STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor SOČ: 10. Elektrotechnika, elektronika a telekomunikace

8x8x8 RGB LED Cube

Jan Studený

Kraj: Olomoucký kraj

Přerov 2014

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor SOČ: 10. Elektrotechnika, elektronika a telekomunikace

8x8x8 RGB LED Cube

Autor: Jan Studený

Škola: Gymnázium Jakuba Škody

Kraj: Olomoucký kraj

Přerov 2014

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v seznamu vloženém v práci SOČ.
Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné. Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V Přerově dne 20. 4. 2014 .....

Obsah

Anotace

Cílem této práce bylo sestrojit 3D RGB LED Cube, zařízení schopné zobrazovat jednoduché 3D obrazy a naprogramované animace. LED Cube byla vytvořena z 512 barevných LED diod, které jsou poskládány do krychle s hranou 8 voxelů, mnou navržené desky plošných spojů, která je osazena čipy pro ovládání LED diod ovládaných mikropočítačem Arduino. V tomto dokumentu se nachází vývoj práce, popis konstrukce, vysvětlení základních principů pro zobrazování animací na LED kostce i jednoduchá 3D hra.

Klíčová slova: 8x8x8 RGB LED Cube, BAM, Arduino, časové přerušení, RGB LED Cube, 3D Snake

Anotation

The aim of this study was to construct a 3D RGB LED Cube, a device capable of displaying simple 3D images and animations. LED Cube was created from 512 RGB LEDs that are set into a cube with edge of 8 voxels, the newly designed PCB with mounted chips for LED controlling chips controlled by microcomputer Arduino. This paper describes development, design description, explanation of the basic principles for displaying animations on the LED cube and simple 3D game.

Keywords: 8x8x8 RGB LED Cube, BAM, Arduino, timer interrupts, RGB LED Cube, 3D snake

Úvod

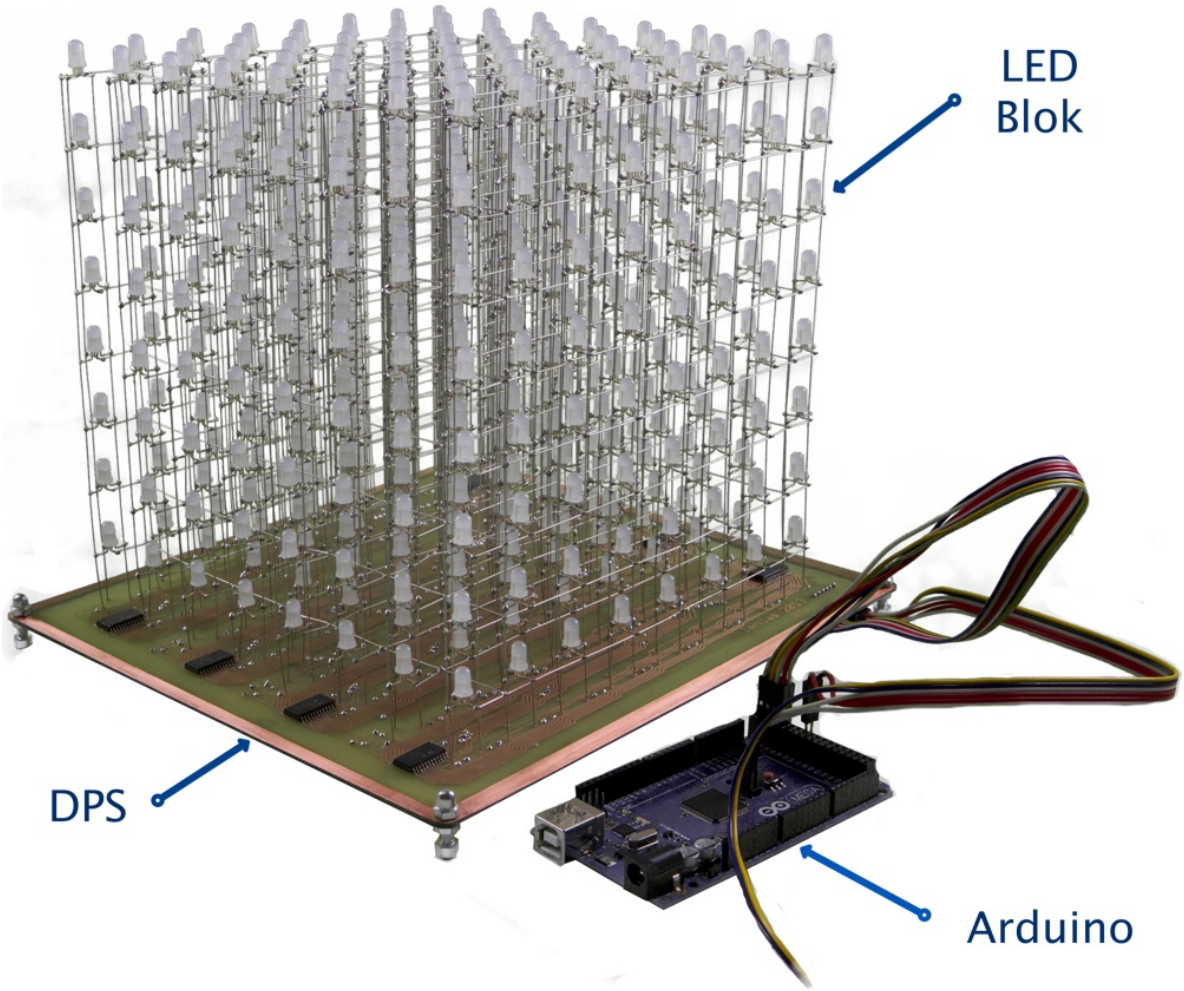
Mým původním cílem bylo pouze sestavit 3D LED Cube v domácích podmínkách, podobnou těm, které jsem našel na internetu.

Postupem času, již během objednání součástek v Číně, v on-line obchodech, jsem začal hledat řešení jak zjednodušit propojení elektronických součástek, jelikož prvotní návrh převzatý z internetu se skládal z velkého množství univerzálních plošných spojů (10 kusů), které byly mezi sebou propojeny drátky. Následně jsem musel vyřešit i problém spojení LED diod s elektronikou, abych se zbavil přebytečné kabeláže, která měla být objemnější než samotná kostka. Proto jsem navrhnul relativně lepší řešení a to v podobě mnou navržené desky plošných spojů. Jenže ani to nebylo výhrou, protože pro sestavení elektronické části a propojení s kostkou by bylo potřeba 3 oboustranných desek plošných spojů o velikosti 30 x 30 cm. Řešení jsem našel až v použití speciálních čipů na ovládání LED diod z anglického obchodu Farnell, které zmenšily DPS na velikost odpovídající velikosti LED krychle.

Po sestavení hardwaru jsem začal upravovat a vytvářet software na animace 3D obrazu. Vytvořil jsem i prostorovou verzi oblíbené hry mého dětství Snake. Na této hře lze dobře pozorovat i různé schopnosti orientace hráčů pro hraní v prostoru.

1 Hardware

Celé zařízení lze rozdělit na 3 části: LED diody (LED blok), DPS (deska plošného spoje) s čipy pro ovládání LED diod a výpočetní část (Arduino).



