4x4 Matrix Membrane Keypad

User Manual

How it Works

Matrix keypads use a combination of four rows and four columns to provide button states to the host device, typically a microcontroller. Underneath each key is a pushbutton, with one end connected to one row, and the other end connected to one column. These connections are shown in Figure 1.

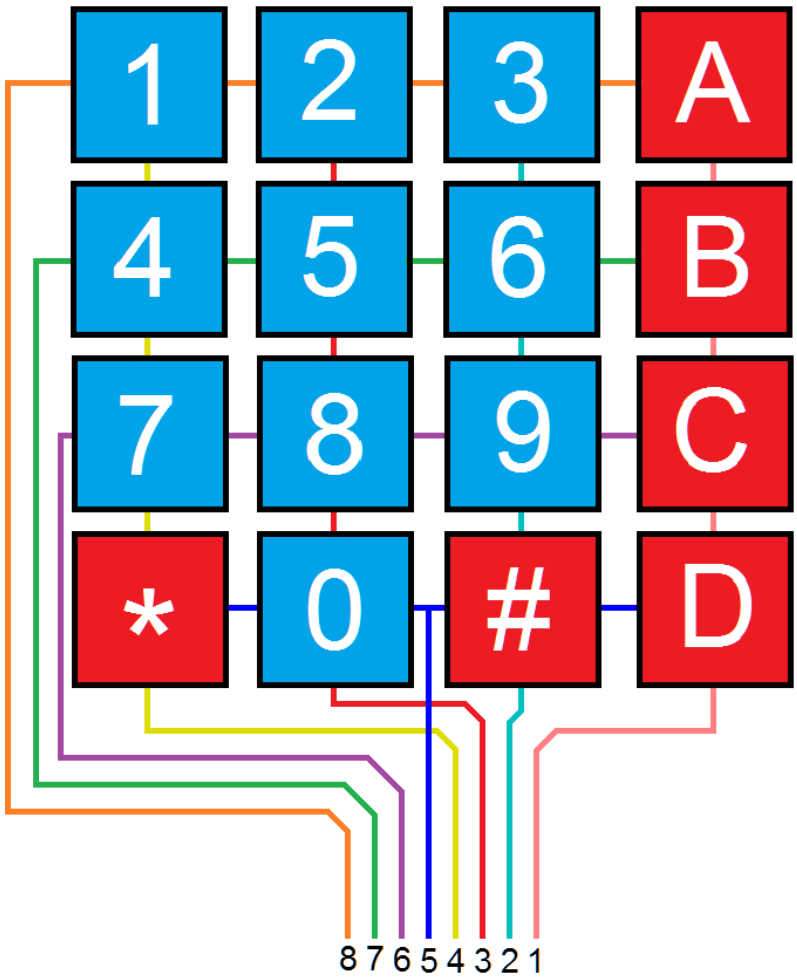


Figure 1: Matrix Keypad Connections

In order for the microcontroller to determine which button is pressed, it first needs to pull each of the four columns (pins 1-4) either low or high one at a time, and then poll the states of the four rows (pins 5-8). Depending on the states of the columns, the microcontroller can tell which button is pressed.

For example, say your program pulls all four columns low and then pulls the first row high. It then reads the input states of each column, and reads pin 1 high. This means that a contact has been made between column 4 and row 1, so button ‘A’ has been pressed.

Connection Diagrams

Figure 2

For use with the BASIC Stamp example program listed below.

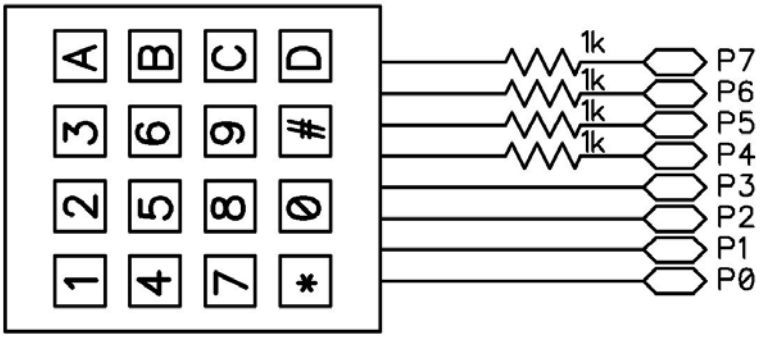
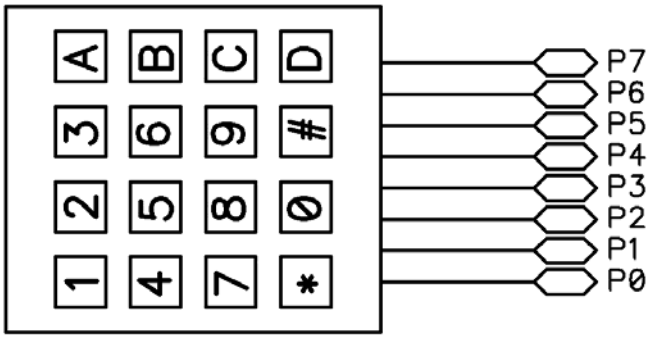


Figure 3

For use with the Propeller P8X32A example program listed below.



