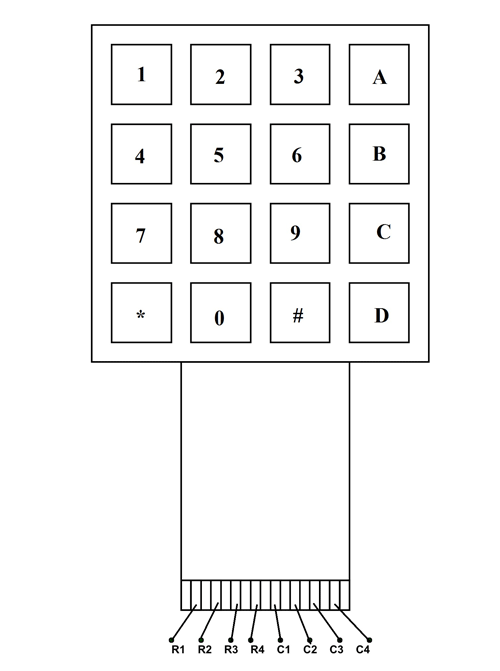
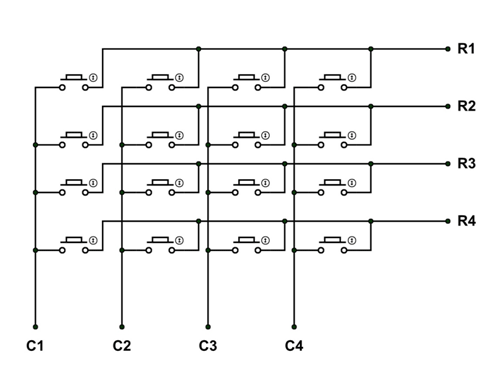
4x4 Keypad ZRX 543



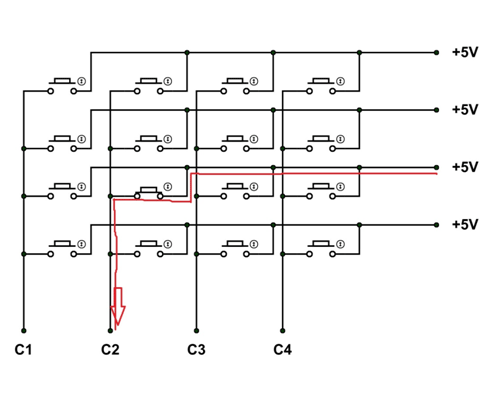
A 4X4 KEYPAD have EIGHT TERMINALS. In them four are ROWS of MATRIX and four are COLUMNS of MATRIX. These 8 PINS are driven out from 16 buttons present in the MODULE. Those 16 alphanumeric digits on the MODULE surface are the 16 buttons arranged in MATRIX formation. The internal structure of 4X4 KEYPAD MODULE is shown below.



In this keypad, as 16 keys are connected in matrix formation the module is a little complex to use. The module gives only 8 pins as a way for interacting with 16 buttons. Here are the steps to use the module:

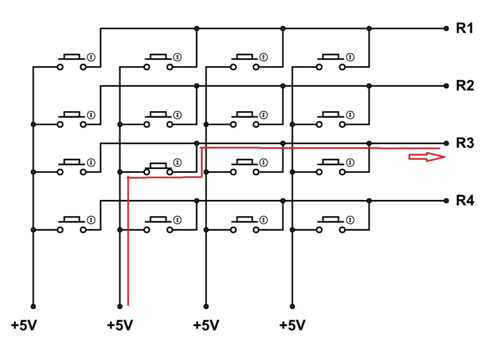
Consider we have connected the KEYPAD MODULE to a TM4C123.

STEP1: First set all ROWS to OUTPUT and set them at +V. Next set all COLUMNS as INPUT to sense the HIGH logic. Now consider a button is pressed on keypad. And that key is at 2ND COLUMN and 3rd ROW.



With the button being pressed the current flows as shown in figure. With that a voltage of +V appears at terminal C2. Since the COLUMN pins are set as INPUTS, the controller can sense C2 going high. The controller can be programmed to remember that C2 going high and the button pressed is in C2 COLUMN.

**STEP2:**Next set all COLUMNS to OUTPUT and set them at +V. Next set all ROWS as INPUT to sense the HIGH logic. Since the key pressed is at 2ND COLUMN and 3rd ROW. The current flows as shown below.



With that current flow a positive voltage of +V appears at R3 pin. Since all ROWS are set as INPUTS, the controller can sense +V at R3 pin. The controller can be programmed to remember the key being pressed at third ROW of KEYPAD MATRIX.

