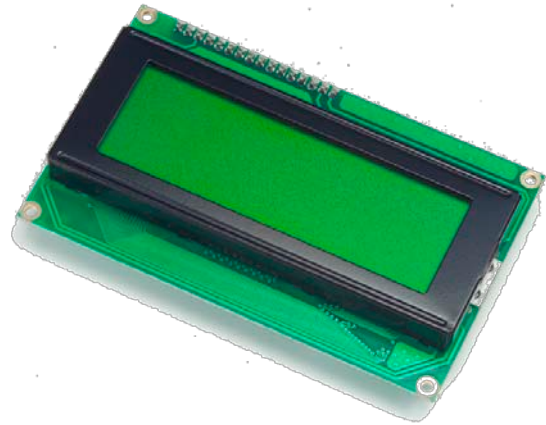


LCD 4x20A

User's Guide

4x20 Characters LCD Display Module

Version: V1.0



Product Overview: Innovati's LCD 4x20 A Module provides versatile display functions. Through the simple connection, it can be directly controlled by Innovati's Basic Commander for various applications. In this module, 4 message lines with 20 characters on each line can be displayed. By using the cursor control command, the characters displayed at any position on the screen can be arbitrarily changed. This module has a backlight function. Turning on the backlight allows the message to be read easily. In addition, it can be configured to use the user-defined characters to display the special characters required by the user.

Application Scope:

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

Product Features:

- It can display 4 lines of 20 characters at the same time.
- Each character has the resolution of 5x8 dots.
- The user can enter the ASCII codes to display the corresponding characters.
- The display related commands can be directly used, and the module will automatically convert the data into the corresponding characters or numeric values according to the entered strings or constants and show the result on the display.
- With proper setting, the backlight can provide 255 steps of brightness control for the display.
- For continuous input, the module will automatically change the cursor positions for displaying messages between the lines and automatically override the originally displayed messages.
- For the cursor movement, the cursor position can be directly assigned for arbitrarily jumping between rows or columns. The Tab value can also be configured to perform the automatic cursor movement backwards by the number of characters, thus it is very easy to meet the requirements for display control. When the cursor position is uncertain, the user can input the Home command to allow the cursor to return to the starting point of the screen.
- Versatile clear screen commands are provided to clear the full screen, clear a single character before the cursor, clear the characters after the cursor to the end of the line, or clear the remaining characters after the cursor position to the end of the screen.
- It allows the user to set the user defined characters for displaying various creative characters.

- While the display is not in use, the command for disabling the display function can be executed for saving the power consumption.

Connection: Put the ID switch to the required number directly, and then connect the cmdBUS to the corresponding pins on the BASIC Commander (Fig. 1) so that the user can perform the required operations through the BASIC Commander. Please connect the Vin and GND to the terminals of a 6-12V battery and the ground.

