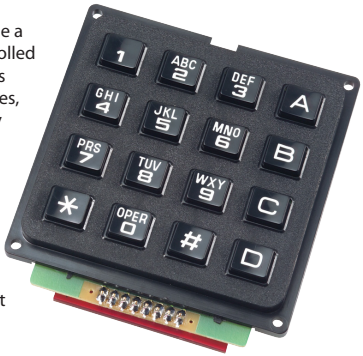
Keypad A
User's Guide

Version: 1.0

Innovati's 16-key Keypad A Module is designed to provide a range of versatile input functions. It can be directly controlled by Innovati's BASIC Commander® for various applications using simple connections. By setting different input modes, it can be converted rapidly to emulate various commonly used input interfaces, such as the common numeric inputs for calculators, English character inputs for mobile phones, hexadecimal inputs for engineering, or even a user-defined returned value for each key. In addition to the input options, the debounce time of the keys can be setup to avoid problems due to mechanical key bounce when the keys are pressed. The automatic repeat (auto-repeat) input can be set to generate a repeat input when a key is pressed and held down. Please use "KeypadA" as the module object name in program.



Trademark

Disclaimer

Full responsibility for any applications using Innovati products rests firmly with the user and as such Innovati will not be held responsible for any damages that may occur when using Innovati products. This includes damage to equipment or property, personal damage to life or health, damage caused by loss of profits, goodwill or otherwise. Innovati products should not be used for any life saving applications as Innovati's products are designed for experimental or prototyping purposes only. Innovati is not responsible for any safety, communication or other related regulations. It is advised that children under the age of 14 should only conduct experiments under parental or adult supervision.

Errata

We hope that our users will find this user's guide a useful, easy to use and interesting publication, as our efforts to do this have been considerable. Additionally, a substantial amount of effort has been put into this user's guide to ensure accuracy and complete and error free content, however it is almost inevitable that certain errors may have remained undetected. As Innovati will continue to improve the accuracy of its user's guide, any detected errors will be published on its website. If you find any errors in the user's guide please contact us via email service@innovati.com.tw. For the most up-to-date information, please visit our web site at http://www.innovati.com.tw.

Table Of Content

Product Overview

Innovati's 16-key Keypad A Module is designed to provide a range of versatile input functions. It can be directly controlled by Innovati's BASIC Commander® for various applications using simple connections. By setting different input modes, it can be converted rapidly to emulate various commonly used input interfaces, such as the common numeric inputs for calculators, English character inputs for mobile phones, hexadecimal inputs for engineering, or even a user-defined returned value for each key. In addition to the input options, the debounce time of the keys can be setup to avoid problems due to mechanical key bounce when the keys are pressed. The automatic repeat (auto-repeat) input can be set to generate a repeat input when a key is pressed and held down. Please use "KeypadA" as the module object name in program.

Application

- When used with an LCD display, it can rapidly emulate the function of a calculator by setting the correct mode.
- Can be used for password input complete with case-sensitive capability.
- The user defined input function allows the user to set and detect different keys through the software to setup the module for versatile operations.
- By setting the hold mode, the keypad can be used as 8-direction keys for a wired remote control.

Product Features

- 4x4 input keypad. Can be operated in 9 different input modes.
  1. Key value mode - default
  2. Hexadecimal mode
  3. Numeric mode
  4. Upper case character input
  5. Lower case character input
  6. Symbol mode
  7. Calculator mode
  8. User defined mode
  9. Extended keypad input mode
- The user can set the debounce value according to their personal preference to avoid repeat inputs.
- By using the extension pins, the product can be expanded with additional extended keypads.
- Determining the pressed keys can be performed either in the event mode or in the polling mode though different program designs.
- Pressing and holding the keys can implement a repetition key input.

Connection

Directly setup the ID switches to the required number, and then connect the cmd-BUS™ cable to the corresponding pins on the BASIC Commander® as 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. If it is required to expand the system with additional keypads, it can be expanded with up to 64 additional keys just by connecting it to the 8x8 Extended Keypad pins.

