CompE 375

Inputs:

Switches, Multiplexing, Keypads

Ken Arnold

 1

Overview

- Basic Input Devices
- Switch Input
- Matrix Keypad Input
- Multiplexing
- Interrupt Driven I/O



 2

Human Interfaces (Input)

- Switches, Buttons, Keypads and Keyboards
- Individual, Panels
- Keypads:
  - Steel, Elastomeric, Membrane and Dome
- Continuous Analog:
  - Rotary: Pots (R), Wheel and Shaft Encoders
  - Linear: Sliders
- Graphic Inputs: Mouse, Touchpad, ...

 3

Switch and Keyboard Input

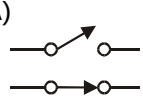
- Switches, Keypads
  - Simple switch inputs
  - Multiplexed Keypads
  - Multiplexed Keyboards
- Intelligent Keyboards
  - PC Style
  - ASCII



 4

Switches

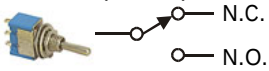
- xPnT switch
  - x Poles, n Throw (Positions)
  - Poles: number of independent circuits
  - Throw: number of switch positions
  - Make-Before-Break vs. Break-Before-Make
- Simplest: SPST (Form A)
- NO: Normally Open
- NC: Normally Closed



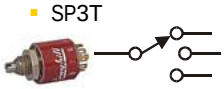
 5

Switch Examples

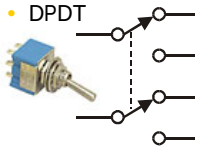
- SPDT (Form B)



- SP3T



- DPDT

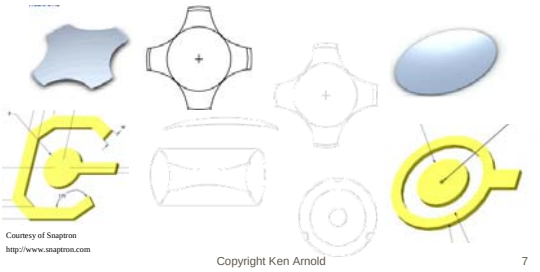


- SPST N.O. Pushbutton

 6

Steel Dome and PCB layout

- Single Sided PCB
- Double Sided PCB



Courtesy of Snaptron
http://www.snaptron.com

 7

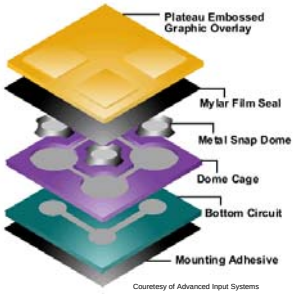
Dome Switch Typ. Specs

- Contact configuration: SPST N.O.
- Contact current capacity: Logic Levels
- Contact Bounce: 1 mS typ; 20 mS max
- Trip Force: 10-100s +/- 30 grams typical
- Material: Stainless Steel, Nickel Plated
- Operating temp: 0 to +50 C or better
- Contact life: Exceeds 5 million cycles
- Dependent Upon PCB

 8

Keypad Assembly

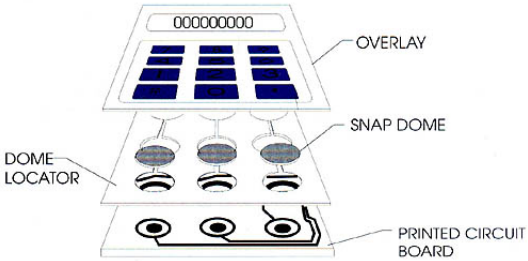
- Laminated Panel
  - Graphic Layer
  - Seal
  - Domes Captured in Spacer
  - Adhesive
  - PCB



Couretesy of Advanced Input Systems
http://www.advanced-input.com

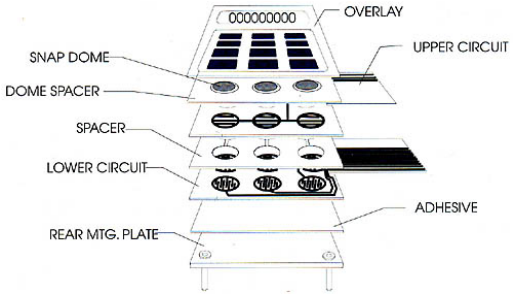
 9

Keypad Switches - Dome



 10

Membrane Switches



 11

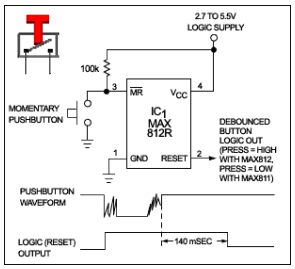
Input

- Input (Reading) the I/O Ports
  - Set DDR to Input with pull-up R (1)
  - External Circuit Pulls I/O Pin Low for 0
  - Switches Must Be Debounced
- Reading a Matrix Keypad

 12

Switch Bounce

- Mechanical Contacts Bounce, Due to Elastic Collision
- Can be De-Bounced with Hardware:
- Usually Done in Code



 13

Debouncing with Software

- Look for first transition low (ON)
- Ignore additional switch transitions for debounce period, typically ~50 mS
- If switch is still on, key is down.
- Look for first transition high (OFF)
- Ignore transitions for ~50 mS
- If switch is still high, key is up.