Obrázek 1. Rozdělení LED Cube na jednotlivé části

1.1 LED blok

Základem LED bloku je právě takové uskupení LED diod v prostoru, aby byl výsledkem 3D obraz. Nejjednodušším prostorovým útvarem na výrobu je krychle, kterou jsem se také rozhodl vyrobit. Její dvě hlavní výhody jsou:

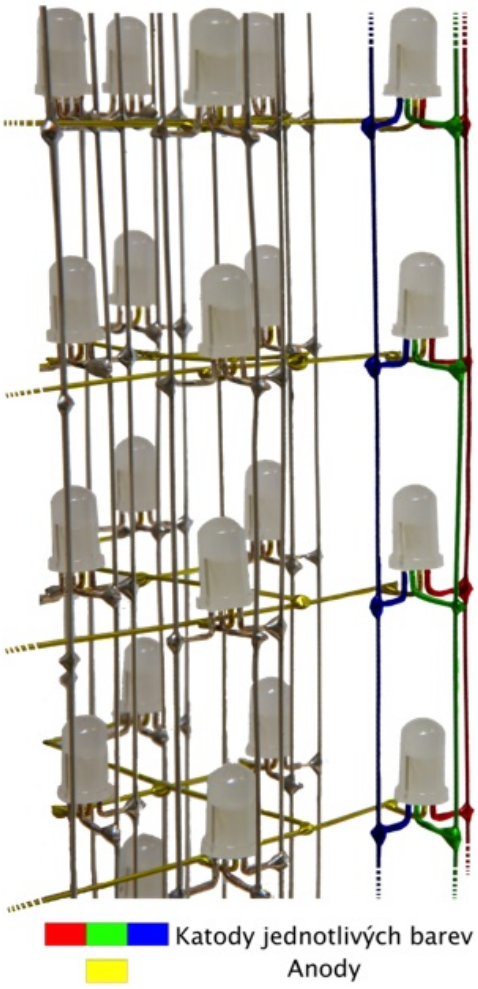
- uniformnost (šířka a hloubka kostky zůstává s její výškou stejná -> jednoduché škálování)
- obrazce není potřeba přepočítávat do polárních souřadnic (např. pro kouli)

Počet LED diod tvořících krychli je 512 (8x8x8). Je zřejmé, že čím více LED diod tvoří kostku, tím je obraz ostřejší. Ale s tímto nárůstem také roste náročnost a náklady na výrobu kostky, takže 512 LED diod se jeví jako ideální počet. Výhoda tohoto počtu tkví také v tom, že posuvné registry, které ovládají LED diody, mají 2x 8 výstupů, tj. ovládají přesně 2 řady na LED bloku.

Součástí LED Bloku mohou být jakékoliv LED diody, ale jednobarevné diody nám nedávají takové možnosti, jako použití tříbarevných LED diod s možností zobrazování celého spektra barev.

Ještě pár slov k propojení LED diod mezi sebou. Aby nebylo nutno všech 3x 512 vývodů z LED diody propojovat s DPS, tak se LED diody ovládají pomocí multiplexingu (popis principu této metody viz Časový multiplex pro LED blok na straně 13). Spojení je nejlépe vidět na obr č. 2.

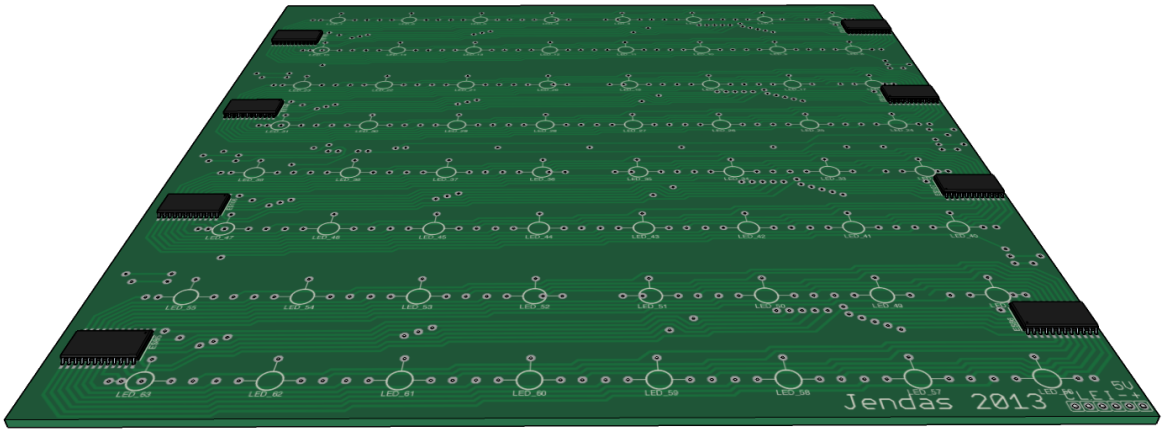
Celá kostka se skládá z 8 pater. Všechny LED diody v jednom patře mají propojené katody, a každé patro se skládá z 8 LED pásků, které mají 8 LED diod. Zbylé 3 vývody z LED diody pro ovládání jednotlivých barev jsou propojeny vertikálně s LED diodami nad sebou a pod sebou a připojeny na desku.



Obrázek 2. Spojení jednotlivých LED diod v LED bloku

1.2 DPS pro LED cube

DPS neboli deska plošných spojů byla v hardwarové části největším posunem vpřed. Většina vyrobených kostek, které jsem viděl na internetu, měla problémy s velikostí elektronické části[1]. Místo toho, abych pouze sestavil tuto část stejně jako ostatní, tak jsem se rozhodl, že vytvořím vlastní desku o stejných rozměrech jako je velikost LED bloku (24 x 24 cm). Tato deska spojí LED blok se všemi potřebnými elektronickými součástkami do jednoho celku. Celou desku můžete vidět na (obr. č. 3), s barevně odlišenými vývody z čipů poté v příloze Obrázky (obr. č. 3).



Obrázek 3. Design dps

1.2.1 Ovládání LED diod

Ovládání LED diod je zajištěno pomocí 12 speciálních čipů[2] (4 na červenou barvu, 4 na zelenou barvu a 4 na modrou barvu) a 12 trimrů na regulování proudu tekoucího jednotlivými LED diodami. Právě schopnost čipu regulovat proud tekoucí LED diodami byl rozhodující faktor pro použití čipu namísto posuvných registrů s tranzistory. Jinak bych musel dávat před LED diody předřadné odpory, kterých by bylo minimálně 300 kusů. Výhodou použitého čipu je také to, že obsahuje netradičně dvojnásobný, tj. 16-ti bitový posuvný registr[3]. Všech 12 čipů je zapojeno kaskádově (tj. vstup dat do čipu je

[1] V příloze Obrázky, obr. č. 1 a 2

[2] stp16cp05, datasheet: `http://www.farnell.com/datasheets/1690168.pdf`

[3] `http://cs.wikipedia.org/wiki/Posuvn%C3BD_registr`

spojen s výstupem předcházejícího čipu) a k ovládání jsou potřeba pouze čtyři vstupy a to:

- data in: binárně reprezentovaná data, která se mají zapsat do registrů (vstup vede pouze do 1. čipu)
- clock: hodinový signál, který určuje, kdy se mají posouvat data v registrech
- latch: aby nedocházelo k problikávání LED diod při načítání nových dat do registrů, tak se do doby než jsou data načtena, uchová aktuální stav výstupů
- output enable: vypnutí/zapnutí zobrazování

Pro realizaci multiplexingu je na desce ještě 8 MOSFET tranzistorů, které zde fungují jako spínače pro jednotlivá patra LED bloku.