BASIC Stamp® Example Code

The example code below displays the button states of the 4x4 Matrix Membrane Keypad. It uses the Debug Terminal, which is built into the BASIC Stamp Editor software. The software is a free download from www.parallax.com/basicstampsoftware.

```
' 4x4MatrixKeypad_Demo.bs2
' Display buttons pressed on the 4x4 Matrix Membrane Keypad
' Author: Parallax HK Engineering

' {$STAMP BS2}
' {$PBASIC 2.5}

row         VAR  Nib                    ' Variable space for row counting
column      VAR  Nib                    ' Variable space for column counting
keypad      VAR  Word                   ' Variable space to store keypad output
keypadOld   VAR  Word                   ' Variable space to store old keypad output
temp        VAR  Nib                    ' Variable space for polling column states

DEBUG CLS                               ' Clear Debug Terminal
GOSUB Update                            ' Display keypad graphic

DO
  GOSUB ReadKeypad                      ' Read keypad button states
  DEBUG HOME, BIN16 keypad, CR, CR,     ' Display 16-bit keypad value
              BIN4 keypad >> 12,CR,     ' Display 1st row 4-bit keypad value
              BIN4 keypad >> 8, CR,     ' Display 2nd row 4-bit keypad value
              BIN4 keypad >> 4, CR,     ' Display 3rd row 4-bit keypad value
              BIN4 keypad               ' Display 4th row 4-bit keypad value
```

```
  IF keypad <> keypadOld THEN               ' If different button is pressed,
    GOSUB Update                            ' update the keypad graphic to clear
  ENDIF                                     ' old display

  IF keypad THEN                            ' Display button pressed in graphic
    GOSUB display
  ENDIF

  keypadOld = keypad                        ' Store keypad value in variable keypadOld
LOOP

' -----[ Subroutine - ReadKeypad ]-------------------------------------------------
' Read keypad button states
ReadKeypad:
  keypad = 0
  OUTL   = %00000000                        ' Initialize IO
  DIRL   = %00000000

  FOR row = 0 TO 3
    DIRB = %1111                            ' Set columns (P7-P4) as outputs
    OUTB = %0000                            ' Pull columns low (act as pull down)
    OUTA = 1 << row                         ' Set rows high one by one
    DIRA = 1 << row

    temp = 0                                ' Reset temp variable to 0
    FOR column = 0 TO 3
      INPUT (column + 4)                    ' Set columns as inputs
      temp = temp | (INB & (1 << column))   ' Poll column state and store in temp
    NEXT

    keypad = keypad << 4 | (Temp REV 4)     ' Store keypad value
  NEXT
RETURN

' -----[ Subroutine - Update ]-----------------------------------------------------
' Graphical depiction of keypad
Update:
  DEBUG CRSRXY,0,7,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+",CR,
    "|   |   |   |   |",CR,
    "+---+---+---+---+"
RETURN

' -----[ Subroutine - Display ]---------------------------------------------------
' Display button pressed in keypad graphic
Display:
  IF KeyPad.BIT15 THEN  DEBUG CRSRXY, 02,08,"1"
  IF Keypad.BIT14 THEN  DEBUG CRSRXY, 06,08,"2"
  IF KeyPad.BIT13 THEN  DEBUG CRSRXY, 10,08,"3"
  IF Keypad.BIT12 THEN  DEBUG CRSRXY, 14,08,"A"
  IF KeyPad.BIT11 THEN  DEBUG CRSRXY, 02,10,"4"
  IF Keypad.BIT10 THEN  DEBUG CRSRXY, 06,10,"5"
  IF KeyPad.BIT9  THEN  DEBUG CRSRXY, 10,10,"6"
  IF Keypad.BIT8  THEN  DEBUG CRSRXY, 14,10,"B"
  IF KeyPad.BIT7  THEN  DEBUG CRSRXY, 02,12,"7"
  IF Keypad.BIT6  THEN  DEBUG CRSRXY, 06,12,"8"
  IF KeyPad.BIT5  THEN  DEBUG CRSRXY, 10,12,"9"
```

```
  IF Keypad.BIT4  THEN  DEBUG CRSRXY, 14,12,"C"
  IF KeyPad.BIT3  THEN  DEBUG CRSRXY, 02,14,"*"
  IF Keypad.BIT2  THEN  DEBUG CRSRXY, 06,14,"0"
  IF KeyPad.BIT1  THEN  DEBUG CRSRXY, 10,14,"#"
  IF Keypad.BIT0  THEN  DEBUG CRSRXY, 14,14,"D"
RETURN
```

Propeller™ P8X32A Example Code

The example code below displays the button states of the 4x4 Matrix Membrane Keypad, and is a modified version of the 4x4 Keypad Reader DEMO object by Beau Schwabe.

Note: This application uses the 4x4 Keypad Reader.spin object. It also uses the Parallax Serial Terminal to display the device output. Both objects and the Parallax Serial Terminal itself are included with the with the Propeller Tool v1.2.7 or higher, which is available from the Downloads link at www.parallax.com/Propeller.

```
{{ 4x4 Keypad Reader PST.spin
Returns the entire 4x4 keypad matrix into a single WORD variable indicating which buttons are
pressed. }}

CON

  _clkmode = xtal1 + pll16x
  _xinfreq = 5_000_000

OBJ
  text : "Parallax Serial Terminal"
  KP   : "4x4 Keypad Reader"

VAR
  word  keypad

PUB start
  'start term
  text.start(115200)
  text.str(string(13,"4x4 Keypad Demo..."))
  text.position(1, 7)
  text.str(string(13,"RAW keypad value 'word'"))

  text.position(1, 13)
  text.str(string(13,"Note: Try pressing multiple keys"))

  repeat
    keypad := KP.ReadKeyPad     '<-- One line command to read the 4x4 keypad
    text.position(5, 2)
    text.bin(keypad>>0, 4)      'Display 1st ROW
    text.position(5,3)
    text.bin(keypad>>4, 4)      'Display 2nd ROW
    text.position(5, 4)
    text.bin(keypad>>8, 4)      'Display 3rd ROW
    text.position(5, 5)
    text.bin(keypad>>12, 4)     'Display 4th ROW
    text.position(5, 9)
    text.bin(keypad, 16)        'Display RAW keypad value
```