ECE 372 – Microcontroller Design

Digital IO Ports

The University of Arizona
Tucson Arizona

ECE 372 – Microcontroller Design
Digital IO Ports

- Digital IO Ports
  - *E.g. Port A, Port B*
  - Used to interface with many devices
    - Switches
    - LEDs
    - LCD
    - Keypads
    - Relays
    - Stepper Motors
  - Interface with digital IO requires us to connect the devices correctly and write code to interface with the devices

ECE 372 – Microcontroller Design

Digital IO Ports

- PIC24F – *Where can we connect external hardware?*
  - Unused ports (most ports has digital IO functionality)
  - Expand through the serial interface
    - SPI serial peripheral interface
    - UART
    - I^2C
  - External Components to Expand IO Ports

ECE 372 – Microcontroller Design

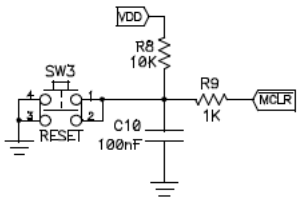
Digital IO Ports

- PIC24F – *Where can we connect external hardware?*
  - 16-bit 28-pin Starter Board
    - Port B (RB0 – RB15)
    - Port A (RA0, RA1, RA4)
    - RB0, RB1, RB4, and RA4 can only be used on stand-alone programming
      - Corresponds to pins 3, 4, 11, and 12

ECE 372 – Microcontroller Design

Digital IO Ports

- Digital Inputs
  - What does the following circuit do?



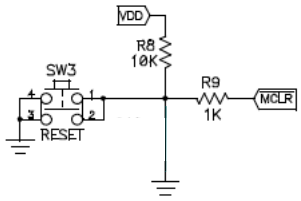
$\overline{\text{MCLR}}$ (Reset) Switch Schematic from 16-bit 28-pin Starter Board

  - Note: $\overline{\text{MCLR}}$ is an active low signal.
    - Active low signals are ***asserted*** if the input is low (0)
    - For PICs, if the $\overline{\text{MCLR}}$ input is 0 the PIC is reset

ECE 372 – Microcontroller Design

Digital IO Ports

- Used for Hardware De-Bouncing of Switch Input

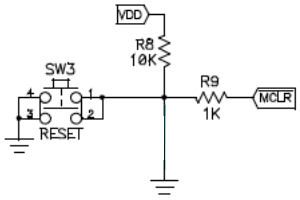


  - Resistor connected to Vdd is used to ***Pull-Up*** the input pin to a logical high
    - Vdd is 3.3V for PIC24F
    - Logic high (1) will be seen when button is not pressed
    - Logic high (1) will be seen when button is not pressed
      - Pressing switch creates conduction path to ground
  - Without capacitor, simple switches exhibit bouncing

ECE 372 – Microcontroller Design

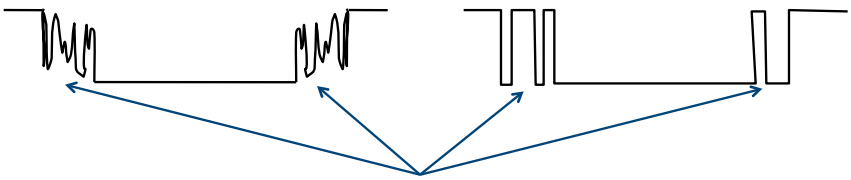
Digital IO Ports

- Used for Hardware De-Bouncing of Switch Input



Analog Voltage at Input Pin:

Digital Value Read by PIC:

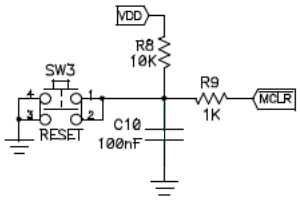


Bouncing

ECE 372 – Microcontroller Design

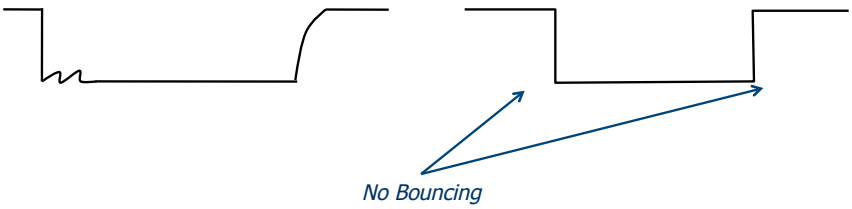
Digital IO Ports

- Charging and discharging of capacitor reduces bouncing behavior



Analog Voltage at Input Pin:

Digital Value Read by PIC:

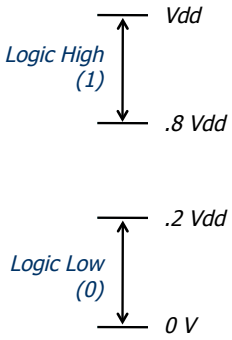


No Bouncing

ECE 372 – Microcontroller Design

Digital IO Ports

- When is a digital input a logic high (1) versus logic low (0)
  - Depends on voltage standard used by the digital device
  - Lots of standards exist
- PIC24F
  - Logic Low Input: < 0.2 Vdd (0.66V)
  - Logic High Input: > 0.8 Vdd (2.64V)
  - What about between 0.2 and .8 Vdd?
    - All digital inputs have Schmitt Triggers

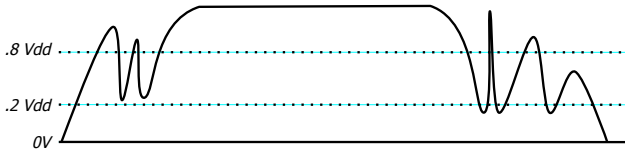


ECE 372 – Microcontroller Design

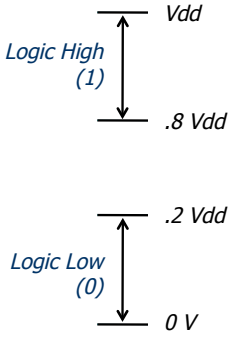
Digital IO Ports

- Schmitt Trigger
  - Removes hysteresis between transition from low to high and high to low
  - Helps in de-bouncing digital inputs *(but does not solve it on its own)*

Analog Voltage at Input Pin:



Digital Value Read by PIC with Schmitt Trigger:



ECE 372 – Microcontroller Design

Digital IO Ports

- Software De-bouncing
  - De-bouncing can be done in software using timed waits
  - Simplifies external hardware needs
  - Pseudocode for software de-bounce for button press

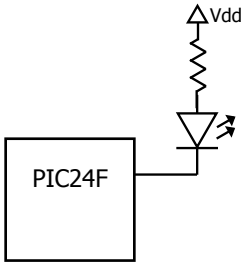
```
// wait for button press
while (switch == 1);

// delay for 5 to 10 ms
for (i=0; i<SOME_LARGER_NUMBER; i++);
```

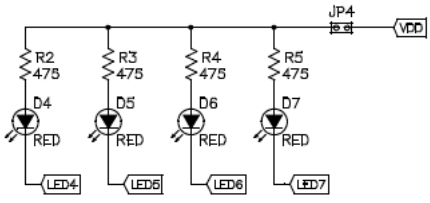
  - Bad example of software de-bounce as timing using for loops is imprecise.
    - Is there a better way?

