Vysoká škola báňská - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Přípravek pro elektronový mikroskop

A Composition for Electron Microscope

2016/2017 Jan Dvořák

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Zadání diplomové práce

Student: **Bc. Jan Dvořák**

Studijní program: N2649 Elektrotechnika

Studijní obor: 2612T041 Řídicí a informační systémy

Téma: Přípravek pro elektronový mikroskop
A Composition for Electron Microscope

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Analýza požadavků a teoretický rozbor.
2. Návrh řídicího modulu.
3. Vytvoření funkčního prototypu modulu.
4. Vytvoření řídicího firmware modulu.
5. Testování.
6. Zhodnocení dosažených výsledků práce a závěr.

Seznam doporučené odborné literatury:

[1] VAN SICKLE, Ted. *Programming microcontrollers in C*. 2nd ed. Eagle Rock, Calif.: LLH Technology Pub., c2001, 454 s. ISBN 18-787-0757-4.
[2] PREDKO, Michael. *Programming and customizing the PIC microcontroller.* 3rd ed. New York: McGraw Hill, c2008, xxiii,1263 p. ISBN 978-0071472876.
[3] MONTROSE, Mark I. *Printed circuit board design techniques for EMC compliance: a handbook for designers.* 2nd ed. New York: IEEE Press, c2000, xxv, 307 p. ISBN 0780353765.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Jaromír Konečný, Ph.D.**

Datum zadání: 01.09.2015

Datum odevzdání: 28.04.2017

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Poděkování

Rád bych poděkoval svému vedoucímu mé diplomové práce Ing. Jaromíru Konečnému, Ph.D. za cenné rady a připomínky, dále Ing. Janu Kotyzovi za inspirativní promluvy do duše a nakonec Ing. Liboru Štramberskému, který umožnil vznik celé této práce.

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. 4. 2017

Dvořák

.....

Jan Dvořák

Abstrakt

Úkolem diplomové práce je navrhnout řídící elektroniku pro modernizovaný laboratorní přístroj používaný pro přípravu vzorků pro elektronový mikroskop pro koncového zákazníka. Součástí práce je podrobný popis hardwaru a softwaru.

Klíčová slova

Elektronický spínač, spínaný zdroj, mikrokontrolér, PIC18F4431, alfanumerický displej, filtrace, zesilovač, klávesnice, MPLAB® X IDE, eliminace zákmitů tlačítek

Summary

The aim of the diploma thesis is to design control electronics for the modernized laboratory apparatus used for the preparation of samples for the electron microscope for the end customer. Part of the thesis is a detailed description of hardware and software.

Key words

Electronic switch, switching power supply, microcontroller, PIC18F4431, alphanumeric display, filtration, amplifier, keyboard, MPLAB® X IDE, debouncing

Obsah

Seznam použitých symbolů a zkratek

A	jednotka elektrického proudu Ampér
ADJ	pin pro nastavení výstupního napětí stabilizátoru (Adjust)
C	kapacita kondenzátoru
DC	stejnosměrné napětí / proud (Direct Current)
DPS	deska plošných spojů
EEPROM	elektricky mazatelná a programovatelná paměť(electrically erasable programmable read-only memory)
EM	elektromagnetické jevy, zde ve spojitosti s vyzařováním
EN	pin pro zapnutí (Enable)
ENC	enkodér (Encoder)
ESD	elektrostatický výboj (ElectroStatic Discharge)
ESR	sériový odpor projevující se ve střídavé oblast (equivalent series resistance)
f_C	frekvence řezu RC členu
f_{SCK}	komunikační frekvence SPI modulu
G	bezrozměrné zesílení (Gain)
GND	nulový potenciál (Ground)
I_{ADJ}	proud jdoucí pinem ADJ u stabilizátoru
$I_{COUT(RMS)}$	RMS hodnota zvlnění proudu výstupního kondenzátoru
ICSP™	proprietární rozhraní pro programování a ladění MCU firmy Microchip
I_f	proud v propustném směru diody
I_K	zkratový proud
$I_{L(PK)}$	špičkový proud cívkou
$I_{L(RMS)}$	RMS hodnota proudu cívkou
I_{LED}	proud procházející strukturou několika LED
I_{LPP}	zvlnění proudu cívkou
I_O	maximální přípustný proud diodou
I_{OUT}	výstupní proud
I_{RMS}	povolená RMS hodnota proudu
I_{SAT}	saturační proud
I_{SCOPE}	proud jdoucí do mikroskopu
k	rozlišení impulsu enkodéru
kHz	jednotka frekvence kilo-Hertz, 1 kHz = 1 000 Hz
K_{IND}	koeficient pro výpočet minimální indukčnosti
kΩ	jednotka elektrického odporu kilo-Ohm
L	indukčnost cívky
LCD	displej s tekutými krystaly (Liquid-Crystal Display)
LED	svítící dioda (Light Emitting Diode)
L_{MIN}	minimální požadovaná indukčnost cívky ve spínaném zdroji
LOG0	logická hodnota 0
LOG1	logická hodnota 1
mA	jednotka elektrického proudu mili-Ampér, 1 mA = 0,001 A

MCU	mikrokontrolér (microcontroller unit)
MHz	jednotka frekvence mega-Hertz, 1 MHz = 1 000 000 Hz
ms	jednotka času mili-sekunda, 1 ms = 0,001 s
mV	jednotka napětí mili-Volt, 1 mV = 0,001 V
n	velikost otáček
n_0	velikost otáček motorku na prázdno
nF	jednotka kapacity nano-Farad, 1 nF = 0,000 000 001 F
P	stoupání závitu
PCB	deska plošného spoje (Printed Circuit Board)
P_D	vyzařovaný tepelný výkon
PH	výstupní pin spínaného zdroje
PWM	pulzně šířková modulace (Pulse Width Modulation)
R_{ESR}	hodnota efektivního sériového odporu kondenzátoru
RMS	efektivní hodnota (Root Mean Square)
SMD	součástky pro povrchovou montáž (Surface Mount Device)
SPI	rozhraní sériové komunikace (Serial Peripheral Interface)
SS	pin pozvolného náběhu spínaného zdroje (Slow Start)
T_a	teplota okolí
T_j	teplota polovodičového přechodu
TQFP44	pouzdro mikrokontroléru (Thin Quad Flat Package 44 pin)
t_R	doba náběhu RC členu
T_{SS}	doba náběhu spínaného zdroje
U_{+5V}	napájecí napětí +5 V pro elektroniku
U_1	vstupní napětí obvodu
U_2	výstupní napětí obvodu
U_{CON}	napětí na signálu *CONTRAST*
U_F	úbytek napětí v propustném směru diody
U_{f_typ}	typický úbytek napětí v propustném směru diody
U_{GS}	napětí mezi elektrodami gate a source polem řízeného tranzistoru
$U_{IN(MAX)}$	maximální vstupní napětí
U_{OUT}	výstupní napětí
$U_{OUT(MAX)}$	maximální výstupní napětí
U_{PWM}	amplituda PWM signálu
U_{PWR}	hlavní napájení elektroniky
U_R	závěrné napětí
U_{REF}	referenční napětí
USB	komunikační sběrnice (Universal Serial Bus)
U_{START}	minimální napětí pro spuštění spínaného zdroje
U_{V_TAB}	výstupní napětí stabilizátoru řídícího BOTTOM motorek
VO	výstupní pin stabilizátoru napětí
ΔU	úbytek napětí na diodě
ΔU_{+5V}	zvlnění výstupního napětí spínaného zdroje
ΔU_{IN}	zvlnění napětí na vstupu spínaného zdroje
Θ_{ja}	tepelný odpor mezi polovodičovým přechodem a okolím

μA	jednotka proudu mikro-Ampér, 1 μA = 0,000 001 A
τ	časová konstanta tlumivky

Seznam ilustrací

Seznam tabulek

Úvod

V první kapitole je pro snadnější vyhledávání uveden abecedně seřazený popis názvosloví a pojmů nacházejících se v práci.

Obsahem druhé kapitoly je analýza požadavků, jež jsou kladeny na řídící elektroniku, a specifických přání a požadavků ze strany zákazníka. Dále jsou zde stručně uvedeny nerealizované varianty řešení s důvody jejich odmítnutí.

Třetí kapitola obsahuje popis hardwaru a blokové schéma řídící elektroniky. Jsou zde také uvedené vztahy a propojení mezi jednotlivými bloky následované podrobným popisem jednotlivých bloků s uvedením hodnot součástek a v případě napájecího zdroje i s podrobným popisem návrhu s matematickými vztahy a výpočty relevantních parametrů.

Pátá kapitola se zaměřuje na softwarovou část práce. Jako první je uveden letmý popis vývojového prostředí, následovaný příkladem vytvoření nového projektu a upozorněním na specifikum práce se zvoleným typem mikrokontrolérů. Poté následuje popis jednotlivých logických částí programu místy doplněný o podrobnější vysvětlení neobvyklých nebo důležitých částí kódu.

1 Specifikace a vysvětlení použitého názvosloví a výrazů

V textu popisující hardware a software jsou použity výrazy a názvosloví, které pozvolna vznikaly při komunikaci se zákazníkem. Zde jsou jednotlivé výrazy abecedně seřazeny a specifikovány a dále v textu jsou pro lepší identifikaci formátovány velkým písmem a kurzívou. Hlavní zastoupení zde mají názvy signálních vodičů.

ARM	označení spojitosti s motorkem v brusném ramenu, také tlačítko pro ovládání brusného motorku
ARM DRIVER	obvod pro napájení brusného motorku
AUTO_STOP	signál umožňující detekovat dokončení broušení do požadované hloubky
BACK COVER CONNECTOR	konektor zadního krytu, přivádí napájecí napětí, je místem vyvedení signálu *AUTO_STOP* a vodiče *V_DIM* na konektory pro banánky
BEEP	signál pro řízení piezoelektrického bzučáku
BOTTOM	označení spojitosti s otočnou plochou, na které je umístěn broušený vzorek, také označení rotačního motorku pohybujícího s otočnou plochou
BOTTOM DRIVER	obvod pro napájení rotačního motorku
BOTTOM_LED	LED použitá k podsvícení broušeného vzorku
BUT_COL0	signál prvního sloupce matice tlačítek
BUT_COL1	signál druhého sloupce matice tlačítek
BUT_COL2	signál třetího sloupce matice tlačítek
BUT_COL3	signál čtvrtého sloupce matice tlačítek
BUT_ROW0	signál prvního řádku matice tlačítek
BUT_ROW1	signál druhého řádku matice tlačítek
BUT_ROW2	signál třetího řádku matice tlačítek
CONTRAST	signál nastavující kontrast znaků na alfanumerickém displeji
DIM_CTRL	signál řídící napájení brusného motorku
DISP_E	signál aktivující zápis do displeje
DISP_RS	signál pro volbu mezi zobrazitelnými znaky a řídícími daty při zápisu do displeje
ICSPCLK	hodinový signál ICSP™ rozhraní pro nahrávání a ladění programu v mikrokontroléru
ICSPDAT	datový signál ICSP™ rozhraní pro nahrávání a ladění programu v mikrokontroléru
LAMP	tlačítko ovládající podsvícení pomocí spodní LED a osvícení mikroskopem

LIGHT_COL0	signál prvního sloupce matice LED párů
LIGHT_COL1	signál druhého sloupce matice LED párů
LIGHT_COL2	signál třetího sloupce matice LED párů
LIGHT_ROW0	signál prvního řádku matice LED párů
LIGHT_ROW1	signál druhého řádku matice LED párů
$\overline{\text{MCLR}}$	resetovací signál ICSP™ rozhraní
MICROSCOPE_CTRL	signál řídící pomocí PWM jas osvětlení vzorku v mikroskopu
PWM_LIGHT	signál řídící pomocí PWM jas podsvícení vzorku
PWM_MOTOR	signál řídící pomocí PWM rychlost otáčení brusného motorku
QEI_A	signál posílající z enkodéru informaci o otáčení
QEI_B	signál posílající z enkodéru informaci o otáčení
SAMPLE_BCKLGHT	vodič vedoucí napájení k diodě pro podsvícení vzorku
SCK	hodinový signál SPI rozhraní
SCREEN_DEPTH	enumerační hodnota pro zobrazení obrazovky s hloubkou zabroušení
SCREEN_IDLE	enumerační hodnota pro indikaci klidového stavu displeje
SCREEN_INTRO	enumerační hodnota pro zobrazení úvodní obrazovky
SCREEN_LIGHT	enumerační hodnota pro zobrazení obrazovky s nastavením jasu LED a mikroskopu
SCREEN_SPEED	enumerační hodnota pro zobrazení obrazovky s nastavením rychlosti
SCREEN_TIMER	enumerační hodnota pro zobrazení obrazovky s nastavením časovače
SDI	datový signál SPI rozhraní, příjem dat
SDO	datový signál SPI rozhraní, vysílání dat
SPEED-	tlačítko pro snížení rychlosti otáčení brusného motorku
SPEED+	tlačítko pro zvýšení rychlosti otáčení brusného motorku
TAB_CTRL	signál řídící napájení rotačního motorku
TABLE	tlačítko pro ovládání rotačního motorku
TIMER	tlačítko pro spuštění a vypnutí časovače
TIMER-	tlačítko pro zmenšení hodnoty časovače
TIMER+	tlačítko pro zvětšení hodnoty časovače
V_DIM	vodič napájející brusný motorek
V_TAB	vodič napájející rotační motorek
ZERO	tlačítko pro nulování proměnných

2 Analýza požadavků

Základním požadavkem na kterém celý projekt stojí je modernizace zastaralého laboratorního zařízení, dále jen přípravek. Stáří původního návrhu je přes 30 let a na jeho vzhledu a technologickém provedení to je znát. Přípravek slouží pro broušení vzorků pro elektronové mikroskopy do mikrometrových tlouštěk. Proto musí přípravek splňovat přísné podmínky na mechanickou přesnost a jemnost. Zároveň s novým návrhem je požadována modernizace elektroniky, aby vyhovovala současným standardům. To znamená maximální možnou miniaturizaci, zlepšení energetické účinnosti, použití moderních způsobů řízení a jako samozřejmý bonus i nějaké funkce navíc, co původní přípravek neměl.

Obr. 1: Původní zařízení [1]

Modernizace elektroniky je relativně snadná. Oproti původnímu přípravku, který používal klasické vývodové součástky, stačí přejít do SMD oblasti. Tím uspokojíme požadavek na miniaturizaci. Energetická účinnost se dá zlepšit použitím spínaných zdrojů, starší přípravek používal pouze lineární stabilizátory. Moderní způsob řízení jednoduše znamená použití nějakého řídícího mikrokontroléru. Díky tomu je také možné relativně lehce uspokojit nevyřčený požadavek na nějaké funkce navíc.

Obr. 2: Nová verze zařízení [2]

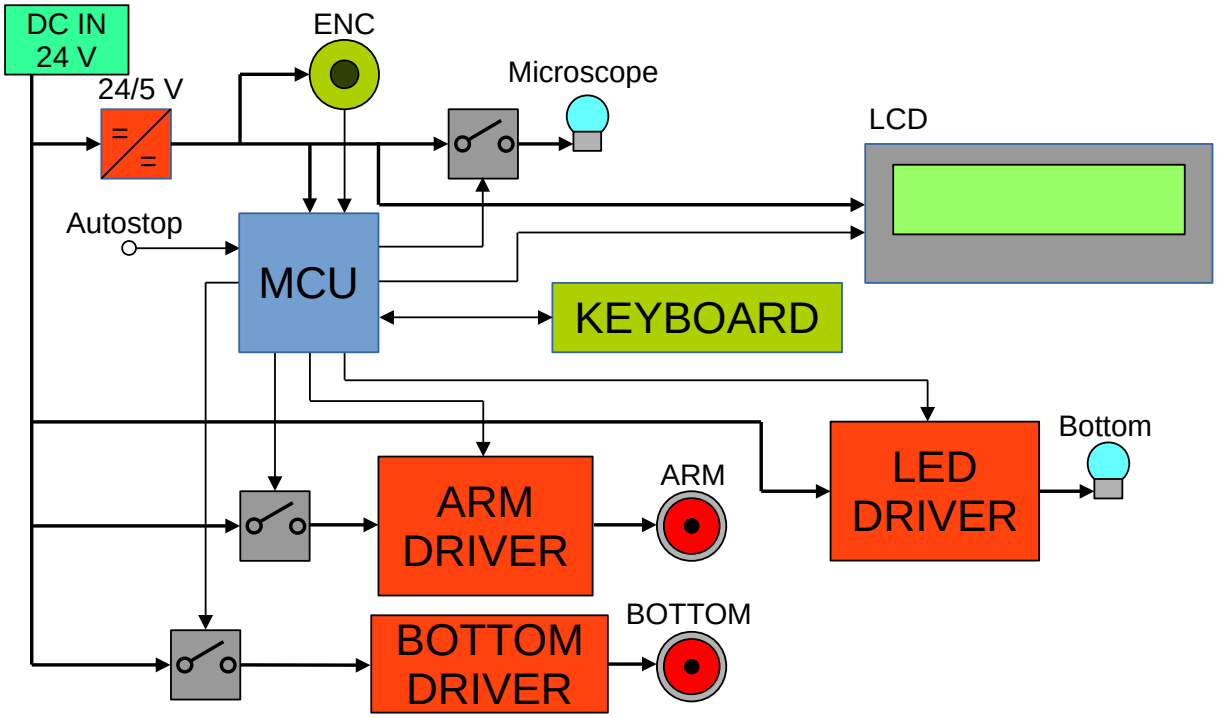
Jako první bylo nabídnuto velice řešení s MCU architektury ARM, to ovšem bylo odmítnuto, neboť zákazník je zvyklý na práci s MCU PIC, které proto také požadoval. Po zvolené platformě je třeba rozhodnout o způsobu měření hloubky zabroušení. Po několika neúspěšných pokusech s mikrometry je zvoleno měření pomocí enkodéru. Nakonec zbývá napájení. Pro napájení elektroniky je zvolen spínaný zdroj. Pokus o použití spínaného zdroje a nebo PWM pro řízení motorků se nesetkává s pochopením, spíš s obavami o plynulost a jemnost jejich běhu a tak je zvoleno použití

lineárních stabilizátorů. Jako zobrazovač je nakonec zvolen modul dvouřádkového displeje, i když krátkou dobu se uvažovalo o OLED modulu s pěknějšími a živějšími barvami.

