Využití vývojových platforem pro výrobu kamery

Bc. Jiří Zmeškal

Diplomová práce
2018

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2017/2018

ZADÁNÍ DIPLOMOVÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jiří Zmeškal**
Osobní číslo:	**A16245**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Bezpečnostní technologie, systémy a management**
Forma studia:	**prezenční**
Téma práce:	**Využití vývojových platforem pro výrobu kamery**
Téma anglicky:	**Using Development Platforms for Camera Production**

Zásady pro vypracování:

1. **Prostudujte a popište princip činnosti kamer a vývojových platforem.**
2. **Zhodnoťte a vyberte komerčně dostupné kamerové moduly a vývojové platformy pro bezpečnostní kamery.**
3. **Navrhněte kameru na základě dostupných vývojových platforem.**
4. **Realizuje a otestujte návrh kamery.**
5. **Vytvořte podklady pro využití produktu v předmětu Kamerové systémy.**

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **MARGOLIS, Michael. Arduino cookbook. Second edition. Sebastopol: O'Reilly, 2011, xx, 699. ISBN 978-1-4493-1387-6.**
2. **WARREN, John-David, Josh S. ADAMS a Harald MOLLE. Arduino robotics. New York: Apress, 2011, xxiv, 601. Technology in action. ISBN 978-1-4302-3183-7.**
3. **BANZI, Massimo. Getting Started with Arduino. Second Edition. Sebastopol:OReilly Media, 2011, 128 s. ISBN 978-14-4930-987-9.**
4. **MANN, Burkhard. C pro mikrokontroléry: ANŠI-C, kompilátory C, spojovací programy - linkery, práce s ATMEL AVR a MSC-51, příklady programování v jazyce C, nástroje programování, tipy a triky,-. Praha: BEN - technická literatura, 2003, 279 s. ?C & praxe. ISBN 80-7300-077-6.**
5. **PINKER, Jiří. Mikroprocesory a mikropočítače. Praha: BEN - technická literatura, 2004, 159 s. ISBN 80-7300-110-1.**
6. **KLEIDERMACHER, David a Mike KLEIDERMACHER. Embedded systems security: practical methods for safe and secure software and systems development. Amsterdam: Elsevier, 2012, xx, 396 s. ISBN 978-0-12-386886-2.**
7. **WHITE, Elecia. Making embedded systems. Sebastopol: O'Reilly Media, 2011, xiv, 310. ISBN 978-1-449-30214-6.**

Vedoucí diplomové práce: **doc. Mgr. Milan Adámek, Ph.D.**
Ústav bezpečnostního inženýrství

Datum zadání diplomové práce: **8. prosince 2017**

Termín odevzdání diplomové práce: **28. května 2018**

Ve Zlíně dne 8. prosince 2017

doc. Mgr. Milan Adámek, Ph.D.
děkan

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

doc. RNDr. Vojtěch Křesálek, CSc.
ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne …….

podpis diplomanta

ABSTRAKT

Práce se zabývá návrhem a realizací kamery z dostupných komerčních prostředků. Vyrobená kamera včetně dokumentace bude sloužit jako výuková pomůcka v předmětu Kamerové systémy. Obsahem teoretické části je objasnění základních pojmů v oblasti kamer a mikropočítačů. Dále jsou popsány základní části vývojových platforem a zhodnocení vybraných platforem podle nastavených kritérií. V praktické části je navrhnuta kamera za použití kamery OV7670 a vývojových platforem Arduino Mega 2560 a Mbed NXP LPC1768. Dále jsou popsány programové vybavení obou platforem a jejich propojení s počítačem. Realizována je také vývojová platforma Raspberry PI s vlastní kamerou verze 2.1. Následně jsou zhodnoceny výstupy obrazů podle kvality. Poslední částí jsou vytvořené úkoly pro studenty pro naučení práce s těmito vývojovými platformy a osvojení programování pro předmět Kamerové systémy.

Klíčová slova: kamera, mikrokontroler, výukový materiál, Arduino Mega 2560, Mbed NXP LPC1768, OV7670 FIFO AL422

ABSTRACT

The focus of this thesis is a development of a video recorder using commercially available components. The resulting system will serve as an educational utility for the Camera Systems course, including its documentation. The theoretical part contains a thorough introduction to basic concepts of camera and microcomputer technology. Development platforms are discussed focusing on a description of their basic components and their comparison based on specific criteria. In the practical part, the system, using a camera OV7670 and development platforms Arduino Mega 2560 and Mbed NXP LPC 1768, is designed, followed by a description of the accompanying software and the connection of the platforms with a computer. Implemented is also a development platform Raspberry PI with a custom camera version 2.1. Then, the quality of obtained images is discussed. Finally, tasks for students are presented, aiming to support their efforts to master work with development platforms and help them to develop the programming skills required for the Camera Systems course

Keywords: camera, microcontroller, studying material, Arduino Mega 2560, Mbed NXP LPC1768, OV7670 FIFO AL422

Na tomto místě bych chtěl poděkovat vedoucímu práce, Doc. Mgr. Milanu Adámkovy, Ph.D. za cenné rady při tvoření práce a dále velmi poděkovat konzultantu Ing. Stanislavu Kováři za věnovaný čas a pomoc při řešení mé diplomové práce. Také bych chtěl poděkovat mým rodičům za podporu při celé době studia.

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

V dnešním světe plném elektroniky a elektrických zařízení najdeme nepřeberné množství aplikací, kde jsou využívány mikrokontrolery. Ať už se jedná o automobil, televizi, mobilní telefon, hračky pro děti, zbraňové systémy a další zařízení. Všechna tato zařízení mají jedno společné. Na prvopočátku byl vývoj programových aplikací a testování funkčnosti těchto zařízení. Právě pro testování jsou mnohdy využívána profesionální prostředí simulující průběh samotného programu využitého pro ovládání. Druhou možností, jak nahlédnout do světa vývoje a naučit se novým věcem, je využití vývojových platforem, kde si můžeme nasimulovat téměř cokoliv.

Právě touto cestou se vydává diplomová práce, kterou právě držíte ve své ruce. Pojednává o možnostech využití vývojových platforem pro studijní účely, kde se zabývá propojováním kamerových i jiných systémů. Vychází z volně dostupných open-source platforem, které mají velkou technickou podporu jak softwaru, tak připojených modulů a hardwarových aplikací. V diplomové práci jsou rozebrány teoretické poznatky ohledně fungování kamer jak po stránce softwarové, tak hardwarové. Jsou řešené principy snímání obrazu, jejich přenosu v samotné kameře a optické prvky kamery, které zajišťují přenos a transformaci světla na světlocitlivý prvek. Dále budou rozebrány vývojové platformy využitelné pro studijní účely, jejich fungování a elektronické součásti, ze kterých se tyto platformy skládají. Ke konci teoretické části práce budou zhodnoceny kamerové moduly, které jsou využity v pozdější praktické části a zhodnocení jednotlivých často využívaných vývojových platforem.

Praktická část se bude zabývat propojením dvou vývojových platforem, a to Arduina Mega 2560 a Mbedu NXP LPC 1768 s kamerou OV7670 s vestavěným FIFO bufferem AL422 včetně programového vybavení a způsobu propojení těchto platforem s PC. Dále bude zhodnocení výstupu obrazu z těchto vytvořených kamer a porovnání jednoduchosti programování vývojových platforem. Následně zde bude vytvořen materiál pro výukové účely a osvojení programování těchto platforem pro studenty do předmětu Kamerové systémy.

I. TEORETICKÁ ČÁST

1 PRINCIP ČINNOSTI KAMER

Princip činnosti kamer by se dal rozdělit do tří hlavních sekcí. První sekce je optická část, kde dochází k usměrňování světelných paprsků přes soustavu čoček na světlocitlivý prvek. Druhá sekce je převedení tohoto světelného toku z elektromagnetické podstaty fotonu na čistě elektrický signál, který dále dokážeme zpracovávat. Na to pak navazuje elektronická část, kde dochází ke zpracování samotného obrazu, jeho přenos a uložení na nějaký druh média, případně pouze zobrazení. V následujících kapitolách bude vysvětlen princip kamery a jednotlivých částí, ze kterých je složena.