1.2.2 Návrh desky

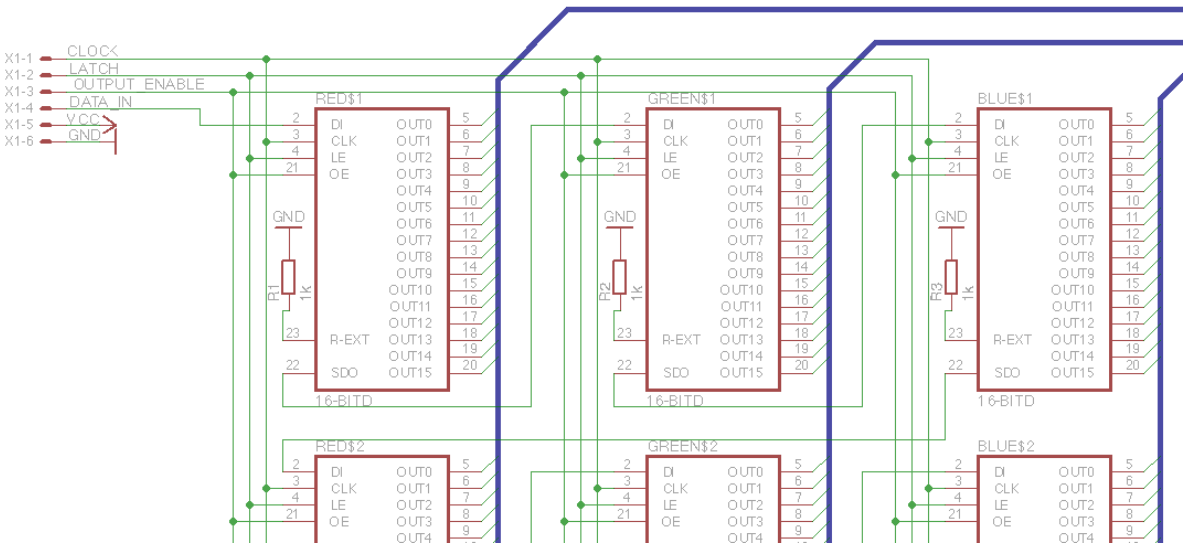
Desku jsem navrhoval v programu nazvaném KiCad, který byl napsán přímo na navrhování schémat a design DPS.[1]

Prvotní návrh desky počítal pouze s posuvnými registry, ke kterým by byly přidány tranzistory (podle schématu Kevina Darraha[4]). Jenže i při nalezení nejmenšího uspořádání prvků by desky musely být desky tři a jejich rozměry by byly 30x30 cm. I počet součástek by byl ohromující, deska by se musela osadit 400 tranzistory a 729 odpory. Při představě sestavování tak velké desky jsem začal hledat lepší řešení. Po dlouhém hledání jsem našel vhodnější řešení, a to rozsvěcet LED diody pomocí speciálních čipů na ovládání LED diod.[2]

Všechny mé předchozí návrhy se staly zbytečnými a já začal navrhovat znovu. Na výsledném designu (obr. č. 3) je patrné, že se celá deska skládá ze 4 stejných částí (modulů), které ovládají vždy 2 řady led diod. Každý čip ovládá jedny ze tří barev LED diod a všechny čipy v návrhu sdílejí společný clock, output enable a latch signál. Pro správné pochopení propojení signálů ve výsledném designu se můžete také podívat na schéma jednoho modulu na obr. č. 4.[5]

[4] `http://www.kevindarrah.com/download/8x8x8/Cube_sch_3_16_13.pdf`

[5] Schéma celé desky naleznete v příloze Obrázky, obr. č. 4



Obrázek 4. Schéma modulu DPS

1.2.3 Výroba desky

Velkým problémem bylo samotné leptání desky, jelikož jsem oboustranný plošný spoj leptal poprvé. Po nezdařeném pokusu vyleptat desku amatérským způsobem (tj. přežehlení toneru z vytištěné předlohy na desku a následné leptání) jsem zjistil, že na tuto velikost desky daná metoda nefunguje.

Ale i s druhou nejpoužívanější metodou na výrobu desek, fotocestou, byla spousta problémů. Domácí laserová tiskárna nebyla schopna vytisknout na průhlednou folii vzor v dostatečné kvalitě, proto jsem si nechal vytisknout vzor pomocí osvitu v místní tiskárně. Ani kvalitní předloha nevedla k uspokojivému výsledku. Na DPS byly některé kontakty propojené a kvalitní desku vyřešila až profesionální firma na výrobu DPS.

S osazením už žádné větší problémy nebyly, pouze bylo potřeba správně upevnit čipy, aby se při pájení neposouvaly.

1.3 Arduino

K ovládání veškeré elektroniky jsem použil mikrokontrolér Arduino pro jeho otevřenost a jednoduchost. Arduino je jednoduchá programovací platforma založená na čipech ATmega výrobce Atmel. Programy pro tento mikrokontrolér se píší v Arduino jazyce, který

je založen na C++ (kromě pár vyjímek je to pouze C++ kód). Arduino u mé kostky funguje jako jednotka, která slouží k propojení celé LED kostky s okolním světem. Verzi Arduina lze použít jakoukoliv, já jsem pro možnost větší variability a volnosti použil Arduino Mega. Arduino jsem nezabudovával do DPS pro případnou změnu ovládání kostky. Velká počáteční výhoda byla, že pro Arduino byl napsán program na ovládání kostky[6], který jsem ale musel upravit pro moji elektronickou část LED kostky. Z arduina vedou čtyři výstupy, které ovládají čipy na ovládání LED diod (latch, output enable, clock a data) a osm výstupů, které určují aktivní patro LED bloku.

1.4 Další

Ke kostce stačí už pouze připojit zdroj 5V s minimálním odběrovým proudem 4 A. K tomuto účelu mi nejlépe posloužil počítačový zdroj, jelikož je schopen dodávat až 30 A a jeho cena se pohybuje v řádu stokorun.

[6] `http://www.kevindarrah.com/download/8x8x8/RGB_CubeV12_BitwiseFix.ino`

2 Software

Kvalitní elektronický výrobek už dnes nemůže vzniknout bez patřičného propojení softwarové a hardwarové části. Stejně jako u hardwaru jsem pouze nepostavil věci podle návodu, tak i zde jsem se snažil co nejvíce zjednodušit jednotlivé procedury, ale přitom zachovat jejich robustnost. Abych se vůbec mohl pouštět do zlepšování, tak jsem musel napsat speciální proceduru, která bude zapisovat patřičné hodnoty do registrů pro LED blok z důvodu vlastního designu desky a propojení jednotlivých čipů. Při tomto kroku jsem narazil na spoustu detailů, jak softwarovou stránku upravit, aby byla jednodušší, rychlejší a přímočařejší. Následující kapitoly vám poodhalí jednotlivé softwarové principy a procedury, které moje LED kostka používá.

