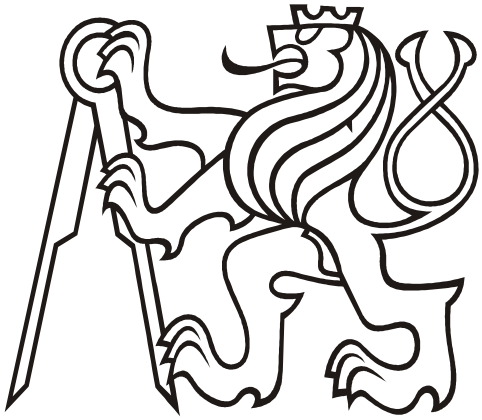
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE

2011 Bc. Michal Tomáš

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra měření



Snímač polohy s projekcí stínového obrazu

Doc. Ing. Jan Fischer, CSc.

Bc. Michal Tomáš

Praha 2011

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

Tato práce vznikla v laboratoři videometrie, katedry měření ČVUT – FEL v Praze pod vedením doc. Ing. Jana Fischera, CSc. Navazuje též na výzkum v rámci MSM6840770015 – „Výzkum metod a systémů pro měření fyzikálních veličin a zpracování naměřených dat“, jehož některé poznatky a výstupy v oblasti optoelektronických senzorů využívá.

V Praze dne ……. …….

podpis

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra měření

Akademický rok **2010-2011**

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	**Bc. Michal TOMÁŠ**
Program:	**Kybernetika a robotika**
Obor:	**Senzory a přístrojová technika**
Název tématu česky:	**Snímač polohy s projekcí stínového obrazu**
Název tématu anglicky:	**Position Sensor Based on Shadow Image Projection**

Pokyny pro vypracování:

Analyzujte možné způsoby realizace přesného určení polohy mechanického akčního členu vyhodnocením stínového obrazu promítaného na senzor CMOS. Navrhněte konkrétní řešení snímače pro určení polohy v jedné a ve dvou osách. Sestavte funkční vzorek snímače a vytvořte potřebnou řídicí elektroniku s využitím procesoru řady STM32, který bude zajišťovat čtení obrazové informace a její zpracování, regulaci polohy akčního členu i přenos potřebných dat na PC. Vytvořte příslušné programové vybavení pro procesor STM 32 i potřebný ovladač pro nadřazené PC v návaznosti na platformu s LabView.

Seznam odborné literatury:

[1] Yiu J.: The definitive Guide to the ARM Cortex- M3. Elsevier, 2007
[2] Gasvik, K. J.: Optical metrology. 3-th edition, Wiley, 2002.
[3] RM0008 - Reference manual , Doc. ID 13902 Rev 11, STMicroelectronics 2010

Vedoucí diplomové práce:	doc. Ing. Jan Fischer, CSc.
Datum zadání diplomové práce:	5. ledna 2011
Platnost zadání do[1]:	29. června 2012

L.S.

Prof. Ing. Pavel Ripka, CSc.
vedoucí katedry

Prof. Ing. Boris Šimák, CSc.
děkan

V Praze dne 5. 1. 2011

[1] Platnost zadání je omezena na dobu tří následujících semestrů.

Anotace

Diplomová práce se zabývá návrhem snímače polohy s využitím zpracování obrazové informace a realizací hardwaru s mikroprocesorem STM32, který spolupracuje s obrazovým senzorem CMOS a prvky pro ovládání stejnosměrných a krokových motorů. Dále se zabývá problematikou využití získané informace o poloze k regulaci polohovacího mechanismu. Snahou je soustředit výpočetní výkon do mikroprocesoru, který se může chovat jako autonomní systém a nahradit PC.

Annotation

This thesis describes the design of position sensor using image information processing and hardware implementation with a microprocessor STM32, which cooperates with a CMOS image sensor and elements to control DC and stepper motors. It also deals with the issue of using knowledge about the position to control the positioning mechanism. Computational effort is concentrated in the microprocessor, which can act as an autonomous system, and replace the PC.

Obsah

Seznam obrázků

Seznam tabulek

Seznam rovnic

1 ÚVOD

S rozvojem vědy a techniky se využívá stále modernějších technologií v oblasti průmyslu, kde je nejen ve výrobě žádoucí zajistit spolehlivost, ale i dobrou kvalitu produktů. Aby mohl být výrobní proces co nejvíce automatizován a obešel se i bez zásahů člověka, který by se neměl pohybovat v nebezpečných či nepříznivých prostředích a neměl by vnášet do procesu negativní ovlivňující faktor jako např. únava, nepřesnost, nespolehlivost, apod., tak se ve stále větším zastoupení používají kamerové systémy. Tyto systémy umožňují dohled nad výrobním procesem, kontrolu rozměrů a jejich měření, identifikaci předmětu, ale lze je použít i ve spoustě jiných aplikací, například viz [8]. Kamera tak může nejen nahradit oko člověka, ale z pořízeného obrazu v kombinaci s číslicovým zpracováním dat lze získat mnoho dalších užitečných informací, které mohou posloužit jako vstupní data pro další zpracování v měřícím řetězci.

Mohutný vývoj právě v oblasti obrazových senzorů CMOS v posledních letech umožňuje využití kamerového systému pro širokou škálu aplikací a jejich hromadná výroba přispívá i k příznivé ceně.

Tato diplomová práce se zabývá návrhem jednoduchého hardwaru v kombinaci s vývojem programového vybavení realizující snímač polohy, který využívá zpracování dat z obrazového snímače CMOS k bezkontaktnímu měření polohy snímaných objektů. Dále problematikou využití získané informace o poloze k řízení akčních členů a jejich případnou regulací. Hlavním smyslem je vytvoření modulu s mikroprocesorem, do kterého bude soustředěn veškerý výpočetní výkon. Bude tak možné vytvořit autonomní systém, který se pro jednoduché zpracování obrazových dat a řízení akčních členů obejde i bez nutného použití počítače.

Srdcem celého systému bude 32-bitový mikroprocesor společnosti *STMicroelektronics*, který je postaven na architektuře *ARM Cortex-M3* a v současnosti je využíván nejen v průmyslové automatizaci, ale i v řadě různých zařízení spotřební elektroniky. Měl by být tedy dobrou volbou pro použití i v kamerovém snímači polohy. Vybavenost tohoto mikroprocesoru řadou integrovaných periferií by zároveň měla minimalizovat hardwarové požadavky.

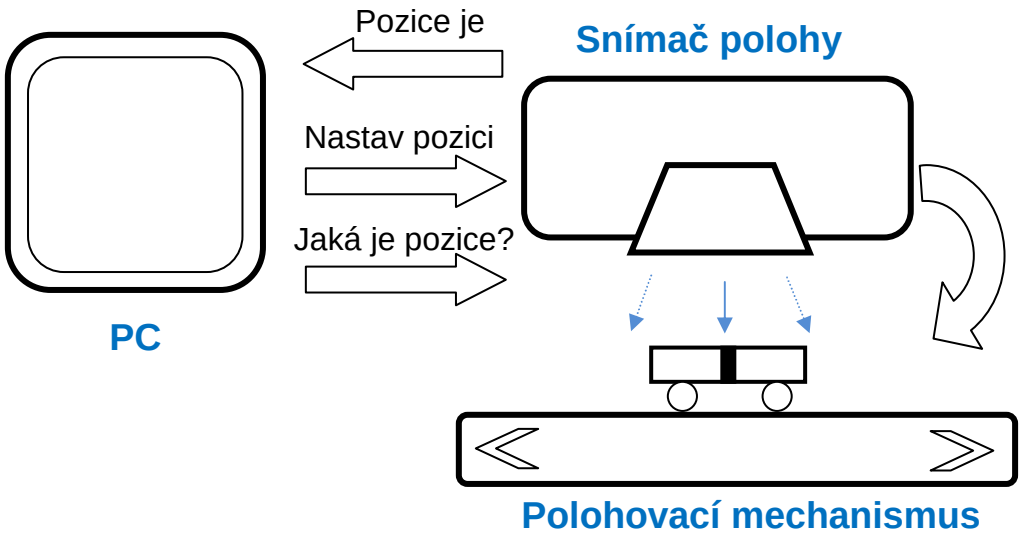
Vzniklý výsledný modul snímače by mohl najít uplatnění v bezkontaktních měřících systémech a sloužit nejen k určování polohy, ale i k řízení polohovacích mechanismů.

2 CÍL VÝSLEDNÉHO ŘEŠENÍ SNÍMAČE POLOHY

Představou řešení výsledného snímače polohy s využitím obrazového senzoru je vytvoření, do značné míry univerzálního, modulu s mikroprocesorem STM32, který bude schopný určovat polohu, ale i jednoduché řízení akčních členů v podobě motorů. Snímač by měl být schopný určit přesnou polohu objektu v jednorozměrné scéně, tj. rychlé snímaní v řádkovém režimu, ale i určení polohy objektu zpracováním plošné scény. S využitím získané informace o poloze by mělo být umožněno polohování nějakého mechanismu v jedné i ve dvou osách zorné scény. Prioritním akčním členem by měl být stejnosměrný motor, ale modul by měl být schopen ovládat i krokový motor.

Modul snímače polohy by měl co nejvíce využívat možnosti integrovaných periferií mikroprocesoru, například použití vestavěného rozhraní řadiče USB ke komunikaci s počítačem. Počítač by však neměl významně zasahovat do procesu měření nebo řízení akčních členů, ale bude pouze sloužit k nastavení parametrů snímače a k prezentaci získaných dat.

Pro pohodlnější komunikaci a parametrizaci s hardwarem modulu snímače, bude vývoj směřován k návrhu grafické obslužné aplikce pro PC. Dále v návaznosti na tuto aplikaci by měla vzniknout podpora ve formě knihovny, která umožní přenositelnost mezi některými vývojovými nástroji pod operačním systémem Windows (například *Microsoft Visual Studio C#, C/C++, VB, J#* anebo také *National Instrument LabView, LabWindows*, atd.). Takováto podpora může umožnit další programování systému snímače a s využitím dostupných funkcí by mohla být sestavena různorodá škála alternativních aplikací.



Obr. 2.1 – Uspořádání měřícího mechanismu se snímačem polohy

3 KONSTRUKCE A PRINCIPY PRVKŮ SNÍMAČE

Proces sběru a zpracování obrazových dat klade důraz na znalosti problematiky a principů v určitém spektru tak, aby je bylo možné využít i pro regulaci polohy. Důležité jsou především znalosti v oblasti optiky, principy vzniku obrazu a jeho reprezentaci v digitální podobě. Dále je nezbytná znalost principů různých typů motorů, kterých je využíváno jako akčních členů pro polohování, a jejich metod řízení.

3.1 obrazový senzor

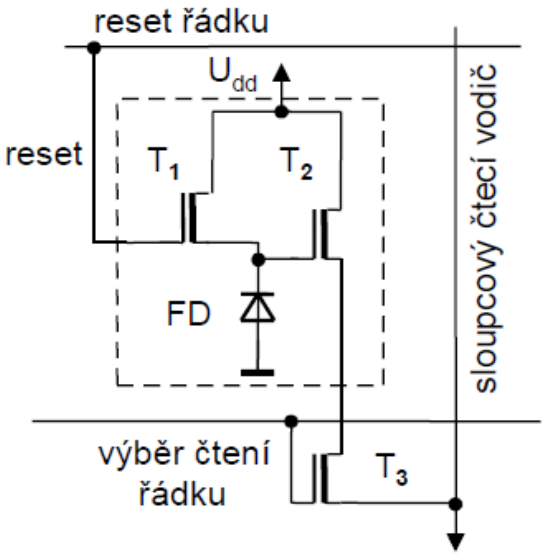
Existují dvě základní technologie výroby obrazových snímačů CCD (Charged Coupled Device) a CMOS (Complementary Metal Oxide Semiconductor). CCD snímače paří stále mezi kvalitnější vzhledem k lepší světelné citlivosti v prostředí se zhoršeným osvětlením, ale také k dražším, což je především vykoupeno výrobním procesem, v němž nelze vytvořit všechny potřebné podpůrné obvody (zesilovače, převodníky, atd.). CMOS snímače jsou vyráběny shodnou technologií jako procesory, a tak lze na jednom čipu vyrobit i všechny důležité podpůrné obvody. Tato výhoda se výrazně projeví nejen v ceně, ale i v menších rozměrech a spotřebě. Nevýhodou CMOS čipů je však stále menši světelná citlivost a větší šum proti CCD.

Ke snímání obrazu je v této práci použito senzoru CMOS, proto v následujícím textu bude popis vztažen vždy k tomuto typu. Konkrétní parametry senzoru jsou uvedeny v *kapitole 4.2*.

3.1.1 Princip vytvoření obrazu

Senzor se zpravidla skládá z polovodičových elementů (pixelů) složených do maticové struktury, které jsou schopny absorbovat světlo v rozsahu určitých vlnových délek. V podstatě se jedná o fotodiodu vytvořenou v křemíkovém substrátu, která přemění dopadající fotony na elektrický náboj. Vzniklý elektrický náboj z jednotlivých buněk je vyčítán, digitalizován a zaznamenáván. Množství akumulovaného náboje závisí na velikosti intensity osvětlení a také na době expozice, po kterou se náboj akumuluje.

Schematické znázornění pixelové buňky je patrné z *obr. 3.1*. Kromě zmíněné fotodiody jsou ve struktuře vytvořeny i tranzistory plnící určitou funkci. Tranzistor *T1* slouží k resetu buňky, která je pak připravena na další fázi záznamu obrazu. Vyčtení náboje se děje pomocí emitorového sledovače *T2* a tranzistoru *T3*, který po aktivaci vodiče řádku připojí čtecí vodič sloupce a umožní tak transport náboje do výstupního obvodu. Tímto způsobem je řešeno vyčítání obrazových dat po řádcích ze senzoru.

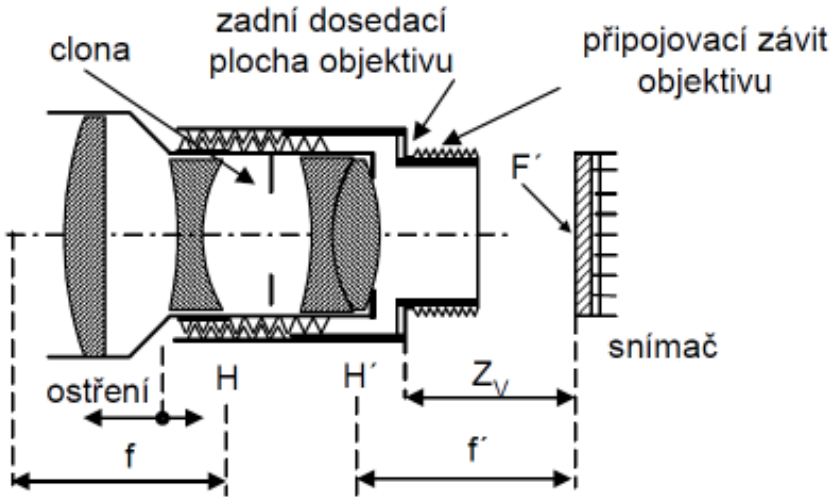


Obr. 3.1 – Schematická struktura buňky pixelu CMOS senzoru, převzato z [1]

3.1.2 Pořízení obrazu v kombinaci snímače s objektivem

Pouhý obrazový senzor většinou nestačí k pořízení obrazu vhodné kvality, a proto se senzory vybavují objektivy, které soustředí svazky světelných paprsků na plochu senzoru. Konstrukční uspořádání objektivu je uvedeno na *obr. 3.2*. Prakticky se jedná o soustavu vhodně uspořádaných čoček s mechanizmem umožňující ostření a clonění obrazu.

Ostření spočívá v posuvu soustavy čoček po dráze závitu, kde je snaha soustředit svazek paprsků, tak aby se vzniklý obraz nepromítnul před nebo za snímačem, ale co nejpřesněji na jeho aktivní ploše. Výsledkem je potom ostrý obraz, ve kterém lze dobře rozpoznat případné hledané objekty.



Obr. 3.2 – Konstrukce objektivu, , převzato z [1]

Podstata clonění je vhodná úprava množství dopadajícího světla na snímač tak, aby nedocházelo k jeho přesvícení. Řada objektivů je vybavena tzv. iris, kterou lze efektivně zmenšit či zvětšit průměr vstupní čočky. Údaj o světlosti objektivu určuje clonové číslo, kde nižší clonové číslo znamená větší světelný tok vstupující do objektivu, což umožňuje krátkou dobu expozice na úkor nízké hloubky ostrosti. Clonové číslo lze vyjádřit následujícím vztahem.

$$k = \frac{f}{D}$$

Rce. 3.1 – Rovnice clonového čísla

Kde: k – clonové číslo, f – ohnisko objektivu, D – průměr vstupní čočky objektivu

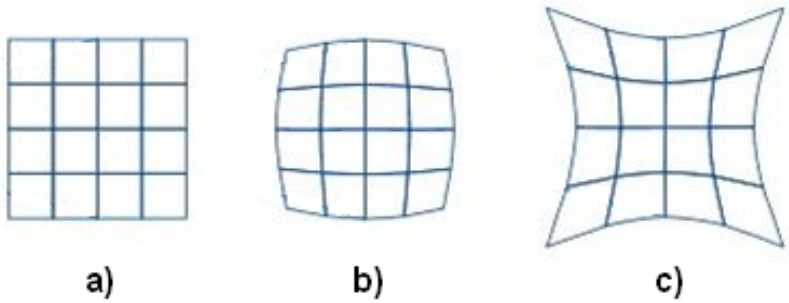
Existuje celá řada objektivů s různými vlastnostmi a parametry vymezující možnou oblast použití. Podle ohniskové vzdálenosti nebo úhlu zorného pole lze rozlišit následující typy objektivů:

- Normální objektiv – obraz srovnatelný s lidským okem
- Širokoúhlý objektiv – široký úhel záběru (až 180°)
- Teleobjektiv – úzký úhel záběru

Různé typy objektivů i v závislosti na kvalitě mohou mít různé typy vad jak žádoucí, tak nežádoucí. Zabývat se všemi možnými typy vad přesahuje rámec tohoto textu, více informací je k dispozici například v [6] nebo [7].

Za zmínku však stojí například rozměrové zkreslení obrazu vlivem různě velkého zvětšení předmětu ve středu a na okrajích obrazu, kde se projevují deformace typu soudkovitost, poduškovitost a jejich kombinace, viz *obr. 3.3*. Soudkovitost obrazu je však žádoucí u objektivů typu „Rybí oko“.

Naopak příkladem jasové deformace může být vinětce, která se projevuje poklesem jasu na okrajích obrazu zejména vlivem konstrukce objektivu. Vinětací trpí širokoúhlé objektivy.



Obr. 3.3 – a) obraz bez zkreslení, b) Soudkovité zkreslení, c) Poduškovité zkreslení

3.1.3 Videometricé výpočty s objektivem

Má-li být použit obrazový senzor v kombinaci s objektivem k měření rozměrů předmětu či určování vzdáleností, je nutné znát potřebné zákonitosti a vztahy. Podle níže uvedených vztahů lze například stanovit, jak se projeví velikost jednoho pixelu obrazového senzoru na velikosti snímaného předmětu anebo jaký možný zorný úhel může objektiv pokrýt.

$$-\beta' = \frac{f'}{z}, \; -\beta' = \frac{y'}{y} = \frac{a'}{a}$$

Rce. 3.2 – Rovnice zvětšení objektivu

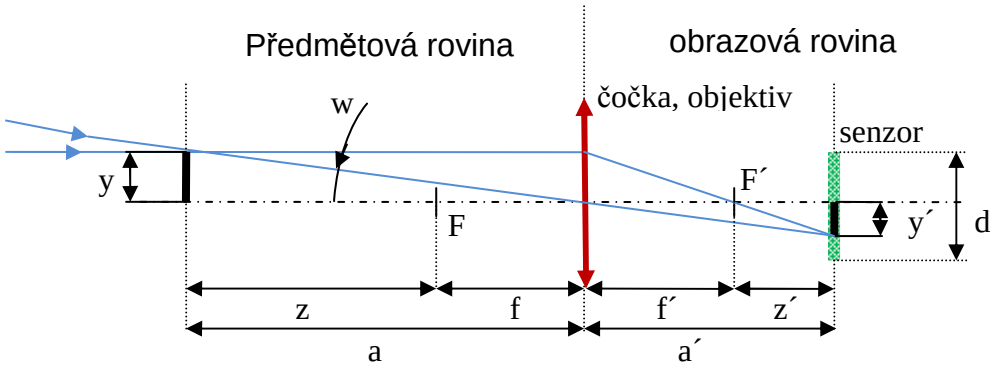
$$\frac{1}{f'} = \frac{1}{a} + \frac{1}{a'}, \; f \cdot f' = z \cdot z', \; f'^2 = z \cdot z'$$

Rce. 3.3 – Zobrazovací rovnice

$$2w = 2arctg\frac{d}{2f'}$$

Rce. 3.4 – Rovnice úhlu obrazového pole objektivu

Kde: β´- zvětšení objektivu, f, f´ - ohnisková vzdálenost, y, y´- velikost předmětu obrazu, z, z´- vzdálenost od ohniska, a, a´- vzdálenost od předmětu, obrazu od objektivu, d – velikost senzoru, w – poloviční úhel obrazového pole



Obr. 3.4 – Chod paprsku při zobrazení objektivem, modifikace z [1]

3.1.4 Zpracování obrazu

Možných metod zpracování obrazu je celá škála a jejich použití se odvíjí od požadavku co je v úmyslu s obrazem provést anebo jaká informace se hledá. Většina videometrických operací se však týká nalezení nějakého charakteristického objektu, a tak základní operací s obrazem je nalezení ohraničení vymezující tvar či polohu objektu.

Hrana v obraze se projevuje skokovou změnou jasu. K nalezení hran se většinou používá některých z následujících metod:

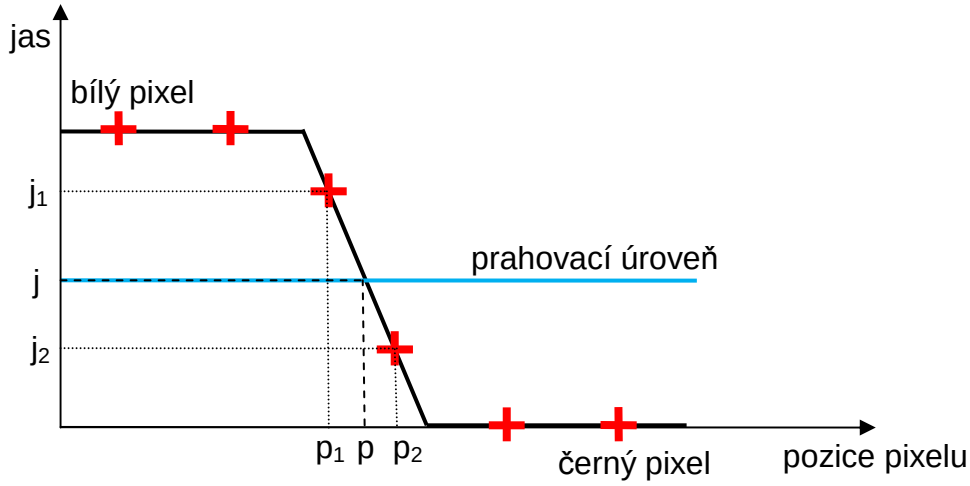
- Komparace, prahování,
- Vlastnosti 1. a 2. derivace, diference,
- Interpolace.

