How to use a 16 x 16 matrix to read 256 switches into the Arduino

Alex Zivanovic
March 2009

Parts needed:

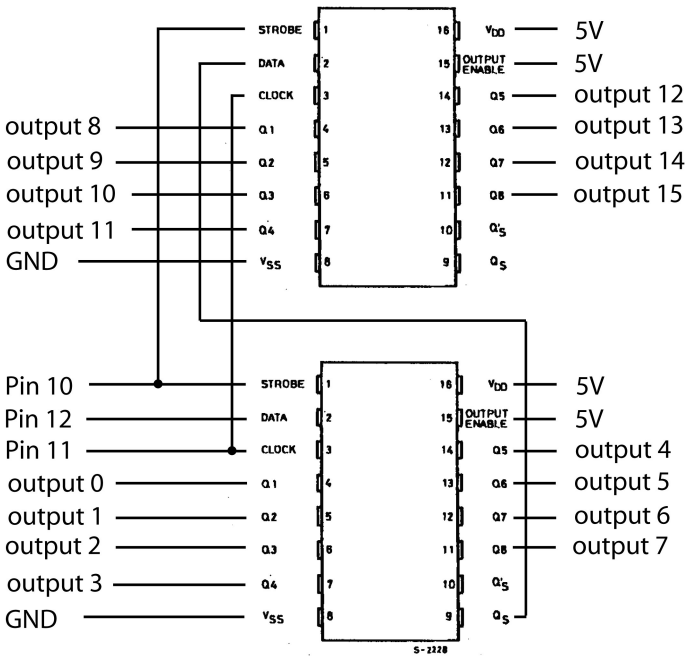
2 off 4094, available here:
http://www.rapidonline.com/Electronic-Components/Integrated-Circuits/Logic/4000-Series-CMOS-logic-family/77018/kw/hcf4094, order code: 83-0422, cost £0.46, Datasheet: http://www.rapidonline.com/netalogue/specs/83-0422.pdf

2 off 4021, available here:
http://www.rapidonline.com/Electronic-Components/Integrated-Circuits/Logic/4000-Series-CMOS-logic-family, order code: 83-0348, cost £0.30, Datasheet: http://www.rapidonline.com/netalogue/specs/83-0272e.pdf

16 off 10K resistors, available here:
http://www.rapidonline.com/Electronic-Components/Resistors-Potentiometer/Carbon-Film-Resistors/CR12-0.125W-Carbon-film-resistors/65192
code: 64-0098, cost £0.70 for 100

Set up 16 outputs:

The following shows how you can use two of 4094 to get 16 outputs using 3 digital pins on the Arduino. Connect the output pins to the columns of your matrix.



Set up 16 inputs:

Follow the tutorial here: http://www.arduino.cc/en/Tutorial/ShiftIn using two 4021 chips. Connect the Clock to pin 7, the Latch to pin 8 and the Data to pin 9 on the Arduino. Don't forget to use the pull-down resistors on each input. Connect the inputs to the rows of your matrix.

Program

The program below will print the x,y coordinates of all the switches that are currently being pressed.

```
// Program for a 16x16 matrix
// coordinate pairs of columns/rows returned in the format x,y

// Arduino pins to 4094
#define IN_clockPin 7
#define IN_latchPin 8
#define IN_dataPin 9

//Arduino pins to 4021
#define OUT_latchPin 10
#define OUT_clockPin 11
#define OUT_dataPin 12

void setup() {
  Serial.begin(57600);          //to send the data to the screen

  pinMode(IN_latchPin, OUTPUT);
  pinMode(IN_clockPin, OUTPUT);
  pinMode(IN_dataPin, INPUT);
  pinMode(OUT_latchPin, OUTPUT);
  pinMode(OUT_clockPin, OUTPUT);
  pinMode(OUT_dataPin, OUTPUT);

  digitalWrite(IN_latchPin,0);    //make sure both latches are low to begin with
  digitalWrite(OUT_latchPin,0);
}

void loop() {
  byte row;
  byte inputs_MSB;
  byte inputs_LSB;

  // put the binary pattern 0000000000000001 into the output shift register
  digitalWrite(OUT_latchPin, 0);
  shiftOut(OUT_dataPin, OUT_clockPin, MSBFIRST, 0);
  shiftOut(OUT_dataPin, OUT_clockPin, MSBFIRST, B00000001);
  digitalWrite(OUT_latchPin, 1);

  for (byte column=0 ; column < 16; column++){   //cycle through the 16 columns
   digitalWrite(IN_latchPin,1);            //latch the input data into the input shift register
   delayMicroseconds(2);
   digitalWrite(IN_latchPin,0);

   inputs_MSB = shiftIn(IN_dataPin, IN_clockPin);  //read in both bytes of the input data
   inputs_LSB = shiftIn(IN_dataPin, IN_clockPin);

   if ((inputs_MSB!=0)||(inputs_LSB!=0)){   //only need to do something if at least one input line is high
     for(byte row=0; row<8; row++){        //test each of the bits in the input byte, one by one
       if (inputs_MSB & B00000001) {
         Serial.print(column, DEC);
         Serial.print(",");
         Serial.println(row+8, DEC);       //+8 because it is the most significant byte
       }
       if (inputs_LSB & B00000001) {
         Serial.print(column, DEC);
         Serial.print(",");
         Serial.println(row, DEC);
       }
       inputs_MSB = inputs_MSB >>1;        // shift the bits along by one to test the next bit
       inputs_LSB = inputs_LSB >>1;
      }
    }
  delay(10);                          // delay between each column going high
```

```
  digitalWrite(OUT_latchPin, 0);          // make sure the latch pin is low
  digitalWrite(OUT_dataPin, 0);           // put a low on the data line
  digitalWrite(OUT_clockPin, 1);          // pulse the clock pin - shifts all the bits in the shift registers along by
one
  delayMicroseconds(20);
  digitalWrite(OUT_clockPin, 0);
  digitalWrite(OUT_latchPin, 1);          // latch the data in the output shift registers

  }
}

//-------------------------------------------------end main loop


byte shiftIn(int myIN_dataPin, int myIN_clockPin) {
  byte myDataIn = 0;
  for (int i=7; i>=0; i--)
  {
    digitalWrite(myIN_clockPin, 0);
    delayMicroseconds(2);
    if (digitalRead(myIN_dataPin)) myDataIn = myDataIn | (1 << i);
    digitalWrite(myIN_clockPin, 1);
  }
  return myDataIn;
}
```