Takže výsledkem veškerých jednání a experimentů je zařízení běžící na platformě PIC firmy Microchip, s uživatelským rozhraním řešeným klávesnicí a dvouřádkovým displejem. Měření je řešeno pomocí enkodéru, motorky jsou řízené lineárními stabilizátory a celá elektronika je napájena spínaným zdrojem.

3 Popis hardwaru řídicí jednotky

Na následujících řádcích je uveden popis řídicí jednotky nejdříve z pohledu vzájemných propojení a vztahů mezi funkčními bloky následovaný podrobným popisem jednotlivých funkčních bloků. Pro lepší orientaci je na Obr. 3: uvedeno blokové schéma celé jednotky i spolu s částmi které se na PCB nenacházejí.



Obr. 3: Blokové schéma řídicí jednotky

3.1 Popis vztahů mezi jednotlivými bloky řídicí jednotky

Na samotném PCB se nacházejí bloky napájení 24/5 V, MCU, elektronické spínače a obvody pro řízení motorků spolu s řízením *BOTTOM_LED*. Blok DC IN 24 V v sobě funkčně slučuje externí adaptér spolu se vstupními napájecími obvody, blok LCD představuje základní obvody pro displej a samotný displej nasazený na řídicí jednotce. Blok ENC odpovídá enkodéru připojenému k řídicí jednotce. Součástí popisu vztahů je i jmenování jednotlivých signálů a komunikačních rozhraní.

3.1.1 DC IN 24 V

Vstupní napájení. Dodává 24 V do zdroje 24/5 V a do obvodů řízení DC motorků.

3.1.2 24/5 V

Zdroj napětí 5 V pro veškerou digitální elektroniku. Napájí mikrokontrolér, displej, LED a enkodér. Zdroj napětí pro pull-up rezistory na vstupech klávesnice a vstupu autostop.

3.1.3 MCU

Řídicí mikrokontrolér. Napájený napětím 5 V. Pomocí signálů *DIM_CTRL* a *TAB_CTRL* řídí elektronické spínače obou motorků a signálem *PWM_MOTOR* ovládá rychlost *ARM* motorku. Kombinací komunikačního rozhraní SPI a logických signálů *DISP_E* a *DISP_RS* řídí displej. Signálem *PWM_LIGHT* řídí jas *BOTTOM_LED* a signálem *MICROSCOPE_CTRL* řídí osvětlovací LED mikroskopu. Enkodér je připojený signály *QEI_A* a *QEI_B*. Klávesnice je připojena signály *BUT_ROW0 – BUT_ROW2* a *BUT_COL0 – BUT_COL3*. MCU také pomocí signálů *LIGHT_ROW0*, *LIGHT_ROW1* a *LIGHT_COL0 – LIGHT_COL2* ovládá signální LED na klávesnici. Signál *AUTO_STOP* je vyvedený mimo zařízení a přes konektor je připojený na brusné rameno.

3.1.4 LCD

Napájený napětím 5 V. Řízený jednosměrnou komunikací po SPI sběrnici a signály *DISP_E* a *DISP_RS*.

3.1.5 ARM DRIVER

Napájený hlavním napětím 24 V. Řídícími signály jsou *PWM_MOTOR a DIM_CTRL*.

3.1.6 BOTTOM DRIVER

Napájený hlavním napětím 24 V. Řídícím signálem je *TAB_CTRL*.

3.1.7 LED DRIVER

Napájený napětím 5 V. Řídícím signálem je *PWM_LIGHT*.

3.1.8 KEYBOARD

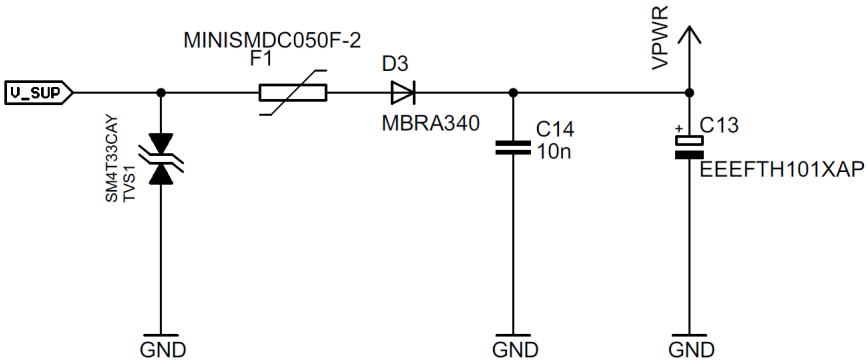
Pracuje s napětím 5 V. Blok je k MCU připojen signály *BUT_ROW0, BUT_ROW1, BUT_ROW2, BUT_COL0, BUT_COL1, BUT_COL2, BUT_COL3*, *LIGHT_ROW0, LIGHT_ROW1, LIGHT_COL0, LIGHT_COL1 a LIGHT_COL2*.

3.1.9 ENC

Encodér je napájený napětím 5 V a s MCU je propojen signály *QEI_A* a *QEI_B*.

3.2 Popis funkčních bloků

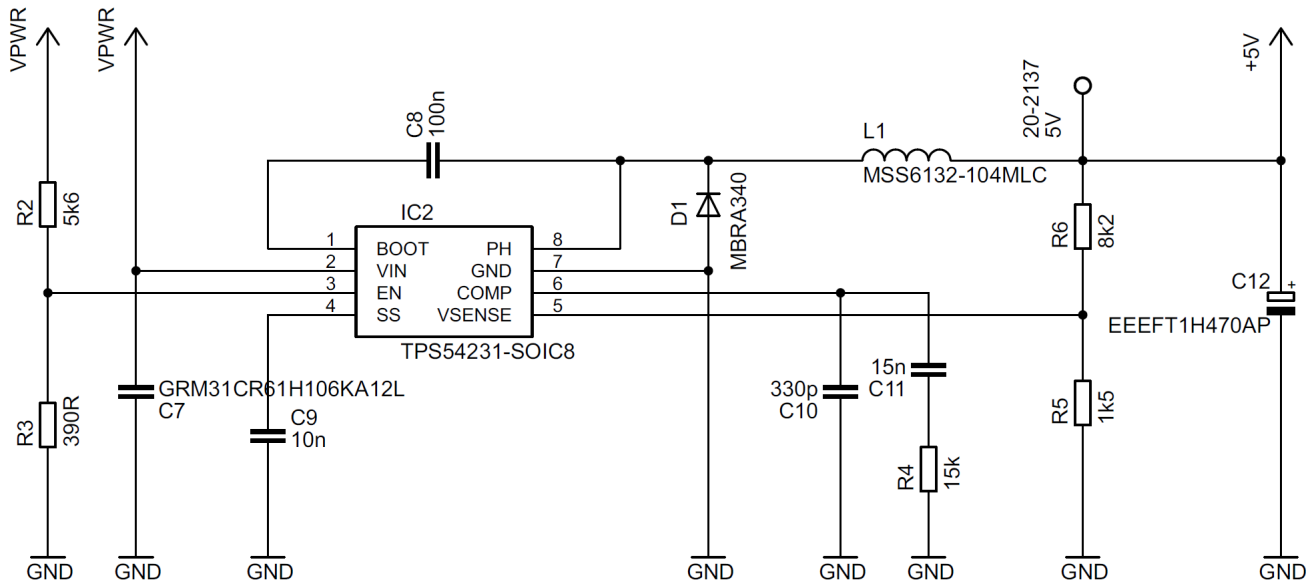
3.2.1 DC IN 24 V



Obr. 4: Schéma vstupního obvodu napájení

Hlavní napájení celého zařízení, jehož účelem je prvotní ochrana elektroniky a základní vyhlazení napájení. Za součást napájecího bloku je také považován vstup napětí do zařízení a hlavní vypínač, ze kterého je vedeno kladné napětí, spolu s vodičem nulového napětí z napájecího konektoru, na *BACK COVER CONNECTOR* popsaný níže. Vstupní obvod na desce sestává z transilu TVS1 pro potlačení případných napěťových špiček jež by se z napájení dostaly až k TVS1. Dalším ochranným prvkem je vratná pojistka F1, jež bez omezení umožní průchod maximálnímu proudu 0,5 A a k plnému omezení dochází od proudu 1 A. Posledním ochranným prvkem je dioda D3 chránící před přepólováním napájecího napětí. Jedná se o výkonovou Schottky diodu pro povrchovou montáž, jejíž maximální úbytek napětí při teplotě přechodu 25 °C je 0,45 V.[12] Posledním funkčním prvkem bloku jsou vyhlazovací kondenzátory pro vyhlazení napájecího napětí U_{PWR}.

3.2.2 24/5 V [18]



Obr. 5: Schéma obvodu spínaného zdroje

Zdroj napětí 5 V pro digitální elektroniku. Základem je obvod TPS54231 firmy Texas Instruments. Maximální vstupní napětí je 28 V, při výstupním proudu až 2 A a typická pracovní frekvence je 570 kHz. Hlavní důvod pro výběr obvodu od firmy TI je zjednodušený proces návrhu spínaného zdroje popsaný v katalogovém listu obvodu, čímž se výrazně urychluje vývoj elektroniky. Požadovanými parametry zdroje jsou: výstupní napětí U_{+5V} = 5 V, výstupní proud 400 mA, maximální zvlnění výstupního napětí při maximálním proudu ΔU_{+5V} = 75 mV (1,5 %).

Jako první se při návrhu specifikuje výstupní napětí zdroje. Výstupní napětí je určeno hodnotou odporového děliče R5 a R6. Nejprve se zvolí odpor rezistoru R6 na hodnotu 8,2 kΩ a pro výstupní napětí 5 V se podle rovnice 1 dopočítá odpor druhého rezistoru R6 = 1,56 kΩ. Zvolí se nejbližší hodnota rezistoru z řady E12, což je 1,5 kΩ. Kontrolním přepočtem podle rovnice 2 vyjde výstupní napětí zdroje U_{+5V} = 5,17 V, což je v toleranci napájení digitálních obvodů i při započtení maximálního povoleného zvlnění výstupního napětí.

$$R5 = \frac{R6 \cdot U_{REF}}{U_{+5V} - U_{REF}} \tag{1}$$

$$U_{+5V} = U_{REF} \cdot \left[\frac{R6}{R5} + 1\right] \tag{2}$$

Dalším krokem je zvolení vstupního kondenzátoru C7. Doporučená hodnota kapacity C = 10 µF je uvedená jako dobře pracující v širokém rozsahu aplikací, proto byl použit kondenzátor s touto kapacitou. Dalšími požadavky na kondenzátor je jeho maximální pracovní napětí, které by mělo být vyšší než maximální vstupní napětí včetně případného zvlnění a hodnota ESR, která ovlivňuje zvlnění napětí na kondenzátoru. ESR zvoleného typu kondenzátoru dosahuje při pracovní frekvenci obvodu TPS54231 hodnoty R = 0,045 Ω. Při této hodnotě ESR je zvlnění napětí na vstupním kondenzátoru ΔU_{IN} = 35,54 mV podle rovnice 3. Maximální pracovní napětí zvoleného kondenzátoru 50 V tedy plně postačuje. Jako poslední je třeba ověřit, jestli RMS hodnota proudu vyvolaného zvlněním napětí procházejícího skrz kondenzátor nezpůsobí jeho poškození. Podle rovnice 4 vychází tento proud na I_{CIN} = 0,2 A. Graf vlivu proudu na ohřev v katalogovém listu pro použitý kondenzátor zobrazuje křivky ohřevu pro frekvence 100 kHz, 500 kHz a 1 MHz v rozsahu 0 - 6 A. Vzhledem k tomu, že křivka pro frekvenci 500 kHz, jež je nejblíže pracovní frekvenci, strmě začíná až od proudu nepatrně nižšího než 1 A, tak se dá předpokládat, že daná RMS hodnota proudu bude mít zanedbatelný vliv na ohřev kondenzátoru a tudíž nezpůsobí jeho poškození. [7]

$$\Delta U_{\mathrm{IN}} = \frac{I_{OUT(MAX)} \cdot 0{,}25}{C_{BULK} \cdot f_{SW}} + I_{OUT(MAX)} \cdot ESR_{MAX} \tag{3}$$

$$I_{CIN} = \frac{I_{OUT(MAX)}}{2} \tag{4}$$

Následuje volba komponent pro filtraci výstupního napětí, cívka L1 a kondenzátor C12. Nejdříve je třeba spočítat minimální hodnotu indukčnosti cívky. Tato vychází na L_{MIN} = 88,49 µH podle rovnice 5 při dosazení za $U_{OUT} = U_{OUT(MAX)}$ = 5,17 V, I_{OUT} = 0,4 A a $U_{IN(MAX)}$ = 23,55 V. Koeficient

K_{IND} se volí podle hodnoty ESR výstupního kondenzátoru. Při nízkém ESR může být $K_{IND} = 0,3$, při vyšším ESR se dosahuje lepších výsledků při $K_{IND} = 0,2$. Volba je ponechána na návrháři, při výpočtu minimální hodnoty indukčnosti bylo použito $K_{IND} = 0,2$ kvůli použití elektrolytického filtračního kondenzátoru na výstupu s vyšší hodnotou ESR. Vzhledem k tomu, že se jedná pouze o minimální hodnotu indukčnosti, tak volba je v tomto případě opět ponechána na návrháři. Katalogový list popisuje vliv vyšší hodnoty indukčnosti pouze ve snížení střídavé části proudu procházejícího cívkou a tím i snížení zvlnění výstupního napětí, proto byla pro návrh zdroje použita cívka s indukčností L = 100 µH. Dalšími důležitými parametry cívky jsou hodnoty jejího RMS a saturačního proudu. Pro určení těchto hodnot je třeba spočítat proudové zvlnění podle rovnice 6 a podle rovnic 7 a 8 spočítat hodnoty RMS a špičkového proudu. Hodnoty při zvolené indukčnosti pro požadovaný proud a výstupní napětí vycházejí: zvlnění $I_{LPP} = 88,49$ mA, RMS proud $I_{L(RMS)} = 400,01$ mA a špičkový proud $I_{L(PK)} = 444,25$ mA. Parametry zvolené cívky jsou L = 100 µH, $I_{RMS} = 690$ mA a $I_{SAT} = 450$ mA.[15]

$$L_{MIN} = \frac{V_{OUT(MAX)} \cdot \left(V_{IN(MAX)} - V_{OUT}\right)}{V_{IN(MAX)} \cdot K_{IND} \cdot I_{OUT} \cdot F_{SW}} \tag{5}$$

$$I_{LPP} = \frac{V_{OUT} \cdot \left(V_{IN(MAX)} - V_{OUT}\right)}{V_{IN(MAX)} \cdot L_{OUT} \cdot F_{SW} \cdot 0,8} \tag{6}$$

$$I_{L(RMS)} = \sqrt[2]{I_{OUT(MAX)}^2 + \frac{1}{12} \cdot I_{LPP}^2} \tag{7}$$

$$I_{L(PK)} = I_{OUT(MAX)} + \frac{I_{LPP}}{2} \tag{8}$$

Další důležitou částí filtru je výstupní kondenzátor C12. Vzhledem k limitům a možnostem vnitřních struktur obvodu TPS54231 je minimální hodnota kapacity výstupního kondenzátoru 3,6 µF, což je nedostatečná hodnota pro případ neočekávaného dočasného zastavení spínaného zdroje, kdy by výstupní kondenzátor měl dočasně převzít roli zdroje energie. Proto byla zvolena hodnota kapacity 47 µF a pro tuto hodnotu byla spočítána maximální hodnota $R_{ESR} = 0,850$ Ω podle rovnice 9. Poslední hodnotou pro určení správného typu kondenzátoru je RMS proudové zvlnění, které dle rovnice 10 vychází na $I_{COUT(RMS)} = 25,54$ mA. Podle těchto hodnot byl nakonec zvolen kondenzátor s těmito parametry: C = 470 µF, ESR = 0,68 Ω a $I_{RMS} = 195$ mA.[6]

$$R_{ESR} = \frac{\Delta U_{+5V}}{I_{LPP}} - \frac{\frac{V_{OUT}}{V_{IN(MAX)}} - 0,5}{4 \cdot f_{SW} \cdot C} \tag{9}$$

$$I_{COUT(RMS)} = \frac{1}{\sqrt{12}} \cdot I_{LPP} \tag{10}$$

TPS54231 používá externí kompenzaci ve formě sério-paralelní kombinace kondenzátorů C10, C11 a rezistoru R4. Externí kompenzace dokáže ještě více vylepšit vlastnosti zdroje, především

stabilitu. Bohužel z důvodu časového skluzu ve vývoji bylo rozhodnuto použít výchozí hodnoty součástek ze staršího projektu.