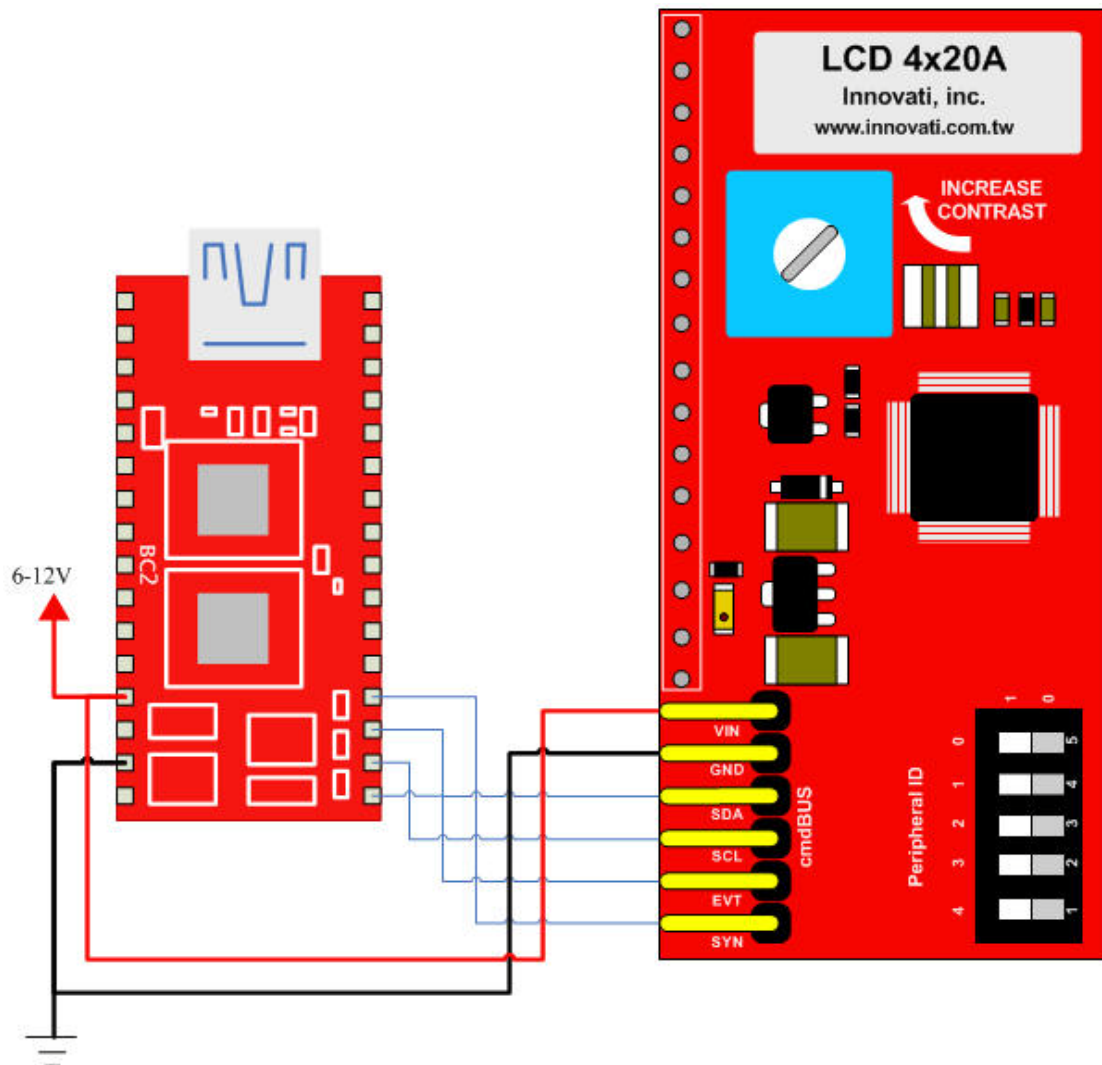


Figure 1 Connect LCD 4x20 A with the BASIC Commander

Product Specifications:

Pins for cmdBUS: Connect these pins to the corresponding pins on the BASIC Commander for controlling the LCD module through the BASIC Commander. While connecting, please notice the pin assignment. Connect Vin to the Vin on the BASIC Commander. Incorrect pin connection may cause damage to the module.)