ECE 372 – Microcontroller Design

Digital IO Ports

- Directly Controlling LEDs
  - PIC24F can directly control LED from output ports
    - Write 0 to port: Turns LED on
    - Write 1 to port: Turns LED off



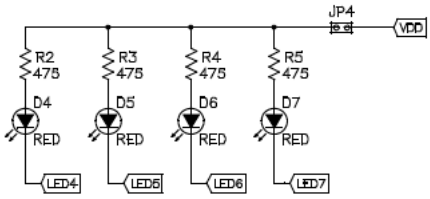


LEDs schematic from 16-bit 28-pin Starter Board

ECE 372 – Microcontroller Design

Digital IO Ports

- Directly Controlling LEDs
  - How much current is drawn by IO?
    - I = V / R = 3.3 / 475 = 6.9 mA
  - How much current can be drawn by the PIC's IO pin?
    - Check the datasheet
    - 25 mA for PIC24F
  - How much current can be drawn by ***ALL*** IO pins of the PIC?
    - Check the datasheet
    - 200 mA



LEDs schematic from 16-bit 28-pin Starter Board

ECE 372 – Microcontroller Design

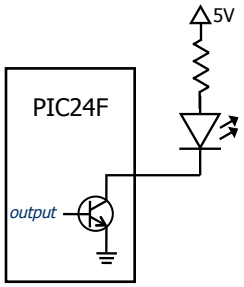
Digital IO Ports

- Configure IO Pins for PIC24F
  - Each IO Port has at least three registers
    - TRIS : Controls tri-state driver for direction (1 -> Input, 0 -> Output)
    - LAT : Output latch for writing value to the port (or reading current value assigned to port)
    - PORT : Value read directly from the port
  - Additional Registers Needed for Some IOs
    - CNPU : Enables weak internal pull-up resistor
    - AD1PCFG : Configures ports as analog or digital
    - ODC : Enables open drain collector configuration

ECE 372 – Microcontroller Design

Digital IO Ports

- Directly Controlling LEDs
  - What if we need to to control a 5V digital output?
    - Can use the Open-Drain Collector configuration for the digital outputs
    - ODC registers control configuration (1 -> open drain configuration)

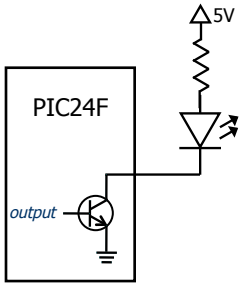


- Use NPN transistor as digital switch
  - NPN in On if base is high (1)
  - Still limited to current limit of digital IO pin

ECE 372 – Microcontroller Design

Digital IO Ports

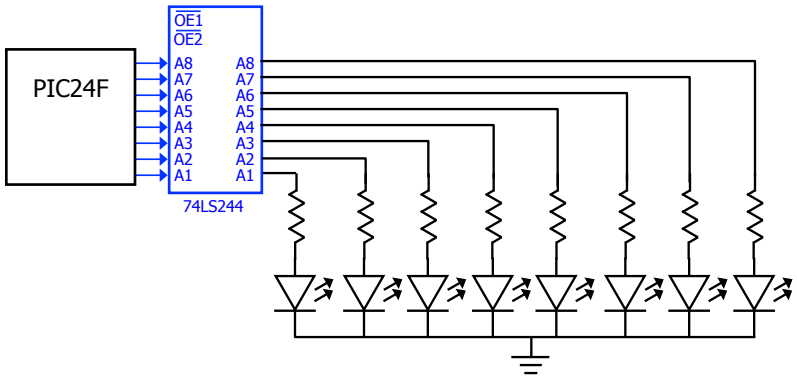
- Directly Controlling LEDs
  - Brightness of LED can be controlled by controlling the current
  - LEDs also have maximum current draw (exceeding this will destroy the LED)
  - Using open collector, what is the smallest value resistor can we use:
    - R = 5V / 25 mA = 200 Ω
    - With 330Ω : I = 5V / 330 Ω = 15 mA