Poslední částí, kterou je potřeba určit je zpětná dioda, přes kterou se uzavírá proudová smyčka v okamžiku, kdy je spínací tranzistor obvodu uzavřen a pin PH odpojen. Proud diody proto musí být v propustném směru větší, než proud zdroje I_{OUT} plus polovina I_{LPP}, nebo-li musí být větší než $I_{L(PK)}$ = 444,25 mA. Závěrné napětí diody musí být větší než napětí přítomné na pinu PH při otevřeném tranzistoru, což je 23,55 V. Zvolená dioda, stejného typu jako ve vstupním obvodu napájení, má závěrné napětí U_R = 40 V a maximální propustný proud I_O = 3 A. Dalším parametrem, určujícím hlavně efektivitu, je úbytek napětí na diodě. Použitá dioda má při teplotě přechodu 25 °C maximální úbytek napětí U_F = 0,45 V.[12]

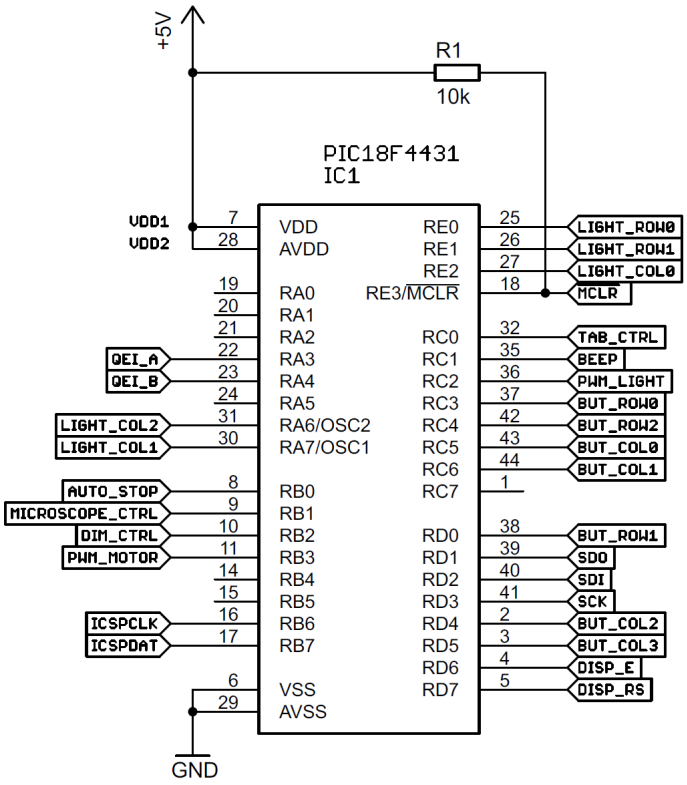
Optimálně je možné obvod vybavit ochraně proti podpětí a funkcí pozvolného náběhu napětí. Pozvolného náběhu napětí je docíleno připojením kondenzátoru C9 na pin SS. Použití této funkce je doporučeno i přímo v katalogovém listu, ale hodnota kondenzátoru by neměla překročit 27 nF. Doba náběhu je nastavena, dle rovnice 11, na T_{SS} = 4 ms. Ochrana proti podpětí je realizována napěťovým děličem z rezistorů R2 a R3 připojených na pin EN, kdy je možné nastavit nejen vypínací napětí, ale díky funkci hystereze i napětí, kterého je třeba dosáhnout pro spuštění obvodu. Při návrhu obvodu byla nastavena hystereze na 16,8 mV podle rovnice 12 použitím rezistoru z řady E12 o odporu R2 = 5600 Ω, čímž bylo dosaženo téměř stejného bodu zapnutí a vypnutí obvodu. Rezistorem R3 se stanoví podle rovnice 13 hodnota spouštěcího napětí U_{START} = 19,19 V při hodnotě odporu rezistoru R3 = 390 Ω.

$$T_{SS} = \frac{C_{SS}(nF) \cdot V_{ref}(V)}{I_{SS}(\mu A)} \tag{11}$$

$$R2 = \frac{U_{START} - U_{STOP}}{3\ \mu A} \tag{12}$$

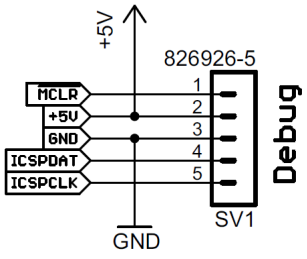
$$R3 = \frac{U_{EN}}{\frac{U_{START} - U_{EN}}{R2} + 1\,\mu A} \tag{13}$$

3.2.3 MCU [16]



Obr. 6: Řídicí MCU

Pro řízení elektroniky a celého zařízení je použit mikrokontrolér PIC18F4431 od firmy Microchip v SMD pouzdře TQFP44. Jeho primární zaměření je na řízení a obsluhu motorů. K tomu účelu je vybaven osmi specializovanými PWM kanály, rozhraním pro kvadraturní enkodér a mnoha analogovými vstupy. Napájení ze zdroje U_{+5V} je přivedeno na dva napájecí piny MCU, na nichž je po dvojici blokovacích kondenzátorů o hodnotách 1 a 100 nF. Mikrokontrolér má nastaven jako zdroj taktovacích hodin vnitřní generátor. Frekvence generátoru je za účelem snížení EM vyzařování nastavena na pracovní frekvenci 8 MHz, při které MCU zvládá provádět nahraný firmware bez pozorovatelného zpoždění a prodlev. Pro nahrávání firmware a případné ladění kódu slouží ICSP™ rozhraní jež se 5-ti pinovým rozhraním připojuje k ICSP™ programátoru, v případě tohoto projektu je použit PICkit™ 3, který je zároveň možné použit i jako In-Circuit Debugger.



Obr. 7: Konektor rozhraní ICSP™

ICSP™ rozhraní sestává z napájecích vodičů sloužících jako zdroj napětí pro elektroniku, jež v ICSP™ programátoru zajišťuje komunikaci s cílovým mikrokontrolérem. Toto řešení umožňuje jednoduše dodržet správné napěťové úrovně při digitální komunikaci. $\overline{MCLR}$ signál slouží jako reset mikrokontroléru a také slouží k programování a ladění. Stejnojmenný pin $\overline{MCLR}$ potřebuje pro svou správnou funki připojený Pull-Up rezistor. Komunikaci s mikrokontrolérem zajišťují signály *ISCPDAT* a *ICSPCLK*. *ICSPCLK* je hodinový signál udávající takt komunikaci jdoucí po obousměrném datovém sériovém vodiči *ICSPDAT*.

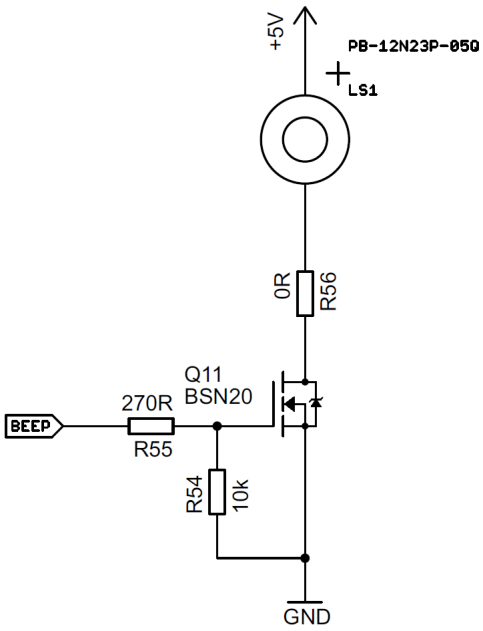
Hlavním důvodem pro výběr tohoto typu je přítomnost rozhraní pro kvadraturní enkodéry umožňující jednoduché a přímé propojení s enkodérem a zpracování signálu z enkodéru na hardwarové úrovni. Tímto řešením se mnohonásobně zvyšuje přesnost, spolehlivost a rychlost využití dat dodaných činností enkodéru v porovnání se softwarovým zpracováním. Enkodér je připojen na dedikované piny RA3 a RA4 signály *QEI_A* a *QEI_B*. Port A je jinak navržen také pro čtení analogových hodnot z případných okolích obvodů. Vedle enkodéru jsou zde přivedeny také signály *LIGHT_COL1* a *LIGHT_COL2* pro maticové řízení LED umístěných na klávesnici.

Port B mikrokontroléru je vedle univerzálních vstupů a výstupů vybaven 6-ti PWM kanály sdruženými do tří párů. Z těchto kanálů jsou použity pouze dva na pinech RB1 a RB3. PWM signály *MICROSCOPE_CTRL* a *PWM_MOTOR* jsou připojeny k odpovídajícím funkčním blokům. Signál *DIM_CTRL* řídí napájení *ARM DRIVERu*. Dále je zde přiveden vnější signál *AUTO_STOP,* vedený z brusného ramene a oznamující mikrokontroléru zda bylo broušení dokončeno. Nakonec jsou na pinech RB6 a RB7 připojeny signály *ICSPCLK* a *ICSPDAT* programovacího rozhraní ICSP™.

Port C je obecný port s připojenými signály modulů čítačů/časovačů, univerzálních vstupů a výstupů a digitálních komunikačních rozhraní. Připojený logický signál *TAB_CTRL* řídí napájení *BOTTOM DRIVERu* a signál *BEEP* slouží k aktivaci piezo bzučáku. Signál *PWM_LIGHT* je výstupem z obecného čítače nakonfigurovaného jako PWM a je přiveden jako řízení *LED DRIVERu*. Zbývající 4 signály *BUT_ROW0*, *BUT_ROW2*, *BUT_COL0* a *BUT_COL1* jsou součástí maticového rozhraní pro připojení ovládací klávesnice.

Port D je smíšený. Je zde vyvedený signál modulu čítače/časovače, signály pro digitální komunikaci a tři piny mají přivedené výstupu PWM kanálů stejného typu jako u portu B. Na pinech RD1, RD2 a RD3 jsou vyvedeny signály SPI modulu *SDO*, *SDI*, *SCK*. I když jsou využity jen signály *SCK* a *SDO*, tak byl vyveden i signál SDI pro případ, že by někdy v budoucnu bylo třeba po SPI sběrnici něco vyčítat a nedošlo k obsazení pinu jinou funkcí. Piny RD6 a RD7 jsou použity jako signální pro řízení činnosti alfanumerického displeje. Zbytek pinů je obsazeno zbylými signály maticového rozhraní klávesnice *BUT_ROW1*, *BUT_COL2* a *BUT_COL3*.

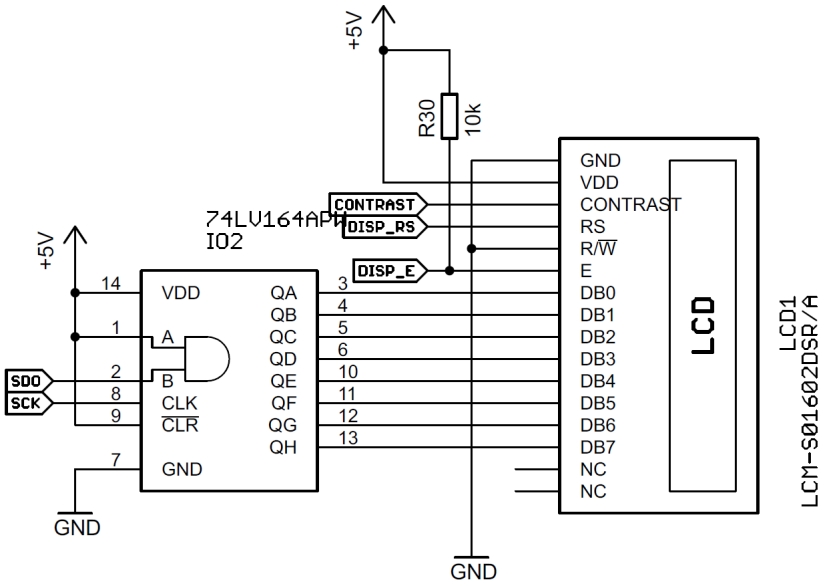
Posledním portem je port E obsahující pouze 4 piny, z nichž jeden je použit pro $\overline{MCLR}$ signál rozhraní ICSP™, zbylé tři jsou použity pro zbylé signály *LIGHT_ROW0*, *LIGHT_ROW1* a *LIGHT_COL0* maticového zapojení LED na displeji.



Obr. 8: Obvod piezo bzučáku

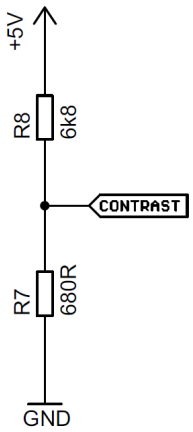
Pod logickým blokem MCU je také umístěn jednoduchý obvod pro řízení piezo bzučáku, sestávající pouze ze bzučáku samotného a elektronického spínače. Rezistor R55 je zde pro zaoblení hrany řídícího signálu zpomalením přechodového jevu, způsobeného kapacitou řídící elektrody tranzistoru, a tím zmenšení EM vyzařování. Rezistor R56 byl v obvodu zapojen pro snížení hlasitosti omezením proudu. Nyní má nulovou hodnotu, ale fyzicky nebyl odstraněn, protože by to znamenalo změnit návrh tištěného spoje a ve výsledku výrobu nové šablony pro výrobu DPS. Z ekonomického hlediska má tak větší smysl pouhá změna hodnoty.

3.2.4 LCD [9]



Obr. 9: Schéma bloku LCD

Pro zobrazení informací uživateli je použit široce rozšířený typ LCD modulu v konfiguraci 2x16 znaků s paralelním komunikačním rozhraním. Z důvodu omezeného počtu pinů mikrokontroléru je displej řízen kombinací SPI rozhraní a signálních vodičů. U konektoru displeje je sériový tok dat po SPI sběrnici přiveden signály *SCK* a *SDO* na posuvný registr, kde je převeden na paralelní uspořádání. Pomocí řídících signálů *DISP_RS* a *DISP_E* jsou data předána řídícím obvodům displeje, kde jsou zpracována jako při normální paralelní komunikaci. Posuvný registr má samozřejmě připojeny zde nezobrazené blokovací kondenzátory na napájecích pinech pro spolehlivou činnost i při napěťových špičkách. Testováním bylo zjištěno, že signál *DISP_E* je třeba posílit pull-up rezistorem. V opačném případě displej nepracuje korektně. Pro správnou viditelnost textu na displeji je také třeba zvolit správnou hodnotu napětí na signálu *CONTRAST*.



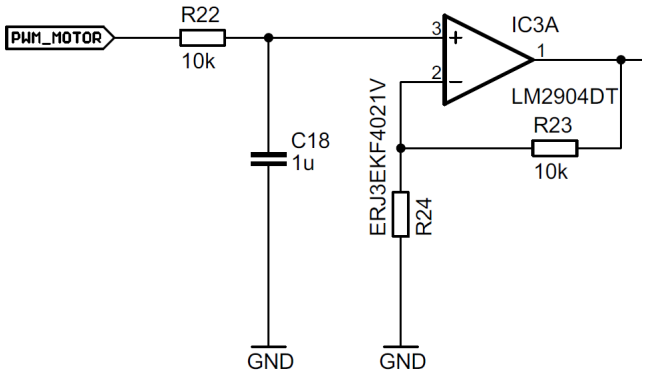
Obr. 10: Odporový dělič pro nastavení kontrastu

Pro viditelnost textu je rozhodující rozdíl mezi napájecím napětím displeje a napětím na signálu *CONTRAST*. Uvedené hodnoty použitých rezistorů byly získány experimentálně. Napětí na signálu *CONTRAST* je podle 14 U_{CON} = 0,45 V.

$$U_{CON} = U_{+5V} \cdot \frac{R7}{R7 + R8} \tag{14}$$

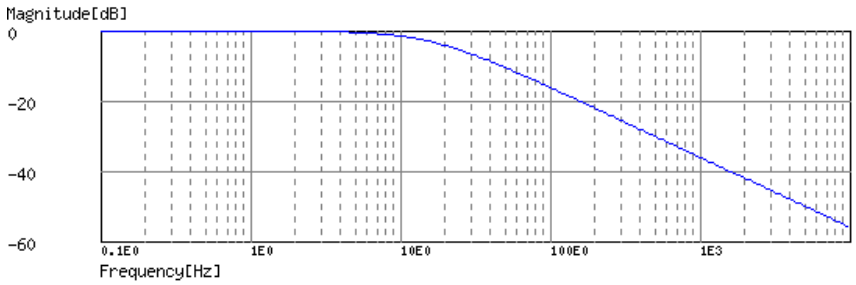
3.2.5 ARM DRIVER

Regulační a výkonové obvody řídící rychlost otáčení *ARM* motorku. Obvody pracují bez zpětné vazby, protože bylo požadováno, aby rychlost rotace byla proměnlivá podle zatížení. V případě použití zpětné vazby by sice rychlost byla stálá téměř bez ohledu na zatížení, avšak s ohledem na požadovanou přesnost by tato vlastnost byla spíše na škodu, protože je potřeba aby broušení probíhalo pozvolna. Navíc u starší verze nebyla použita žádná zpětná vazba a výstupní vzorky byly vybroušeny s dostatečnou přesností a jemností.

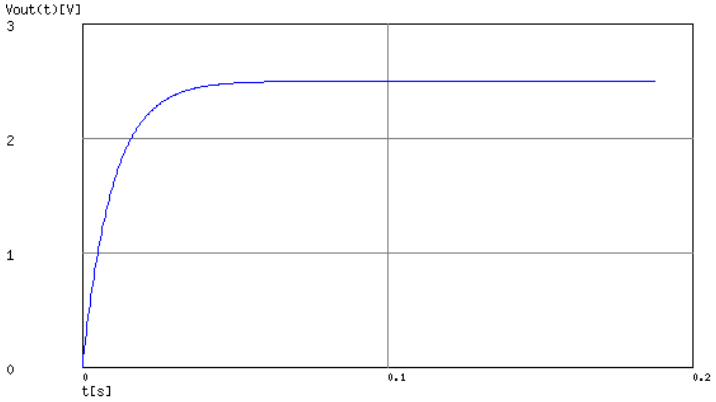


Obr. 11: Obvod pro filtraci a zesílení signálu

PWM_MOTOR signál je nejdříve vyfiltrován na RC členu R22 a C18. Frekvence řezu RC členu je f_C = 15,915 Hz a doba náběhu je t_R = 23 ms. Dále z grafu amplitudové charakteristiky je možné odhadnout, že útlum členu při frekvenci PWM signálu 20 kHz je pod −60 dB což podle rovnice 15 odpovídá zesílení menšímu než G = 0,001.



