Středoškolská technika 2019

Setkání a prezentace prací středoškolských studentů na ČVUT

Moderní chladicí podložka

Pavel Stenchlý

Střední průmyslová škola, Karviná, příspěvková organizace

Žižkova 1818, Karviná - Hranice

Anotace

Práce se zabývá výrobou chladicí podložky pod notebook, kterou jsem vyrobil, abych snížil teplotu notebooku. Cílem práce je navrhnout a vyrobit chladicí podložku a rozšířit ji o další praktické a zábavné funkce. Práce je rozdělena na teoretickou a praktickou část. V teoretické části jsou vysvětleny základní pojmy, které jsem potřeboval znát pro výrobu. V praktické části je detailně zachycen postup výroby, popsány funkce a program chladicí podložky.

Klíčová slova

prototyp; chladicí podložka; Arduino; funkce; využití

Annotation

The work is focused on making a laptop cooling pad which will reduce the temperature of my laptop. The aim of the work is to design and make the cooling pad and to extend it with other practical and fun functions. The work is divided into the theoretical and practical part. The theoretical part explains the basic concepts I needed to know to make the cooling pad. The practical part describes the production process, the functions and the program of the laptop cooling pad.

Keywords

prototype; cooling board; Arduino; functions; usage

OBSAH

ÚVOD

Ve své práci se zabývám vývojem a výrobou chladicí podložky k notebooku. Hlavním důvodem výroby chladicí podložky byl můj notebook, který se přehříval. Potřeboval jsem snížit jeho teplotu, tím zvýšit jeho výkon a prodloužit životnost. Rozhodl jsem se však pro výrobu vlastní chladicí podložky. Chladicí podložka, kterou jsem vyrobil, má funkce, které nejsou v klasických chladicích podložkách obvyklé.

Práce je rozdělena do dvou části, a to teoretické a praktické. Teoretická část se věnuje práci s mikropočítačem Arduino a programováním v jazyce Wiring. Dále popisuji princip chlazení.

V praktické části se zabývám vývojem chladicí podložky, výrobou konstrukce, elektronikou, programováním a funkcemi chladicí podložky. Funkčnost a účinnost chladicí podložky jsem ověřil v další kapitole praktické části.

TEORETICKÁ ČÁST

1.1 Arduino

Arduino je elektrická vývojová platforma, která je snadno programovatelná, a proto je vhodná i pro začátečníky. Platforma Arduino je hodně rozšířená, tato platforma může vnímat okolí pomocí vstupních senzorů nebo jej ovlivňovat pomocí výstupních periferií. Je mnoho desek, klonů a shieldů pro rozšíření možností práce s Arduinem.

Oproti jiným kitům jsou levnější, mají jednoduché zapojení a existuje mnoho návodů. Ve světě a ČR je široká komunita zabývající se Arduinem. Základní části většiny desek je mikroprocesor ATMega328. To je výkonný 8bitový mikroprocesor firmy ATMEL, deska dále obsahuje krystal, napájení na 5 V a převodník pro sériovou komunikaci s počítačem.

Využití je velmi široké, protože se vyrábí spousta doplňujících senzorů a výstupních periferií. Může sloužit pro zábavu, výuce programování, rozšíření svých znalosti a dovednosti. Záleží, co daný uživatel potřebuje a jak to naprogramuje. Arduino může sloužit například jako bezpečnostní systém nebo pro detekci kouře či plynu CO. [1, 2]

1.2 O jazyce programování

Aby Arduino deska vykonávala to, co potřebujeme nebo chceme, musíme ji naprogramovat. Nejrozšířenějším programovacím jazykem je Wiring a programátorské prostředí Arduino IDE. Tento software je volně dostupný a spolu s ním i velké množství knihoven pro správnou funkci připojených periferií. Prvním autorem jazyka Wiring je Hernando Barragán, který jej představil ve své diplomové práci v roce 2003.

Wiring patří k vyšším programovacím jazykům, byl vytvořený pro snadné programování mikropočítačů bez větších znalostí hardware. Tento jazyk je vyvíjen v C a C++. Software je funkční na běžných operačních systémech a dnes už i na mobilních zařízení. Nejčastěji využívané programátorské prostředí je již zmíněné Arduino IDE. Jedná se o přehledné a jednoduché prostředí.

Po napsání zdrojového textu a před nahráním strojového kódu do mikropočítače dojde ke kompilaci a poté pomocí převodníku k nahrání do čipu. Tímto způsobem je kód zapsán do paměti EEPROM mikropočítače a běží neustále do kola a plní své naprogramované funkce. [3, 11]

1.3 Chlazení

Základním principem chlazení je odvod tepla z místa, které se zahřívá a k tomu je zapotřebí neustálá cirkulace vzduchu. Při chlazení proudící vzduch neustálé obtéká nebo naráží na chlazenou část a tím odvádí teplo z povrchu tělesa.

Pro správné chlazení musí být zajištěn prostor pro dostatečné nasátí vzduchu větráku, proudící vzduch musí obtékat okolo tělesa, aby mohl odvést teplo. Dále musí být prostor pro únik vzduchu, aby nedocházelo k víření a na těleso mohl působit nový proud vzduchu.[1]

[1] Zdroj – Autor

PRAKTICKÁ ČÁST

2 VÝVOJ

2.1 Vývoj chladicí podložky

Začal jsem pracovat na vývoji prototypu chladicí podložky. Nejdříve jsem zjistil, že potřebuji výkonnější větráky. Jenomže silnější větráky s menším odběrem proudu jsem nenašel. Proto jsem zakoupil step-up měnič a výkonnější větráky na 12 V.

V původním návrhu bylo, že chladicí podložka bude napájená z USB notebooku. Bohužel můj notebook to proudově neutáhnul a docházelo k většímu úbytku napětí. Snažil jsem se vyhnout externímu 12 V napájení, které se však nakonec ukázalo výhodným. Napájení Arduina je na 5 V, ale má pin Vin, kam mohu přivést vyšší napětí. Protože Arduino má zabudovaný stabilizátor a chod Arduina je na klasických 5 V, tím jsem vyřešil problém větráku na 12 V a chod Arduina na 5 V. Jiné možné řešení by bylo například odporovým děličem napětí.

Další vývoj probíhal v samotném programu. Postupoval jsem po malých krocích. Zkoušel jsem a zároveň se učil, jak samotné Arduino a programátorský jazyk Wiring fungují. Základem bylo rozjet IR komunikaci, což se mi také bez větších problému povedlo. Následovala volba vhodného ovladače. Zjistil jsem, že se ovladače trochu liší a to nejenom v samotném kódu. Nechtěl jsem použít velký a těžký ovladač. Proto jsem zvolil ten menší dálkový ovladač ze starého rádia.

