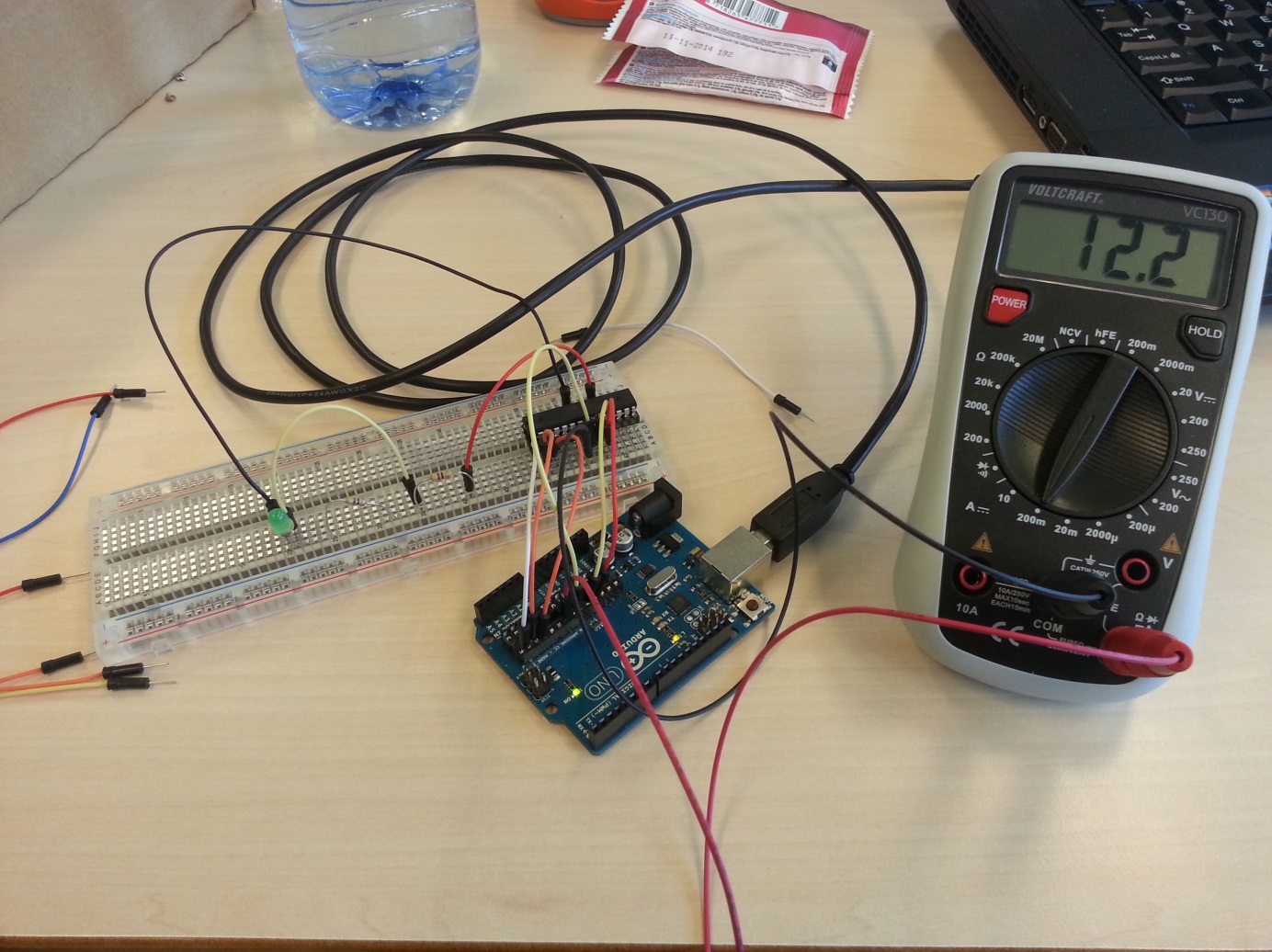
# LowPower

Kasper Rijksen

Joery van den Hoff

Wij hebben de Atmega328P van de arduino gehaald en geinstalleerd op een breadboard. Wij hebben een led aangesloten op poort 13 met een weerstand van 470 Ω. Wij wilden eerst twee weerstanden van 470 Ω in serie zetten om een weerstand van ongeveer 1kΩ te krijgen. Maar Tommy vertelde ons dat een weerstand van 470 Ω ook ruim voldoende was. Hieronder een foto van de setup:

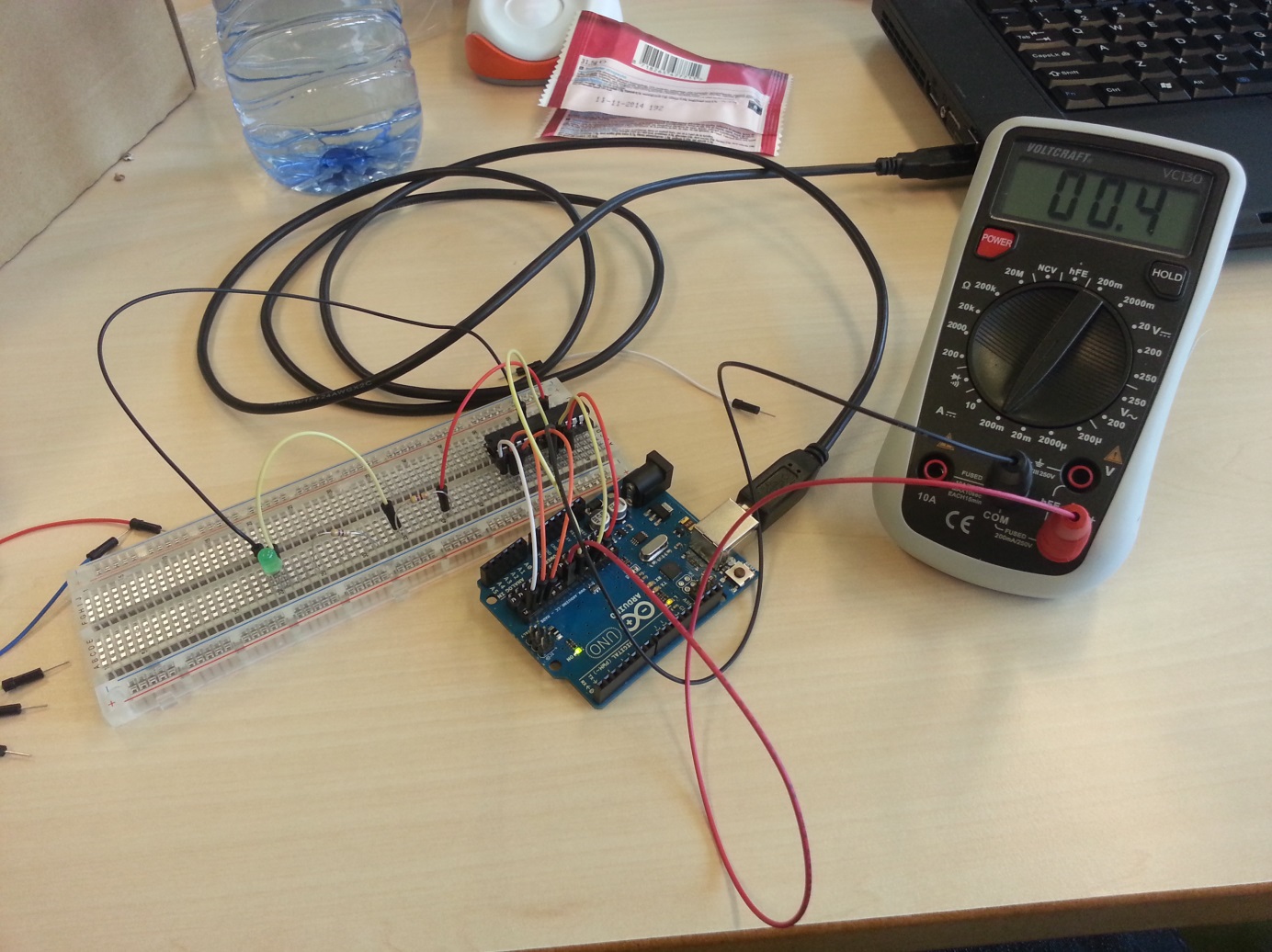


Vervolgens hebben wij de stroom gemeten die de arduino verbruikt als de led aan of uit is. Al onze metingen staan in het bestand ‘**Current consumption.xlsx**’.