Komparační metody spočívají vhodným nastavením úrovně jasu, blízko které se předpokládá nalezení hrany. Metody založené na derivaci předpokládají nalezení inflexního bodu (změna úrovně jasu), ve kterém jasová funkce nabývá svého maxima (1. derivace) a průchod v inflexním bodě nulou (2. derivace). Na nalezené hrany se poté aplikují interpolační metody, které zpřesňují její polohu i více než na rozlišení jednoho pixelu.

Modelová situace nalezení spádové hrany je znázorněna na *obr. 3.5* a výpočet pozice podle *rce. 3.5*. Hrana je zde nalezena nastavením vhodné úrovně prahu a po té nad sousedními pixely obklopující prahovací úroveň je zpřesněna její pozice.

$$p = p_1 + |p_2 - p_1| \frac{j - j_1}{j_2 - j_1} = p_1 + \frac{j - j_1}{j_2 - j_1}$$

Rce. 3.5 – Stanovení pozice pixelu



Obr. 3.5 – Nalezení pozice interpolací na spádové hraně

Výše zmíněné operace jsou zejména vhodné ke zpracování řádku obrazu. V plošném obraze jich samozřejmě lze také využít, ale k nalezení polohy objektu se spíše využívá metod nalezení těžiště za předpokladu, že hledaný objekt má charakter nějakého geometrického tvaru a potom jeho těžiště odpovídá právě jeho středu.

$$X = \frac{\sum_{y}^{ROW}\sum_{x}^{COL} x \cdot \mathrm{Im}(x,y)}{\sum_{y}^{ROW}\sum_{x}^{COL} \mathrm{Im}(x,y)} \qquad Y = \frac{\sum_{y}^{ROW}\sum_{x}^{COL} x \cdot \mathrm{Im}(x,y)}{\sum_{y}^{ROW}\sum_{x}^{COL} \mathrm{Im}(x,y)}$$

Rce. 3.6 – Rovnice výpočetu souřadnic těžiště

Kde: X,Y - souřadnice těžiště, Im(x,y) - souřadnice v obraze.

3.2 Stejnosměrný motor

3.2.1 Konstrukce a princip stejnosměrného motoru

Stejnosměrný motor je točivý stroj složený ze statoru a rotoru. Z hlediska buzení lze motory rozdělit do několika skupin:

- Motory s cizím buzením (nebo permanentními magnety),
- Motory s paralelním buzením,
- Motory se sériovým buzením,
- Motory se smíšeným buzením.

Základní princip činnosti všech těchto motorů je však stejný, proto následující text bude popisovat motor s permanentními magnety, který se nejvíce využívá pro motory malých výkonů a je použit i v této práci.

Stator je tvořen permanentním magnetem vytvářející magnetické pole, v němž je umístěn rotor složený z kotvy a komutátoru. Obvod kotvy tvoří cívky (nejméně dvě) napájené přes komutátor, který přepíná směr elektrického proudu a polaritu magnetického pole procházejícího kotvou, a tím je zajištěno roztočení rotoru motoru.

Vztah mezi svorovým napětím a indukovaným napětí vytvořeným v obvodu kotvy lze jednoduše popsat podle *rce. 3.7*, přičemž je vycházeno z náhradního schématu na *obr. 3.6*. Točivý moment motoru závisí na velikosti magnetického toku statoru a na proudu protékaném v obvodu kotvy podle *rce. 3.9*. Výsledný užitečný moment na hřídeli je však zmenšen o ztráty v železe a ztráty mechanické.

$$u(t) = R \cdot i(t) + L \cdot i'(t) + u_i(t)$$

Rce. 3.7 – Rovnice obvodu kotvy stejnosměrného motoru

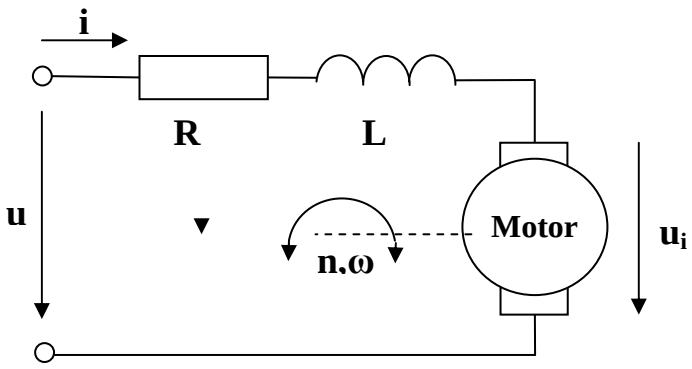
$$u_i(t) = k \cdot \Phi \cdot \varphi'(t) = k \cdot \Phi \cdot \omega(t) = k \cdot \Phi \cdot n(t)$$

Rce. 3.8 – Rovnice indukovaného napětí kotvy stejnosměrného motoru

$$M(t) = J\varphi''(t) = k \cdot \Phi \cdot i(t)$$

Rce. 3.9 – Rovnice momentu stejnosměrného motoru

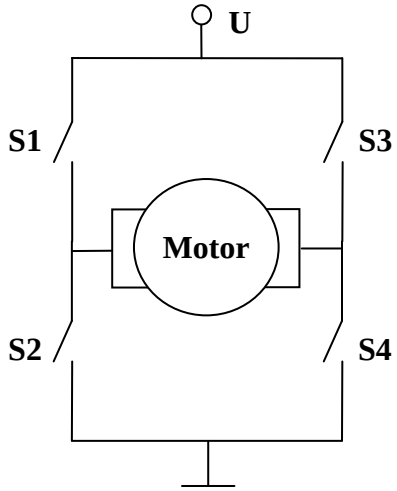
Kde: u – svorkové napětí, R – ohmický odpor kotvy, L – indukčnost vinutí kotvy, i – proud obvodu kotvy, u_i – indukované napětí kotvy, k – konstrukční konstanta motoru, Φ - magnetický tok statoru, ω – úhlová rychlost, n – otáčky hřídele rotoru, φ – úhel natočení hřídele, J – moment setrvačnosti, M – točivý moment motoru



Obr. 3.6 – Náhradní schéma stejnosměrného motoru

3.2.2 Řízení stejnosměrného motoru

Nejen u stejnosměrných motorů je hlavním parametrem řízení rychlost otáčení jeho hřídele nebo síla záběrného momentu. Velikost momentu lze přímo úměrně ovlivnit velikostí proudu v obvodu kotvy. Rychlost otáčení motoru je závislá na velikostí přiloženého svorkového napětí.



Obr. 3.7 – Principiální schéma obousměrného ovládání stejnosměrného motoru pomocí H -můstku

Z hlediska číslicového ovládání se motory nejčastěji řídí na bázi pulzní šířkové modulace – PWM (*Pulse Width Modulation*), kde se na motor s vhodnou frekvencí přivádí svorkové napětí. Střední hodnota tohoto napětí potom určuje výsledné otáčky motoru (případně i velikost záběrné síly).

Je-li potřebné ovládat motor pouze v jediném směru otáčení, postačí pouze jeden spínací prvek. Situace se poněkud zkomplikuje, je-li vyžadováno řízení obousměrné. Jelikož pro reverzaci směru otáčení je nutné přepólování svorek motoru, je již nutno použít čtyř spínacích prvků a jejich zapojení do tzv. H-můstku viz *obr. 3.7*. Směr je pak řízen sepnutím vždy dvou protilehlých spínačů, tedy *S1* a *S4* nebo *S2* a *S3*.

3.3 Krokový motor

3.3.1 Konstrukce a princip krokového motoru

Speciální druh točivého stroje je i krokový motor přizpůsobený ke změně polohy po krocích vycházejících z jeho konstrukce. Činnost spočívá v natáčení rotoru z prstence permanentních magnetů s vystupujícími póly, které se snaží zaujmout polohu v magnetickém poli vytvořeném průchodem proudu procházejícího ve vinutí pólových nástavců v obvodu statoru. Rotor se tedy vždy snaží natočit své póly do nesouhlasné orientace vůči magnetické orientaci v cívkách statoru. Vhodným cyklickým přepínáním cívek je vyvoláno točivé magnetické pole a otáčení rotoru. Zvyšováním rychlosti přepínání jednotlivých cívek se zvyšuje i úhlová rychlost rotoru a zároveň se snižuje krouticí moment. V okamžiku snížení momentu na hodnotu menší než je velikost zátěže, kterou rotor musí překonat, začne motor ztrácet kroky nebo se úplně zastaví. Minimální prodleva mezi jednotlivými kroky se u běžných krokových motorů pohybuje v řádu jednotek až desítek milisekund.

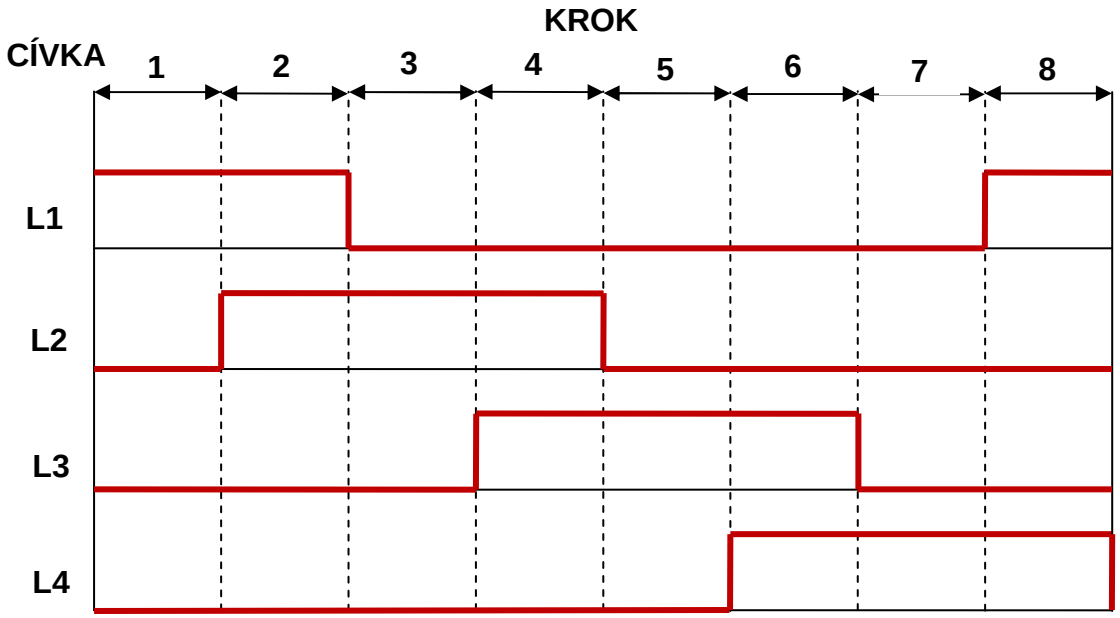
3.3.2 Řízení krokového motoru

Z hlediska řízení lze tyto motory rozdělit do dvou základních skupin vycházejících z propojení a vyvedení jednotlivých cívek. Nejsou-li vyvedeny středy jednotlivých vinutí (vyvedeny 4 vodiče), pak lze hovořit o bipolárním krokovém motoru, který se vyžaduje při spínání cívek i jejich přepólování. K řízení jednoho motoru se nejčastěji využívá dvou H-můstků. V situaci, kdy jsou vyvedeny i středy vinutí (vyvedeno 5 až 6 vodičů), jde o motor unipolární a k jeho řízení stačí pouze spínání cívek jedné polarity. Další informace lze nalézt v [13].

V závislosti na spínání počtu cívek je možné provádět buď plné (čtyřtaktní řízení) nebo poloviční kroky (osmitaktní řízení). Režim s polovičními kroky je vynucen spínáním vždy

dvou sousedních cívek, kdy je rotor svými pólovými nástavci přitahován mezi dvě cívky statoru. Princip spínání cívek je znázorněn na *obr. 3.8.*

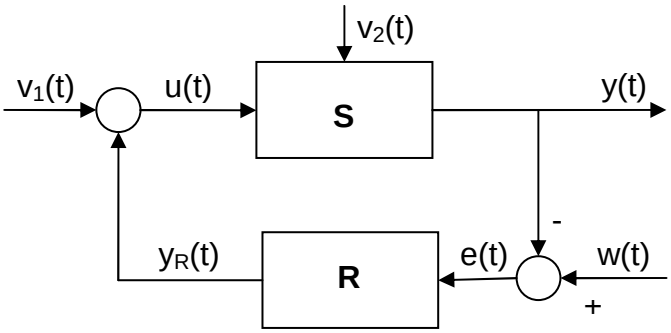
Tato varianta lze ještě vylepšit o preciznější a přesnější polohování, kdy bude cívkami procházet různě velký proud a dojde tak k přitažení pólového nástavce rotoru k jedné cívce méně a k druhé více. Realizace na tomto principu je označována jako mikrosteping a jeden krok motoru umožňuje rozdělit na několik dílčích kroků v závislosti na poměru proudů protékajících mezi sousedními cívkami. Takové řízení však vyžaduje regulaci spínaného proudu.



Obr. 3.8 – Příklad řízení unipolárního krokového motoru v režimu s poloviční krokem

3.4 Regulační smyčka

K výhodnému a efektivnímu řízení téměř kteréhokoli systému se používá tzv. regulační smyčka podle *obr. 3.9.* Činnost regulační smyčky je založena na neustálém srovnávání vstupní požadované veličiny s výstupní dosaženou veličinou prostřednictvím záporné zpětné vazby. Základní strukturu tvoří regulátor a regulovaná soustava. Na vstup regulátoru je přivedena regulační odchylka a podle její velikosti regulátor nastaví vhodnou řídící veličinu pro regulovanou soustavu, aby odchylka byla co nejmenší. Takovýto přístup umožní nejen dynamické přizpůsobení ke změně vstupní žádané veličiny, ale i přizpůsobení k možným poruchovým veličinám ovlivňující celý regulační proces.



Obr. 3.9 – Blokové schéma regulační smyčky, převzato z [2]

Kde: S – regulovaná soustava, R – regulátor, y – regulovaná veličina, w – žádaná hodnota regulované veličiny, e - regulační odchylka, y_R – akční veličina, u – řídící veličina, v – poruchová veličina

3.4.1 Popis regulátoru

Regulátory je možné třídit z několika hledisek, ale s mírným nadhledem je lze rozdělit do dvou základních skupin podle charakteru průběhu vstupní veličiny. Regulátory spojité v podobě PID a nespojité v podobě PSD. Více informací nejen o rozdělení regulátorů, ale i o principech regulace pojednává literatura [2], ze které bylo čerpáno.

PID regulátory (Proporcionálně-Integrační-Derivační regulátor) v sobě zahrnují jednotlivé složky různě ovlivňující regulační zásah. Proporcionální člen je určujícím faktorem regulační odchylky a jeho konstanta *P* udává zesílení regulátoru. Integrační člen stanovuje reakci na dobu trvání regulační odchylky a reakci lze ovlivnit konstantou *I*. Derivační člen a jeho konstanta *D* určují faktor rychlosti změny regulační odchylky. Vlastní regulátor lze sestavit z jednotlivých kombinací členů zejména ve variantě P, I, D, PI, PS, PID. Výpočet výsledné akční veličiny pro variantu PID popisuje *rce. 3.10* nebo v operátorovém tvaru po Laplaceově transformaci ve tvaru podle *rce. 3.11*.

$$y_R(t) = P \cdot e(t) + I \cdot \int_0^t e(\tau) d\tau + D \cdot \frac{de(t)}{dt}$$

Rce. 3.10 – Rovnice PID regulátoru

$$\frac{Y_R(s)}{E(s)} = P + \frac{I}{s} + D \cdot s$$

Rce. 3.11 – Rovnice ideálního PID regulátoru v operátorovém tvaru

PSD regulátor (Proporcionálně-Sumační-Diferenční regulátor) je číslicovou analogií PID regulátoru a je vhodnou variantou pro realizaci číslicovými prostředky, jako je mikroprocesor. Zásadním rozdílem je definice akční veličiny pouze v diskrétních krocích *k* se vzorkovací periodou *T*. Výslednou akční veličinu pro variantu PSD popisuje rce. 3.12 nebo v operátorovém tvaru po Z-transformaci ve tvaru podle *rce. 3.12*.

$$y_R(kT) = P \cdot e(kT) + I \cdot T \cdot \sum_{i=1}^{k} e(iT) + D \cdot \frac{1}{T} \cdot \left(e(kT) - e[(k-1)T]\right)$$

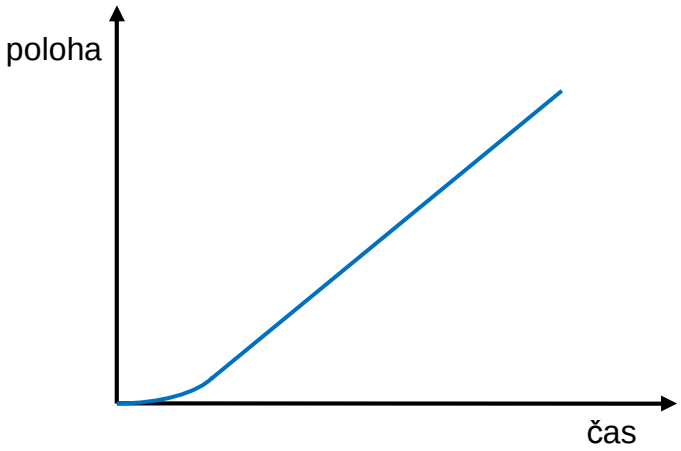
Rce. 3.12 – Rovnice PSD regulátoru

$$\frac{Y_R(z)}{E(z)} = P + I \cdot T \cdot \frac{z}{z-1} + D \cdot \frac{1}{T} \frac{z-1}{z}$$

Rce. 3.13 – Rovnice ideálního PSD regulátoru v operátorovém tvaru

3.4.2 Identifikace regulované soustavy

Ke kvalitnímu návrhu regulátoru je nezbytné znát matematický popis statických a dynamických vlastností soustavy, která se má regulovat. K tomu účelu je nutné podrobit regulovanou soustavu tzv. identifikaci, tj. sestavit matematický model, změřit vstupní a výstupní veličiny soustavy a následně vyhodnotit závislosti mezi nimi. Základními metodami identifikace je přechodová nebo frekvenční charakteristika.



Obr. 3.10 – Přechodová charakteristika modelu stejnosměrného motoru

K sestavení analytického modelu stejnosměrného motoru lze využít vztahů z *kapitoly 3.2.1* a popis závislosti vstupního napětí na úhlu (poloze) je pak definován podle *rce. 3.14.* Přenosem soustavy podle *rce. 3.15* vznikne systém třetího řádu, jehož chování demonstruje přechodová charakteristika na *obr. 3.10.* Pro tento popis soustavy jde o skokovou změnu svorkového vstupního napětí, které je přivedeno na motor a výstupem je pak příslušná změna

polohy. Charakteristika slouží především pro odhad dynamického chování stejnosměrného motoru jako regulované soustavy a zároveň pro lepší návrh regulátoru.

$$u(t) = \frac{L \cdot J}{k \cdot \Phi} \varphi'''(t) + \frac{R \cdot J}{k \cdot \Phi} \varphi''(t) + k \cdot \Phi \cdot \varphi'(t)$$

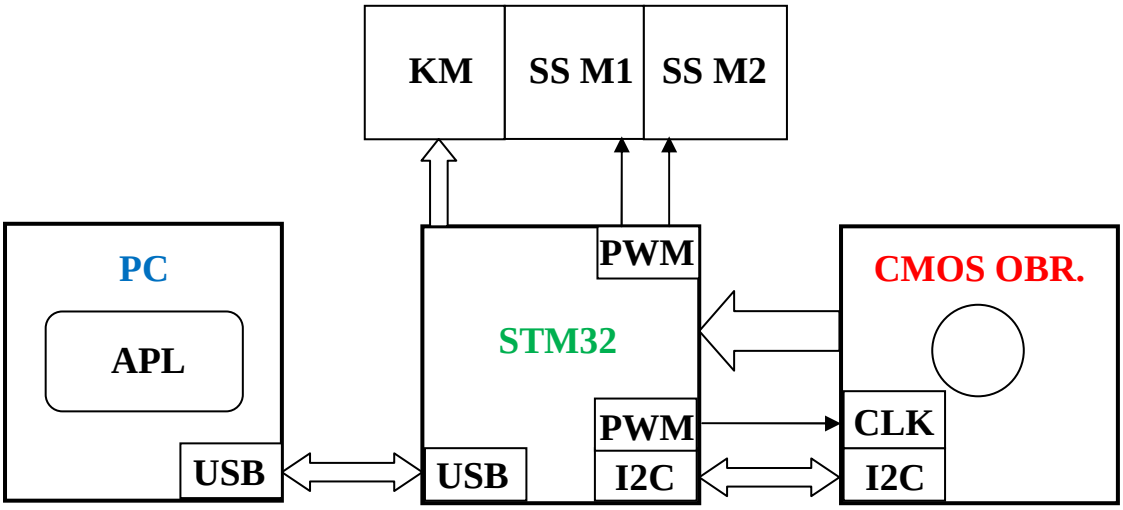
Rce. 3.14 – Rovnice závislost vstupního napětí motoru na úhlu, poloze

$$\frac{\Phi_P(s)}{U(s)} = \frac{1}{s\left(\frac{L \cdot J}{k \cdot \Phi} s^2 + \frac{R \cdot J}{k \cdot \Phi} s + k\Phi\right)}$$

Rce. 3.15 – Rovnice přenosu stejnosměrného motoru

4 OBVODOVÉ ŘEŠENÍ SNÍMAČE

Nezbytnou součástí každého elektronického zařízení je návrh kvalitního a spolehlivého hardwaru. Tento požadavek by měl být splněn i za předpokladu, kdy návrh disponuje minimálním množstvím součástek potřebných ke správné funkci celého výrobku. I tato práce se snaží docílit návrhu s minimálním množstvím součástek nezbytných k funkci a ochraně jednotlivých prvků. Základní komponenty tohoto návrhu jsou mikroprocesor, obrazový senzor a budiče pro výkonové součástky jako jsou motory.



Obr. 4.1 – Blokové uspořádání modulu

4.1 Deska vývojového modulu s STM32F105

Srdcem celého návrhu je mikroprocesor STM32F105. Je využito již navrženého modulu[1] původně určeného pro shodný mikroprocesor nižší řady STM32F103, který kromě zmíněného mikroprocesoru také obsahuje základní vybavení k jeho oživení, tlačítko resetu, bootování a rozhraní RS232 využitelného i k zavedení aplikačního programu do paměti. Deska modulu je osazena i jednou diodou LED, která je volně k dispozici pro uživatele. Vyvedeny jsou všechny piny pouzdra procesoru a jsou přizpůsobeny k připojení na desku nepájivého kontaktního pole.

4.1.1 Mikroprocesor STM32F105

Mikroprocesor STM32F105 byl v době návrhu vzhledem k dostupnosti a vybavenosti nejlepší volbou z rodiny STM32. Výkonnější a vybavenější nástupce, řada STM32F2xx, nebyl ještě

[1] Schéma zapojení desky modulu poskytuje *příloha A, modul s mikroprocesorem STM32 vznikl na katedře měření ČVUT FEL*

na trhu dostupný. Použitý mikroprocesor je produktem společnosti *STMicroelectronic*, který je postaven na 32-bitovém jádru *ARM Cortex-M3* a je vybaven velkým množstvím periferií vestavěných přímo na čipu. Shrnutí základních parametrů a výbavy obsahuje následující tabulka.