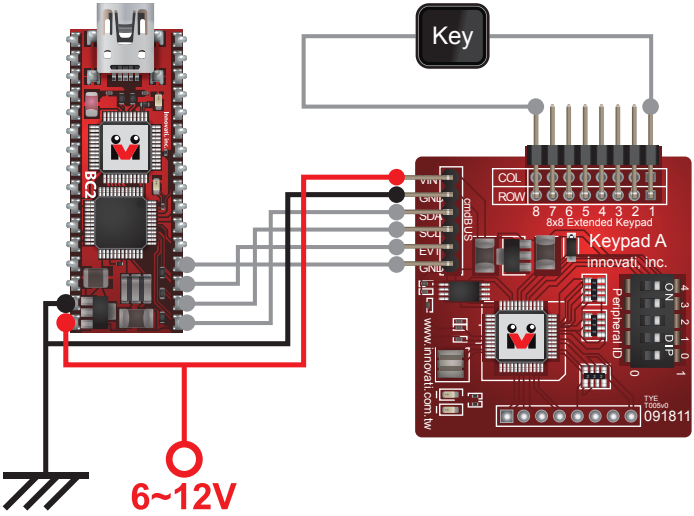


Figure 1: The BASIC Commander® and the Keypad A use one power supply.

Product Specifications

Extension pins for connecting the extended keypad. It can be extended with up to 64 extended keys for key detection without the need of additional circuit designs - just connect them with the keypad. (For the returned key values of each key, please refer to Appendix 3.)

cmdBUS™ pins. Connect these pins to the corresponding pins of the BASIC Commander® to allow the user to use the BASIC Commander® to control the Keypad module. (When connecting, please pay attention to the pin assignment. Connect Vin to the Vin on the BASIC Commander®. Incorrect pin connection may cause damage to the module).

The module number setting switches. Set the module number of the Keypad module in a 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 operation (please refer to Appendix 2).

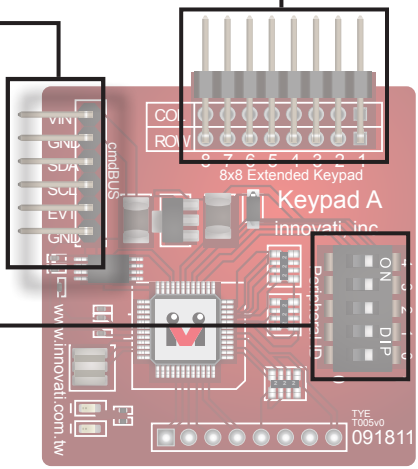


Figure 2: Pin assignment and module switches

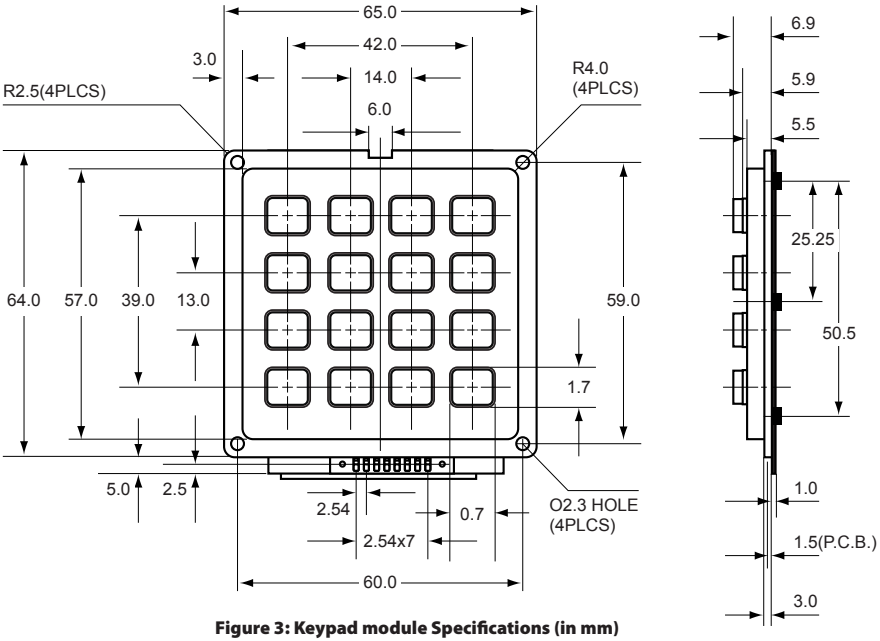


Figure 3: Keypad module Specifications (in mm)

