH	CG RAM (8)	1	'	7	G	W	w	g	a	a	g	g				x	→	^	u	
HLLL	CG RAM (1)	1	(8	H	X	h	x	e	e	x	x				÷	←	E	K	
HLLH	CG RAM (2)	1)	9	I	Y	i	y	e	e	i	i				Δ	Γ	Π	λ	
HLHL	CG RAM (3)	*	*	#	J	Z	j	z	e	e	z	z				7	Σ	μ	¶	
HLHH	CG RAM (4)	1	+	:	K	C	k	c	i	a	a	i				*	L	Γ	v	
HHLL	CG RAM (5)	≡	,	<	L	\	l	l	a	n	a	n				*	1	κ	ξ	
HHLH	CG RAM (6)	ω	—	—	M	M	m	m	i	a	a	i				*	φ	π	—	
HHHL	CG RAM (7)	±	.	>	N	^	n	^	a	a	n	n				0	Ω	ρ	β	
HHHH	CG RAM (8)	±	/	?	O	_	o	_	a	a	o	o				—	0	o	o	