2.1 Teoretické principy

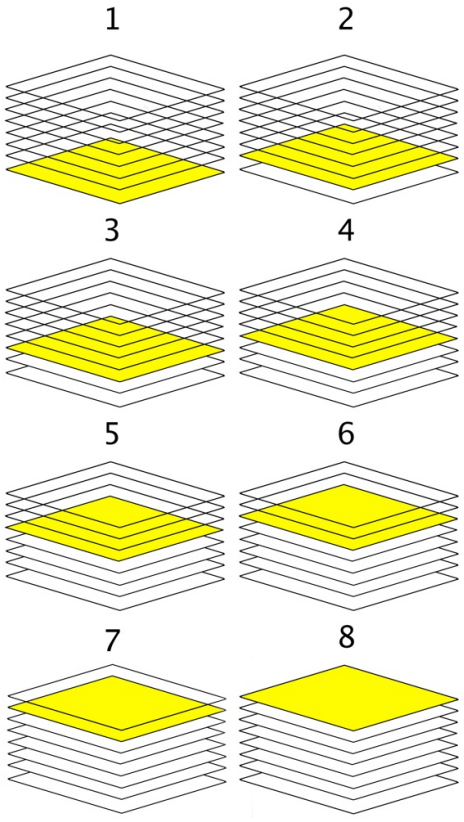
Program na ovládání kostky používá několik velmi zajímavých principů, které zajištují správné zobrazení požadovaných LED diod z programového kódu na LED blok.

2.1.1 Časový multiplex pro LED blok

Multiplexing je metoda přenosu více datových proudů přes jedno společné médium.[3] Pro analogové systémy se nejčastěji používá frekvenční multiplex, kdy se jednotlivé signály posunou do daných frekvenčních spekter a tento složený signál je možné poslat po jediném kabelu. U digitálního signálu se naopak používá zmíněný časový multiplex, který rozdělí svoji propustnost na jednotlivé časové sloty. Tyto sloty mají přesně danou délku a periodu. Jednotlivá zařízení pošlou svá data do tzv. kódovače, který je rozdělí do již zmíněných slotů, které budou pak dále poslány po společném datovém kabelu. Na konci je tzv. dekodér, který jednotlivé časové sloty stejných zařízení spojí a nasměruje na příslušný datový kabel. Takto to funguje i na mé kostce. Data jsou zde informace o jednotlivých LED diodách a časové sloty jsou jednotlivá patra LED bloku. Kódovač tedy přepíná mezi jednotlivými patry a rozsvící vždy pouze jedno patro (stejně jako na datovém kabelu nemohou probíhat dva přenosy ve stejný okamžik současně). Dekodérem, který je schopen vidět celou kostku svítit, se stane naše oko. Lidské oko totiž vnímá blikající světlo s frekvencí nad 60 Hz jako světlo s konstantním svitem. Takže místo toho, aby musel svítit celý LED blok najednou, stačí za 1 interval

(1/60 Hz ≈ 17 ms) jakkoli rozsvítit všechny potřebné LED diody. Je ale potřeba připomenout, že čím kratší dobu bude v intervalu jednotlivá LED dioda svítit, tím bude méně jasná. Jinými slovy: Všechny LED diody v 1 patře jsou spojeny a připojeny na MOSFET tranzistor určující, na kterém patře se vykresluje obraz. Nejdříve se rozsvítí všechny potřebné LED diody pouze z prvního patra, poté zhasnou. Dále se rozsvítí druhé patro, třetí patro, a tak dále až do posledního patra a poté zase začne znovu, ale s aktualizovanými daty o stavu LED diod. Názornou ukázku lze vidět na obr. č. 5.

Pomocí této technologie jsem zmenšil počet potřebných vývodů z LED bloku z 1537 (512 LED * 3 barvy + společná anoda) na 200 (64 pro červené, 64 pro zelené, 64 pro modré LED diody + 8 na ovládání pater). Nejdůležitější bylo ale to, že LED blok nepřekypuje dráty a lze vidět na všechny LED diody



Obrázek 5. Multiplexing pater LED Cube

2.1.2 Míchání barev pomocí BAM (bit angle modulation)

V digitálním světě jedniček a nul udává jas LED diody poměr mezi dobou, kdy je rozsvícená a kdy nikoli. Je ale důležité připomenout, že aby oko vidělo LED diodu ztmavenou, je potřeba, jako u časového multiplexu, frekvence větší než 60 Hz, jinak LED dioda pouze bliká.

Nejčastěji se pro digitální ztmavování nebo zjasňování LED diod používá pulzně šířková modulace, zkráceně PWM (pulse width modulation). Většina mikrokontrolérů má tuto modulaci hardwarově implementovanou a na spuštění stačí jeden řádek kódu. Princip metody je velmi jednoduchý:

- čítač zvyšuje postupně svoji hodnotu od nuly až po uživatelem nastavenou maximální hodnotu, tj. rozlišení (čím větší, tím lépe), poté se resetuje
- Při každém navýšení zjistí, zda je v registru (zde je napsána intenzita svitu LED diody) větší než hodnota čítače. Pokud ano, tak je výstup log. 1, jinak log 0.

Tento způsob jsem ale nemohl použít u mé LED kostky z těchto důvodů:

- při použití speciálních čipů na ovládání LED diod nelze hardwarovou PWM modulaci použít, protože funguje pouze pro výstupní piny z arduina, ne z LED čipu
- při použití softwarového PWM by byla taková zátěž na procesor, že by nestíhal, kromě vykreslování LED diod, nic jiného (kostka by tedy nemohla zobrazit jakékoliv animace)

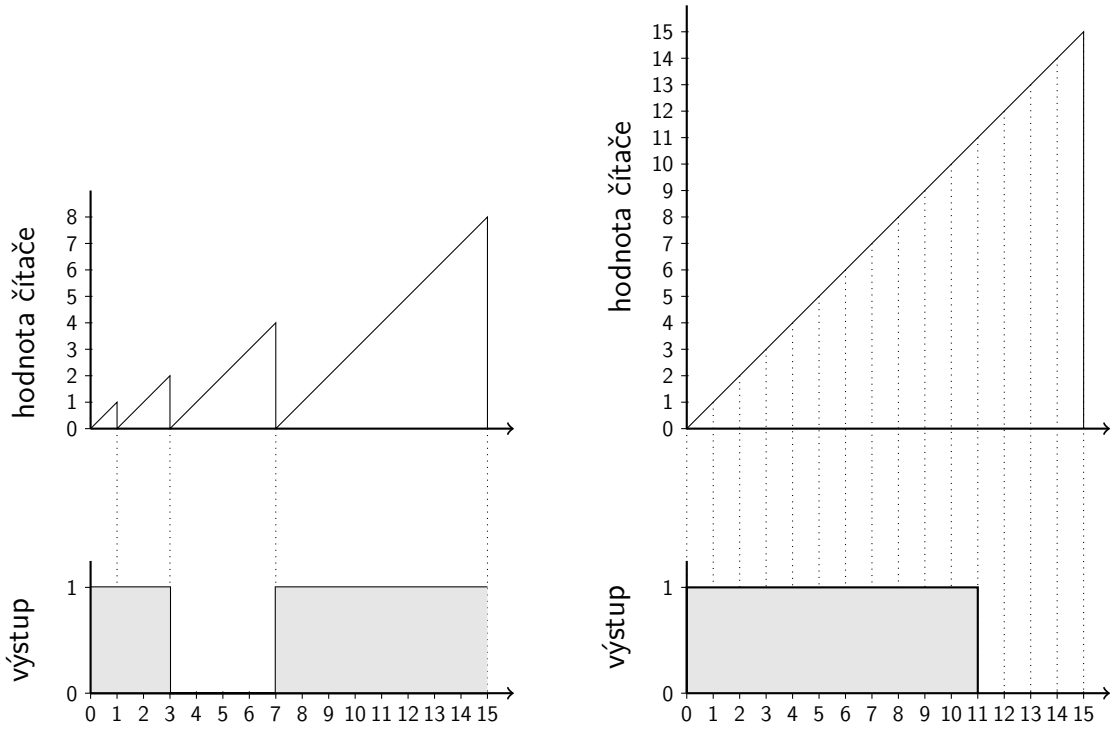
Místo této metody je na LED kostce použita méně známá, ale efektivnější metoda Bit Angle Modulation (zkráceně BAM), někdy také nazývaná jako Bit Code Modulation (BCM).[4]

Na rozdíl od PWM, kdy je časový úsek rozdělen na n stejně velkých částí, tak při BCM je rozdělen na m částí a přitom každá následující část je 2x větší než předchozí.