1.1 Optické prvky kamery

Jedná se o všechny části kamery, kterými prochází světlo předtím, než dopadne na světlocitlivý prvek. Světlo se na těchto prvcích může lámat (čočky), zastavit (clona) nebo měnit svoje vlastnosti (filtry). Všechny prvky objektivu jsou přitom v optické rovině a je snahou, aby nedocházelo k žádným deformacím obrazu kvůli špatné osové souměrnosti či samotným vadám jednotlivých prvků. [1]

1.1.1 Objektiv

Hlavním úkolem objektivu je usměrnění paprsků a projekce zmenšeného obrazu snímané scény na fotocitlivý prvek. Důležité přitom je, aby byl zmenšený obraz bez jakýchkoliv rušivých elementů.

Objektiv je složen z několika čoček a dalších částí, které jsou opticky centrované, tedy jsou všechny v jedné optické ose. Jednotlivé části objektivu se vůči sobě pohybují a tím mění optické parametry soustavy. To se děje při ostření či zoomování, kdy se mění ohnisková vzdálenost. [1]

Ohnisková vzdálenost

Ohnisková vzdálenost nám udává, v jaké vzdálenosti se zobrazí objekt ležící před čočkou v nekonečnu. Udává se v milimetrech a má značku ***f***. Obecně platí, že čím kratší je ohnisková vzdálenost, tím je úhel záběru objektivu větší. Vzorec pro výpočet ohniskové vzdálenosti jedné čočky je poté:

$$\frac{1}{f} = \left(\frac{n_2}{n_1} - 1\right) * \left(\frac{1}{r_1} + \frac{1}{r_2}\right) \quad (1)$$

kde:

f = ohnisková vzdálenost

n_1 = index lomu okolního prostředí

n_2 = index lomu materiálu čočky

$r_{1,2}$ = poloměr křivosti lámavých ploch

Při výpočtu musíme dodržovat znaménkovou konvenci. Poloměr křivosti vypuklých optických ploch uvažujeme s kladným znaménkem, poloměr křivosti dutých optických ploch se záporným znaménkem. Tudíž spojka bude mít vždy kladnou ohniskovou vzdálenost a rozptylka zápornou. Index lomu je bezrozměrná veličina. [1]

Ohniskovou vzdálenost některých objektivů můžeme plynule měnit, zařízení, které k tomuto účelu slouží, nazýváme transfokátor. Ohniska rozdělujeme:

- Pevné ohnisko – ohnisková vzdálenost nastavena pevně při výrobě
- Proměnné ohnisko – ohniskovou vzdálenost můžeme ručně nastavovat otáčením části objektivu
- Elektronicky řízené ohnisko – ohnisková vzdálenost se nastavuje elektronicky pomocí motorku. [1]

Optická mohutnost čočky

Pro popis čoček používáme pomocnou veličinu, a to optickou mohutnost čočky. Označujeme ji $\boldsymbol{\varphi}$ a její jednotkou je dioptrie se značkou D. Definovaná je jako převrácená hodnota ohniskové vzdálenosti čočky. Obdobně jako u ohniskové vzdálenosti, spojky mají kladnou hodnotu a rozptylky zápornou. [1] Vzorec je pak:

$$\varphi = \frac{1}{f} \tag{2}$$

Světelnost objektivu

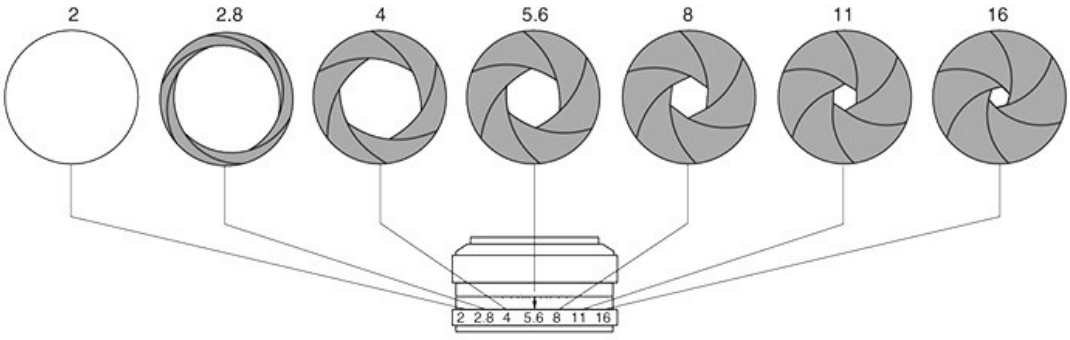
Vyjadřuje, kolik světla je objektiv schopen využít z dopadajícího světla a soustředit ho do vytvářeného obrazu na fotocitlivém prvku. Hodnoty jsou udávány clonovým číslem $\boldsymbol{k}$, které je bezrozměrné. Čím je clonové číslo menší, tím je lepší světelnost, ale zhoršuje se

ostrost. Clonové číslo je udáváno pro objektivy, a to jak pro ty s pevnou ohniskovou vzdáleností, tak s proměnnou (zoomem). Hodnoty potom dosahují u kvalitních objektivů s pevnou ohniskovou vzdáleností kolem 1,8. U objektivů se zoomem pak od 3,5 do 5,6. [2]

1.1.2 Clona

Clona bývá realizována mechanicky a je tvořena z kovových lamel, které regulují množství dopadajícího světla na fotocitlivý prvek, mění tedy světelnost objektivu. Kovové lamely clony jsou uspořádány tak, že mění průměr otvoru pro průchod světla. Míra zaclonění je udávána jako bezrozměrné číslo ***k*** označené jako clonové číslo. Udává poměr ohniskové vzdálenosti k vstupnímu otvoru clony a je dána geometrickou řadou následujících čísel:

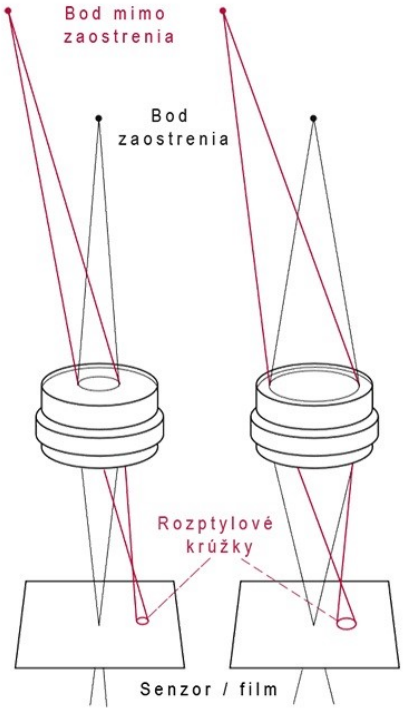
1; 1,4; 2; 2,8; 4; 5,6; 8; 11; 16; 22; 32 [2]



Obr. 1. Zavírání clony při změně clonového čísla [3]

Každé vyšší číslo způsobí to, že na světlocitlivý prvek dopadne světlo o poloviční intenzitě. Na obr. 1. je vyobrazeno, co se děje se vstupním otvorem při změně clony. [1]

Další věcí, kterou clona ovlivňuje, je hloubka ostrosti. Je to způsobeno tím, že zmenšujeme velikost vstupního otvoru pro světlo a tedy zužujeme kužel paprsků, který by se jinak rozprostřel po větší ploše světlocitlivého prvku. To můžeme vidět na obr. 2. [2]



Obr. 2. Hloubka ostrosti pole v závislosti na otevření clony [3]

1.1.3 Další příslušenství

Následující prvky jsou nadstandardní prvky pro fotoaparát či videokameru, ale jejich používání nám poskytuje jisté výhody.

