How to Enable Interrupts on ANY pin
7/8/2010
Louis Van Blarigan

Every pin on the ATMEGA 328 (Heart of Arduino) is capable of registering a pin change interrupt. The arduino board is only set up to register external interrupts on digital pins 2 and 3. To get around this limitation, it is necessary to speak directly to the ATMEGA chip through the arduino interface.

A pin change interrupt is triggered by the state of a pin rising, falling, changing, or being low. To enable this feature on the arduino, we must identify the pins that we want to enable this feature on, and direct the chip what it needs to do when the interrupt is called.

Since arduino uses its own chip assignments, there is a map we will use to find the arrays and pin assignments that the ATMEGA uses to control these pins. The best source for this information is the datasheet for the ATMEGA 328 (Linked on webpage). Upon examination, we find:

Arduino Pin	Pin PCINT #	PCIE #	PCMSK#	Main PCINT #
D0-D7	16-23	2	2	2
D8-D13	0-5	0	0	0
A0-A5 (D14-D19)	8-13	1	1	1

We are familiar with the pins on the arduino, so the first column will be our starting place. Decide which pin you want to add an interrupt on. Usually it would be wise to avoid D2 and D3 since these are pre-set with external interrupts. Use the table to find the associated pin PCINT#. Notice that each set of pins is associated with a different number for PCIE, PCMSK, and PCINT. Each of these is associated with a separate pin change interrupt. They are labeled 0, 1, 2. This would indicate that there are three additional interrupts that can be handled by our Arduino board. Each PCINT is capable of being caused by any of the pins associated with it. This would indicate that you could have 6+ switches that all trigger the same interrupt. However, each of these pins would have to be attached to the interrupt routine separately.

To incorporate this information into your arduino code, we only need to consider the setup and ISR routines. An ISR routine is called whenever an interrupt is called and the parameter defines which PCINT triggers the ISR. For instance:

```
ISR ( PCINT#_void ) {
  "Do some stuff."
}
```

This format is valid for each of the three PCINT's. Note that the _void is always necessary and this is the main PCINT from the table above.

In the setup of your Arduino sketch, there are several lines of code that must be included to attach the interrupts. We must first tell the chip that we want to enable an interrupt:

```
PCICR |= (1<<PCIE#);
```

This changes the interrupt enable bit (PCIE) to "on" in the interrupt register (PCICR). 1 turns the interrupt on, 0 turns the interrupt off.

```
PCMSK# |= (1<<PCINT#);
```

This tells the pin change mask (PCMSK#) that it needs to watch for a change on pin (#). Use the table above to find the # for both the PCMSK and the PCINT (this will be the pin PCINT).

MCUCR = (1<<ISC0#) | (1<<ISC0#);

This line of code tells the chip which type of change triggers the interrupt. ISC0 is the value we are modifying. Inputting a 1 or a 0 at the # will determine the state. 0 0 triggers at any low level. 01 triggers at any change. 1 0 triggers at a falling edge. 1 1 triggers at a rising edge.

These are all of the commands that are necessary for implementing a switch as an interrupt on any available pin on the Arduino. There are other complication which will be left up to the end user, such as switch bounce etc..

Attached Below is an example code that uses two switches (default set to digital pins 7 and 8) to switch the LCD display between displaying “HIGH” and “LOW”

```
/*  How to set up additional interrupts in Arduino (ATMEL 328)
                    7/7/2010
                    Louis Van Blarigan

This code uses Pin Change interrupts to switch between two display states.
*/


/* Pin to interrupt map:
 * D0-D7 = PCINT 16-23 = PCIE2 = pcmsk2
 * D8-D13 = PCINT 0-5 = PCIE0 = pcmsk0
 * A0-A5 (D14-D19) = PCINT 8-13 = PCIE1 = pcmsk1
 */

#include <SoftwareSerial.h>       //(Include the needed libraries)
#define interrupt_pin 7           //Define Analog Pin (analog pins are 16-21)
#define interrupt_pin2 8            //Define Analog Pin (analog pins are 16-21)

volatile int bump = 0;

SoftwareSerial mySerial =  SoftwareSerial(13, 13);  //Change LCD Tx and Rx Pins to pins of our choosing

void setup() {
  PCICR |= (1<<PCIE2);
  PCMSK2 |= (1<<PCINT23);
  MCUCR = (1<<ISC01) | (1<<ISC00);

  PCICR |= (1<<PCIE0);
  PCMSK0 |= (1<<PCINT0);

  //LCD Configuration
  pinMode(13, OUTPUT);
  mySerial.begin(9600);
  mySerial.print("?f");          //Sends clear screen command to LCD

  pinMode(interrupt_pin, INPUT);       //Make pin an input
  pinMode(interrupt_pin2, INPUT);       //Make pin an input

  digitalWrite(interrupt_pin,HIGH);   //Enable pullup resistor on Analog Pin
  digitalWrite(interrupt_pin2,HIGH);   //Enable pullup resistor on Analog Pin
  bump = 0;
```

```
  interrupts();
}

void loop() {
  if (bump == 0) {
    mySerial.print("?f");
    mySerial.print("LOW");
  }
  else {
    mySerial.print("?f");
    mySerial.print("HIGH");
  }


}

ISR(PCINT2_vect) {
  bump = 1;
}
ISR(PCINT0_vect) {
  bump = 0;
}
```