Pro rozlišení jasu (n) stačí místo n přerušení (pro PWM) pouze log(n) přerušení. Na obr. č. 6 lze vidět porovnání průběhu obou metod.

2.1.3 Gamma korekce

Při vytváření některých animací, které obsahovaly barevné přechody (hlavně přechod v spektru šedé) jsem si všiml, že druhá polovina přechodu (směrem k bílé barvě) byla



(a) BAM modulace (b) PWM modulace

Obrázek 6. Porovnání modulací pro svit LED diod

jednolitá a jednotlivé odstíny barev se tvořily pouze v první části přechodu. Abych odstranil tento neduh LED kostky, tak jsem musel použít metodu zvanou gamma korekce.

Aby lidské oko mohlo vnímat obrovské spektrum intenzity světla (od světla světlušky až po zářící slunce), tak vnímá jednotlivé odstíny logaritmicky, ne lineárně. Naproti tomu v počítačích se intenzity světla zapisují lineárně. Přechodníkem mezi počítačovým vyjádřením a výslednou hodnotou obstarává gamma korekce. Většinou se implementuje pomocí tabulky, kde přemapovává hodnoty zadané programem (animací) do procentuální intenzity svitu.[7]

2.1.4 Time interrupts

Time interrupts neboli časové přerušení je funkce mikroprocesoru v určitých časových intervalech nezávisle na vykonávané činnosti přeskočit z hlavního programu na danou

[7]Více informací naleznete na: `http://en.wikipedia.org/wiki/Gamma_correction` (anglicky)

proceduru. Podobně jak u hardwarové pulzně šířkové modulace zvyšuje čítač svoji hodnotu, srovnává s referenční hodnotou (při shodě změní hodnotu výstupu) a po dosáhnutí maxima se resetuje, časové přerušení pracuje na stejném principu. Při každém tiku krystalu zvýší čítač v čipu svoji hodnotu a porovná s referenční hodnotou. Pokud se referenční hodnota shoduje s hodnotou v čítači, tak nezmění jako u PWM hodnotu výstupu, ale přeruší daný program a zavolá danou proceduru. Hned po zavolání se čítač resetuje a začne znovu. Jelikož ale např. u Arduina má krystal frekvenci 16 MHz a největší čítač je 16-ti bitový (dokáže zvyšovat svoji hodnotu na maximálně $2^{16} = 65536$), tak by čítač mohl volat funkci nejpomaleji s frekvencí 122 Hz což je např. pro blikání LED diody frekvence, při které žádné blikání lidské oko neuvidí. Pro tyto případy se před každý čítač dává tzv. prescaler, který zvyšuje hodnotu při každém osmém, šedesátém čtvrtém ale klidně i při tisíci dvacátém čtvrtém tiku. Výběrem správného prescaleru a referenční hodnoty lze dosáhnout velkého rozsahu frekvencí (od 8 MHz, který se používá např. při SPI komunikaci až po 0,2 Hz, jenž je vhodný pro logování hodnot z teplotních čidel). Potřebnou referenční hodnotu lze vypočítat pomocí vzorce [5]:

$$Ref.\ hodnota = \frac{frekvence\ krystalu}{prescaler * požadovaná\ frekvence} - 1$$

Pokud je vypočtená hodnota větší než hodnota, kterou je schopen pojmout čítač (256 pro 8-mi bitový, 65535 pro 16-ti bitový), tak je potřeba zvýšit hodnotu prescaleru. Pokud je naopak výsledkem desetinné číslo, tak je vhodné prescaler snížit (zmenší se odchylka od požadované a produkované frekvence).

Stejným vzorcem jsem vypočítal i potřebnou referenční hodnotu pro aktualizování čipů na ovládání LED diod. Kostka se obnovuje s frekvencí 90 Hz. Pro správnou funkci multiplexingu je potřeba za tuto dobu vykreslit všechny patra na LED bloku, takže je nutné vynásobit požadovanou frekvenci počtem pater (tj. 8*90 Hz = 720 Hz). K BAM modulaci je ještě potřeba, aby doba svitu jednoho patra byla rozdělena na čtyři různě velké části. Pro nejmenší z nich je potřeba frekvenci vynásobit 24 (tj. 16*720 Hz = 11,5 kHz, resp. pro další části 5,8 kHz, 2,9 kHz a 1,4 kHz). Prescaler jsem použil s hodnotou 64 a výsledky výpočtu referenčních hodnot jsou následující:

- 1. část BAM = 20
- 2. část BAM = 40
- 3. část BAM = 80

- 4. část BAM = 160

2.2 Cube knihovna

Při každé úpravě nebo rozšíření funkcí LED kostky docházelo k zvětšování zdrojového kódu, který pomalu stal neobratným a nesrozumitelným. Tato skutečnost mě donutila všechny důležité a potřebné funkce přepsat a spojit do knihovny, která se postará o všechny důležité věci k správnému zobrazení dat na LED blok a zjednoduší uživateli (vývojáři) naprogramování vlastních animací a programů. Celou knihovnu lze rozdělit na 2 části:

- zobrazovací (privátní) – část, která zodpovídá za správné zobrazení dat na LED blok
- animační (veřejná) – zde se aktualizují pole dat s novými snímky v animaci

Obě části spojuje pole JAS_LED_DIOD, do kterého animační část zapisuje změny v jasu diod a zobrazovací část čte hodnoty a zobrazuje je na LED blok.

2.2.1 Popis zobrazovací části programu

Pro co největší zjednodušení a pochopení principů, jak funguje zobrazovací část knihovny, jsem se rozhodl, že základní konstrukce programového kódu budu v této práci zapisovat pomocí pseudokódu, což je zápis programu, který je jednoduše čitelný pro i pro laika. V pseudokódu se píší hlavně algoritmy, které by při své implementaci byly velice obtížně pochopitelné.
Vysvětlivky k pseudokódu:
//text = pomocné informace, které nemají žádnou funkci
příkazové řádky jsou psány v češtině bez diakritiky

```
sviticiPatro;
aktualni_blok_BAM_modulace;//pomocné proměné

Procedure AktualizujLEDBlok:
//část kódu zodpovědná za svícení LED diod
```

```
//tato procedura se vykonává v daných intervalech nezávisle na hlavním programu
ZhasniLEDky();
//aby nedocházelo k nechtěnému rozsvícení LED diod během načítání nových dat
jsou po dobu nahrávání diody vypnuty
Pokud sviticiPatro <  PocetPater
Tak sviticiPatro = sviticiPatro + 1;//pokud není aktivní patro poslední, tak zvyš o 1
Jinak sviticiPatro = 0;//začni vykreslovat od začátku

//pokud není dokončená BAM_modulace tak posun cteni bitu o 1
Pokud aktualni_blok_BAM_modulace < BAM_rozliseni
Tak aktualni_blok_BAM_modulace = aktualni_blok_BAM_modulace + 1;
Jinak aktualni_blok_BAM_modulace = 0;

AktualniLED = PrevedNaBinarni(JAS_LED_DIOD.aktualniPatro);
data = VyberDanyBit(AktualniLED, aktualni_blok_BAM_modulace);
NahrajDataDoChipuProLED(data);
RozsvitLEDky();
poIntervaluSpust(124^(aktualni_blok_BAM_modulace) μs, AktualizujLEDBlok);
End procedure
```

