Zadanie práce

Ako finálny projekt z predmetu sme si zvolili generátor náhodných číseľ v uživateľom nastavitelnom rozsahu medzi 0-255. Nastavenie prebiehalo pomocou reostatu a AADC prevodníku, ktorý z analógového napäria vyrobil číslo od 0 do 255. To bolo použité v softwarovom generátore náhodných čísel. Zobrazovanie horného limitu a náhodného čísla bolo pomocou zelených a červených LED svetiel. Zobrazenie bolo kvôli jednoduchosti v dvojkovej sústave.

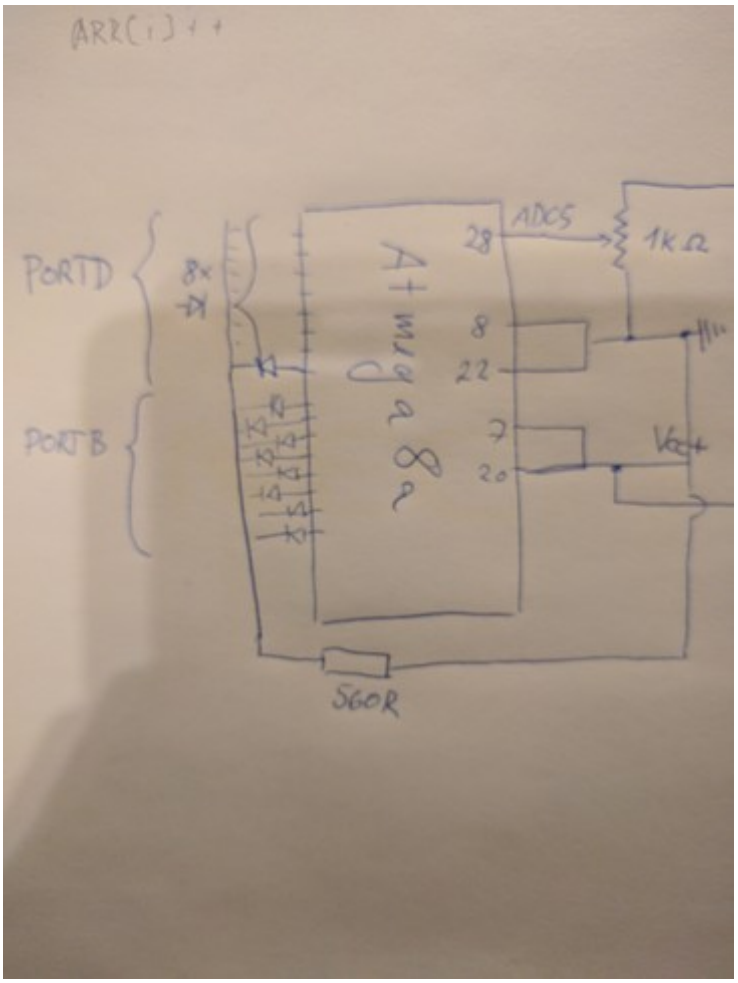
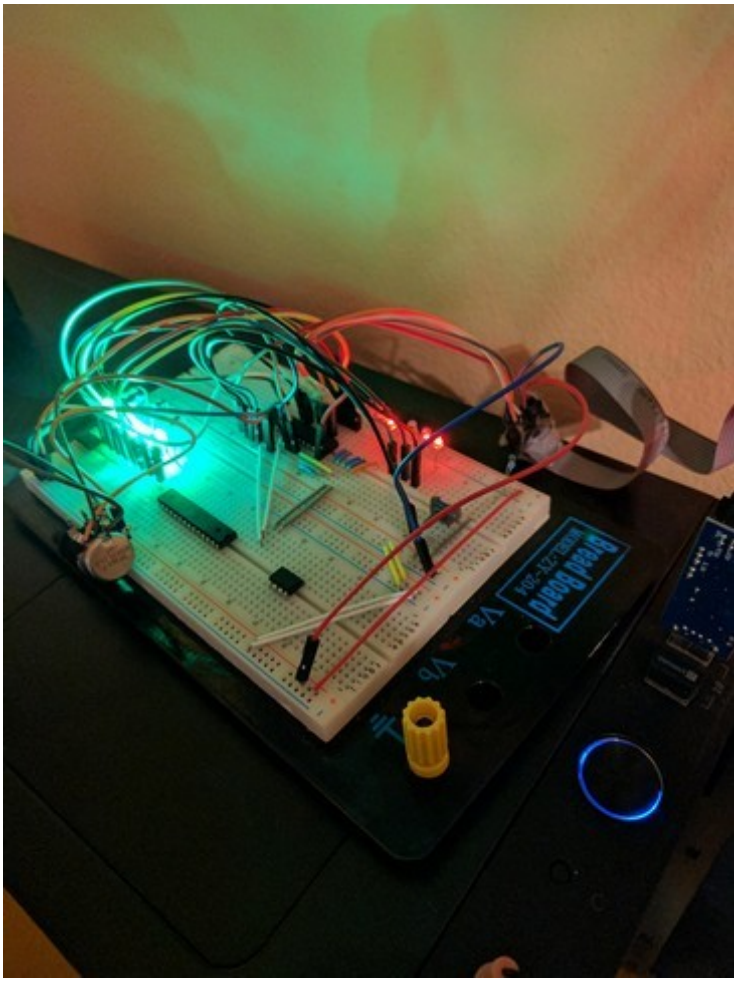