Obr. 12: Amplitudová charakteristika RC členu [13]

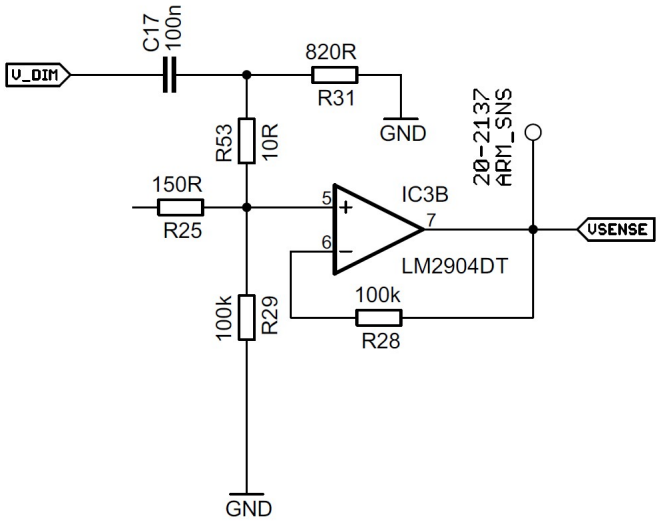


Obr. 13: Přechodová charakteristika RC členu při PWM 20 kHz, 50% [13]

$$G = 10^{\frac{G_{dB}}{20}} \tag{15}$$

Vzniklý vyfiltrovaný analogový signál je poté zesílen operačním zesilovačem IC3A podle rovnice 16. Při amplitudě PWM signálu $U_1 = U_{PWM} = 5$ V a střídě 100 % je na výstupu zesilovače $U_2 = 17{,}44$ V. Takto zesílený signál je přiveden do součtového zapojení následujícího operačního zesilovače.

$$U_2 = U_1 \cdot \left(1 + \frac{R23}{R24}\right) \tag{16}$$

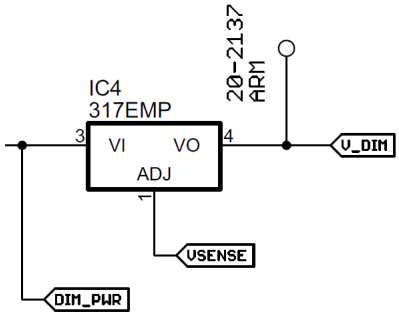


Obr. 14: Obvod pro řízení ADJ pinu lineárního stabilizátoru

V součtovém zapojení operačního zesilovače IC3B je vyfiltrovaný a zesílený signál sečten se signálem získaným na spojení kondenzátoru C17 a rezistoru R31 zapojených do série, na něž je přivedeno výstupní napětí lineárního stabilizátoru. Výstup operačního zesilovače je přiveden přímo na regulační pin lineárního stabilizátoru.

Celý obvod, spolu se zesilovačem vyfiltrovaného signálu, pro řízení lineárního stabilizátoru je výsledkem experimentování než bylo dosaženo požadovaného rozsahu výstupního napětí. Proto

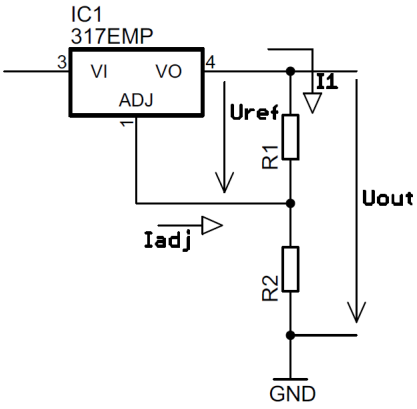
hodnoty součástek R23, R24, R25, R31, R53 a C17 nejsou matematicky podloženy. Na místě kondenzátoru C17 byl původně umístěn rezistor tvořící s rezistorem R31 napěťový dělič, avšak celkové chování obvodu nebylo uspokojivé, proto nám zákazník při komunikaci poradil použít zapojení s kondenzátorem C17, neboť už měl zkušenosti s podobným druhem zapojení.



Obr. 15: Stabilizátor pro napájení ARM motorku

Použitý lineární napěťový stabilizátor je obvod LM317EMP/NOPB firmy Texas Instruments. Důvodem použití lineárních stabilizátorů namísto přímého řízení PWM, nebo použití modernějšího řešení se spínaným zdrojem jako u napájení digitálních obvodů je obava zákazníka, aby se případné zvlnění nebo pulzující napětí neprojevilo nepravidelnosti otáčení *ARM* a *BOTTOM* motorku.

Řízení výstupního napětí na signálu *V_DIM* napájející *ARM* motorek stabilizátorem IC4 je řešeno jiným způsobem, než bývá u lineárních stabilizátorů obvyklé. Při návrhu řízení stabilizátoru se vycházelo z popisu napěťových vztahů vyobrazených na Obr. 16:, kdy napětí mezi pinem VO a pinem ADJ odpovídá referenčnímu napětí stabilizátoru $U_{REF} = 1{,}25$ V. Fakt, že ve schématu je zakreslen proud I1 tekoucí rezistorem R1 a v rovnici 17 zmíněn není vedl k dedukci, že je možné stabilizátor LM317 řídit pouze změnou velikosti napětí na pinu ADJ. Dále, protože bylo třeba zajistit aby se jakákoliv změna na pinu VO projevila změnou na pinu ADJ, byl na místo rezistoru R1 umístěn dříve popsaný obvod součtového zesilovače, který byl nejdříve teoreticky navržen a poté experimentálně vyladěn. Výstup zesilovače zároveň slouží jako odběr proudu I_{ADJ}, který zde na rozdíl od standardního zapojení s rezistory nezpůsobuje chybu výstupního napětí. Typická hodnota I_{ADJ} je u LM317 50 µA až maximálně 100 µA.

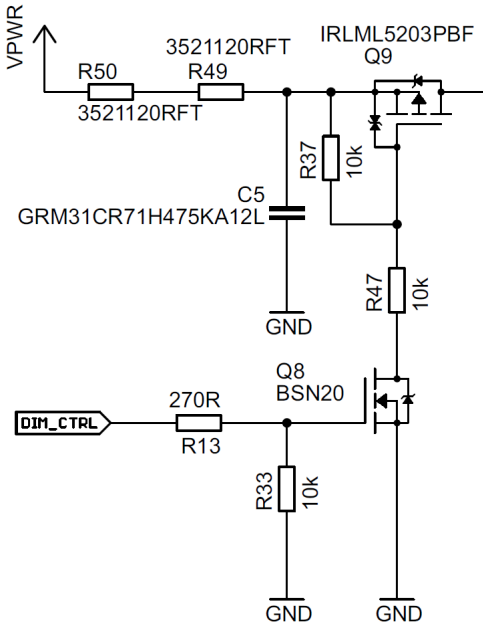


Obr. 16: Vztahy na lineárním stabilizátoru LM317 [10]

$$U_{OUT} = U_{REF} \cdot \left(1 + \frac{R2}{R1}\right) + I_{ADJ} \cdot R2 \tag{17}$$

Důležitým faktorem, který je třeba při práci s lineárními stabilizátory brát v potaz je zahřívání, jehož hlavním zdrojem je úbytek napětí, což je principem činnosti lineárních stabilizátorů. Nejdříve je třeba spočítat tepelný výkon pro nejhorší pracovní případ, což je maximální úbytek napětí $\Delta U = 19$ V a maximální specifikovaný proud $I_{OUT} = 20$ mA. Podle rovnice 18 vychází tepelný výkon $P_D = 0{,}38$ W, vliv proudu I_{ADJ} je zanedbatelný. Dále je třeba určit minimální tepelný odpor mezi přechodem a okolím, při kterém při okolní teplotě $T_a = 25$ °C, nepřesáhne teplota přechodu $T_j = 85$ °C. Podle katalogového list stabilizátoru je tepelný odpor pouzdra $\Theta_{ja} = 140$ °C/W, a podle rovnice 19 je požadovaný tepelný odpor menší než $\Theta_{ja} = 157{,}89$ °C/W. Samotné pouzdro stabilizátoru tedy dokáže udržet teplotu přechodu pod požadovanou úrovní. Zákazník ovšem chtěl mít jistotu správné funkce, proto požadoval přidání chladící plochy. Ve výsledku tedy bylo využito volné místo na DPS pro malou chladící plochu.[10]

$$P_D = \Delta U \cdot I_{OUT} + I_{ADJ} \cdot U_{IN} \tag{18}$$



$$\Theta_{ja} = \frac{T_j - T_a}{P_D} \tag{19}$$

Obr. 17: Spínač napájení bloku ARM DRIVER

Celý blok je možné díky elektronickému spínači vypnout. Napájení je ovládáno signálem *DIM_CTRL* přes rezistor R13, který slouží k utlumení přechodu napětí při změně logické úrovně, stejně jako bylo popsáno u ovládání piezo bzučáku. Z důvodu obtížné sehnatelnosti tranzistoru, jehož

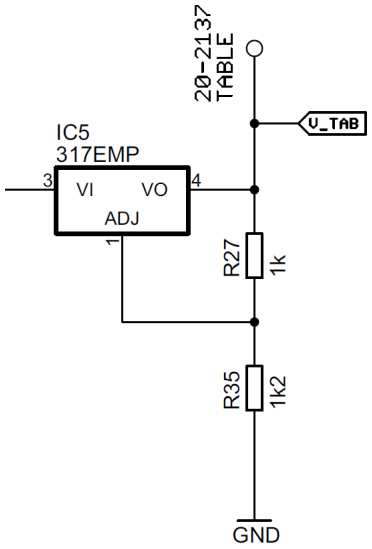
povolené napětí U_{GS} by bylo větší než napájecí napětí U_{PWR}, je třeba snížit rozdíl napětí mezi elektrodami G a S tranzistoru Q9 při sepnutém tranzistoru Q8. Z tohoto důvodu je přidán rezistor R47, který s rezistorem R37 tvoří napěťový dělič, s jehož pomocí je nyní rozdíl napětí mezi elektrodami G a S tranzistoru Q9 při sepnutém tranzistoru Q8 pouze $U_{GS} = U_{PWR} / 2 = 12$ V. Toto napětí je nyní menší než maximální povolená hodnota uvedená v katalogovém listu zvoleného tranzistoru. Rezistory R50 a R49 jsou přidány na žádost zákazníka pro ochranu pro případ zkratu. Při zkratu je dle 20 proud omezen jenom těmito rezistory na $I_K = 98$ mA. Tepelný výkon při zkratu je podle 21 na každém rezistoru $P_D = 1{,}16$ W, což je v limitu použitého typu rezistoru. [4][8]

$$I = \frac{U}{R} \tag{20}$$

$$P_D = R \cdot I^2 \tag{21}$$

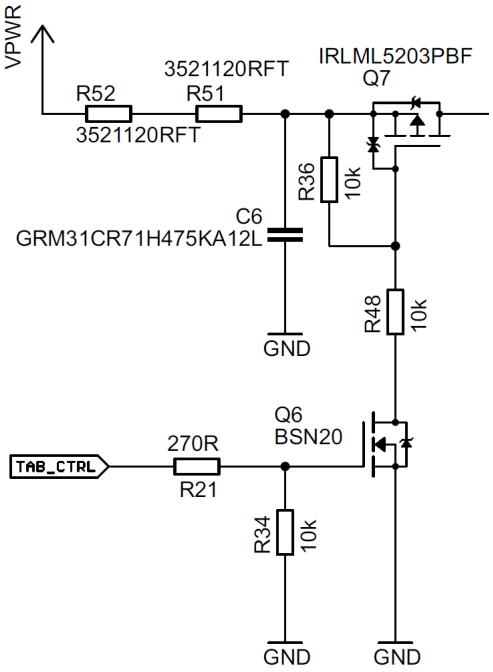
3.2.6 BOTTOM DRIVER

Stabilizátor v *BOTTOM DRIVERu* napájí *BOTTOM* motorek fixním napětím a na rozdíl od *ARM DRIVERu* je napětí tohoto stabilizátoru nastaveno způsobem typickým pro lineární stabilizátory.



Obr. 18: Stabilizátor pro napájení BOTTOM motorku

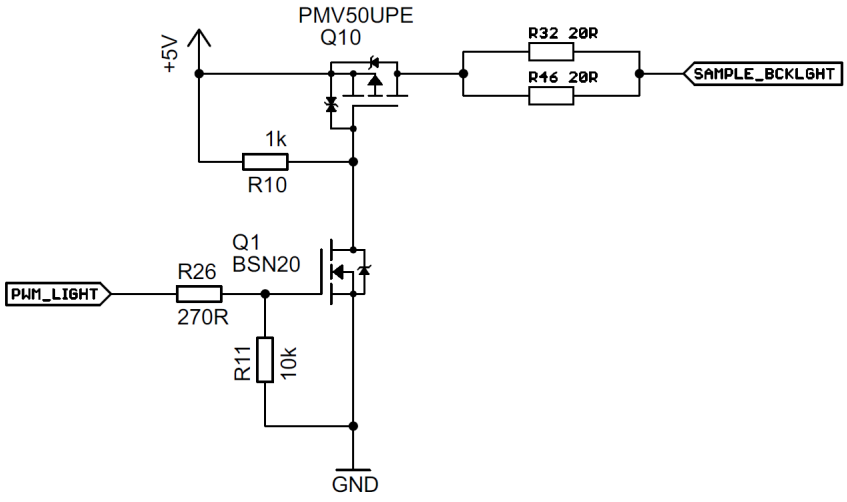
Výstupní napětí přivedené na *V_TAB* je podle obecné rovnice 17 $U_{V\_TAB} = 2{,}75$ V. Použitý *BOTTOM* motorek dosahuje bez zátěže při tomto napětí rychlosti $n_0 = 3052{,}5$ rpm. Po snížení rychlosti pomocí převodů v poměru 809:1 se broušený vzorek otáčí rychlostí n = 3,77 rpm, která je považována za optimální. [3]



Obr. 19: Spínač napájení bloku BOTTOM_DRIVER

Elektronický spínač *BOTTOM DRIVERu* je identický se spínačem použitým v případě ARM DRIVERu, jenom s rozdílem řídicího signálu, v tomto případě *TAB_CTRL*.

3.2.7 LED DRIVER

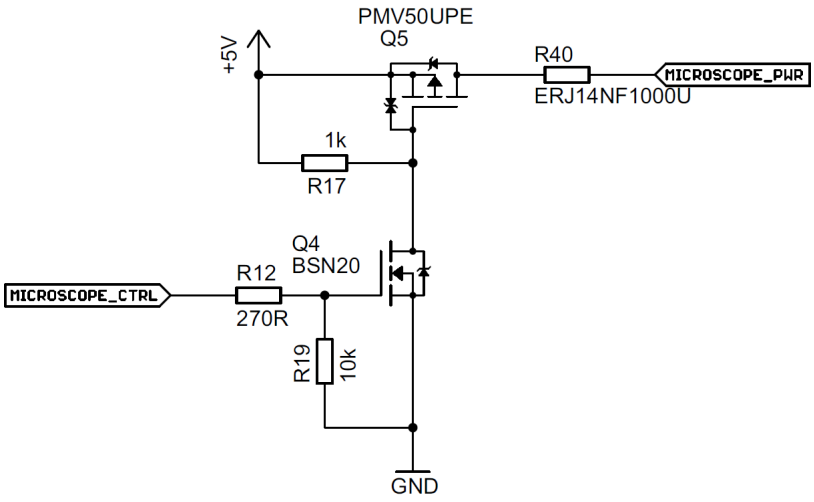


Obr. 20: Obvod pro ovládání podsvícení vzorku

Označení DRIVER je u tohoto bloku pozůstatek po dřívější verzi řízení *BOTTOM_LED*, kdy bylo použito složitější a dražší zapojení, které ovšem nesplnilo očekávání. Proto bylo rozhodnuto, že se bude dále pracovat s jednodušším PWM řízením. Jas je řízen signálem *PWM_LIGHT*, jehož průběh je utlumen stejně jako u ostatních elektronicky ovládaných bloků. Proud jdoucí do LED prochází paralelně spojenými rezistory R32 a R46, které ho omezí na $I_{LED} = 210$ mA, viz 22, a pokračuje

vodičem *SAMPLE_BCKLGHT* na konektor připojující *BOTTOM_LED*. Typická hodnota úbytku napětí na diodě je $U_{LEDf} = 2{,}9$ V. [20]

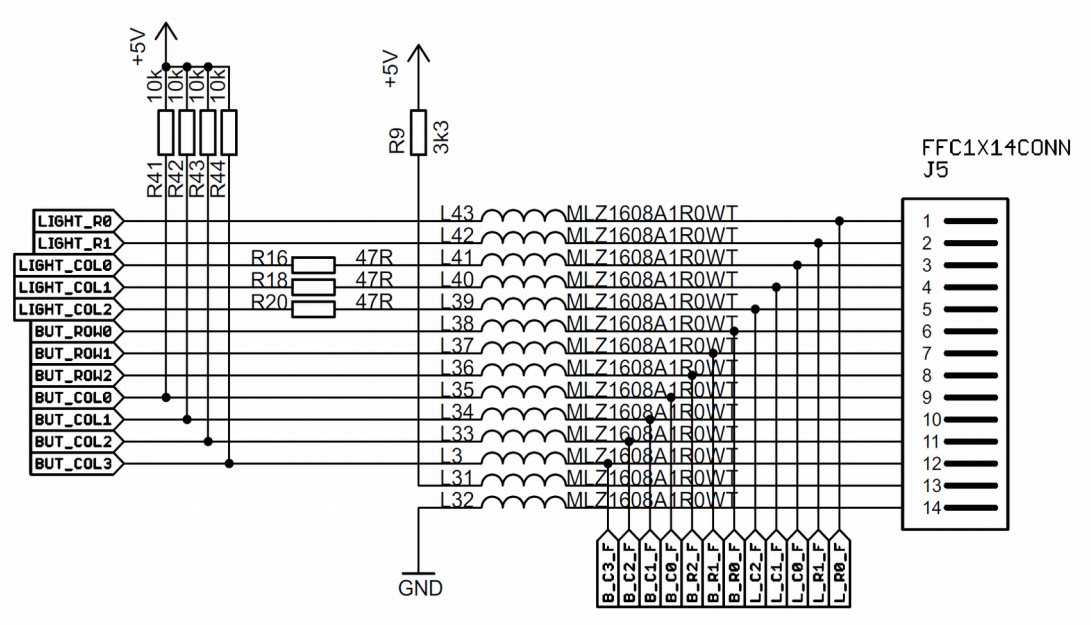
$$I_{LED} = \frac{U_{+5V} - U_{LEDf}}{\left(R32^{-1} + R46^{-1}\right)^{-1}} \tag{22}$$



Obr. 21: Obvod pro řízení jasu osvětlení v mikroskopu

Stejně jako u *BOTTOM_LED* tak i u řízení jasu v mikroskopu je využito PWM řízení spínače signálem *MICROSPCOPE_CTRL*. Dioda v mikroskopu je chráněna omezujícím rezistorem R40 o hodnotě $R = 100\ \Omega$. Maximální proud protékající diodou v mikroskopu tedy odpovídá $I_{SCOPE} = 21$ mA podle 20. Přestože je obvod řízení jasu mikroskopu v blokovém schématu umístěn samostatně, při popisu elektroniky je zařazen pod blok LED DRIVER, neboť sem tématicky a technickým provedením zapadá.

