ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTRONIKY A TELEKOMUNIKACÍ

BAKALÁŘSKÁ PRÁCE

Detektor světelného režimu

Jan Čechura **2017**

Originál (kopie) zadání BP/DP

Abstrakt

Cílem bakalářské práce je realizace modulu řízeného mikropočítačem, který kontinuálně měří a vyhodnocuje světelný režim v laboratoři. Kontrola správnosti přednastaveného světelného režimu je nezbytná pro sestavování relevantních závěrů z výzkumu. Detektor byl sestaven za pomoci platformy Arduino a modulů BH1750FVI a DS1307. Software byl naprogramován ve vývojovém prostředí Arduino IDE. Výstup zařízení je dvoustavový. Potřebné parametry zařízení je možno nastavit v servisním módu prostřednictvím USB přes sériový port. Přínosem této práce je funkční detektor světelného režimu, který lze uplatnit v praxi.

Klíčová slova

Arduino ATmega2560, BH1750FVI, DS1307, Zabezpečení světelného režimu

Abstract

The goal of this bachelor thesis is the realization of the microcomputer controlled module which continuously measures and evaluates the light mode in the laboratory. Checking the precision of the preset light mode is essential to develop relevant research findings. The detector was built using the Arduino platform and the BH1750FVI and DS1307 modules. The software was programmed in the Arduino IDE development environment. The output of the device is two-state. The required device parameters can be set in service mode via USB through the serial port. The benefit of this work is a functional light detector that can be applied in practice.

Key words

Arduino ATmega2560, BH1750FVI, DS1307, Lighting control system

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

.....

podpis

V Plzni dne 6.6.2017 Jan Čechura

Obsah

Úvod

Předkládaná práce se zaměřuje na návrh a následnou realizaci detektoru světelného režimu. Cílem hotového zařízení je vyhodnotit, zda-li je světelný režim v konkrétním prostoru v souladu s předem definovaným světelným a časovým režimem. Při neshodě přednastaveného režimu s aktuálním, vyšle zařízení signál do periferie, která uživateli odešle informativní sms nebo uskuteční telefonát v závislosti na nastavení. V první části je popsána teorie pro měření světla a platforma Arduino. V druhé části je popsáno hardwarové a softwarové řešení.

Detektor světelného režimu je vlastně forma zabezpečovacího systému, který upozorní uživatele, je-li v daném místě opravdu světlo nebo tma v závislosti na jeho předvolbě. Technické řešení bylo navrženo s ohledem na požadavek spolupráce s externím modulem Jablotron GD-04, který je k dispozici v místě, kde se bude detektor světelného režimu provozovat. Potřeba vytvoření detektoru světelného režimu vychází z požadavku dodržet a kontrolovat přesné laboratorní podmínky, které jsou nezbytné pro výzkum praktikovaný na živých zvířatech či rostlinách. Experimenty a výzkum v laboratoři probíhají autonomně, není prakticky možné, aby byl světelný režim kontrolován nepřetržitě některým z pracovníků. Díky potřebě kontroly a záruk, že světelný režim probíhá bezchybně dle nastavení, vznikla poptávka po zařízení, které světelný režim vyhodnocuje a v případě nesouladu s režimem přednastaveným, upozorní v součinnosti s externím modulem David GD-04 odpovědnou osobu. Prostředí provozu realizovaného zařízení je uzavřená laboratoř s umělým osvětlením, bez přístupu denního světla. Zařízení bylo zrealizováno s ohledem na požadavky pracovníků laboratoře. Umožňuje běžné uživatelské funkce jako je nastavení světelného režimu, nastavení data a času, nastavení ochranného limitu spuštění alarmu a výpis aktuálního nastavení. Samozřejmostí je uložení nastavení do paměti. Obsahuje také tlačítko pro přepnutí mezi tzv. aktivním režimem, kdy hlídá aktuální světelný režim a neaktivním režimem, kdy je detektor v nečinnosti a lze jej případně nastavit po sériové lince. Dále obsahuje stavovou diodu, která indikuje aktivní či neaktivní režim. Zařízení je schopno uchovávat nastavení po dobu výpadku proudu a je uzpůsobené na provoz z externího síťového adaptéru. Poptávka po zařízení tohoto typu, které provádí kontrolu přednastaveného světelného režimu, vznikla také díky nedostupnosti podobného

zařízení na trhu. Takto specifické zařízení by pravděpodobně nenašlo masové nebo komerční využití, avšak může být vhodným doplňkem při chovu zvířat a pěstování rostlin pod umělým osvětlením nebo například při výše zmíněném laboratorním výzkumu. Předností realizovaného zařízení je také cena, která v součtu za všechny komponenty nepřevýšila 1000 Kč. V porovnání s realizací od specializované firmy, kde se částka za realizaci pohybuje kolem 20 000Kč, je tato cena pouze zlomkem nákladů.

Seznam symbolů a zkratek

AAA..... Velikostní varianta galvanických článků

ABS..... Akrylonitrilbutadienstyren, druh plastu

AD..... Analagově-digitální převodník

ARM ..... Označení architektury procesorů

COM ..... Označení konektoru fáze relé

DC..... Stejnosměrný proud

E ..... Intenzita osvětlení [$lx \cdot m^{-2}$]

F ..... Frekvence [Hz]

EEPROM ..... Elektronicky vymazatelná paměť pro čtení

GND..... Označení zemního vodiče

GSM..... Globální systém pro mobilní komunikaci

I ..... Elektrický proud [A]

I2C ..... Komunikační sběrnice

IDE..... Integrované vývojové prostředí

NC..... Rozpínací kontakt relé

NO..... Spínací kontakt relé

NV SRAM ..... Nevolatilní statická paměť

PC.....Osobní počítač

PN .....Rozhraní polovodiče typu P a N

RISC..... Typ architektury mikroprocesorů

RS-232 ..... Standard sériové linky

RTC..... Hodiny reálného času

RX..... Označení pinu pro přijímání po sériové lince

S .....Osvětlená plocha [m^2]

SMS ..... Krátká textová zpráva

SRAM ..... Statická paměť

TÜV ..... Technické kontrolní sdružení

TX ..... Označení pinu pro odesílání po sériové lince

UART..... Univerzální asynchronní přijímač/vysílač

USB..... Univerzální sériová sběrnice

U..... Elektrické napětí [V]

V-A ..... Volt-Ampérová charakteristika

ϕ ..... Světelný tok [lm]

1. Měření intenzity osvětlení

Světlo je nezbytná fyzikální veličina pro všechny životní formy na planetě Zemi. Patří k důležitým environmentálním faktorům a bez světla by příroda nemohla existovat. Viditelné světlo pro člověka je elektromagnetické záření o vlnové délce v intervalu od 400 nm do 700 nm. Existuje několik různých technických součástek, které reagují na světlo, zkráceně se označují jako fotosoučástky. Jsou jimi fotorezistor, fotodioda, lavinová fotodioda, fototranzistor, fototyristor a optron. Činnost těchto součástek je založena na vnitřním fotoelektrickém jevu. Při dopadu světla o vhodné vlnové délce na povrch optického přijímače součástky dochází na polovodičové bázi k přerušení vazeb atomů a vznikají volné nosiče, které se přemění na elektrický náboj. Čím více světla narazí na povrch, tím větší náboj je vytvořen. Základní princip měření intenzity osvětlení spočívá v konvertování dopadajícího světelného záření na elektrický proud nebo napětí. Základní jednotka intenzity osvětlení je 1 lx. Intenzita světelného toku je dána vztahem (1).

$$E = \frac{d\Phi}{dA} \tag{1}$$

kde

E - intenzita v $lx \cdot m^{-2}$

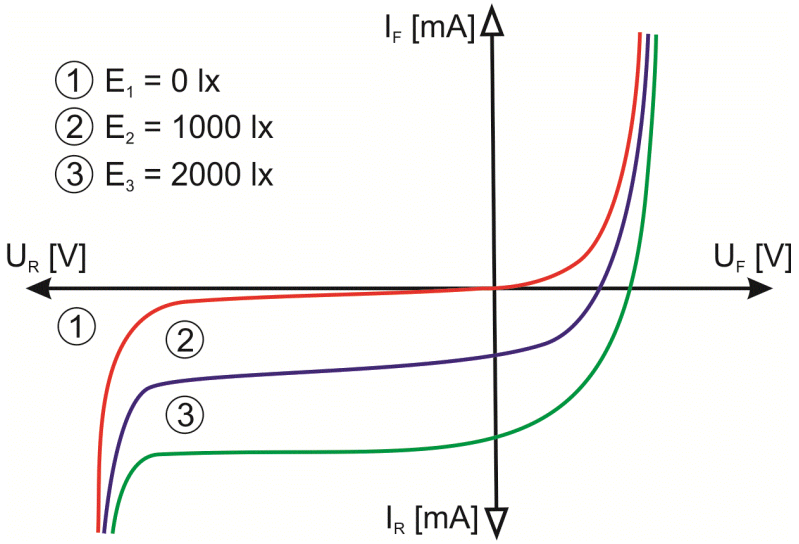
Φ - světelný tok v lm

S - osvětlená plocha v m^2

1.1 Fotodioda

Při realizaci předkládaného vzorku byl použit modul BH1750FVI, který je osazen fotodiodou. Princip fotodiody je odvozen od klasické diody s PN přechodem. Oproti klasické diodě je však umožněno světlu dopadat na vlastní PN přechod, kde jsou excitované elektrony z valenčního pásu polovodiče přesunuty do vodivostního pásu a tím se zvětšuje vodivost PN přechodu. Tento efekt se přesněji nazývá hradlový

fotoefekt. Výsledná V-A charakteristika fotodiody je pak závislá na intenzitě osvětlení. Když není PN přechod osvětlen, chová se fotodioda jako klasická dioda. Vlastní excitace elektronů probíhá při nulové intenzitě osvětlení přechodu pouze a jen díky tepelné energii. Tento parametr se také označuje jako temný proud. S narůstající intenzitou osvětlení PN přechodu se lineárně zvyšuje vodivost PN přechodu v závěrném směru, což ilustruje obr. 1.



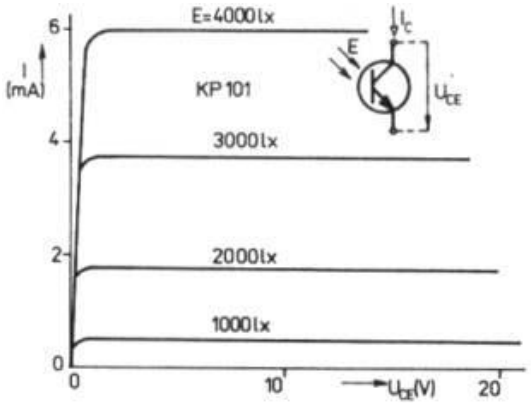
Obr. 1 V-A charakteristika fotodiody [1]

Při aplikaci fotodiod v praxi využíváme pouze III. a IV. kvadrant. Ve III. kvadrantu se využívá jako rezistor citlivý na světlo. Ve IV. kvadrantu chování diody vychází z již zmíněného hradlového fotoelektrického efektu a chová se jako zdroj elektrické energie. Fotodiody byly v historii využívány zejména v aplikacích s děrnými štítky. Později přichází použití v audiovizuální technice, bezpečnostních zařízeních, optických komunikací, snímačů polohy aj.