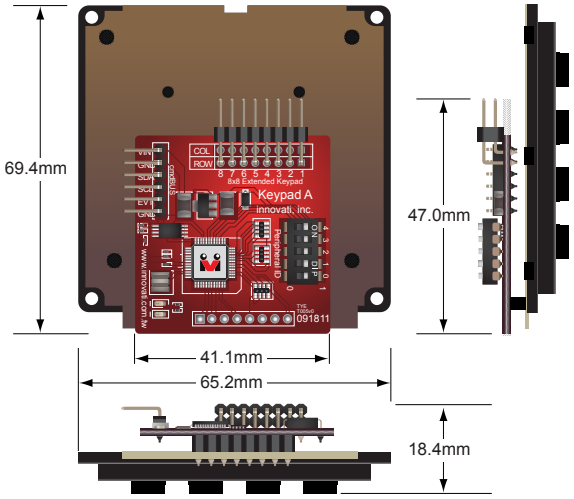


Figure 4: Keypad module Overview

Precautions For Operations

Symbol	Parameter	Test Conditions (V_{IN}=7.5V)		Min	Typ	Max	Unit
I_{IN}	Operating Current	7.5	—	—	6.5	—	mA

Table 1: Operating current characteristics (ambient temperature 25 ℃)

Absolute Maximum Ratings:

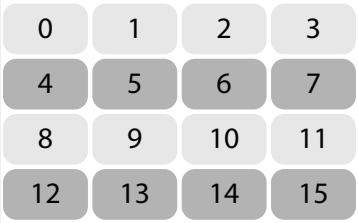
Operating Temperature of the Module: 0 ℃ ~ 70 ℃

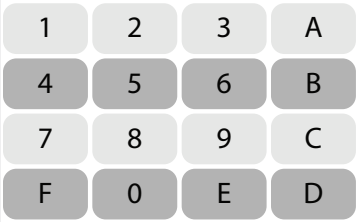
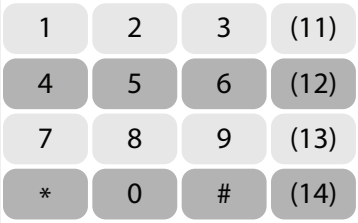
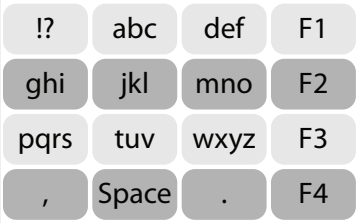
Storage Temperature of the Module: -20 ℃ ~ 80 ℃

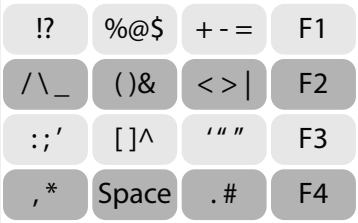
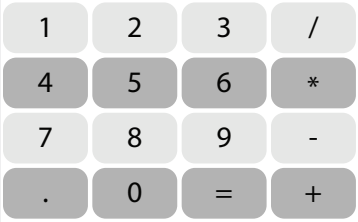
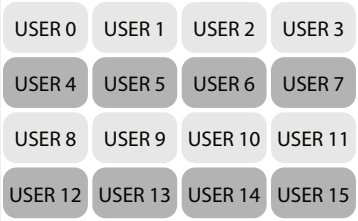
Commands and Events

The following tables list all the unique commands and events provided with the Keypad A 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
ClearKeyBuffer()	Clear all the key press values in the buffer.
DisableBufferFullEvent()	Disable the key buffer full event.
DisableKeypressedEvent()	Disable the key pressed event.
EnableBufferFullEvent()	Enable the key buffer full event.
EnableKeypressedEvent()	Enable the key pressed event
GetCustomTable(*CustomTable*)	Get the custom key values and store them in the array ***CustomTable***, which should be able to store 16 bytes.
GetCustomTableIndex(*Index*)	Get the number of the currently used ***CustomTable*** and store it in the byte variable ***Index***. (See Note 1.)