Obrázek 1 - Prototyp

Obrázek 2 - Dálkový ovladač

2.2 Problémy v průběhu projektu

V průběhu mé práce jsem narazil na problémy. Jeden byl špatně přeložený strojový kód. Měl jsem sice dobře napsaný příkaz, ale po nahrání do Arduina však příkaz nefungoval. Příčinou bylo například špatné přeložení mezery před rovnítky, a to i přesto, že v jiné části programu to fungovalo bezchybně.

Dalším problémem je, že po zapnutí napájení a spuštění programu, se na OLED displeji nezobrazila daná zpráva. Nevím, proč k tomu dochází, neboť při restartování programu nebo opětovném zapnutí napájení v krátkém čase se na displeji zpráva zobrazí správně. Tento problém jsem odstranil funkcí, která restartuje program po zmáčknutí nastaveného tlačítka na ovladači. Tím jsem ale narazil na další problém. Aby se dokázal nahrát nový program do desky Arduino, musí dojít k restartování. Tomu ale brání spojení pinu RST a dalšího pinu, který mám nastavený. Na pinu RST je stále 5 V, ale pro vyvolání resetu musí být na pinu 0 V. A to už Arduino není schopno samo provést. Pokud mám spojený RST s jiným pinem, kde je nastaveno 5 V k nahrání programu nedojde, dokud ho nerestartuji.

Další poznatek se týká samotného programovacího jazyku. Když na displeji zobrazuji informaci o jasu, proměnná *jas* je číslo, ke kterému přičítám nebo odčítám 0,1. A na displeji se zobrazí například místo hodnoty 40 %, hodnota 39 %. Ale v jiném typu proměnné se zobrazovalo správně 40.00 %. U hodnoty 0 bylo však na displeji zobrazeno -0.00 %. Z toho jsem usoudil, že chyba je při počítání v programu. A dochází k chybnému výpočtu. Tento problém je vyřešen připočtením 0,0005 k hodnotě *jasu*, aby bylo zajištěno správné zaokrouhlení a zobrazení na displeji. Vysvětlil mi to jeden programátor, tak že některé programovací jazyky s tím mají problém a v nejjednodušších případech stačí přičíst jen 0,0005.

3 KONSTRUKCE

3.1 Základní část

Pro výrobu konstrukce jsem použil starý dveřní práh, sololitovou desku, plastové lišty a spojovací materiál.

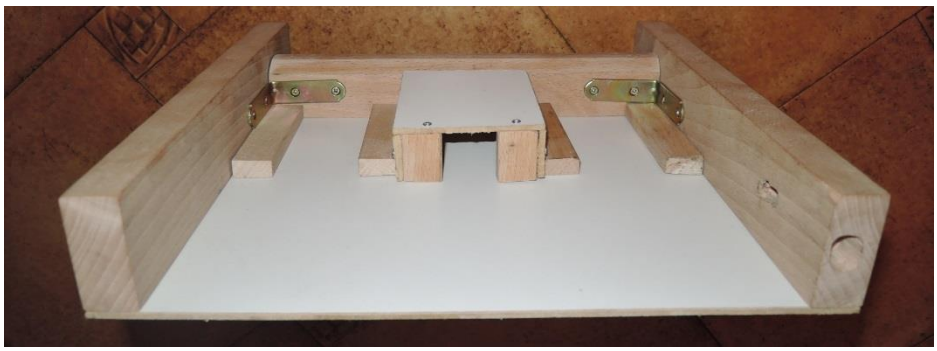
Nejdříve jsem zbrousil vibrační bruskou starý lak z prahu, novou vrstvu jsem již nedával. Poté jsem si rozměřil jednotlivé díly a nařezal pomocí kotoučové pily. Kolmé strany jsem k sobě připevnil pomocí úhelníku. Dále pomocí vrutů jsem přimontoval sololitovou desku k dřevěné konstrukci z prahu.

V zadní části mezi větráky je vytvořena krabička pro uložení plánovaného plošného spoje. Nyní se v ní nachází zapojení na nepájivých polích. Pod větráky jsou dřevěné podpěry, aby větráky měli dostatečný prostor pro nasávání vzduchu. Rozměry konstrukce jsou: 296 × 267 × 53 mm. Šířka je s přesností 2 mm na rozteč nožiček. Takže notebook se nemůže posouvat do strany. Hloubka je s přesahem, pro nastavení svitu LED do okolí. Výška je přizpůsobená tak, aby byl dostatečný prostor pro nasátí vzduchu, výšku větráku a pro prostor mezi větráky a notebookem. K pohybu vpřed a vzad nedochází, notebook zůstane na místě díky své hmotnosti a tření.

3.2 Další části

Plastové lišty jsou uchyceny tavnou pistolí. A jsou v nich vedeny rozvody pro OLED displej IR čidlo a RGB LED. IR přijímač je umístěn na pravém rameni na čelní ploše, aby nebyla nějak narušena či omezena komunikace s dálkovým ovladačem. IR čidlo je částečně zasunuté v díře, přes kterou jsou k němu vedeny vodiče pro napájení a odesílání dat do Arduina. A to proto, aby vodiče nebyly vidět a zbytečně čidlo nevyčnívalo.

OLED je umístěn na vlastním podstavci a je přišroubován k levému rameni. S displejem se dá otáčet pro lepší viditelnost. Momentálně je však podstavec dotažen na maximum tím zafixován v dané pozici. V případě potřeby však není problém si displej natočit podle potřeby. Čidlo teploty a vlhkosti je umístěno tak, aby nebyla ovlivněna data o teplotě, a to jak teplotou notebooku nebo prouděním vzduchu z větráku.



Obrázek 3 - Konstrukce

4 ELEKTRONIKA

4.1 Hlavní součást – mikropočítač

Základní součástí, která vše řídí je Arduino nano. Tento mikropočítač se stará o celý chod chladicí podložky. Nejdůležitějším obvodem je čip ATMEGA328P. Následně mohu vytvořit program, který se na čip nahraje, po zapnutí se program vykoná. Je to dobrá věc, se kterou si člověk, může hrát a vzdělávat se. Například mám v plánu vytvořit si menší domácí meteostanici a automatické rolování žaluzií.

4.2 Další součástky

Další součástí jsou napájecí zdířka 2,1mm pro připojení napájení, NPN tranzistory pro spínání PWM[2] větráků a jednotlivých barev. Dále RBG LED pro širší možnosti barev, IR čidlo a OLED displej. Stabilizovaný zdroj pro případné další použití. Použil jsem také několik rezistorů, čidlo teploty a vlhkosti a nesmím opomenout dva PC větráky o celkovém výkonu 4,8 W. Vše je prozatím zapojeno na dvou malých nepájivých polích.

