

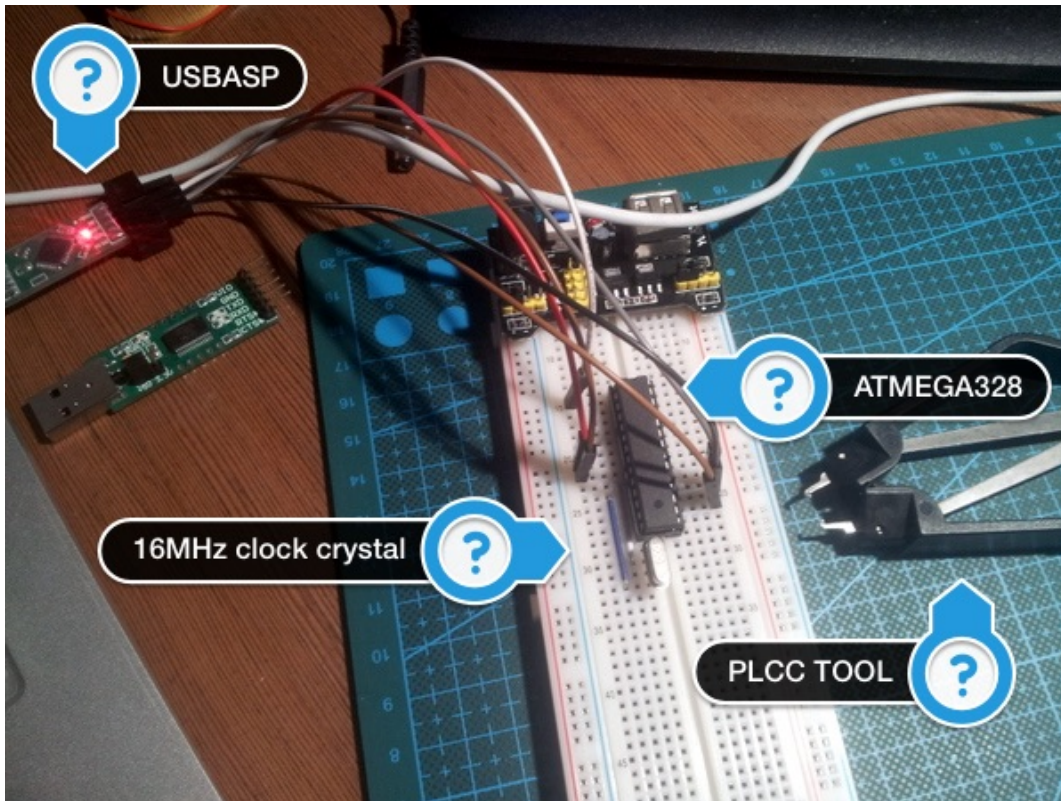
# voltage display

2014-12-07 XnnYygn

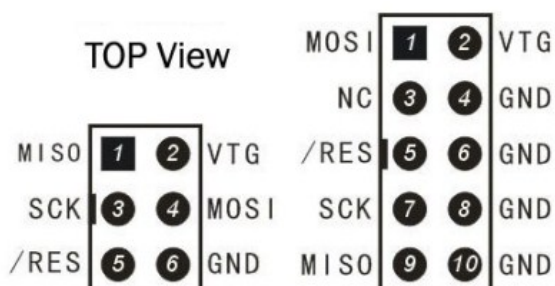
ここまでは、一般的な腕があるんでしょう。今回Arudino抜きで直接AVRの集成电路を利用しました。

そう、元々電子工作と言うのはArduinioだけではないでしょう。Arduinoのサイトでも自分がAVR開発ボードを言いました。(http://playground.arduino.cc/Main/AVR)

Arduino UNOの中心はATMega328P-PUです。今回使ったのはDIPの方です。(SMDの方に比べて)



USBASPを使って、BOOTLOADERをArudino IDEでアップロードしてみました。clock crystalは必要です。ないならアップロードできません。