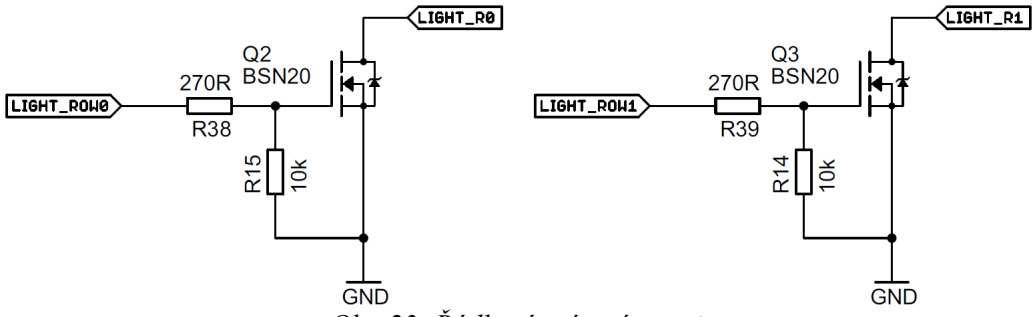
3.2.8 KEYBOARD



Obr. 22: Obvody pro připojení klávesnice

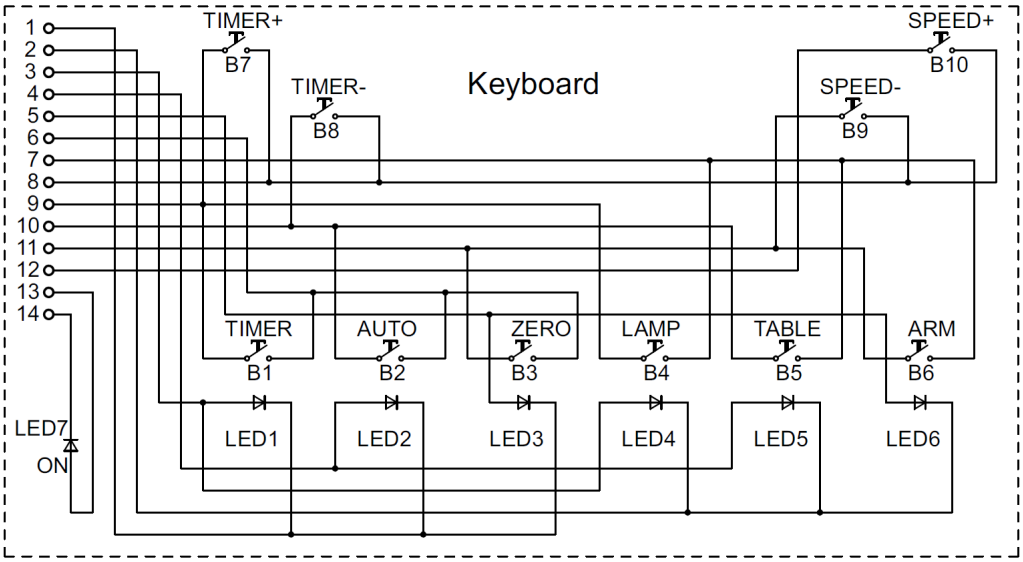
Tlačítka a LED na klávesnici jsou zapojena do matice z důvodu omezeného počtu vstupně výstupních pinů na mikrokontroléru. Čtení stavu kláves je prováděno na signálech *BUT_COL0*, *BUT_COL1*, *BUT_COL2* a *BUT_COL3* uvažovaných jako sloupce matice. Na tyto signály jsou připojeny Pull-Up rezistory pro definování neaktivní logické úrovně. Aktivace skupin tlačítek probíhá po řádcích signály *BUT_ROW0*, *BUT_ROW1* a *BUT_ROW2*. Skupina se aktivuje nastavením LOG0 na odpovídajícím signálu. Na klávesnici jsou signalizační LED dvojího druhu. První druh je zastoupen jedním kusem a slouží pro indikaci stavu napájení – jestli spínaný zdroj pracuje a dodává napětí. Druhým druhem jsou mikrokontrolérem řízené LED zapojené do matice tří párů LED na každém řádku. Aktivace jednotlivých párů LED je prováděna stavem LOG1 na kterémkoliv ze signálů *LIGHT_COL0*, *LIGHT_COL1* a *LIGHT_COL2*. Maximální proud jdoucí do každého páru LED v rámci jedné řádkové skupiny na klávesnici je omezen rezistory R16, R18 a R20 na 38,3 mA. Každou diodou tedy ve výsledku protéká proud If = 19,15 mA podle rovnice 23, což je akorát v limitu maximálního proudu.[5] Zároveň musí být aktivní stavem LOG1 právě jeden z řádkových signálu *LIGHT_ROW0* nebo *LIGHT_ROW1*, které otevřou řádkové spínací tranzistory.

$$I_f = \frac{U_{+5V} - U_f}{R \cdot 2} \tag{23}$$



Obr. 23: Řádkové spínací tranzistory

Signály jdoucí do řádkových spínacích tranzistorů mají jako v předchozích případech seriovými rezistory utlumenou náběžnou hranu pro omezení vyzařování EM záření.



Obr. 24: Schématický nákres zapojení klávesnice

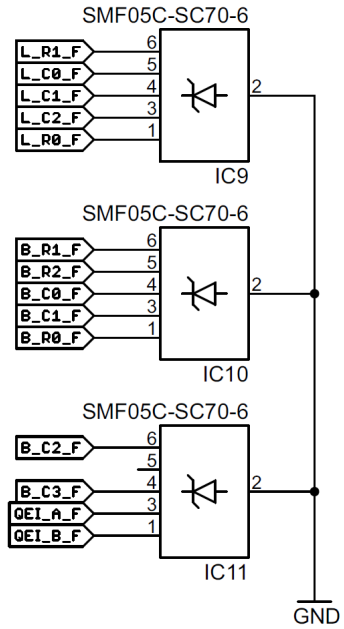
Tab. 1: Zapojení tlačítek klávesnice v matici

	BUT_COL0	*BUT_COL1*	*BUT_COL2*	*BUT_COL3*
BUT_ROW0	TIMER	AUTO	ZERO	
BUT_ROW1	LAMP	TABLE	ARM	
BUT_ROW2	TIMER+	TIMER-	SPEED-	SPEED+

Jak již bylo v textu dříve zmíněno, diody uvedené ve schématickém nákresu jsou ve skutečnosti páry paralelně spojených LED. Důvodem použití dvou led u každého tlačítka byla snaha dosáhnout rovnoměrného podsvícení tlačítek. Protože ovšem každá LED pracuje od poměrně vysokého napětí U_{f_typ} = 3,2 V, bylo třeba diody zapojit paralelně.

Tab. 2: Zapojení párů LED v matici

	LIGHT_COL0	LIGHT_COL1	LIGHT_COL2
LIGHT_ROW0	LED1 (TIMER)	LED2 (AUTO)	LED3 (ZERO)
LIGHT_ROW1	LED4 (LAMP)	LED5 (TABLE)	LED6 (ARM)

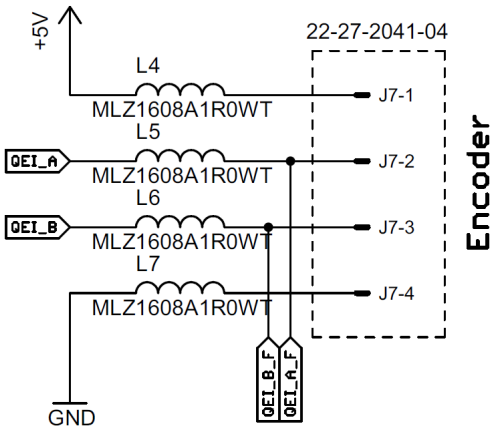


Obr. 25: Ochrana vstupů proti ESD

Na pinech konektoru jsou připojeny diodová pole chránící elektroniku před ESD jevy a napěťovými špičkami. Pro zvýšení účinnosti ochrany jsou do cesty signálům vloženy cívky. Při normální činnosti elektroniky je vliv indukčnosti na procházející signál zanedbatelný, například oproti prodlevám mezi změnou úrovně signálu na pinu mikrokontroléru a projevem této změny na vnitřních logických strukturách mikrokontroléru. Naopak při prudkém nárůstu napětí na pinech konektoru J5, způsobeném třeba elektrostatickým výbojem do klávesnice, může i sebemenší zpomalení průchodu výboje skrz cívky zachránit elektroniku tím, že prodlouží čas, po který bude ESD ochrana cestou s nejmenším odporem, kudy se výboj může nejsnáze dostat na zem. Časová konstanta použité cívky při jejím DC odpor R = 0,15 Ω je $\tau = 6{,}67$ µs podle rovnice 24. Cívky zároveň slouží také jako tlumivky tlumící rušení jdoucí do elektroniky.[14][17]

$$\tau = \frac{L}{R} \tag{24}$$

3.2.9 ENC



Obr. 26: Připojení enkodéru k elektronice

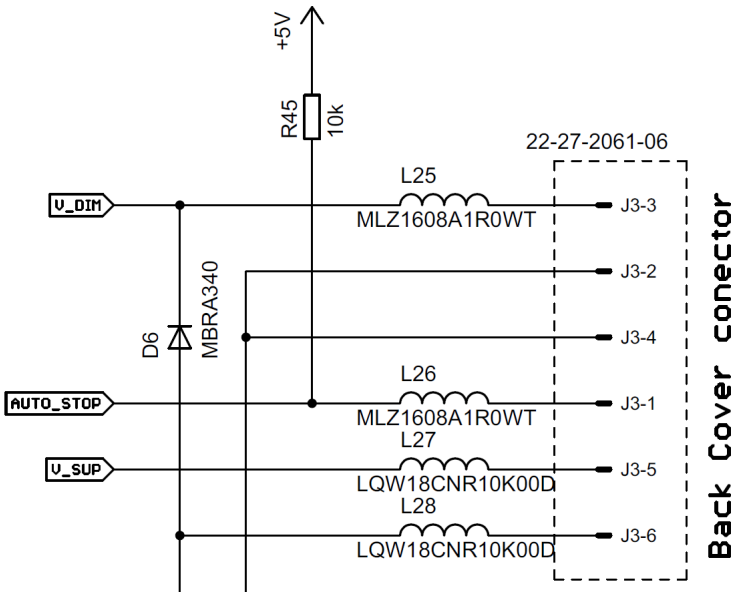
Enkodér je připojený přímo k mikrokontroléru na piny s dedikovaným rozhraním pro kvadraturní enkodéry, jež je v mikrokontroléru součástí modulu pojmenovaného Motion Feedback Module (MFM). Signály jdoucí z enkodéru do mikrokontroléru jsou proti ESD chráněny stejným způsobem jako signály přivedeny na piny konektoru pro klávesnici. Napájení U_{+5V} a GND je chráněno pouze cívkami převážně proti rušení. Výstupem z enkodéru jsou dva obdélníkové signály, jež jsou vůči sobě vázově posunuty o 90°, přičemž směr posunu odpovídá směru otáčení enkodéru. Testy bylo prokázáno, že cívky na signálech *QEI_A* a *QEI_B* nemají výrazný vliv na čtení signálů z enkodéru. Enkodér slouží k nastavení hloubky zabroušení nastavením výšky mechanické zarážky, o kterou se při procesu broušení opře brusné rameno. Jako mechanická zarážka slouží krátká hřídelka umístěná uprostřed enkodéru, jež se díky vysoustruženému závitu vertikálně posouvá otáčením enkodéru. Enkodér generuje 2500 impulsů na jednu celou otáčku což při stoupání závitu P = 0,5 mm podle rovnice 25 dává rozlišení k = 0,2 μm na impuls.

$$k = \frac{P}{n} \tag{25}$$

Vhodným nastavením kvadraturního rozhraní je možné zvýšit rozlišení až na k = 50 nm, kdy se vyhodnocují náběžné i sestupné hrany obou signálů *QEI_A* a *QEI_B*. Jako ideální je zvoleno rozlišení k = 0,1 μm.

3.2.10 Konektorová výbava

Z konektorů nacházejících se v elektronice byly zatím popsány konektor rozhraní ICSP™, konektor pro připojení klávesnice a konektor enkodéru. Základním požadavkem na konektory je jejich nezaměnitelnost, což urychluje a zjednodušuje montáž elektroniky do zařízení. Každý pin na každém konektoru má jako minimální ochranu připojenou cívku pro odrušení.



Obr. 27: Konektor zadního krytu

Zadní kryt zařízení je připojen na 6-ti pinový konektor. Napájení je na zadním krytu připojené přes konektor typu Jack, signály *AUTO_STOP* a *V_DIM* jsou vyvedeny na banana konektor. Vzhledem k tomu, signálem *V_DIM* je napájený motorek, jehož vinutí tvoří značnou indukční zátěž, je třeba připojit k tomuto signálu ochranou diodu v závěrném zapojení. Na signál *AUTO_STOP* je připojen pull-up rezistor, protože signalizaci dokončení zabrušování na požadovanou tloušťku, je řešena vodivým spojením s uzemněnou základnou zařízení.

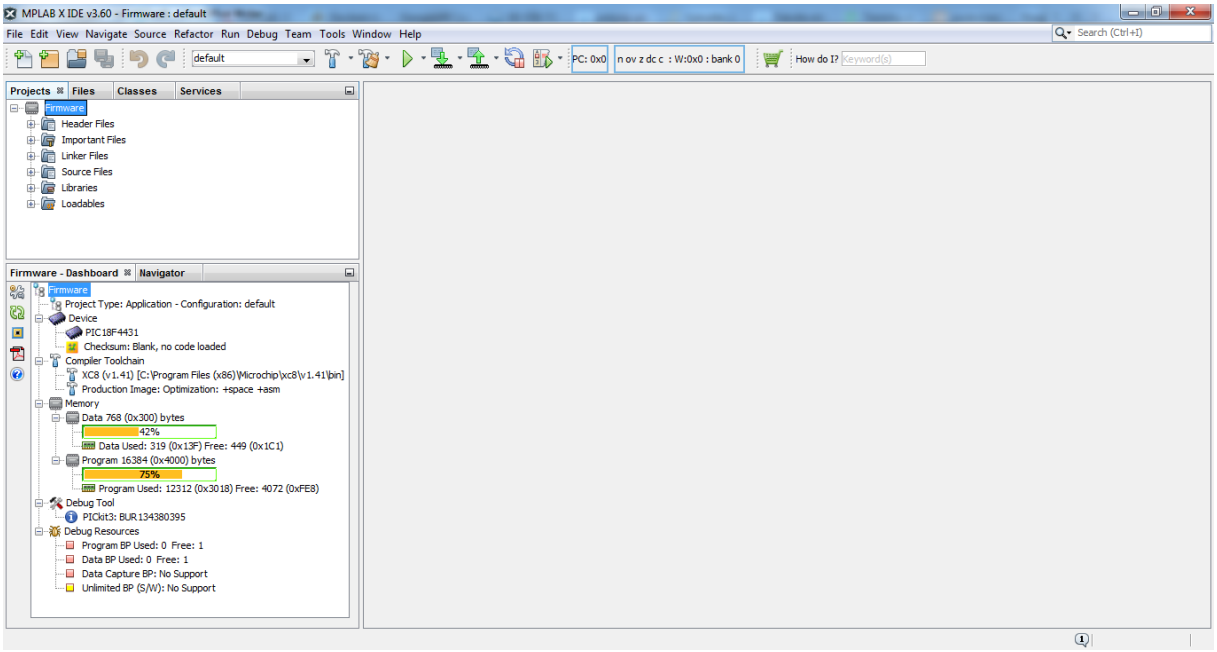
Zbývajícími konektory jsou Cover conector pro napájení osvětlení mikroskopu a Base conector napájející *BOTTOM* motorek a *BOTTOM_LED*. Stejně jako u signálu *V_DIM* je i k signálu *V_TAB*, napájející *BOTTOM* motorek, připojena ochranná dioda.