Napájení je z adaptéru o výstupních hodnotách DC 12 V a 2 A. Jako další údaje jsou uvedeny odběr a spotřeba. Proud je změřen ampérmetrem. Minimální odběr pro chod Arduina a napájení čidel je 30 mA. Maximální odběr činí 470 mA. Toto je odběr při maximálním využití – větráky na plný výkon, svit RGB LED i provoz displeje.

4.2.1 Tabulka spotřeby[3]

Příkon zdroje	7,2 W
Výkon zdroje	5,7 W
Účinnost zdroje	cca 80 %
Spotřeba	7,2 Wh
1 kWh	139 h = cca 6 dnů
1 h	0,036 Kč
1 Kč	28 h

Tabulka 1 Spotřeba el. Energie (zdroj – autor)

Z tabulky je zřejmé, že tato chladicí podložka není nákladná na provoz.
(cena se může lišit, záleží na tarifu)

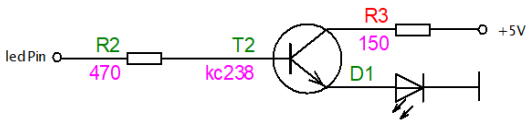
[2] Pulzně šířková modulace
[3] Hodnoty jsou při maximálním využití. Spotřeba je změřená přístrojem na měření spotřeby elektrické energie. Další údaje jsou vypočteny.

4.3 Popis součástek

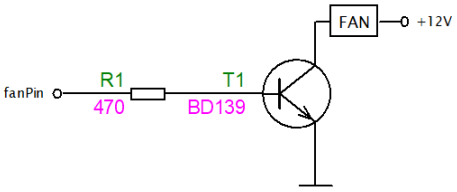
4.3.1 Tranzistory a stabilizovaný zdroj

Mám použité dva druhy tranzistorů. První z nich jsou výkonové BD139 pro větráky. Druhé jsou málo výkonové tranzistory typu KC238 pro jednotlivé barvy. Dva druhy proto, že KC238 jsou slabé pro větráky a BD139 jsou zbytečně silné pro LED.

Stabilizovaný zdroj LM317. Jeho výhodou je nastavení výstupního napětí. Mám nastavenou hodnotu 5,05 V, která by měla být udržována až do proudu 1,5 A. Tento stabilizovaný zdroj je zde pro případnou větší zátěž.



Obrázek 5 - Zapojení LED



Obrázek 4 - Zapojení větráků

4.3.2 LED a rezistory

Použil jsem RGB LED 5 mm se společnou katodou. Protože mají široké možnosti nastavení barev.

4.3.3 Čidla a displej

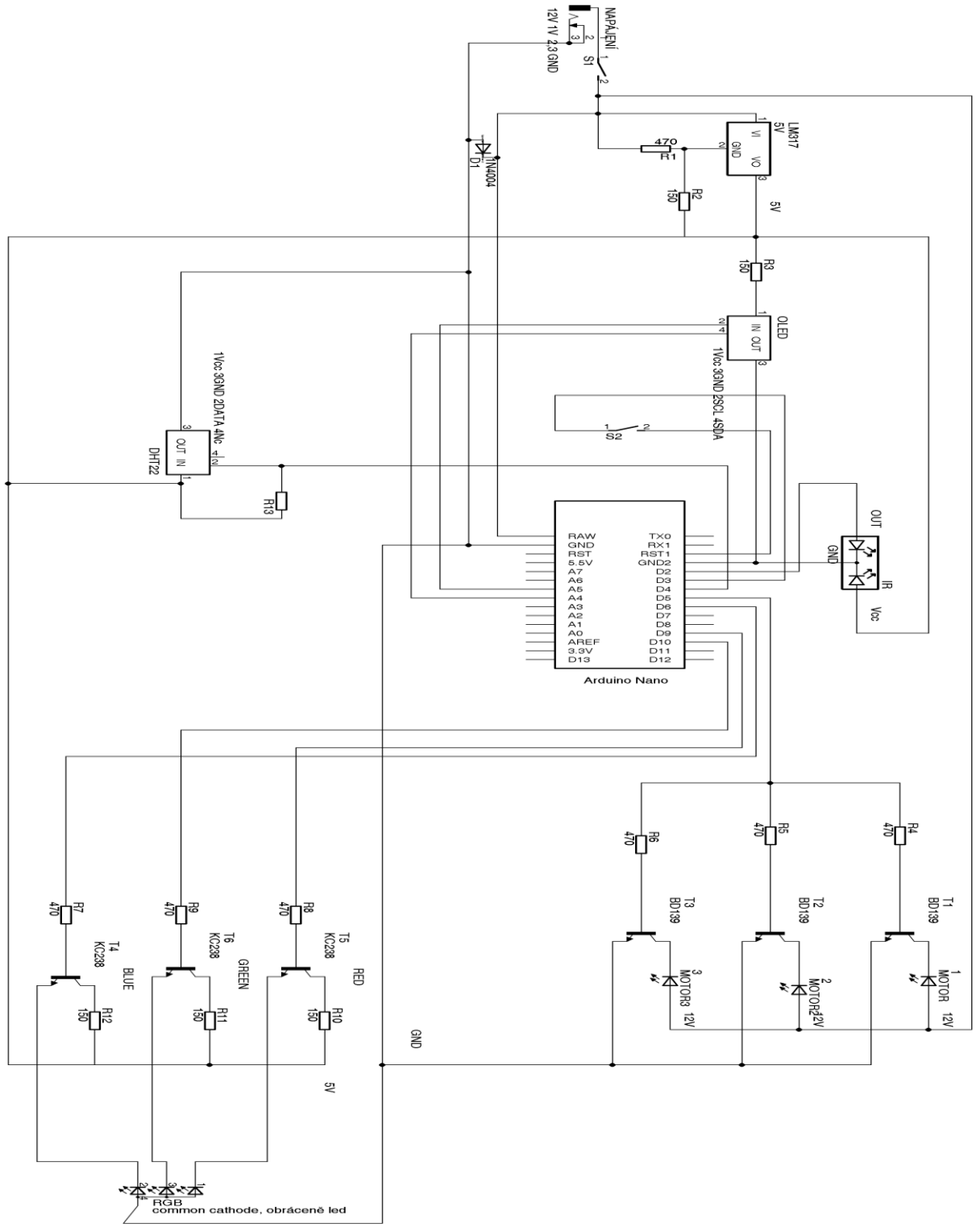
IR čidlo je velice jednoduché. Má jen tři piny a to jsou Vcc, GND a OUT. Čidlo teploty a vlhkosti DHT22. Má tři piny a jsou to rovněž Vcc, GND a OUT. Výstup dat je v digitální podobě. Měřící rozsah: teplota -40 až 80 °C s přesností ± 0,5 °C; relativní vlhkost 0 až 100 % s přesností ± 2 %. Toto čidlo je sice trošku dražší, ale má větší rozsah a přesnost než například čidlo DHT11. OLED displej má čtyři piny – Vcc, GND, pro sériovou komunikaci s Arduinem SDA (data) a SCL (hodiny). Výhodou je, že má velice nízkou spotřebu. Spotřeba se odvíjí od počtu rozsvícených bodů. [7]

4.3.4 Větráky

Použil jsem dva PC větráky každý o výkonu 2,4 W. Tyto větráky jsem získal rozebráním starého nefunkčního počítače. Větráky jsem vybíral podle následujících kritérií: dobrá regulace, dostatečný výkon a nízký hluk. Rozměry větráku jsou 80 x 80 x 25 mm.