Obr. 2 Schematická značka diody

1.2 Fototranzistor

Fototranzistor je polovodičová elektronická součástka, která přeměňuje světlo na elektrický proud. Funguje jako spínač nebo proudový zesilovač, jehož činnost závisí na působení světelného záření. Dopadající světelné záření na přechod báze-emitor ovlivňuje velikost kolektorového proudu, který je tím větší, čím je větší dopadající záření, které obvykle prochází přes čočku zabudovanou v pouzdře tranzistoru. V podstatě se jedná o bipolární tranzistor, který zpravidla nemá vyvedenou bázi. Přechod báze-emitor funguje na stejném principu jako fotodioda, kde volné částice rekombinují při dopadu záření a proud báze je zesilován v závislosti na intenzitě osvětlení. Fototranzistory se používají k detekci světelného záření a přeměně záření na digitální elektrické signály. V porovnání s fotodiodou je reakční doba fototranzistoru menší avšak citlivost větší. V porovnání s klasickým bipolárním tranzistorem je plocha báze a kolektoru větší. Pro zvětšení citlivosti se tranzistory vyrábějí v tzv. Darlingtonově zapojení. Používají se prakticky ve všech elektronických zařízeních, které reagují na světlo. Všechny křemíkové fotoelektrické snímače (fototranzistory) reagují na celý rozsah viditelného i infračerveného záření. Nevýhodou fototranzistorů je relativně nízká mezní frekvence, která činí 50 kHz.



Obr. 3 charakteristika fototranzistoru[2]

Obr. 4 schematická značka fototranzistoru

2 Platforma Arduino

Mikrokontroléry Arduino jsou fenoménem posledních let. Prvopočátky Arduina se datují na území Itálie v roce 2003, kde student Hernando Barragán pracoval na své diplomové práci na fakultě Interaction Design Institute Ivrea. Jeho cílem bylo ulehčit práci designérům a umělcům pracujících s elektronikou. Začal pracovat na zařízení, které neslo označení Wiring. Wiring byl zpočátku komerčně úspěšný, prodej byl v globálním měřítku relativně velký. Později v roce 2005 se od projektu Wiring odpoutalo několik osob a založili vlastní projekt s názvem Arduino. Projekt Wiring skončil. Zajímavostí je, že jméno Arduino, bylo přejato od jména baru, kde se zakladatelé projektu často potkávali. Následné spory uvnitř skupiny zakladatelů projektu Arduino vedly k rozdělení na dvě skupiny vývojářů. První skupina je označena jako Arduino SRL, která provozuje web www.arduino.cc. Druhá skupina Arduino LLC provozuje web www.arduino.org.

Arduino funguje na bázi open-source platformy a těžiště úspěchu spočívá v silném marketingu a sociální inovaci. Statisíce uživatelů na fórech tvoří části kódů, které nezištně sdílejí na webových fórech. Arduino je jak open-source hardware tak open-source software platforma. Tvůrci Arduina pod licencí *Creative Commons Attribution Share-Alike* poskytli všechny potřebné informace k tvorbě Arduina samotného. Přednost platformy Arduino je především parametr cena/výkon, proto je určen spíše pro nekomerční využití. Arduino je předurčeno především pro designéry, umělce, je optimální volbou pro domácí hobby projekty, studijní účely a v neposlední řadě pro rychlé prototypování, výrobě proof-of-concept řešení. Z licenčních podmínek produktů Arduina vyplývá, že by nemělo být použito tam, kde jsou vysoké nároky na bezpečnost. Čili všude tam, kde by na zařízení závisely lidské životy a mohly být případným vyřazením funkcionality ohroženy. Vyloučeno je použití ve zdravotnictví, leteckém, vojenském a energetickém průmyslu, kde jsou nároky na bezpečnost extrémně vysoké.

K vytváření programů pro Arduino slouží jednoduché integrované vývojové prostředí IDE. Toto softwarové prostředí, kromě psaní aplikací, umožňuje kompilaci programů a následné nahrávání zkompilovaného kódu do desky Arduino. K programování Arduino aplikací lze využít programovací jazyku C nebo C++. Je možno také využít knihovnu Wiring, což bývá uživatelsky nejpřijatelnější. V některé literatuře se o ní píše jako o samostatném jazyku, díky její komplexnosti. IDE umožňuje importaci knihoven, což vývojářům usnadňuje práci, odpadá tak nutnost psaní vlastních knihoven k již zakoupeným periferiím či jiným součástkám.

V současné době je na trhu k dispozici 22 různých verzí Arduina. Arduino je vlastně jednodeskový počítač využívající zpravidla mikrokontroléry ATmega od společnosti Atmel. Na trhu je k dispozici pestrá škála desek s různým výkonem a různými parametry. Jak lze vidět z tabulky 1, procesory u nejslabších verzí jsou taktovány na 8 MHz, nejvýkonnější verze disponují 32-bitovým procesorem taktovaného na 400MHz. Další předností Arduina je velký počet vstupních a výstupních pinů, díky kterým je možno připojit velké množství periférií.

Téměř všechny mikroprocesory, obdobných jednodeskových mikropočítačů využívají architektury ARM. Za architekturou procesorů ARM stojí společnost ARM holding. Zkratka ARM pochází z obchodního názvu Advanced RISC machine. Procesory architektury ARM jsou v 90% mobilních zařízení celosvětově a v mobilních počítačích je to 31%. [3] Další ze známých mikrokontrolérů jsou „Raspberry“ a „STM“. „Raspberry“ je výkonnější než Arduino a vyznačuje se tím, že má standardní konektory jako klasické výkonově slabší PC. Také disponuje operačním systémem na linuxovém jádře. Ve srovnání s Arduinem je dle odborné veřejnosti „Raspberry“ spíše mikropočítačem a Arduino spíše mikrokontrolérem. V „Raspberry“ spouštíme vytvořený program pod pevně daným operačním systémem, kdežto u Arduina nahráváme celý operační program do paměti. Mikrokontroléry STM jsou hojně využívány pro komerční účely, v roce 2015 dosahovaly prodeje mikrokontrolérů STM 12,5% trhu.[4]

Name	Processor	Operating/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [kB]	SRAM [kB]	Flash [kB]	USB	UART
101	Intel® Curie	3.3 V/ 7-12V	32MHz	6/0	14/4	-	24	196	Regular	-
Gemma	ATtiny85	3.3 V / 4-16 V	8 MHz	1/0	3/2	0.5	0.5	8	Micro	0
LilyPad	ATmega168V ATmega328P	2.7-5.5 V / 2.7-5.5 V	8MHz	6/0	14/6	0.512	1	16	-	-
LilyPad SimpleSnap	ATmega328P	2.7-5.5 V / 2.7-5.5 V	8 MHz	4/0	9/4	1	2	32	-	-
LilyPad USB	ATmega32U4	3.3 V / 3.8-5 V	8 MHz	4/0	9/4	1	2.5	32	Micro	-
Mega 2560	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Micro	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
MKR1000	SAMD21 Cortex-M0+	3.3 V/ 5V	48MHz	7/1	8/4	-	32	256	Micro	1
Pro	ATmega168 ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	0.512 1	1 2	16 32	-	1
Pro Mini	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	1	2	32	-	1
Uno	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/6	1	2	32	Regular	1
Zero	ATSAMD21G18	3.3 V / 7-12 V	48 MHz	6/1	14/10	-	32	256	2 Micro	2
Due	ATSAM3X8E	3.3 V / 7-12 V	84 MHz	12/2	54/12	-	96	512	2 Micro	4
Esplora	ATmega32U4	5 V / 7-12 V	16 MHz	-	-	1	2.5	32	Micro	-
Ethernet	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/4	1	2	32	Regular	-
Leonardo	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
Mega ADK	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Mini	ATmega328P	5 V / 7-9 V	16 MHz	8/0	14/6	1	2	32	-	-
Nano	ATmega168 ATmega328P	5 V / 7-9 V	16 MHz	8/0	14/6	0.512 1	1 2	16 32	Mini	1
Yún	ATmega32U4 AR9331 Linux	5 V	16 MHz 400MHz	12/0	20/7	1	2.5 16MB	32 64MB	Micro	1
Arduino Robot	ATmega32u4	5 V	16 MHz	6/0	20/6	1 KB (ATmega32u4)/ 512 Kbit (I2C)	2.5 KB (ATmega32u4)	32 KB (ATmega32u4) of which 4 KB used by bootloader	1	1
MKRZero	SAMD21 Cortex-M0+ 32bit low power ARM MCU	3.3 V	48 MHz	7 (ADC 8/10/12 bit)/1 (DAC 10 bit)	22/12	No	32 KB	256 KB	1	1

Tab. 1 seznam desek Arduino [5]

3 Realizace

3.1 Návrh hardwarového řešení detektoru

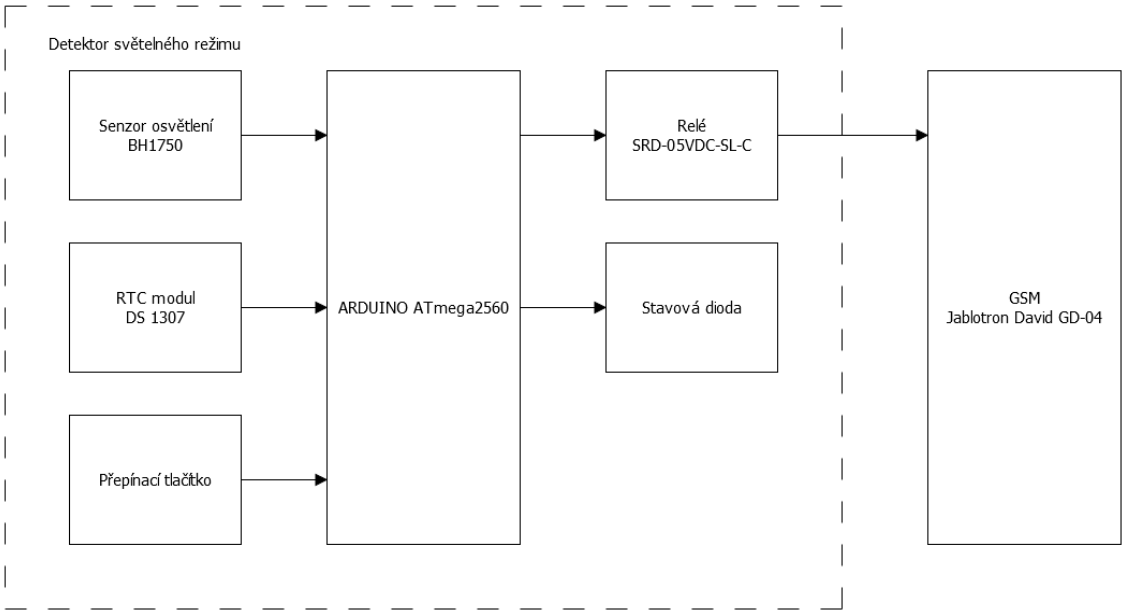
Při návrhu architektury systému detektoru byly uvažovány rozdílné požadavky. V zásadě je můžeme rozdělit do dvou skupin. Požadavky ze strany uživatele na jednoduchou ovladatelnost a spolehlivost v daném místě, v tomto případě laboratoř a omezené možnosti výrobních kapacit tvůrce. Návrh řešení je tedy kompromisem mezi těmito dvěma skupinami kladených nároků na technické řešení daného problému. *„Z praktických zkušeností se zjistilo, že pro pocit spokojenosti s vytvořeným řešením musí být splněno alespoň 80% vznesených požadavků každé zájmové skupiny“* [6]

K realizaci předkládaného vzorku detektoru světelného režimu byl využit jednodeskový počítač Arduino s mikrokontrolérem ATmega 2560 od firmy ATmel (dále jen Arduino), který díky svému bohatému hardwarovému vybavení naprosto postačuje požadavkům řešeného problému. Arduino Mega 2560 disponuje osmibitovým procesorem a je taktován na 16 MHz, což dostatečně obslouží periferie a vyhodnocování času světelného režimu. Toto Arduino bylo zvoleno také díky svému USB konektoru, který zajistí snadnou uživatelskou přístupnost zařízení ve spolupráci s UART portem, prostřednictvím textového menu. V prvopočátcích projektu byl pro komunikaci se zařízením zamýšlen jednoduchý dotykový displej, který by uživatelům ulehčil práci, avšak po důkladném zvážení bylo od této myšlenky upuštěno. Jednak by cena detektoru vzrostla o přibližně 50% a dále pak nastavení v místě laboratoře bude probíhat maximálně dvakrát do roka.