これはUSBASPのPINです（右）。使ったのは左と同じ六のPINしかありません。

## Atmega168 Pin Mapping

### Arduino function

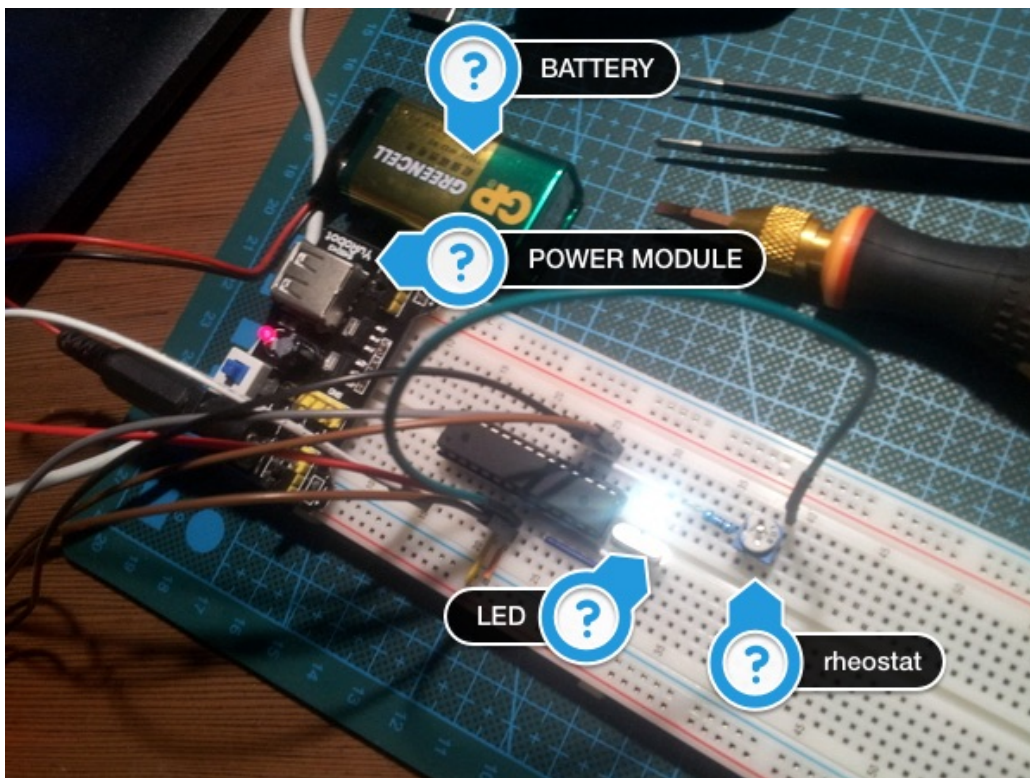
reset	(PCINT14/RESET) PC6	1
digital pin 0 (RX)	(PCINT16/RXD) PD0	2
digital pin 1 (TX)	(PCINT17/TXD) PD1	3
digital pin 2	(PCINT18/INT0) PD2	4
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5
digital pin 4	(PCINT20/XCK/T0) PD4	6
VCC	VCC	7
GND	GND	8
crystal	(PCINT6/XTAL1/TOSC1) PB6	9
crystal	(PCINT7/XTAL2/TOSC2) PB7	10
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12
digital pin 7	(PCINT23/AIN1) PD7	13
digital pin 8	(PCINT0/CLKO/ICP1) PB0	14

### Arduino function

28	PC5 (ADC5/SCL/PCINT13)	analog input 5
27	PC4 (ADC4/SDA/PCINT12)	analog input 4
26	PC3 (ADC3/PCINT11)	analog input 3
25	PC2 (ADC2/PCINT10)	analog input 2
24	PC1 (ADC1/PCINT9)	analog input 1
23	PC0 (ADC0/PCINT8)	analog input 0
22	GND	GND
21	AREF	analog reference
20	AVCC	VCC
19	PB5 (SCK/PCINT5)	digital pin 13
18	PB4 (MISO/PCINT4)	digital pin 12
17	PB3 (MOSI/OC2A/PCINT3)	digital pin 11 (PWM)
16	PB2 (SS/OC1B/PCINT2)	digital pin 10 (PWM)
15	PB1 (OC1A/PCINT1)	digital pin 9 (PWM)