From previous step, we have known the COLUMN number of key pressed and now we know ROW number. With that we can match the key being pressed. We can take the key INPUT provided by this way for 4X4 KEYPAD MODULE.

/// Create definition for PORT B registers  
#define GPIO\_PORTB\_DATA\_R       (\*((volatile unsigned long \*)0x400053FC))  
#define GPIO\_PORTB\_DIR\_R        (\*((volatile unsigned long \*)0x40005400))  
#define GPIO\_PORTB\_PUR\_R        (\*((volatile unsigned long \*)0x40005510))  
#define GPIO\_PORTB\_DEN\_R        (\*((volatile unsigned long \*)0x4000551C))  
#define GPIO\_PORTB\_CR\_R         (\*((volatile unsigned long \*)0x40005524))  
#define GPIO\_PORTB\_AMSEL\_R      (\*((volatile unsigned long \*)0x40005528))  
#define GPIO\_PORTB\_PCTL\_R       (\*((volatile unsigned long \*)0x4000552C))  
#define SYSCTL\_RCGC2\_R          (\*((volatile unsigned long \*)0x400FE108))  
  
//Create definitions for Port E registers  
#define GPIO\_PORTE\_DATA\_R       (\*((volatile unsigned long \*)0x400243FC))  
#define GPIO\_PORTE\_DIR\_R        (\*((volatile unsigned long \*)0x40024400))  
#define GPIO\_PORTE\_PUR\_R        (\*((volatile unsigned long \*)0x40024510))  
#define GPIO\_PORTE\_DEN\_R        (\*((volatile unsigned long \*)0x4002451C))  
#define GPIO\_PORTE\_CR\_R         (\*((volatile unsigned long \*)0x40024524))  
#define GPIO\_PORTE\_AMSEL\_R      (\*((volatile unsigned long \*)0x40024528))  
#define GPIO\_PORTE\_AFSEL\_R      (\*((volatile unsigned long \*)0x40024420))  
#define GPIO\_PORTE\_PCTL\_R       (\*((volatile unsigned long \*)0x4002452C))  
  
//Create definitions for Port F registers  
#define GPIO\_PORTF\_DATA\_R       (\*((volatile unsigned long \*)0x400253FC))  
#define GPIO\_PORTF\_DIR\_R        (\*((volatile unsigned long \*)0x40025400))  
#define GPIO\_PORTF\_AFSEL\_R      (\*((volatile unsigned long \*)0x40025420))  
#define GPIO\_PORTF\_PUR\_R        (\*((volatile unsigned long \*)0x40025510))  
#define GPIO\_PORTF\_DEN\_R        (\*((volatile unsigned long \*)0x4002551C))  
#define GPIO\_PORTF\_LOCK\_R       (\*((volatile unsigned long \*)0x40025520))  
#define GPIO\_PORTF\_CR\_R         (\*((volatile unsigned long \*)0x40025524))  
#define GPIO\_PORTF\_AMSEL\_R      (\*((volatile unsigned long \*)0x40025528))  
#define GPIO\_PORTF\_PCTL\_R       (\*((volatile unsigned long \*)0x4002552C))  
  
//Create definitions for Port D registers  // Keypad ROW Pins as output  
#define GPIO\_PORTD\_DATA\_R       (\*((volatile unsigned long \*)0x400073FC))  
#define GPIO\_PORTD\_DIR\_R        (\*((volatile unsigned long \*)0x40007400))  
#define GPIO\_PORTD\_AFSEL\_R      (\*((volatile unsigned long \*)0x40007420))  
#define GPIO\_PORTD\_PUR\_R        (\*((volatile unsigned long \*)0x40007510))  
#define GPIO\_PORTD\_DEN\_R        (\*((volatile unsigned long \*)0x4000751C))  
#define GPIO\_PORTD\_LOCK\_R       (\*((volatile unsigned long \*)0x40007520))  
#define GPIO\_PORTD\_CR\_R         (\*((volatile unsigned long \*)0x40007524))  
#define GPIO\_PORTD\_AMSEL\_R      (\*((volatile unsigned long \*)0x40007528))  
#define GPIO\_PORTD\_PCTL\_R       (\*((volatile unsigned long \*)0x4000752C))  
  