Parametr	Hodnota
Maximální frekvence CPU	72 MHz (1,25 DMIPS/MHz)
Paměť Flash	256 kB
Paměť SRAM	64 kB
Počet využitelných vývodů (GPIO)	51
Maximální frekvence změny na GPIO	18 MHz
Počet rozhraní USB 2.0	1 (variant Full-Speed OTG)
Počet čítačů/časovačů	4 (16-bitové, různé funkční módy)
Počet rozhraní USART	5
Počet rozhraní I2C	2
Počet rozhraní SPI	3
Počet rozhraní CAN	2
Počet převodníků A/D	2 (12-bitové s 16 kanály)
Počet převodníků D/A	2 (12-bitové s 2 kanály)
Rozsah napájecího napětí	2 - 3,6 V

Tab. 4.1 – Parametry mikroprocesoru STM32F105 (pouzdro LQFP64)

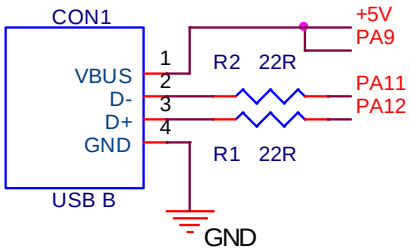
4.1.2 Řadič USB mikroprocesoru STM32

Hlavní komunikační a datový kanál pro připojení mikroprocesoru a PC je zvolena sběrnice USB. Mikroprocesory STM32 řady F103 a F105 (F107) podporují přímo na čipu periferii USB 2.0 ve specifikaci Full-Speed (tj. maximální teoretická přenosová rychlost 12 Mbit/s), ale liší se svoji strukturou a také svoji reálnou dostupnou přenosovou rychlostí.

Schéma připojení konektoru USB typu B k mikroprocesoru ukazuje *obr. 4.2*.

Nižší řada F103 má sice poměrně jednoduchou strukturu USB, ale velikost vysílaných dat je závislá na velikosti dostupné paměti pro jeden endpoint. Celková paměť, která je k dispozici pro všechny endpointy, je limitována maximální velikostí do 1,25 kB. Není zde již žádná pomocná větší bufferovací paměť.

Vyšší řada F105 obsahuje vylepšené USB s označením OTG (On The Go), které lze provozovat v režimu Device, ale i v režimu Host. Vysílání dat je již bufferováno pomocí rychlé FIFO fronty o velikosti 4kB pro každý výstupní endpoint (Endpoint IN – datový tok vzhledem k Host).



Obr. 4.2 – Schéma připojení konektoru USB B k mikroprocesoru STM32F105

4.2 obrazový senzor CMOS MT9M001

Ke snímání obrazové informace je využit monochromatický CMOS obrazový senzor od společnosti *Micron*. Opět bylo využito již hotové desky[2], která obsahuje vlastní senzor, patici se šroubením pro objektiv a přizpůsobené vývody ke snadnějšímu připojení k procesorové desce. Základní parametry senzoru shrnuje *tab. 4.2*.

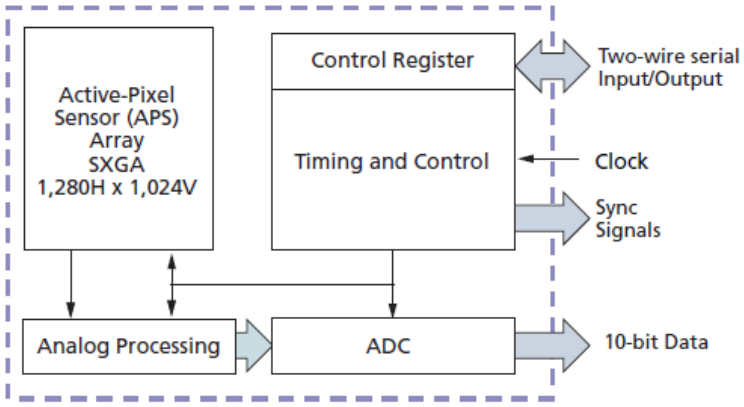
Senzor je svými datovými vývody připojen přímo na bránu mikroprocesoru. Vzhledem k rychlosti zpracování objemu dat je připojení pouze 8-bitové (D2-D10). Dále jsou připojeny signálové, synchronizační a napájecí vodiče. Senzor obsahuje skupinu konfiguračních registrů programovatelných přes dvouvodičovou sběrnici I2C. Na *obr. 4.3* je blokově znázorněna vnitřní struktura včetně vyvedených signálových a datových vodičů a na *obr. 4.4* schéma připojení k mikroprocesoru.

Parametr	Hodnota
Formát	½ palce (5:4)
Aktivní velikost čipu	6,66mm x 5.4mm
Aktivní počet pixelů	1280 x 1024 (H x V)
Velikost pixelu	5,2µm x 5,2µm
Typ závěrky	ERS (Electronics Rolling Shutter)
Maximální taktování	48 MHz
Maximální snímkování při 1280 x 1024	30 fps

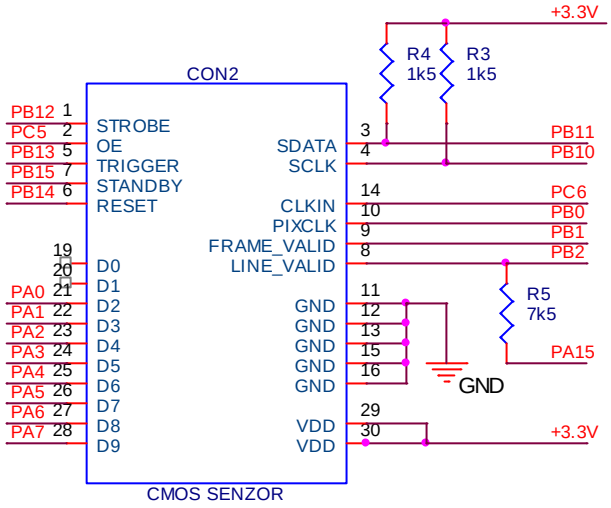
[2] *Deska modulu s obrazovým senzorem MT9M001 vznikla na katedře měření ČVUT FEL*

Rozlišení pixelu	10 bitů
Citlivost	2,1 V/lux-sec
Dynamický rozsah	62,2 dB
Maximální SRN	45 dB
Rozsah napájecího napětí	3 – 3,6 V

Tab. 4.2 – Parametry CMOS obrazového senzoru MT9M001



Obr. 4.3 – Blokové uspořádání obrazového senzoru MT9M001, převzato z [14]



Obr. 4.4 – Schéma připojení konektoru CMOS senzoru k mikroprocesoru STM32F105

Označení vodiče	Typ	Význam
CLKIN	Vstup	Taktovací signál senzoru
/OE	Vstup	Povolovací signál pro signály FRAME_VALID, LINE_VALID, PIXCLK a STROBE

/RESET	Vstup	Resetovací signál, nastaví registry senzoru na výchozí úroveň
SCLK	Vstup	Hodinový signál pro I2C
STANDBY	Vstup	Signál pro úsporný napájecí mód
TRIGGER	Vstup	Trigerovací signál pro Snapshot mód
SDATA	Vstup, Výstup	Datový vodič pro I2C
DOUT(0-9)	Výstup	Datové vodiče
FRAME_VALID	Výstup	Synchronizační signál snímku
LINE_VALID	Výstup	Synchronizační signál řádku
PIXCLK	Výstup	Synchronizační signál pixelu
STROBE	Výstup	Indikační signál pro režim Strobe

Tab. 4.3 – Seznam signálových a synchronizačních vodičů obrazového senzoru MT9M001

4.3 Výkonové prvky

Výkonové prvky jako jsou motory nelze, bez možného rizika zničení okolních citlivých součástek, přímo připojit na bránu mikroprocesoru. Při přímém připojení nemusí být schopna brána mikroprocesoru motor vybudit a hrozí zde také velká pravděpodobnost zničení vlivem vysokého špičkového napětí, protože motor je součástka ryze indukčního charakteru. Motory je tedy vhodné pro potřeby řízení mikroprocesorem připojit přes budiče a vybavit ochrannými rezistory a diodami, pokud ve struktuře budícího obvodu již nejsou obsaženy.

4.3.1 Připojení stejnosměrného motoru k mikroprocesoru

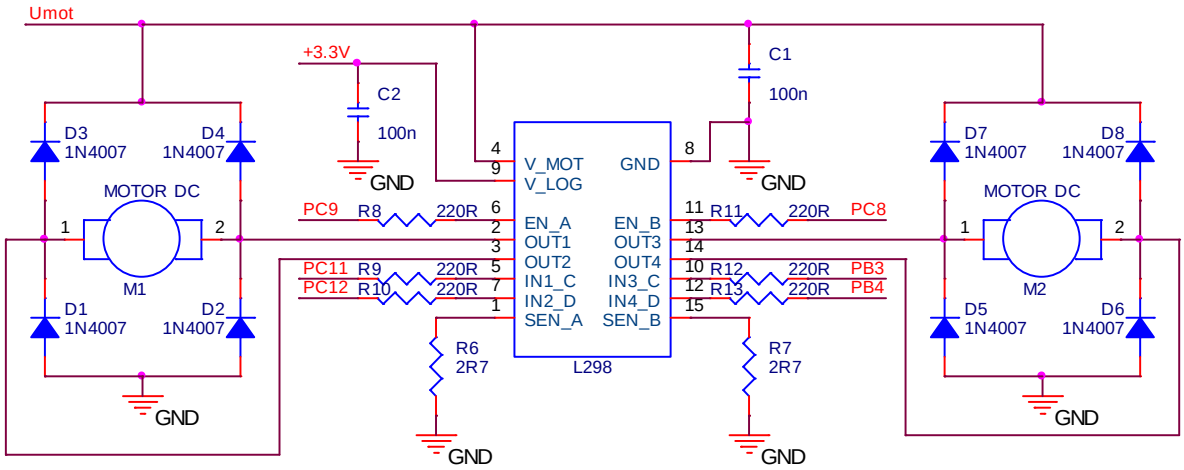
Z hlediska plného řízení stejnosměrného motoru se využívá obvodu *L298*, který obsahuje dva H-můstky a možnost připojení snímacích rezistorů monitorujících odebíraný proud motorem. Tento obvod lze trvale zatížit až 2A na každý můstek (při velké výkonové ztrátě je nutné obvod chladit).

Řídící signály		**Činnost motoru**
EN = 1	C = 1, D = 0	Otáčení
	C = 0, D = 1	Reverzované otáčení
	C = D	Rychlé zastavení
EN = 0	C = x, D = x	Volnoběžné otáčení či zastavení

Tab. 4.4 – Možnosti řízení stejnosměrného motoru obvodem L289

Kde: 0 – úroveň log. 0, 1 – úroveň log. 1, x – úroveň log. 0 nebo 1

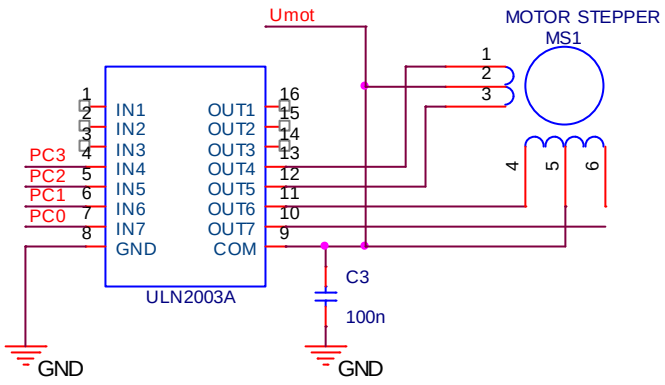
Vlastní řízení pak spočívá v přepínání signálů *C* a *D* (změna směru otáčení nebo zastavení) a rychlost otáčení motoru pak v periodickém spínání a změnou činitele plnění signálu *EN*, tj. řízení pomocí PWM. K mikroprocesoru jsou připojeny řídící signály *C, D, EN* přes ochranné rezistory.



Obr. 4.5 – Schéma připojení DC motorů k mikroprocesoru STM32F105

4.3.2 Připojení krokového motoru k mikroprocesoru

Při připojení krokového motoru z hlediska řízení je nutné dbát na propojení vinutí uvnitř motoru a odhadnout tak typ krokového motoru, zda je unipolární nebo bipolární. Od tohoto typu se odvíjí i způsob jeho řízení.



Obr. 4.6 – Schéma připojení krokového motoru k mikroprocesoru STM32F105

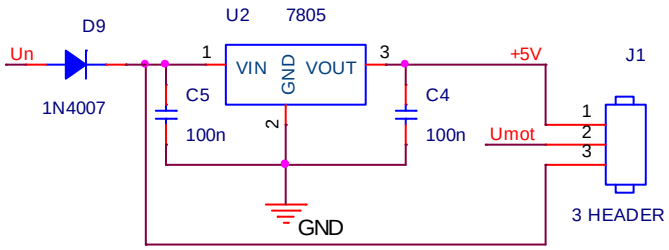
Z hlediska jednodušší obvodové struktury byl vybrán unipolární krokový motor, kde k řízení postačuje pouze spínání jedné polarity vinutí. Motor byl použit z vyřazené 5,25 palcové

disketové mechaniky s 200 kroky na otáčku (na jeden krok je změna úhlu o 1,8°, poloviční krok pak 0,9°). K jeho buzení je použit obvod *ULN2003A*, který obsahuje sedmici tranzistorových spínačů v darligtonově zapojení spolu s ochrannými diodami. Každý z těchto spínačů je schopen sepnou proud až 500mA.

Schéma připojení budiče krokového unipolárního motoru k mikroprocesoru ukazuje *obr. 4.6*.

4.4 Zdroje napájení desky modulu

Všechny jednotlivé prvky je možné napájet z externího zdroje o rozsahu napětí 6 až 12V. Toto přivedené napětí je přes ochrannou diodu přivedeno na stabilizátor 7805, kde je vytvořen zdroj napětí 5V pro napájení desky s mikroprocesorem a zároveň je propojeno s USB. K napájení motorů je možné pomocí propojky využít zdroje externího napětí anebo zdroje 5V na desce, které je ale omezeno výstupním proudem stabilizátoru do 1A a při současném napájení z USB i do 1,5A. Pro motory malých výkonů lze celou desku napájet pouze z USB, tedy celkový odebíraný proud nesmí překročit 0,5A. obrazový senzor je napájen z 3,3V, který dává k dispozici deska s mikroprocesorem.



Obr. 4.7 – Schéma napájení desky

Napětí motorů U_{mot}	Propojené vývody HEADERu
5V	1,2
6-12V	2,3

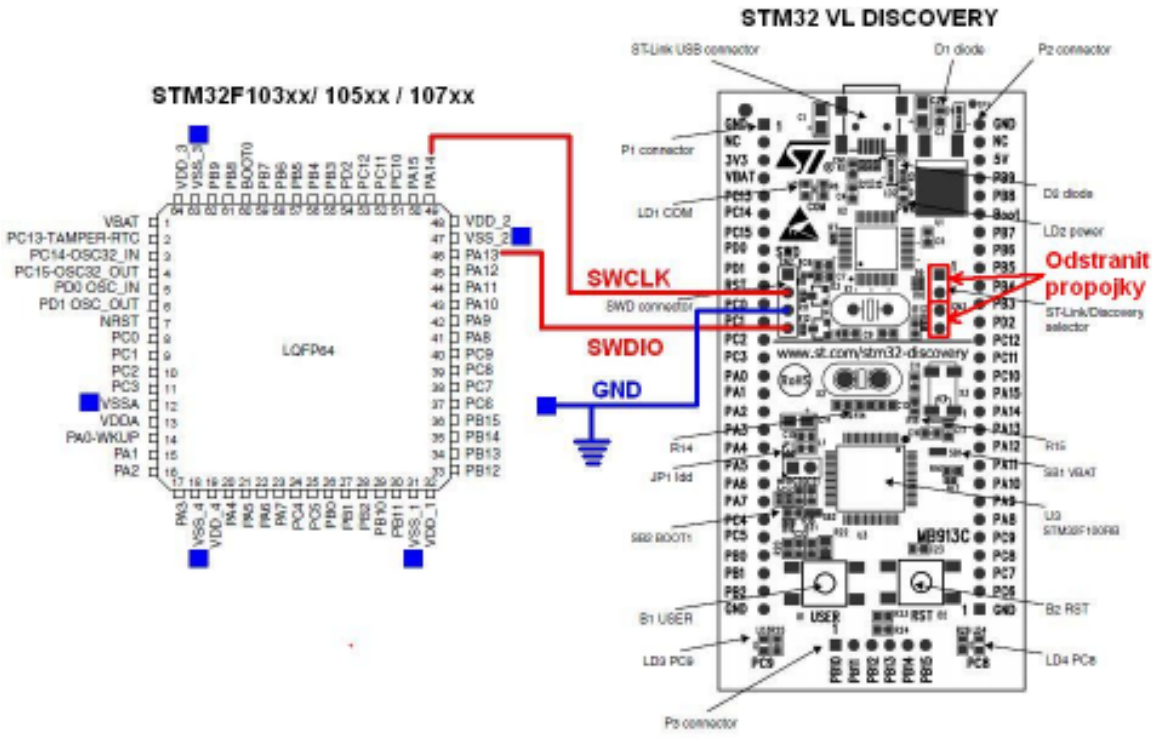
Tab. 4.5 – Propojení vývodů pro napájení motorů

4.5 STM32 VL DISCOVERY jako ladící modul mikroprocesoru

Mikroprocesory STM32 podporují přímé ladění aplikace na hardwaru a mají k tomuto účelu vyvedeny vodiče pro připojení ladícího rozhraní *JTAG* a také jeho jednodušší sériové variantě označované jako *SWD*. Tato ladící rozhraní však nelze provozovat bez příslušného softwarového a hardwarového vybavení.

Jednou možných z variant je využiti vývojového prostředí *Keil µVision* (profesionální volně dostupný nástroj omezený pouze velikostí kódu), kde je podpora obou těchto rozhraní. Pro propojení mezi tímto vývojovým prostředím a laděným hardwarem mikroprocesoru je použit „*Low Cost*" modul s označením *STM32 VL DISCOVERY*. Tato varianta podporuje ladící rozhraní *SWD* a připojení v *Keil µVision* jako *ST-Link*. Modul kromě tohoto použitelného „*ladítka*" obsahuje i jeden programovatelný mikroprocesor řady STM32F100.

Vlastní propojení s laděným mikroprocesorem ukazuje *obr. 4.8*. Pokud není laděným mikroprocesorem právě ten obsažený na modulu, lze po odstranění propojek *CN3* připojit pouze pomocí tří vodičů (*GND, SWCLK, SWDIO*) i jiný podporovaný mikroprocesor. Po realizaci tohoto propojení je možné přímo v prostředí *Keil µVision* příslušný procesor přes USB *Discovery Kitu* nejen ladit, ale také programovat.



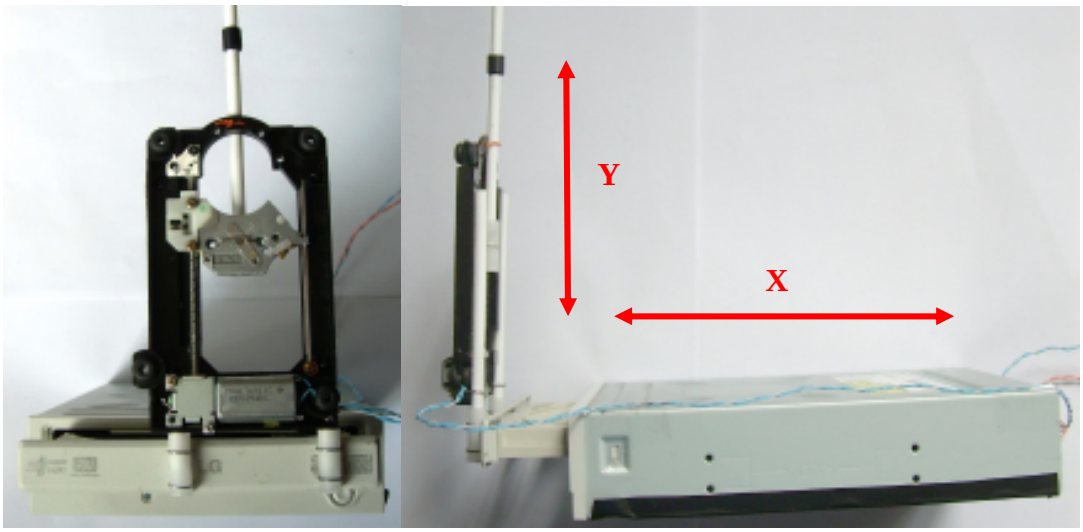
Obr. 4.8 – Připojení ladícího Kitu ST32 VL DISCOVERY, editováno z [10 a [11]

4.6 Konstrukce polohovacího mechanismus

K testovacím účelům pro potřeby regulace polohy byl sestaven dvouosý polohovací mechanismus. Ke konstrukci je využito pojezdů z vyřazených optických mechanik z běžného PC. Posuv je tak možný ve vodorovném směru, tj v ose *X*, a ve svislém směru, tj v ose *Y*. Obě osy je možné polohovat spojitě v určitém omezeném rozsahu. Ovládání vozíků je malými stejnosměrnými motorky. Parametry sestaveného polohovacího mechanismu jsou uvedeny v následující tabulce.

Parametr	Hodnota
Rozsah posuvu v ose X	133mm
Rozsah posuvu v ose Y	39mm
Rozsah napájecích napětí motorků	5-12V

Tab. 4.6 – Parametry polohovacího mechanismu



Obr. 4.9 – Foto sestaveného polohovacího mechanismu

5 ČTENÍ OBRAZOVÝCH DAT PROCESOREM

Nejdůležitější součástí snímače polohy je spolehlivé a přesné získání informace o aktuální pozici. Ke snímání polohy je v této práci využíváno informace z obrazového snímače, proto je zde silná závislost na pořízení rychlého a kvalitního obrazu. Vybavovací rychlost snímání obrazu je ovlivněna obrazovým senzorem, tak i řídícím mikroprocesorem.

5.1 Rychlost obrazového senzoru a mikroprocesoru

obrazový senzor *MT9M001* je schopen na svém výstupu dodat informaci o jednom pixelu s maximální frekvencí 48 MHz a toto odpovídá 30-ti snímkům za sekundu při rozlišení 1280x1024. Zachování takovéto rychlosti však závisí i na stejné rychlosti vyčítání těchto dat mikroprocesorem pro následné zpracování. Mikroprocesor STM32F105 není hardwarově přizpůsoben ke čtení dat přímo z obrazového senzoru. Toto umožňuje až vyšší řada STM32F2xx, která již obsahuje tzv. obrazový interface a data pak lze číst velmi rychle s využitím DMA přímo do paměti mikroprocesoru. U použitého mikroprocesoru nezbývá nic jiného, než čtení obrazových dat realizovat softwarově na úkor nižší přenosové rychlosti a vytíženosti výkonu mikroprocesoru. Omezujícím faktorem je hlavně maximální změna úrovně na bráně mikroprocesoru, kde i při taktování na frekvenci 72 MHz je na bráně přípustná změna pouze 18 MHz.

5.2 Softwarové čtení obrazových dat

Při hledání optimálního čtení obrazových dat je využito přístupů, kde je taktování obrazového senzoru realizováno pomocí PWM signálu generovaného čítačem mikroprocesoru v příslušném módu. Výhoda této metody je zejména snadná změna frekvence přetaktování (pouze v rozsahu dělícího poměru taktovací frekvence mikroprocesoru) a synchronizace signálů mezi oběma prvky.

