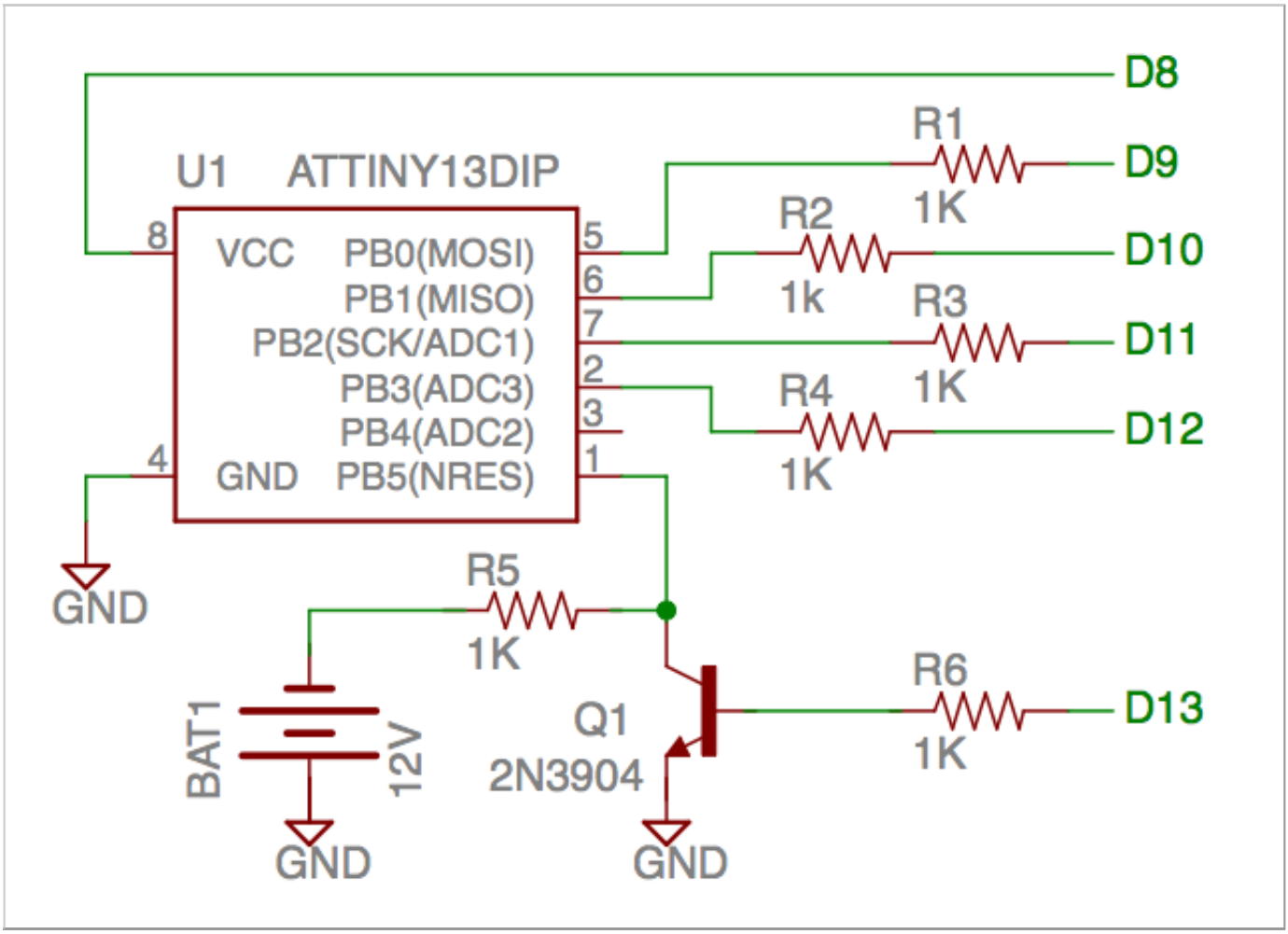
Wayne's Tinkering Page

ATTiny Fuse Reset

Note: for an updated version of this project see "ATTiny Fuse Reset with 12 Volt Charge Pump" which adds an integrated 12 volt power supply.

I like working with the Atmel's ATTiny series of microcontrollers and, from time to time, I'll mess up and "brick" one of them by setting a fuse value to something that disables further reprogramming of the chip with my AVR ISP MKII programmer. The way out of this impasse is to use a High Voltage programmer but, as I don't own one, I looked for another solution. My salvation came from a page by Paul Willoughby that describes how he built a HV Fuse Programmer using a few resistors, one 2N3904 NPN transistor, a 12 volt power source (I used 8 AAA batteries, but an A23 battery would be a more elegant solution), and an Arduino. After doing a bit of research to figure out how his code worked, I added a few improvements, such as the ability to read the device signature and use this to reset the fuses to the correct factory default for each device. The result is shown below.

If you want to use this code, start by building Paul's simple circuit on an Arduino Protoboard. Here's my version of the needed circuit set to reprogram an ATTiny13 (or another 8 pin ATTiny, such as as the ATTiny25-45-85 series). The same design will also work with the 14 pin ATTiny24-44-84 series, but the pin connections to the device will be different.



Note: the battery polarity is reversed the diagram above. Positive goes to R5.