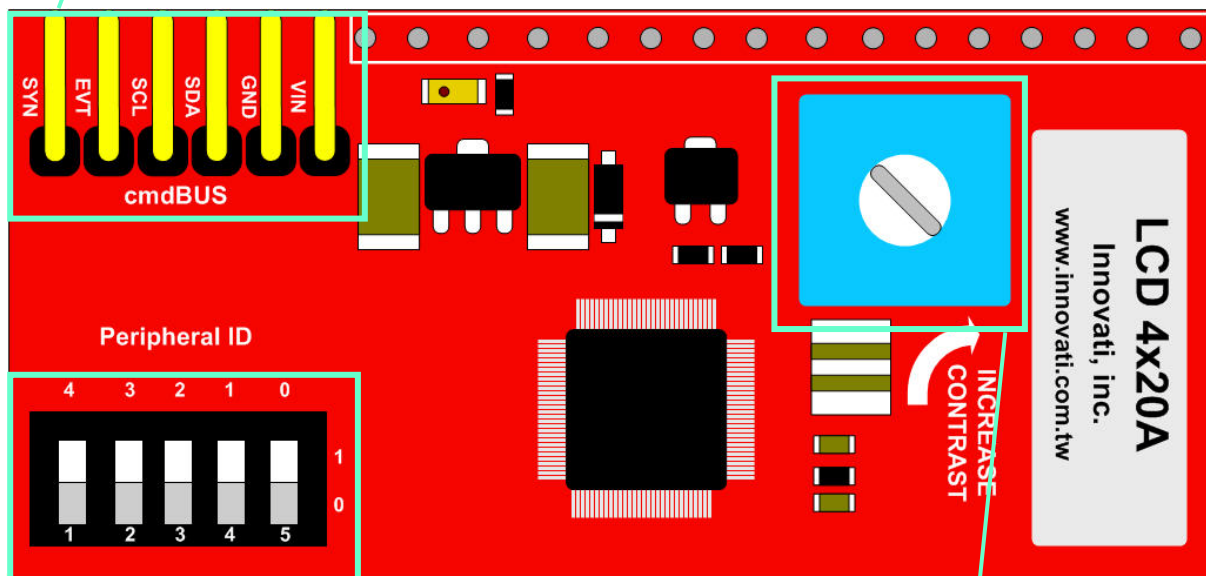


Figure 2 Description of pins and switches on the module

Module ID Setting Switch: The module ID of the LCD module can be configured with the binary digits from the right to the left. This ID number allows the BASIC Commander to determine the module to be controlled during the operation.(Please refer to Appendix 2)

Contrast adjustment screw. Please rotate it with the Phillips screwdriver. By rotating clockwise, the contrast can be increased. By rotating counterclockwise, the contrast can be decreased. There is a limit for the adjustment. Please do not rotate too much to avoid damage to the component.

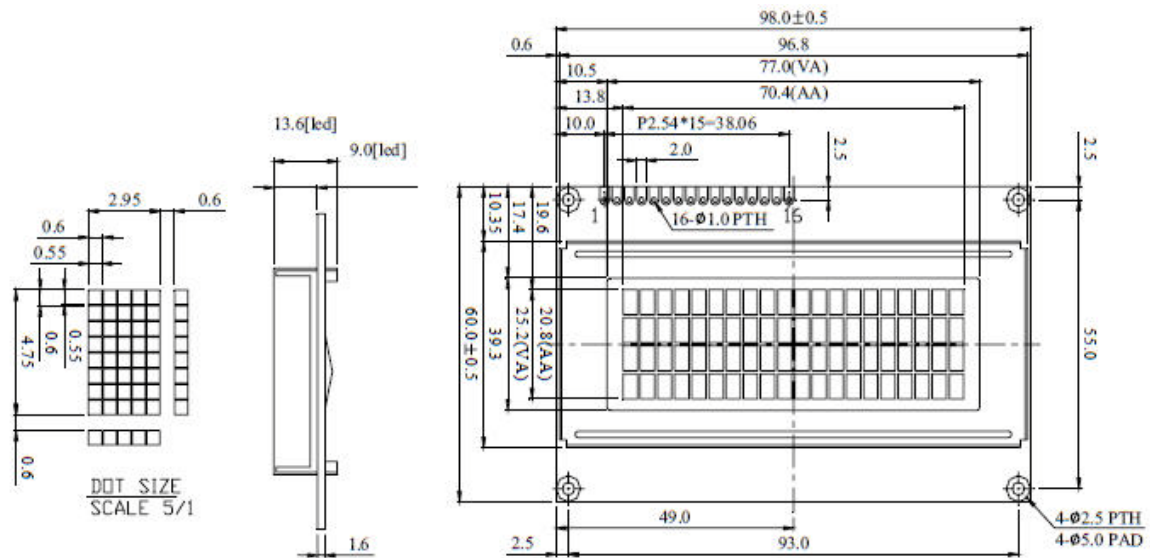


Figure 3 Dimensions of the LCD panel (in mm)