Synchronizace hraje velmi významnou roli hlavně v přístupu, kdy jsou data ze senzoru vyčítána pouze na základě synchronizace signálů *FRAME_VALID* (synchronizace na snímek) a *LINE_VALID* (synchronizace na řádek). Signál *PIXCLK* (synchronizace na pixel) pak lze zanedbat za předpokladu vyčítání dat ve vhodnou dobu, tedy když jsou k dispozici platná data. Tento přístup tolik “nezdržuje” pomalý mikroprocesor při kontrole platnosti každého pixelu a celková frekvence vyčtení obrazu se tak může zrychlit. Naopak je však obtížnější zvolit vhodnou taktovací frekvenci obrazového senzoru a vhodný počet instrukcí tak, aby právě jednomu čtení dat z brány mikroprocesoru odpovídalo právě vyčtení jednoho pixelu. Cílem je zabezpečit spolehlivé souvislé softwarové čtení dat s největší možnou frekvencí.

5.3 Možnost optimalizace rutiny v jazyce assembler

Někdy je potřeba zajistit velmi rychlé a přesné vykonání části kódu jako je právě čtení obrazových dat. Rychlost kódu je závislá na jeho optimalizaci, tedy na délce trvání použité instrukce a na výsledném počtu prováděných instrukcí. Délka vykonávání jedné instrukce je závislá na typu operace a někdy souvisí s typem paměti, do které přistupuje (např. vkládání čekacích stavů k přístupu na flash paměť). Tímto faktem je ovlivněna i skutečnost, že maximální změna na výstupu brány mikroprocesoru STM32F1xx je 18 MHz, tj. zápis mikroprocesoru na bránu. Naopak maximální frekvence čtení brány a uložení dat do SRAM je limitováno frekvencí 12 MHz. Tato skutečnost je experimentálně změřena na základě testovacích dat, kdy byla příslušnou frekvencí čtena předem známá sekvence dat a následně ověřena jejich platnost.

Při programování kritických rychlých kódů v jazyce C je programátor závislý na kvalitě překladače zdrojového kódu pro danou platformu. Výsledný kód muže být velmi rychlý a ve většině případů i je, ale zároveň mohou nastat situace, kdy lze kód napsat, v rámci instrukční sady procesoru, rychleji a efektivněji. Na takovéto situace lze s výhodou využít hierarchicky nižšího programovacího jazyka assembler. Hlavní výhodou je plná kontrola nad vykonávanými instrukcemi a možnost zabezpečení lepší optimalizaci rutiny. V rozsáhlém programu je lepší používat jen tzv. vkládaný assembler nebo jen rutiny realizované jako podprogramy, protože asembler klade na programátora jednu velkou nevýhodu a to je správa paměti. Je tedy nutné i v podprogramech psaných v assembleru zohlednit, aby nedocházelo k přepsání jiných proměnných, než je nezbytné a nedošlo ke zhroucení celého kódu.

5.4 Metoda rychlého čtení obrazového řádku

V této metodě byl využit jazyk assembler, aby bylo zabezpečeno rychlé přečtení obrazové informace z brány mikroprocesoru a uložení do paměti. Bylo nutné uskutečnit skok po instrukcích realizující přenos do pamětí a přizpůsobivost vzhledem k proměnlivosti velikosti jednoho obrazového řádku.

Metoda spočívá v synchronizaci na signál *LINE_VALID* indikující, že jsou k dispozici data pro jeden platný řádek (signál *PIXCLK* je zanedbán). Zavoláním metody v C *Wait_LINE_VALID* jsou naplněny registry potřebnými hodnotami a následně se začne vykonávat podprogram v assembleru se stejným jménem. V tomto podprogramu je povoleno přerušení na signál *LINE_VALID* a dále se čeká na vlastní přerušení. V obsluze rutiny přerušení se nejdříve v závislosti na velikosti čteného řádku vypočte adresa (vzhledem k stavu programového čítače), od které se budou zaznamenávat data do paměti. Následně je vykonán

skok na tuto adresu a záznam dat. Princip záznamu je pouze prosté necyklické přečtení hodnoty na bráně mikroprocesoru a uložení do paměti. Toto je možné si dovolit pouze za předpokladu, kdy při každém následném čtení jsou k dispozici data právě následujícího pixelu a zároveň délka čtených dat odpovídá právě počtu instrukcí realizujících čtení a zápis do paměti. Když je tento předpoklad splněn, je při vyčtení posledního pixelu v řádku ukončeno i vyčítání z brány a obsluha přerušení je po zakázání přerušení na signál *LINE_VALID* dokončena.

Před každým zavolání této metody čtení obrazového řádku, musí být zakázána veškerá ostatní přerušení, jinak hrozí pád programu. Důvodem je ztráta návratové adresy z obsluhy přerušení.

Metoda v C volající podprogram v assembleru:

```
Wait_LINE_VALID(&GPIOA->IDR, &Buffer[index], offset,  &EXTI_LINE_VALID);
```

- & GPIOA->IDR – adresa brány = R0
- &Buffer[index] - adresa paměti pro uložení dat = R1
- offset – hodnota indikující velikost skoku v závislosti na velikosti čteného řádku = R2
- &EXTI_LINE_VALID – adresa registru pro povolení/zakázání přerušení = R3

Podprogram v assembleru čekající na přerušení na signál LINE_VALID:

```
Wait_LINE_VALID          ; návěští podprogramu čekající na LINE_VALID
        PUSH  {R4, LR}   ; záloha registru R4 a návratové adresy do zásobníku
        MOV   R4, #0x4   ; povolení přerušení od LINE_VALID, R4=hodnota,
                         ; R3 = adresa
        STR   R4, [R3]
        WFI              ; čekaní na přerušení od LINE_VALID
        POP   {R4, PC}   ; obnova registru R4 a návratové adresy ze zásobníku
```

Obsluha přerušení v assembleru při detekci signálu LINE_VALID:

```
LINE_VALID_IRQHandler    ; návěští přerušení
        ADD   R2, R15, R2    ; součet R2 (velikost skoku) a R15 (hodnota prog. čítače)
        BX    R2             ; skok na výslednou adresu v R2
        LDRB  R2, [R0]       ; čtení bajtu z adresy v R0 (brána) do R2
        STRB  R2, [R1, #32]  ; uložení hodnoty v R2 do paměti na adrese v R1 + offset (32)
        LDRB  R2, [R0]
        STRB  R2, [R1, #33]
        :
        LDRB  R2, [R0]
```

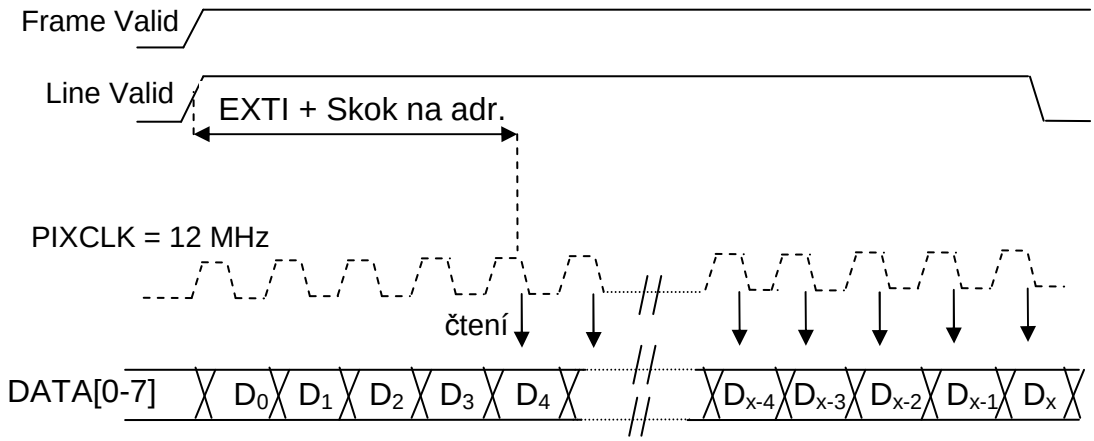
```
STRB   R2, [R1, #1310]
LDRB  R2, [R0]
STRB   R2, [R1, #1311]
MOV     R2,#0x0          ; zakázání přerušení od LINE_VALID
STR      R2, [R3]
BX      LR                ; návrat z přerušení
```

Realizace takovéto rutiny se jen v C kódu nepodařilo uskutečnit. Příčinou selhání bylo zejména nepředvídatelná délka instrukce zápisu (STRB) dat do paměti. Překladač tyto instrukce realizoval s proměnlivou délkou v paměti (lepší optimalizace kódu a využití paměti), a tak nebylo možné přesně skákat na požadované adresy instrukcí zabezpečující čtení dat. Kód napsaný v assembleru má stanovenu pevnou délku instrukce o velikosti 4 bajty, a tím lze bezpečně docílit skoku v programu na požadovanou instrukci.

Příklad proměnlivé délky instrukce STRB:

```
STRB   R1, [R2, #X];  kde X  > 31 -  obsadí v paměti 4 bajty
STRB   R1, [R2, #X];  kde X = 0 až 31 - obsadí v paměti 2 bajty
```

Tato metoda umožňuje vyčítání dat z obrazového senzoru o frekvenci 12 MHz a zároveň je obrazový senzor taktován PWM signálem o shodné frekvenci. Na úkor této rychlosti jsou zatraceny první čtyři pixely z každého čteného řádku, což je právě doba potřebná k výpočtu adresy skoku a vlastního skoku na požadovanou instrukci. Jelikož tato ztráta je vždy konstantní, lze ji kompenzovat zvětšením velikosti vyčítaného řádku o právě čtyři pixely (sloupce), které představují tzv. černé pixely.



Obr. 5.1 – Diagram čtení obrazových dat při ztrátě prvních 4 pixelů

5.5 Alternativní metody čtení obrazového řádku

V průběhu vývoje v oblasti softwarového čtení obrazových dat ze senzoru bylo odzkoušeno několik přístupů, které vždy nevedly ke spolehlivému řešení. Úspěšnou rychlou metodu popisuje předešlá kapitola.

První pokusy vedly na zcela běžný způsob, kdy se procesor synchronizuje na všechny signály zabezpečující platné přečtení dat, tedy na *LINE_VALID* a *PIXCLK*. Procesor však „nedokázal zvládnout“ data číst uspokojivou rychlostí a frekvence čtení platných dat se pohybovala kolem 2MHz. Ztrátu podstatné rychlosti představovala právě neustálá kontrola signálu *PIXCLK*. Signál tedy musel být zanedbán.

Další pokusy čtení dat z brány byly s ohledem na synchronizaci na signál *LINE_VALID* a na zvolení vhodného taktování senzoru tak, aby byla data platně vyčtena. Současně muselo být zohledněno možné nastavení taktování obrazového senzoru. Dostupné frekvence taktování jsou uvedeny v *tab. 5.1,* která uvádí dělící poměry vzhledem k taktování procesoru při 72MHz.

DIV	1	2	3	4	5	6	7	8	9	10	11
f [MHz]	72	36	24	18	14,4	12	10,29	9	8	7,2	6,55
DIV	12	13	14	15	16	17	18	19	20	21	22
f [MHz]	6	5,54	5,14	4,8	4,5	4,24	4	3,79	3,6	3,43	3,27

Tab. 5.1 – Dostupné frekvence taktování odvozené od 72 MHz

Následující příklady ukazují možné řešení spolehlivého čtení dat v cyklu.

Příklad čtení dat v C kódu:

```
do{
        Buffer [index++] = (GPIOA->IDR );
}while(LINE_VALID);
```

Dosažená frekvence čtení platných dat je 3,79MHz, kde překladač použil v cyklu 7 instrukcí. Optimalizací stejného algoritmu v asembleru se frekvence zvýšila na 4MHz, podařilo se snížit počet instrukcí na 6.

Vylepšené řešení spočívalo v začlenění přerušení na signál LINE_VALID a vhodnou úpravou editovat a testovat příznak pro rozhodnutí instrukcí skoku v asembleru. V kódu je generováno přerušení, které v obsluze změní příznak Z (indicator nulového výsledku operace), tím je pak ukončeno vyčítání dat. Přerušení je vygenerováno při sestupné hraně LINE VALID.

Příklad čtení dat v asembleru s využitím přerušení na Line_Valid:

```
L1      LDRB  R1, [R0]       ; R0 = adresa brány (GPIOB->IDR)
        STRB  R1, [R2]       ; R2 = adresa pole (buffer_CMOS_DATA)
        ADD   R2, R2, #1     ; inkrementace v poli
        BNE   L1             ; skok na návěští L1 (skok při příznaku Z = 0),
                             ; při návratu z přerušení je Z = 1
```

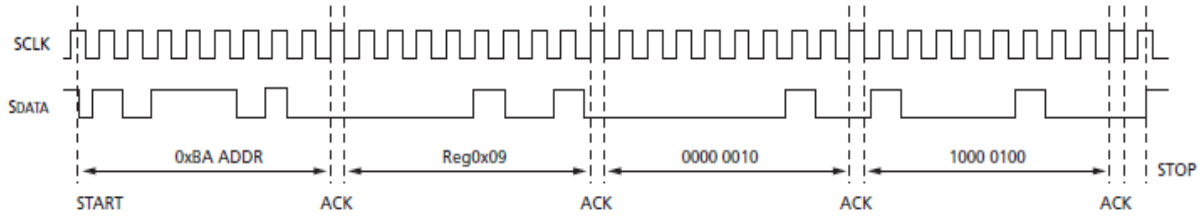
Dosažená frekvence čtení je 6,55MHz a počet instrukcí v kritické části kódu byl snížen na 4.

Poslední metodou, jak realizovat čtení dat z brány procesoru, byla myšlenka použití kanálu DMA, která nezávisle na procesoru bude číst data a ukládat do paměti. Toto řešení by odlehčilo práci procesoru a celý proces zpracování i transportu dat by se mohl zrychlit. Při dostupném sortimentu frekvencí však data nebyla nikdy správně vyčtena. Na frekvenci 7,2MHz bylo přečteno více dat, než bylo požadováno a na frekvenci 8MHz bylo naopak přečteno méně dat. Z dostupných naměřených dat by pro platná data byla vyžadována frekvence přibližně 7,78MHz, která však mikroprocesorem nelze nastavit. Tento přístup tak byl také zavrhnut vzhledem k obtížné konstrukci žádané frekvence a zároveň možné ztrátě fázová synchronizace s mikroprocesorem.

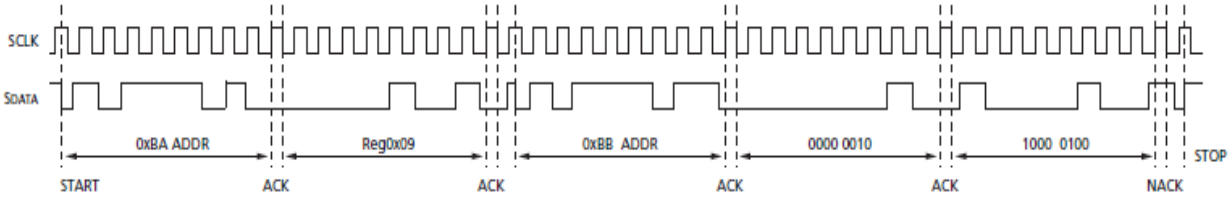
5.6 Konfigurace obrazového senzoru přes I2C

obrazový senzor obsahuje dvouvodičové konfigurační rozhraní na principu I2C (Inter-Integrated Circuit), přes které se nastavují registry pro různé módy a parametry senzoru. Jeden vodič slouží jako datový (*SDA*) a druhý udává frekvenci změny platných dat (*SCLK*). Příkladem konfigurace senzoru je změna rozlišení či výřezu obrazu, zesílení, doba závěrky a další. Obdobný přístup umožňuje většina podobných senzorů, ale i spousta jiných zařízení. Řídící mikroprocesor disponuje ve své struktuře hned dvěma programovatelnými rozhraními I2C, které lze nezávisle provozovat v režimu master i slave. Maximální komunikační rychlost dosahuje 400kB.

V implementaci je mikroprocesor nastaven v režimu master a senzor je typu slave. Navázání komunikace spočívá ve vyslání podmínky start a následně 7-bitové adresy a jednoho bitu indikující čtení nebo zápis. Adresa senzoru je pevně stanovena na hodnotu 186. Po adresaci jsou již vysílána data v podobě 8-bitové adresy registru v obrazovém senzoru a 16-bitových dat. Ukončení vysílání je indikováno podmínkou stop. Po odeslání či přijetí každého bajtu master potvrzuje data. Řízení a časování sběrnice obstarává hardware mikroprocesoru.



Obr. 5.2 – Příklad zápisu hodnoty 0x0284 do registru 0x09 přes I2C, převzato z [14]



Obr. 5.3 – Příklad čtení hodnoty (0x0284) z registru 0x09 přes I2C, převzato z [14]

5.7 Čtení snímku z obrazového senzoru

Objem dat jednoho snímku s plným, ale i redukovaný rozlišením téměř vždy přesahuje paměťový prostor mikroprocesoru STM32F105. Mikroprocesor má k dispozici paměť SRAM o velikosti 64kB a obrazový senzor v plném rozlišení zaujme paměťový prostor 1,25MB. Je proto nutné data číst po blocích, zpracovat je nebo odeslat například do PC a následně přečíst jiný ještě nepřečtený blok dat a totéž opakovat dokud není získán kompletní snímek.

V paměti mikroprocesoru je pro obrazová data rezervován prostor o velikosti 46 řádků při velikosti jednoho řádku 1280 pixelů, tj. 57,5kB. Zbylý paměťový prostor je rezervován pro potřeby programu a jiných dat.

Rozlišení	Max. počet řádků v bloku	Max. velikost bloku	Počet bloků na snímek
1280x1024	46	57,5kB	23
640x512	92	57,5kB	6
320x256	184	57,5kB	2
160x128	128	20kB	1

Tab. 5.2 – Využití paměťového prostoru v mikroprocesoru pro obrazová data

Jelikož mikroprocesor může být vytížen i jinou úlohou než čtením obrazových dat, je použit čítač mikroprocesoru svázaný se signálem *LINE_VALID*, který neustále aktualizuje informaci o číslu řádku na výstupu senzoru. Čítač je při prvním čtení obrazových dat synchronizován signálem *FRAME_VALID* a při jeho konfiguraci je nastaveno automatické nulování při hodnotě vyčtení dat posledního řádku. Jelikož si od senzoru nelze vyžádat pouze potřebný blok dat, musí se pokaždé čekat na požadované číslo řádku, a tak jeden kompletní snímek lze získat v průběhu několika snímků realizovaných senzorem.

Aby vyčítání dat bylo co nejefektivnější, byla vyvinuta metoda, která vyčítá aktuálně nejbližší blok dat ve snímku.

Princip spočívá v rozdělení snímku na bloky a vygenerování seznamu s adresami každého bloku. Jako první se vždy čte blok s adresou 0 (počátek řádku číslo 0) a po jeho zpracování, případně odeslání do PC je čten nejbližší blok s nejbližší vyšší adresou. V případě, kdy není aktuálně k dispozici nejbližší blok s vyšší adresou, je naopak čten nejbližší nepřečtený blok s nejnižší adresou. Při každém výběru je blok označen v seznamu jako přečtený až do okamžiku vyčtení posledního bloku dat, potom je seznam obnoven a vyčítání se opakuje. Metoda má smysl při větším počtu bloků ve snímku, kdy je reálné vyčíst více bloků v jediném snímku realizovaném senzorem. Možný sled vyčítání bloků dat ze senzoru popisuje následující obrázek.

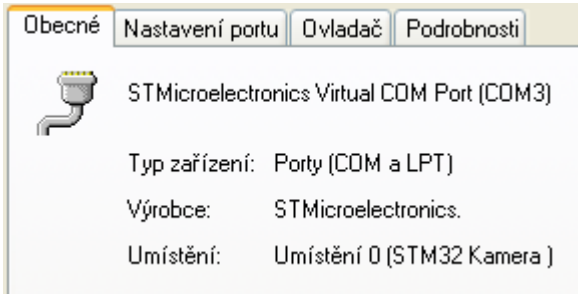
Obr. 5.4 – Příklad čtení obrazových dat ze senzoru

6 PŘENOS DAT PROSTŘEDNICTVÍM ROZHRANÍ USB

Jedno z velmi používaných možností připojení periferií k počítači je USB (Universal Serial Bus). V porovnání s klasickým sériovým připojením RS232 jde o rychlou sběrnici pro přenos dat. Jak již bylo zmíněno v *kapitole 4.1.2*, mikroprocesor STM32F105 je vybaven na čipu USB 2.0 ve specifikaci Full-Speed a pro rychlý přenos bloku dat využívá FIFO fronty o velikosti 4kB. Tato periferie vyžaduje poměrně složitou konfiguraci a obsluhu, ale programátor tak má poměrně velké možnosti nastavení vzhledem k vyžadované funkci.

6.1 Konfigurace a komunikace rozhraní USB

Vzhledem ke složitosti této periferie je využito příkladů nastavení od výrobce, který má zdrojový kód volně k dispozici. Konkrétně je modifikován projekt s názvem *Audio_Streaming*, který je přímo napsán pro mikroprocesor STM32F105 (F107) a je do značné míry upraven podle projektu s názvem *Virtual_COM_Port*, který byl původně napsán pro řadu STM32F103. Kód programu je upraven tak, aby USB mikroprocesoru bylo schopné přijímat a odesílat data v co nejkratším čase. Redukovány jsou nadbytečné rutiny volající prázdné nebo pro danou funkci nepotřebná těla metod a nevýznamné proměnné. Hlavním cílem je vytvořit jednoduchou periferii, kterou lze bez problémů připojit k PC a jednoduše přes ni se standardními prostředky komunikovat.



Obr. 6.1 – Identifikace USB v režimu Virtual COM Port

Komunikace po USB probíhá na základě dotazování zařízení typu Host na zařízení typu Device. V tomto případě je Host zařízení PC a Device mikroprocesor STM32. Aby se toto zařízení hlásilo v PC jako Virtual COM Port, musí být nakonfigurovány příslušné komunikační roury s koncovými body tzv. endpointy.

- **Endpoint 0** – Používá ho každé USB zařízení a je definován pro přenos v režim Control, který slouží právě k počáteční konfigurace a identifikaci USB.

- **Endpoint 1** – Tento koncový bod je provozován v režimu Bulk a slouží k hlavní datové komunikaci ve směru od Device k Host.
- **Endpoint 2** – Je nastaven v přenosovém režimu Interrupt, ale v této aplikaci se nevyužívá, je jen nutností k identifikaci ovladače.
- **Endpoint 3** – Slouží pro datovou komunikaci ve směru od Host k Device v režimu Bulk.

Identifikace a veškerá obsluha příjmu dat posílaných z PC vyvolává přerušení, ve kterém je dekódován komunikační endpoint a podle toho jsou následně spravována data. Jde-li o komunikaci USB přes endpoint 0, probíhá vysílání a příjem tzv. descriptorů, na které se Host postupně dotazuje a tím se blíže specifikuje dané zařízení.

Existuje několik základních USB descriptorů:

- **Device Descripror** – základní informace o specifikaci a rychlosti, atd.,
- **Configuration Descriptor** – blíže specifikuje typ a požadavky zařízení,
- **Interface Descriptor** – určuje typ, druh rozhraní,
- **Endpoint Descripror** – charakterizuje význam endpointu,
- **String Descriptor** – popisovače pro PC.

Více informací o specifikaci konfigurace standardu USB 2.0 přináší [15] nebo [16]. Ovladač zařízení se přiděluje hlavně na základě položek *idVendor: 0x0483* a *idProduct: 0x5740* v *Device Descriptoru*.

