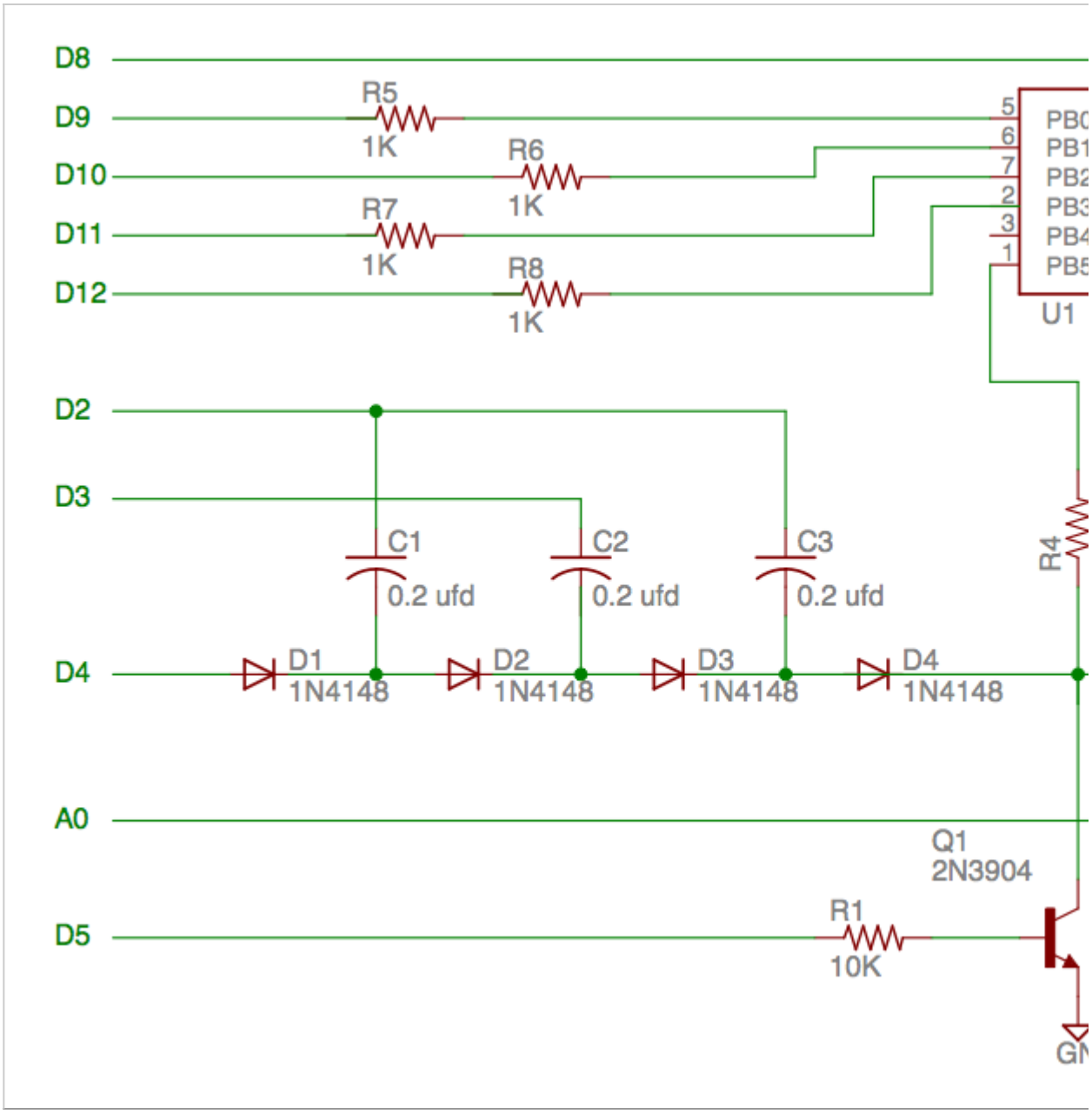
Wayne's Tinkering Page

ATTiny Fuse Reset with 12 Volt Charge Pump

This page shows how to build a more advanced version of the ATTiny Fuse Reset tool that uses a Charge Pump controlled by an Arduino interrupt routine to generate the 12 volts needed to program the fuses on the ATTiny chips. The details on how how to reset the fuses and how to build a charge pump are described on the pages linked above, so I won't repeat those details here. The schematic of the circuit needed is shown below:



I'm working on a PC Board that implements a simple Arduino Shield for this circuit. I plan to make this design available via SparkFun's BatchPCB service, but I'll post more details

when that design is ready. When the PCB is ready, here are some suggested components you can use to build this circuit:

Description	Quan	Mouser Part #	Price ea	Extd
1N4148 Diode	4	771-1N4148,133	$.02	$.08
0.22 uFd Ceramic Cap	3	810-FK14X7R1H224K	$.26	$.78
2.2 ufd Ceramic Cap	1	810-FK14Y5V1E225Z	$.25	$.25
10K 1/8 Watt resistor	1	299-10K-RC	$.09	$.09
1K 1/8 Watt resistor	5	299-1K-RC	$.09	$.45
510K 1/8 Watt resistor	1	299-510K-RC	$.09	$.09
100K 1/8 Watt resistor	1	299-100K-RC	$.09	$.09
2N3904 Transistor	1	512-2N3904TAR	$.07	$.07
8 pin socket	1	649-DILB8P223TLF	$.12	$.12
14 pin socket	1	649-DILB14P-223TLF	$.17	$.17
			Total	$2.19

The Arduino program needed to drive this circuit is shown next (also available for download as an Arduino project at the bottom of the page.)

```
#include <TimerOne.h>
// AVR High-voltage Serial Fuse Reprogrammer with 12 Volt Charge Pump
// 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  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: 0xFF   14 pin
#define  ATTINY45   0x9206  // L: 0x62, H: 0xDF, E: 0xFF    8 pin
#define  ATTINY84   0x930C  // L: 0x62, H: 0xDF, E: 0xFF   14 pin
#define  ATTINY85   0x930B  // L: 0x62, H: 0xDF, E: 0xFF    8 pin

// Define Direct I/O pins for Charge Pump
#define P1  0x04  // Pin D2
#define P2  0x08  // Pin D3
#define PWR 0x10  // Pin D4
#define GND 0x20  // Pin D5
#define REF 404   // 12 volt reference

// Variables used by Charge pump
volatile char phase = 0;
volatile char onOff = 0;
volatile char pwrOn = 0;
```

```
void ticker () {
  if (onOff) {
    DDRD = P1 | P2 | PWR | GND;
    int volts = analogRead(A0);
    if (volts < REF) {
      if (phase) {
        PORTD = P1 | PWR;
      } else {
        PORTD = P2 | PWR;
      }
      phase ^= 1;
    } else {
      pwrOn = 1;
    }
  } else {
    pwrOn = 0;
    DDRD = GND;
    PORTD = GND;
  }
}

void setup() {
  pinMode(VCC, OUTPUT);
  pinMode(SDI, OUTPUT);
  pinMode(SII, OUTPUT);
  pinMode(SCI, OUTPUT);
  pinMode(SDO, OUTPUT);     // Configured as input when in programming mode
  Serial.begin(57600);
  // Setup timer interrupt for  charge pump
  analogReference(DEFAULT);
  Timer1.initialize(500);
  Timer1.attachInterrupt(ticker);
}

void loop() {
   if (Serial.available() > 0) {
    Serial.read();
    pinMode(SDO, OUTPUT);     // Set SDO to output
    digitalWrite(SDI, LOW);
    digitalWrite(SII, LOW);
    digitalWrite(SDO, LOW);
    onOff = 0;                // 12v Off
    digitalWrite(VCC, HIGH);  // Vcc On
    delayMicroseconds(20);
    onOff = 1;                // 12v On
    while (pwrOn == 0)
      ;
    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
    onOff = 0;                 // 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;
  }
```

ATTinyFuseReset2.pde (4k) Wayne Holder, Nov 28, 2010, 11:... v.3