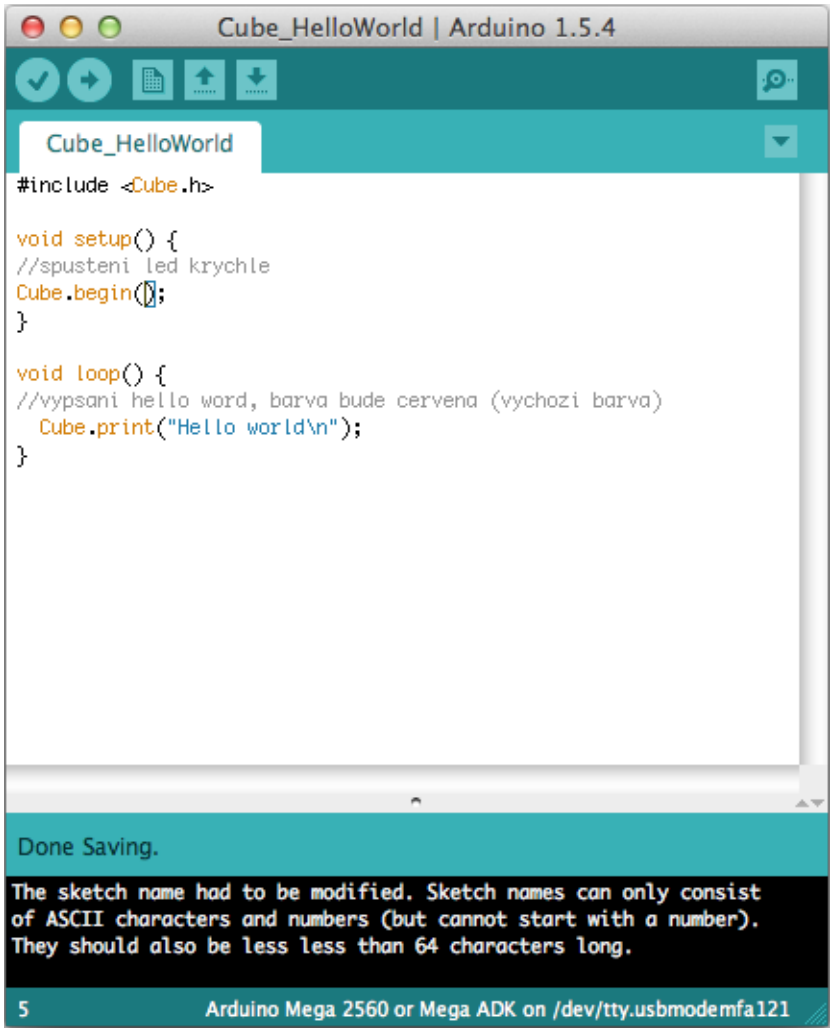
2.2.2 Popis animační části programu

Nejzákladnější, a také nejvíce používanou funkcí pro vytváření nových programů a animací je bezesporu funkce, která rozsvítí LED diodu v krychli s danou barvou. Syntaxe příkazu plně vystihuje její funkci. LED(souradnice_xyz, barva) zapisuje na správná místa do pole JAS_LED_DIOD hodnoty jednotlivých barev dané LED diody. Barvu LED diody lze zapsat trojím způsobem:

- název barvy – nejintuitivnější způsob (knihovna obsahuje sadu základních barev)
- jas červené LED, jas zelené LED, jas modré LED – pokud chceme využít celé spektrum barev
- hexadecimální zápis – klasický zápis barev v počítačovém světě

Knihovna obsahuje také jednoduché zobrazování textu, který koluje na vnější straně LED kostky. Pro zobrazení textu stačí pouze zavolat funkci print("text \n", barva,

prodleva) (\n značí ukončení řádku tj. text po objetí vnějších stran krychle zmizí).



Obrázek 7. Vypsání Hello world na 3d kostku

2.3 Ukázkový program - Snake

Na kostce není nutné pouze zobrazovat animace, ale lze ji i ovládat. To jsem se snažil ukázat na klasické hře známé z mobilních telefonů značky Nokia. Žádná jiná hra není tak ikonická jako právě Snake. Zdrojový kód jsem upravil a přepsal do 3D z mého předchozího portu této hry na LED maticový display. Úprava hry do 3D nebyla až tak složitá, horší bylo vymyslet ovládání, kterým se bude had ovládat.

Závěr

Na LED Cube lze předvést všechny základní principy zobrazování obrazců z Arduina na LED diody. Její výhoda tkví především v kompaktnosti, které se mi podařilo pomocí DPS dosáhnout. S pomocí vytvořeného softwaru lze využít LED kostku jako zobrazovací 3D display.

Splnil jsem cíl, který jsem si dal. Sestrojit LED Cube s dalšími možnostmi využití, které vidím ve vytváření vlastních animací a programů novými uživateli. Pro ně jsem vytvořil Cube knihovnu, kde jsou základní nástroje na ovládání LED kostky.

Potenciál dalšího použití LED kostky vidím v programech pro analyzování zvuku, pro vizualizaci hudby, remake starších her do 3D, pro využití ve výuce nejen při programování a informatice, ale i v matematice při zobrazování řezů a průniků přímek, rovin, těles ve stereometrii a mnoho dalších.

Seznam použité literatury

[1] KiCad. *EDA Software Suite - Kicad EDA* [online]. 2014 [cit. 15-01-2014]. Dostupné z: http://www.kicad-pcb.org/display/KICAD/KiCad+EDA+Software+Suite.

[2] Nick Schulze. *RGB LED CUBE. How not to engineer* [online]. 2013 [cit. 16-01-2014]. Dostupné z: http://www.hownottoengineer.com/projects/rgb-LED-cube.html.

[3] Wikimedia Commons. *Multiplexování* [online]. 2013 [cit. 15-03-2014]. Dostupné z: http://cs.wikipedia.org/wiki/Multiplexov%E1n%ED.

[4] Nigel Batten. *LED dimming using Binary Code Modulation* [online]. 2009 [cit. 03-03-2014]. Dostupné z: http://www.batsocks.co.uk/readme/art_bcm_3.htm.

[5] Amanda Ghassaei. *Arduino timer interrupts* [online]. 2013 [cit. 03-03-2014]. Dostupné z: http://www.instructables.com/id/Arduino-Timer-Interrupts/step1/Prescalers-and-the-Compare-Match-Register/.