Nakonfigurované USB se tváří jako Virtual COM Port od společnosti *STMicroelectronics*, viz *obr. 6.1*. Toto řešení má jednu velkou výhodu, protože se využívá standardních digitálně podepsaných ovladačů pro tento typ zařízení a uživatel se může bez problémů připojit pod OS Windows XP a vyšší. Potřebný ovladač je většinou součástí Windows a uživatel nemusí zařízení instalovat.

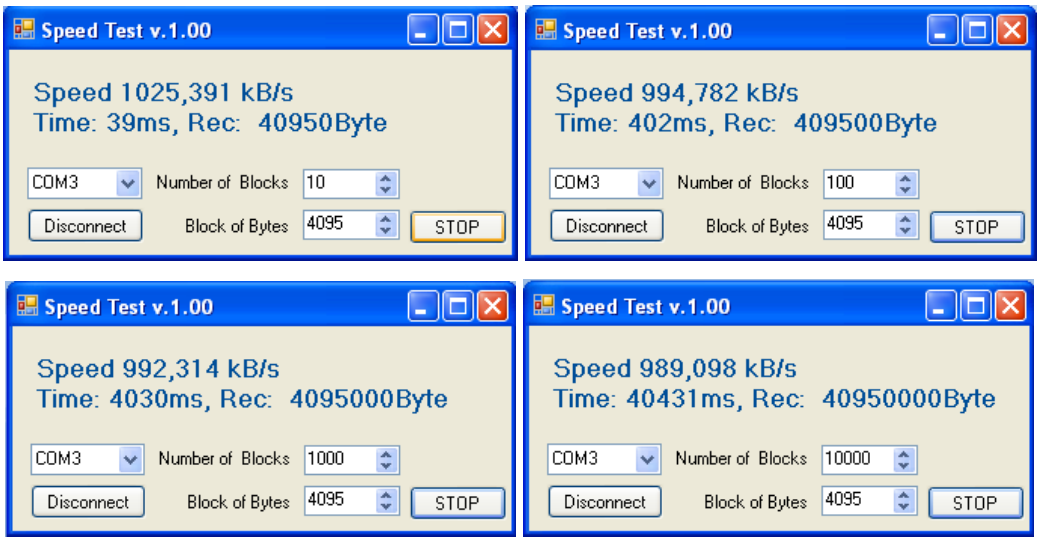
Při rozpoznání endpointu, který slouží pro přenos již uživatelských dat z PC k mikroprocesoru je po vykonání obsluhy přerušení indikován nový požadavek a po jeho dekódování je zpracován, případně jsou zpětně vyslány požadovaná data.

6.2 Rychlost datové komunikace

Je očekáváno, že reálná komunikační rychlost bude nižší, než je teoretická, která je dána příslušnou specifikací. Konkrétně je tedy očekáván datový přenos nižší než 12 Mbit/s. Na tento pokles má nejvíce vliv realizace režie pro přípravu a odeslání dat po USB. Na vkládání určitých zpoždění může mít vliv jak zařízení typu Device, tak i Host. Závisí tedy

na vybavovací rychlosti, jakou Host vyšle požadavek na data nebo vlastní data a na prodlevě, která nastane mezi příjmem požadavku nebo dat na straně Device.

Protože v konkrétním řešení tohoto USB bude největší objem dat přenášen ve směru k PC, tedy přenos obrazových dat, byla vytvořena aplikace testující rychlost v tomto směru přenosu viz *obr. 6.2*. Program je navržen s ohledem na minimalizaci režie zpracování požadavku na data a hned po jeho příjmu je vyslán žádaný objem dat.



Obr. 6.2 – Aplikace Speed Test na přenosovou rychlost USB

Jelikož je k dispozici pouze FIFO fronta o velikosti 4kB, je výhodné tuto velikost v rámci jednoho požadavku nepřekračovat a spíše data rozložit do více požadavků od Host. Zároveň je žádoucí se snažit využít co největší kapacitu této fronty. Za těchto téměř ideálních podmínek bylo dosaženo reálné přenosové rychlosti USB kolem 1MB/s.

6.3 Rychlost přenosu obrazových dat

K tomu aby byl přenesen obraz v reálném čase, tj. kdy lidské oko není schopno rozlišit patrné zpoždění mezi jednotlivými snímky, je nutno dosáhnout určité frekvence snímkování, tedy alespoň 25 snímků za sekundu (FPS). obrazový senzor je schopen dosáhnout v plném rozlišení rychlosti až 30 FPS a v ideálním případě lze mikroprocesorem takovýto snímek vyčíst rychlostí přibližně 9 FPS s ohledem na frekvenci vyčítání 12 MHz. Hlavním omezujícím faktorem je tedy přenosová rychlost po USB, kdy i při teoretické rychlosti 1,5MB/s by trvalo přenesení jednoho snímku v plném rozlišení přibližně 0,84 sekundy, což odpovídá 1,19 FPS.

V reálném případě je však dosaženo ještě mnohem nižších rychlostí. Výrazný vliv na zpoždění má zejména režie obsluhy USB a malá paměť procesoru, kdy je nutné jeden

snímek číst ze senzoru po blocích. Nepatrné zpoždění je i při vizualizaci obrazových dat. Dosažené frekvence snímkování při různém nastavení rozlišení senzoru ukazuje *tab. 6.1*.

Rozlišení	FPS	Délka závěrky
1280x1024	0,57	256
	0,57	1
640x512	2,11	256
	2,11	1
320x256	9,52	256
	9,52	1
160x128	30,30	256
	37,03	1

Tab. 6.1 – Dosažené rychlosti snímkování při různých rozlišení obrazu

7 ŘÁDKOVÉ SNÍMÁNÍ POLOHY A JEJÍ REGULACE

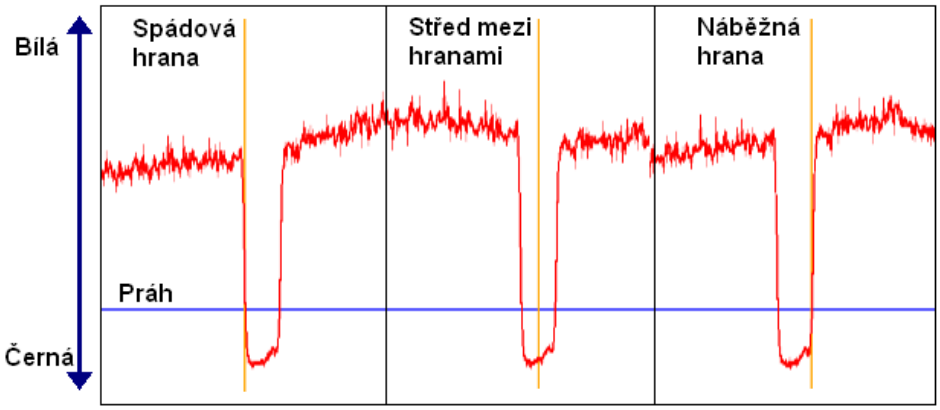
Smyslem snímání jednorozměrné polohy a její regulace v jedné ose je docílit změření požadované pozice a nastavením polohovacího mechanismu pouze pohybem jednoho motoru na základě obrazové informace ze senzoru (kamery). Jako základní osa je zvolena osa X, tj. snímání a případný posuv ve vodorovném směru. Je však možné zvolit jakoukoli jinou osu.

7.1 Řádkové zpracování obrazu

K určení nějaké pozice pomocí obrazového senzoru je nutné mít k dispozici spolehlivě detekovatelnou značku, nejlépe tedy takový bod, který je v zorném poli senzoru unikátní. Tento bod může být například pixel nebo shluk pixelů s minimální nebo maximální hodnotou jasu. U monochromatického senzoru je ideální případ výrazně světlý nebo tmavý bod.

Jelikož jde pouze o snímání v jediné ose, lze plošný obrazový senzor degradovat na řádkový. Význačný bod může potom ve skutečnosti být i „tyčka“ orientována kolmo na snímaný řádek, ve kterém se projeví jako „bod“. Zpracování takovéto obrazové informace pak není příliš výpočetně náročné a tedy i rychle opakovatelné.

Ze senzoru je vyčítán neustále jeden řádek o maximálním rozlišení 1280 pixelů pro dosažení co nejlepší citlivosti pro danou pozici značky. Senzor je nakonfigurován tak, aby bylo vytvořeno okno o velikosti jednoho řádku umístěného ve středu aktivní plochy čipu, tj. řádek číslo 511 (rozsah řádků 0 až 1023). Protože velikost značky je zpravidla vždy více pixelů, hledá se v obraze daná náběžná hrana, spádová hrana nebo střed mezi oběma hranami.



Obr. 7.1 – Nalezení významného bodu v jasovém profilu obrazového řádku

Algoritmus takového vyhledávání porovnává pixely, jejichž hodnota je větší nebo menší než zvolený práh. Náběžná či spádová hrana je určena mezi sousedními pixely, kde jeden má hodnotu větší a druhý menší než je prahovací úroveň. Pro nalezení středu mezi hranami je

pouze vypočten průměr nad souřadnicemi po sobě jdoucích pixelů vyhovující úrovni pro daný práh. Pro dosažení přesnější hodnoty je aplikována lineární interpolace hodnoty mezi dvěma pixely.

Mikroprocesorem je realizována výpočetní přesnost na 0,01 pixelu, ale při statickém snímání polohy objektu změřená hodnota kolísá v rozsahu do 0,1 pixelu. Toto může být způsobeno šumem obrazového senzoru (způsobený například kolísáním teploty v místnosti).

Takovéto zpracování a určení pozice v obraze je jednoduché pouze v ideálním případě, kdy se vyskytuje pouze jediný významný bod. V opačném případě je situace složitější, proto je pro jednoduchost vhodné zabezpečit ideální podmínky pro rychlé zpracování obrazu.

7.2 Časování rutin

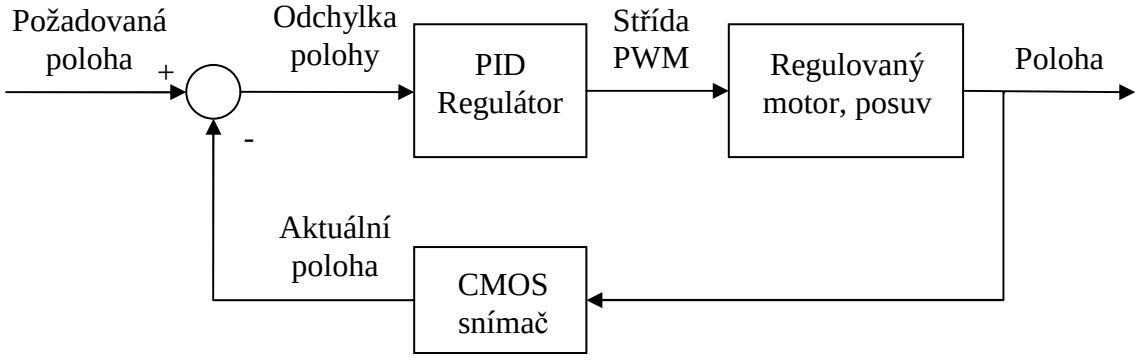
Pro některé operace je nutné zabezpečit spolehlivé opakování vykonávané rutiny, aby nedocházelo k jejich nepředvídatelnému chování, nebo je časování potřebné k jeho požadované funkci. V regulační smyčce je žádoucí spolehlivě opakovat regulační zásah, protože při nedodržení by docházelo k častému přeregulování žádané hodnoty. Časování je potřebné i v řízení krokového motoru, kdy při dodržení konstantní doby prodlevy mezi jednotlivými kroky je docíleno plynulého otáčení motoru. U stejnosměrného motoru je někdy vyžadováno spuštění jeho chodu jen na určitou dobu.

Aby mohly být zabezpečeny všechny tyto funkce, je v mikroprocesoru nakonfigurován systém reálného času (RTC), od kterého jsou odvozovány časově náročné operace. Tento systém se ve své podstatě chová jako 32-bitový čítač, který však oproti běžným 16-bitovým čítačům v mikroprocesoru dokáže zabezpečit dlouhodobější časování bez přetečení svého rozsahu a je k tomuto účelu přímo navržen. Dalším důvodem je i nedostatečné množství čítačů, které jsou rezervovány pro jiné operace. Taktování RTC je odvozeno od 8MHz krystalu umístěného na desce mikroprocesoru. Nakonfigurováním děličky je docílena inkrementace každou milisekundu, což je uspokojivé rozlišení pro operace v rámci navrženého modulu.

Původním alternativním řešením bylo využití tzv. jednotky *SysTick*, která slouží ještě k přesnějšímu časování, ale neumožňuje zařazení děličky pro prodloužení čítací doby a indikace přetečení jeho hodnoty přes přerušení byla nežádoucí. Maximální doba bez přetečení hodnoty registru *SysTick* je přibližně 1,8 sekundy, proti tomu registr systému RTC nepřeteče v řádu několik desítek dnů s nastaveným inkrementem jedné milisekundy.

7.3 Regulace polohy stejnosměrným motorem

K dosažení požadované polohy se používá regulační myčka se zpětnou vazbou. Takovéto řešení je výhodné, protože je neustále sledována odchylka od požadované hodnoty polohy a celý systém tak může reagovat odpovídající změnou akční veličiny. Smyčka se skládá ze základních bloků podle *obr. 7.2*. Vstupem do této smyčky je požadovaná hodnota akční veličiny, kterou je v tomto případě poloha, tj. pozice pixelu v zorném poli obrazového senzoru. Dále následuje sumační člen, kde je stanovena odchylka od požadované polohy. V bloku regulátoru je z odchylky vypočtena akční veličina pro regulovanou soustavu, kterou zde představuje stejnosměrný motor (jeden z motorů v polohovacím mechanismu z *obr. 4.9*), který ovládá pohyb vozíku, a tím se na výstupu nastaví poloha. Velikost odchylky polohy se projeví změnou střídy PWM signálů ovládající motor a zároveň podle orientace odchylky (kladná nebo záporná odchylka) lze určit směr otáčení motoru. Zpětnou vazbu uzavírá snímač v podobě obrazového senzoru, kde jeho hlavním úkolem je zjistit aktuální nastavenou polohu, která je odečtena od požadované a celý cyklus se opět opakuje.



Obr. 7.2 – Regulační smyčka pro jednu osu souřadnic se stejnosměrným motorem

7.3.1 Realizace PSD regulátoru

Hlavním úkolem regulátoru je vhodně zareagovat na změnu požadované veličiny v návaznosti na orientaci a velikosti regulační odchylky polohy. Protože k řízení je primárně určen mikroprocesor, musí být regulátor implementován programově. V mikroprocesoru je naprogramován PSD regulátor podle *rce. 7.1*. Jednotlivé složky mají vliv na výslednou hodnotu akční veličiny ve formě střídy PWM. Výpočet hodnoty PWM a zároveň akční zásah je dán periodou vzorkování v řádu milisekund. Minimální vzorkovací perioda je přibližně 3 milisekundy (doba trvání průběhu regulační smyčky).

Vlastní perioda vzorkování je však nejvíce ovlivněna obrazovým senzorem a závisí na jeho nastavené době expozice, proto je vhodné snímanou scénu co nejlépe nasvítit, aby mohla být

expoziční doba co nejkratší a tím i perioda vzorkování. Pro dodržení průchodu regulační smyčkou v čase 3 milisekundy, musí být délka elektronické závěrky senzoru nastavena na hodnotu do 28. Při nedodržení této hodnoty se perioda vzorkování prodlužuje. I přesto však vzorkovací perioda může kolísat v řádu desetin až jednotek milisekund v závislosti na synchronizaci na platná obrazová data.

$$e = setX - X$$
$$suma = suma + e$$
$$PWM = P \cdot e + S \cdot suma + D \cdot (e - eLast)$$
$$eLast = e$$

Rce. 7.1 – Implementace PSD regulátoru

Kde: setX – požadovaná poloha, X – aktuální poloha, e – regulační odchylka, eLast – regulační odchylka z předchozího kroku, suma – hodnota růstu regulační odchylky, P – zesílení proporcionální složky, S – zesílení sumační složky, D – zesílení diferenční složky.

Dále je v algoritmu regulátoru realizován *anit-windup*, kde je omezena hodnota sumační složky, aby nedocházelo k příliš dlouhé době zpoždění výstupu. Omezena je i hodnota akční veličiny, protože hodnota střídy PWM je maximálně 100%, kdy se motor otáčí plnou rychlostí.

Reakce motoru může být dále v algoritmu ovlivněna tzv. tolerančním pásem v hodnotách pixelů, v němž se může pohybovat absolutní hodnota regulační odchylky. Hodnota tohoto pásu udává možnou chybu dosažené polohy. Tento parametr zamezuje rozkmitání regulované soustavy při přeregulování požadované polohy, což je významné zejména při rychlé dynamice motoru anebo požadavku na rychlost regulace na úkor přesnosti.

7.4 Demonstrační aplikace polohování se stejnosměrným motorem

Běžným požadavkem regulátoru polohy je nastavit žádanou pozici a i za působení vnějších vlivů ji udržet. S využitím polohovacího mechanismu je sestavena aplikace, která pomocí statické kamery řídí polohovací mechanismus v ose *X*. Uživatel může nastavovat pozice v pixelech v rozsahu rozlišení řádku senzoru (1280 pixelů) a zpětně si vyžádat aktuální dosaženou pozici. Senzor je provozován s objektivem a při stanovení převodních konstant je možné pozice v pixlech přepočítat na metrické hodnoty viz *kapitola 3.1.3*.

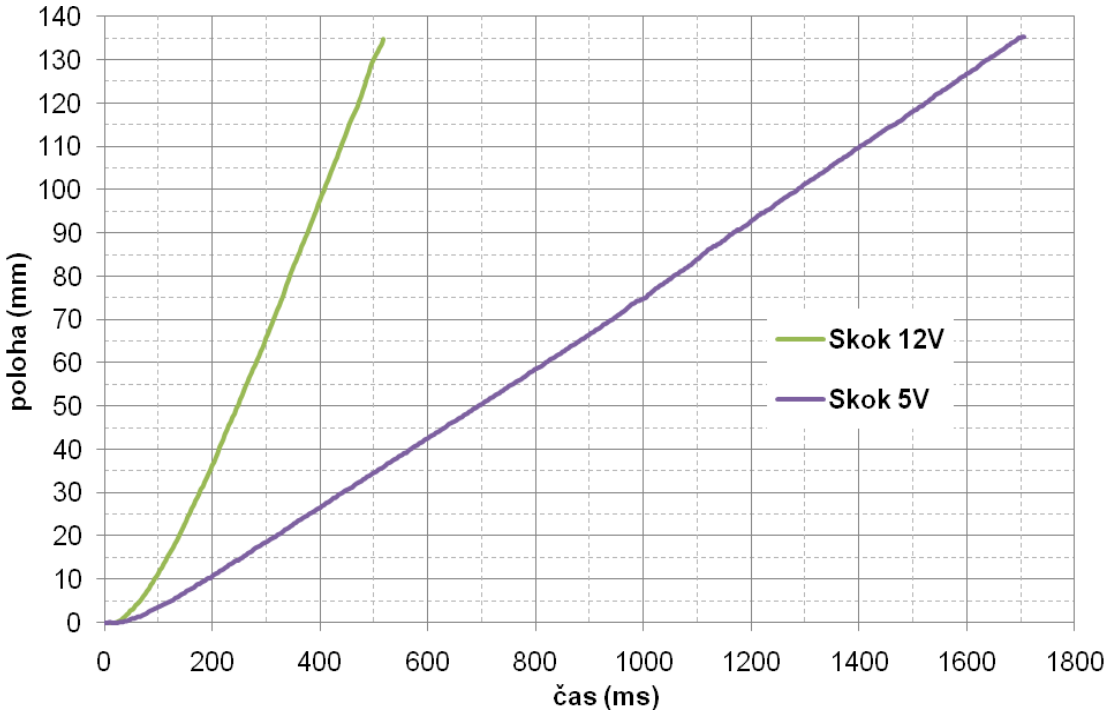
Příklad výpočtu určení rozměru pomocí senzoru s objektivem:

- Objektiv s ohniskovou vzdáleností $f´ = 25\text{mm}$
- Vzdálenost objektu od objektivu $a = 1000\text{mm}$
- Velikost pixelu senzoru $y´ = 5{,}2\mu\text{m}$

$$\beta' = \frac{f'}{z} = \frac{f'}{a - f'} = \frac{25}{1000 - 25} \cong 0{,}02564$$

$$y = \frac{y'}{\beta'} = \frac{0{,}0052}{0{,}02564} \cong 0{,}203\,mm$$

Z výpočtu vyplývá, že velikost jednoho pixelu odpovídá metrickému rozměru objektu 0,203mm a naopak velikost objektu 1mm zaujme na snímači 4,9 pixelu (5 pixelů).



Obr. 7.3 – Přechodová charakteristika polohovacího mechanismu v ose X

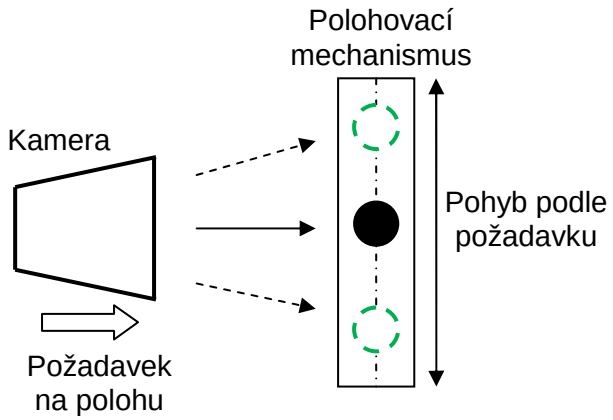
K představě dynamiky chování polohovacího mechanismu a k lepšímu stanovení složek regulátoru byla změřena přechodová charakteristika, kdy byl motor rozběhnut skokovou změnou svorkového napětí. Na *obr. 7.3* je vidět průběh změny polohy při skokové změně z 0V na 5V a na 12V. Vlastní parametry regulátoru jsou stanoveny na základě empirických metod viz [2], a zejména experimentálním přizpůsobení na základě pozorování procesu regulace.

Polohovací systém má vzhledem k nastavené vzdálenosti kamery od pozorovaného objektu parametry podle *tab. 7.1*. Pro jinou vzdálenost se hlavně nastavení jednotlivých složek PSD regulátoru může lišit, neboť se mění rozlišitelnost sledovaného objektu.

Parametry pro vzdálenost 1m kamery od objektu	**Hodnota**
Zesílení P složky regulátoru	100
Zesílení S složky regulátoru	0,3
Zesílení D složky regulátoru	0
Rozsah polohy	1280px
Perioda vzorkováni	3ms
Toleranční pole	±0,2px
Rozlišení výpočtu pozice	0,01px
Chyba měření pozice	0,1px
Přesnost regulace	0,4px
Napájení	5V

Tab. 7.1 – Parametry regulace se stejnosměrným motorem v ose X

Přesnost regulace polohy na zdanou hodnotu je stanovena na schopnosti opětovného návratu na danou pozici při zásahu vnějšího vlivu (zakrytí a odkrytí objektivu kamery).