Item	Standard Value	Unit
Display type	20 characters x 4 Lines	—
Module dimension (L x W x H)	98.0 x 60.0 x 13.1 (Max) - LED array B/L STN Positive / 6 o'clock / Transflective	mm
Viewing Area	77.0 x 25.2	mm
Active Area	70.4 x 20.8	mm
Dot Size	0.55 x 0.55	mm
Dot Pitch	0.60 x 0.60	mm
Character size (L x W)	2.95 x 4.75	mm
Character pitch (L x W)	3.55 x 5.35	mm

Table 1 Specifications of the LCD Panel

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	—		110	170	ms
	T fall	—		150	200	ms

Table 2 Viewing angle and contrast ratio of the LCD Panel

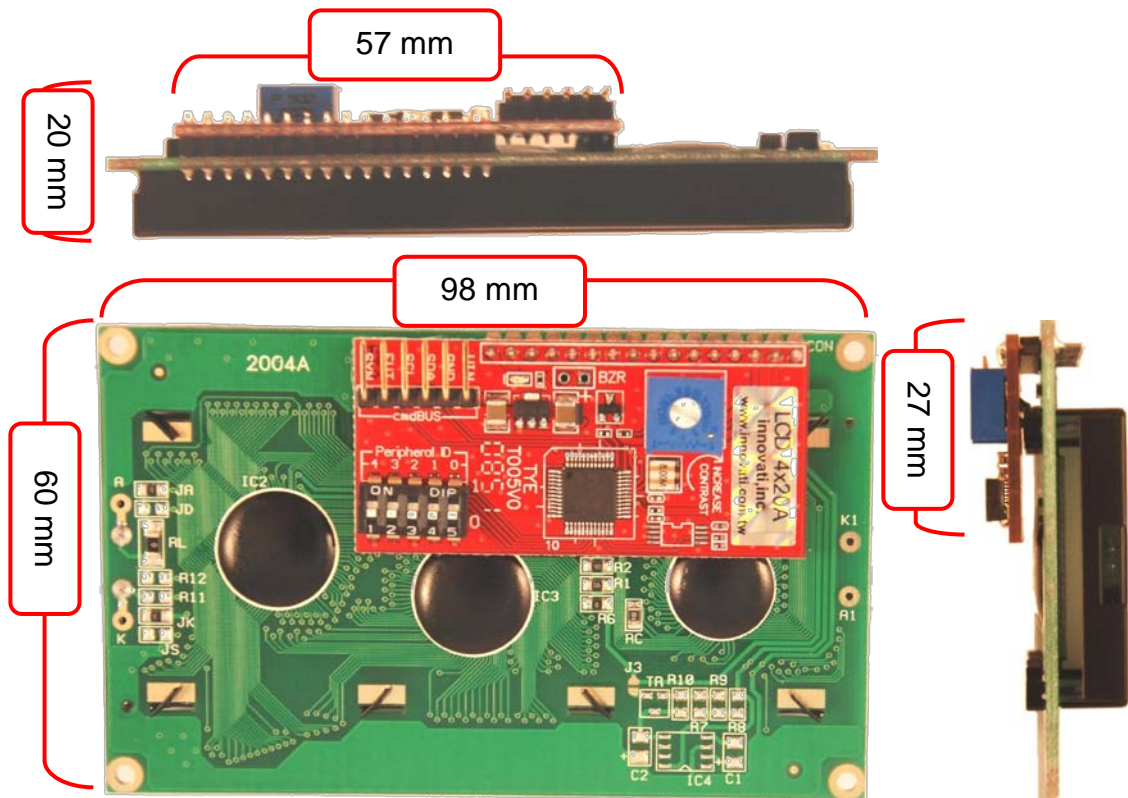


Figure 4 Dimensions of the Module

Precautions for Operations:

Operating Temperature 0 °C ~ 50 °C (>180 hr)
 Storage Temperature -20 °C ~ 70 °C (>180 hr)