Světelný režim je definovaný dle času, proto bylo do projektu nezbytné zahrnout nezávislý RTC modul. Nejvhodnější periferií k Arduinu byl shledán modul reálného času DS1307, který pro komunikaci využívá sběrnici I2C. Předností modulu DS1307 je nízká spotřeba energie, dle výrobce na jednu kvalitní baterii vydrží pracovat až 9 let. Pro samotné vyhodnocení režimu „světlo“ a „tma“ byl použit jednoduchý senzor osvětlení BH1750, který využívá principu fotodiody. Rozlišení senzoru je vysoké, proto lze v případném vylepšení zařízení uvažovat o rozšíření, kdy detektor bude vyhodnocovat samotnou intenzitu osvětlení v jednotkách lux. Senzor osvětlení, taktéž jako modul DS1307, komunikuje po sběrnici I2C. Zařízení pro indikaci stavu využívá dvoubarevné diody, kdy zelená značí aktivní režim a červená značí neaktivní

režim, ve kterém je možné spustit uživatelské rozhranní. Pro tuto funkci je zařízení vybaveno přepínacím tlačítkem. Dvoustavový výstup zařízení je řešen jednoduchým relé, kde je řídící a výkonový obvod spojen optoelektrickou vazbou.

V prvopočátku projektu bylo zamýšleno zařízení napájet adaptérem ze sítě přes USB konektor. Po dokončení zařízení se však jako efektivnější varianta jevilo napájení přímo z adaptéru pro zařízení Jablotron GD-04. Adaptér s výstupním napětím 12 V a maximálním proudem 500 mA, umožňuje napájet jak detektor, tak zařízení Jablotron GD-04. Jelikož tyto dvě zařízení pracují v cílovém prostoru společně, jeví se tato varianta napájení jako nejefektivnější.



Obr. 5 Blokové schéma detektoru

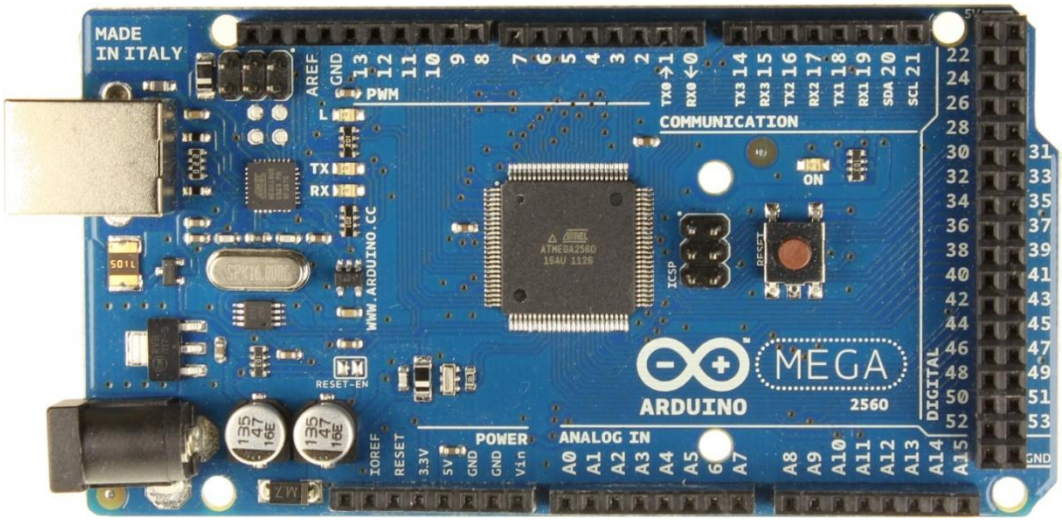
3.2 Arduino ATmega 2560

Pro realizaci detektoru světelného režimu byl vybrán mikropočítač Arduino ATmega 2560. Je vhodným kompromisem mezi cenou a výkonem. Mezi jeho přednosti patří open-source vývojové prostředí a pestrá škála integrovaných funkcí. Před psaním samotného kódu nebylo předem známo detailní řešení jednotlivých funkcí detektoru, proto byla zvolena verze ATmega 2560 s přibližně průměrnými parametry výkonu, které se po dokončení realizace jeví jako mírně naddimenzované. Je využito celkem

pouze 3% Flash paměti a 18% SRAM paměti. Toto zařízení by se tedy dalo zrealizovat za ještě menších nákladů s využitím méně výkonného mikropočítače z řad Arduina, například ARDUINO Uno. Mikropočítač je ve finální verzi napájen 12 V z externího adaptéru k zařízení Jablotron GD-04 přes vstupní napájecí piny. Komunikaci s počítačem zajišťuje port USB typu B. Napájecí konektor nebyl nevyužit. Z integrovaných hardwarových prostředků byl využit UART pro nastavování detektoru. Dále pak paměť EEPROM, díky které lze uložit nastavení světelného režimu, které je zde uchováno pro případný výpadek napájení. Samotné Arduino je vybaveno čtyřmi otvory pro šroubky jak je patrné z obr. č. 5 a s pouzdrem detektoru je spojeno čtyřmi šrouby.

Mikrokontrolér	ATmega2560
Operační napětí	5 V
Vstupní napětí (doporučené)	7-12 V
Vstupní napětí (limitní)	6-20 V
Digitální I/O piny	54 (15 PWM výstup)
Analogové vstupní piny	16
DC proud I/O pinu	20 mA
DC proud 3.3V pinu	50 mA
Flash paměť	256 KB / 8 KB použita bootloaderem
SRAM	8 KB
EEPROM	4 KB
Frekvence procesoru	16 MHz
Délka	101.52 mm
Šířka	53.3 mm
Hmotnost	37 g

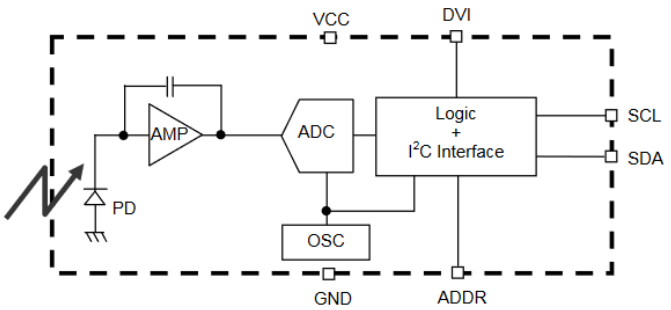
Tab. 3 Technické parametry Arduino ATmega2560



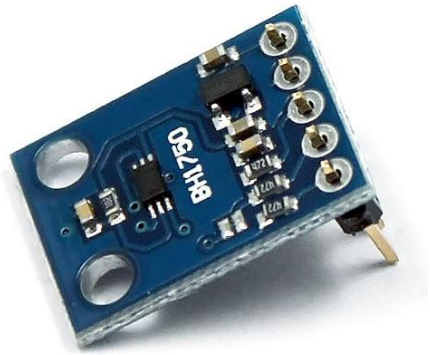
Obr. 6 Arduino ATmega2560[7]

3.3 Senzor osvětlení BH1750FVI

Senzor BH1750FVI byl vybrán pro aktuální rozpoznání intenzity osvětlení. Senzor umožňuje rozpoznávat intenzitu osvětlení ve vysokém rozlišení, a sice 1 až 65535 lx. K senzoru byla využita open source knihovna, která byla mírně upravena pro kompatibilitu s mikropočítačem Arduino ATmega 2560. Tento senzor využívá principu fotodiody, která má přibližně stejnou odezvu na světelné záření jako lidské oko. Integrovaný operační zesilovač dále převádí proud diody na napětí, které je AD převodníkem převáděno na číslicovou hodnotu. Senzor využívá pro komunikaci s mikropočítačem sběrnici I2C. Blokové schéma senzoru je na obr. č. 7.



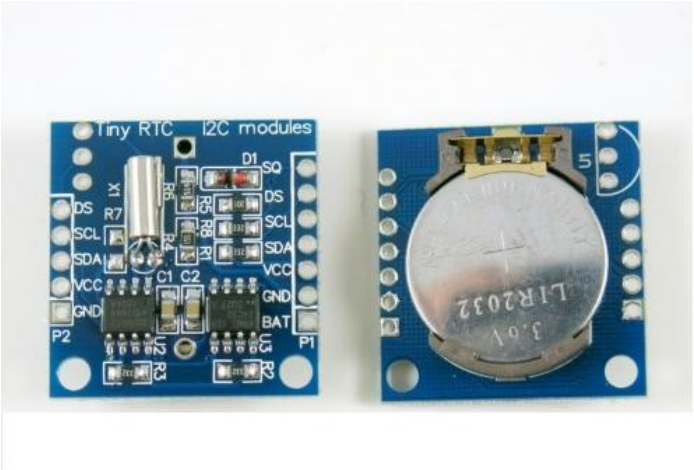
Obr. 7 Blokové schéma senzoru BH1750FVI[8]



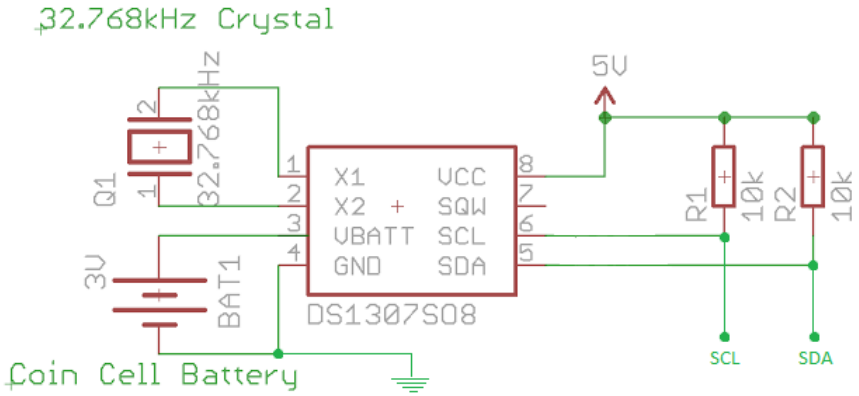
Obr. 8 Senzor BH1750FV[9]

3.4 Modul reálného času DS1307

Jedním z úskalí realizace detektoru světelného režimu byla implementace funkce reálného času. Po krátké rešerši možností jak zajistit modulu přesný čas, se jevila varianta zakoupení modulu DS1307 jako nejefektivnější. Modul DS1307 nabízí za relativně nízkou cenu okolo 100Kč poměrně slušný výkon. Proud modulu v chodu na záložní baterii je menší než 500nA, výrobce uvádí na jednu plně nabitou baterii výdrž až 9 let. Modul operuje jak s časem, tak s přesným datem s přídavnou korekcí na přestupný rok. Čili kromě reálného času lze z modulu načítat i přesný datum, čehož bylo v bakalářské práci taktéž využito. Tento modul pracuje s krystalovým oscilátorem taktovaným na 32.768 kHz a modul DS1307 lze využít jako programovatelný generátor obdélníkových kmitů, což však nebylo v realizovaném projektu využito. Další předností modulu je vnitřní 56 bajtová paměť NV SRAM. Tato paměť oproti standardní paměti SRAM umožňuje uchovávat data i při ztrátě napájení. Modul lze provozovat od -40 °C do +85°C, což plně uspokojí potřeby chodu v laboratoři. Komunikaci s mikropočítačem zajišťuje sběrnice I2C. Operace s modulem DS1307 zajišťuje open-source knihovna stažená z webových stránek www.arduino.cc.