 14

Switch Matrix

- Switches organized as Row/Column
- Switch shorts row line to column line
- Walking zero on rows to activate one row at a time
- Check for low level on column inputs to determine which key in the current row is pressed

 15

Why Multiplex I/O?

- It saves the limited I/O pins on a uC
- Non-multiplexed requires 1 bit per I/O:
  - 16 switches requires 16 I/O pins
- Multiplexing shares I/O lines:
  - 16 switches require 4+4 = 8 lines
- This can be done for displays as well, due to persistence of vision effect

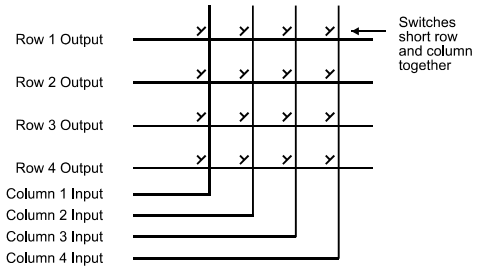
 16

Keyboards - Matrix

- Reduces the number of I/O pins req'd
- Software Driven is lowest cost
  - Increases Software Complexity
  - MUST debounce in software
- Hardware Multiplexed
  - Independent, interrupt driven
  - Hardware compatibility issues

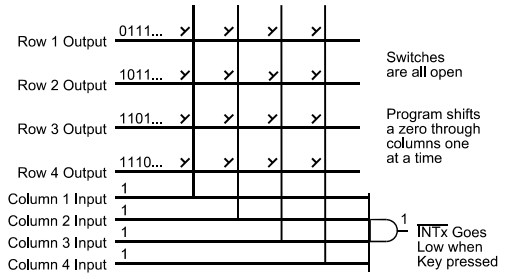
 17

Simple Switch Matrix



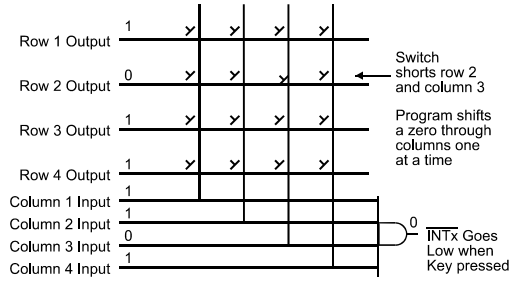
 18

Scanning the Switch Matrix



 19

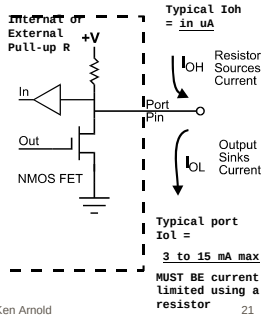
Multiplexed Switch Matrix



 20

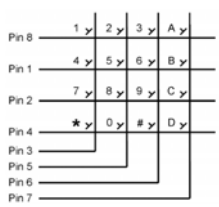
Simplified Port Circuit

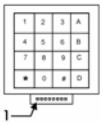
- Open-Drain:
  - Pull-up R
  - Switch is a pull-down (nFET Sinks Current)
- Active Switch sinks more current, so
- Sink current is large compared to source



 21

Keypad Connections





Keypad is jameco.com part #169244, with an 8 pin single row male header, with pin spacing = 0.1" soldered to the keypad, so it can be plugged directly into the solderless proto board.

 22

Keypad Connections to CPU:

CPU Signal	Keypad	CPU Pin#	Keypad Pin#
P1.0	Row 2	1	1
P1.1	Row 3	2	2
P1.2	Col 1	3	3
P1.3	Row 4	4	4
P1.4	Col 2	5	5
P1.5	Col 3	6	6
P1.6	Col 4	7	7
P1.7	Row 1	8	8

Connections are shown for the sample keypad1.c program, which uses a software driven, bit-banged interface.

 23

Scanning a Keypad

```
unsigned char code row[] = {0xEF, 0xDF, 0xBF, 0x7F };
            /* for strobing each row */
    /* 4 MSBs are row outputs, 4 LSBs are col inputs */

unsigned int keys /* for storing key info */

void keypress (void) interrupt 0 // ISR for /INT0
  {
   unsigned char I;
   keys = 0xFFFF;  // get ready to input key press data
   for (I=0; I <= 3; I++)
    {
     P1 = row[I];  // strobe low one row at a time
     keys = (keys<<4) | (P1&0x0F);
        // read and store info from each column
    }
```

 24