Sluneční clona

Zabraňuje pronikání bočního světla do objektivu, interakci světla uvnitř a vytváření optických vad barevného, či světelného charakteru. Jedná se o nástavec, který se namontuje na přední část objektivu. [1]

Předsádková čočka

Upevňuje se na přední část objektivu a slouží pro makro fotografování. Principiálně je předsádková čočka lupa, která nedovolí zaostřit do nekonečna. Zaostřit lze pouze do vzdálenosti odpovídající převrácené velikosti hodnoty předsádkové čočky. Udává se v dioptriích a jsou vyráběny převážně v řadách 1D, 2D, 4D a 10D, kdy je možné je navzájem různě kombinovat. Pokud máme tedy čočku o velikosti 4D, maximální vzdálenost, na kterou dokážeme zaostřit, je 25 cm. [1]

Filtry

Filtry slouží k upravení určitého spektra světla a pomáhají nám docílit mnohdy nenahraditelných fotografií, kterých bychom ani v počítači nedokázali docílit. Používanými filtry jsou: UV a IR filtry, polarizační, barevné, přechodové, šedé a jiné. Šroubují se do přední části objektivu. [4]

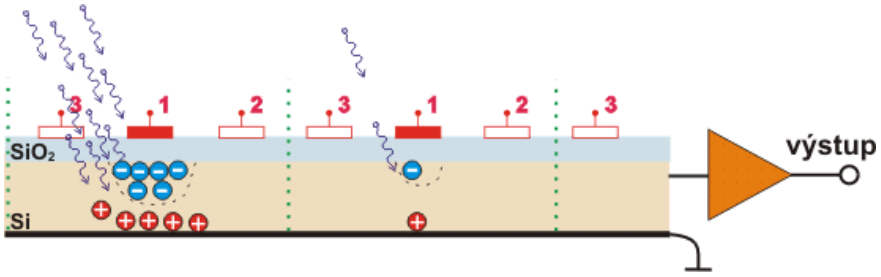
1.2 Snímání obrazu

Jedná se o hardwarové vybavení kamery, které převádí dopadající světlo na signál elektrické podstaty. V následujících bodech bude vysvětlen princip dvou nejpoužívanějších čipů pro snímání obrazu, a to CCD čipu a CMOS čipu.

1.2.1 CCD

Zkratka CCD znamená Charge-coupled Device (zařízení pracující s propojeným nábojem). Tím je myšleno to, že signál generovaný na čipu je propojený a jeho vyčítání má spojitý charakter.

Nejdříve dochází k expozici CCD čipu a uložení náboje do jednotlivých buněk (pixelů) CCD čipu. Zde je důležité si uvědomit, že pixel je bod obrazce už zpracované datové informace, zatímco buňka je hardwarová část čipu, kde se převádí viditelné světlo na elektrickou podstatu. Dopadající světlo o určité energii *e* vyrazí ze substrátu volné elektrony, které se začnou shlukovat u kladné elektrody, zatímco atomy s odtrženými elektrony (díry) se začnou shlukovat na záporné elektrodě. Minimální energie fotonu, při které dojde k uvolnění elektronu z křemíku, je 1,2 eV. To odpovídá $\lambda = 1033$ nm (infračervené oblast spektra). Ve viditelném spektru je spodní hranice 1,6 eV odpovídající $\lambda = 700$ nm (červená) a vrchní hranice viditelného spektra 3,4 eV, což odpovídá fialové barvě o $\lambda = 400$ nm. Energie fotonu nám určuje, kolik elektronů dokáže jeden foton vyrazit z křemíkového substrátu. Zatímco nízké energie vyrazí pouze 1 elektron, u vysokých energií to mohou být jednotky až desítky elektronů. [5][6]

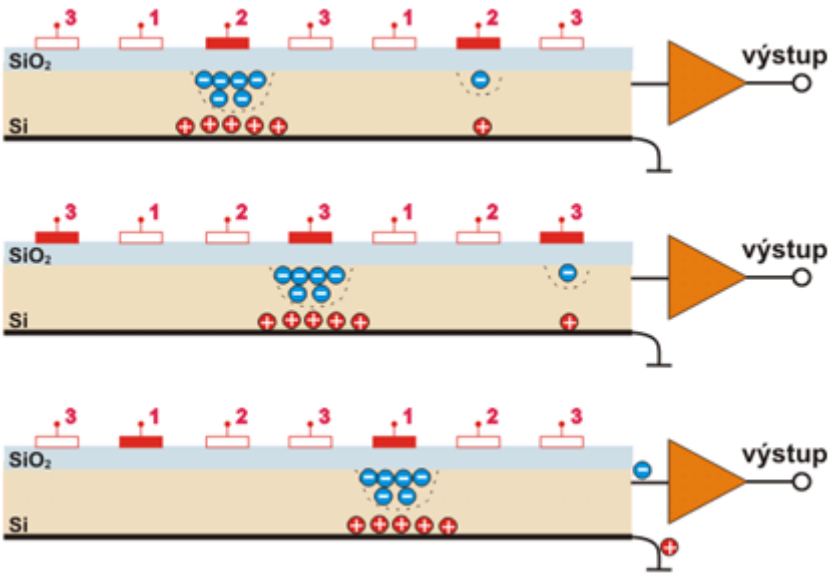


Obr. 3. Dopad světla na CCD čip [7]

1.2.1.1 Princip vyčítání pixelů po řádcích

Po skončení osvitu CCD čipu nastává proces převedení naakumulovaného náboje na elektrický signál, který je zpracováván procesorem na výsledný obraz.

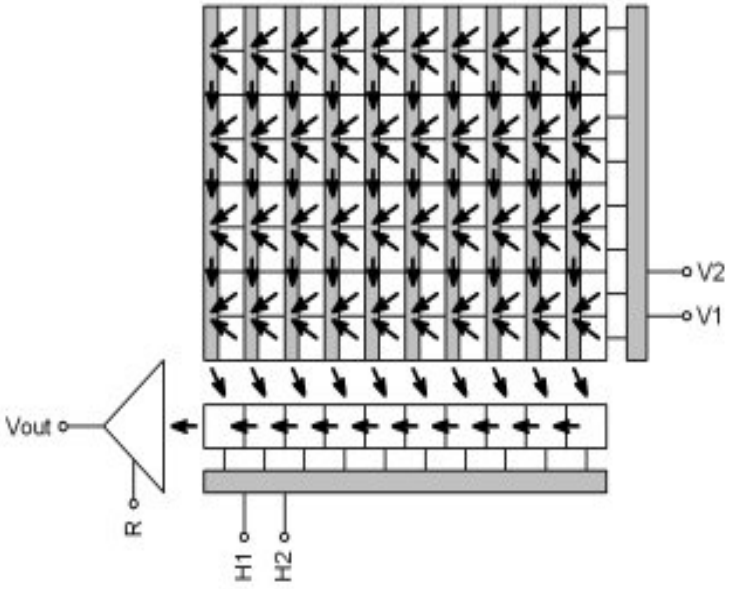
V první fázi nastává přesun elektronů pomocí přepínání napětí na jednotlivých elektrodách. Tyto elektrody jsou označeny čísly 1, 2, 3. V první fázi (obr. 3.) jsou elektrony soustředěny u elektrody č. 1, na které je napětí (reprezentováno červenou barvou elektrody). Následně je přivedeno napětí na elektrodu č. 2 a odebráno napětí z elektrody č.1. Elektrony se nám posunou směrem k druhé elektrodě. Tímto principem postupujeme dále s elektrodami č. 2 a 3. A následně 3 a 1. Pokud dojdou elektrony na konec řádku (sloupce), je záporný náboj odebrán operačním zesilovačem a převeden na hodnotu napětí. Tento princip je popsán na obrázku 4. [6][7]



Obr. 4. CCD princip vyčítání náboje z pixelů [7]