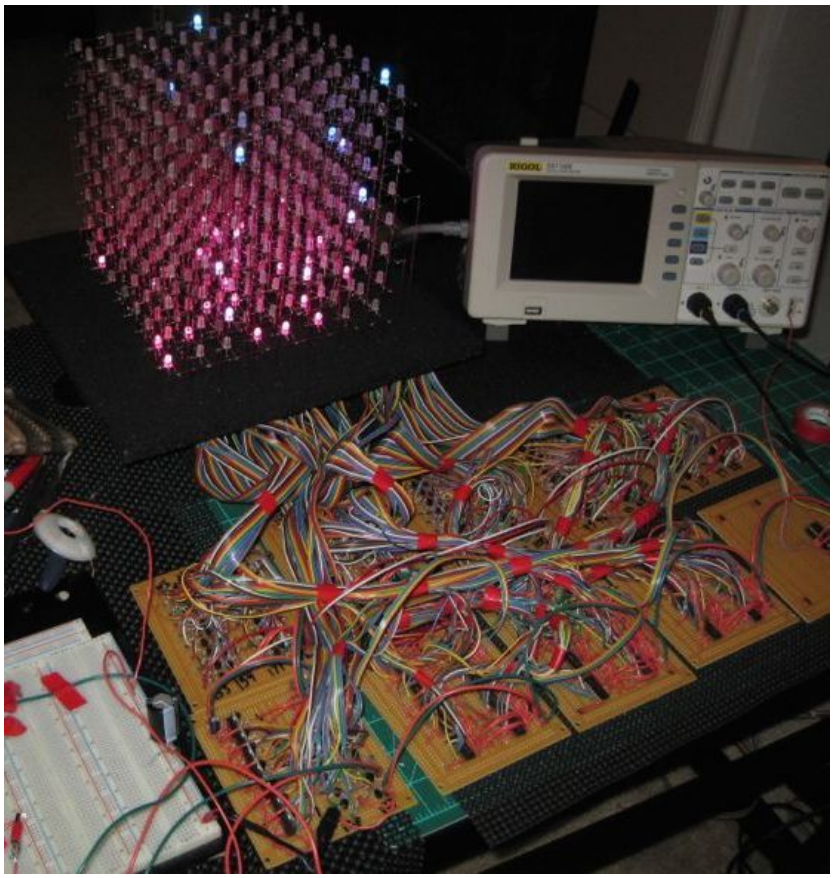
[6] Kevin Darrah. *RGB CUBE 8x8x8. Dedicated to desing* [online]. 2013 [cit. 15-01-2014]. Dostupné z: http://www.kevindarrah.com/?cat=99.

[7] Razorconcepts. *Arduino is slow and how to fix it* [online]. 2010 [cit. 03-03-2014]. Dostupné z: http://www.instructables.com/id/Arduino-is-Slow-and-how-to-fix-it/.

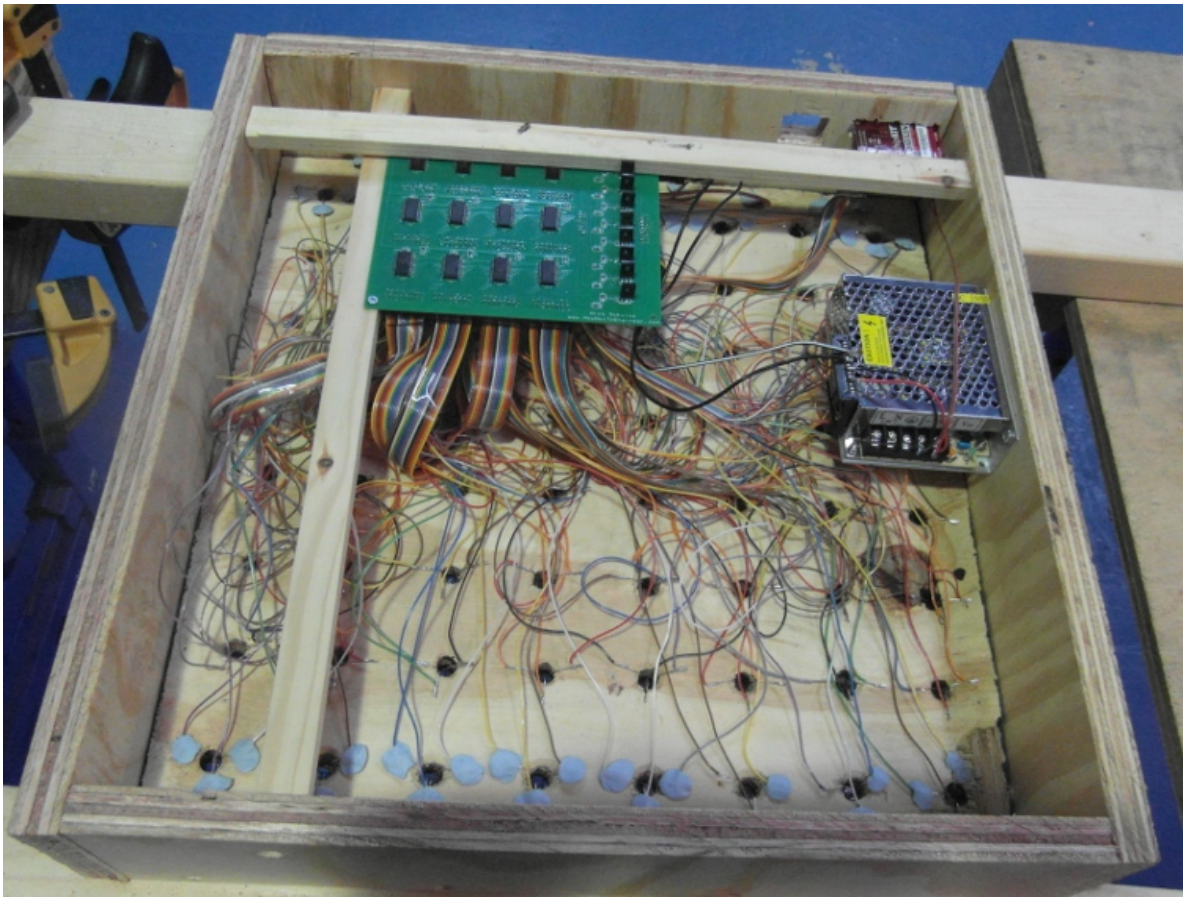
[8] Kevin Darrah. *RGB CUBE 8x8x8 wiring* [online]. 2013 [cit. 25-03-2014]. Dostupné z: http://www.kevindarrah.com/wp-content/gallery/cubular/154.jpg. File: KevinWiring.jpg.

[9] Nick Schulze. *RGB LED CUBE wiring* [online]. 2013 [cit. 25-03-2014]. Dostupné z: http://www.hownottoengineer.com/projects/images/rgb_led_cube/cubebox.JPG. File: SchulzeWiring.jpg.