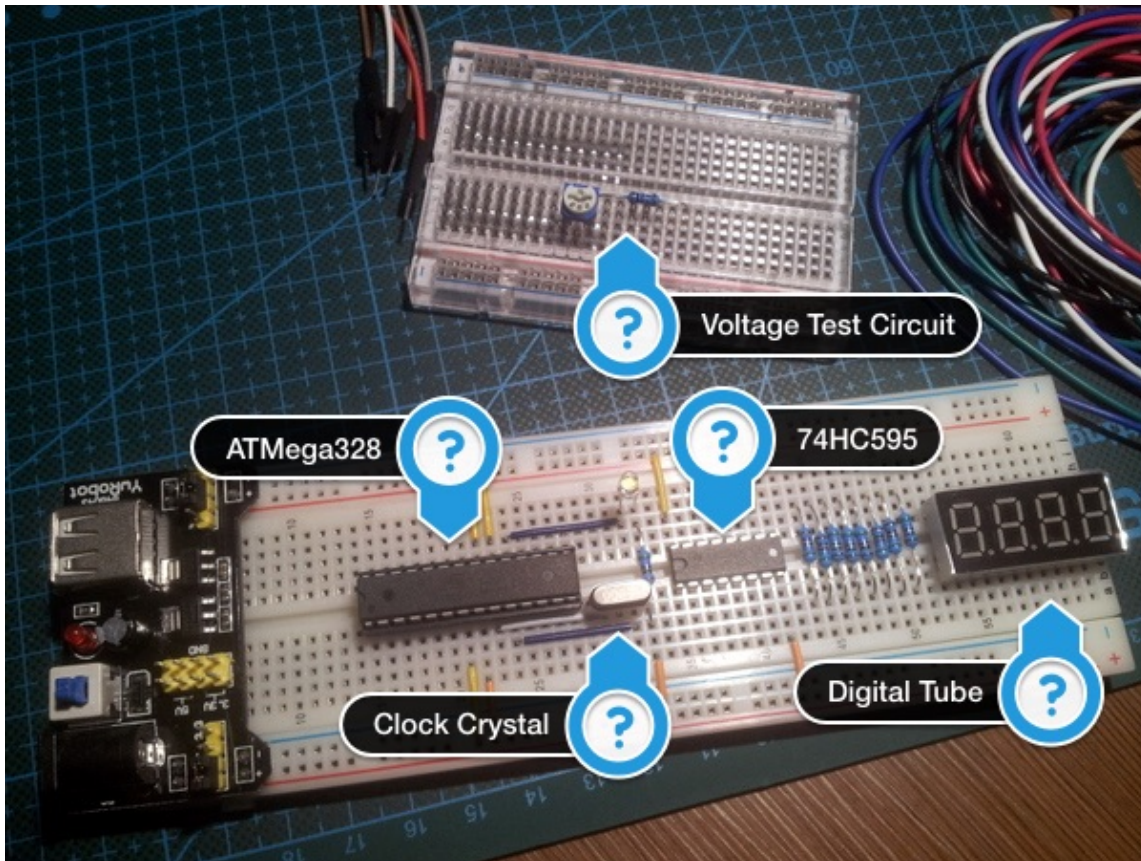
Digital Pins 11, 12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17, 18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

これはATMega168の図ですが、ATMega328とほぼ同じです。  
 BOOTLOADERをアップロードしたら、後でFT232RLなどでアップロードするんですか。いいえ、今回は全部直接USBASPでアップロードします。もしFT232RLでアップロードすると、RESETボタンは必要です。「アップロードと同時にRESETボタンを押す」とかちょっと面倒です。これもArudinoが流行ってるわけの一つでしょう。  
**D13のLEDのテスト。**なんか眩しい。