Next, you can add 8 batteries, or connect up a lab power supply to supply the 12 volts needed. Next, load the sketch (there's a download link at the bottom of the page), plug in an ATTiny device that needs its fuses fixed (see list of supported devices in the source code) and open the Serial Monitor in the Arduino IDE and select 19200 baud as the serial rate. Then, send a character to the Arduino using the Serial Monitor's "send" button to trigger the fuse reset cycle. The code will print out the device signature and before and after values of the fuses and, that it. You should now have a functional ATTiny!

```
// AVR High-voltage Serial Fuse Reprogrammer
// Adapted from code and design by Paul Willoughby 03/20/2010
//   http://www.rickety.us/2010/03/arduino-avr-high-voltage-serial-programmer/
//
// Fuse Calc:
//   http://www.engbedded.com/fusecalc/

#define  RST     13    // Output to level shifter for !RESET from transistor
#define  SCI     12    // Target Clock Input
#define  SDO     11    // Target Data Output
#define  SII     10    // Target Instruction Input
#define  SDI      9    // Target Data Input
#define  VCC      8    // Target VCC

#define  HFUSE  0x747C
#define  LFUSE  0x646C
#define  EFUSE  0x666E

// Define ATTiny series signatures
#define  ATTINY13   0x9007  // L: 0x6A, H: 0xFF             8 pin
#define  ATTINY24   0x910B  // L: 0x62, H: 0xDF, E: 0xFF   14 pin
#define  ATTINY25   0x9108  // L: 0x62, H: 0xDF, E: 0xFF    8 pin
#define  ATTINY44   0x9207  // L: 0x62, H: 0xDF, E: 0xFFF  14 pin
#define  ATTINY45   0x9206  // L: 0x62, H: 0xDF, E: 0xFF    8 pin
#define  ATTINY84   0x930C  // L: 0x62, H: 0xDF, E: 0xFFF  14 pin
#define  ATTINY85   0x930B  // L: 0x62, H: 0xDF, E: 0xFF    8 pin

void setup() {
  pinMode(VCC, OUTPUT);
  pinMode(RST, OUTPUT);
  pinMode(SDI, OUTPUT);
  pinMode(SII, OUTPUT);
  pinMode(SCI, OUTPUT);
  pinMode(SDO, OUTPUT);     // Configured as input when in programming mode
  digitalWrite(RST, HIGH);  // Level shifter is inverting, this shuts off 12V
  Serial.begin(19200);
}

void loop() {
   if (Serial.available() > 0) {
    Serial.read();
    pinMode(SDO, OUTPUT);     // Set SDO to output
    digitalWrite(SDI, LOW);
    digitalWrite(SII, LOW);
    digitalWrite(SDO, LOW);
    digitalWrite(RST, HIGH);  // 12v Off
    digitalWrite(VCC, HIGH);  // Vcc On
    delayMicroseconds(20);
    digitalWrite(RST, LOW);   // 12v On
    delayMicroseconds(10);
    pinMode(SDO, INPUT);      // Set SDO to input
    delayMicroseconds(300);
    unsigned int sig = readSignature();
    Serial.print("Signature is: ");
    Serial.println(sig, HEX);
    readFuses();
    if (sig == ATTINY13) {
```

```
      writeFuse(LFUSE, 0x6A);
      writeFuse(HFUSE, 0xFF);
    } else if (sig == ATTINY24 || sig == ATTINY44 || sig == ATTINY84 ||
              sig == ATTINY25 || sig == ATTINY45 || sig == ATTINY85) {
      writeFuse(LFUSE, 0x62);
      writeFuse(HFUSE, 0xDF);
      writeFuse(EFUSE, 0xFF);
    }
    readFuses();
    digitalWrite(SCI, LOW);
    digitalWrite(VCC, LOW);    // Vcc Off
    digitalWrite(RST, HIGH);   // 12v Off
  }
}

byte shiftOut (byte val1, byte val2) {
  int inBits = 0;
  //Wait until SDO goes high
  while (!digitalRead(SDO))
    ;
  unsigned int dout = (unsigned int) val1 << 2;
  unsigned int iout = (unsigned int) val2 << 2;
  for (int ii = 10; ii >= 0; ii--)  {
    digitalWrite(SDI, !!(dout & (1 << ii)));
    digitalWrite(SII, !!(iout & (1 << ii)));
    inBits <<= 1;
    inBits |= digitalRead(SDO);
    digitalWrite(SCI, HIGH);
    digitalWrite(SCI, LOW);
  }
  return inBits >> 2;
}

void writeFuse (unsigned int fuse, byte val) {
  shiftOut(0x40, 0x4C);
  shiftOut( val, 0x2C);
  shiftOut(0x00, (byte) (fuse >> 8));
  shiftOut(0x00, (byte) fuse);
}

void readFuses () {
  byte val;
        shiftOut(0x04, 0x4C);  // LFuse
        shiftOut(0x00, 0x68);
  val = shiftOut(0x00, 0x6C);
  Serial.print("LFuse: ");
  Serial.print(val, HEX);
        shiftOut(0x04, 0x4C);  // HFuse
        shiftOut(0x00, 0x7A);
  val = shiftOut(0x00, 0x7E);
  Serial.print(", HFuse: ");
  Serial.print(val, HEX);
        shiftOut(0x04, 0x4C);  // EFuse
        shiftOut(0x00, 0x6A);
  val = shiftOut(0x00, 0x6E);
  Serial.print(", EFuse: ");
  Serial.println(val, HEX);
}

unsigned int readSignature () {
  unsigned int sig = 0;
  byte val;
  for (int ii = 1; ii < 3; ii++) {
          shiftOut(0x08, 0x4C);
          shiftOut(  ii, 0x0C);
          shiftOut(0x00, 0x68);
```

```
    val = shiftOut(0x00, 0x6C);
    sig = (sig << 8) + val;
  }
  return sig;
}
```

ATTinyFuseReset.pde (4k) Wayne Holder, Nov 27, 2010, 9:51 PM v.3