1.2.1.2 Prokládané snímače

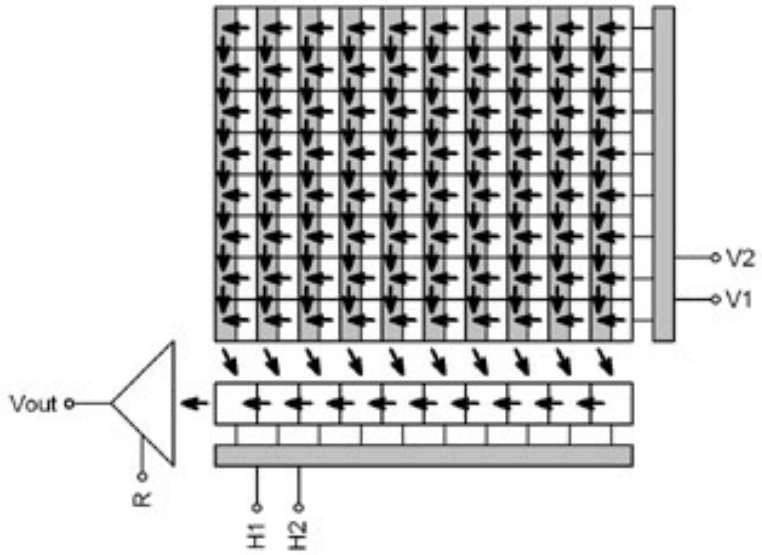
Jedná se o způsob vyčítání obrazu využívaný pro kamerové aplikace. Princip je takový, že se vždy vyčítají liché řádky a následně sudé. Tím ušetříme část datové kapacity a zrychlíme frekvenci půlsnímků. Při složení do videa dojde k posunu scény na lichých snímcích oproti sudým, čímž dostaneme plynulejší obraz na úkor ostrosti (ve složeném obraze). Můžeme použít elektronickou uzávěrku. Jednotlivé půlsnímky jsou poté díky vyčtení do registru ostré a nedochází na nich k nežádoucím efektům způsobeným vyčítacím procesem. [6]



Obr. 5. Prokládané snímače CCD [6]

1.2.1.3 Progresivní čtení

K vyčítání celého obrazu dochází najednou, všechny buňky jsou v jednom okamžiku vyčteny do posuvných registrů, kde jsou dále zpracovávány, zatímco už můžeme snímat další snímek. Výhoda je v možnosti rychlé elektronické uzávěrky, v ostrosti obrazu, kde nedochází vlivem vyčítání buněk k nežádoucímu rozmazávání, a ve frekvenci snímkování. [6]

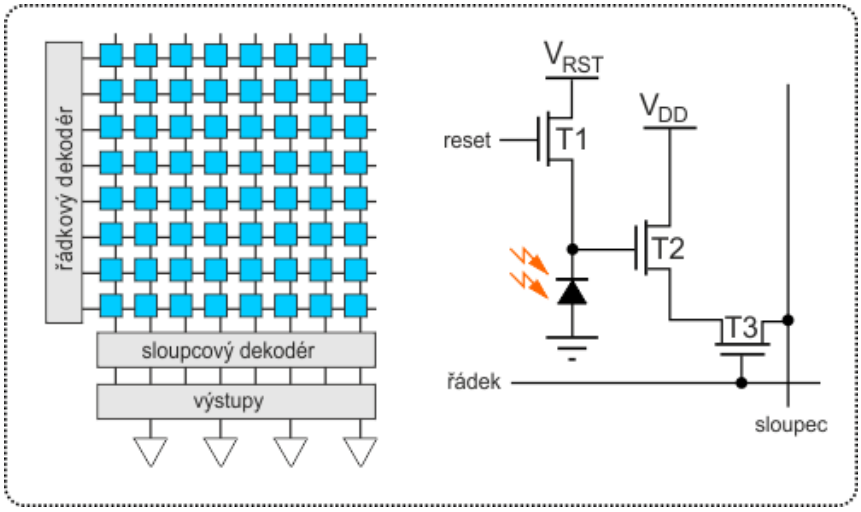


Obr. 6. Progresivní čtení u CCD [6]

1.2.2 CMOS

CMOS znamená Complementary Metal-Oxide Semiconductor. CMOS čip pracuje na podobném principu jako CCD čip, ovšem způsob vyčítání je rychlejší a hlavním rozdílem je hardwarové uspořádání samotného čipu. Oproti CCD čipu jsou všechny obvody pro snímání i vyčítání integrovány na jedné plošné desce křemíku. To má za následek zlevnění samotného čipu, ale za cenu zmenšení fotocitlivé plochy čipu. Pro redukci tak velkého útlumu světelného zisku jsou nad všechny jednotlivé buňky čipu umístěny mikroskopické čočky, které soustřeďují světelný tok přímo na světlocitlivou plošku buňky. [6][8]

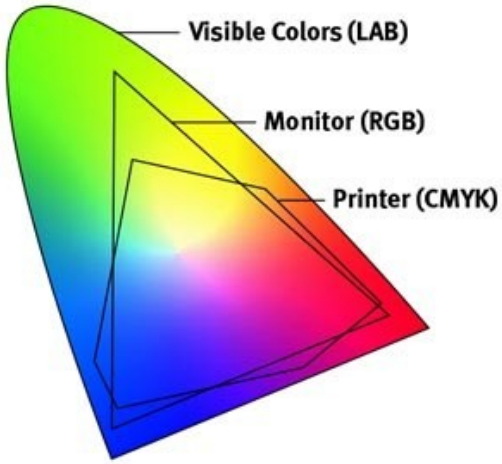
Na obr. 7. můžeme vidět na levé straně matici buněk připojenou na řádkový a sloupcový dekodér, to nám zajistí přístup na jednotlivé buňky čipu přímo. Na pravé straně můžeme vidět zapojení samotné buňky čipu. Světlocitlivý prvek je realizován světlocitlivou diodou, tranzistor T1 pak slouží pro reset buňky (vymazání hodnoty z buňky). Tranzistor T2 funguje jako zesilovač signálu pro diodu a tranzistor T3 slouží k identifikaci buňky. Právě kvůli integraci všech těchto obvodů na jednu vrstvu čipu dochází k velkým ztrátám světlocitlivé plochy. [8]



Obr. 7. CMOS čip [8]

1.3 Barevné modely

Barevné modely se používají pro zobrazování barevného obrazu. Nejčastěji se setkáme s modely RGB a CMYK. RGB slouží pro vyobrazování barev ze světlo-vyzařovacích ploch a naopak CMYK slouží pro tištěnou formu. Následně si rozebereme jednotlivé barevné modely. Na obrázku 8 můžeme vidět barevnou škálu lidského oka a obou modelů RGB a CMYK.



Obr. 8. Porovnání barev RGB a CMYK [10]

1.3.1 RGB

RGB model je pojmenovaný po prvních písmenech barev v angličtině; Red (červená), Green (zelená) a Blue (modrá). Také se jim říká základní barvy, jelikož z těchto barev dokážeme v určité barevné škále, vyobrazené na obr. 9., vytvořit odstíny velké spousty barev.

Omezením je pouze kvalita zobrazovacích prvků a také to, že tento model lze použít jen na zobrazovací prvky, které aktivně vyzařují světlo. Dochází zde tedy k míchání barev tak, že čím víc barev složíme dohromady, tím světlejší se nám zdají, a tedy se přibližují bílé barvě. Tento systém se nazývá aditivní způsob a míchá vyzařované světlo. [9]

Pro přirovnání, světlo od Slunce se nám zdá bílé, ale když ho rozložíme na spektrum pomocí hranolu, zobrazí se nám velká škála barev přes fialovou po červenou. Když je spojíme zase dohromady pomocí dalšího hranolu, bude světlo zase bílé.