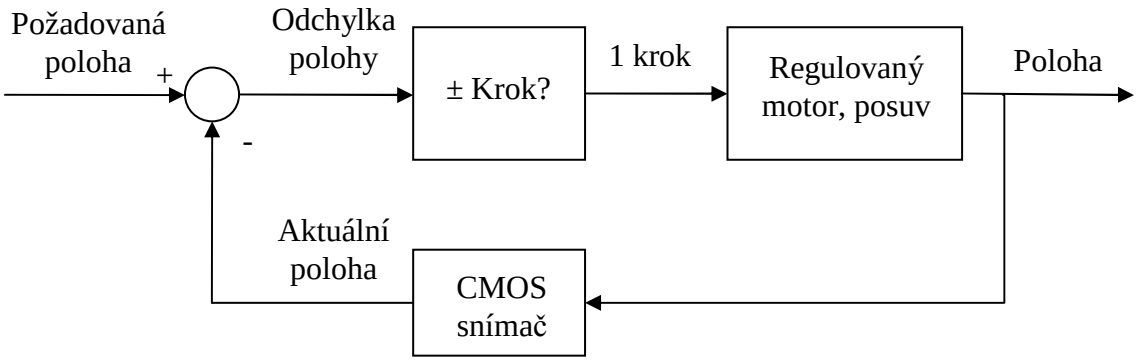
Obr. 7.4 – Aplikace se stejnosměrným motorem

7.5 Polohování krokového motoru

Zařazení krokového motoru do regulační smyčky se z hlediska řízení akčního zásahu může lišit. Hlavní rozdíl je, že na stejnosměrný motor lze nezávisle na procesoru přivést velikost napětí (hardwarový generátor PWM jako periferie mikroprocesoru), a ten pak koná samočinný pohyb. Pro krokový motor je situace složitější, protože každá změna polohy o jeden krok vyžaduje zásah procesoru, pokud by nebylo využito jiné nezávislé řízení. Aby se systém obešel bez dalšího hardwaru, je krokový motor řízen softwarově.

Krokový motor je sám o sobě konstrukčně přizpůsoben k nastavení polohy, zde v tomto případě je ale využíván jen jako hybná síla, kde za pomocí obrazového senzoru je nepolohován do žádané pozice. V této kombinaci je možné i eliminovat případné ztráty kroků při polohování motoru, neboť nezávisí na tom, kolik kroků motor vykonal, ale do jaké polohy se dostavil.

Při řízení rychlosti krokování je nutné dbát na to, aby prodleva mezi jednotlivými kroky byla dostatečně dlouhá a bylo tak docíleno otáčení. Jelikož rychlost snímání a zpracování obrazové informace senzoru v řádkovém režimu je větší nebo srovnatelná s minimální velikostí prodlevy pro motor, lze využít principu, kdy velikost jednoho regulačního zásahu je rovna právě jednomu kroku. Odpadá řešení regulátoru ve formě PSD a zároveň je možné ve většině případů dosáhnout žádané polohy bez přeregulování. Rychlost regulace je však závislá na schopnosti nastavení jednoho kroku motoru.



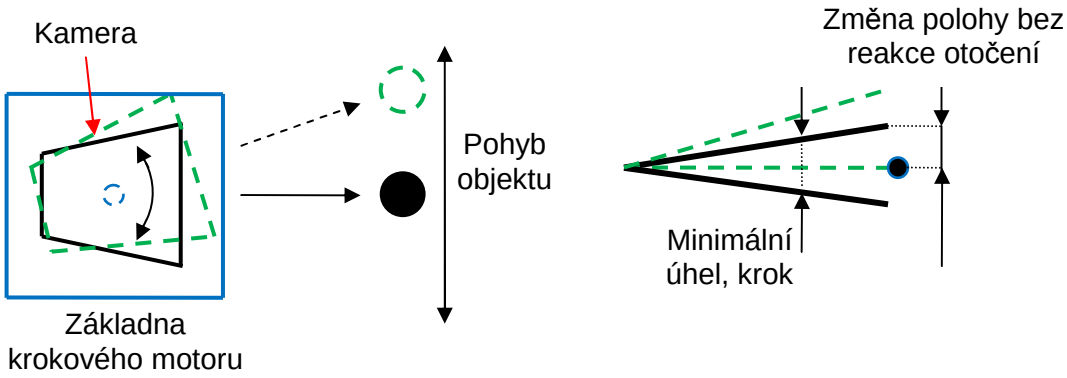
Obr. 7.5 – Regulační smyčka pro jednu osu souřadnic s krokovým motorem

7.6 Demonstrační aplikace s krokovým motorem

S využitím pouhého krokového motoru bez převodového mechanizmu je sestavena aplikace, kdy se kamera (obrazový snímač) umístěná přímo na hřídel motoru natáčí podle posuvu snímané značky. Lze tak demonstrovat automatické sledování pohybujícího se objektu v zorném poli až 360°, ale jen za předpokladu vyřešení připojení přívodních vodičů, které

omezují pohyb, a tím možný zorný úhel. Změna úhlu natočení je limitována počtem možných kroků motoru na otáčku, respektive minimální změna úhlu natočení.

U použitého krokového motoru je možné v režimu s poloviční krokem dosáhnout minimální změny úhlu natočení o 0,9°. Kamera se neustále snaží mít objekt zájmu ve středu zorného pole, proto aby nedocházelo k rozkmitání kamery, je nastaveno vhodné toleranční pole dosažené hodnoty pozice vzhledem ke vzdálenosti objektu od kamery. Platí, že čím bude objekt dále od kamery (zaujme méně pixelů na senzoru), tím užší toleranční pole a zároveň je definováno při jaké změně pozice se má kamera otočit za objektem.



Obr. 7.6 – Aplikace s krokovým motorem

Parametry pro vzdálenost 1m kamery od objektu	**Hodnota**
Perioda vzorkováni/prodleva kroku	10ms
Toleranční pole	±20px/±4,16mm
Minimální změna úhlu	0,9°
Rozlišení výpočtu pozice	0,01px
Chyba měření pozice	0,1px
Napájení	12V

Tab. 7.2 – Parametry regulace s krokovým motorem

8 PLOŠNÉ SNÍMÁNÍ POLOHY A JEJÍ REGULACE

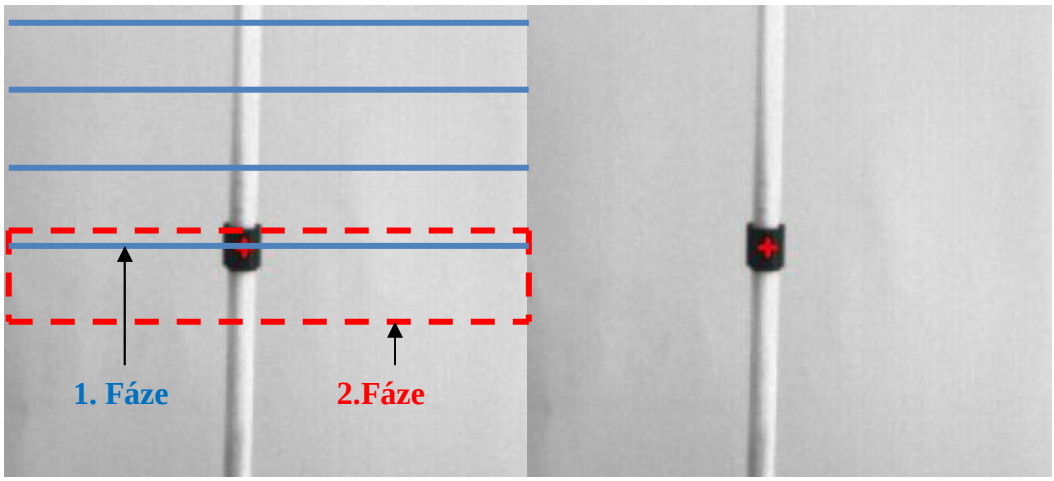
Oproti snímání a regulaci polohy v jedné ose je situaci komplikovanější. K určení polohy je nutné snímat a zpracovávat plošný obraz, a pro nastavení polohovacího mechanizmu je zapotřebí současně ovládat dva motory, které zprostředkovávají posuv ve dvou nezávislých osách. Posuv tedy například probíhá zároveň ve vodorovné i ve svislé ose.

8.1 Plošné zpracování obrazu

K vyhledání souřadnic umístění objektu ve dvourozměrné scéně, je nutné zpracovávat plošnou informaci z obrazového senzoru. Opět je vyžadováno mít v zorném poli senzoru nějakou jednoznačně identifikovatelnou značku, tak jako tomu bylo u řádkového zpracování obrazu. Pro rychlé a spolehlivé vyhledání značky, je nutné volit její charakter co nejjednodušší tak, aby bylo možné spolehlivě identifikovat její pozici v obraze. Tvar by měl tedy nelépe odpovídat nějakému geometrickému obrazci (kruh, čtverec, obdélník, atd.). Na takovýto obrazec lze poměrně jednoduše aplikovat algoritmus výpočtu těžiště (středu), které se chová jako hmotný bod a má jednoznačné souřadnice jak v ose X, tak v ose Y.

Vzhledem k dostupnému výpočetnímu výkonu mikroprocesoru je nutné zvolit kompromis mezi velikostí rozlišení zpracovávaného obrazu a rychlosti získání polohy. Nezpracovává se tedy obraz v plném rozlišení, ale pouze v poloviční rozlišení (640x512), aby mohl být regulační zásah dostatečně rychlý. Využívá se tzv. skip módu senzoru, kdy je zachována zorná plocha snímače, ale čte se každý druhý pixel jak ve vertikálním, tak horizontálním směru. Na rychlost zpracování obrazu přispívá i fakt, že čtení obrazové informace je plně softwarové, mikroprocesor je maximálně vytížen a zároveň nelze zpracovávat celý obraz najednou, ale pouze po blocích, které je možné uložit do paměti mikroprocesoru. Velikost jednoho bloku odpovídá 92 řádkům při jeho rozlišení 640 pixelů.

Vlastní algoritmus nalezení těžiště spočívá ve dvou fázích. V první fázi je orientačně nalezena značka tak, že je z aktuálního snímku přečten každý pátý řádek a v průběhu čekání na následující řádek je zapracováván předešlý řádek. Hledají se pixely, které mají hodnotu větší nebo menší než zadaný práh. Po nalezení shluku pixelů (alespoň tři po sobě jdoucí pixely) se pokračuje ve druhé fázi, kdy je do paměti mikroprocesoru vyčten blok právě 92 řádků od pozice řádku nalezené značky v předchozím kroku. Ve skutečnosti se však čte blok od pozice o 5 řádků menší, tak aby byla eliminována chyba identifikace z první fáze. Nad tímto blokem je proveden výpočet těžiště opět vzhledem k zadanému prahu. Obecný algoritmus výpočtu je uveden v *rce. 3.6* a příklad nalezení značky obraze na *obr. 8.1*.



Obr. 8.1 – Nalezení významného bodu (těžiště – červený křížek) v obrazovém snímku

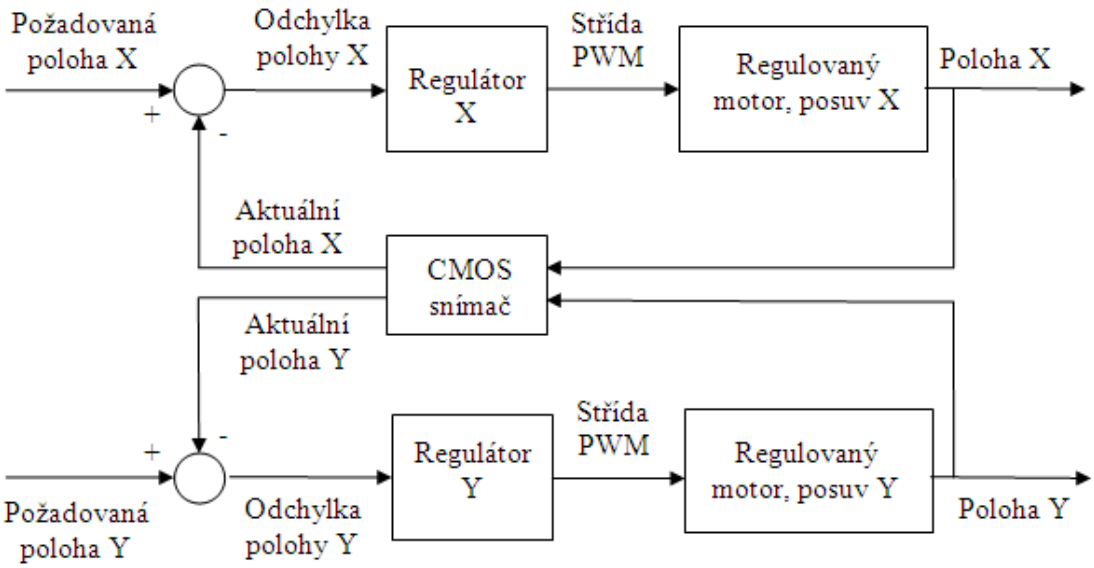
Algoritmus je optimalizován na rychlost, a tak předpokládá výskyt pouze jediné značky v zorném poli senzoru, což je pro potřeby regulace na jediný objekt dostačující. Také je předpokládáno, že velikost značky v ose sloupců bude minimálně alespoň 10 fyzických pixelů a maximálně 184 pixelů na obrazovém senzoru, pak je spolehlivě nalezeno těžiště. Není-li dodržena spodní hranice, pak značka nalezena vůbec nebude a je-li překročena horní hranice, pak je těžiště nalezeno s chybou v závislosti na velikosti překročení této horní hranice. Ve výsledku je výpočet těžiště uskutečněn právě ve dvou po sobě jdoucích snímcích senzoru.

Přesnost výpočtu je realizována na 0,01 pixelu s chybou určení polohy 0,1px (chyba vypočtena průměrováním z odečtení polohy statického objektu).

8.2 Regulace polohy se dvěma stejnosměrnými motory

Nastavení požadované polohy pro dvě souřadnicové osy z hlediska řízení akčního členu, který zde představují dva stejnosměrné motory, se příliš neliší proti regulaci polohy v jediné ose. Prakticky se jedná o zdvojenou regulační smyčku, kde zpětnou vazbu uzavírá obrazový senzor udávající informaci o aktuální poloze v plošném prostoru, tj. regulační odchylka a poté i regulační zásah je stanoven pro každý motor nezávisle.

Časová náročnost čtení a zpracování obrazových dat na informaci o poloze je však podstatně delší, než tomu je u jednoosé regulace. Perioda vzorkování se již pohybuje v řádech desítek až stovek milisekund a při minimální expoziční době senzoru lze docílit jednoho průchodu regulační smyčkou kolem 80 milisekund. Pro zachování této hodnoty musí být nastavena velikost elektronické závěrky senzoru do hodnoty 530, při překročení se prodlužuje průchod smyčkou.



Obr. 8.2 – Blokové uspořádání regulační smyčky pro dvě osy souřadnic

8.3 Demonstrační aplikace se dvěma stejnosměrnými motory

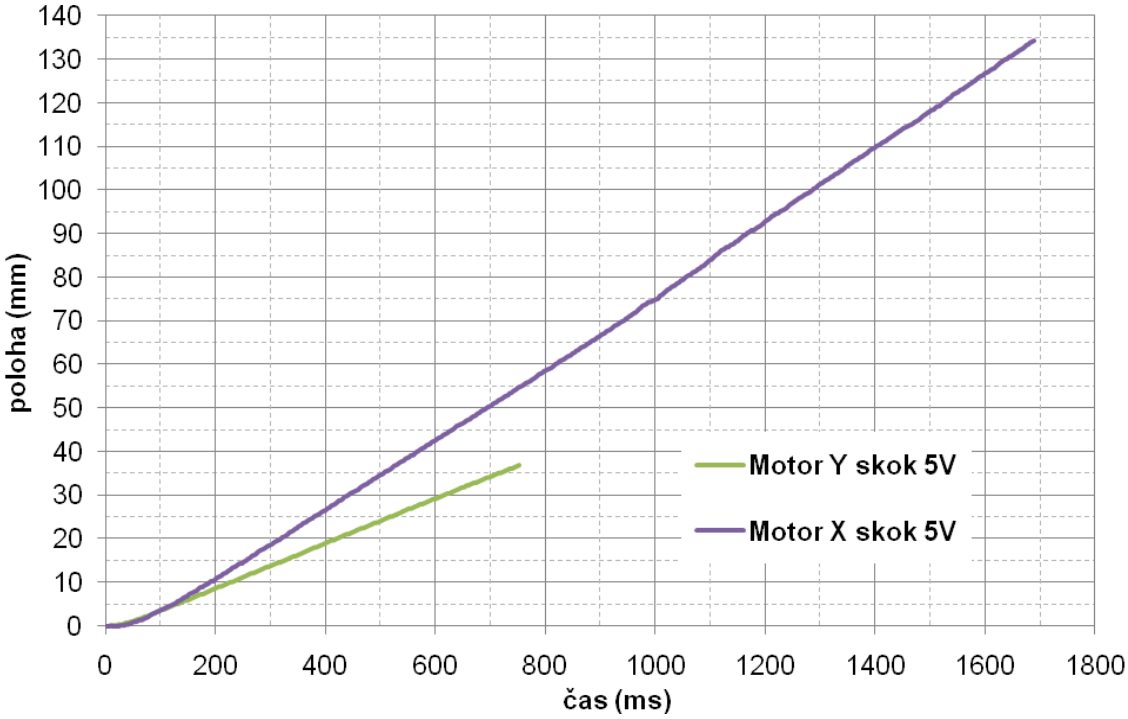
Principiálně je výsledná aplikace podobná aplikaci v *kapitole 7.4*, pouze nastavování žádané polohy je možné ve dvou nezávislých osách, a to vyžaduje i stanovení parametrů dvou PSD regulátorů. Parametry a dynamika výsledného regulačního systému jsou však horší a jsou zejména ovlivněny nízkou vzorkovací periodou v důsledku časově náročného sběru obrazových dat i v případě, kdy je degradováno rozlišení senzoru na polovinu.

Dosažené parametry ukazuje následující tabulka, kde rozměry parametrů v pixelech odpovídají pixelům při použití skip módu senzoru, tj. jeden pixel je roven dvou fyzickým na senzoru.

Parametry pro vzdálenost 1m kamery od objektu	**Hodnota**
Zesílení P složky regulátoru X	90
Zesílení S složky regulátoru X	0,3
Zesílení D složky regulátoru X	0
Zesílení P složky regulátoru Y	120
Zesílení S složky regulátoru Y	0,4
Zesílení D složky regulátoru Y	0
Rozsah polohy X	640px
Rozsah polohy Y	512px
Perioda vzorkováni	80ms

Toleranční pole	±0,5px
Rozlišení výpočtu pozice	0,01px
Chyba měření pozice	0,1px
Přesnost regulace	2px
Napájení	5V

Tab. 8.1 – Parametry regulace se stejnosměrnými motory v ose X a Y



Obr. 8.3 – Přechodová charakteristika polohovacího mechanismu v ose X a Y

9 PROGRAMOVÁ APLIKACE A KNIHOVNA PRO VYTVOŘENÝ MODUL

V průběhu návrhu systému snímače polohy vznikala ve vývojovém prostředí *Microsoft Visual Studio C#* grafická aplikace s názvem *Show Image*. Slouží především ke komunikaci a nastavení parametrů mikroprocesoru STM32, který řídí obrazový senzor a připojené motory. V tomtéž prostředí je vytvořena i knihovna *STM32.CAMERA.DLL* podporující všechny do značné míry totožné funkce aplikace.

9.1 Komunikační protokol mezi mikroprocesorem a PC

Vzhledem k velkému objemu přenášených dat a množství konfiguračních parametrů přenášených mezi mikroprocesorem a PC je navržen komunikační protokol v podobě rámců[3]. Lze tak dobře rozlišit, jaký typ dat je přenášen a zabezpečit jejich správné čtení a zápis.

Základní strukturu každého rámce tvoří hlavička, identifikátor typu dat, délka přenášených dat a vlastní data, viz následující tabulka. Hlavička *HEAD* slouží k rozpoznání dat, které přísluší datové výměně mezi mikroprocesorem a PC. Identifikátor *ID* blíže specifikuje typ přenášených dat (videokomunikace, I2C komunikace, komunikace s motory, atd.). Následuje položka *DATA LENGTH* udávající množství přenášených dat, a tím lze zabezpečit rezervaci paměti pro vlastní data. Vylepšení zabezpečení pro datový rámec by ještě mohlo spočívat v zakódování vlastních dat například CRC. Implementace však nebyla považována za zcela nutnou vzhledem k faktu, že přenos po USB je tímto kódováním již zabezpečen na linkové vrstvě.

HEAD	ID	DATA LENGHT	DATA

Název	Datový typ	Pořadí	Hodnota	Popis
HEAD	WORD	0	0x00FF8072	Hlavička identifikace komunikačního rámce
ID	BYTE	4	0xXX	Identifikace typu dat
DATA LENGHT	WORD	5	0xXXXXXXXX	Počet bajtů dat
DATA	BYTE, HALFWORD, WORD	9	X * BYTE	Počet bajtů podle DATA LENGHT

Tab. 9.1 – Struktura datového rámce přenášených dat po USB

[3] Podobný přístup byl již řešen kolegy v rámci katedry měření ČVUT FEL

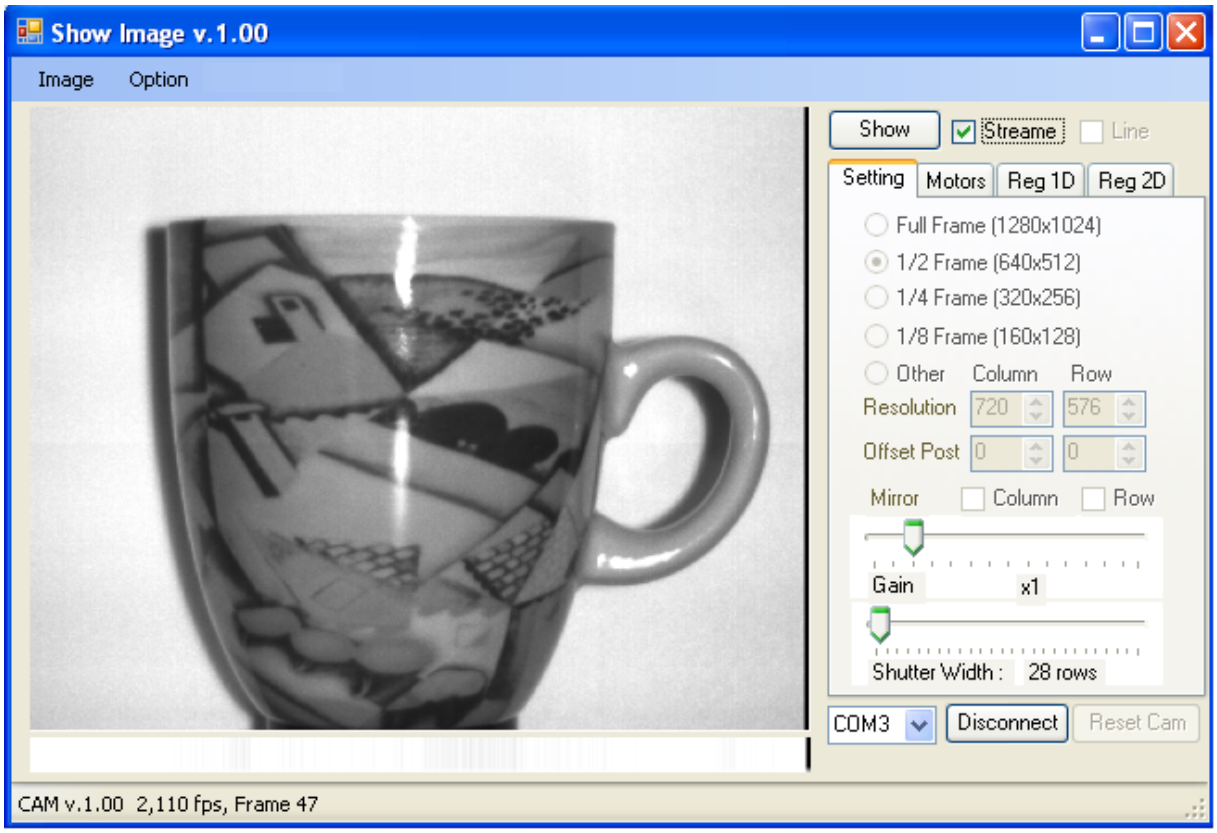
9.2 Rozvržení prvků aplikace

Základní obrazovka obsahuje čtyři části:

- **Menu lištu** – horní část okna,
- **Stavovou lištu** – dolní část okna,
- **Zobrazovací plochu** – centrální část okna,
- **Konfigurační blok** – pravá část okna.