ECE 372 – Microcontroller Design

Digital IO Ports

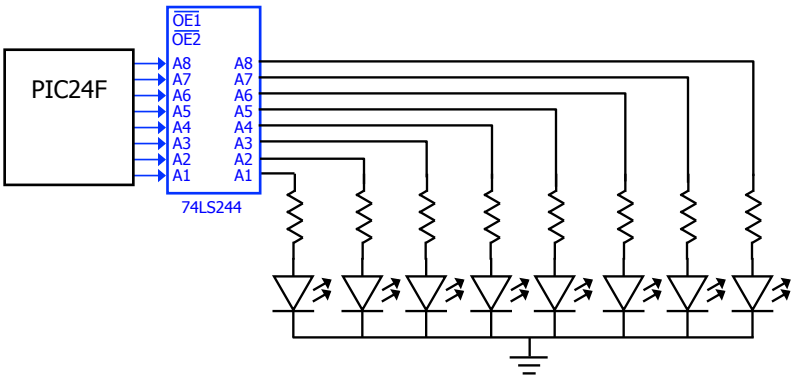
- Directly Controlling LEDs
  - We can use external IC to drive the LEDs
    - Reduce current driven by microcontroller and protects it
  - 74LS244IC Tri-state Buffer/Line Driver/Line Receiver
    - Used to provide power for LEDs
    - Has OE signals that can be used to disable the output



ECE 372 – Microcontroller Design

Digital IO Ports

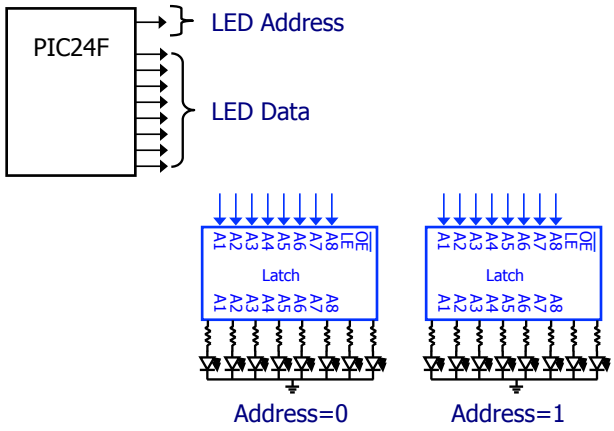
- Directly Controlling LEDs
  - Limitation - Can only drive as many LEDs as available pins
  - Solutions:
    - Latching
    - Scanning
    - Multiplexing



ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Latching
  - Control LED by using a latch hardware
  - Use additional hardware to control when to latch values
    - Similar to memory mapped IO

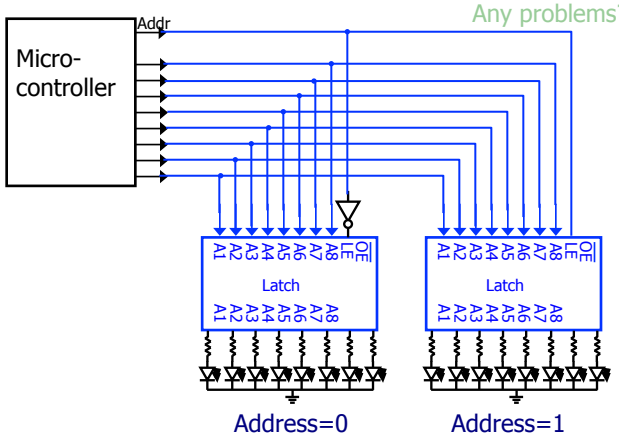


ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Latching
  - Control LED by using a latch hardware
  - Use additional hardware to control when to latch values
    - Similar to memory mapped IO

Always writing last LED value!
Any problems??

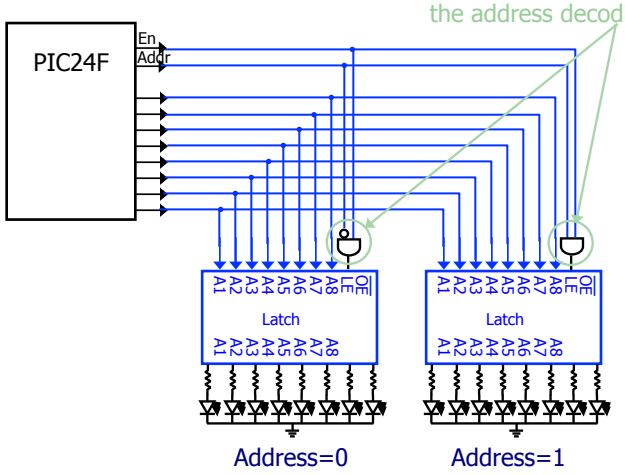


ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Latching
  - Add enable signal from microcontroller

What did we build?
Is there another way to implement the address decoding?