4.3.5 Schéma a návrh DPS

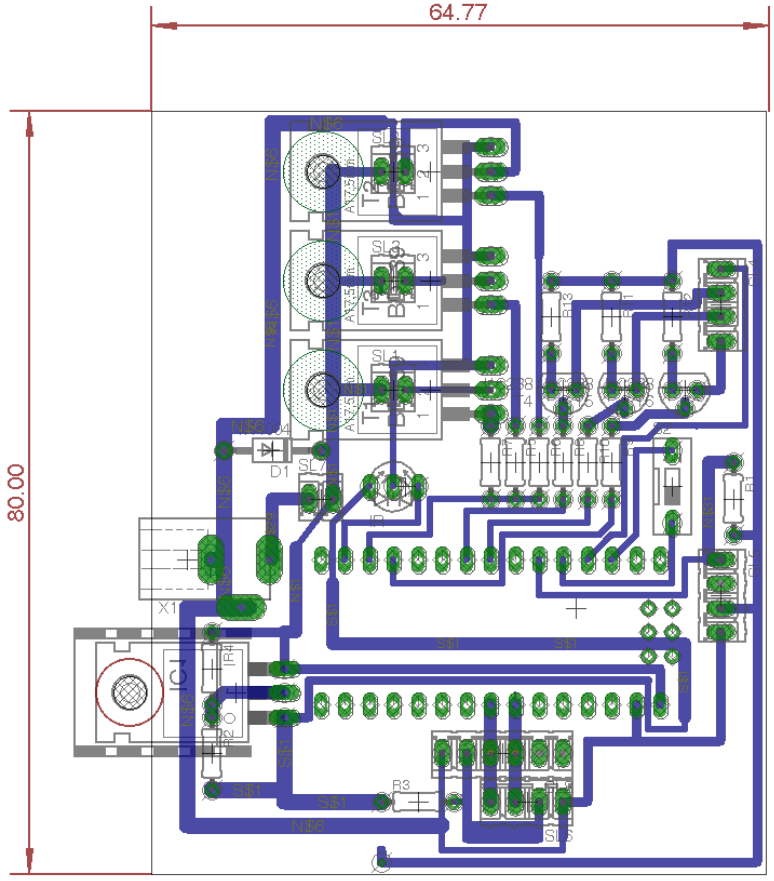
Schéma zapojení jsem nejdříve nakreslil na papír. Poté jsem jej nakreslil v počítači v programu EAGLE 7.7.0. S tímto programem nemám dostatečné zkušenosti, proto to bylo trošku obtížnější a časově náročnější. Potom je zapotřebí schéma převést do formátu pro DPS. S tím mi pomohl můj konzultant Vojtěch Tomala.



Obrázek 6 – Schéma zapojení

Jak lze ve schématu vidět, tak například místo motoru tam jsou LED, protože to bylo moje první schéma v EAGLE a nenašel jsem schématickou značku motorů nebo nějakých konektorů. Na funkčnost to nemá vliv, protože některé součásti jsou připojeny externě. A na DPS stačí mít vývody.

Ačkoli vše je připraveno pro výrobu DPS, ještě jsem tak neučinil, protože mám v plánu ještě nějaké úpravy. Z toho důvodu je vše prozatím zapojeno na nepájivých polích.



Obrázek 7 - Návrh DPS

5 PROGRAM

5.1 Program

Samotný program není napsán nějak složitě, jsou použity spíše jen základní funkce a příkazy. Základem bylo zprovoznit IR komunikaci, což jsem zvládnul s pomocí literatury na internetu. Poté jsem začal programovat regulaci výkonu, postupně jsem přidal i LED. Následně jsem chtěl zobrazovat data na displeji, protože bez notebooku nemám přístup k sériovému monitoru, abych se mohl podívat na data třeba o výkonu či o době provozu. Proto jsem přidal OLED displej. Jako další rozšíření jsem zvolil čidlo teploty a vlhkosti, tyto údaje se rovněž zobrazují na displeji.

5.2 Základní údaje programu[4]

Velikost na disku	13 kB
Využití paměti čipu	90 %
Využití proměnných	40 %
Počet řádků celkem	550

Tabulka 2 Program – údaje (zdroj – Arduino IDE)

[4] Data po skončení kompilace zobrazena v informačním panelu

5.3 Části kódu

5.3.1 Čas provozu

Základem části programu, pomocí které zobrazuji čas provozu je funkce millis(). Tato funkce vrací počet milisekund od spuštění programu. Tuto funkci v průběhu programu nelze restartovat ani ji přiřadit hodnotu.

Popis funkce:

Proměnné *x* jsem přiřadil funkci millis(). Sekundy se rovnají *x – y*. Protože na začátku *y* == 0 tak sekundy se v tomto případě rovnají počtu milisekund.

Když sekundy jsou větší než 60 000, tak se program dostane do podmínky. V této podmínce přičte 1 k proměnné *minuty*. Proměnná *minuty* je počet minut od doby spuštění. Dále v této podmínce přiřadí proměnné *y* hodnotu *x*, kde *x* se stále rovná millis(). A jako poslední se přiřadí *sekundám* hodnota 0. *Sekundy == x – y*, *x* se neustále mění. Ale od něj se odčítá *y*, které má hodnotu 60 001.

Po dalších 60 002 milisekund získá proměnná *sekundy* hodnotu větší než 60 000 a program se dostane znovu do podmínky. Toto se neustále opakuje. Když *minuty* > 60 tak se proměnné *hodiny* přičte 1. Proměnná *hodiny* představuje počet hodin od doby spuštění.