Menu lišta obsahuje položku *Image->Clear* pro smazání snímku ze zobrazovací části a položku *Option*, kde lze vyvolat okno pro nastavení registrů kamery (*Register Camera*) anebo kontrolní položku nastavené vzorkovací periody pro regulaci (*Get Time Regulation*).

Na stavové liště se zobrazují informace, hlášky a chyby vzniklé při obsluze aplikace a při komunikaci s mikroprocesorem.



Obr. 9.1 – Aplikace Show Image – rozvržení prvků

Zobrazovací plocha slouží k vizualizaci přenesených obrazových dat v plošném i řádkovém režimu, případně se zobrazují i zpracovaná dat v podobě grafu. V řádkovém režimu se zobrazuje jak skutečný obraz, tak jasový profil.

Konfigurační blok slouží k parametrizaci a ovládání mikroprocesoru a tvoří tedy nejdůležitější část aplikace, proto bude detailněji popsán v následujícím textu.

9.3 Navázání komunikace mikroprocesoru s aplikací

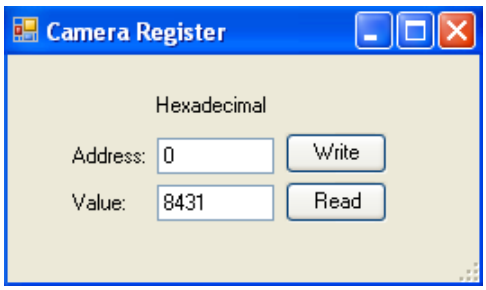
Komunikace mezi aplikací *Show Image* a mikroprocesorem je prostřednictvím virtuálního COM portu. Uživatel by měl vědět, na kterém portu je mikroprocesor připojen.

V knihovně *STM32.CAMERA* je navázání komunikace podporováno následujícími funkcemi:

- **openPort** – oteření vybraného portu,
- **closePort** – uzavření portu,
- **autoOpenPort** – automatické otevření portu,
- **getCOM** – vrací seznam dostupných portů,
- **getIdentification** – vrací identifikaci zařízení.

9.4 Nastavení a ovládání kamery

Ke snadnému a rychlému nastavení parametrů kamery, jako je rozlišení snímků, překlápění obrazu, zesílení a velikost elektronické závěrky je k dispozici záložka *Setting*. Čtyři základní rozlišení (*Full Frame*, *½ Frame*, *¼ Frame*, *1/8 Frame*) jsou konfigurována ve skip módech kamery, a tak je zachována aktivní plocha snímače. Dále lze v plném rozlišení zvolit výřez jakékoliv velikosti a pozice v obraze (*Other*). Tlačítkem *Show* je vyžádán jeden statický snímek a při zaškrtnutí *Streame* je vraceno video s maximální dostupnou rychlostí. Zaškrtnutím volby *Line* je vrácen jasový profil jednoho řádku (řádek 639) i skutečný obraz. Vyvoláním dialogového okna *Register Camera (Option->Register Camera)* je zpřístupněna editace registrů kamery, kde lze nastavit i jiné parametry, viz *obr. 9.2*.

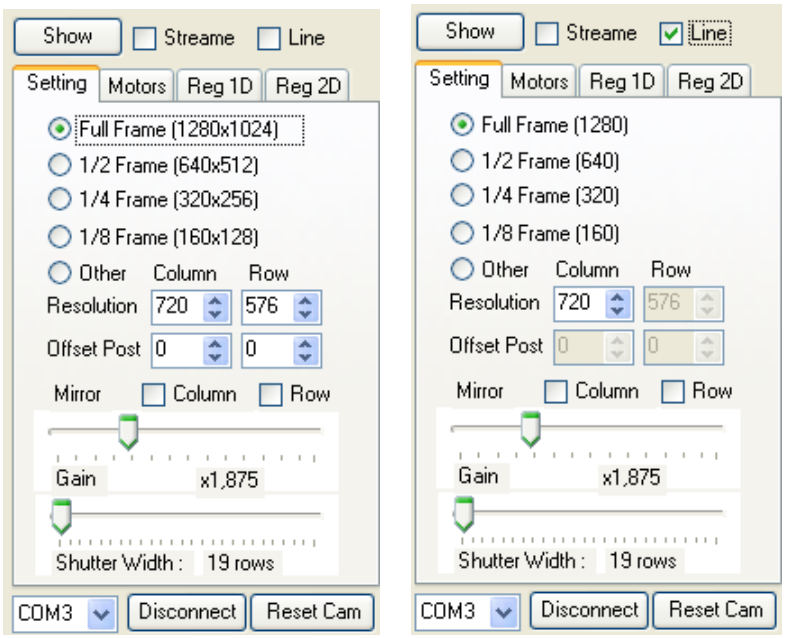


Obr. 9.2 – Aplikace Show Image – modifikace registrů kamery

V knihovně *STM32.CAMERA* je konfigurace a komunikace podporována následujícími funkcemi:

- **configImage** – podle parametrů nakonfiguruje kameru pro plošný obraz,
- **getImage** – vrátí jeden plošný snímek,
- **configLineImage** - podle parametrů nakonfiguruje kameru pro plošný řádkový obraz,
- **getLineImage** – vrátí jeden řádkový snímek (jasový profil),

- **writeRegCamera** – zápis do registrů kamery,
- **readRegCamera** – čtení registrů kamery,
- **setGainCamera** – nastaví zesílení kamery,
- **setShutterCamera** – nastaví délku elektronické závěrky kamery,
- **resetCamera** – provede původní nastavení registrů kamery.



Obr. 9.3 – Aplikace Show Image – nastavení parametrů kamery

9.5 Ovládání motorů

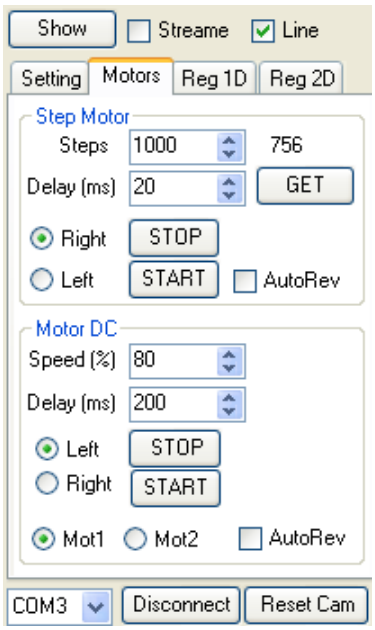
K ovládání motorů a nastavení jejich parametrů před spuštěním slouží záložka *Motors*. Lze ovládat jeden krokový a dva stejnosměrné motory.

Krokový motor je možné spustit zvoleným směrem (*Right*, *Left*) s definovanou prodlevou mezi kroky v milisekundách (*Delay*) a počtem vykonávaných kroků (*Steps*). Při každém novém spuštění se nečeká na dokončení předešlého počtu korků, ale uplatní se nové nastavení, je však možné se dotázat na počet zbývající kroků. Při zaškrtnutí položky *AutoRev* motor cyklicky mění směr otáčení pro zadanou konfiguraci.

Stejnosměrný motor je konfigurován a ovládán podobně jako krokový. Rychlost otáčení motoru je modifikovatelná změnou střídy PWM v rozsahu 0 až 100% (*Speed*). Dobu chodu motoru lze měnit v řádu milisekund (*Delay*), a je-li zadána nulová doba, motor má trvalý chod. Položka *AutoRev* mění cyklicky směr otáčení, pokud je zadána nenulová doba. Parametry lze uplatnit pro oba motory.

V knihovně *STM32.CAMERA* je ovládání motorů podporováno následujícími funkcemi:

- **startMotorDC** – spuštění a konfigurace stejnosměrného motoru,
- **stopMotorDC** – zastavení stejnosměrného motoru,
- **startMotorSTEPER** – spuštění a konfigurace krokového motoru,
- **stopMotorSTEPER** – zastavení krokového motoru,
- **getStep** – vrátí zbývající počet kroků krokového motoru.



Obr. 9.4 – Aplikace Show Image – ovládání motorů

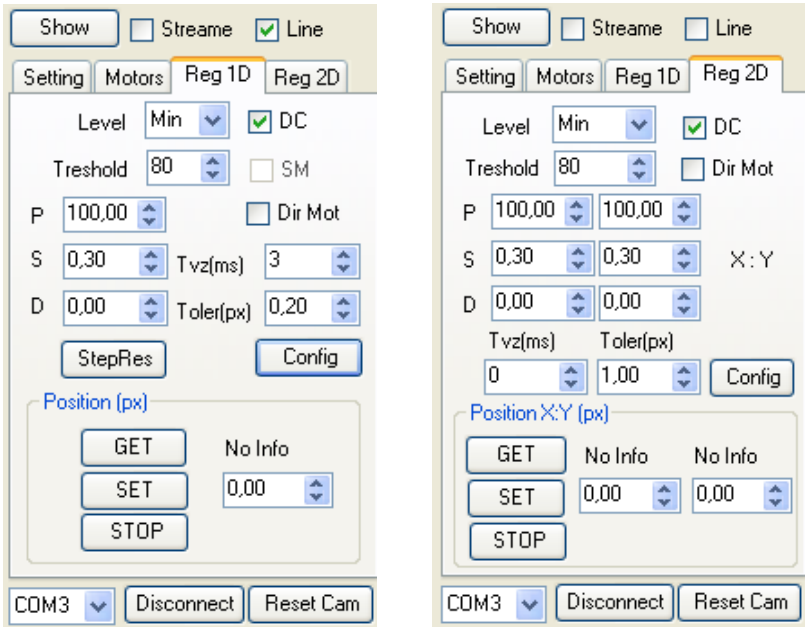
9.6 Nastavení parametrů snímače polohy a regulace

Parametry snímače polohy a regulace jsou nastavitelné v záložce *Reg1D* pro jednoosé řádkové polohování nebo *Reg2D* pro dvouosé plošné polohování. Jednotlivé položky jsou obdobné pro obě varianty.

Charakter hledaného bodu značky je volitelným parametrem *Level* a rozlišuje se hledání jasu minima, maxima, náběžné a spádové hrany pro řádkový režim. V plošném režimu lze vyhledávat minimum nebo maximum v podobě jasu těžiště. K nastavení úrovně rozhodujícího prahu jasu je položka *Treshold*. Parametry regulátoru lze stanovit zesílením jednotlivých složek PSD, kterým odpovídají položky *P, S, D*. Dosažení požadované polohy je ovlivnitelné tolerančním polem (*Toler*). Perioda vzorkování v milisekundách je modifikovatelná přes položku *Tvz*, ale skutečné vzorkování je závislé i na nastavené délce doby závěrky kamery, proto lze přes menu *Option ->Get Time Regulation* zpětně zkontrolovat nastavenou dobu průchodu regulační smyčkou a neodpovídá-li hodnota zvolené době, je nutné zkrátit dobu závěrky. Zaškrtnutím volby *DC* nebo *KM* je zvolen stejnosměrný nebo krokový motor jako

akční člen regulátoru a zároveň nezaškrtnutím žádné položky nejsou motory uvedeny do provozu, pak je pouze vracena aktuální pozice. Volba *Dir Mot* umožní automatické rozpoznání a přizpůsobení směru otáčení motoru, jinak nemusí zvolené zapojení motorů odpovídat platnému směru otáčení a regulace je tak nemožná. Nastavení těchto parametrů je aplikováno tlačítkem *Config*.

Odstartování snímání polohy nebo regulace je zahájeno tlačítkem *SET* a získání aktuálně dosažené pozice lze vyžádat tlačítkem *GET*. Zastavení činnosti procesu je aplikováno tlačítkem *STOP*. Je-li provedena konfigurace jednoosého regulátoru, je možné změřit přechodovou charakteristiku změny polohy stisknutím tlačítka *StepRes*. Počet vzorků jednoho odměru je nastaven na 512 vzorků a je vrácen graf na zobrazovací plochu i textový soubor s odměřenými vzorky (*DataStepRes.txt*).



Obr. 9.5 – Aplikace Show Image – nastavení regůlátorů

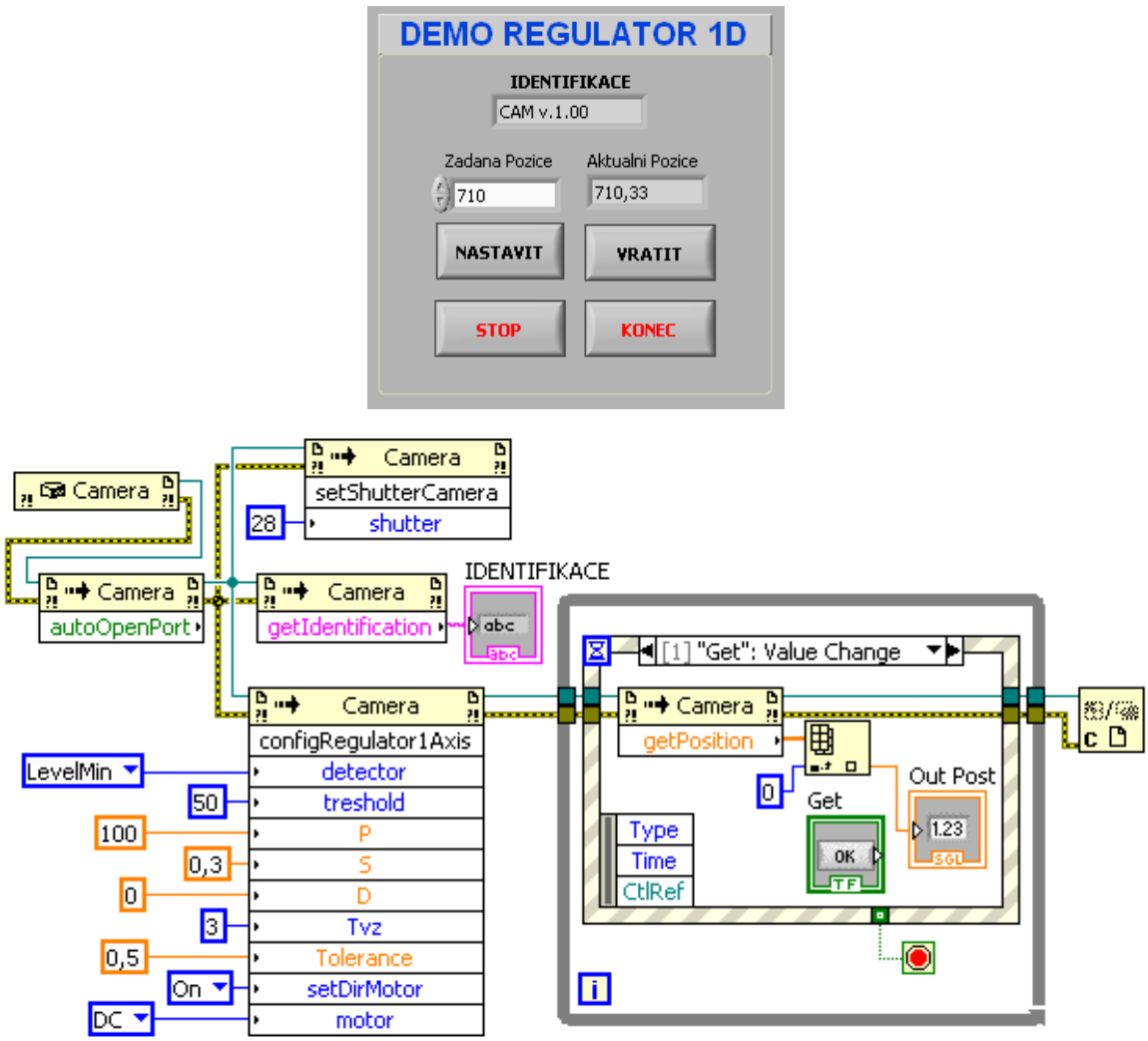
V knihovně *STM32.CAMERA* je nastavení snímače polohy a regulačních parametrů podporováno následujícími funkcemi:

- **configRegulator1Aixs** – konfigurace snímače, regulátoru pro jednoosý režim,
- **configRegulator2Aixs** – konfigurace snímače, regulátoru pro dvouosý režim,
- **setPosition** – nastavení požadované pozice,
- **getPosition** – vrátí aktuální dosaženou pozici,
- **stopRegulator** – zastaví regulátor,
- **startStepResponse –** spustí odměr přechodové charakteristiky,

- **isCompleteStepRes –** dotaz na komplexnost odměru,
- **getStepResponse** – vrátí data přechodové charakteristiky (čas, pozice),
- **getTimeRegulation** – vrátí změřený čas průchodu regulační smyčky v milisekundách.

9.7 Použití knihovny v prostředí LabView

Použití programové knihovny je přenositelné nejen pro běžné textové programovací jazyky, ale i pro poněkud netypické tzv. grafické programování jako je vývojové prostředí *LabView*. Struktura kódu je ve formě bloků, které vykonávají určitou funkci v závislosti na případných parametrech. Vykonávání jednotlivých operací závisí hlavně na tom, zdali má daná funkce k dispozici data, ale také na propojení bloků vodiči, jejichž barevné odlišení symbolizuje datový typ. Názorná ukázka grafického kódu s použitím knihovny pro vytvořený modul je viditelná na *obr. 9.6*, kde je sestaven regulátor polohy v jedné ose s obrazovým senzorem.



Obr. 9.6 – Použití knihovny v prostředí LabView – regulátor polohy v jedné ose

Začlenění knihovny do prostředí LabView je dostupné přes blok funkcí *Connectivity.Net.* Návrh kódu spočívá ve vytvoření bloku konstruktoru (instance třídy *Camera),* přes který jsou následně připojeny potřebné funkce.

Ve vlastním kódu je nejprve navázána komunikace s mikroprocesorem STM32, následuje blok identifikace, dále nastavení vhodné hodnoty elektronické závěrky a konfigurace jednoosého regulátoru se stejnosměrným motorem. Spuštění procesu regulace je uzavřeno do smyčky, ve které jsou registrovány události nastavení a vrácení pozice, zastavení procesu regulace a ukončení programu s uzavřením reference na objekt.

10 ZÁVĚR

Cílem diplomová práce bylo navrhnout a prakticky realizovat snímač polohy na základě obrazové informace a řízení akčního členu pro regulaci polohy. Hlavním řídícím členem měl být mikroprocesor společnosti *STMicroelectronics* s označením STM32. Zejména měl být kladen důraz na možnost využití integrovaného USB ke komunikaci s počítačem.

Výsledkem práce je sestavený do značné míry multifunkční modul, který obsahuje univerzální desku s mikroprocesorem STM32F105, desku s CMOS obrazovým senzorem a budící obvody pro řízení dvou stejnosměrných a jednoho krokového motoru. Pro mikroprocesor byl vyvinut firmware, který je schopný při nakonfigurování samostatně obsluhovat každou z možných připojených periferií, ale fungovat i jako systém uskutečňující proces snímání nebo regulace polohy.

Nejkomplikovanější částí řešení byla spolupráce obrazového senzoru s mikroprocesorem, protože použitý mikroprocesor neobsahuje hardwarovou podporu pro obrazové senzory, byla vyvinuta softwarová implementace sběru obrazových dat s rychlostí čtení obrazových dat 12MHz. Toto řešení lze považovat za velmi uspokojivé, neboť bylo dosaženo rychlosti, se kterou mikroprocesor dokáže maximálně indikovat změnu na svém vstupním portu. V implementaci bylo zejména využito teoretických poznatků programování mikroprocesoru STM32 v assembleru, kterým se zabývala moje předešlá bakalářské práce viz [3]. Právě návrhem rutiny v asembleru bylo docíleno precizního rychlého čtení dat, které pomocí jazyka C nebylo možné dosáhnout.

V průběhu vývoje snímače polohy vznikala grafická obslužná aplikace pro konfiguraci a ovládání mikroprocesoru a jeho periferií prostřednictvím rozhraní USB. Aplikace umožňuje vizualizaci obrazových dat včetně zobrazení snímané polohy, ovládání motorů, nastavení parametru regulátorů a další.

obrazový senzor ve spolupráci s mikroprocesorem a obslužnou aplikací může fungovat jako kamera s možností změny rozlišení obrazu. Při plném rozlišení senzoru 1280x1024 však bylo dosaženo rychlosti pouze 0,5 snímku za sekundu. Hlavní limitující faktor v rychlosti obrazové reprezentace dat hraje nízká přenosová rychlost integrovaného řadiče USB (specifikace *Full Speed)* v mikroprocesoru a jeho malá paměť, problematikou se zabývá *kapitola 6.3*. Toto řešení je tedy spíše vhodné pro snímání statické scény. Při degradaci rozlišení senzoru se však frekvence snímkování zvyšuje a například při rozlišení 320x256 je již dosaženo 9,5 snímků za sekundu.

Vlastní snímač polohy s obrazovým snímáním byl realizován pro určení pozice objektu z jednorozměrné scény i z plošné scény. Pro přesné určení polohy v jednorozměrné scéně byl implementován algoritmus lineární interpolace s výpočetní rozlišitelností 0,01 pixelu, výsledná přesnost dosahuje chyby 0,1 pixelu. Tato chyba se projevila při opakovaném snímání statických objektů v běžném pokojovém prostředí. Je však možné, že přesnost může být i lepší při nasazení do prostředí s lepšími klimatickými podmínkami, aby byl eliminován šum obrazového senzoru. Snímání polohy v plošné scéně dosahuje přibližně dvojnásobně horších parametrů vzhledem k menšímu použitému rozlišení obrazového senzoru, zejména z důvodů urychlení čtení obrazových dat a jejich zpracování.

Ve spoluprácí s obrazovým senzorem byl realizován jednoosý regulátor se stejnosměrným motorem pro posuv vozíku s přesností regulace na 0,4 pixelu. Na dosažené parametry má negativní vliv charakter použitého polohovacího mechanismu (vůle, vibrace a hystereze při pohybu vozíku) a lze předpokládat dosažení ještě lepších výsledků při kvalitnějším konstrukčním uspořádání jiného polohovacího mechanismus.

Dále byla sestavena aplikace s krokovým motorem, která umožňuje zpracováním jednorozměrné scény sledovat pohybující se objekty, tak že se snímač polohy (kamera) natáčí v diskrétních krocích motoru za pozorovaným objektem. Sledovaný objekt se tedy může pohybovat v ideálním případě v zorném poli až 360°, omezení však spočívá v nevyřešení kabeláže pro napájení a řízení motoru.

Realizován byl i regulátor v plošném obraze, kde je řízením dvou stejnosměrných motorů možné nastavení polohy ve dvou osách. Přesnosti jsou však opět horší než pro jednoosé aplikace, což je dáno nejen charakterem polohovacího mechanismu, ale také pomalým regulačním zásahem. Docílení uspokojivějších parametrů by také mohlo být dosaženo pomoci lepšího návrhu regulátoru, ale to nebylo primárním cílem této práce.