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V _{DD}	Conditions				
I _{DD}	Operating Current	7.5	Backlight On	—	185	—	mA
			Backlight Off	—	5	—	mA

Table 3 Characteristics of the operating current (room temperature at 25 °C)

Commands And Events:

The following list shows the commands dedicated for controlling the LCD 4x20 A module. The command names and parameters which should be input are shown in bold or bold-italic typefaces. The words in bold typeface should not be changed while being input. The words in bold-italic typefaces can be filled with parameters in properly defined format by the user. Please note that the words in uppercase or lowercase are regarded as the same word while entering the command in the innoBASIC Workshop.

Before executing the command for LCD 4x20 A, please define the corresponding parameters and the module ID at the beginning of the program, for example:

Peripheral *ModuleName* As LCD2x16A @ *ModuleID*

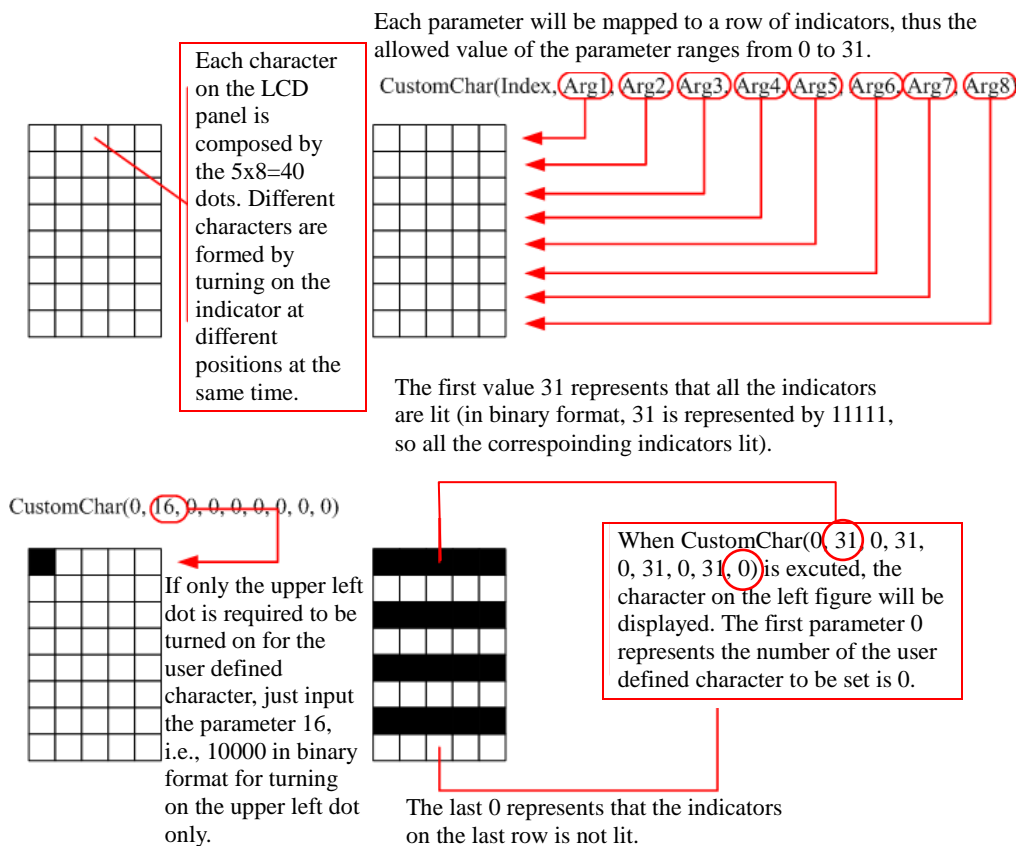
Command Format	Function of the Command
Commands for the Cursor Movements	
CarriageReturn() CR()	Move the cursor to the next row for carriage return
CursorColumn(<i>Col</i>) CursorCol(<i>Col</i>)	Move the cursor to the column specified by <i>Col</i> . The input value of <i>Col</i> should be an integer within the range of 1-20.
CursorDown()	Move the cursor down to the next line.
CursorLeft()	Move the cursor to the left by one character.
CursorRC(<i>Row</i>, <i>Col</i>)	Move the cursor to the row and the column specified by <i>Row</i> and <i>Col</i> . The input value of <i>Row</i> should be an integer within the range of 1-4. The input value of <i>Col</i> should be an integer within the range of 1-20.
CursorRight()	Move the cursor to the right by one character.
CursorRow(<i>Row</i>)	Move the cursor to the row specified by <i>Row</i> . The input value of <i>Row</i> should be an integer within the range of 1-4.
CursorUp()	Move the cursor up to the previous line.
Home()	Move the cursor to the first column in the first row.
Tab()	Move the cursor to the right by the number of characters specified by <i>Tab</i> .
Commands for Clearing the Display	
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.
Commands for Displaying Characters	
Display(<i>Parameter</i>)	The message will be displayed according to the type of <i>Parameter</i> . If the type is String , the string will be displayed directly; if it is a floating-point number, it will be displayed in the scientific notation; other numeric values will be displayed in the decimal format.
DisplayBin(<i>Value</i>)	The input parameter <i>Value</i> will be displayed in binary format. The parameter <i>Value</i> should be an integer.
DisplayChar(<i>Chr</i>, ...)	Display the characters specified by <i>Chr</i> . The parameter <i>Chr</i> should be an integer within the range of 0-255. The values 0-7 can also be input to specify the corresponding user defined characters. It is allowed to input multiple characters and parameters. The input values will be displayed as characters according to their ASCII codes. Please refer to Appendix 3.