4 Řídící software elektroniky

V této kapitole je popsáno vývojové prostředí používané pro práci s mikrokontroléry firmy Microchip, dále je popsáno nastavení projektu a základní konfigurační úkony které je nezbytné udělat před zahájením vlastního vývoje. Dále následuje popis řídícího softwaru a jeho jednotlivých částí.

Řídící software je napsán v jazyce C v vývojovém prostředí MPLAB® X IDE, dodávaném firmou Microchip. Je třeba varovat, že vývojové prostředí v sobě neintegruje kompilátory a hlavičkové soubory obsahující definice konstant specifických pro různé typy mikrokontrolérů. Firma Microchip má tyto nezbytné části rozdělené do tří samostatných balíčků podle cílové architektury. Pro 8-mi bitová MCU je třeba na pracovní počítač nainstalovat balíček MPLAB® XC8 Compiler. Pro případ práce na mikrokontrolérech dalších architektur, jež má firma Microchip v nabídce, jsou k dispozici také balíčky MPLAB® XC16 Compiler, MPLAB® XC32 Compiler pro 16 a 32 bitové mikrokontroléry. Nahrávání a ladění zkompilovaného kódu je řešeno pomocí USB nástroje PICkit™ 3 připojeného k mikrokontroléru přes ICSP™ rozhraní. [11][19]

4.1 Prostředí MPLAB® X IDE[19]



Obr. 28: Prostředí MPLAB® X IDE

Prostředí je ve výchozím nastavení rozdělené na tři části. Po levé straně je umístěné navigační okno projektu a pod ním je okno s informacemi o projektu. V pravé větší části se zobrazí po spuštění úvodní informační stránka a také se zde zobrazují soubory otevírané přes navigátor, výstupní informace při kompilaci kódu a různá dodatečná data a informace při ladění. Nástrojů a informačních oken je ve vývojovém prostředí více a při jejich otevírání se zobrazují na předdefinovaných pozicích,

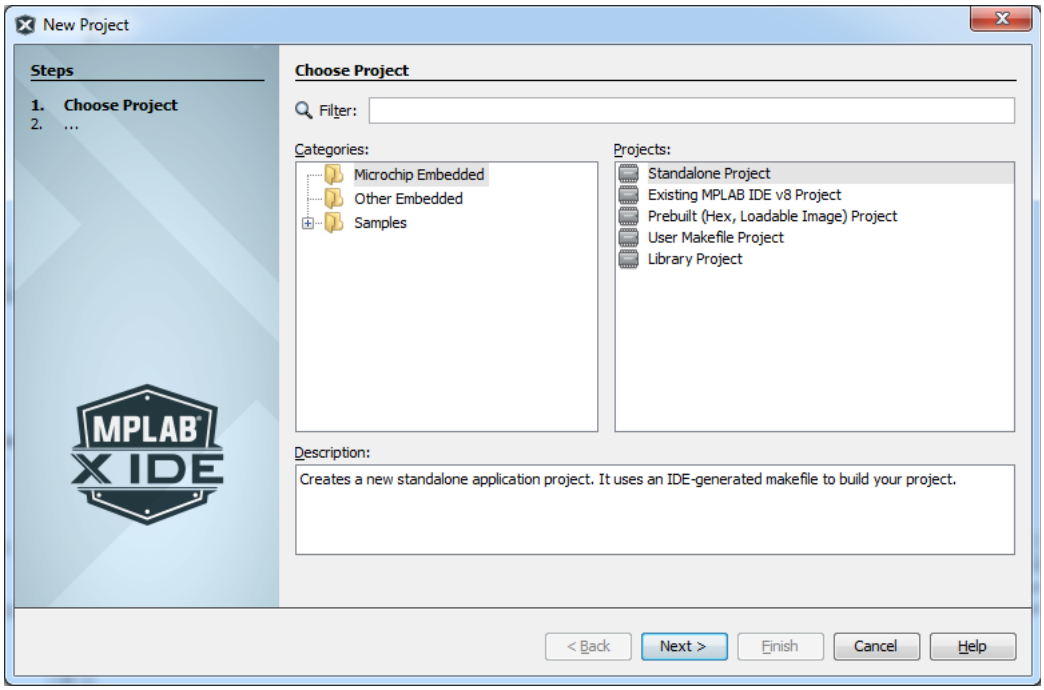
buďto v nových oknech, nebo jako další záložky v těch existujících. Ovšem samozřejmostí je možnost si kompozici celého prostředí libovolně přizpůsobit potřebám vývojáře.

Navigační okno obsahuje záložky Projects, Files, Classes a Services. V záložce Projects jsou strukturovaně zobrazeny všechny otevřené projekty a jejich obsah. V záložce Files jsou zobrazeny soubory a složky otevřených projektů tak, jak jsou uloženy v souborovém systému. Záložka Classes obsahuje seznam definicí, proměnných, struktur a funkcí seskupených podle typu, seřazených abecedně a přiřazených otevřeným projektům. Záložka Services obsahuje právě běžící služby.

V projektovém informačním oknu jsou informace o typu projektu a jaká konfigurace je použita, typ použitého mikrokontroléru, nastavení a umístění kompilátoru, informace o zaplnění datové a programové paměti, informace o použitém programovacím/ladícím nástroji a informace o parametrech ladění, počet použitých a volných bodů zastavení programu.

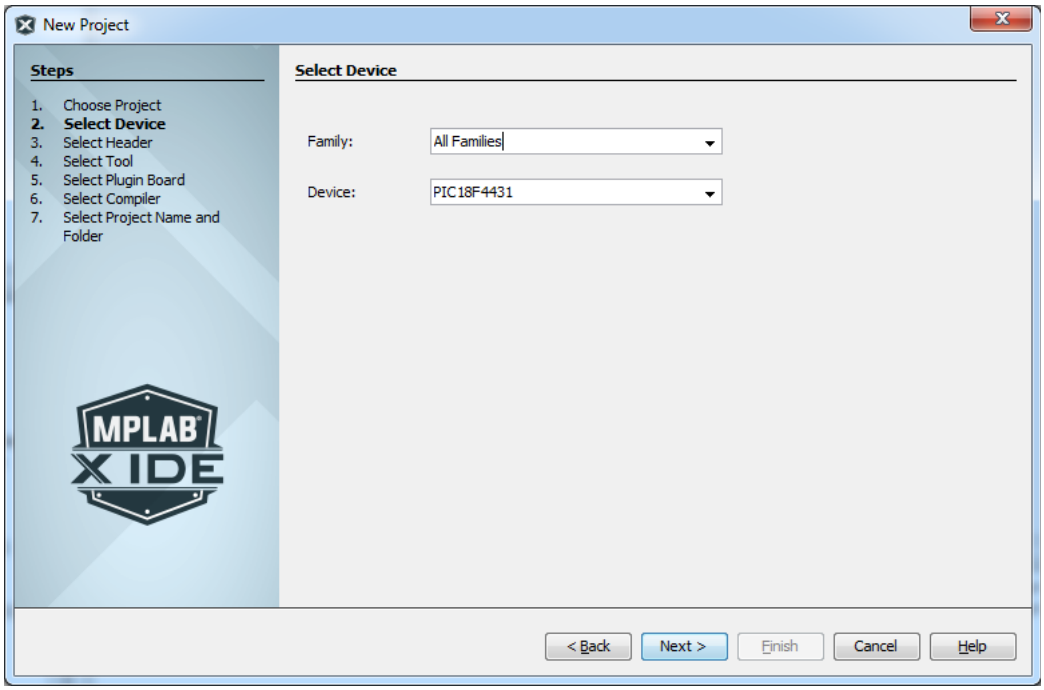
4.1.1 Vytvoření projektu v MPLAB® X IDE

Dříve než je možné začít psát kód řídícího softwaru, je třeba vytvořit a správně nastavit projekt. Prvním krokem při vytváření projektu je volba kategorie projektu.



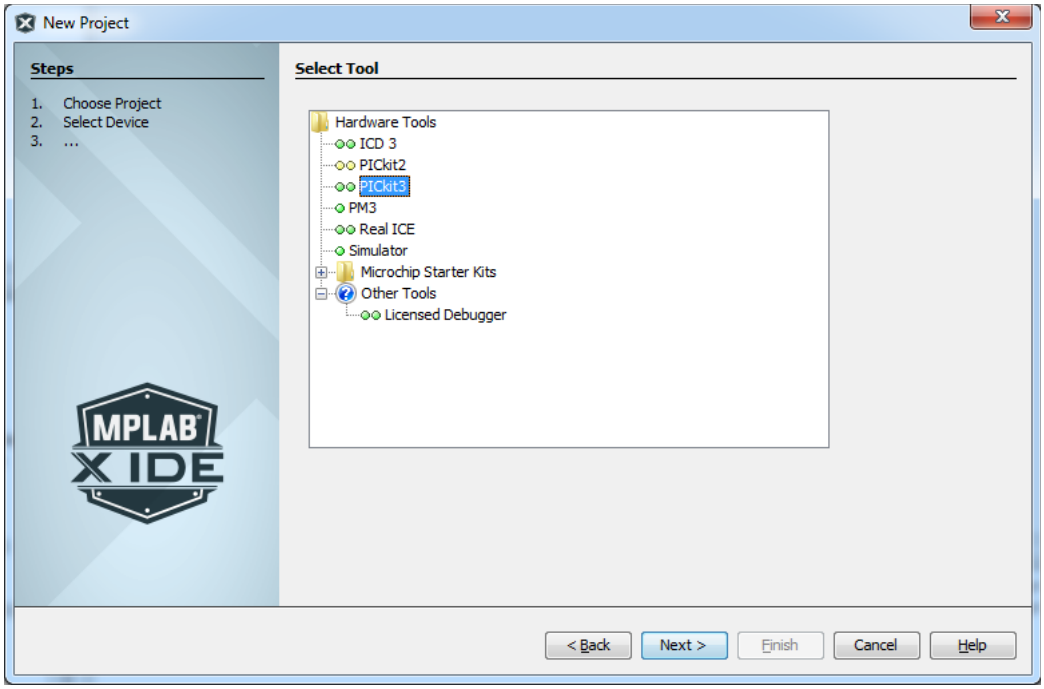
Obr. 29: Volba kategorie projektu

Poté následuje výběr mikrokontroléru. Pro zjednodušení nalezení konkrétního zařízení je možné nejdříve zvolit v jaké rodině se nachází a poté se vybere použitý mikrokontrolér PIC18F4431.



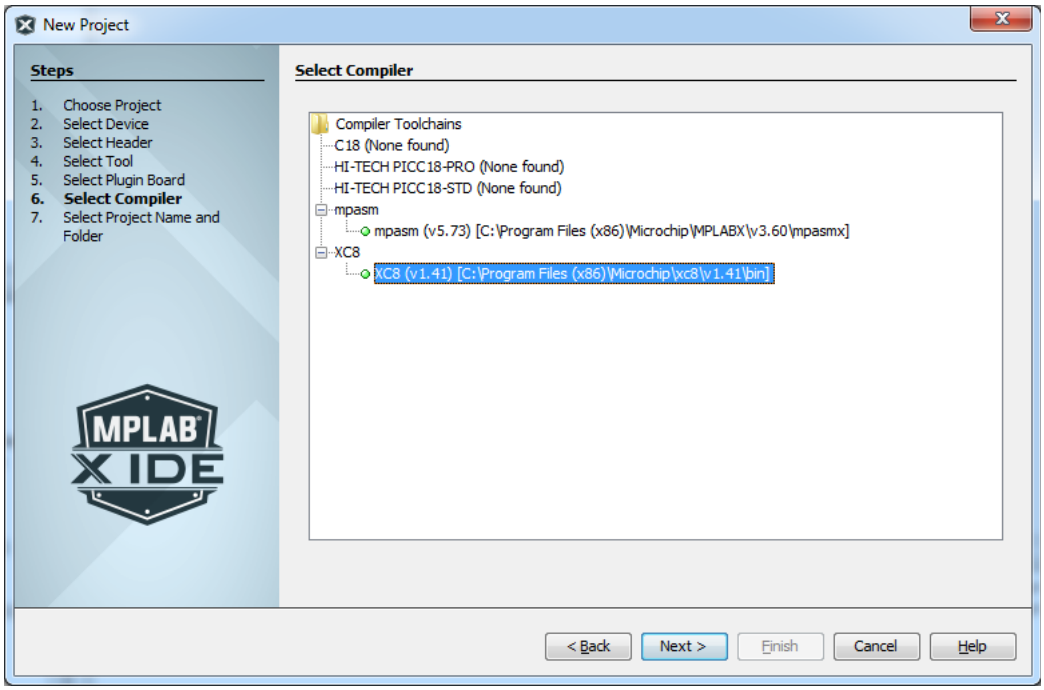
Obr. 30: Volba cílového zařízení

Jakmile je vybráno zařízení, je třeba zvolit nástroj, přes který bude probíhat samotný vývoj a ladění kódu. Nástrojem použitým při vývoji a ladění je PICkit™ 3.



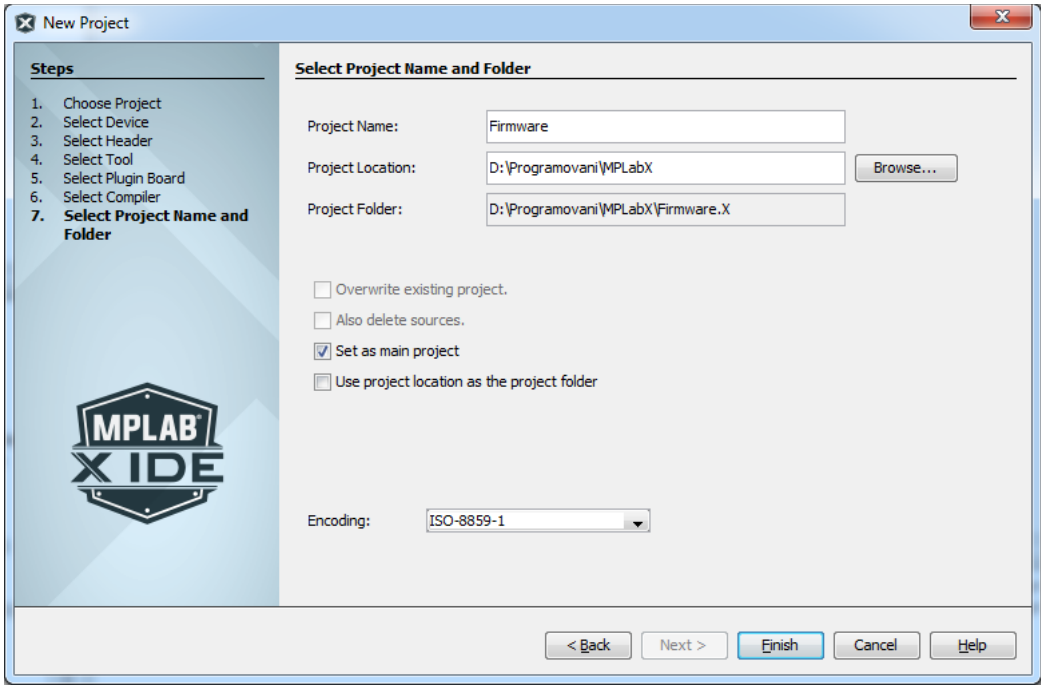
Obr. 31: Volba nástroje pro vývoj

Dalším krokem je volba kompilátoru. Zde je volba jednoduchá. Protože použitý mikrokontrolér je 8-mi bitový, zvolí se XC8.



Obr. 32: Volba kompilátoru

Jako poslední krok je třeba zvolit jméno a umístění projektu. Také je možnost vytvořený projekt nastavit jako hlavní, pak se bude otevírat vždy při spuštění vývojového prostředí.



Obr. 33: Volba jména a umístění

Po vytvoření je projekt prázdný a je možné do něj začít vkládat hlavičkové soubory s definicemi a soubory se zdrojovými kódy.