Zprava je proměnná, které přiřazuji data, která chci zobrazit na displeji. A přiřazuji ji všechna data k zobrazení. Takže když se změní obsah displeje, přepíše se zprava. Ovšem Serial.println není vypisování dat na displej, ale na výpis dat na sériový monitor.

```
14 void loop() {
15   Serial.print(" cas od spusteni programu: ");
16   x = millis();
17   sekundy = x - y;
18   if(sekundy > 60000){
19     minuty = minuty+1;
20     y = x;
21     sekundy = 0;
22   }
23   if(minuty > 60){
24     hodiny = hodiny+1;
25     minuty = 0;
26   }
27
28   zprava = hodiny;
29   zprava += ":";
30   if(minuty < 10){
31     zprava += "0";
32   }
33   zprava += minuty;
34   zprava += ":";
35   if(sekundy/1000 < 10){
36     zprava += "0";
37   }
38   zprava += sekundy/1000;
39   Serial.println(zprava);
```

Obrázek 8 - Čas provozu

5.3.2 Světelné funkce

Mám nadefinované celkem čtyři režimy (mody) pro RGB LED. Níže popíši dva z nich.

Popis funkce

První mód se jmenuje RANDOM. Je velice jednoduchý. Proměnným *red, green a blue*, což jsou proměnné, které mají hodnotu úrovně bary. Přiřadím jim funkci random a to v rozsahu 0 až 255.

Funkce random vrátí náhodnou hodnotu v uvedeném rozsahu. Rozsah 0 až 255 je proto, že příkazem analogWrite můžu zadat hodnotu v tomto rozsahu. Takže když mám spuštěný tento mód, tak LED každou sekundu změní barvu, která je náhodná.

Druhý mód je RBG, když je zapnut, tak LED mají červenou, zelenou nebo modrou barvu. Proměnná *x* nabývá náhodné hodnoty od 1 do 3. Barva LED se mění podle aktuální hodnoty *x*. Frekvence změny barev je 0,2 Hz.

```
408 void Random(){
409     red = random(0,256);
410     green = random(0,256);
411     blue = random(0,256);
412 }
413 
414 void Rgb(){ //funkce Rgb
415 int  x = random(1,4);
416   if(x==1){
417     red = 255;
418     green = 0;
419     blue = 0;
420   }
421   if(x==2){
422     red = 0;
423     green = 255;
424     blue = 0;
425   }
426   if(x==3){
427     red = 0;
428     green = 0;
429     blue = 255;
430   }
431 }
```

Obrázek 9 - Světelné funkce

5.3.3 Výpis hodnot

V této krátké části programu je proměnným přiřazena hodnota, která se projeví na výstupu.

Popis funkce

Příkaz analogWrite je zápis hodnoty na výstup, používá se pro PWM regulaci, analogWrite (pin, hodnota); pin je číslo pinu na který má zapsat hodnotu. Do hodnoty mohu zadat číslo od 0 do 255. V mém případe mám čísla pinu uložených v proměnných *redPin, greenPin* atd..

Hodnota, kterou vypisuji je dána proměnnou *red, green*… a proměnou *jas*. Jako příklad uvedu proměnnou *red*. Mám manuální nastavení barvy, zvyšuji či snižuji hodnotu proměnné *red*. A ta hodnota se objeví na výstupu. Je zde, ale i proměnná *jas*. *Jas* je číslo, kterým násobím všechny proměnné barev. Slouží ke snížení celkového jasu nikoli změně jednotlivých barvy. Ve skutečnosti se trochu změní i barva a to i přes to, že poměr barev zůstane zachován. *Jas* nabývá hodnot 0 do 1 a mění se vždy o 0,1. Tato část kódu slouží jen k zápisu hodnot na výstup.

```
383 //výpis hodnot
384 vysledek.value = 0;
385 analogWrite(redPin, red*jas);
386 analogWrite(greenPin, green*jas);
387 analogWrite(bluePin, blue*jas);
388 analogWrite(fanPin, vykon);
389 delay(200);
390
```

Obrázek 10 - Výpis hodnot

5.3.4 Zmáčknutí tlačítka

Co se stane, když zmáčknu nějaké tlačítko na dálkovém ovladači?

Popis funkce

Do proměnné *vysledek.value* se zapíše číslo, které jsem získal z IR čidla. Potom už jenom pomocí podmínek je určeno, co se stane, když zmáčknu zvolené tlačítko.

V tomto případě u příkazu na řádku 199 dojde k tomu, že se na displeji zobrazí data. Pokud bude zmáčknuto dané tlačítko, tak proměnná *ZPRAVA1* = pravda. Jinde v programu je napsané, že pokud *ZPRAVA1* == pravda, tak proměnná, která vypisuje zprávu na displej získá obsah zprávy a ten zobrazí na displeji.

Podmínka na řádku 203. Po zmáčknutí tlačítka na ovladači, konkrétně tlačítka POWER, vypnu všechny součásti jako jsou větráky, OLED a LED. Vypnutí LED je zajištěno tak, že u proměnných *red, green a blue* je přiřazena hodnota 0. Proměnným pro nadefinované módy se přiřadí nepravda, takže se nespustí. U displeje ZPRAVA1 = nepravda, obsah zprávy je prázdný.

```
198
199 if(vysledek.value ==16762935){ //mute zapni displej
200   ZPRAVA1 = true;
201 }
202
203 if(vysledek.value ==16722135){ // POWER - všechno vypni
204   red=0;
205   green=0;
206   blue=0;
207   vykon=0;
208   RANDOM=false;
209   RGB=false;
210   COLOR = false;
211   ZPRAVA1 = false;
212   program = 0;
213 }
```

Obrázek 11 - Zmáčknutí tlačítka

6 FUNKCE CHLADICÍ PODLOŽKY

6.1.1 Dálkové ovládání

Celou chladicí podložku ovládám pomocí dálkového ovladače od rádia a IR přijímače. Přes mikropočítač Arduino nano.

6.1.2 Regulovatelné nastavení výkonu

Mohu regulovat výkon, tím se mění otáčky větráku, a to se projevuje na výsledné teplotě notebooku. Výkon nastavuji dálkovým ovladačem a jeho hodnotu měním vždy o 5 %.

6.1.3 Regulovatelný jas a barva osvětlení

Každou ze základních barev mohu regulovat zvlášť a tím mám širokou škálu možností barev. Jednotlivé barvy nastavuji také po 5 %.

Také můžu regulovat jas. Jas sníží hodnotu všech barev. Takže poměr barev zůstane stejný, a tudíž i barva. Bohužel teorie vždy neplatí v praxi a barva se trochu změní. Funkce podsvícené podložky je spíše jen pro efekt a zábavu.

Je také i praktická funkce. Když vlhkost > 65 %, tak LED budou blikat červeně. Jako upozornění, že vlhkost v místnosti je vysoká. Nebo vlhkost < 35 % tak LED začnou blikat modře.

6.1.4 Teplota a vlhkost

Pomocí čidla DHT22 na displeji zobrazuji také teplotu a relativní vlhkost v místnosti.

6.1.5 Zobrazování dat na displeji

Na OLED displeji zobrazuji data o době provozu, výkonu, jasu, módu, teploty a vlhkosti.