Obr. 9 RTC modul DS1307[10]



Obr. 10 Senzor BH1750FVI[11]

3.5 Relé SRD-05VDC-SL-C

Upozornění na nedodržení světelného režimu je dáno technickými možnostmi zařízení Jablotron DAVID GD-40, který je k dispozici v místě provozování detektoru. Vstupní svorky tohoto zařízení při spojení se svorkou GND vybudí v zařízení přednastavenou formu upozornění. Na tyto svorky se nesmí přivádět žádné vnější napětí - musí být ovládány bezpotenciálovým spínačem (kontaktem).[12] Nejjednodušší tedy při realizaci zařízení bylo pořízení jednokanálového relé, které obsahuje svorky na

šroubky. Tím odpadla starost vodiče do relé při vývoji opakovaně pájet. Relé je přepínací s klasickými porty NO, NC a COM. Technické parametry jsou uvedeny v tabulce č. 3. Výkonový obvod relé je od řídícího obvodu oddělen optoelektrickou vazbou. Toto relé má certifikaci TÜV a vyhovuje standardu ISO9002.

Operační napětí	5 V
Budící proud	90 mA
Výstup	AC250V 10A ; DC30V 10A
Spotřeba	0,45W
Délka	4,7 cm
Šířka	2,9 cm
Výška	1,8 cm
Hmotnost	10 g

Tab. 4 - Parametry relé SRD-05VDC-SL-C

Obr. 11 relé SRD-05VDC-SL-C[13]

3.6 Pouzdro detektoru

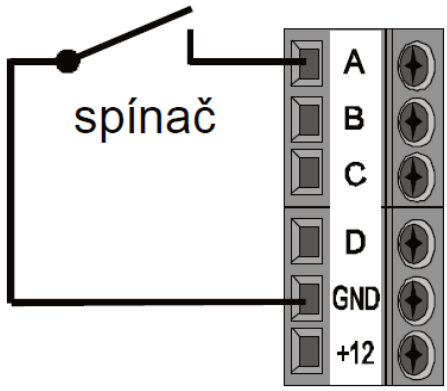
V prvotních fázích vývoje byla jako pouzdro detektoru zamýšlena standardní neprůhledná plastová krabička. Při úvahách jak přivádět světlo okolního prostředí k fotodiodě senzoru bylo rozhodnuto pro krabičku s čirým obalem. Takto odpadla nutnost vytvořit do pouzdra díru pro upevnění čočky nad fotodiodu. Do krabičky byl pouze vyříznut otvor pro USB a vyvrtán otvor pro dvoubarevnou diodu a přepínací tlačítko a následně do podstavy vyvrtán otvor pro kabelovou průchodku. Plastový čirý kryt je pevně spojen čtyřmi šroubky s podstavou, na které je přichyceno Arduino. Rozměry krabičky jsou 143x83x36mm a je vyhotovena z termoplastického kopolymeru ABS.

Obr. 12 Pouzdro detektoru 052C ABS [14]

3.7 Jablotron David GD-04

GSM ovladač a hlásič Jablotron David GD-04 je jediná komponenta v předkládaném projektu, která pracuje jako samostatná jednotka. Jedná se o zabezpečovací zařízení, které disponuje GSM modulem a lze díky němu realizovat upozornění prostřednictvím sms a telefonátu. Obsahuje čtyři vstupní svorky, které upozorní uživatele na vzniklou událost při propojení se svorkou GND. Dále obsahuje dvě výstupní relé, které na přednastavenou sms či telefonát zareagují sepnutím či rozepnutím. Relé lze ovládat i zadáváním z klávesnice mobilního telefonu při aktivním hovoru s tímto modulem. V zařízení je slot na sim kartu, díky které lze uskutečnit

upozornění. Nastavení probíhá za pomoci webového formuláře na stránkách výrobce nebo přímo za pomoci programovacích sms. Modul obsahuje i záložní akumulátor, který udrží zařízení v chodu při výpadku síťového napájení. Při chodu na zálohovací akumulátor není napájení vyvedeno na svorku +12 V. To znamená, že při výpadku napájení není detektor napájen ze záložního zdroje, nýbrž je napájena pouze elektronika modulu GD-04. Při úplném vybití záložního modulu, odešle GD-04 informativní sms - POWER FAIL a modul se vypne. Po dalším připojení na síť se akumulátor znovu dobije přibližně za 72 hodin. Na záložní akumulátor lze GD-04 napájet cca. 12 až 24 hodin v závislosti na spotřebě GSM modulu a spínacích relé. Na obrázku č. 11 jsou znázorněny vstupní svorky GD-04 a schematický spínač, který vyvolá upozornění.[12]

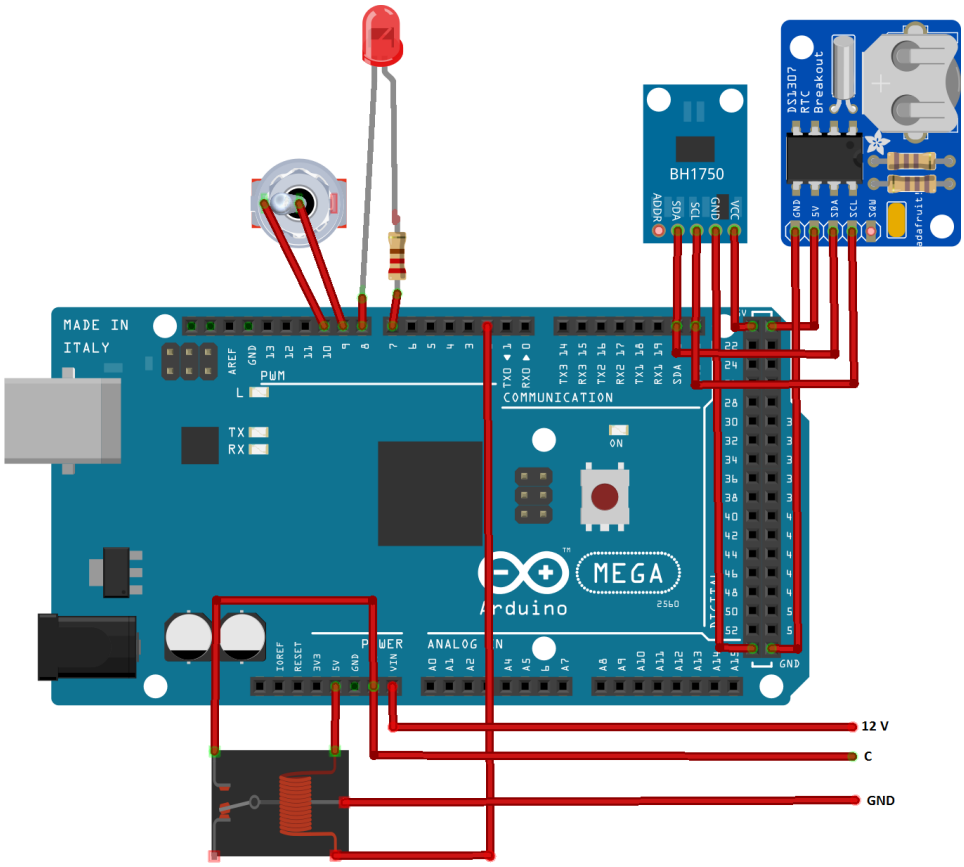


Obr. 13 Znázornění realizace upozornění [12]

3.8 Hardwarová realizace detektoru

Detektor byl vyvíjen za pomoci drátových propojek a nepájivého pole. V prvotních fázích bylo pracováno se světelným senzorem a RTC modulem, současně s tím byl vyvíjen kód pro tyto dvě komponenty. Posléze bylo přidáno relé a stavová dioda s přepínacím tlačítkem. Finální podoba detektoru byla vyřešena za pomoci univerzálního plošného spoje. Nejprve byl plošný spoj zkrácen na rozměry 50x55 *mm*. Všechny tři hlavní komponenty byly za pomocí pinových kolíků připájeny k univerzálnímu plošnému spoji. Následně byly na plošný spoj připájeny pinové kolíky tak, aby bylo možno plošný spoj s připájenými moduly vhodně zasunout do Arduina,

vznikl tak v podstatě Arduino zásuvný modul, který lze z Arduina vytáhnout a opětovně zasunout. Poté byly vyvrtány otvory do zadní části pouzdra pro distanční šroubky, na které se přišroubovalo Arduino. Poloha otvorů byla situována tak, aby Arduino bylo přibližně na středu zadní části pouzdra. Do výkonového relé se následně na porty COM a NO našroubovaly vodiče zakončené dutinkami, které byly vhodně slisovány lisovacími kleštěmi. Tyto dva vodiče propojují svorku GND a vstupní svorku v hlásiči Jablotron GD-04. Dále byly vyvedeny vodiče zakončené taktéž pinovým kolíkem pro externí napájení Arduina. Tyto vodiče byly taktéž zakončeny kolíkovými svorkami a následně zalisovány. Všechny čtyři vodiče byly vyvedeny průchodkou v zadní části pouzdra. Do vrchní části pouzdra byl vyříznut a následně dopilován otvor pro USB konektor, aby se pouzdro při připojování k počítači nemuselo rozmontovávat. Následně byl vyvrtán otvor pro stavovou diodu a přepínací tlačítko, které je k tělu pouzdra přichyceno maticí. Stavová dioda byla do série zapájena s rezistorem o odporu 200 Ω pro omezení proudu diodou, který se skryl do smršťovací bužírky. Vodiče stavové diody i přepínacího tlačítka byly zakončeny opět pinovým kolíkem. Schematické znázornění propojení pinů v detektoru je na obr. č. 14. Díky čirému pouzdru je zajištěn bezproblémový chod senzoru, který nepřetržitě snímá intenzitu osvětlení. Detektor je uzpůsoben jako samostatná jednotka, kterou lze napájet i z jiného zdroje než-li ze zařízení Jablotron GD-04, například ze standardních AAA tužkových baterií. Dvoustavový výstup, který je realizován relé s maximálním dovoleným proudem 10 A, nemusí nutně propojovat pouze dvě svorky v externím zařízení. Do úvahy připadá možnost použít detektor jako spínač jiného zařízení například nouzového osvětlení, které se má aktivovat při nedodržení vhodně nastaveného světelného režimu. Níže uvedený diagram součástek byl vytvořen v softwaru Fritzing. Fritzing je open-source aplikace umožňující kreslení diagramů a schémat pro ARDUINO[15]. Propojení pinů v diagramu přesně odpovídá zapojení v reálném provedení.



Obr. 14 Výstup ze softwaru Fritzing - diagram zapojení

3.9 Rozpočet realizace

Předností tohoto projektu je jeho cena, v porovnání s profesionálním řešením na zakázku od specializované firmy je to pouze 5% ceny. U profesionálního řešení lze čekat vyšší spolehlivost a přívětivější uživatelské rozhranní. Celkový rozpočet zařízení je v tab. č. 5.

Součástka	Typ	Výrobce	Cena
Arduino	AtMega2560	Rohm semiconductor	424 Kč
Senzor osvětlení	BH1750FVI	Rohm semiconductor	122 Kč
RTC modul	DS1307	Maxim integrated	61 Kč
Relé	SRD-05VDC-SL-C	Ningbo *Songle* Relay	73 Kč
Univerzální plošný spoj	RM 2,54mm / 1mm	GM electronics	72 Kč
Přepínací tlačítko	P-KNX1	ningbo jietong electronic	16 Kč
Dvoubarevná dioda	L57EGW	Kingbright	6 Kč
Rezistor	RM 200R 0207 0,6W	GM electronics	2 Kč
Pouzdro	052C ABS	Serpac	125 Kč
Celkem			901 Kč

Tab.5 Cena komponentů

3.10 Popis softwaru

Vytvořený kód pro řízení detektoru světelného režimu byl psán ve vývojovém prostředí Arduino software pod operačním systémem Windows. Vývojové prostředí je napsáno v programovacím jazyku Java. Programovací jazyk pro Arduina je odvozen od programovacího jazyku Wiring a funguje na bázi jazyku C++. Standardní program pro Arduino se skládá ze čtyř částí. Za první lze považovat tzv. hlavičku programu, do které patří deklarace globálních proměnných a konstant, dále pak načtení externích knihoven. V případě kódu pro řízení detektoru byly v hlavičce definovány externí open-source knihovny pro moduly DS1307 a BH1750, dále pak knihovna *„Wire.h“*, kvůli možnosti volání příkazů pro komunikaci prostřednictvím sběrnice I2C a konečně knihovna *„EEPROM.h“* pro práci s vnitřní EEPROM pamětí, do které se ukládá nastavení světelného režimu. Dále jsou v hlavičce deklarovány proměnné pro nastavování data a času a také logické proměnné typu boolean. Každá proměnná byla deklarována s ohledem na její předpokládaný rozsah tak, aby zabírala co možná nejméně místa v paměti. Proměnné jsou ošetřeny tak, aby uložená hodnota v nich nepřekročila deklarovanou velikost. Pokud by se tak stalo, program by vykázal chybu a následoval by fatální konec programu. Zadávání vstupů v uživatelském menu podléhá kontrolnímu mechanismu, při případném špatném zadání hodnoty, která není v intervalu povolených hodnot, vyzve uživatele k opětovnému zadání hodnoty do doby než-li je zadání správné.