Vervolgens hebben wij een hoop dingen uitgezet, o.a. timers en ADC. Hiervoor moesten wij een eigen delay methode schrijven (timer die delay() gebruikt, hebben wij ook uit gezet). Wij hebben met behulp van de logic analyzer bepaalt hoe groot de for loop counter moest zijn. Daarna hebben we weer een meting gedaan en er blijkt dat de arduino een hoop minder stroom verbruikt, ongeveer 18%.

Vervolgens hebben we de CLK van de arduino aangepast en gemeten per divider value van 1 – 256. 256 betekent dat de CLK ipv 16Mhz, 62,5KHz is (dus delen door 256). Voor alle waarden hebben we de stroom consumptie gemeten met de led aan en uit. Hieronder is een grafiek van de uitkomsten. Deze staat ook in ‘Current consumption.xlsx’.

De laatste meting is het stroom verbruik als de arduino in ‘Power Down’ modus is. Dit is heel erg laag (0,4 mA). Vervolgens haalt de watch dog ons uit ‘Power Down’ mode na 4 seconden. Hieronder een foto genomen toen de arduino in ‘Power Down’ modus was:



## opdracht5.ino

In setup() zetten wij een hoop dingen uit om stroom te besparen:

* 0 naar ADEN: zet de ADC uit.
* 1 naar ACD: Analog Comparator gaat uit. (gebruikt in ADC)
* 1 naar ADC0-5D: zet de digitale input uit voor ADC0-5 pinnen
* 1 naar AIN1D en AIN0D: zet de digitale input uit voor AIN1-0 pinnen
* 1 naar alles in PRR: zet timers, watch dog, seriële communicatie en ADC uit.

In loop zetten wij de CLKPR op 128. Dit geeft ons de mogelijkheid om de CLKPS0-3 aan te passen. Dat doen wij iedere loop en hogen de prescaler één op. (tot 8 en dan gaan we daarna terug naar 0).

Ook in loop laten wij LED13 knipperen, met een eigen gemaakte delay() methode. Deze gebruikt de huidige prescale instelling om lang genoeg te “busy wait-en” om 3 seconden te wachten. Dit geeft ons genoeg tijd om de waardes duidelijk af te lezen.

## opdracht5\_wtd.ino

In setup() zetten wij de watchdog aan. Dit moet met een rare volgorde: eerst zetten we enable op 1 (WDCE). En dan binnen 4 clock cycles zetten we WDCE op 0 en op het zelfde moment zetten we WDP3 op 1. Dit zorgt ervoor dat de watchdog 4,0 seconden wacht voordat hij een interrupt en reset doet.

In loop() doen wij de watchdog “aaien” met wdt\_reset() (dit verwijst naar een assembly instructie en komt uit avr/wdt.h). Dit zorgt ervoor dat de timer wordt gereset, en de watchdog begint weer op 0,0 s te tellen.

Daarna doen we LED13 weer knipperen. Dit herhaalt zich 5 keer, dan gaat de arduino in ‘Power Down’ modus. Hiervoor zetten we SM1 en SE in SMCR. En we roepen sleep\_cpu() aan uit avr/sleep.h.

Als de arduino in ‘Power Down’ modus is dan wordt loop() dus niet meer uitgevoerd. Dat betekent dat de watchdog na 4,0 seconden gaat “blaffen” en roept de ISR(WDT\_vect) aan en wordt het systeem gereset.