Obrázek 12 - Provoz – displej

Obrázek 13 - Čas provozu – displej

7 MÓDY

Mám nadefinované módy, které se spustí po zmáčknutí nastaveného tlačítka. Dále jsou ještě dvě funkce. První z nich je, že zmáčknutím tlačítka vypnu OLED displej a hodnoty barev = 0. Proto jas může být i 100 % a LED nesvítí. Druhá je kompletní vypnutí.

7.1 Módy pro větráky

- MANUAL

Manuální nastavení výkonu větráku, a to vždy o 5 %.

- 75 %

Po zmáčknutí tlačítka na ovladači konkrétně 0 větráky najedou na 75 % svého výkonu.

7.2 Módy pro RGB LED

- MANUAL

Manuální nastavení každé barvy zvlášť.

- RANDOM

Tento mód nastaví každou sekundu LEDkám jinou barvu a to náhodnou. Barva se mění každou sekundu.

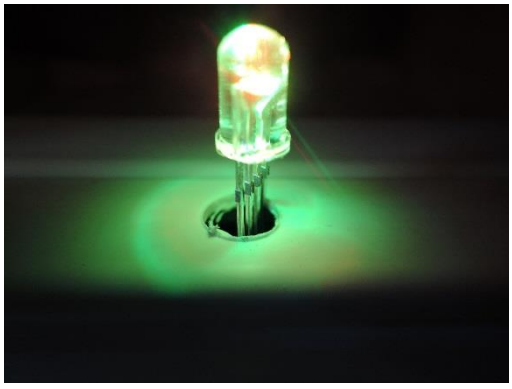
- RGB

Tady mám jen tři barvy a to modrou, červenou a zelenou. Jejich zobrazení je taktéž náhodné.

- COLOR

V tomto módu mají LED nadefinované 14 barev, které se mění po 2,6 sekundách.

Obrázek 14 - RGB



Obrázek 15 - RANDOM

8 MĚŘENÍ ÚČINNOSTI CHLAZENÍ

Měřil jsem teplotu CPU v závislosti na výkonu větráku. Data o teplotě jsem získal ze základní desky a použil jsem k tomu dva programy, a to WHMonitor a Core Temp. Používal jsem dva programy pro dosažení přesnějších výsledků.

8.1 Postup měření

Pracoval jsem na notebooku. A když teplota CPU byla okolo 90 °C, spustil jsem časovač nastavený na 15 minut. Během měření jsem stále pracoval na notebooku. Občas jsem se podíval, jak se teploty vyvíjejí. Po 15 minutách jsem zkontroloval teplotu a chvíli ji sledoval. Poté jsem teplotu zapsal do tabulky. Vypnul jsem chlazení, nechal notebook zahřát a měření jsem opakoval s jiným výkonem. Hodnoty výkonu byly: 0; 25; 50; 75 a 100 %. Pozor! Teplota závisí také na využití CPU. Proto jsem se snažil pracovat na notebooku konstantně. Ale nemůžu zaručit, že výsledky jsou stoprocentní. Proto by to chtělo provést více měření pro dosažení přesnějších výsledků, a to i na jiných zařízeních. Dále by bylo zapotřebí program, který bude dělat vzorky třeba každou sekundu a bude je zapisovat (datalogger). Takové laboratorní podmínky jsem doma však neměl, a proto jsem nemohl provést tato přesnější měření.

Dříve, než vám ukáži graf, sdělím vám svůj názor a zkušenosti, na základě kterých můžu prohlásit, že je moje chladicí podložka funkční a účinná.

8.2 Moje poznatky

8.2.1 První poznatek.

Dříve, když jsem dal restartovat notebook, třeba pro dokončení instalace, notebook se vypnul, ale už znovu nenajel. Nešel ani zapnout asi dalších 5 minut, dokud trochu nevychladl. Nyní, když používám chladicí podložku, jsem se s tímto problémem nesetkal.

8.2.2 Druhý poznatek

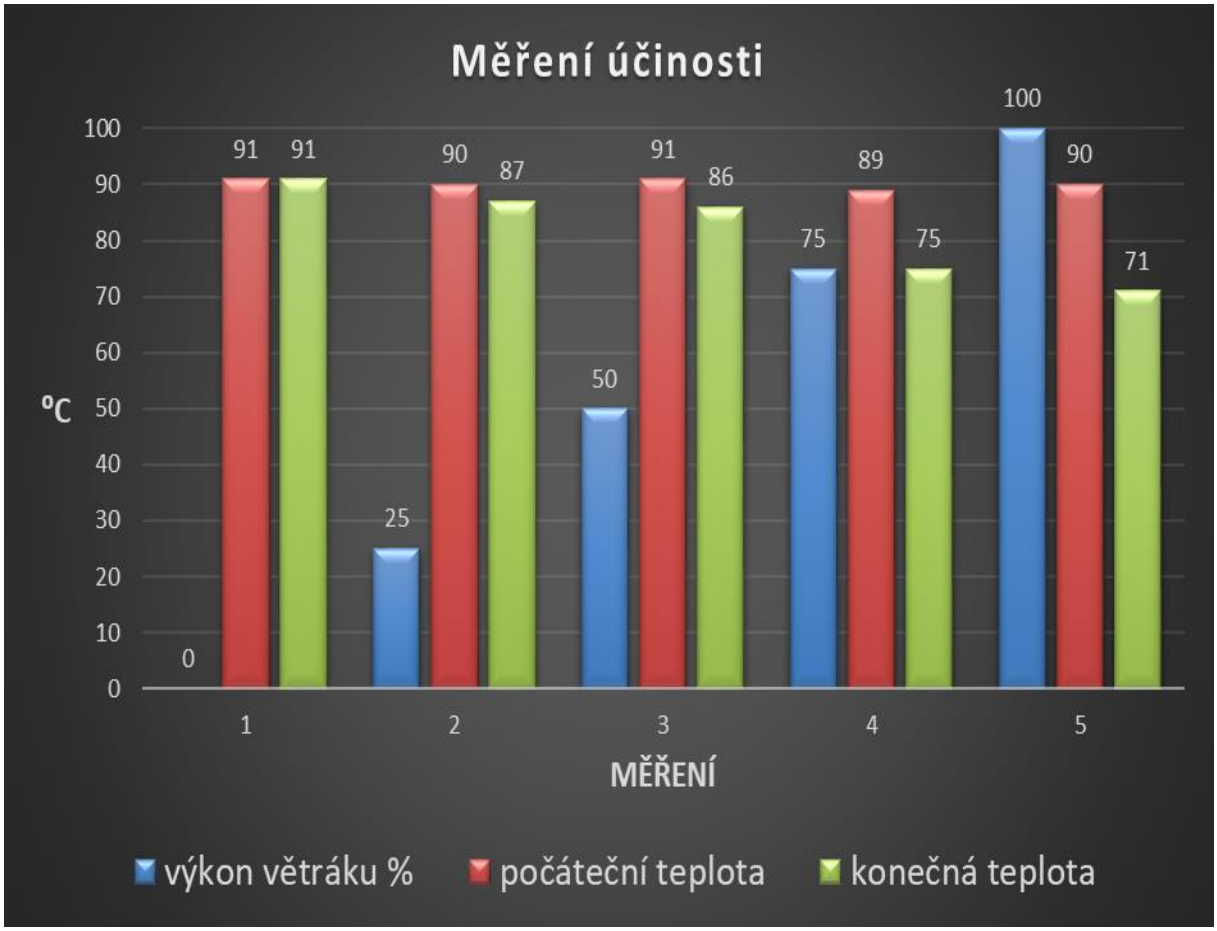
Občas jsem pracoval s notebookem na stehnech a po chvíli jsem musel přestat, protože to pálilo. Je sice pravda, že s chladicí podložkou se moc pracovat na klíně nelze. Ale při chlazení, když přiložím ruku na spodní část notebooku, tak si ruku nespálím a můžu ji tam bez problému držet a není to nepříjemné.