Schéma zapojenia



Obrázok finálneho projektu

O microcontroleroch

Microcontrolery (skrátene uC) sú výborným pomocníkom v elektronike, kedže ich účel nie je vopred definovaný, ale funkciu si na rozdiel od väčšiny “chrobáčikov” určuje sám technik tým, že ich naprogramuje v C. Spoločnosti, ktoré tieto uC vyrábajú, je niekoľko, a medzi najznámešie patria Atmel a Mikrochip, ktoré majú uC *avr* a *PIC*. Medzi nimi pre začiatočníkov nie je veľký rozdiel, ale keďže sa

nám podarilo zohnať programátor pre avr uC skôr ako pre PIC, rozhodli sme sa pre riešenie od firmy ATMEL.

Architektúra uC

Atmel ponúka široké spektrum rôznych mikročipov, ktoré sa líšia hlavne veľkosťou flash pamäte, počtom nožičiek, typom architektúry (väčšina je 8bit) a napájacím napätím VCC. Každý mikroprocessor sa dá kúpiť vo viacerych formách DIP, TQFP a QFN. My sme väčinou pracovali s Atmega8a-PU, kde *a* znamená refresh čipu, ktorý zmenšil spotrebu čipu optroti pôvodnej verzií.

Každý uC sa skladá z flash pamäte, kde je skompilovaný C program v hex zápise, EEPROM, ktorá je podobne ako flash *non volatile*, ale má väčší počet prepísaní a je pomalšia (využíva sa hlavne na ukladanie premenných, ktore je potrebné zachovať v prípade straty napätia) a taktiež má aj SRAM, ktorá funguje ako klasická RAM v PC.

Fuse sú špeciálne tri byty, ktoré určujú správanie procesora, ako napríklad čo sa má diať v prípade nedostatočného napätia (brown-out), prípadne zablokovanie komunikácie s programátorom, čo slúži na zablokovanie čítania flash pamäte s zdrojovým kom na produkčných čipoch. Taktiež sa tu dá nastaviť, ktorý oscilátor má uC použiť. Dá sa vybrať z viacerých, napríklad externý oscilátor pripojený na pin označený XTAL1 a XTAL2. Najednoduchšie je použit online/Android aplikáciu, ktorej výstup, po zvolení vlastností uC, je príkaz pre avrdude.

F_CPU je dôležitý parameter každého uC - je to frekvencia, na ktorej operuje. Pri samotnom programovaní je doležité si pamätať, že pri komunikácií medzi programátorom a uC nie je možné použiť frekvenciu väčšiu ako ¼ F_CPU. Naštastie vačšina nových programátorov je schopná upraviť svoju frekvenciu podľa -B možnosti pre avrdude (`-B 17kHz`)

Vývojové prostredie

Pre Windows je spravené IDE priamo od Atmelu, ktoré má výborné debugovacie prvky, kde sa dajú sledovať hodnoty jednotlivých pinov a registrov, programovanie fuse bitov cez checkbox-i a veľa iného. Nevýhodou však je nemožnosť programovať cez lacnejšie programátory, ako je napríklad usbasp. Taktiež je to v podstate Visual Studio, čo nemusí každému vyhovovať.

Pri kompilovaní je najednoduchšie použiť Makefile, ktorý sme získali z programu WinAvr pre Windows. Treba len upraviť hodnoty F-CPU a MMCU a následne použiť príkaz `make`.