To se týká všech hodnot, které figurují v programu jako vstupní - ať už se jedná o nastavení data a času či nastavení času režimu.

Jako druhá následuje definice vlastních funkcí, které budou dále využity v programu. Sem patří funkce, která rozsvítí diodu zeleně a červeně, dále test polohy přepínacího tlačítka, vypsání času přes sériovou linku, vymazání bufferu sériové linky a ukládání do EEPROM paměti. Díky takto definovaným funkcím bylo ušetřeno místo v paměti a zpřehledněn kód programu.

Následuje funkce *„setup“*, která je vykonána pouze jednou po spuštění programu. Ve funkci *„setup“* lze definovat vlastnosti pinů, nastavit proměnné, případně také inicializovat knihovny. Funkce *„setup“* musí být obsažena v každém programu pro Arduino i kdyby měla být prázdná a neobsahovala by žádné příkazy. Nastavení pinů probíhá za pomoci příkazu *„pinMode“* jako vstupní parametr je po řadě číslo pinu a mód pinu - vstup/výstup. V programu pro detektor byl nastaven pin číslo 2 jako výstupní pro ovládání dvoustavového výstupu - relé. Piny číslo 7 a 8 pro ovládání diody byly nastaveny také jako výstupní. Přepínáním napěťových úrovní mezi těmito dvěma piny se mění polarita proudu dvěma antiparalelně zapojenými diodami uvnitř pouzdra, které jsou v sérii s rezistorem o odporu 200 Ω. Pin číslo 9 je nastaven jako *„INPUT_PULLUP“* a pin číslo 10 jako výstupní. Když je tlačítko v poloze pro nečinnost detektoru, jsou piny 9 a 10 propojeny a na vstupu pinu 9 je nízká napěťová úroveň. V opačném případě, kdy je tlačítko v druhé poloze, je díky internímu pull-up rezistoru na pinu číslo 9 vysoká napěťová úroveň. Čtením vstupní napěťové úrovně na pinu 9 je pak v kódu programu prostřednictvím funkce *„TestTlacitko“* zjišťována poloha tlačítka. Ve funkci *„setup“* jsou pak dále za pomocí příkazu *„digitalWrite“* nastaveny počáteční napěťové úrovně pro piny, které obsluhují relé, diodu a přepínací tlačítko. Je zde také inicializován senzor BH1750 a načteny hodnoty z EEPROM paměti - ochranný limit a čas světelného režimu.

Kód zakončuje funkce *„loop“*, která se vykonává ve smyčce. Do této funkce je vepsán primární kód programu, který vykonává všechny podstatné funkce detektoru. Spolu s funkcí *„setup“* je to další funkce, která musí být obsažena v programu pro Arduino, které je psáno ve vývojovém prostředí IDE. Ve funkci *„loop“* jsou nejprve deklarovány lokální proměnné, které jsou použity ihned při deklaraci pro načtení aktuální intenzity osvětlení a aktuálního času za pomocí příkazů z knihoven. Dále jsou

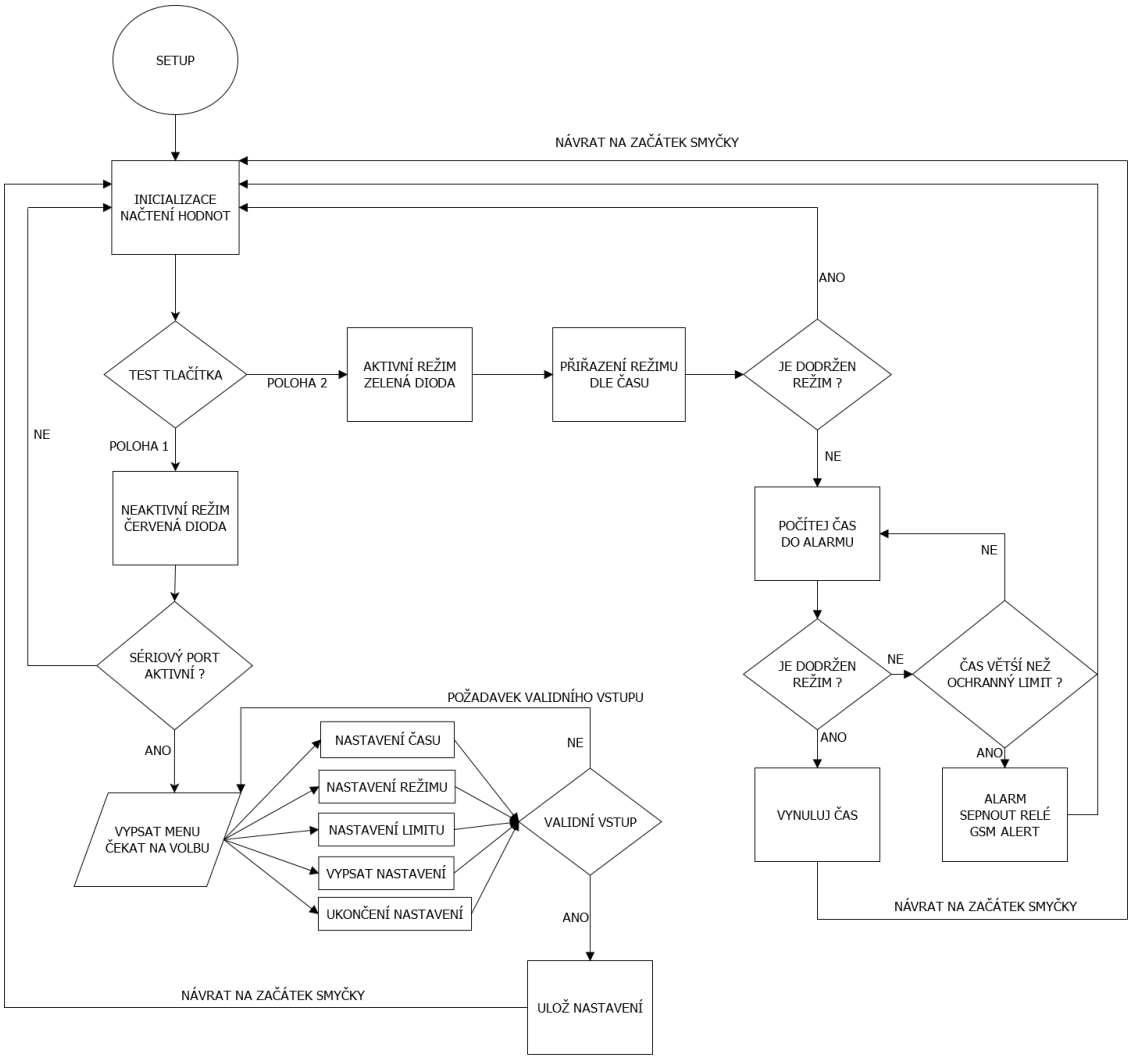
zde deklarované pomocné proměnné pro práci s časem, každé jednotce času náleží jedna proměnná - hodiny,minuty a sekundy. Činnost kódu začíná testem polohy přepínacího tlačítka, jak lze vidět z vývojového diagramu na obr. č.15. Zde se kód v zásadě rozděluje na dvě větve, a sice aktivní a neaktivní režim. Pokud je tlačítko v poloze 1, detektor je uveden do neaktivního stavu, svítí červená dioda, je-li zároveň v bufferu sériového portu dostupný znak pro přečtení, aktivuje se spuštění uživatelského rozhranní. Do uživatelského rozhranní se tedy detektor dostane tím, že ho uživatel „vzbudí“ odesláním libovolného znaku přes sériovou linku. Sériová komunikace probíhá prostřednictvím UART portu a konektoru USB a znaková rychlost této komunikace je nastavena na 9600 Bd. Uživatelské rozhranní začíná vypsáním menu, kde jsou vypsány volby, které uživatel může zvolit a dostat se tak na jednotlivé funkce. Menu je řešeno příkazem switch-case. Celé uživatelské rozhranní je ošetřeno tak, aby nebylo možné uložit do proměnných chybnou hodnotu, která nevyhovuje deklaraci proměnné nebo logickému limitu proměnné. Nelze například zadat čas 25 hodin a 99 minut. Tyto ošetření jsou v zásadě realizovány do-while cykly. Uživatel je v rozhranní navigován textovými příkazy a nelze se v něm dostat do stavu, kde by jakýkoli vstup z klávesnice způsobil fatální konec programu. Vyčištění bufferu sériové linky, které probíhá při ukončení každé dílčí části menu prostřednictvím volání funkce *„CistiKlavesnici“*, zabezpečuje hladké procházení menu bez toho aniž by znak z předchozího oddílu menu fungoval jako vstup v jiném oddílu.

Funkce *„loop“* dále pokračuje přiřazením logické hodnoty pro režim světlo dle přednastaveného času. Tento krok je vykonán v rámci druhé větve programu, kdy je režim v aktivním režimu a zabezpečuje světelný režim. Následuje vyhodnocení světelného režimu. Hranice pro světlo a tmu byla stanovena na 2 lx. Jestliže není režim dodržen, začne se počítat čas od této události za pomocí příkazu millis. Tento příkaz počítá počet milisekund od vzniklé události. Je-li tento čas větší než-li uživatelem nastavený limit, sepne se relé a tím vzniká upozornění na nedodržení režimu. Je-li však během tohoto měření času od vzniklé události, kdy režim není ve shodě s přednastaveným, režim opět ve shodě, počítání je ukončeno a k upozornění nedojde. Je to tedy určitá forma hystereze, nebo-li paměťového efektu. K vyvolání upozornění je tedy zapotřebí kontinuální nedodržení světelného režimu, které je delší, než je přednastavený limit upozornění. Zde je nutné, aby se uživatel zamyslel nad vhodným nastavením. Lze si představit situaci, kdy umělé světlo v cílovém prostoru bliká s

určitou periodou, která bude menší než nastavený limit a k upozornění prakticky nedojde. Tuto funkci hystereze je samozřejmě možné jednoduše deaktivovat nastavením limitu na 0 sekund. Když detektor aktivuje relé a tím vyšle signál do zařízení GD-04, jediný způsob jak deaktivovat upozornění je fyzicky přepnout tlačítko na detektoru, tím se rozepne relé. Detektor je možné ihned přepnout zpět do aktivního režimu vrácením tlačítka zpět do polohy 2 pro aktivní režim.