8.2.3 Třetí poznatek

Podle mého názoru notebook pracuje lépe a rychleji. Samozřejmě pokud není CPU využit téměř na 100 %, třeba skrytou aktualizací.

8.3 Graf

V grafu je zaznamenána počáteční teplota měření, konečná teplota po skončení měření a hodnota výkonu, při kterém měření probíhalo. Doba měření trvala 15 minut.


Měření účinosti
100
90
80
70
60
50
40
30
20
10
0
°C
91
91
90
87
91
86
89
75
75
100
90
71
25
50
0
1
2
3
4
5
MĚŘENÍ
výkon větráku %
počáteční teplota
konečná teplota

ZÁVĚR

V této práci jsem vám popsal výrobu a funkce mé chladicí podložky. Mým hlavním cílem bylo snížení teploty procesoru a zvýšení výkonu notebooku. Toto se mi podařilo. Funkčnost chladicí podložky pozoruji na provozu notebooku.

Chladicí podložka zahrnuje i světelné efekty pro zvýšení užitné hodnoty. Praktické využití LED je upozornění na vysokou nebo nízkou hodnotu vlhkosti. Zobrazení dat na displeji je perfektní, hlavně teplota a vlhkost. Dálkové ovládání se mi osvědčilo jako skvělý ovládací prvek. S hlavní funkcí regulovatelného chlazení jsem spokojen, další výhodou je snadno upravitelný program, aby vyhovoval každému uživateli.

Stále mohu tuto chladicí podložku vyvíjet, a to co se týče programu nebo konstrukce. Už mám v plánu další vylepšení např. nastavitelný sklon chladící podložky.

Elektronika a programování jsou obory, které mě baví a rád se v nich vzdělávám i mimo školu. Při práci na tomto projektu jsem získal nové vědomosti, zkušenosti a dovednosti.

POUŽITÁ LITERATURA

1 *Arduino: Co je Arduino* [online]. [cit. 2018-03-26]. Dostupné z: https://arduino.cz/co-je-to-arduino/

2 *Arduino: co je to Arduino* [online]. [cit. 2018-03-26]. Dostupné z: http://czechduino.cz/?co-je-to-arduino,29

3 BARRAGÁN, Hernando. Wiring. *Wiring* [online]. [cit. 2018-03-26]. Dostupné z: http://wiring.org.co/

4 *Dálkové ovládaní infračervené* [online]. [cit. 2018-03-26]. Dostupné z: http://navody.arduino-shop.cz/arduino-projekty/pouziti-jakehokoliv-dalkoveho-ovladace.html

5 *OLED I2C displej 128×32 0,91* [online]. [cit. 2018-03-26]. Dostupné z: http://navody.arduino-shop.cz/navody-k-produktum/oled-i2c-displej-128x32.html

6 *Použití jakéhokoliv dálkového ovladače* [online]. [cit. 2018-03-26]. Dostupné z: http://navody.arduino-shop.cz/arduino-projekty/pouziti-jakehokoliv-dalkoveho-ovladace.html

7 *SNÍMAČ TEPLOTY A VLHKOSTI DHT22* [online]. [cit. 2018-03-26]. Dostupné z: https://www.hwkitchen.cz/snimac-teploty-a-vlhkosti-dht22/

8 *Teploměr a vlhkoměr DHT11 a DHT22* [online]. [cit. 2018-03-26]. Dostupné z: http://navody.arduino-shop.cz/navody-k-produktum/teplotni-senzor-dht11.html

9 *Užitečné funkce: millis()* [online]. [cit. 2018-03-26]. Dostupné z: https://arduino.cz/uzitecne-funkce-3/

10 VODA, Zbyšek a TÝM HW KITCHEN. *PRŮVODCE SVĚTEM ARDUINA* [online]. [cit. 2018-03-26]. Dostupné z: https://arduino.cz/e-book-zdarma/

11 *Wiring: Programovací jazyk* [online]. [cit. 2018-03-26]. Dostupné z: https://cs.wikipedia.org/wiki/Wiring_(programovac%C3%AD_jazyk)