//Create definitions for Port A registers  // Keypad Column Pins as input  
#define GPIO\_PORTA\_DATA\_R       (\*((volatile unsigned long \*)0x400043FC))  
#define GPIO\_PORTA\_DIR\_R        (\*((volatile unsigned long \*)0x40004400))  
#define GPIO\_PORTA\_AFSEL\_R      (\*((volatile unsigned long \*)0x40004420))  
#define GPIO\_PORTA\_PUR\_R        (\*((volatile unsigned long \*)0x40004510))  
#define GPIO\_PORTA\_DEN\_R        (\*((volatile unsigned long \*)0x4000451C))  
#define GPIO\_PORTA\_LOCK\_R       (\*((volatile unsigned long \*)0x40004520))  
#define GPIO\_PORTA\_CR\_R         (\*((volatile unsigned long \*)0x40004524))  
#define GPIO\_PORTA\_AMSEL\_R      (\*((volatile unsigned long \*)0x40004528))  
#define GPIO\_PORTA\_PCTL\_R       (\*((volatile unsigned long \*)0x4000452C))  
  
//Global variables  
int numbers[16] = {0x40,0x79,0x24,0x30,0x19, // Each value turns on bits needed  
0x12,0x02,0x78,0x00,0x10,   
0x08, 0x00, 0x46, 0x40, 0x06, 0x0E}; // to show numbers in display  
int digit1, digit2, digit3, digit4; // Number to be displayed in each digit  
unsigned long SW2; // On-board push button  
unsigned long count=9999; // Counter for button presses  
  
int before=1; // States if button was pressed in previous  
// loop cycle to enable counter.  
  
void PortAD\_Init(); // PORTA: Keypad column, PORTD: keypad row  
//   Function Prototypes  
void PortBEF\_Init(void); // PORT B: 7-SED a, b, c, d, e, f, g, LED selection  
// PORT E: Select Digit 1, 2, 3, 4 (L-R)  
// PORT F: SW2 to turn on  
void Delay(void);  
void Display(int number);  
int cdelay = 100;  
int vdelay = 0;  
unsigned long swCont = 0;  
//  Subroutines Section  
unsigned char getKey(void);  
//const unsigned char keymap[4][4];  
  
void PortAD\_Init(){  
SYSCTL\_RCGC2\_R |= 0x00000009; // clk for A and D  
// Initialize PORTD as Row output  
GPIO\_PORTD\_LOCK\_R = 0x4C4F434B;   // Unlock PortD  
  GPIO\_PORTD\_CR\_R = 0x0F;           // Allow changes to PD3-PD0      
  GPIO\_PORTD\_AMSEL\_R = 0x00;        // Disable analog function  
  GPIO\_PORTD\_PCTL\_R = 0x00000000;   // GPIO clear bit PCTL    
  GPIO\_PORTD\_DIR\_R = 0x0F;          // PD3-PD0 output    
  GPIO\_PORTD\_AFSEL\_R = 0x00;        // No alternate function  
  GPIO\_PORTD\_PUR\_R = 0x00;          // No Enable pullup resistor        
  GPIO\_PORTD\_DEN\_R = 0x0F;          // Enable digital pins PD3-PD0     
// Initialize PORTA as column input  
GPIO\_PORTA\_LOCK\_R = 0x4C4F434B;   // Unlock PortA  
  GPIO\_PORTA\_CR\_R = 0xF0;           // Allow changes to PA7-PA4      
  GPIO\_PORTA\_AMSEL\_R = 0x00;        // Disable analog function  
  GPIO\_PORTA\_PCTL\_R = 0x00000000;   // GPIO clear bit PCTL    
  GPIO\_PORTA\_DIR\_R = 0x00;          // PA7-PA4 input    
  GPIO\_PORTA\_AFSEL\_R = 0x00;        // No alternate function  
  GPIO\_PORTA\_PUR\_R = 0xF0;          // Enable pullup resistors on PA7-PA4        
  GPIO\_PORTA\_DEN\_R = 0xF0;          // Enable digital pins PA7-PA4  
  
}  
  
// Subroutine to initialize ports B, E and F  
void PortBEF\_Init(void){  
  SYSCTL\_RCGC2\_R |= 0x00000002;     // Port B clock initialized  
  GPIO\_PORTB\_CR\_R = 0x7F;           // Allow changes to PB6-PB0        
  GPIO\_PORTB\_AMSEL\_R = 0x00;        // Disable analog function  
  GPIO\_PORTB\_PCTL\_R = 0x00000000;   // GPIO clear bit PCTL    
  GPIO\_PORTB\_DIR\_R = 0x7F;          // Set PB6-PB0 outputs    
  GPIO\_PORTB\_PUR\_R = 0x00;          // Enable pullup resistors on PB4,PF0        
  GPIO\_PORTB\_DEN\_R = 0x7F;          // 7) Enable digital pins PB6-PB0          
  