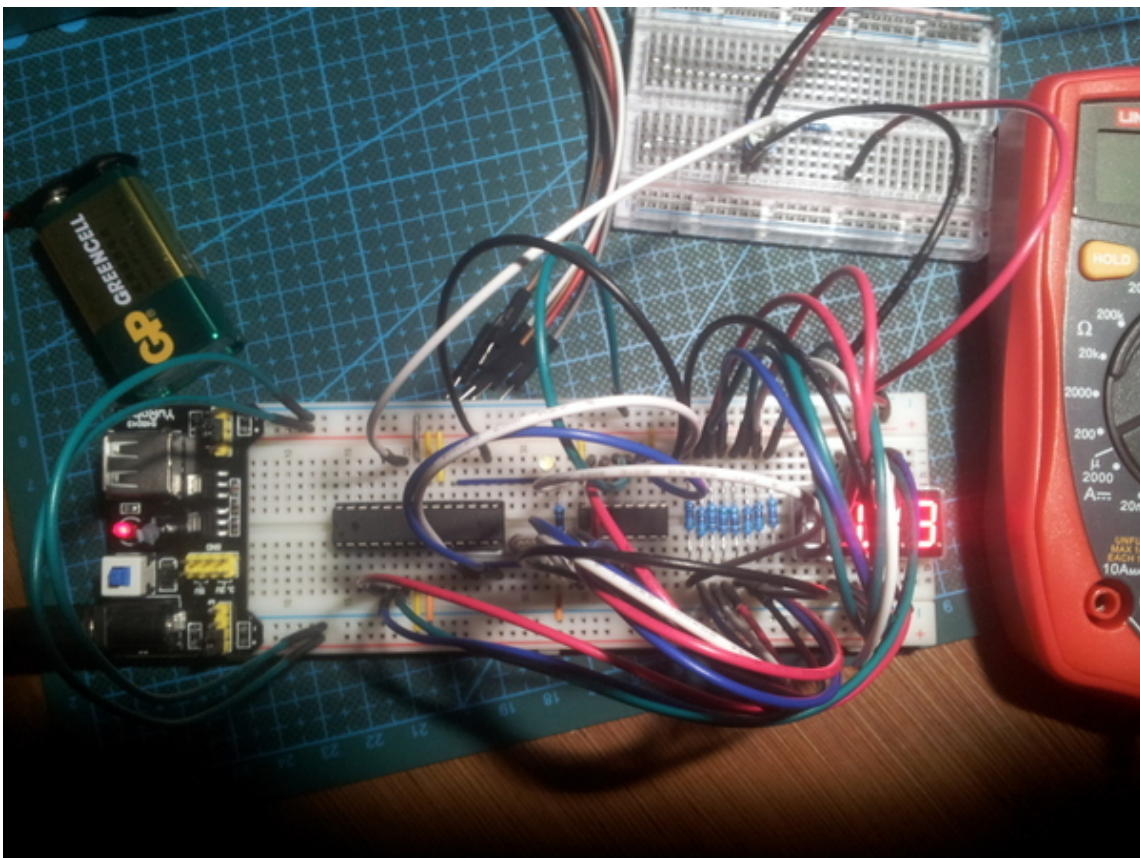




これから作品を作る時間です。

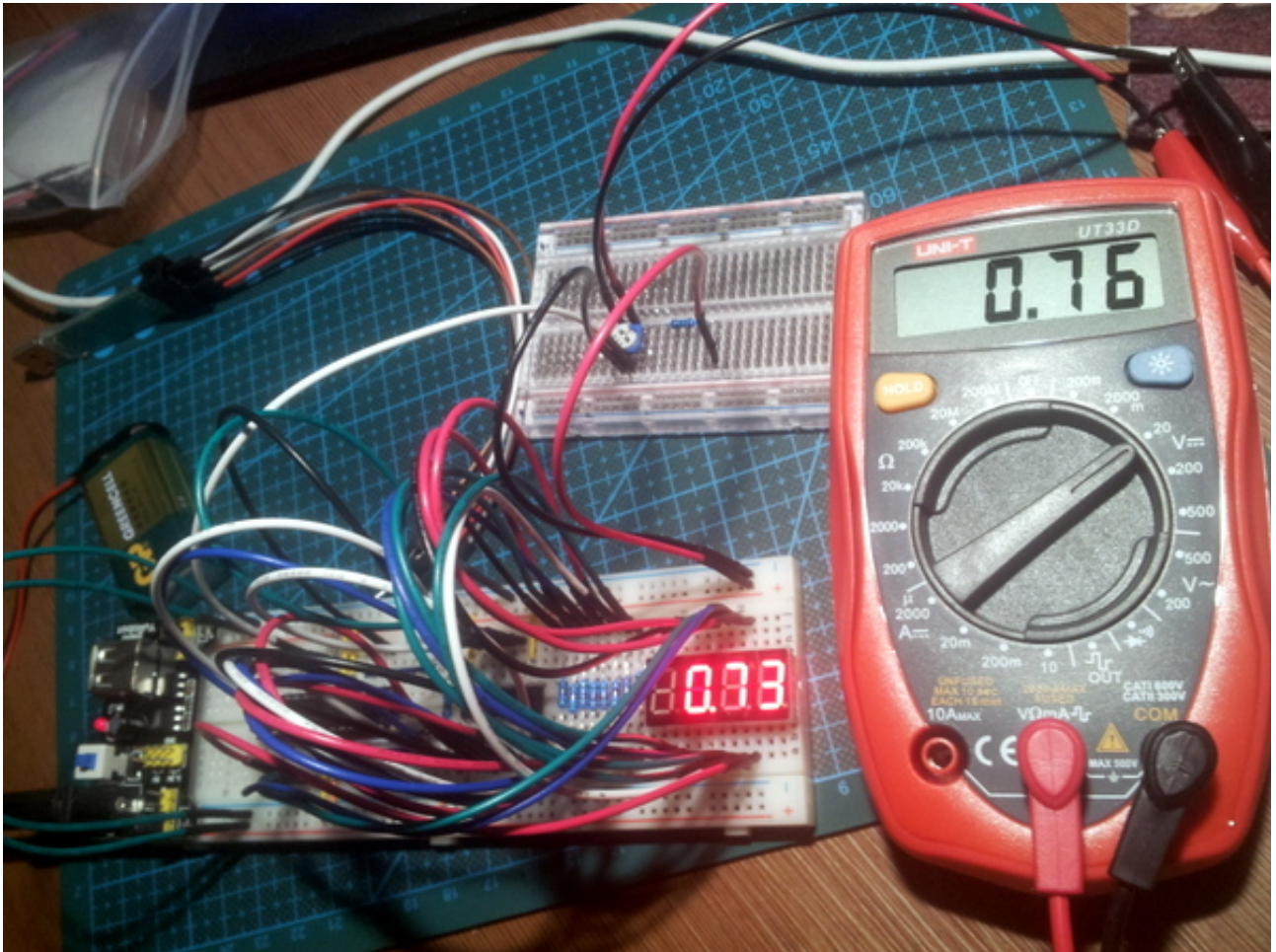


これは主な部品です。次はラインつきの写真です。

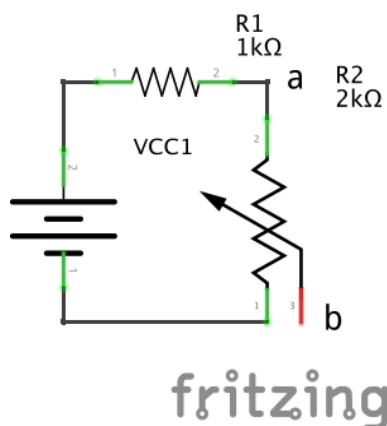




製品です。



まず上の電圧の回路を紹介します。



測るのはaとbの間の電圧です。

次は数字を表す部分です。普通の8-segment LED displayは八のPINがあります。でもこの部品は4つの数字はあるのに、ただ12のPINがあります。直接全部のLEDをコントロールするのはできません、4 x 8, 32のPINが必要です。でも何で12なの？ 実は8 + 4で計算されました。4つのPINはどの数字をコントロールし、残った8つのPINはこの数字のLEDをコントロールします。実際12のPINそんなに多いではないが、毎回12のPINをコントロールのは面倒なので、74HC595という集成电路を使いました。でも、数字を選ぶ4つのPINは直接コントロールします。次はどのように表示できます。毎回一つの数字を選んで、この数字をコントロールします。でも、毎回4つの数字を表すんでしょう。だから、ちょっとネタみたいなものの出番です。

確かに毎回4つの数字を表して、そしてこの作品は実際毎回一つの数字しかコントロールできません。では、4つの数字を別々で表します。これ、すぐばれるんでしょう。でも、もしとても早いならどうでしょう。実際別々の数字を表すのは見られません。コードはこちらです。