Command Format	Description
GetDebounceTime(*Time*)	Get the debounce time setting and store it in the byte variable ***Time***.
Status = GetKeyID(*KeyID*)	Return the keypressed status in the byte variable ***Status*** and store the key press value and store it in the byte variable ***KeyID***. If the keypad is not pressed, value 0 will be returned, otherwise value 1 will be returned. When keypad is working in mode 3, 4 or 5 for multiple-character input, if the returned value is 1, it indicates the character stored in the ***KeyID*** is the final confirmed character. If the returned value is 2, it indicates the character stored in the ***KeyID*** is a transient character. (See Note 1.)
GetKeypadMode(*Mode*)	Get the current keypad working mode and store it in the byte variable ***Mode***.
GetRepeatRate(*Rate*)	Get the current repeat rate setting and store it in the byte variable ***Rate***.
GetRepeatTime(*Time*)	Get the current repeat time setting and store it in the byte variable ***Time***.
LoadCustomTable(*Index*)	According to the value of the byte variable ***Index***, load the user-defined custom table stored in the EEPROM.
SaveCustomTable(*Index*)	Store the currently user-defined custom table to the EEPROM at the address specified by the byte variable ***Index*** ranging from 0 to 15.
SetCustomTable(*KeyID*)	Set the elements of array ***KeyID*** as the user defined key press values mapped to key 1, 2, 3, A, 4, 5, 6, B, 7, 8, 9, C, *, 0, # and D. A total of 16 user-defined key press values should be given in the ***KeyID*** array.
SetDebounceTime(*Time*)	Set the debounce time specified by the byte variable ***Time***, the units are 10 ms. The setting command is effective only for externally connected keypads; it will not affect the operation of the built-in keypad.
SetKeypadMode(*Mode*)	Set the keypad mode specified by the byte variable ***Mode***, ranging from 0 to 8 representing one of the following keypad operation modes, which returns the key value as shown below, respectively. (See Note 2.) 0: Key value mode (After the key is pressed, it returns the key press value of the ***Key ID*** value in order.) 

Command Format	Description
	1: Hexadecimal mode (After the key is pressed, it returns the key press value of 0~F.)  2: Numeric mode (After the key is pressed, it returns the key press value of the ASCII codes of 0~9.)  3: Upper case character input  4: Lower case character input 

Command Format	Description
	5: Symbol mode  6: Calculator mode  7: User defined mode  8: Extended keypad input mode (For the corresponding ***key ID*** values of the extended keypads, please refer to Appendix 3.) 

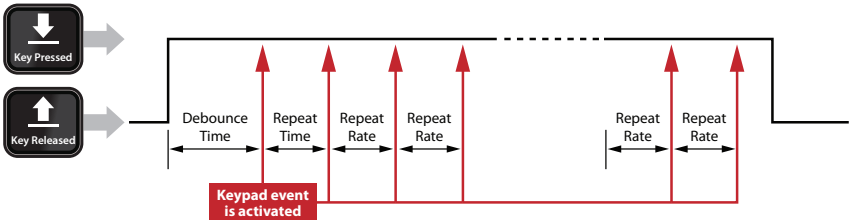
Command Format	Description
SetRepeatRate(*Rate*)	Set the repeat rate specified by the byte variable ***Rate*** for determining the repeated triggering rate. The unit is 10 ms. (See Note 3.)
SetRepeatTime(*Time*)	Set the repeat time specified by the byte variable ***Time*** for determining the time before the repeated triggering starts. The unit is 10 ms. (See Note 3.)

Table 2: Command Table

Note 1: If the "Repeat" related settings are not configured, after the key is pressed and GetKeyID is executed, the status will be reset to 0, even the key is still held down. If there are several keys being pressed continuously and GetKeyID is executed, all the key press values will be stored. When the number of key presses exceeds the capacity of the buffer (32 key press values), the follow-up key presses will be ignored.