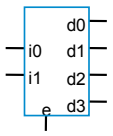


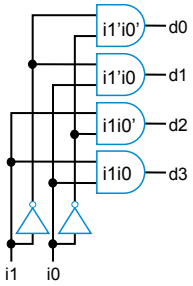
ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Latching
  - Use decoder for addressing LED banks

- Decoder - Converts input binary number to *single* high output
  - 2-input decoder: four possible input binary numbers
    - Thus, four outputs, one for each possible input binary number
  - Enable signal
    - Outputs all 0 if e=0
    - Regular behavior if e=1

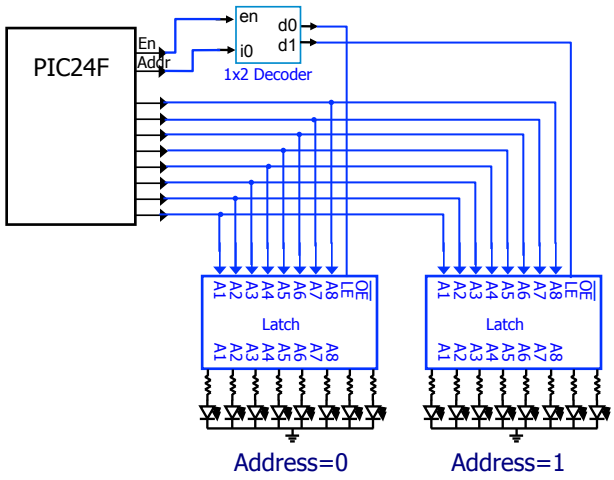




ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Latching
  - Use decoder for addressing LED banks



ECE 372 – Microcontroller Design

Digital IO Ports

- LEDs - Scanning
  - Use software executing on the microcontroller to control LEDs row by row, col by col, or bank by bank (depending on organization)
  - Operation
    - Continually scan LEDs fast enough that human eye cannot detect it
      - Similar to refresh rate of TVs and Monitors
    - Enable one column of LEDs by writing 0 to output port controlling desired column
    - Write LEDs value for select row
- LEDs – Multiplexed
  - Similar to scanning but uses decoders to specify the row and columsn
    - Continually scan LEDs fast enough that human eye cannot detect it
      - Similar to refresh rate of TVs and Monitors
    - Enable LED column by outputting column address to column decoder
    - Enable individual LEDs by outputting row address to row decoder
  - Cannot access all LEDs simultaneously

ECE 372 – Microcontroller Design

Digital IO Ports

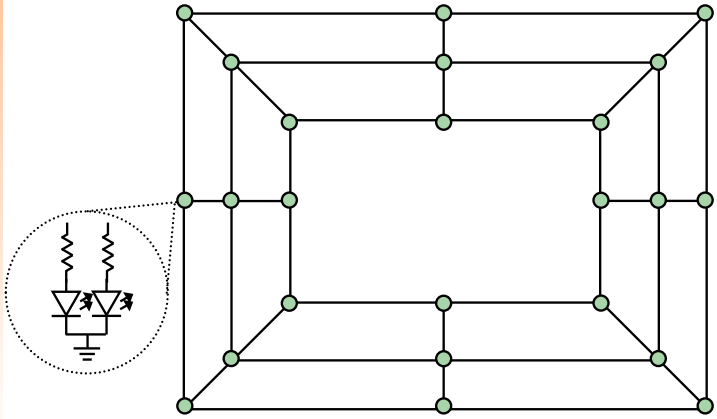
- How many LEDs can I control using only eight output pins using the following schemes?
  - Latched:
  - Scanned
  - Multiplexed:

ECE 372 – Microcontroller Design

Digital IO Ports

- How many LEDs can I control using only eight output pins using the following schemes?
  - Latched:
    - 64 LEDs – Individual latch used for each LED
    - 1 Enable Output, 6 Address Lines, and 1 Data Line
    - Six address lines allow for access to 2^6 latches (or LEDs)
  - Scanned
    - 16 LEDs – 4 columns by 4 rows
  - Multiplexed:
    - 256 LEDs – 16 columns by 16 rows
    - Enabled by using 4x16 decoder to select both row and columns

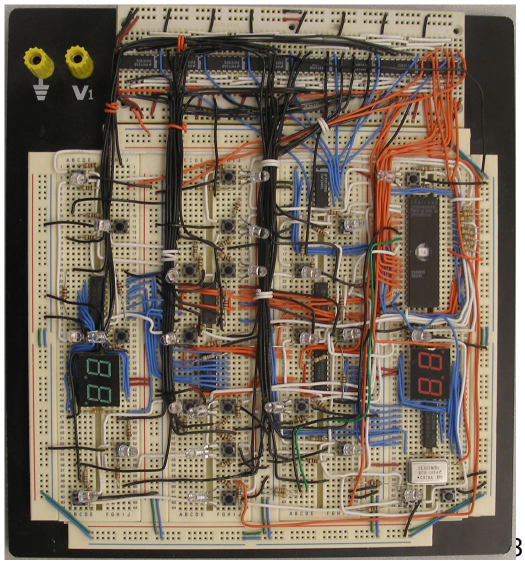
- Mill Game
  - *How many pins do we need to connect the 48 LEDs in the following diagram?*



ECE 372 – Microcontroller Design

Digital IO Ports

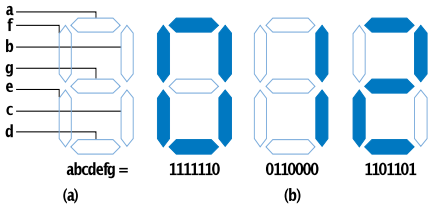
- Mill Game



ECE 372 – Microcontroller Design

Digital IO Ports

- 7-Segment LEDs
  - Can directly control all seven segments using PIC IOs
  - Or, driver IC can be used to control display using fewer pins
    - 4 pins instead of 7pins
    - Typically uses BCD to represent displayed value



ECE 372 – Microcontroller Design

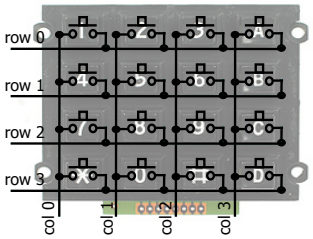
Digital IO Ports

- Keypads
  - Collection of several keys grouped into a single device
    - Can buy keypads in many different configurations
    - Build your own keypad using individual buttons
  - How can we read dozens of keys on the keypad without requiring dozens on individual input ports on our microprocessor?
    - *Hint: It's similar to how we can interface with dozens of outputs using only a few pins*

ECE 372 – Microcontroller Design

Digital IO Ports

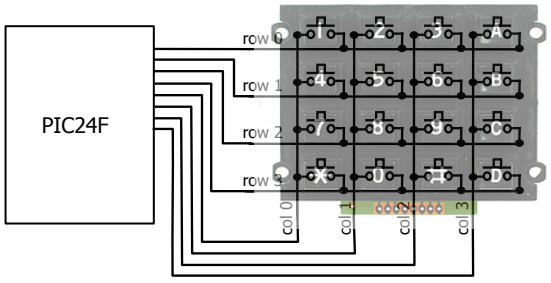
- Keypad Scanning
  - Similar to scanning for controlling multiple output LEDs
  - Scan the keys one row at a time and read the column lines to see if any key on that row were pressed