3.11 Vývojový diagram

Logická posloupnost dílčích kroků vytvořeného softwaru pro detektor je znázorněna na obr.č.15. Tento vývojový diagram, lze dle literatury zařadit to kategorie tzv. *flow* diagram.[16] Tento diagram určuje logiku a chování celého softwaru jako celku. Většina bloků představuje soubor příkazů a podmínek, některé však charakterizují jen jeden příkaz či jednu funkci. Podrobnosti jsou uvedeny v příloze 1, kde je zdrojový kód softwaru. Blok „setup“ v sobě zahrnuje nezbytné kroky k nastavení mikrokontroléru pro správný chod softwaru. Blok inicializace/načtení hodnot lze chápat jako deklaraci jak globálních tak lokálních proměnných a načítání hodnot z externích modulů. V kosočtvercích jsou hlavní logické rozhodování. Kosodélník představuje vstup z klávesnice či jiného zařízení přes sériovou linku. Konečné instrukce jsou vždy zakončeny návratem na začátek - inicializaci, to charakterizuje chod softwaru ve smyčce.



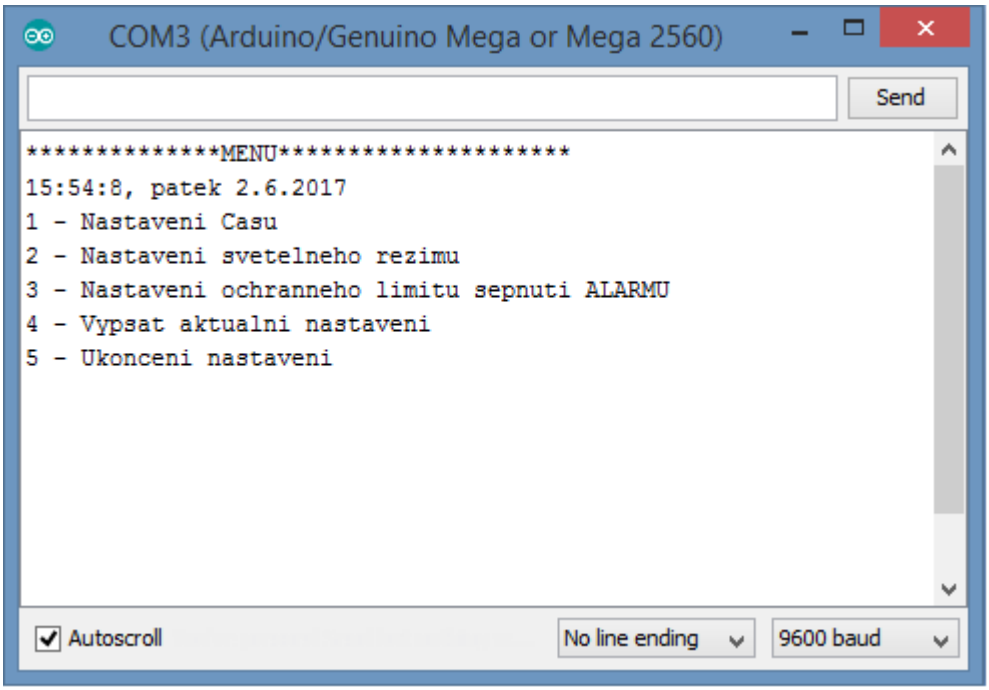
Obr. 15 Vývojový diagram obslužného programu

3.12 Uživatelské rozhraní

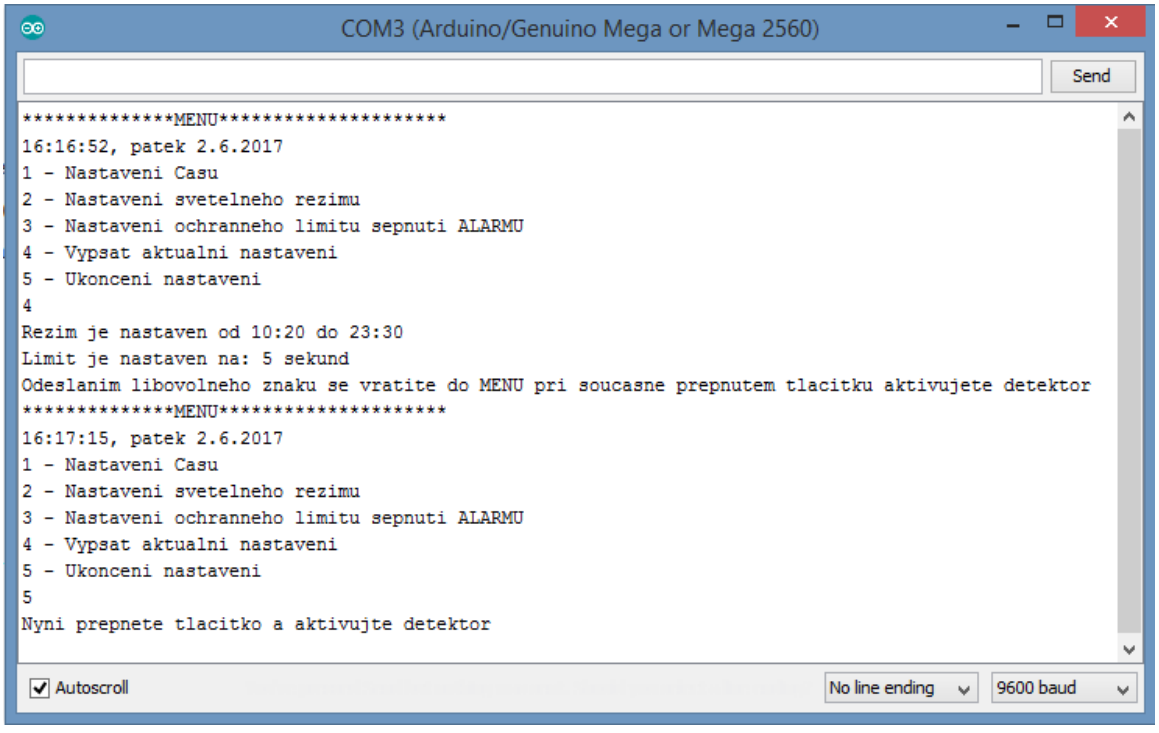
Komunikace s detektorem probíhá prostřednictvím USB přes komunikační rozhranní UART. V případě Arduina, při použití příkazu „Serial.print()“, jsou bajty přenášeny prostřednictvím UART na čipu do FTDI FT232 USB sériového konvertoru, odkud putují dále do PC. Zkratka FTDI je název Britské společnosti Future Technology Devices International, která se specializuje na výrobu převodníků mezi standardem RS-232 a USB. Při vysílání dat z počítače do Arduina jsou data ukládána do „bufferu“, odkud jsou čtena. Přijaté a odeslané bity jsou indikovány prostřednictvím led diod s označením TX a RX na desce Arduino. V PC, ve kterém bude docházet k nastavování

Arduina je nutné nainstalovat virtuální sériový port, který dokáže komunikovat přes sériovou linku rychlostí 9600 Bd.

Po připojení prostřednictvím USB je k aktivaci uživatelského rozhranní nezbytné učinit dva kroky. První krok je přepnutí tlačítka do polohy 1, režim indikuje červeně svítící dioda. Dále pak odeslání libovolného znaku přes sériovou linku. Po odeslání znaku se vypíše úvodní textové menu, které je zobrazeno na obr. č. 16. Menu obsahuje pět různých voleb. V menu je možné nastavit tři položky - datum a čas, čas světelného režimu a nastavit ochranný limit sepnutí alarmu. Volba čtyři vypíše aktuální nastavení a volbou číslo pět lze nastavování ukončit. Do aktivního režimu se lze dostat buď přepnutím tlačítka po ukončení zvolené položky nebo položkou číslo pět. Poté po přepnutí tlačítka do polohy 2 je detektor aktivován. Na obr. č. 17 lze vidět posloupnost vypisovaných údajů při práci s detektorem přes sériovou linku.



Obr. 16 Menu obslužného programu



Obr. 17 Ukázka nastavování detektoru

Závěr

V prvé části textu je popsán princip měření intenzity osvětlení a součástky, které se v elektronice k měření intenzity osvětlení používají nejčastěji. Dále je popsána platforma Arduino.

Návrh detektoru světelného režimu je popsán v kapitole 3.1. K realizaci byl použit mikrokontrolér Arduino ATmega2560 spolu s moduly BH1750FVI, DS1307 a relé SRD-05V-DC. Detektor umožňuje rozlišovat dvě úrovně vnějšího osvětlení. Hranice světla a tmy byla stanovena na 2 lx. Když je detektor v aktivním režimu, monitorování vnějšího osvětlení probíhá kontinuálně. Výstup detektoru je dvoustavový, realizován prostřednictvím relé a reaguje na nedodržení přednastaveného světelného režimu. Mezi neaktivním a aktivním režimem lze přepínat za pomoci přepínacího tlačítka.

Program, který řídí detektor, byl napsán v programovacím jazyku Wiring, který je prakticky totožný s jazykem C++. Vstupní hodnoty lze nastavit pomocí USB přes virtuální sériový port. Komunikace probíhá přes hardwarový UART v Arduinu. Uživatelské rozhranní bylo vytvořeno s ohledem na jednoduchou ovladatelnost a bylo ošetřeno tak, aby špatný uživatelský vstup nezpůsobil chybnou funkci programu. Obslužný program umožňuje nastavení času a data, dále pak nastavení světelného režimu a ochranného limitu spuštění upozornění. Ochranný limit lze označit za jistý způsob hystereze, podrobný popis lze najít v kapitole 3.10.

Hardware i software detektoru byl vytvořen s ohledem na jednoduchost a uživatelskou přívětivost. Doprovodný modul Jablotron GD-04 lze zaměnit za jiné zařízení. Do budoucna je možné řídící program pozměnit a vylepšit funkce detektoru.

Seznam literatury a informačních zdrojů

[1] (Online převzatý obrázek)

http://physics.mff.cuni.cz/kfpp/skripta/elektronika/kap2/2_2_5.html

[2] (Online převzatý obrázek)

http://moryst.sweb.cz/elt2/stranky1/elt007.htm

[3] © 2017 Forbes Media [online] [cit. 30.5.2017] Dostupné z https://www.forbes.com/

[4] Globenewswire*:* [online]. [cit. 30.5.2017]. Dostupné z: https://globenewswire.com/

[5] (Online převzatá tabulka)

https://www.arduino.cc/en/Products/Compare

[6] Selecký, Matúš. *Arduino uživatelská příručka*. 1. vyd. Brno Computer Press, 2016 s.44 ISBN 978-80-251-4840-2

[7] (Online převzatý obrázek)

http://www.santy.cz/arduino-c2/arduino-mega2560-r3-i81/

[8] Datasheet BH1750

[9] (Online převzatý obrázek)

http://www.esp8266learning.com/bh1750fvi-ambient-light-sensor-example.php

[10] (Online převzatý obrázek)

http://hub360.com.ng/shop-2/ds1307-rtc-module/

[11] (Online převzatý obrázek)

http://www.electroschematics.com/8921/digital-clock-with-arduino-and-ds1307/

[12] Datasheet Jablotron GD-04

[13] (Online převzatý obrázek)

http://www.instructables.com/id/Home-Automation-How-to-Add-Relays-to-Arduino/

[14] (Online převzatý obrázek)

https://www.gme.cz/krabicka-plastova-052c-abs

[15] Selecký, Matúš. *Arduino uživatelská příručka*. 1. vyd. Brno Computer Press, 2016 s.272 ISBN 978-80-251-4840-2

[16] Selecký, Matúš. *Arduino uživatelská příručka*. 1. vyd. Brno Computer Press, 2016 s.43 ISBN 978-80-251-4840-2