Note 2: When keypad is working in mode 3, 4 or 5 for multiple-character input, if the returned value is 1, it indicates the character stored in the module is the final confirmed character. If the returned value is 2, it indicates the character stored in the module is a transient character.

Note 3: The effects of setting the Repeat Time and the Repeat Rate are as follows.



After the key is pressed and held for a time interval specified by the Repeat Time, another keypressed event will be activated. After this, the same event will be activated repeatedly for every subsequent time interval specified by the Repeat Rate until the key is released.

Event Name	Description
KeyBufferFullEvent()	After EnableBufferFullEvent() is executed, if the key pressed exceeds 32, the capacity of the buffer, the KeyBufferFullEvent() will be triggered.
KeyPressedEvent()	After EnableKeyPressedEvent() is executed, if any key is pressed, the KeyPressedEvent() will be triggered.

Table 3: Event Provided By The Module

Example Program

```
Peripheral myKeypad As KeypadA @ 0    'Set module number to 0

Dim PressKeyD As Byte       'Store the event whether D is pressed
Dim KeyStatus As Byte       'Determine whether a key press value is obtained
Dim KeyID As Byte           'Store the obtained key press value
Dim RepeatTime As Byte      'Store the obtained value of repetition time
Dim RepeatRate As Byte      'Store the obtained value of repetition rate
Dim DebounceTime As Byte    'Store the obtained value of the debounce time
Dim RepeatCount As Byte     'Store the number of repeated key presses.
Dim CustomTable(15) As Byte 'Array for storing the user defined key press values
Dim i As Byte               'Store the loop counts
Sub Main()
Debug CLS

KEYID_CHECK:
'The following loop will be repeatedly executed. According to the pressed key
'determined by KeyStatus,
'the input key press messages will be shown in the Terminal Window.
'If the pressed key is determined to be "D", the loop will be exited.
'This part is used for determining the key press value by using Event.

myKeypad.SetKeypadmode(0)   'Set the keypad mode to 0, the default mode
   Pause 100
   myKeypad.GetKeyID(KeyID)                    'Get the initial key status
   RepeatCount=0
   KeyStatus=0
   myKeypad.EnableKeyPressedEvent()            'Enable the keypad event
   Debug "Press D to exit the loop.", CR
   PressKeyD=0
Do
   Loop Until PressKeyD>0

REPEAT_CHECK:
myKeypad.GetRepeatTime(RepeatTime)
'Get the initial RepeatTime set in the system
myKeypad.GetRepeatRate(RepeatRate)
'Get the initial RepeatRate set in the system
'Display the initial Repeat Time set in the system
```

```
Debug "Repeat Time is currently set as ", RepeatTime, " * 10 ms...", CR
'Display the initial Repeat Rate set in the system
Debug "Repeat Rate is currently set as ", RepeatRate, " * 10 ms...", CR

myKeypad.SetRepeatTime(50)            'Set the RepeatTime as 500 ms
myKeypad.SetRepeatRate(2)             'Set the RepeatRate as 20 ms
RepeatCount=1

'The following loop will be executed repeatedly. According to the pressed key
'determined by KeyStatus.
'If the key remains pressed and held for over half a second, the repeated keypad
'events will be activated according to the RepeatTime setting.
'The keypad event will be activated repeatedly every 20 ms according to the
'RepeatRate setting.

Debug "Please press and hold any key", CR
Do
   Loop Until RepeatCount>100
   myKeypad.DisableKeyPressedEvent()  'Disable the keypad event

'Set RepeatTime and RepeatRate as 0 to disable the repeated keypad event while
'pressing and holding the key

myKeypad.SetRepeatTime(0)
myKeypad.SetRepeatRate(0)
CUSTOM_TABLE:
For i=0 To 15
CustomTable(i)=100+i
Next
myKeypad.SetCustomTable(CustomTable)  'Set the user defined key press values as
                                      '100~115
myKeypad.SaveCustomTable(0)           'Store the user defined key press values in
                                      'Table 0
