Pin-change interrupts on Arduino

Arduino has only two hardware interrupts: INT0 and INT1. However, the AVR microcontroller can have an interrupton any pin change. The code for having more than two pin-interrupts is given here.

Hardware interrupts, also known as INT0 and INT1, call an interrupt service routine when something happens with one of the associated pins. The advantage is that Arduino has a simple set-up routine to connect the interrupt service routine to the event: attachInterrupt(). The disadvantage is that it only works with two specific pins: digital pin 2 and 3 on the Arduino board. This is shown in the first example.

Regular Arduino hardware interrupt INT0 and INT1

```
void setup()
{
  pinMode(13, OUTPUT);      // Pin 13 is output to which an LED is connected
  digitalWrite(13, LOW);    // Make pin 13 low, switch LED off
  pinMode(2, INPUT);        // Pin 2 is input to which a switch is connected = INT0
  pinMode(3, INPUT);        // Pin 3 is input to which a switch is connected = INT1
  attachInterrupt(0, blink1, RISING);
  attachInterrupt(1, blink2, RISING);
}

void loop() {
  /* Nothing to do: the program jumps automatically to Interrupt Service Routine "blink"
     in case of a hardware interrupt on Ardiono pin 2 = INT0 */
}


void blink1(){              // Interrupt service routine
  digitalWrite(13, HIGH);
}

void blink2(){              // Interrupt service routine
  digitalWrite(13, LOW);
}
```

Pin Change Interrupt on Arduino

If you need more pins, or other physical pins, there is a mechanism to generate an interrupt when any pin is changed in one of the ports of 8 bits. You don't know which single bit, but only which port. The example below generates an interrupt when one of the ADC0 to ADC5 pins (used as a digital input) is changed. In that case, the interrupt service routine ISR(PCINT1_vect) is called. In the routine you can figure out which of the specific pins within that port it has been.

First, the Pin Change Interrupt Enable flags have to be set in the PCICR register. These are bits PCIE0, PCIE1 and PCIE2 for the groups of pins PCINT7..0, PCINT14..8 and PCINT23..16 respectively. The individual pins can be enabled or dsiabled in the PCMSK0, PCMSK1 and PCMSK2 registers. In the Arduino circuit, in combination with the Atmel Atmega328 datasheet, you can figure out that the PCINT0 pin corresponds to pin 0 in port B (called PB0). This is pin 14 on the DIL version of the chip and digital pin 8 on the Arduino Uno. Another example is pin A0 on the Arduino board, which can be used as a digital input like in the example below. It is pin 23 on the Atmega328 (DIL version) which is called ADC0. The datasheet shows that it is PCINT8 which means it is part of PCINT14..8 and therefore enabled by the bit PCIE1 in PCICR.

```
void setup()
{
  Serial.begin(9600);
  Serial.println("Boe");
  InitialiseIO();
  InitialiseInterrupt();
}

void loop() {
  /* Nothing to do: the program jumps automatically to Interrupt Service Routine "blink"
   in case of a hardware interrupt  */
}

void InitialiseIO(){
  pinMode(A0, INPUT);       // Pin A0 is input to which a switch is connected
  digitalWrite(A0, HIGH);    // Configure internal pull-up resistor
  pinMode(A1, INPUT);       // Pin A1 is input to which a switch is connected
  digitalWrite(A1, HIGH);    // Configure internal pull-up resistor
  pinMode(A2, INPUT);       // Pin A2 is input to which a switch is connected
  digitalWrite(A2, HIGH);    // Configure internal pull-up resistor
}

void InitialiseInterrupt(){
  cli();                // switch interrupts off while messing with their settings
  PCICR =0x02;          // Enable PCINT1 interrupt
  PCMSK1 = 0b00000111;
  sei();                // turn interrupts back on
```

```
}

ISR(PCINT1_vect) {    // Interrupt service routine. Every single PCINT8..14 (=ADC0..5) change
            // will generate an interrupt: but this will always be the same interrupt routine
  if (digitalRead(A0)==0)  Serial.println("A0");
  if (digitalRead(A1)==0)  Serial.println("A1");
  if (digitalRead(A2)==0)  Serial.println("A2");
}
```

 7-10-2015