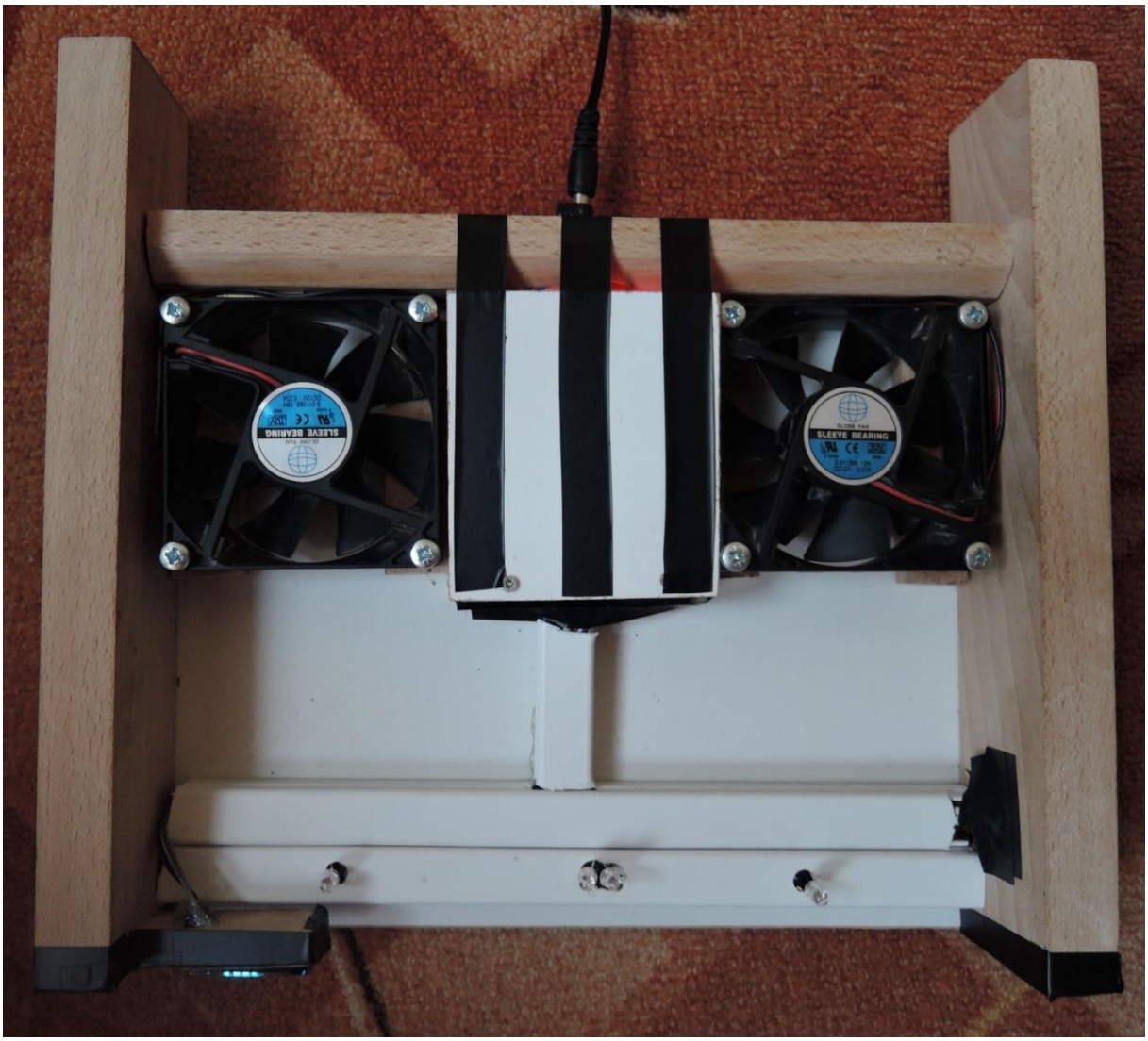
SEZNAM OBRÁZKŮ

SEZNAM TABULEK

PŘÍLOHY

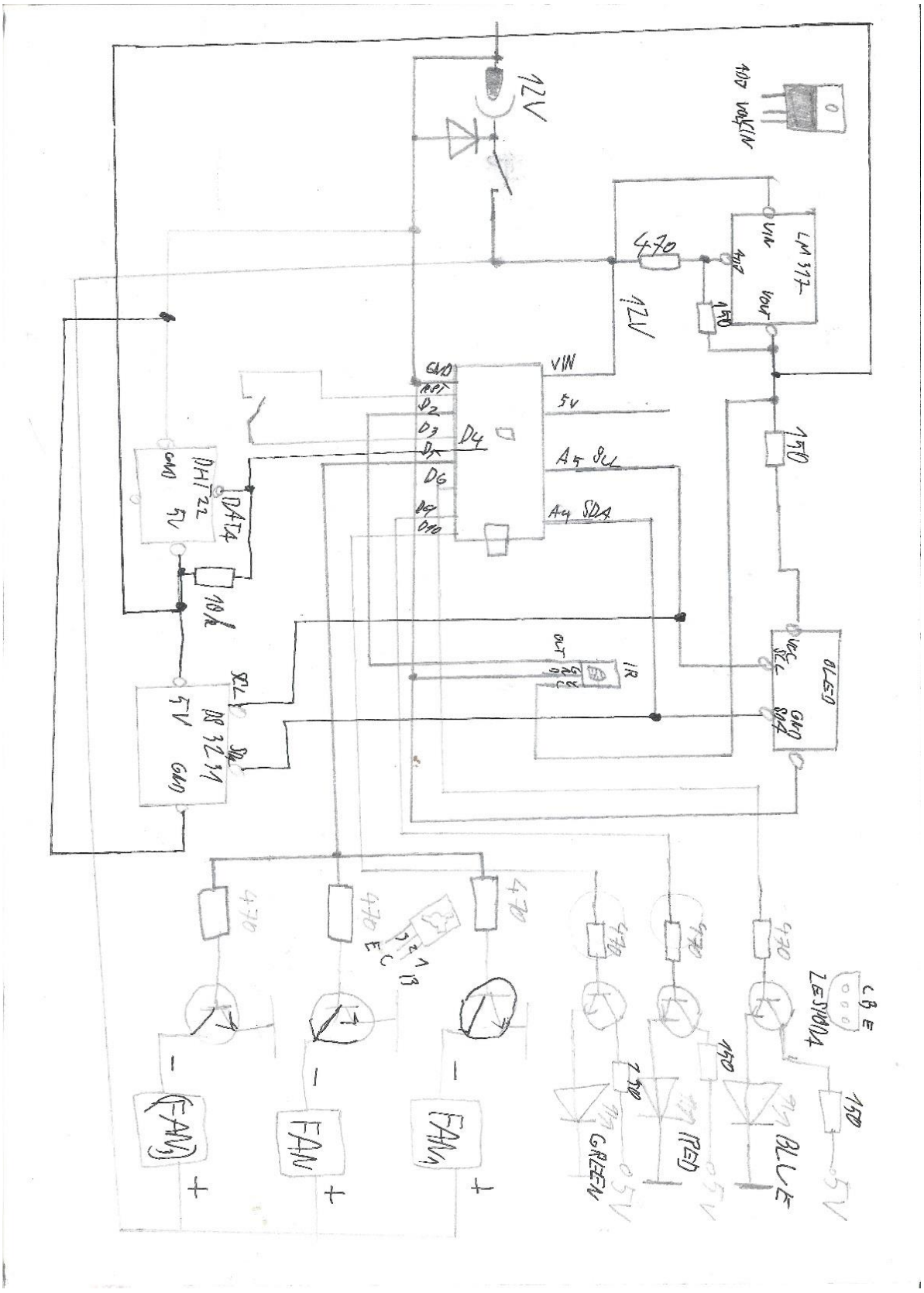
PŘÍLOHA 1 – CHLADICÍ PODLOŽKA

Obrázek chladicí podložky



PŘÍLOHA 2 – SCHÉMA

Ručně nakreslené schéma zapojení



PŘÍLOHA 3 – NÁVRH KONSTRUKCE

Konstrukce navrhnuta v programu Solid Edge ST6