ECE 372 – Microcontroller Design

Digital IO Ports

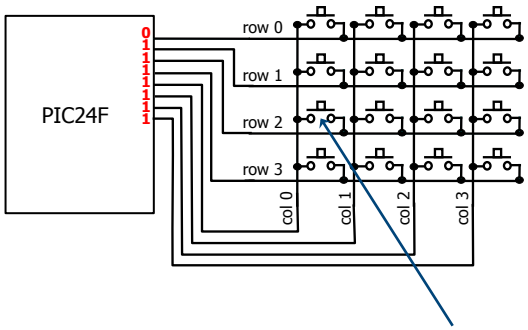
- Keypad Scanning
  - Similar to scanning for controlling multiple output LEDs
  - Scan the keys one row at a time and read the column lines to see if any key on that row were pressed



ECE 372 – Microcontroller Design

Digital IO Ports

- Keypad Scanning
  - Similar to scanning for controlling multiple output LEDs
  - Scan the keys one row at a time and read the column lines to see if any key on that row were pressed



1. Set each row output sequentially to output 0

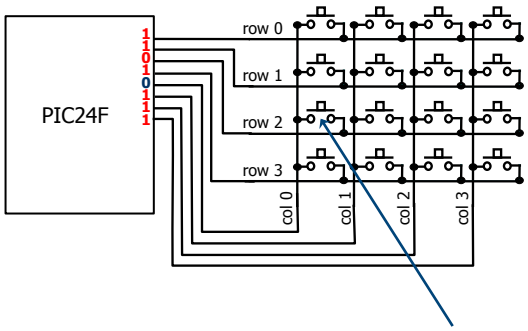
2. Check column inputs to see if key was pressed

What if this key is pressed?

ECE 372 – Microcontroller Design

Digital IO Ports

- Keypad Scanning
  - Similar to scanning for controlling multiple output LEDs
  - Scan the keys one row at a time and read the column lines to see if any key on that row were pressed



1. Set each row output sequentially to output 0

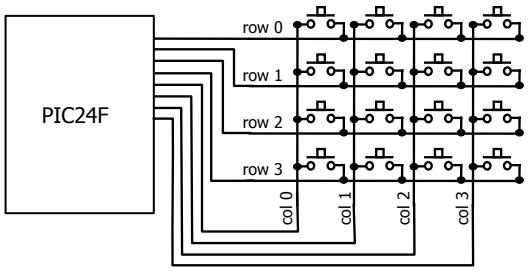
2. Check column inputs to see if key was pressed

What if this key is pressed?

ECE 372 – Microcontroller Design

Digital IO Ports

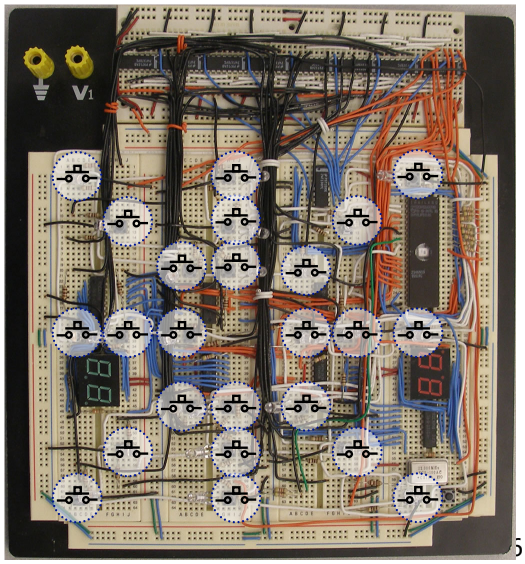
- Keypad Scanning
  - *Can you detect if more than one key is pressed?*
  - *How do you detect if no keys are pressed?*



ECE 372 – Microcontroller Design

Digital IO Ports

- Mill Game



ECE 372 – Microcontroller Design

Digital IO Ports

- Mill Game