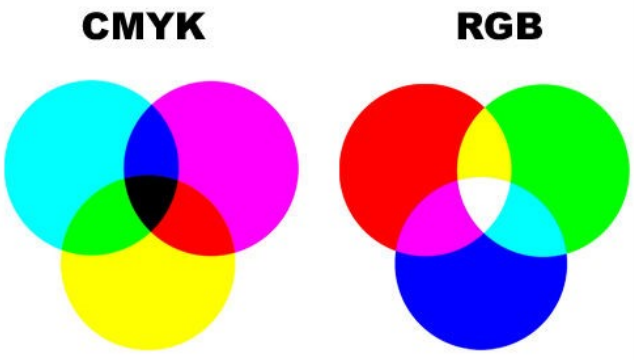
Obecně platí, že RGB model dokáže vyobrazit pestřejší barvy než samostatný CMYK model. RGB model se při 8bitové hloubce dělí tak, že každá barva je reprezentována hodnotami 0 až 255 a jejich mícháním dostáváme patřičné odstíny. [9]

1.3.2 CMYK

Název CMYK je z anglických slov Cyan (azurová), Mangeta (purpurová), Yellow (žlutá) a Key (černá). Je to barevný model užívaný pro tisk, jelikož na rozdíl od RGB modelu, kdy je nutné vyzařovat světlo, při CMYK modelu omezujeme barevné spektrum odrážené od povrchu. Tento styl míchání barev se nazývá subtraktivní míchání barev. Základní barvy jsou tedy světlejší a jejich mícháním dostáváme barvy tmavší (obr. 9.). [9]

V ideálním případě by nám stačily 3 barvy, ale smícháním všech tří barev nedostaneme čistě černou barvu, ale tmavou hnědošedou. Proto je k tomuto modelu přidána čistě černá barva. Pro ještě věrohodnější barvy jsou přidávány další odstíny barev, které doplňují tento model, a nešlo by je jinak realizovat. Pro profesionální aplikace může být tisk i pomocí 12 a více barev a tedy bude dosaženo reálnějšího vyobrazení obrazu. [9]

CMYK model je řešený pomocí procentuálního zastoupení jasu jednotlivých barev. Tedy od 0 do 100 procent. [9]



Obr. 9. CMYK a RGB model [9]

1.3.3 Převod RGB na CMYK

Pro převod RGB modelu na CMYK model můžeme použít následující pravidla, přitom uvažujeme, že barevná hloubka je 8bitová a tedy je RGB v rozmezí 0 až 255. Přitom CMYK je v % v rozmezí od 0 do 100. [9]

V prvním kroku si převedeme hodnoty RGB na pomocné proměnné R_C, G_C a B_C které jsou v rozmezí 0 až 1. [9]

$$R_C = \frac{R}{255} \tag{3}$$

$$G_C = \frac{G}{255} \tag{4}$$

$$B_C = \frac{B}{255} \tag{5}$$

Dalším krokem je vypočítat hodnotu černé, abychom mohli přes ni dopočítat hodnoty ostatních tří barev (rovnice č. 6). Nebo, pokud máme model pouze CMY, použijeme vzorec z rovnice č. 7. [9]

$$K = 1 - \max(R_C, G_C, B_C) \tag{6}$$

$$K = 0 \tag{7}$$

Posledním krokem je dopočítání hodnot barev CMY, přičemž dopočítáváme barvy z jejich přirozených protějšků, tedy C z R (rovnice č.8), M z G (rovnice č. 9) a Y z B (rovnice č. 10). [9]

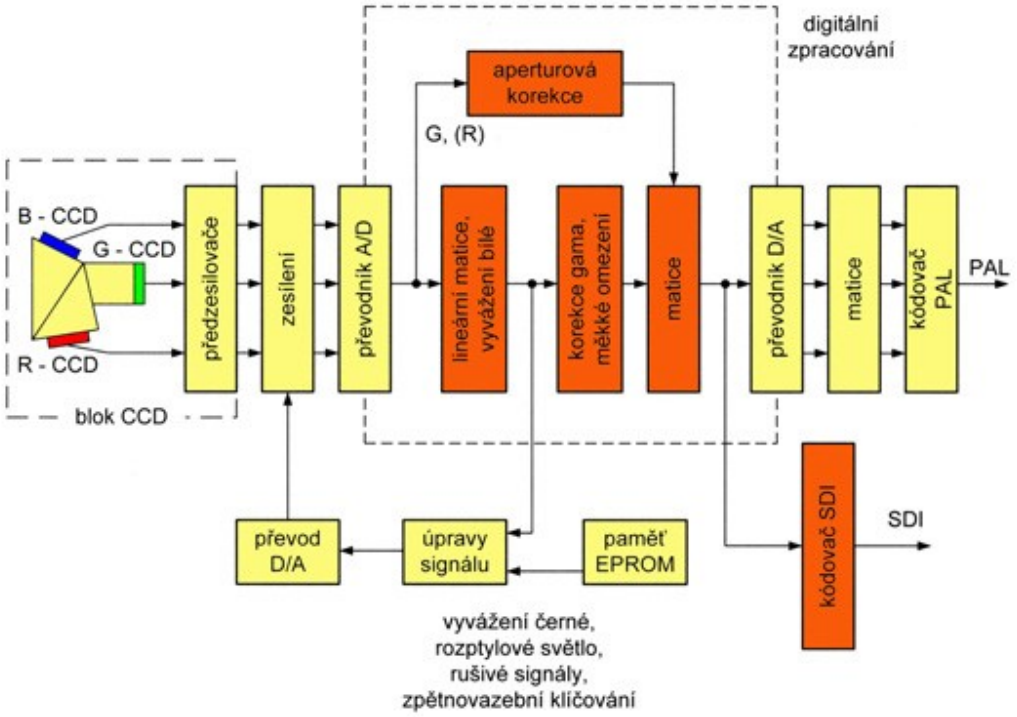
$$C = \frac{1 - R_C - K}{1 - K} \tag{8}$$

$$M = \frac{1 - G_C - K}{1 - K} \tag{9}$$

$$Y = \frac{1 - B_C - K}{1 - K} \tag{10}$$

1.4 Elektronická část kamery

Hodně důležité je mít kvalitní optickou část kamery a snímací prvek. Neméně důležitou částí je ovšem i elektronická část kamery, která zajišťuje vlastní zpracování obrazu, ovládání korekcí a uložení, případně přenos obrazu.



Obr. 10. Blokové schéma zpracování signálu v kameře [11]

Na obr. 10. je vyobrazeno blokové schéma barevné videokamery. Jednodušší kamery nemusí mít všechny části vyobrazené a popsané na tomto obrázku. Pro některé aplikace není potřeba takové složitosti HW a SW, která se mimo jiné krom kvality odráží i na výsledné ceně.

Nejprve světlo dopadá na CCD (CMOS) čip, kde je světlo převedeno do analogové podoby. Zde jsou také signály zesíleny v předzesilovači pro jednodušší zpracování. Následně dochází k dalšímu zesílení signálu, ale zde už je zesílení ovládáno zpětnovazebně. Kompenzuje se tedy množství dopadajícího světla, zesílení jednotlivých barevných složek a potlačení rušivých vlivů. Dochází zde k částečnému vyvážení bílé a barevnosti obrazu. [11]

V další části se analogový signál převádí pomocí A/D převodníku na digitální signál. V tomto bloku probíhají matematické operace, které upravují výsledný obraz. Dochází zde k vyvážení bílé, gama korekci, udržování úrovně černé pro CCD čip a měkkému omezení.

Také zde probíhá aperturová korekce, která upravuje zelenou složku světla a zahlazuje částečně signálové přechody. Tím se sice sníží rozlišovací schopnosti, ale zmenší se objem dat. [11]

Vyvážení bílé

Obvody automaticky vyhodnocují a nastavují amplitudu signálů v červeném a modrém kanálu vůči zelenému. To umožňuje plynulé přecházení z venkovního prostředí do vnitřního a do různých druhů osvětlení (žárovka, zářivka). Vyvážení bílé se provádí v průběhu snímání scény v digitální části a ovlivňuje vstupní analogovou část. [11]

Korekce gama

Gama korekce je matematická operace, která upravuje rozložení tmavých a světlých oblastí a jejím cílem je zoptimalizovat množství informace uložené v tmavých oblastech. Světlocitlivý prvek kamery snímá množství světla lineárně, tedy 3x více fotonů vybudí 3x větší napětí. Lidské oko oproti tomu funguje nelineárně a změny při slabém osvětlení vnímá intenzivněji, než změny při silném osvětlení. Pro kompenzaci tohoto rozdílu a přiblížení obrazu tomu, jak jej vnímá lidské oko, upravuje dynamický rozsah a bitovou hloubku tak, že tmavší barvy zabírají větší bitový prostor než světlejší barvy. [12]

Udržování úrovně černé

Využívají se k tomu okrajové buňky CCD (CMOS) čipu, které jsou zastíněny maskou uvnitř fotoaparátu a slouží jako referenční hodnoty pro všechny RGB kanály za pochodu. Kompenzuje se zde tepelný šum, který je zvláště při vyšších teplotách silnější, a nastavuje hranice černé barvy, aby bylo podání černé co nejvěrohodnější. [11]

Měkké omezení

Pracuje v rozmezí vrchních 20 % obrazového signálu, kde se obvod snaží vykompenzovat přesvícené a jinak jasné oblasti. Nejvyšší signálové amplitudy jsou komprimovány, ale stále si zachovávají rozlišitelné obrazové podrobnosti. [11]

Dále je zpracovaný obraz převeden buď na standard SDI, kde je signál v digitální podobě, nebo se převede zpátky na analogovou podobu a přenos je podle standardu PAL.