DisplayFloat(<i>FloatNum</i>, <i>Digits</i>)	Set the number of effective digits specified by <i>Digits</i> and display the value of <i>FloatNum</i> in the scientific notation. <i>Digits</i> should be an integer within 0-255. <i>FloatNum</i> should be a floating-point value.
DisplayHex(<i>Value</i>)	The input parameter <i>Value</i> will be displayed in hexadecimal format. The parameter <i>Value</i> should be an integer.
DisplayLeft(<i>Value</i>, <i>Num</i>)	According to the width specified by <i>Num</i> , which should be an integer within 0-255, display the input parameter <i>Value</i> in decimal format aligned to the left. If the length of the input parameter exceeds the width, it will be automatically adjusted to a proper width. <i>Value</i> should be a non-string number.
DisplayReal(<i>Real</i>, <i>Digits</i>)	According to the number of effective digits specified by <i>Digits</i> , display the number <i>Real</i> in real format. <i>Real</i> should be a floating-point number. <i>Digits</i> should be an integer within 0-255.
DisplayRight (<i>Value</i>, <i>Num</i>)	According to the width specified by <i>Num</i> which should be an integer within 0-255, display the input parameter <i>Value</i> in decimal format aligned to the right. If the length of the input parameter exceeds the width, it will be automatically adjusted to a proper width. <i>Value</i> should be a non-string number.
Commands for Various Settings	
BacklightOff()	Turn off the backlight.
BacklightOn(<i>Time</i>)	Set the value of <i>Time</i> to specify the time interval to turn on the backlight. If it is 0, the backlight will be constantly turned on. <i>Time</i> should be an integer within 0-255.
CursorBlinkOff()	Disable the cursor blinking.
CursorBlinkOn()	Enable 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 number of the character to be defined by the value of <i>Index</i> which should be an integer within the range of 0-7. The values <i>Arg1-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*1.
DisplayOff()	Turn off the display.
DisplayOn()	Turn on the display.
GetTab(<i>TabCount</i>)	Get the preset <i>Tab</i> value and store it in <i>TabCount</i> . The returned <i>TabCount</i> will be an integer within the range of 0-255.
RotateLeft(<i>Line</i>, <i>Spd</i>)	Move the characters on the line specified by <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 <i>Spd</i> . A smaller value of <i>Spd</i> indicates a faster speed. <i>Line</i> should be an integer within 1-4. <i>Spd</i> should be an integer within 0-255.
RotateOff()	Disable the automatic rotation movement to the left or to the right.
RotateRight (<i>Line</i>, <i>Spd</i>)	Move the characters on the line specified by <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 Spd . A smaller value of Spd indicates a faster speed. Line should be an integer within 1-4. Spd should be an integer within 0-255.
SetBacklight (Arg)	Set the backlight brightness with the value of Arg which should be an integer within 0-255.
SetTab(TabCount)	Set the number of columns by the value of TabCount for the cursor movement each time Tab command is executed. TabCount should be an integer within 0-255.