Přílohy

Příloha A - Zdrojový kód obslužného programu

```
#include <BH1750.h>
#include <EEPROM.h>
#include <RTClib.h>
#include <Wire.h>
#include <BH1750.h>
RTC_DS1307 DS1307;
BH1750 lightMeter;
char seznamDni[7][8] = {"nedele", "pondeli", "utery", "streda", "ctvrtek", "patek", "sobota"};
int YYYY,LIMIT;
char volba,rele=2;
boolean svetlo=false,ALARMING=false;
unsigned long CAS,CASALERT;
unsigned char FROMHH,FROMMM,TOHH,TOMM,HH,MM,Sec,MONTH,DAY,vypinac;
void cervena () {
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  }

void zelena () {
  digitalWrite(8, HIGH);
  digitalWrite(7, LOW);
  }


void CistiKlavesnici(){
  while(Serial.available() > 0) {
  char t = Serial.read();
  }
}

void VypisCas (){
      DateTime datumCas = DS1307.now();
      Serial.print(datumCas.hour());
      Serial.print(':');
      Serial.print(datumCas.minute());
      Serial.print(':');
      Serial.print(datumCas.second());
      Serial.print(", ");
      Serial.print(seznamDni[datumCas.dayOfTheWeek()]);
      Serial.print(" ");
      Serial.print(datumCas.day());
      Serial.print('.');
      Serial.print(datumCas.month());
      Serial.print('.');
      Serial.println(datumCas.year());
      }

void TestTlacitko (){
vypinac=digitalRead(9);
if (vypinac==1)
  {//zelena
    zelena ();
  }
```

```
  else if (vypinac==0)  //cervena
  {
    cervena ();
    digitalWrite(2, HIGH);
    ALARMING=false;
  }
}


void UlozNastaveni (unsigned char FROMHH, unsigned char FROMMM, unsigned char TOHH,
unsigned char TOMM, int LIMIT ) {

  EEPROM.write(0, FROMHH);
  EEPROM.write(1, FROMMM);
  EEPROM.write(2, TOHH);
  EEPROM.write(3, TOMM);
  int a = LIMIT/256;
  int b = LIMIT % 256;
  EEPROM.write(4,a);
  EEPROM.write(5,b);
  }



void setup()
{
  // nastaveni pinu a nacteni hodnot z pameti
  pinMode(2, OUTPUT);
  digitalWrite(2, HIGH);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, INPUT_PULLUP);
  pinMode(10, OUTPUT);
  digitalWrite(10, LOW);
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  lightMeter.begin();
  FROMHH=EEPROM.read(0);
  FROMMM=EEPROM.read(1);
  TOHH=EEPROM.read(2);
  TOMM=EEPROM.read(3);
  int a=EEPROM.read(4);
  int b=EEPROM.read(5);
  LIMIT=a*256+b;

}

void loop()
{
  // opakujici se smycka
  uint16_t lux = lightMeter.readLightLevel();
  TestTlacitko ();
  DateTime datumCas = DS1307.now();
  HH=datumCas.hour();
  MM=datumCas.minute();
  Sec=datumCas.second();
  Serial.begin(9600);

        if ((Serial.available()>0)&&((vypinac==0)))
```

```
{
Serial.println("***************MENU**********************");
VypisCas ();
Serial.println("1 - Nastaveni Casu");
Serial.println("2 - Nastaveni svetelneho rezimu");
Serial.println("3 - Nastaveni ochranneho limitu sepnuti ALARMU");
Serial.println("4 - Vypsat aktualni nastaveni");
Serial.println("5 - Ukonceni nastaveni");
CistiKlavesnici();
do
  {
    while(!Serial.available()) {}
    volba=Serial.read();
    if ((volba<49)||(volba>53))
    {
      Serial.println("Opakujte zadani spravne");
    }
  } while ((volba<49)||(volba>53));
  Serial.println(volba);
  CistiKlavesnici();
  switch (volba)
    {
  case 49: //volba 1
      Serial.println("Postupne zadejte YY:MM:DD / HH:MM:SS");

      do                //Nastaveni roku
      {
       Serial.println("Nastavte prosim rok: YYYY");
       while(!Serial.available() ){}
       YYYY=Serial.parseInt();
       if ((YYYY<2000) || (YYYY> 2099))
        Serial.println("Opakujte prosim zadani spravne");
      } while ((YYYY<2000) || (YYYY> 2099));
      CistiKlavesnici();
      do                //Nastaveni mesice
      {
       Serial.println("Nastavte prosim mesic: MM");
       while(!Serial.available() ){}
       MONTH=Serial.parseInt();
       if ((MONTH<1) || (MONTH> 12))
        Serial.println("Opakujte prosim zadani spravne");
      } while ((MONTH<1) || (MONTH> 12));
      CistiKlavesnici();
      do                //Nastaveni dne
      {
       Serial.println("Nastavte prosim den: DD");
       while(!Serial.available() ){}
       DAY=Serial.parseInt();
       if ((DAY<1) || (DAY> 31))
        Serial.println("Opakujte prosim zadani spravne");
      } while ((DAY<1) || (DAY> 31));
      CistiKlavesnici();
      do                //Nastaveni HH
      {
       Serial.println("Nastavte prosim hodiny: HH");
       while(!Serial.available() ){}
       HH=Serial.parseInt();
       if ((HH<0) || (HH> 23))
        Serial.println("Opakujte prosim zadani spravne");
      } while ((HH<0) || (HH> 23));
```

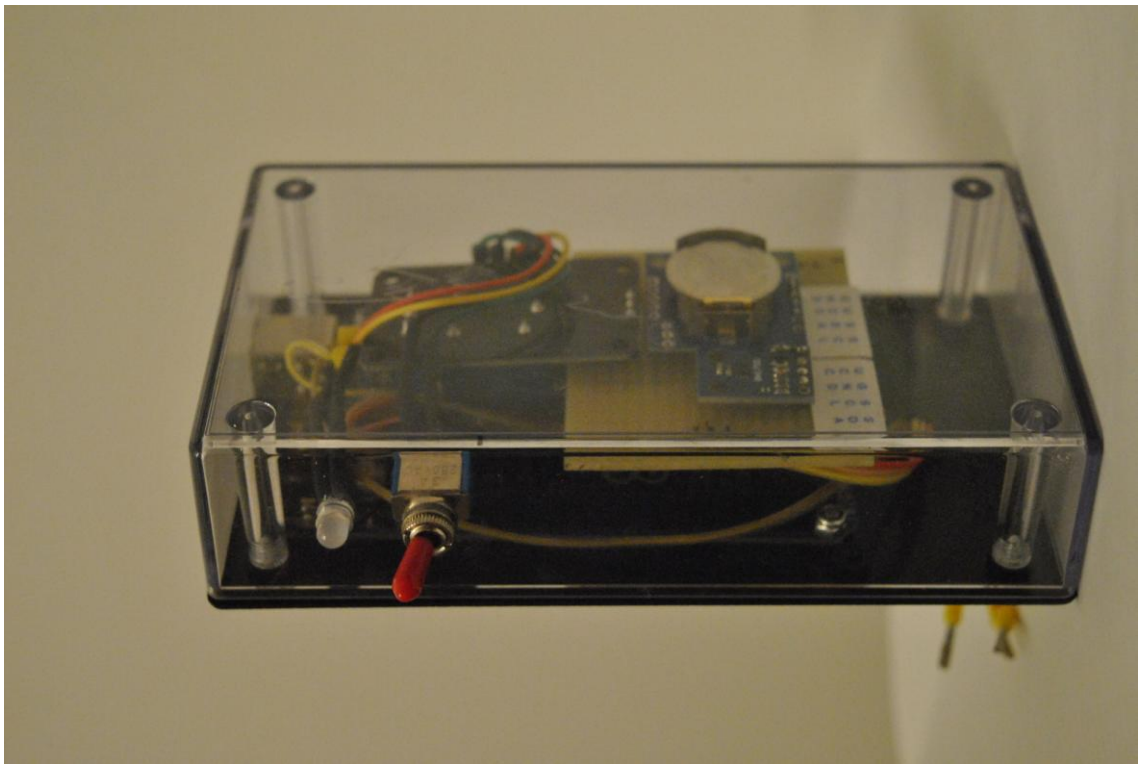
```
          CistiKlavesnici();
          do              //Nastaveni MM
          {

           Serial.println("Nastavte prosim hodiny: MM");
           while(!Serial.available() ){}
           MM=Serial.parseInt();
           if ((MM<0) || (MM> 59))
            Serial.println("Opakujte prosim zadani spravne");
          } while ((MM<0) || (MM> 59));
          CistiKlavesnici();
          do              //Nastaveni SS
          {

           Serial.println("Nastavte prosim hodiny: SS");
           while(!Serial.available() ){}
           Sec=Serial.parseInt();
           if ((Sec<0) || (Sec> 59))
            Serial.println("Opakujte prosim zadani spravne");
          } while ((Sec<0) || (Sec> 59));

          CistiKlavesnici();
          DS1307.adjust(DateTime(YYYY, MONTH, DAY, HH, MM, Sec));
          Serial.println("Aktualne nastaveny cas:");
          VypisCas ();
          UlozNastaveni(FROMHH,FROMMM,TOHH,TOMM,LIMIT);
          Serial.println("Odeslanim libovolneho znaku se vratite do MENU pri soucasne prepnutem
tlacitku aktivujete detektor");
          while(!Serial.available() ){}
          TestTlacitko ();
          break;



      case 50: // volba 2 nastaveni svetelneho rezimu

          Serial.println("2 - Nastaveni svetelneho rezimu");
          do              //Nastaveni Svetla OD HH
          {

           Serial.println("Zadejte pocatek rezimu svetlo ve formatu: HH:MM");
           Serial.println("Nastavte prosim hodiny od: HH");
           while(!Serial.available() ){}
           FROMHH=Serial.parseInt();
           if ((FROMHH<0) || (FROMHH> 23))
            Serial.println("Opakujte prosim zadani spravne");
          } while ((FROMHH<0) || (FROMHH> 23));
          CistiKlavesnici();
          do              //Nastaveni Svetla OD MM
          {


           Serial.println("Nastavte prosim minuty od: MM");
           while(!Serial.available() ){}
           FROMMM=Serial.parseInt();
           if ((FROMMM<0) || (FROMMM> 59))
            Serial.println("Opakujte prosim zadani spravne");
          } while ((FROMMM<0) || (FROMMM> 59));
          CistiKlavesnici();
```

```
            Serial.println("Zadejte konec rezimu svetlo ve formatu: HH:MM");

            do              //Nastaveni Svetla DO HH
            {


             Serial.println("Nastavte prosim hodiny do: HH");
             while(!Serial.available() ){}
             TOHH=Serial.parseInt();
             if ((TOHH<0) || (TOHH> 23))
              Serial.println("Opakujte prosim zadani spravne");
            } while ((TOHH<0) || (TOHH> 23));
            CistiKlavesnici();
            do              //Nastaveni Svetla DO MM
            {


             Serial.println("Nastavte prosim hodiny do: MM");
             while(!Serial.available() ){}
             TOMM=Serial.parseInt();
             if ((TOMM<0) || (TOMM> 59))
              Serial.println("Opakujte prosim zadani spravne");
            } while ((TOMM<0) || (TOMM> 59));
            Serial.print("Rezim je nastaven od ");
            Serial.print(FROMHH);
            Serial.print(":");
            Serial.print(FROMMM);
            Serial.print(" do ");
            Serial.print(TOHH);
            Serial.print(":");
            Serial.println(TOMM);
            Serial.println("Odeslanim libovolneho znaku se vratite do MENU pri soucasne prepnutem
tlacitku aktivujete detektor");
            TestTlacitko();
            UlozNastaveni(FROMHH,FROMMM,TOHH,TOMM,LIMIT);
            while(!Serial.available() ){}
            break;


          case 51: //volba 3 nastaveni limitu

             Serial.println("3 - Nastaveni ochranneho limitu sepnuti ALARMU, v tomto intervalu se
neaktivuje spinac pri nedodrzeni rezimu");
             do              //Nastaveni limitu v sec.
            {

             Serial.println("Zadejte limit ve formátu SS, max 300s");
             while(!Serial.available() ){}
             LIMIT=Serial.parseInt();
             if ((LIMIT<0) || (LIMIT> 300))
              Serial.println("Opakujte prosim zadani spravne");
            } while ((LIMIT<0) || (LIMIT> 300));
            CistiKlavesnici();
            Serial.print("Limit byl nastaven na: ");
            Serial.print(LIMIT);
            Serial.println(" sekund");
            UlozNastaveni(FROMHH,FROMMM,TOHH,TOMM,LIMIT);
            Serial.println("Odeslanim libovolneho znaku se vratite do MENU pri soucasne prepnutem
tlacitku aktivujete detektor");
            while(!Serial.available() ){};
```