2 PRINCIP ČINNOSTI VÝVOJOVÝCH PLATFOREM

Vývojové platformy jsou hardwarové zařízení s vlastním softwarovým vybavením určené pro programování, které umožňují testování, vývoj, ladění a připojování různých periferních zařízení. Jedná se většinou o desku plošných spojů, kde je už vše integrované na jedné desce a stačí ji pouze připojit k počítači a začít programovat. V následujících podkapitolách bude obecně vysvětleno, jak takové platformy vypadají, co nejčastěji obsahují a jaké periferie lze na ně připojovat.

2.1 Druhy vývojových platforem

Vývojových platforem je na trhu nepřeberné množství od velmi známých a používaných až po speciální druhy, které jsou využívány na specifické aplikace. Základní účel je ale pro všechny stejný, a to je možnost ovládat, řídit a připojovat periferní zařízení.

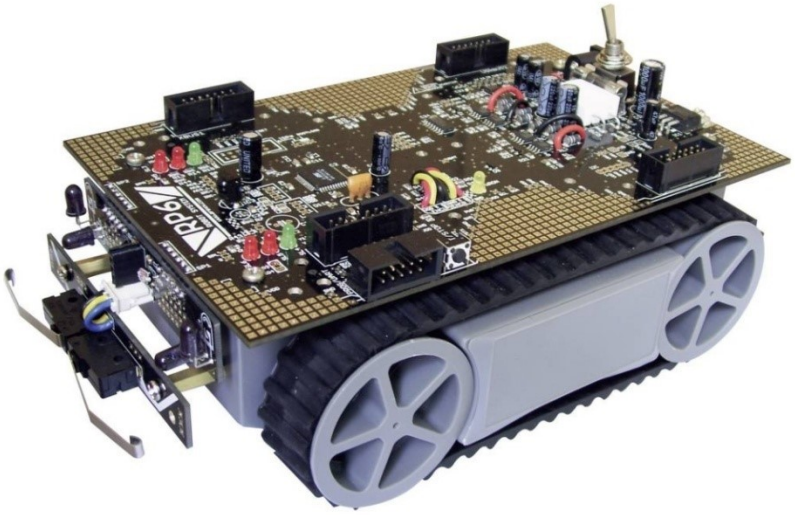
2.1.1 Roboty

V oblasti vývojových platforem je robotem myšleno hotové zařízení, se kterým už není potřeba žádné další manipulace a stačí jej připojit k PC a naprogramovat. Je to velmi dobrý systém pro začátečníky, kteří si mohou vyzkoušet všechny periferní zařízení připojené (nebo připojitelné) k robotu. Nejčastěji je takový robot ve formě auta, pojízdného nebo pochodujícího zařízení. Tento systém je velmi jednoduchý na ovládání a programování, ale má omezené použití a nedá se „využít“ na jiné aplikace, než ke kterým byl původně vytvořen.

Zástupce z této kategorie může být například RP6 V2 C-Control (obr. 11.). Tento robot je pásového typu, kdy je sestavený a už se s ním nemusí nic dělat. Využívá procesor ATMEGA32, což je 8bitový mikrokontroler. Obsahuje 32 kB flash paměť a 2 kB SDRAM, taktovací frekvence procesoru je 8 MHz. Obdobné mikrokontrolery jsou využité i v programovatelných platformách. Programování probíhá v jazyce C a je možné použít libovolné programovací prostředí podporující GCC (GNU Compiler Collection). [14] Lze také stáhnout velké množství příkladových programů, které vysvětlují chování jednotlivých periferií. Připojení je pomocí rozhraní USB. [13]

Obsahuje infračervené senzory, tlakové senzory, světlocitlivý senzor, 6 stavových LED, otáčkoměry a elektromotory pro pohyb robota, IR komunikátor a baterie. Přes I2C sběrnici je možné připojit další periferní zařízení a ty je možné „stackovat“ na sebe v přední a zadní

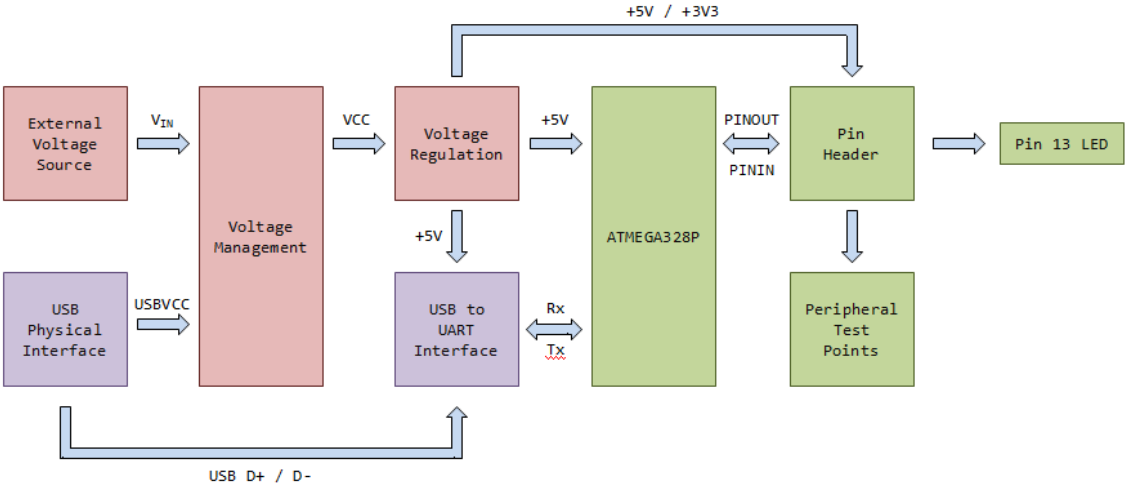
části robota. Efektivní doba použitelnosti robota je kolem 4 hodin. Napájet se dá z baterií, nebo přídavných akumulátorů, které jdou připojit pomocí napájecího konektoru. Napájecí konektory jsou 2 x 5 V a 1 x 7,2 V. [13]



Obr. 11. Pojízdný robot RP6 V2 C-Control [13]

2.1.2 DPS s integrovanými vstupně výstupními konektory

Do této kategorie spadá největší množství vývojových platforem, jelikož jsou velmi multifunkční a snadně obsluhovatelné. Na obr. 12. je vyobrazené blokové schéma vývojové platformy Arduino UNO. Základem tohoto typu vývojových desek je mikrokontroler (zde ATMEGA328P), který ovládá veškeré procesy na desce. Na blokovém schématu na pravé straně jsou vstupně výstupní piny, které jsou programovatelné a jsou zároveň napájené 5V. Dále jsou tu piny pro GND, 5 V a 3,3 V. Je tu připojený také speciální pin 13, který funguje jako stavová LED. Při přivedení logické 1 se LED rozsvítí a přivedením logické 0 zhasne. Levá strana blokového schématu slouží jako komunikační a napájecí rozhraní. Napájení může probíhat z externího zdroje (adaptéru) nebo z rozhraní USB. Právě rozhraní USB je použito i pro propojení s PC při programování, a tudíž při ladění a psaní programu není potřeba desku napájet externě. Externí napájení může být v rozmezí 7 až 12 V, kdy si napájecí rozhraní samo upraví potřebné napětí pro správný chod desky. Poslední částí je konvertor USB na univerzální sériové UART rozhraní. [15]



Obr. 12. Arduino UNO – blokové schéma vnitřního zapojení [15]