Pro realizovaný modul snímače polohy byla vytvořena programová knihovna. K dispozici je tedy systém, který lze nadále programovat a obsluhovat v různorodých vývojových prostředích. Ukázkovým příkladem je sestavení regulátoru pod platformou *LabView viz kapitola 9.7*. Uživatel tohoto modulu si může v rámci možností sestavit vlastní virtuální přístroj, například pro měření polohy, regulaci polohy nebo použít jen obrazový senzor jako kameru anebo jen ovládat připojené motory.

LITERATURA

[1] FISCHER, Jan. *Optoelektronické senzory a videometrie*. Praha : Vydavatelství ČVUT, 2002. 141 s. ISBN 80-01-02525-X.

[2] HINIOVÁ, Kateřina. *Řídicí technika*. Praha : Nakladatelství ČVUT, 2006. 148 s. ISBN 80-01-03368.

[3] TOMÁŠ, Michal. *Vývojový modul s procesorem STM32 a jeho programování*. Praha, 2009. 68 s. Bakalářská práce. ČVUT FEL.[cit. 3.5.2011]. Dostupné z WWW: http://measure.feld.cvut.cz/cs/system/files/files/cs/vyuka/zaverecne_prace/BP_2009_Tomas_Michal_locked.pdf.

[4] DVOŘÁK, Petr. *Polohovací řídicí systém využívající obrazové informace*. Praha, 2007. 120 s. Diplomová práce. ČVUT FEL. [cit. 3.5.2011]. Dostupné z WWW: http://measure.feld.cvut.cz/cs/system/files/files/cs/vyuka/zaverecne_prace/DP_2007_Dvorak_Petr_locked.pdf.

[5] BĚLÍK, Petr. *Snímač polohy využívající obrazové informace*. Praha, 2011. 98 s. Diplomová práce. ČVUT FEL. [cit. 3.5.2011]. Dostupné z WWW: http://measure.feld.cvut.cz/cs/system/files/files/cs/vyuka/zaverecne_prace/DP_2011_Belik_locked.pd.

[6] Objektivy, *Photo Mysteria* [online]. 2010, 11.10.2010 [cit. 2011-05-08]. Dostupné z WWW: http://photo.mysteria.cz/clanky/objekt2.html.

[7] Objektiv. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2011-04-22]. Dostupné z WWW: http://cs.wikipedia.org/wiki/Objektiv.

[8] PALATKA, Petr. Kamerové systémy pro přesné měření a kontrolu kvality v průmyslu. *Automatizace* [online]. 2010, 53, 1-2, [cit. 2011-05-08]. Dostupný z WWW: www.automatizace.cz.

[9] STMICROELECTRONICS, *RM0008 Reference Manual STM32* – rev *12* [online]. 2011 [cit. 2011-04-05]. Dostupné z WWW: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/REFERENCE_MANUAL/CD00171190.pdf.

[10] STMICROELECTRONICS, *Datasheet STM32F105xx, STM32F107xx* – rev 5 [online]. 2011 [cit. 2011-04-05]. Dostupné z WWW: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/DATASHEET/CD00220364.pdf.

[11] STMICROELECTRONICS, *UM0919 User Manual STM32VLDISCOVERY – rev 1* [online]. 2010 [cit. 2011-04-05]. Dostupné z WWW: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/USER_MANUAL/CD00267113.pdf

[12] Neuron wiki, *Firmware Library STM32*. [online]. 2010 [cit. 2011-04-08]. Dostupné z WWW: http://neuron.feld.cvut.cz/wiki/datasheets/FirmLib_STMF103_UM0427.pdf.

[13] ŘEZÁČ, Kamil. *Robotika* [online]. 2002 [cit. 2011-04-10]. Krokové motory. Dostupné z WWW: http://robotika.cz/articles/steppers/cs.

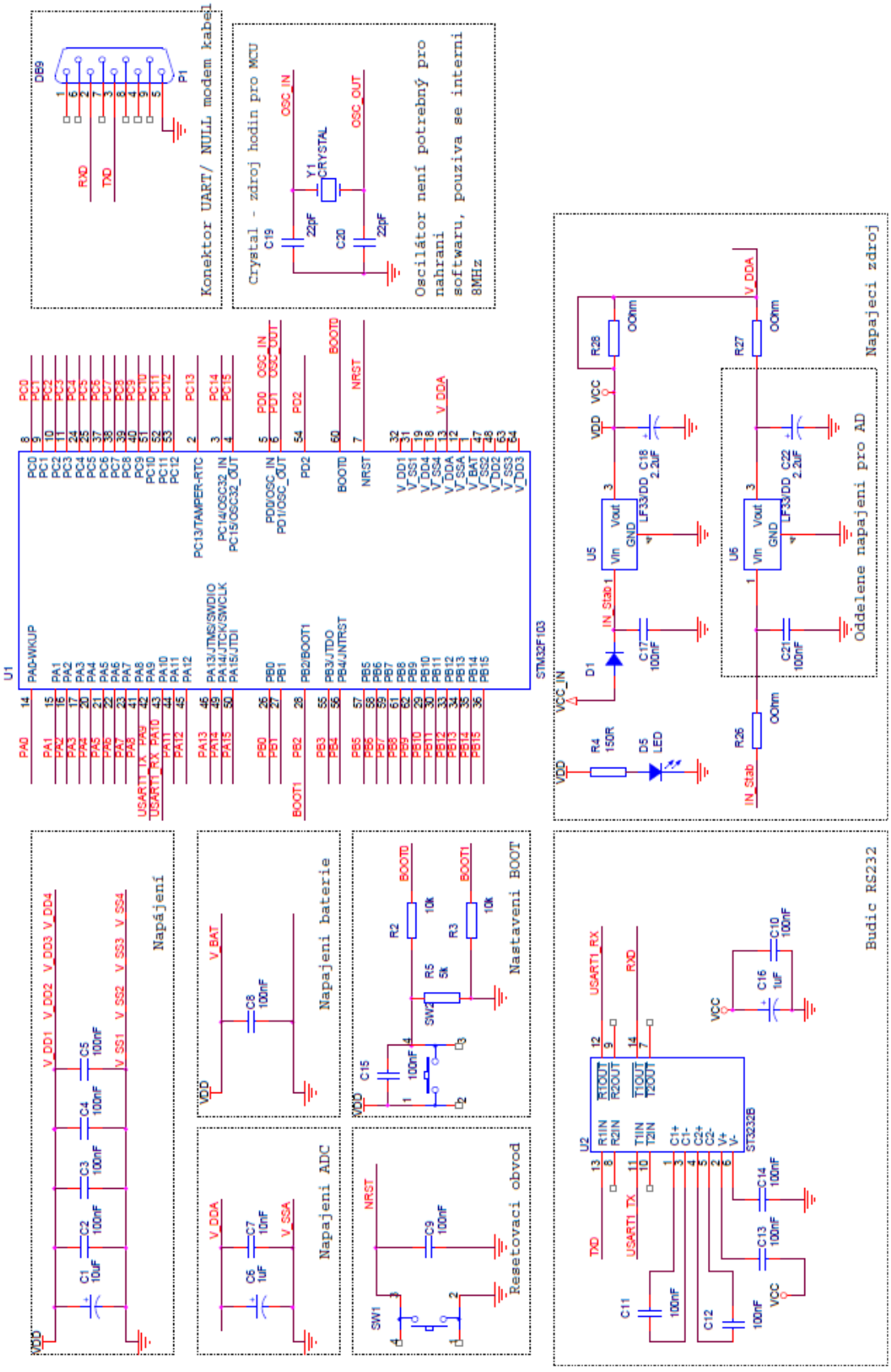
[14] MICRON TECHNOLOGY, Inc, *Datasheet MT9M001* – rev *F* [online]. 2010 [cit. 2011-11-10]. Dostupné z WWW: http://pdf1.alldatasheet.com/datasheet-pdf/view/115168/MICRON/MT9M001.html.

[15] USB TECHNOLOGY, *USB 2.0 Specification* [online]. 2000 [cit. 2011-10-12]. Dostupné z WWW: http://www.usb.org/developers/docs/usb_20_021411.zip.

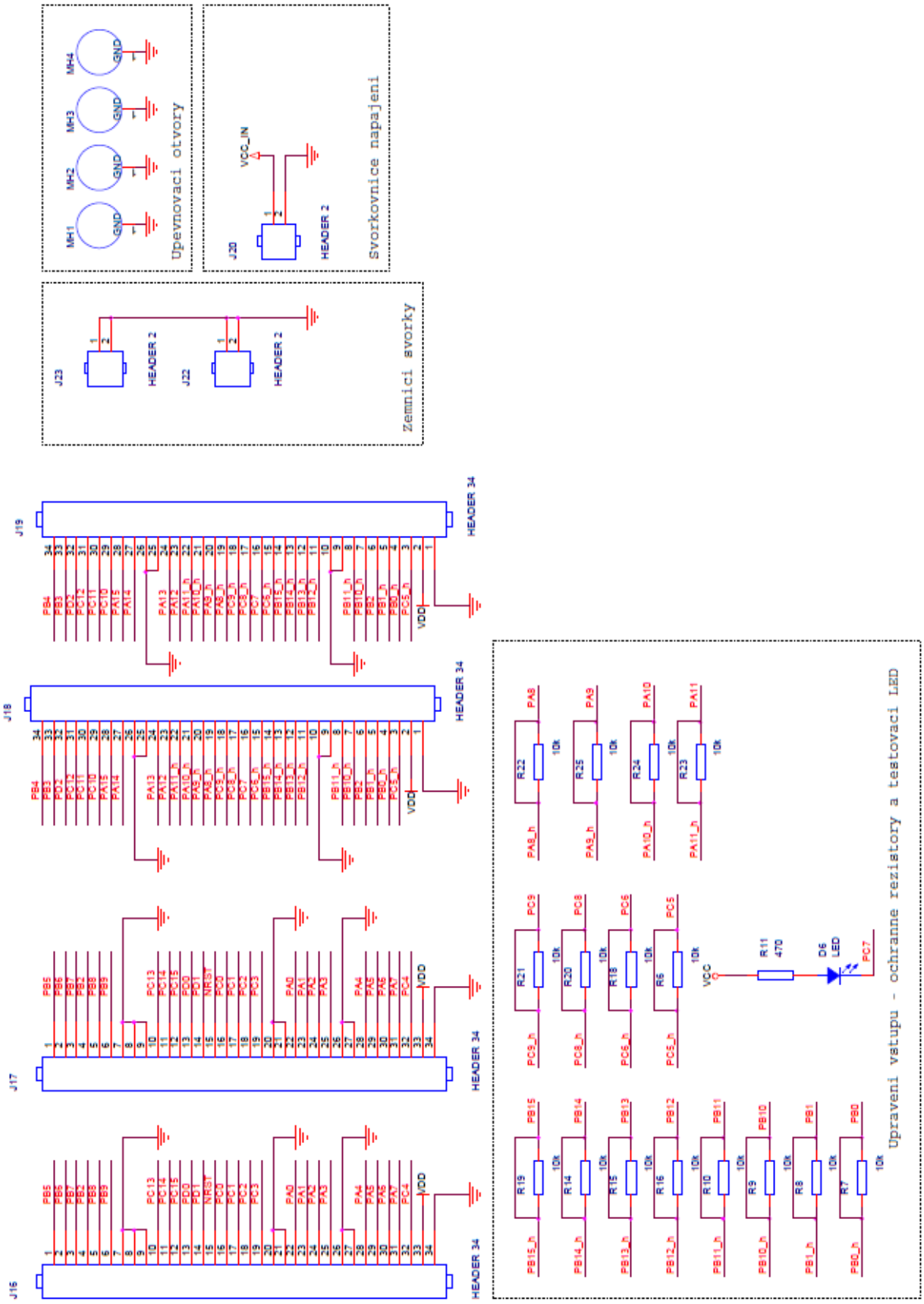
[16] REDAKCE HW SERWERU, *USB 2.0* [online]. 2005 [cit. 2011-10-12]. Dostupné z WWW: http://hw.cz/Rozhrani/ART1232-USB-2.0---dil-1.html.

PŘÍLOHY

A - Zapojení desky s STM32F103/F105, 64 pinů, část 1



A - Zapojení desky s STM32F103/F105, 64 pinů, část 2



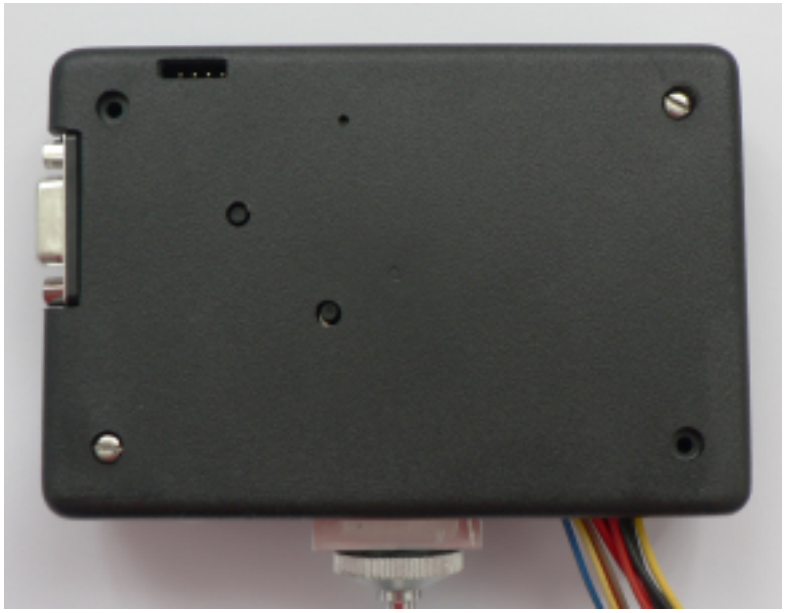
B - Disk CD-ROM

OBSAH:

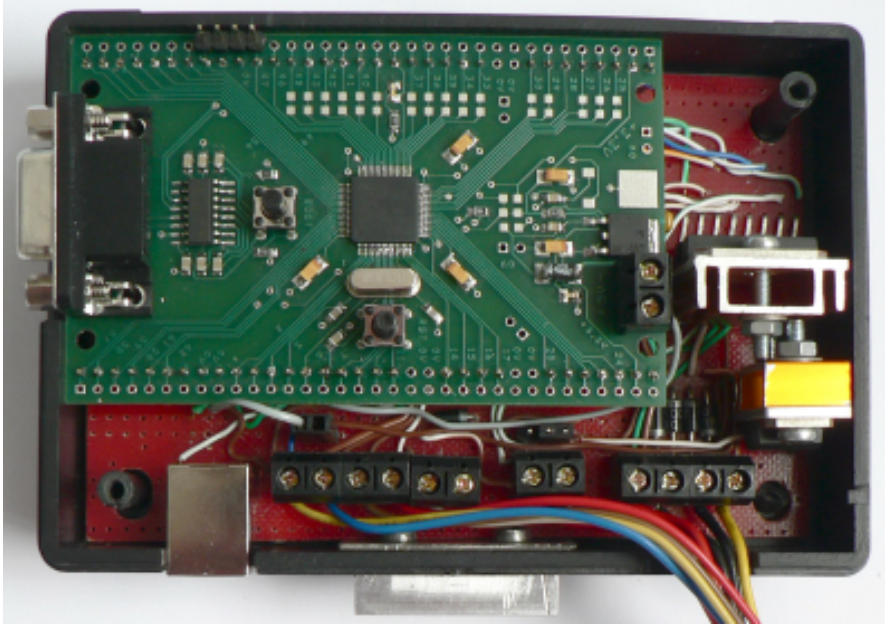
- **Zkompilované programy**
    - STM32 Firmware
    - PC Aplikace
- **Zdrojové kódy k programům**
    - STM32 Firmware
    - PC Aplikace
- **Dokumentace**
    - Datasheety k hardwaru
    - Komunikační protokol
- **Elektronická forma diplomové práce**

C - Fotodokumentace

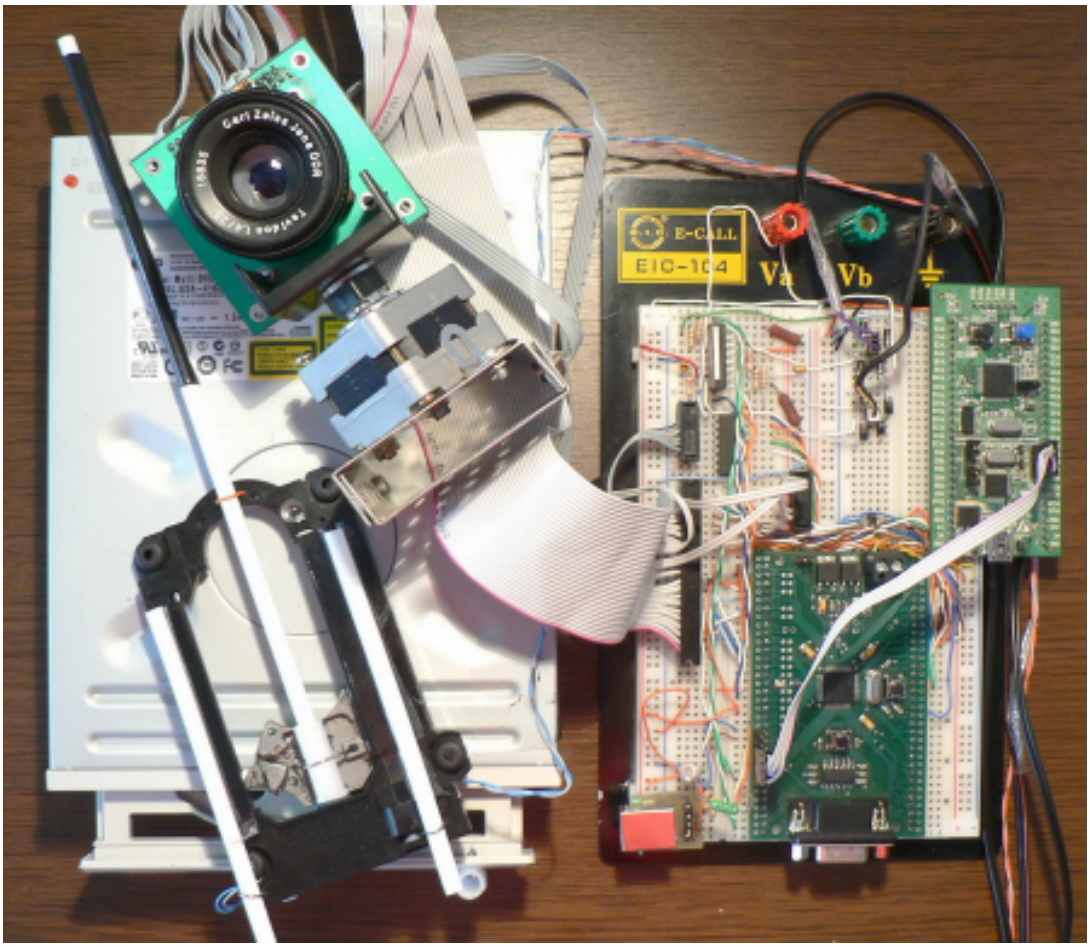
Modul snímače – pohled zepředu



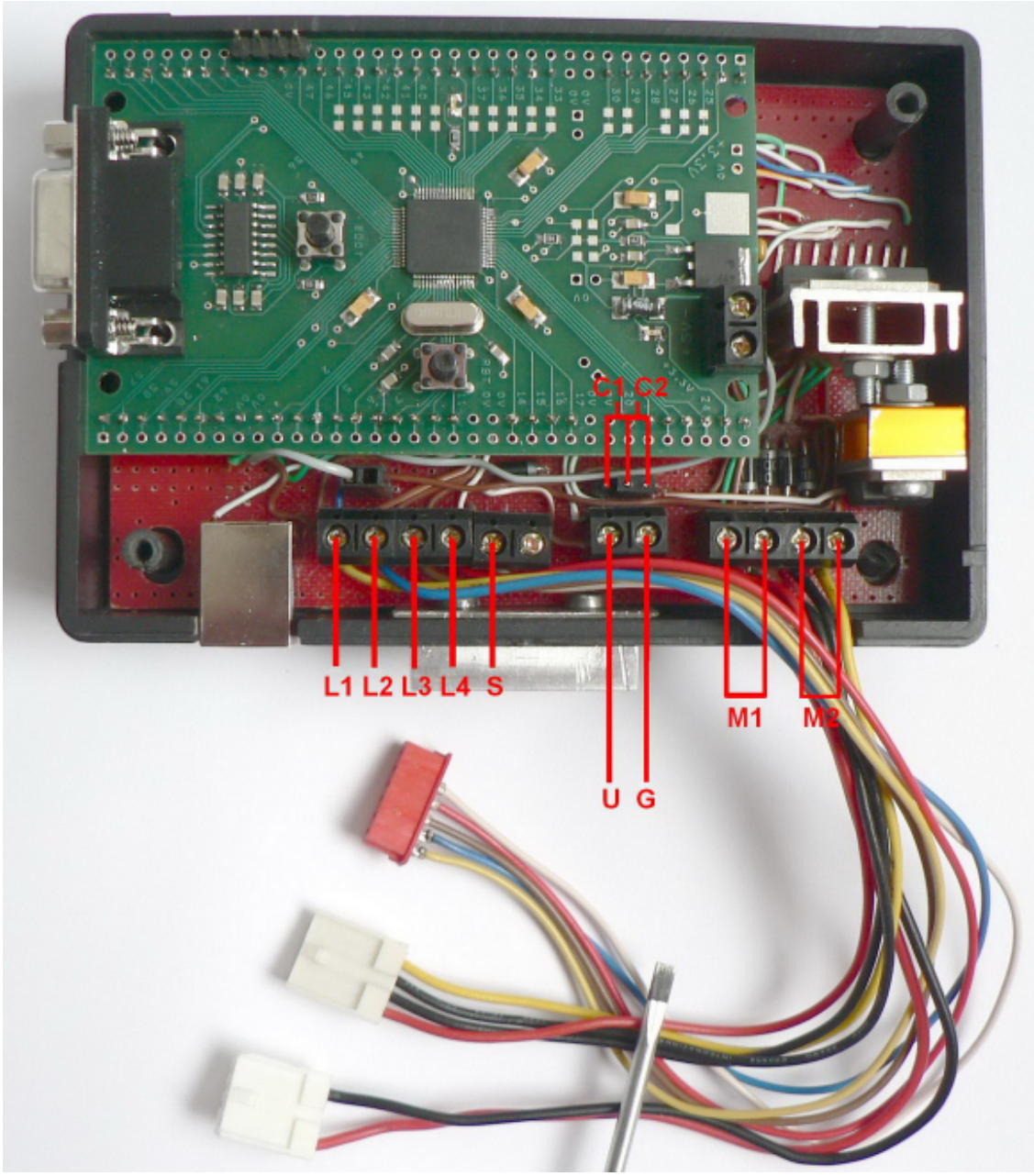
Modul snímače – pohled zezadu



Modul snímače – pohled na detail



Modul snímače na kontaktním poli s mechanickými prvky a motory



Modul snímače s popisem připojení přídavného napájení a motorů

- ***C1*** *– propojení pro napájení motorů 6-12V*
- ***C2*** *– propojení pro napájení motorů z 5V nebo USB*
- ***L1, L2, L3, L4*** *– připojení cívek unipolárního krokového motoru*
- ***S*** *– připojení společného středu cívek krokového motoru*
- ***M1*** *– připojení stejnosměrného motoru č.1 (osa X)*
- ***M2*** *– připojení stejnosměrného motoru č.2 (osa Y)*
- ***G*** *– připojení zemnícího vodiče napájení*
- ***U*** *– připojení napájecího napětí 6-12V*