Table 1 :Command Table

*1 Refer to the following example for the display of the native characters and user defined characters on the LCD:



Demonstration Program:

```
Peripheral myLCD As LCD4x20A @ 0 ' Set the module ID as 0

































Sub Main() ' Main program
    myLCD.DisplayOn() ' Turn on the display.
    myLCD.SetBacklight(255) ' Set LCD brightness to the maximum
                             value
    myLCD.Backlighton(0) ' Set the LCD backlight to be turned on
                           constantly
    myLCD.Display("Hello Word!") ' Show the message "Hello Word!" on the
                                   display
    Pause 3000
    myLCD.RotateRight(1, 10) ' Rotate the message "Hello Word!" from
                               the left to the right
    Pause 5000
    myLCD.RotateOff() ' Stop rotating the message "Hello Word!"
    myLCD.Clear() ' Clear all the characters on the display

' Set the No. 0 user defined character to be composed of 4 horizontal lines on the
' 1st, 3rd, 5th, and 7th rows
' The number 31 is represented as 11111 in binary format, so the value of 31
' represents all the indicators 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)
' Show 8 repeated No. 0 user defined characters on the display
' myLCD.DisplayChar(0, 0, 0, 0, 0, 0, 0, 0)
' Pause 2000
' Set No. 1 user defined character to be composed of 3 vertical lines on the 1st, 3rd
' and 5th columns
' The number 21 is represented as 10101 in binary format, so the indicators on the
' 1st, 3rd and 5th columns will be lit.
' When this character is displayed, a pattern composed of three vertical lines will
' be shown on the display
' myLCD.Customchar(1, 21, 21, 21, 21, 21, 21, 21,21)
' Show 8 repeated No. 1 user defined characters on the display
' myLCD.DisplayChar(1, 1, 1, 1, 1, 1, 1, 1)
End Sub
```

Appendix

1. Known Problem:

2. List of the Configuration of the Module ID Switch:

	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

3. Table of ASCII Codes:

- American Standard Code for Information Interchange (ASCII) is a computer coding system based on the Latin letters. The ASCII codes used here are a slightly modified version of the standard codes. The number input by the user will be converted into the corresponding character.
- The left column represents the lower four bits in the binary format, and the upper column represents the higher four bits in the 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.
- The upper left corner in the table represents the output character pattern corresponding to the input ASCII value 0. For example: CG RAM1 means to display the No. 1 user defined character set by the user. Moving down along the table, the number increases. Moving to the right of the table by one column, the ASCII code is increased by 16, and so on. The character at the bottom right corner is the pattern shown on the display with the input value of 255.

Upper 4 bit Lower 4 bit	LLLL	LLLH	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH	HHHL	HHHH
CG RAM (1)																
LLLL (0)																
LLLH (1)																
LLHL (2)																
LLHH (3)																
LHLL (4)																
LHLH (5)																
LHHL (6)																
LHHH (7)																
HLLL (8)																
HLLH (9)																
HLHL (10)																
HLHH (11)																
HHLL (12)																
HHLH (13)																
HHHL (14)																
HHHH (15)																

The pattern shown on the display with the input ASCII code 0.

The pattern shown on the display with the input ASCII code 33.

The pattern shown on the display with the input ASCII code 255.

The pattern shown on the display with the input ASCII code 47.

The pattern shown on the display with the input ASCII code 15.