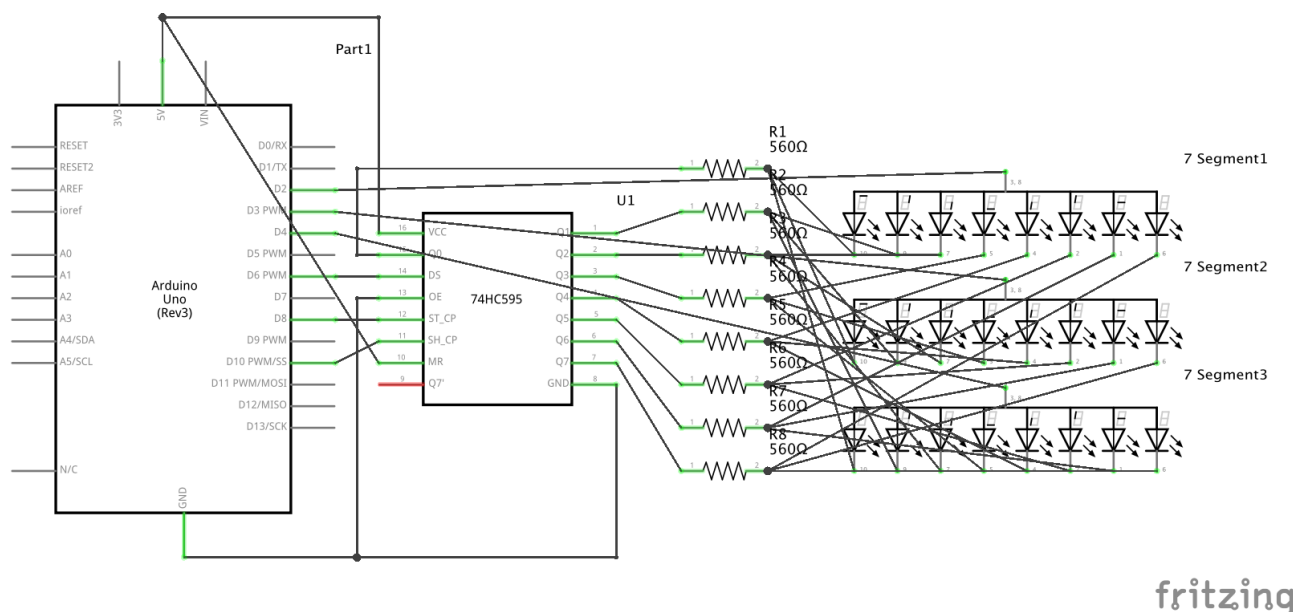
```

28 void loop() {
29   // read voltage
30   int analogValue = analogRead(0);
31   float voltage = 0.0048 * analogValue;
32   byte numbers = voltage * 100;
33
34   for(byte i = 0; i < 20; i++) {
35     displayNumber(numbers / 100, true, PIN_D_2);
36     displayNumber((numbers % 100) / 10, false, PIN_D_3);
37     displayNumber(numbers % 10, false, PIN_D_4);
38   }
39 }
40
41 void displayNumber(byte n, bool withPoint, byte pin) {
42   digitalWrite(pin, LOW);
43   digitalWrite(PIN_LATCH, LOW);
44   shiftOut(PIN_DATA, PIN_CLOCK, LSBFIRST, digits[n] + (withPoint ? 1 : 0));
45   digitalWrite(PIN_LATCH, HIGH);
46   delay(10);
47   digitalWrite(pin, HIGH);
48 }

```

ここのdelay(10)は調整した結果です。うえのLOOPを気にしないでください。それはただ電圧の読むのはスピードを遅くされたいです。  
digitsは計算した結果です。

```
byte digits[] = {252, 96, 218, 242, 102, 182, 190, 224, 246, 238};
```



これは電路です。

実際AVRの集成电路でものを作ると直接Arudinoで作るは別に大きな違うところはないと思います。でもArudinoはほかの機能をたとえば安全措置はあります。直接より人を安心させるんでしょう。最初はArudinoでテストして、後で直接ATMega328など集成电路にアップロードするとはいいと思います。これも一般的に研究から実際応用する流れでしょう。