4.1.2 Konfigurace mikrokontroléru

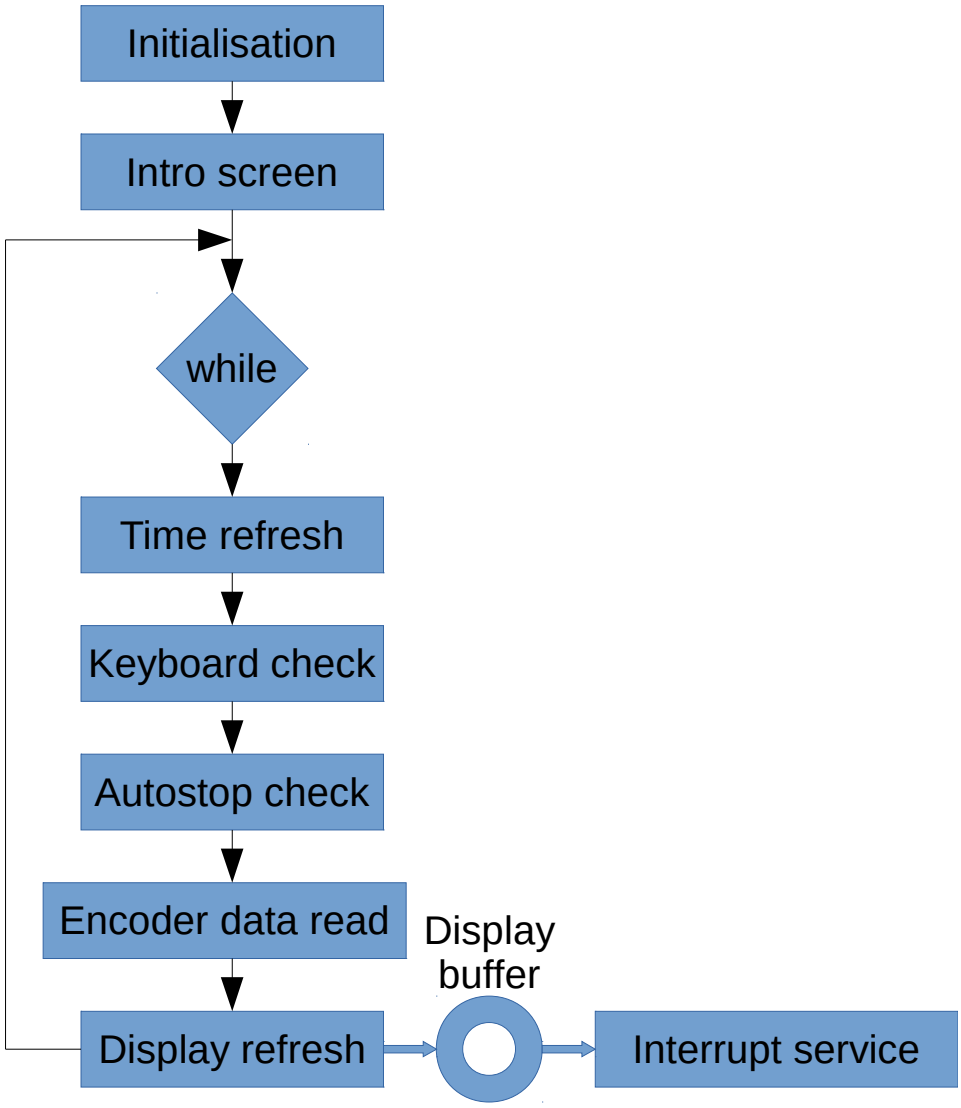
Pro správnou činnost je třeba mikrokontrolér nakonfigurovat pomocí konfiguračních bitů. Hodnoty konfiguračních bitů je možné napsat ručně do souboru zdrojového kódu, což ale skýtá riziko chyby. Mnohem lepší je využít nástroje PIC Memory Views, umístěného v nabídce Window, a v podnabídce zvolit Configuration Bits. Zde se nastavují konfigurační bity pomocí rolovacích nabídek s tím, že návrhář má k dispozici popis kategorie každého nastavitelného pole a pro každou možnost je k dispozici její krátký popis. Jakmile je návrhář spokojen s nastavením, stiskne tlačítko *Generate Source Code to Output* a tím se vygeneruje do nově otevřené záložky kód, který je pak třeba zkopírovat do zdrojového souboru v projektu. Nakopírovaný kód je zpracováván preprocesorem:

```
// CONFIG2L
#pragma config PWRTEN = OFF     // Power-up Timer Enable bit (PWRT disabled)
#pragma config BOREN = ON       // Brown-out Reset Enable bits (Brown-out Reset
enabled)
// BORV = No Setting
```

Při nahrávání nové verze softwaru se pak data zpracované preprocesorem předají zvolenému nástroji a ten je nahraje na správné místo v paměti mikrokontroléru.

4.2 Struktura řídícího softwaru

Řídící software sestává z několika málo logicky odlišitelných bloků a jeho vstupním bodem je funkce main. Po zapnutí zařízení a dosažení stabilní úrovně napájecího napětí U_{+5V} se provede inicializace softwaru a periferií mikrokontroléru. Po inicializaci se zobrazí úvodní obrazovka po dobu minimálně 1 s až do okamžiku uživatelovy interakce. Po uběhnutí 1 s software vstoupí do hlavní programové smyčky. Na začátku smyčky se nejprve provede kontrola uběhlého času a obsluha časově závislých procesů. Dále se provede vyčtení stavu tlačítek na klávesnici a obsluha jednotlivých tlačítek. Následuje čtení hodnoty otočení enkodéru a kontrola signálu *AUTO_STOP*. Nakonec se provede překreslení displeje při přechodu na jinou obrazovku. Při překreslování obrazovek se příkazy a data pro displej nahrávají do kruhového bufferu, ze kterého se čtení provádí v obsluze přerušení. V kódu jsou místy zakomentované části obsahující metody a řešení, které mohou být použity později. Přestože je zařízení již ve výrobě, vývoj ještě nebyl oficiálně ukončen, takže struktura kódu není úplně čistá jak by mohla a měla být.



Obr. 34: Vývojový diagram řídícího softwaru

4.2.1 Inicializace

V inicializační části se provádí všechny nezbytné úkony a nastavení pro správný chod programu. Jako první se volá funkce InitHW, jež obsahuje volání inicializačních funkcí pro jednotlivé periferie:

```
void InitHW(void)
{
	InitTime();		//inicializace citace 1 ms
	DisplayInit();		//inicializace displeje
	PWMInit();		//nastaveni PWM modulu
	ButtonInit();		//inicializace tlacitkoveho vstupu
	InitIO();		//inicializace ostatnich IO
//	MemoryInit();
	infoLight = 0;
	lightState = LIGHT_ROW0;
}
```

Jako první z periferií se musí správně nastavit čítač sloužící jako zdroj hodin o rozlišení 1 ms, který se používá pro časování časově závislých procesů v softwaru.

Po inicializaci časovače je na řadě displej. Zde se provádí nastavení pinů připojených na signály *DISP_E* a *DISP_RS*. Dále se nakonfiguruje SPI modul s přenosovou rychlostí f_{SCK} = 62,5 kHz. Posledním krokem v inicializaci displeje je příprava kruhového bufferu a nahrání inicializační sekvence příkazů pro modul LCD displeje do kruhového bufferu. Od tohoto bodu je programové rozhraní displeje plně funkční.

Funkce PWMInit nastaví PWM modul řídící rychlost brusného motorku *ARM* signálem *PWM_MOTOR* a intenzitu osvětlení vzorku mikroskopem signálem *MICROSCOPE_CTRL*. Také se zde nastavuje modul čítače v konfiguraci PWM řídící intenzitu podsvětlení brouseného vzorku signálem *PWM_LIGHT*.

Ve funkci ButtonInit se nastaví odpovídající piny jako vstupní pro signály *BUT_COL0*, *BUT_COL1*, *BUT_COL2* a *BUT_COL3* nebo výstupní pro signály *BUT_ROW0*, *BUT_ROW1* a *BUT_ROW2*.

Poslední funkce InitIO nastavuje zbylé piny a periferie. Jako první se vypínají analogové vstupy na všech pinech, protože ve výchozím stavu jsou zapnuté a měření analogových hodnot není potřeba. Dále se nastavuje rozhraní pro kvadraturní enkodér, nastavují se výchozí hodnoty čítacích registrů, maximální hodnota načtených impulzů a povoluje se přerušení při přetečení maximálního počtu impulzů. Piny připojené na signály *TAB_CTRL*, *DIM_CTRL* pro řízení napájení motorků se nastavují jako výstupní. Další v řadě je nastavení výstupních pinů připojených na signály *LIGHT_COL0*, *LIGHT_COL1*, *LIGHT_COL2, LIGHT_ROW0* a *LIGHT_ROW1* řídících LED signalizaci na klávesnici. Nakonec se nastaví vstupní pin pro připojení signálu *AUTO_STOP* a výstupní pin pro připojení signálu *BEEP*.

Po poslední funkci se do výchozích hodnot nastavují proměnné určující rozsvícené LED a řídící ovládání LED matice. Tímto končí funkce InitHW, následuje nastavení několika lokálních proměnných a vyčtení uložených hodnot z vnitřní EEPROM:

```
byte speed = LoadSpeedValue();
byte scopeLight, bottomLight;
scopeLight = LoadMicroscopeLightLevel();
bottomLight = LoadBottomLightLevel();
if (scopeLight > 100)
       scopeLight = DEFAULT_LGHT_LVL;
if (bottomLight > 100)
       bottomLight = DEFAULT_LGHT_LVL;
if (speed > 99)
       speed = DEFAULT_SPEED;
```

Po načtení uložených hodnot poslední nastavené rychlosti, úrovně osvícení vzorku mikroskopem a úrovně podsvícení se načtené data otestují zda se nacházejí v limitních hodnotách. Pokud jsou data mimo limit, tak se nastaví na výchozí hodnotu. Tato kontrola je nezbytná hlavně v případě prvního spuštění elektroniky, kdy mají paměťové buňky v EEPROM ve výchozím stavu

nastavené všechny bity na LOG1. Každá z načítacích funkci volá s různými parametry určujícími adresu uložené hodnoty jednu univerzální funkci ReadByte:

```
byte ReadByte(byte address)
{
        EEADR = address;
        EECON1 &= (~(BIT7 | BIT6));
        EECON1 |= BIT0;
        while ((EECON1 & BIT0) != 0);
        return EEDATA;
}
```

Čtení z EEPROM je řízené hardwarově, kdy se zadá adresa dat, nastaví se čtení z EEPROM a zahájí se čtecí sekvence. Poté se čeká než jsou data připravena a nakonec se vrátí obsah datového registru.

Následuje nastavení časovacích lokálních proměnných a načtení poslední délky běhu brousícího cyklu z EEPROM, čímž je inicializace softwaru dokončena.

4.2.2 Úvodní obrazovka

Zobrazení úvodní obrazovky probíhá ve dvou fázích. Nejprve se na 500 ms zobrazí první řádek úvodní obrazovky:

```
MicroscopeLight(scopeLight);
BottomLight(bottomLight);
DispControl(CLEAR_DIS,0);
DispControl(SET_POS, 0x05);
WriteStr("GATAN");
DispControl(SET_POS,0x50);
ButtonsLight(LIGHT_OVERRIDE_ON);
do
{
        time = GetTicks();        //nactení ubehlých ms
        dT = time – timePass;     //zjistení doby trvání predchozí programové smycky
        timePass = time;          //ulození aktuálne ubehlých ms
        pulse += dT;
}       while (pulse <= (INTRO_TIME >> 1));
```

Nastaví se načtené světelné úrovně mikroskopu a podsvícení. Vymaže se displej, vypíše první řádek úvodní obrazovky a schová se kurzor. Rozsvítí se všechny LED na klávesnici a pak se čeká na uběhnutí první poloviny času úvodní obrazovky, aby se zobrazila úvodní obrazovka celá:

```
ShowScreen(LCD_State);
DispControl(SET_POS,0x50);
do
{
        time = GetTicks();  //nactení ubehlých ms
        dT = time - timePass;   //zjistení doby trvání predchozí programové smycky
        timePass = time;    //ulození aktuálne ubehlých ms
        pulse += dT;
}       while (pulse <= (INTRO_TIME >> 1));

LampLight(OFF);
ButtonsLight(LIGHT_OVERRIDE_STOP);
```

Zavolá se funkce starající se o zobrazování předdefinovaných obrazovek a schová se kurzor. Po uběhnutí druhé poloviny času úvodní obrazovky se vypne *BOTTOM_LED* a indikační LED na klávesnici a software vstoupí do hlavní programové smyčky.

4.2.3 Aktualizace časových proměnných a řízení časovaných procesů

Úplně prvním krokem na začátku smyčky je změření uběhlého času a aktualizace proměnných řídících časované procesy:

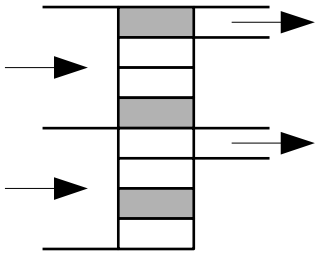
```
time = GetTicks();  //nactení ubehlých ms
dT = time - timePass;   //zjistení doby trvání predchozí programové smycky
if (dT > 0)
{
        ButtonRefresh();    //obnovení stavu tlacítek
        buttonInput = ButtonState();    //nactení stavu vstupu
}
timePass = time;    //ulození aktuálne ubehlých ms

//aktualizace casovych prommenych
lightTime += (byte) dT;
holding += (byte)dT;
displayRefresh += (word) dT;
secCounter += (word) dT;
```

Protože funkce GetTicks pracuje s proměnnou, která má větší bitovou délku než datová sběrnice mikrokontroléru, tak je třeba dočasně zakázat přerušení časovacího čítače, poté zkopírovat obsah proměnné, povolit přerušení a teprve pak je možné opustit funkci a vrátit hodnotu uběhlých ms:

```
dword GetTicks(void)
{
    DIS_INT;    //docasne zakazani preruseni
    timeCntRet = timeCount; //ulozeni posledního stavu nactených ms
    EN_INT;     //povoleni preruseni
    return timeCntRet;
}
```

Důležitou funkcí z hlediska spolehlivosti je funkce ButtonRefresh. Problémem u tlačítek jsou totiž zákmity při jejich stisknutí, jež by byly vyhodnocovány jako samostatná stisknutí. Proto je třeba provést filtraci signálů jednotlivých tlačítek. Dalším problémem, zjištěným experimentálně, tentokrát na straně mikrokontroléru, je fakt, že zápis změny logické úrovně na výstupní pin se při čtení projeví až po určité nenulové prodlevě, protože se čte hodnota na pinu a ne na odpovídající pozici v registru. Procesor mikrokontroléru provádí nejdříve čtení obsahu registru a až poté jeho modifikaci, i když se provádí jenom zápis bez čtení. Tyhle dvě vlastnosti dohromady můžou mít při rychlé sekvenci modifikací hodnoty registru portu za následek chybné nastavení pinů, protože některé dříve provedené změny se na pinu nemusí stihnout projevit. Problém chybovosti výstupních pinů je spolu s nežádoucími zákmity řešen právě v této jediné funkci. Pro filtraci zákmitů je k dispozici jedna struktura obsahující bitová pole tvořená proměnnými typu byte. K této struktuře je přistupováno dvojím způsobem znázorněným na Obr. 35:. Pokud je tlačítko na klávesnici stisknuto, 4-bitová hodnota v bitovém poli se při každém zavolání funkce inkrementuje, v opačném případě dekrementuje, do limitních hodnot 0x0 nebo 0xF. Samotná filtrace je pak řešena vyčítáním nejvyššího

bitu z každého pole. Problém se zpožděním je pak řešen prokládáním čtení a zápisů na portech s vyhodnocováním stisknutí tlačítek a následnou inkrementací a dekrementací bitových polí.



Obr. 35: Filtrace tlačítek

Po vykonání funkce ButtonRefresh se vyčtou přefiltrované hodnoty stisknutí tlačítek a software dále pokračuje v aktualizaci časových proměnných a kontrole běhu časovaných procesů.

Mezi časované procesy patří stavový automat řídící blikání tlačítek signalizující dokončení brusného cyklu a vteřinový čítač řídící a kontrolující dobu běhu brusného cyklu pokud je software v časovém režimu. Pokud brusný cyklus běží na čas, tak každou vteřinu dochází ke změně stavu LED pod tlačítkem *TIMER* pro signalizaci časovaného režimu. V okamžiku, kdy čas běhu vyprší, tak se aktivuje blikání klávesnice a po definovanou dobu zní bzučák.

4.2.4 Obsluha klávesnice

Při obsluze klávesnice je nejdříve třeba rozlišit stisknutí a uvolnění tlačítka, k čemuž slouží jednoduchý algoritmus testující aktuální a minulý stav tlačítka reprezentovaného jedním bitem v proměnné:

```
if (((buttonInput & BUTTON_X) != 0) && ((lastButtonInput & BUTTON_X) == 0))
{
        lastButtonInput |= BUTTON_X;
}
else if (((buttonInput & BUTTON_X) == 0) && ((lastButtonInput & BUTTON_X) != 0))
{
        lastButtonInput &= (~BUTTON_X);
}
```

V první podmínce se zjistí, jestli bylo tlačítko právě stisknuto. Pokud ano, zaznamená se informace o stisku a provede se případný kód. Pokud bylo tlačítko zrovna uvolněno, informace o stisknutí se vymaže a pokud je pro tuto událost napsaný nějaký kód, tak se vykoná.

Jako první jsou obslouženy inkrementační a dekrementační tlačítka jež slouží buď k nastavení rychlosti *ARM* motorku a časovače pro brousící cyklus nebo k změně jasu *BOTTOM_LED* a osvícení mikroskopu. Vzhledem k tomu, že počet kroků pro nastavení ať už jasu, času nebo rychlosti je poměrně vysoký, je třeba při držení tlačítka pozvolna zvětšovat krok a z toho důvodu také detekovat trvalé držení tlačítka. Proto je při stisku nastavena stavová proměnná indikující držení daného tlačítka. Po nastavení stavové proměnné a krátké prodlevě umožňující měnit hodnoty i jenom pomocí opakovaných stisknutí, se začne odpovídající veličina měnit s periodou 250 ms. Po každých deseti změnách dojde ke zvětšení kroku a v případě, že by se proměnná po změně měla dostat mimo svůj povolený rozsah, dojde ke zmenšení kroku. V případě nastavení rychlosti jakmile je tlačítko uvolněno,

tak se nastavená rychlost uloží do EEPROM zavoláním funkce pro zápis s odpovídají adresou jako parametrem:

```
void WriteByte(byte data, byte address)
{
        byte tmp;
        EECON1 |= BIT2;         //povoleni zapisu
        EEADR = address;        //nastaveni adresy pro zapis
        EEDATA = data;          //pripraveni zapisovanych dat
        EECON1 &= (~(BIT7 | BIT6));
        DIS_INT;
        EECON2 = 0x55;          //bezpecnostni
        EECON2 = 0xAA;          //sekvence
        EECON1 |= BIT1;         //zahajeni zapisovaciho procesu
        EN_INT;
        while ((EECON1 & BIT1) != 0);
        EECON1 &= (~BIT2);
//      if (ReadByte(address) == data)
                return;
/*      else
                asm("NOP");*/
}
```

Zápis do EEPROM je rozsáhlejší než čtení. Nejdříve je povolen zápis do paměti, poté se nastaví adresa a připraví zapisovaná data a nastaví se jako cíl operace EEPROM. Poté se na doporučení katalogového listu vypne přerušení, zadá se bezpečnostní sekvence a zahájí zápis. Od tohoto bodu je možné přerušení opět povolit a po dokončení zápisu se zruší povolení zápisu do paměti. Povolení zápisu, bezpečnostní sekvence a kontrola zahájení zapisovacího procesu dohromady tvoří ochranu proti náhodnému zápisu do paměti při chybě programu nebo problémům s napájením.