myKeypad.SetKeypadmode(7)             'Set the keypad mode to 7, the user defined
                                      'mode
Pause 100
myKeypad.GetKeyID(KeyID)
KeyStatus=0
'The following loop will be executed repeatedly. According to the pressed key
'determined by KeyStatus,
'The input key press messages will be shown in the Terminal Window.
'If the pressed key is determined to be "D", the loop will be exited.
'It can be observed that the returned Key ID will become the user defined key
'press value
'Here, the status and key value of the pressed key is obtained by polling.
```

```
'There is no keypad event.
Debug "Press D to exit the loop", CR
Do
   If myKeypad.GetKeyID(KeyID)<>0 Then
      Select Case KeyID
      Case 100 : Debug "Press the button 1! (The returned value is 100)", CR
      Case 101 : Debug "Press the button 2! (The returned value is 101)", CR
      Case 102 : Debug "Press the button 3! (The returned value is 102)", CR
      Case 103 : Debug "Press the button A! (The returned value is 103)", CR
      Case 104 : Debug "Press the button 4! (The returned value is 104)", CR
      Case 105 : Debug "Press the button 5! (The returned value is 105)", CR
      Case 106 : Debug "Press the button 6! (The returned value is 106)", CR
      Case 107 : Debug "Press the button B! (The returned value is 107)", CR
      Case 108 : Debug "Press the button 7! (The returned value is 108)", CR
      Case 109 : Debug "Press the button 8! (The returned value is 109)", CR
      Case 110 : Debug "Press the button 9! (The returned value is 110)", CR
      Case 111 : Debug "Press the button C! (The returned value is 111)", CR
      Case 112 : Debug "Press the button *! (The returned value is 112)", CR
      Case 113 : Debug "Press the button 0! (The returned value is 113)", CR
      Case 114 : Debug "Press the button #! (The returned value is 114)", CR
      Case 115 : Debug "Press the button D! (The returned value is 115)", CR
      End Select
      If KeyID=115 Then
        Goto KEYID_CHECK
      End If
   End If
   Loop
End Sub
Event myKeypad.KeyPressedEvent()
KeyStatus=myKeypad.GetKeyID(KeyID)    'Store the obtained key press value in the
                                      'parameter KeyID
If RepeatCount>100 Then
Return
Elseif RepeatCount>0 Then
RepeatCount+=1
Debug "Count the number of key presses ", RepeatCount, CR
Elseif RepeatCount=0 Then
Select Case KeyID
Case 0: Debug "Press the button 1!", CR
Case 1: Debug "Press the button 2!", CR
Case 2: Debug "Press the button 3!", CR
Case 3: Debug "Press the button A!", CR
Case 4: Debug "Press the button 4!", CR
Case 5: Debug "Press the button 5!", CR
Case 6: Debug "Press the button 6!", CR
```

```
Case 7: Debug “Press the button B!”, CR
Case 8: Debug “Press the button 7!”, CR
Case 9: Debug “Press the button 8!”, CR
Case 10: Debug “Press the button 9!”, CR
Case 11: Debug “Press the button C!”, CR
Case 12: Debug “Press the button *!”, CR
Case 13: Debug “Press the button 0!”, CR
Case 14: Debug “Press the button #!”, CR
Case 15: Debug “Press the button D!”, CR : PressKeyD=1
End Select
End If
End Event
```

Appendix

Known problem:

While switching the mode, the determination of events will be reset. If the mode switching operation is performed while a key is being pressed and held, even if the "Repeat" related settings are disabled, a keypad event will still be activated.

Module numbers and switch tables:

DIP Switch	ID	DIP Switch	ID	DIP Switch	ID	DIP Switch	ID
	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

Table 4: Module ID Setting Table

Mode 8 extended keypad returned key press values:

	ROW 1	ROW 2	ROW 3	ROW 4	ROW 5	ROW 6	ROW 7	ROW 8
COL 1	0	1	2	3	4	5	6	7
COL 2	8	9	10	11	12	13	14	15
COL 3	16	17	18	19	20	21	22	23
COL 4	24	25	26	27	28	29	30	31
COL 5	32	33	34	35	36	37	38	39
COL 6	40	41	42	43	44	45	46	47
COL 7	48	49	50	51	52	53	54	55
COL 8	56	57	58	59	60	61	62	63

Table 5: Mode 8 Extended Keypad Table