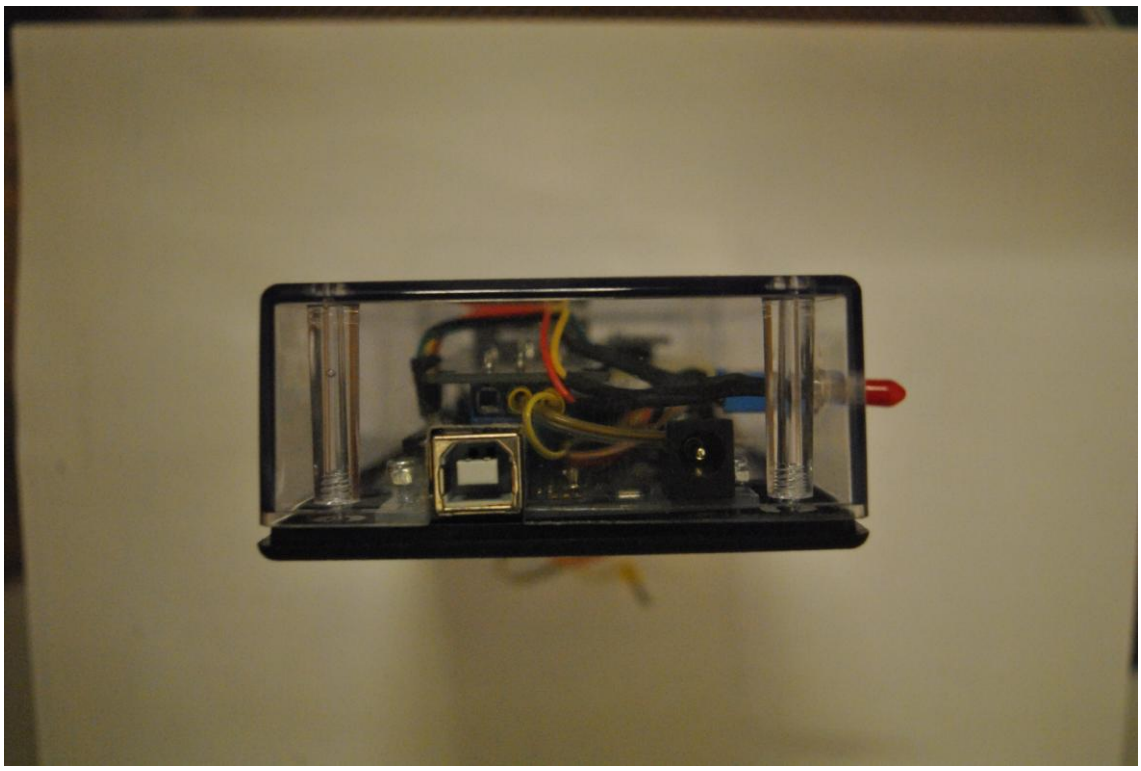
```
            break;

          case 52: //volba 4 - vypsani aktualniho nastaveni
           {

           CistiKlavesnici();
           Serial.print("Rezim je nastaven od ");
           Serial.print(FROMHH);
           Serial.print(":");
           Serial.print(FROMMM);
           Serial.print(" do ");
           Serial.print(TOHH);
           Serial.print(":");
           Serial.println(TOMM);
           Serial.print("Limit je nastaven na: ");
           Serial.print(LIMIT);
           Serial.println(" sekund");
           Serial.println("Odeslanim libovolneho znaku se vratite do MENU pri soucasne prepnutem tlacitku
aktivujete detektor");
           while(!Serial.available() ){}

           }
           break;
          case 53:
          Serial.println("Nyni prepnete tlacitko a aktivujte detektor");
          do
           {
             TestTlacitko ();
           } while (vypinac!=1);
           break;
            }


       }  // KONEC MENU



  if (vypinac==1)
  {
    if
((((FROMHH==HH)&&(MM>=FROMMM))||(FROMHH<HH))&&(((TOHH==HH)&&(MM<TOMM)
)||(HH<TOHH)))

    {
      svetlo=true;
    }
    else svetlo=false;

    if ((svetlo==true)&&((lux)<2)&&(ALARMING==false))
      {
           CASALERT = millis();
           ALARMING=true;
      }

    else if ((svetlo==false)&&((lux)>2)&&(ALARMING==false))
      {
          CASALERT = millis();
          ALARMING=true;
      }
```

```
    CAS=millis();


 if ((ALARMING==true)&&((CAS-CASALERT)<(LIMIT*1000))&&(svetlo==true)&&(lux>2))
//zhasnuti v intervalu v case mensi jak LIMIT -> nevyhodnoti a vynulovani casovani
    {
     ALARMING=false;
    }

  if ((ALARMING==true)&&((CAS-CASALERT)<(LIMIT*1000))&&(svetlo==false)&&(lux<2))
//probliknuti svetla v case mensi jak limit nevyhodnoti -> nevyhodnoti a vynulovani casovani
    {
     ALARMING=false;
    }

  if ((ALARMING==true)&&((CAS-CASALERT)>(LIMIT*1000))&&(svetlo==true)&&((lux)<2)) //
KONECNY ALARM pro svetlo
   {
    digitalWrite(rele, LOW);
    ALARMING=false;
   }


  if ((ALARMING==true)&&((CAS-CASALERT)>(LIMIT*1000))&&(svetlo==false)&&((lux)>2)) //
KONECNY ALARM pro tmu
    {
    digitalWrite(rele, LOW);
    ALARMING=false;
    }
  }

  TestTlacitko ();

}
```

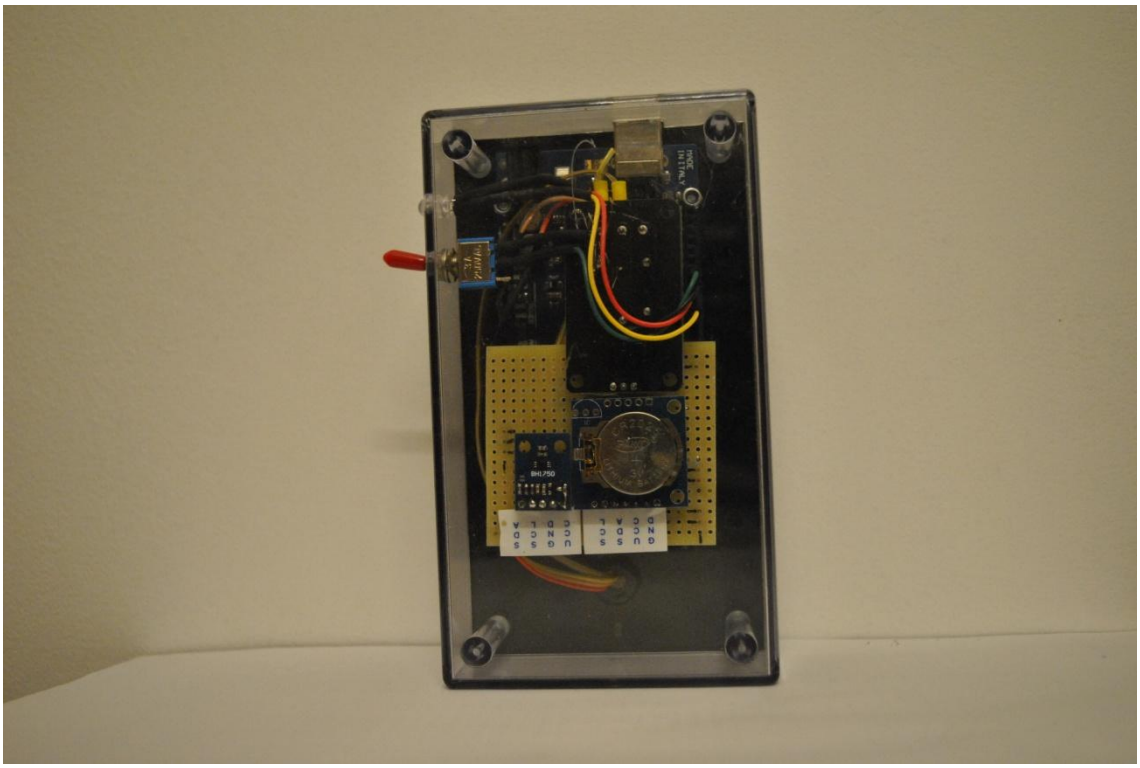
Příloha B - Fotodokumentace detektoru



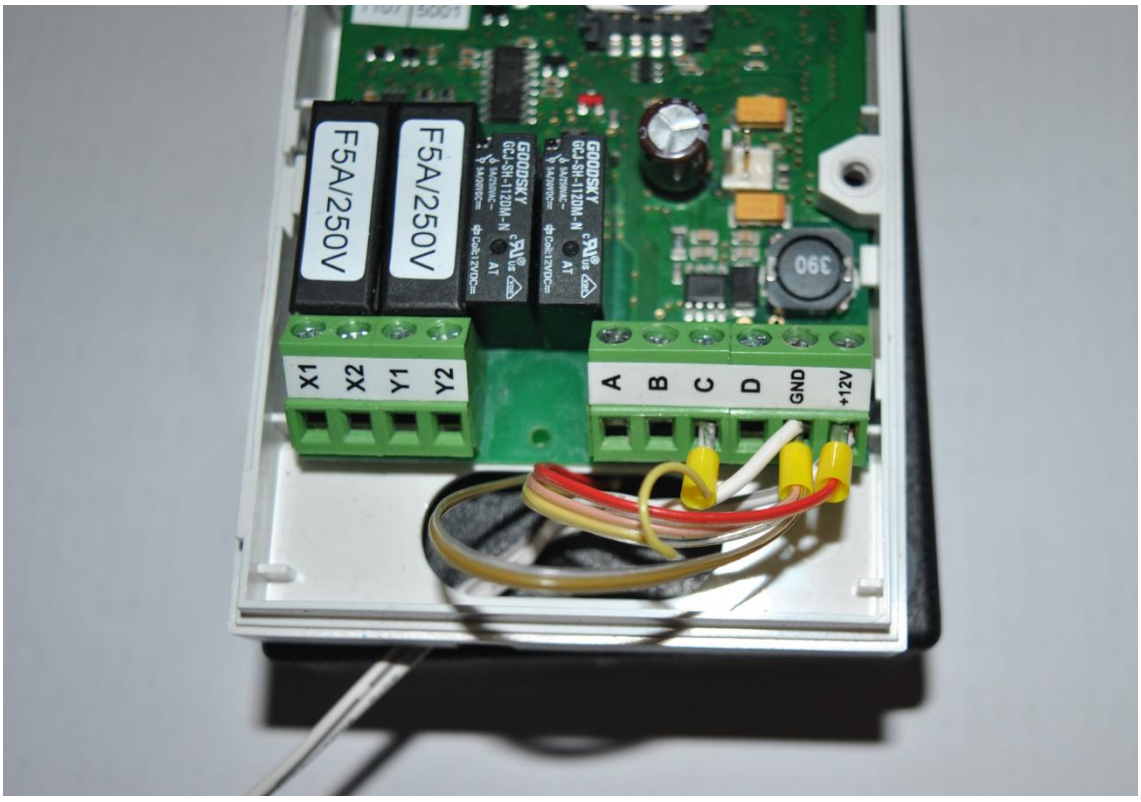
Obr. 18 Pohled na přepínací tlačítko a diodu



Obr. 19 Pohled shora



Obr. 20 Pohled na přední stranu



Obr. 21 Zapojení konektorů v externím zařízení Jablotron GD-04