Na používanie v Linuxe je potrebné mať stiahnuté tieto package:

```
gcc-avr binutils-avr gdb-avr avr-libc avrdude
```

Bitové operácie v C

Bez bitových operáci sa človek nedostane o moc ďalej ako Hello World program pre uC (čo je blikajúca LED) a aj tam je to s nimi nápomocné. Tutoriálov je na internete dosť a dá sa to naučiť aj za pochodu.

ADC

Analog Digital Converter, ako názov napovedá, je časť uC, ktorý je zodpovedný za prekladanie alanógového signálu na digitálny. V prípade, že sa používa, treba ho napájať špeciálne na pin AVCC. Štandardne je výsledok 10-bitové číslo od 0 – 1023, pričom 0 značí 0V a 1023 je VCC prípadne AREF (nastaviteľné cez registre). Pre naše potreby stačí aj 8-bitové rozlíšenie a navyše sa to aj ľahšie zobrazuje. Na to nám treba prečítať register ADCW, z ktorého získame 10-bitové číslo a vydeliť ho 4, ale jednoduchšie riešenie je nastaviť v registri ADMUX bit ADLAR na *high* (čiže jednotku).

ADC vie najlepšie pracovať s frekvenciu medzi 50-200kHz, preto je potrebné nastaviť pomocou bitov ADPSn v registri ADCSRA delič hlavnej frekvencie CPU, aby bol v požadovanom intervale.

Ďalšou časťou, ktorú chceme nastaviť, je ako často má prebiehať konverzia (kontinuálne/iba raz a potom ju treba znova spustiť). *Freeruning* aj *single-conversion* mód vieme použ s/bez interuptov.

System Interupts

Na prerušenie akejkoľvek činnosti (napríklad `_delay_ms(1000)` je vhodné použiť interupty. Sú na to dva spôsoby. Prvý, ktorý využíva dva piny INT0 a INT1 je sofistikovanejší, keďže sa dá nastaviť pri akej zmene napätia sa má zavolať interupt (napríklad iba pri poklese napätia). Druhý menej sofistikovaný spôsob je použiť port PCINTx. Väčšina modených uC ich má na každom pine (okrem napájania), čiže pre Atmega328 ich je 24. Nastavenie, ktoré piny majú zavolať interupt sa nastavuje cez PCMSK0-2. Problém však je, že piny volajú interupt “naviazaný” na svoj PCMSKx, čiže ak máme viac aktívnych pinov v tom istom PCMSKx nevieme povedať, ktorý interupt spustil bez kontroly registrov.

Ďalšou možnosť ako pracovať s interuptami je použiť Timers. Existujú dva 8-bitový a 16-bitový, každému sa dá nastaviť s akou frekvenciou (vzhľadom na CPU) má pracovať. Jeho princíp je veľmi jednoduchý, každý takt pripočíta 1 ku číslu v pamäti až kým sa nedostane na 255 (2^16-1) (toto číslo sa dá prestaviť na vlastnú hodnotu pomocou registru OCR0A, resp. OCR1AH a OCR1AL), vtedy zavolá interupt. Následne vráti counter na 0 a cyklus pokračuje od začiatku. Táto funkcia je veľmi vhodná ak treba čítať hodnoty z pinov nie kontinuálne ale iba raz za čas

S touto témou súvisia ešte funkcie, `sei()`-start gloval interupt a `cli()`-stop global interupts. Slúžia na to, aby CPU nebolo prerušené pokiaľ vykonáva nejakú činnosť, ktorá by nemala byť prerušená. Prepisujú len posledný bit v registri SREG.

Poznámky ku kódu

Pri `tiny13a/ledblink.c` si môžeme všimnúť, že nastavenie DDRB sa dá spraviť dvoma spôsobmi (jeden je zakomentovaný) z toho vyplýva, že DDB4 je vlastne len šikovne napísaná 4. Je na sprehľadnenie kódu a všetky konštanty sú definované v `/usr/avr/include/avr`, kde je .h file špecifický pre každý microcontroler. Je vhodné si pozrieť tento file aspoň raz, aby človek získal predstavu o tom, čo používa.

Poznámky k debugovaniu

Jeden z mikročipov bol z druhej ruky a nešiel preprogramovať. Pravdepodobne išlo o zmenené fuse bity buď tak, že vyžadovali externé hodiny(potrebné aj na zmenenie týchto fuse bitov, ktoré sme žiaľ nemali) alebo bol uC úplne zablokovaný.

Literatúra a zdroje:

1. Joe Pardue: C Programming for Microcontrollers, http://dsp-book.narod.ru/CPMicro.pdf
2. avrfreaks.com
3. eevblog.com/forum