  SYSCTL\_RCGC2\_R |= 0x00000010;     // Port E clock initialized  
  GPIO\_PORTE\_CR\_R = 0x0F;           // Allow changes to PE4-0        
  GPIO\_PORTE\_AMSEL\_R = 0x00;        // Disable analog function  
  GPIO\_PORTE\_PCTL\_R = 0x00000000;   // GPIO clear bit PCTL    
  GPIO\_PORTE\_DIR\_R = 0x0F;          // PE3-PE0 output    
  GPIO\_PORTE\_PUR\_R = 0x00;          // Disable pullup resistors        
  GPIO\_PORTE\_DEN\_R = 0x0F;          // Enable digital pins PE3-PE0  
  
  SYSCTL\_RCGC2\_R |= 0x00000020;     // Port F clock initialized  
  GPIO\_PORTF\_LOCK\_R = 0x4C4F434B;   // Unlock PortF  
  GPIO\_PORTF\_CR\_R = 0x1F;           // Allow changes to PB4-PB0      
  GPIO\_PORTF\_AMSEL\_R = 0x00;        // Disable analog function  
  GPIO\_PORTF\_PCTL\_R = 0x00000000;   // GPIO clear bit PCTL    
  GPIO\_PORTF\_DIR\_R = 0x0E;          // PF4,PF0 input    
  GPIO\_PORTF\_AFSEL\_R = 0x00;        // No alternate function  
  GPIO\_PORTF\_PUR\_R = 0x11;          // Enable pullup resistors on PF4,PF0        
  GPIO\_PORTF\_DEN\_R = 0x1F;          // Enable digital pins PF4-PF0          
}  
  
  
int keymap[4][4] = {  
{ 1, 2, 3, 10},  
{ 4, 5, 6, 11},  
{ 7, 8, 9, 12},  
{ 14, 0, 15, 13},  
};  
   
int row, column;  
  
unsigned char getKey(void){  
// first check if anykey has been pressed  
GPIO\_PORTD\_DATA\_R = 0x00;  
column = GPIO\_PORTA\_DATA\_R &0xF0;  
if(column == 0xF){  
return 0;  
}  
while (1){ // find out which row has been activated  
row = 0;  
GPIO\_PORTD\_DATA\_R = 0x0E;  
Delay();  
column = GPIO\_PORTA\_DATA\_R &0xF0;  
if(column != 0xF0)   
{break;}  
  
row = 1;  
GPIO\_PORTD\_DATA\_R = 0x0D;  
Delay();  
column = GPIO\_PORTA\_DATA\_R &0xF0;  
if(column != 0xF0)   
{break;}  
  
row = 2;  
GPIO\_PORTD\_DATA\_R = 0x0B;  
Delay();  
column = GPIO\_PORTA\_DATA\_R &0xF0;  
if(column != 0xF0)   
{break;}  
  
row = 3;  
GPIO\_PORTD\_DATA\_R = 0x07;  
Delay();  
column = GPIO\_PORTA\_DATA\_R &0xF0;  
if(column != 0xF0)   
{break;}  
}  
  
if (column == 0xE0) return keymap[row][0]; /\* key in column 0 \*/  
if (column == 0xD0) return keymap[row][1]; /\* key in column 1 \*/  
if (column == 0xB0) return keymap[row][2]; /\* key in column 2 \*/  
if (column == 0x70) return keymap[row][3]; /\* key in column 3 \*/  
return 0; /\* just to be safe \*/  
}  
  
unsigned char key;  
int main(void){  
  PortBEF\_Init();         // Initialize microcontroller ports  
PortAD\_Init();  
  //PortE\_Init();  
  //PortF\_Init();  
  
while(1){  
key = getKey();  
if(key != 0){  
Display(key);  
}  
else Display(key);  
  }  
Delay();  
}  
  
// Creates 0.1ms delay  
void Delay(void){  
unsigned long volatile time;  
time = 727240\*200/91000;  // 0.1 ms  
  while(time){  
time--;  
  }  
}  
  
// Takes digit and number for LED display  
void Display(int number){  
int number1;  
//number1 = atoi(key);  
GPIO\_PORTB\_DATA\_R = 0x00; // Turns off LEDs  
GPIO\_PORTE\_DATA\_R = 8; // Selects digit  
GPIO\_PORTB\_DATA\_R = numbers[number]; // Turns on number in selected digit  
Delay(); // Wait 0.1 ms  
}