2.1.3 Programovatelné mikrokontrolery

Jedná se programovatelné integrované obvody, které slouží pro jednoúčelové aplikace. Jsou vyráběny v podobě pouzdra s mnoha piny, kde je uvnitř integrované vše potřebné pro chod takového zařízení. Obsahuje minimálně CPU, operační paměť RAM, paměť pro program (EEPROM, FLASH) a vstupně výstupní porty. Dále může obsahovat další nadstandardní prvky, jako jsou například: vnitřní oscilátor, sériové a paralelní sběrnice, A/D a D/A převodníky, komparátory, časovače a další prvky. [16]

Programování těchto mikrokontrolerů je pro zkušenější programátory, jelikož programování probíhá v programovacím jazyce assembly language. V širší veřejnosti i mezi programátory se jazyk pro programování tohoto typu označuje jako assembler. Jedná se o nízkoúrovňový programovací jazyk, který pracuje přímo se stavy registrů a přímým přístupem k procesoru a paměti. Vytváří se zde tedy strojový kód na konkrétní typ procesoru, který je napsaný pomocí strojových instrukcí. Program napsaný pro jeden typ procesoru je tedy nefunkční pro jiný typ. [16]

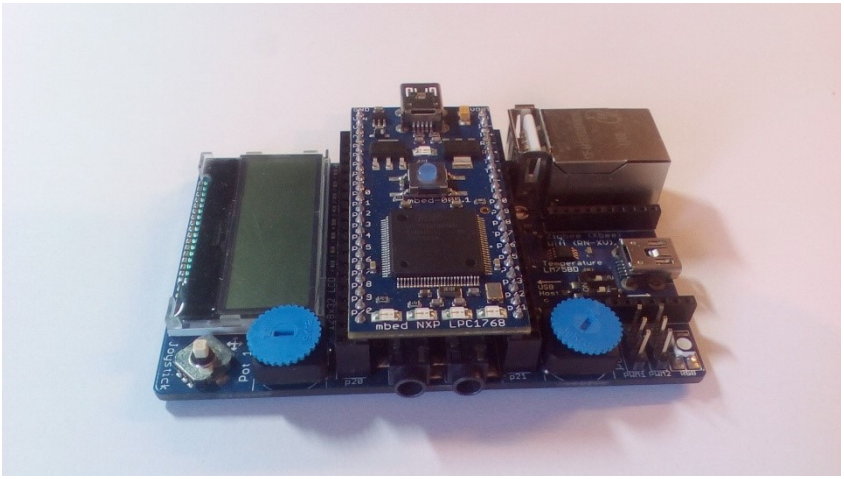
Pro jednoduché aplikace je to velmi výhodný systém, jelikož připojíme potřebné zařízení a naprogramujeme přesně to, co potřebujeme. Takové zařízení má zpravidla daleko menší spotřebu energie a je levnější.

2.1.4 Shield

Jedná se o nadstavbu vývojových platforem, která přináší určité výhody. Mnohdy jsou tyto nadstavby řešené formou DPS s paticí na zasunutí do vývojové platformy, nebo opačně

samotné vývojové platformy zasunuté do shieldu. Výhodou je, že například displej s klávesnicí nemusíme připojovat externě a vést drátově, ale stačí zasunout shield a vše je už integrované na připojené desce. Tato rozšiřující deska bývá dost často stejně veliká jako deska vývojové platformy. Díky tomu můžeme zapojit takto nad sebe i několik rozšiřujících desek. Ovšem nejde takto rozšiřovat pořád dál a dál. Vycházíme z množství pinů na primární desce, které ovládáme, a které pouze „propůjčíme“ zapojenému shieldu. Pokud máme 12 vstupně výstupních pinů a 10 z nich použijeme pro ovládání motorků, připojením dalšího shieldu nedostaneme dalších 12 pinů, ale musíme uvažovat, že těch 10 už je použitých a kdybychom je chtěli využít, nepracovalo by to korektně. Docházelo by ke kolizím a přepisování stavů.

Je velké množství zařízení, která můžeme takto zapojit. Pro představu to můžou být: displeje, zobrazovače, expanzní desky, výukové moduly, konvertory a převodníky, GPS a GSM moduly, ethernet shield a další. Na obr. 12. je vidět shield rozšiřující desky pro platformu Mbed se zasunutou vývojovou platformou Mbed NXP LPC1768.



Obr. 13. Mbed shield [autor]

2.2 Vnitřní elektronické části vývojových platforem

Pro snazší pochopení programování a toho, jak vývojové platformy fungují, si v této kapitole rozebereme jednotlivé části vývojových platforem a součásti, z kterých se skládají. Uvedeme nejčastější používané součásti a vysvětlíme si, v čem se liší a jaký podíl na celku mají.

2.2.1 Mikroprocesor

Mikroprocesor tvoří mozek celého systému. Zpracovává instrukce nahraného programu a obhospodařuje periferie. Důležité je si uvědomit, že zpracovává instrukce programu, jak jsou zapsané za sebou. To je dobré vědět při psaní instrukcí a počítat s tím. Mikroprocesor se může dělit podle mnoha kritérií, například taktovací frekvence, vnitřní šířka dat a podle programové výbavy. Jsou dva hlavní typy mikroprocesoru a to jsou procesory koncepce RISC a CISC.

RISC – Reduced Instruction Set Computer

U tohoto typu procesoru je programová výbava zredukována na nejčastěji používané instrukční sady. Díky tomu je ušetřena instrukční paměť a čas zpracování výsledného programu. Jednotlivé instrukce jsou zřetězovány za sebou a zpracování každé instrukce probíhá v jednom taktu procesoru. Má to za následek delší program než u CISC, ale výsledný čas zpracování trvá kratší dobu. [17]

CISC – Complete Instruction Set Computer

Mikroprocesory tohoto typu mají co nejúplnější instrukční sadu. Některé instrukce mohou být tak složité, že velikostně jsou jako celé bloky programu. Výhodou těchto mikroprocesorů je jednodušší kompilace programu do strojového kódu. Vykonání těchto instrukcí může trvat i několik cyklů, což může mít za následek zpomalování chodu mikroprocesoru. [17]

Stavba a základní části využívané mikroprocesorem:

ALU (Aritmeticko – logická jednotka) – probíhají zde hlavní výpočty procesoru.

Matematický koprocesor – pro výpočet desetinných míst.

Řadič – řídící jednotka obhospodařující jednotlivé části.

Registry – vnitřní paměť mikroprocesoru.

Vnitřní šířka dat – udává, kolika bitový procesor je, tedy kolik instrukcí dokáže zpracovat v jednom taktu

2.2.2 Paměť

Paměť slouží jako zásobník veškerých informací systému. Procesor paměť využívá a čerpá z ní potřebné informace a zase do ní zpracované informace ukládá. Důležité je, aby paměť vyhovovala taktovací frekvenci mikroprocesoru a dokázala ukládat a posílat informace včas. Vkládání čekacích taktů by zpomalovalo celý systém. Rozdělit paměti můžeme podle způsobu použití a to na datové a programové a také podle fyzických vlastností zpracování. [17] Nejčastěji použité paměti jsou:

ROM – Read Only Memory

Umožňuje z paměti číst, ale nedovoluje ukládat informace. Ty jsou vloženy při výrobě.

PROM – Programmable ROM

Dovoluje jednorázově zapsat požadovaná data, ale nejdou posléze už změnit.

EPROM – Erassable PROM

Vymazání obsahu čipu probíhá pomocí UV záření, které vymaže paměť. To dovoluje opakované naprogramování paměti.

EEPROM – Electrically Erasable PROM

Stejné jako EPROM, akorát vymazání probíhá elektricky. Maže se po jednotlivých adresách, tedy není nutné vymazat celý obsah čipu.

FLASH

Nevolatilní (elektricky nezávislá) programovatelná paměť. Rychlý přístup k datům, ale mažou se celé bloky paměti, proto je rozdělená na menší sektory. Velká hustota dat na jednotku plochy.

RAM – Random Access Memory

Jediná paměť, kde se po odpojení elektřiny data vymažou. Velmi rychlá paměť, kde je přístup k datům stejně rychlý bez ohledu uložené pozice. Kvůli velké rychlosti se využívá jako paměť pro práci s mikroprocesorem.