Po obsluze nastavovacích tlačítek jsou na řadě akční tlačítka mající vliv na chod zařízení. Akční tlačítka jsou podsvícená, přičemž způsob a funkce podsvícení závisí na konkrétním tlačítku.

Tlačítko *ZERO se* při stisknutí rozsvítí a při uvolnění zhasne. Podle toho v jaké obrazovce se software nachází, tak se nulují odpovídající proměnné. V obrazovce *SCREEN_DEPTH* se nulují a resetují proměnné vztažené k enkodéru. V obrazovce *SCREEN_TIMER* se nuluje nastavený čas.

Tlačítko *ARM* řídí běh *ARM* motorku, svítí když by motorek měl běžet a je zhasnuté když by měl stát.

Tlačítko *LAMP* řídí osvětlení mikroskopem a podsvětlení pomocí LED. Když svítí tlačítko, je aktivní LED a mikroskop zhasnutý. Když tlačítko nesvítí, LED je zhasnutá a svítí mikroskop. Po rychlém dvojitém stisknutí se aktivuje obrazovka pro nastavení jasu obou světelných zdrojů. Zároveň také oba svítí aby bylo možné sledovat změnu jasu jak LED, tak i mikroskopu. Po opětovném stisknutí tlačítka v obrazovce nastavení jasu se nastavené hodnoty uloží do EEPROM a software se přesune do úvodní obrazovky.

Tlačítkem *AUTO* software přejde do režimu automatického zastavení a spustí oba motorky, jež běží do té doby, než dojde k aktivaci na signálu *AUTO_STOP*. Software může pracovat v režimu automatického zastavení i časovém režimu zároveň. Tlačítko je podsvícené když je režimu automatického zastavení aktivní.

Tlačítkem *TIMER* se spustí oba motorky, zapne se odpočítávání nastaveného času a software přejde do časového režimu, při němž LED *TIMERu* mění svůj stav jednou za vteřinu. Zároveň se stisknutím tlačítka *TIMER* se nastavený čas uloží do EEPROM.

Tlačítko *TABLE* ovládá *BOTTOM* motorek, svítí když by motorek měl běžet a je zhasnuté když by měl stát.

4.2.5 Kontrola signálu AUTO_STOP

V případě, že je software v režimu automatického zastavení, tak software pravidelně kontroluje signál *AUTO_STOP*. V případě, že dojde k aktivaci signálu *AUTO_STOP* motorky se vypnou, tlačítko *AUTO* zhasne a rozezní se bzučák.

4.2.6 Čtení enkodéru

V tomhle bodě se provede vyčtení nasnímaných impulzů z enkodéru a kontrola, zda došlo ke změně počtu impulzů. Pokud ano, software přejde na obrazovku *SCREEN_DEPTH* aby uživatel viděl jak se mění nastavená hloubka zabroušení. Protože registr čítající impulzy z enkodéru je široký 16 bitů což umožňuje uložení maximálně 65535 impulsů, je třeba generovat přerušení při přetečení a podtečení registru pro počítání otáček registru:

```
void QEI_Interrupt(void)
{
    if ((QEICON & BIT5) != 0)
        revolutions++;  //otoceni o jednu otacku dopredu
    else
        revolutions--;  //otoceni o jednu otacku dozadu*/
}
```

Při čtení enkodéru se z počtu otáček a aktuálního počtu impulzů spočítá celkový počet impulzů od posledního nulování rozhraní enkodéru:

```
long PulseCount(void)
{
/*    DIS_INT;    //docasne zakazani preruseni
    pulseRet = pulse;   //ulozeni posledniho stavu nactenych pulzu
    EN_INT;     //povoleni preruseni*/
    depth = revolutions * COUNT_PER_REV;
    depth += (int)POSCNT - (int)QEI_MID;
    return depth;    //vrácení nactených pulzu
}
```

4.2.7 Řízení displeje

Displej se překresluje s periodou 200 ms. Tato rychlost sice neumožňuje provádět plynulé změny, ale pro daný účel je dostatečná. Při řízení displeje se nejdříve vykreslí základní rozložení obrazovky, které se překresluje pouze jednou při změně obrazovky:

```
byte lastScreen = 0;
void ShowScreen(byte screenType)
{
       if ((screenType != lastScreen) && (screenType != SCREEN_IDLE))
       {
              lastScreen = screenType;
              screenChange = TRUE;
```

```
switch (screenType)
{
case SCREEN_INTRO:
      DispControl(CLEAR_DIS,0);
      DispControl(SET_POS, 0x05);
      WriteStr("GATAN");
      DispControl(SET_POS, 0x40);
      WriteStr("DIMPLE   GRINDER");
      break;
```

Obr. 36: Úvodní obrazovka

```
case SCREEN_DEPTH:
      DispControl(CLEAR_DIS,0);
      DispControl(SET_POS, 0x02);
      WriteStr("DIMPLE DEPTH");
      DispControl(SET_POS, 0x49);
      WriteStr("um");
      break;
```

Obr. 37: Měření nastavované hloubky zabroušení

```
case SCREEN_SPEED:
      DispControl(CLEAR_DIS,0);
      DispControl(SET_POS, 0x03);
      WriteStr("ARM SPEED");
      break;
```

Obr. 38: Nastavení rychlosti

```
case SCREEN_TIMER:
      DispControl(CLEAR_DIS,0);
      WriteStr("    MIN    SEC  ");
      DispControl(SET_POS, 0x40);
      WriteStr("TIMER: PRESS");
      break;
```

Obr. 39: Nastavení časovače

```
		case SCREEN_LIGHT:
			DispControl(CLEAR_DIS,0);
			WriteStr("BOTTOM    SCOPE ");
			DispControl(SET_POS, 0x41);
			WriteStr("     %        % ");
			break;
```

Obr. 40: Nastavení jasu

```
		default:
			break;
		}
	}
}
```

Po vykreslení základního rozložení obrazovky, přichází na řadu překreslení hodnot proměnných, pokud se od posledního překreslování změnili. Hloubka zabroušení se před samotným testováním na změnu nejdříve zaokrouhlí z desetin μm na jednotky μm:

```
if (pulse > 0)
	pulse += 5;
else
	pulse -= 5;
pulse /= 10;
```

Zaokrouhlování zde využívá principu dělení celých čísel, kdy se zbytek po dělení ztrácí a nijak neovlivňuje výsledek. V případě, že zaokrouhlovaná část čísla je větší než 5, což by matematicky bylo zaokrouhleno směrem k vyšší dekádě, tak přičtením hodnoty 5 dojde k navýšení následující dekády. V případě zaokrouhlované části menší než 5, což by matematicky bylo zaokrouhleno směrem k nižší dekádě, se přičtení hodnoty 5 následující dekáda nezmění. Vydělením takto upravených čísel hodnotou 10 vznikne zaokrouhlené číslo. V záporných číslech je funkce stejná, jenom mají čísla jiné znaménko.

4.2.8 Kruhový buffer a obsluha přerušení

Pro posílání dat na displej je použito rozhraní SPI, které je nastaveno na relativně nízkou rychlost pro omezení EM vyzařování. Proto pro komunikaci s displejem je zvolen asynchronní přístup, kdy data jsou nahrávána do kruhového bufferu a z něj postupně odesílána po SPI do posuvného registru. Každou milisekundu probíhá při obsluze přerušení ms čítače volání funkce, jejímž úkolem je nahrávat data z bufferu do vysílacího registru SPI a nastavovat signál *DISP_RS*:

```
    //ovladac displeje
void DisplayDriver(void)
{
	dword timeTemp = GetTicksInIntExecution();
		//testy: na ubehnuti potrebne doby, zda buffer neni prazdny, zda se s
bufferem nepracuje
	if (((timeTemp - timePassed) >= (waitTime & 0x7F))
		&& (bufferSpace != TR_BUFF_LEN) && ((waitTime & 0x80) == 0))
```

```
	{
		if (bufferSpace < TR_BUFF_LEN)
		{
			PORTD |= ((RSpinCtrl & BIT0) << 7);
			SSPBUF = transmitBuff[transmitBuffInd & BITS6];
			if (((transmitBuff[transmitBuffInd & BITS6] & 0x01) != 0)
				|| ((transmitBuff[transmitBuffInd & BITS6] & 0x02) != 0))
				waitTime = 2;
			else
				waitTime = 1;
			timePassed = timeTemp;
			transmitBuff[transmitBuffInd & BITS6] = 0;
			transmitBuffInd++;
			bufferSpace++;
			RSpinCtrl >>= 1;
			RSpinCtrl |= ((RSpinCtrl2 & BIT0) << (DWORD_BITS - 1));
			RSpinCtrl2 >>= 1;
		}
	}
}
```

Funkce vedle obyčejného posílání dat také kontroluje zda přerušení nenastalo v průběhu manipulace s bufferem, nebo jestli jsou v bufferu vůbec data. Také protože na některé příkazy potřebuje modul displeje více času, tak se zjišťuje, zda od posledního transferu uběhla požadovaná doba. Po projití těmito podmínkami se nastaví signál *DISP_RS* do odpovídající úrovně a data se překopírují do SPI vysílacího registru. Provedou se nezbytné úpravy v bufferu a funkce končí. Jakmile je dokončen přenos, SPI vyvolá přerušení, kde teprve dojde k přijetí dat displejem:

```
void SPI_Interupt(void)
{
	LATD &= (~BIT6);
	while ((PORTD & BIT6) != 0);
	LATD |= BIT6;
	PORTD &= (~BIT7);
}
```

Obsluha SPI přerušení překlopí signál *DISP_E* do LOG0, displej detekuje sestupnou hranu a načte data na vstupních pinech. Z důvody prodlevy s jakou se signál objevuje na pinech mikrokontroléru, se v obsluze čeká na změnu logické úrovně na pinu a až po změně se signál *DISP_E* znovu překlopí do LOG1 signál *DISP_RS* se nastaví do LOG0.

Procesor mikrokontroléru PIC18F4431 má pouze jeden signál přerušení, na který jsou přivedeny všechny zdroje přerušení a je tudíž na programátorovi, aby obsloužil přerušení správné periferie:

```
//obsluha preruseni
void interrupt IntHandler(void)
{
	//identifikace zdroje preruseni podle priority
	if ((PIR3 & BIT0) != 0)   //test na preruseni od casovace 1 ms
	{
		TickInterrupt();
		PIR3 &= (~BIT0);  //vymazani priznaku preruseni
	}
	if ((PIR1 & BIT3) != 0)
	{
```

```
            SPI_Interupt();
            PIR1 &= (~BIT3);
      }
      if ((PIR3 & BIT2) != 0)   //test na preruseni od QEI modulu
      {
            QEI_Interrupt();
            PIR3 &= (~BIT2);  //vymazani priznaku preruseni
      }
}
```

Prioritu přerušení si udává programátor seřazením obsluh přerušení jednotlivých periferií, podle důležitosti, pro případ, že by se v jeden okamžik sešlo více přerušení. Přestože to v projektu jako je tento není nijak kritické, tak nejdůležitější je obsloužit čítač generující vnitřní hodiny o přesnosti 1 ms. Další v pořadí je obsluha SPI, aby nedocházelo ke zbytečným prodlevám při přenosu dat na displej. A nakonec se obslouží modul rozhraní enkodéru z důvodu případného přetečení.

Závěr

Cílem této práce bylo vyvinout řídící elektroniku pro modernizovaný laboratorní přístroj, jehož účelem by bylo nahradit zastaralou verzi a zlepšit postavení na trhu. Požadovaného cíle bylo dosaženo v celém rozsahu nejen zadání diplomové práce, ale i požadavků zákazníka a k veliké spokojenosti všech členů vývojového týmu.

V první kapitole byl uveden abecední seznam a vysvětlení názvosloví a výrazů používaných v průběhu celé práce, které v průběhu vývoje vznikly ze vzájemné komunikace se zákazníkem a neustálého pokroku ve vývoji.

V druhé kapitole bylo popsáno pozadí vývoje a jaké požadavky na řídící elektroniku nového přístroje ze vzájemných diskuzí se zákazníkem vyplynuli. Byly také popsány varianty řešení, které se na počátku vývoje zkoušely a důvody pro jejich odmítnutí.

Obsahem třetí kapitoly byl popis celé elektroniky rozdělené podle jednotlivých modulů. Nejdříve byly sepsány vazby mezi jednotlivými moduly a poté byl popsán podrobně každý modul. Nejrozsáhlejší byl popis spínaného zdroje, s uvedením celého postupu návrhu a všemi potřebnými výpočty.

Čtvrtá kapitola shrnula popis řídícího softwaru, rozděleného na jednotlivé bloky. Byla popsána inicializace, úvodní obrazovka. Vstupem do hlavní programové smyčky byly popsány obsluhy časovaných funkcí, obsluhy tlačítek, kontrola ukončovacího signálu a vyčítání z enkodéru a nakonec byla popsána funkce pro zobrazování na displej a obsluhu přerušení.

Literatura a zdroje

[1] Image_preview.jpg. In: *University of Leuven* [online]. België: University of Leuven, 2016© [cit. 2017-04-28]. Dostupné z: https://www.mtm.kuleuven.be/equipment/dimple/img/dimple1.jpg/image_preview

[2] 03_Dimple Grinder Closed-web.png. In: *Gatan* [online]. United States: Gatan, ©2017 [cit. 2017-04-28]. Dostupné z: http://www.gatan.com/sites/default/files/styles/300w/public/03_Dimple%20Grinder%20Closed-web.png?itok=0LkBiS9n

[3] FAULHABER. *1624_S_DFF.PDF* [online]. 2017 [cit. 2017-04-28]. Dostupné z: https://fmcc.faulhaber.com/resources/img/EN_1624_S_DFF.PDF

[4] TE CONNECTIVITY. *3521.pdf* [online]. 2016 [cit. 2017-04-28]. Dostupné z: http://www.te.com/commerce/DocumentDelivery/DDEController?Action=showdoc&DocId=Data+Sheet%7F9-1773463-5%7FD%7Fpdf%7FEnglish%7FENG_DS_9-1773463-5_D.pdf%7F2176070-1

[5] DOMINANT OPTO TECHNOLOGIES. *DSW-NSG-V2W-1.pdf* [online]. 2012 [cit. 2017-04-28]. Dostupné z: http://www.dominant-semi.com/userfiles/file/Right_Angle_DomiLED_InGaN_White_DSW-NSG-Catalogue-v6.pdf

[6] PANASONIC. *EEEFTxxxxxA(P%R).pdf* [online]. 2017 [cit. 2017-04-28]. Dostupné z: https://industrial.panasonic.com/cdbs/www-data/pdf/RDE0000/ABA0000C1240.pdf

[7] MURATA. *GRM31CR61H106KA12.pdf* [online]. 2016 [cit. 2017-04-28]. Dostupné z: http://search.murata.co.jp/Ceramy/image/img/A01X/G101/ENG/GRM31CR61H106KA12-01.pdf

[8] INTERNATIONAL RECTIFIER. *IRLML5203PBF.pdf* [online]. ©2014 [cit. 2017-04-28]. Dostupné z: http://www.infineon.com/dgdl/irlml5203pbf.pdf?fileId=5546d462533600a40153566868da261d

[9] LUMEX. *LCM-S01602DSR%A.pdf* [online]. 2011 [cit. 2017-04-28]. Dostupné z: http://www.lumex.com/content/files/productattachment/lcm-s01602dsr-a.pdf

[10] TEXAS INSTRUMENTS. *Lm317a.pdf* [online]. 2015 [cit. 2017-04-28]. Dostupné z: http://www.ti.com/lit/ds/symlink/lm317a.pdf

[11] MPLAB® XC Compilers. *Microchip* [online]. Arizona, USA ,2017 [cit. 2017-04-28]. Dostupné z: http://www.microchip.com/mplab/compilers

[12] ON SEMICONDUCTOR. *MBRA340T3-D.PDF* [online]. 2017 [cit. 2017-04-28]. Dostupné z: https://www.onsemi.com/pub/Collateral/MBRA340T3-D.PDF

[13] RC Low-pass Filter Design Tool. *OKAWA Electric Design* [online]. 2017 [cit. 2017-04-28]. Dostupné z: http://sim.okawa-denshi.jp/en/CRlowkeisan.htm

[14] TDK. *MLZ.pdf* [online]. 2016 [cit. 2017-04-28]. Dostupné z: http://www.mouser.com/ds/2/400/nductor_commercial_decoupling_mlz1608_en-837720.pdf

[15] COILCRAFT. *MSS6132.pdf* [online]. 2015 [cit. 2017-04-28]. Dostupné z: http://www.mouser.com/ds/2/597/mss6132-270629.pdf

[16] MICROCHIP. *39616d.pdf* [online]. 2010 [cit. 2017-04-28]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/39616d.pdf

[17] SEMTECH. *SMF05C.pdf* [online]. 2004 [cit. 2017-04-28]. Dostupné z: http://www.semtech.com/images/datasheet/smf05c.pdf

[18] TEXAS INSTRUMENTS. *TPS54231.pdf* [online]. 2014 [cit. 2017-04-28]. Dostupné z: http://www.ti.com/lit/ds/symlink/tps54231.pdf

[19] MPLAB® X Integrated Development Environment (IDE). *Microchip* [online]. Arizona, USA, 2017 [cit. 2017-04-28]. Dostupné z: http://www.microchip.com/mplab/mplab-x-ide

[20] LUMILEDS. *DS61-pdf* [online]. 2016 [cit. 2017-04-28]. Dostupné z: http://www.lumileds.com/uploads/17/DS61-pdf

Přílohy

Příloha I Archiv použitých katalogových listů

Příloha II Archiv zdrojových kódů

Příloha III Archiv podkladů pro elektroniku