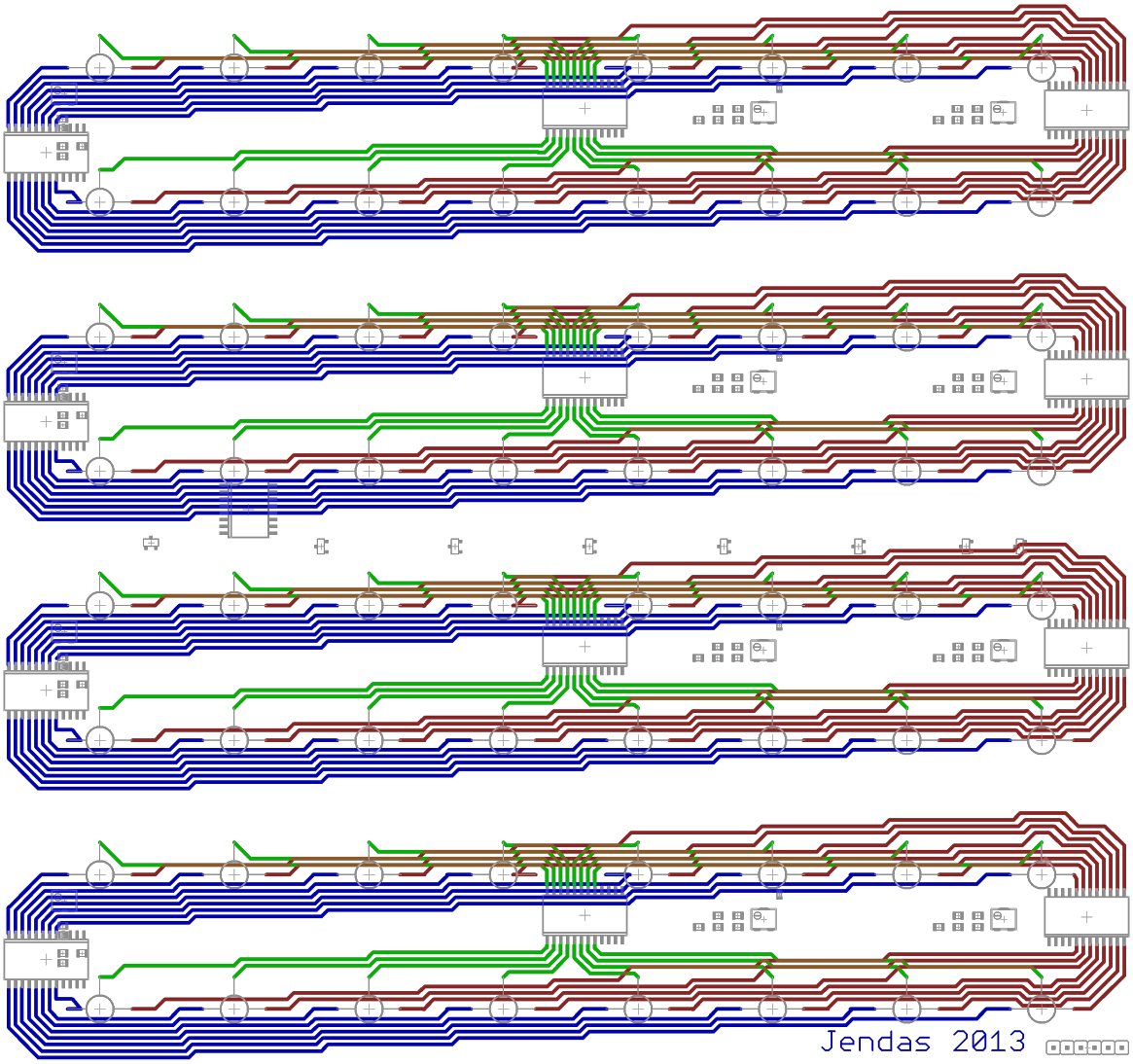
Příloha A Obrázky



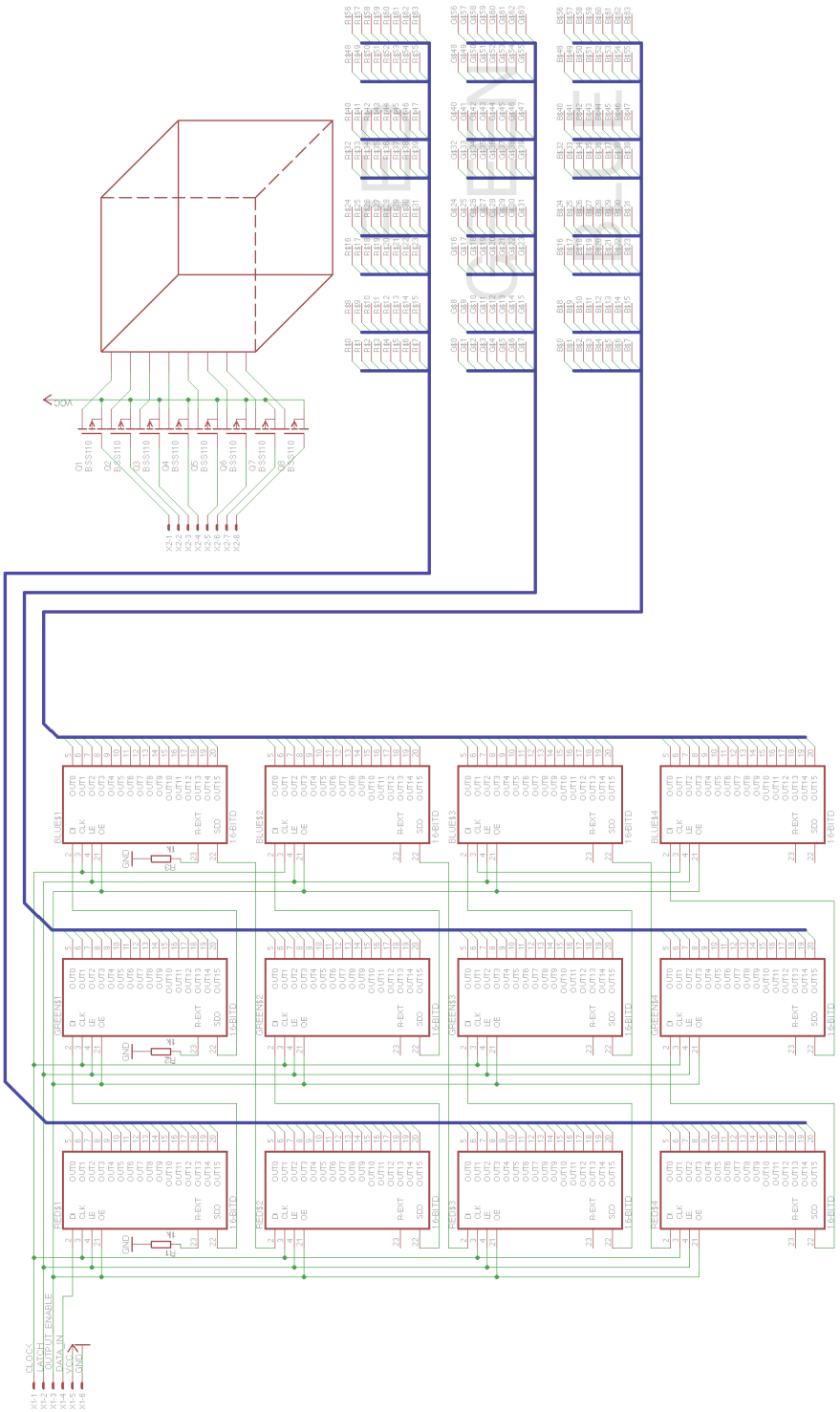
Obrázek 1. Elektronika Kevin Darrah



Obrázek 2. Elektronika HNTE.COM



Obrázek 3. DPS pro LED CUBE s barevně rozlišenými vývody



Obrázek 4. Schéma DPS

Příloha B Zdrojový kód

Zdrojový kód pro LED kostku si můžete stáhnout na stránce `https://github.com/jendas1/Cube`. Uložení zdrojového kódu online je výhodné oproti jiným úložisťím (CD, DVD) např. kvůli jednoduchým aktualizacím, možnosti upravovat a navrhovat změny v kódu, a také že si kód bude moct stáhnout opravdu kdokoliv. K využití online úložisť vede ale i jeden prozaičtější důvod, a to že můj počítač již nemá mechaniku. Zdrojový kód pro tento dokument ve formátu .tex nebo .pdf můžete nalézt na stránce `https://www.sharelatex.com/project/536a362ffa8c49153a5e0fa7`