2.3 Komunikační rozhraní

Komunikační rozhraní slouží pro zařízení přenosu dat mezi dvěma a více zařízeními nebo částmi systému mezi sebou. Pro tento účel slouží příslušné konektory, kabely a sběrnice. „Důraz je kladen převážně na datovou propustnost, univerzálnost, rozměry a kompatibilitu.“ [Mobilní robot jako pomůcka pro výuku programování mikropočítačů, Zmeškal 2016, s. 13] V dnešní době je pro komunikaci dvou a více zařízení velmi často využíváno rozhraní USB kvůli dobré kompatibilitě s ostatními zařízeními a velkou datovou propustností. Nevýhodou tohoto rozhraní je omezená délka přenosu. Pro průmyslové aplikace se používá rozhraní RS-232, které dovoluje několikanásobně delší přenos než přenos přes USB.

Tab. 1. Porovnání parametrů komunikačních rozhraní

Komunikační rozhraní	Typ	Počet vodičů	Datová propustnost	Maximální délka vodičů [m]	Napájecí napětí [V]
USB	1.1	4	1,5 MB/s	5	5
	2.0	4	60 MB/s	5	5
	3.1 Gen1	8	5 GB/s	3	5
	3.1 Gen2	10	10 GB/s	3	5
	3.2	10	20 GB/s	1	5
RS 232	-	9	115200 Bd (14,4 kB/s)	900 (při 2400 Bd)	5 - 15
USART	-	2	31,25 KB/s	5	5
I2C	-	2	420 KB/s	<1	5

2.3.1 USB

Jedná se o univerzální sériové rozhraní, které má v dnešní době zabudováno většina vyráběných zařízení. Výhody jsou rychlý přenos dat, zpětná kompatibilita se staršími verzemi tohoto rozhraní a možnost napájení zařízení přímo přes kabel. Nevýhodou je kratší délka vodičů, která je ale pro běžné použití plně dostačující.

Standard 1.1 a 2.0 – jsou použity 4 vodiče, z čehož jsou 2 datové – jeden pro příjem dat a druhý pro vysílání, napájení + 5 V a GND. [18]

Standard 3.1 Gen1 – nové označení pro USB 3.0. Využívá 8 vodičů, 2 jsou pro zpětnou kompatibilitu, 2 pro napájení a zbylé 4 tvoří dvojice kroucených vodičů pro přenos dat.

Rychlost přenosu je 5 GB/s. USB 3.1 Gen2 umožňuje rychlejší přenos dat (10GB/s), ale je nutné využít konektor typu C. [18]

Standard 3.2 – používá konektor typu C s oboustrannými 24 piny (12 na každé straně). Hlavní výhody jsou větší rychlost, až 20 GB/s a vyšší napájecí proud, který umožňuje napájet větší zařízení, například notebook. Využívá 2 vodiče pro napájení (v napájecím režimu 4) a 8 vodičů pro přenos dat. [19]

2.3.2 RS-232

Toto rozhraní pro přenos dat je hojně využito hlavně v průmyslu, kvůli lepšímu odstínění rušení a velkou použitelnou délkou vedení. Slouží pro plně duplexní sériovou komunikaci dvou propojených zařízení. Napětí může být od 5 do 15 V, ale pro průmyslové aplikace je vhodnější vyšší napětí kvůli odstupu signálu od šumu. Nejhlavnějšími vodiči jsou RxD pro odesílání dat, TxD pro příjem a GND. [17]

Rychlost přenosu se kromě „byte per second“ [B/s] může udávat i v baudech [Bd]. Značí počet bitů přenesených za sekundu, takže je jednotka totožná s „bit per second“ [bps]. Pro přepočet na B/s stačí hodnotu v baudech vydělit 8.

Efektivní délka vedení pro přenos při 9600 Bd je okolo 150 metrů. Pro sériovou komunikaci vývojových platforem s PC se často využívá právě tato hodnota. Kdybychom ovšem rychlost přenosu značně omezili, a to na 2400 Bd, dokázali bychom se dostat až na efektivní délku vedení 900 metrů. Obecně platí, že čím rychlejší přenos, tím kratší vedení lze použít. [17]

2.3.3 I2C

Sériová sběrnice využívající pouze dvou vodičů – jeden datový vodič pro posílání dat (SDA) a druhý vodič pro synchronizaci (SCL). Díky otevřeným kolektorům tranzistorů na sběrnici je snadné připojit další zařízení, mnohdy jsou také piny vyvedeny externě, aby bylo snazší připojení externích zařízení. Vysílání na této sběrnici probíhá tak, že je vyslána značka START, která je specifická hodnotami 0 a 1 na obou vodičích, které se běžně nevyskytují. Po ní je odvysílána adresa zařízení (7 bitů) a 1 bit, který označí typ následujícího přenosu (vysílání nebo přijímání). Přenášená zpráva je rozdělena na 8bitové slabiky. Po odvysílání dané zprávy následuje specifická značka STOP. Kontrola integrity dat není řešená hardwarově, ale je čistě na programu tuto sběrnici obsluhující, jakým způsobem zabezpečí bezchybný přenos dat. [17]

2.3.4 USART

Univerzální synchronní a asynchronní rozhraní pro přenos dat mezi zařízeními. Podporuje plně duplexní přenos. Přenos signálu probíhá po dvou vodičích, Tx pro odesílání dat a Rx pro příjem. Rychlost přenosu se nastavuje při programování a oproti RS-232 může být přenos dvakrát tak rychlý. Hodnoty můžeme nastavit v rozmezí od 1200 bps do 250 kbps. Na vývojových platformách toto rozhraní můžeme najít v podobě pinů, na které se lze připojit. Vysílání probíhá vysláním sekvence bitů, kde první startovací bit je logická 0 a za ním následuje sled bitů od nejnižší úrovně po nejvyšší. Datových bitů může být 5 až 9. Za sekvencí je jeden nebo dva stop bity úrovně logické 1. [17]

3 ZHODNOCENÍ KOMERČNĚ DOSTUPNÝCH PROSTŘEDKŮ PRO VÝROBU BEZPEČNOSTNÍCH KAMER

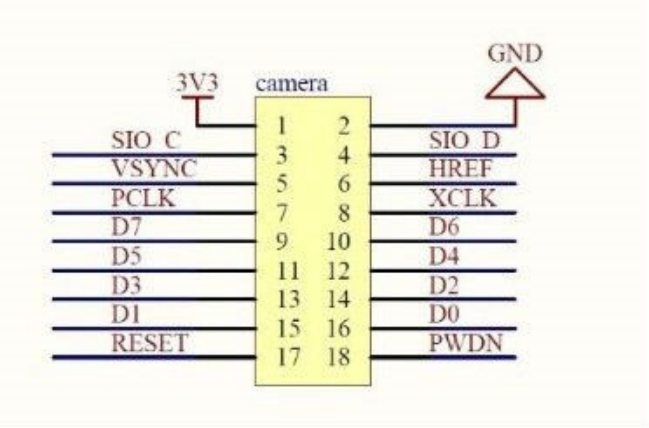
Tato kapitola bude zaměřena na zhodnocení současných prostředků využitelných pro výrobu bezpečnostních kamer. Zabývat se bude low-cost kamerovými moduly a vybranými vývojovými platformami, které jsou vhodné pro výukové účely a je na nich možné zprovoznit a naprogramovat kameru.

3.1 Kamerové moduly

Na trhu je velké množství různých kamerek a kamerových modulů, které jdou snáze či hůře naprogramovat. Také se liší způsobem komunikace, snadností připojení konektorů, rozlišovacími schopnostmi a dalšími kritérii. Následuje popis některých kamerek.

3.1.1 OV7670 without FIFO

Oba typy kamer a to OV7670 s FIFO buferem nebo i bez, mají značně podobné vlastnosti, liší se převážně rychlostí přenosu dat a počtem pinů kamerky. Dnes existuje velká spousta verzí tohoto druhu kamery, avšak všechny vesměs fungují stejně. Mění se počet pinů a novější verze dokážou zpracovávat obraz kvalitněji a rychleji. Největší rozdíl oproti kamerce s FIFO je ten, že všechny procesy jsou zpracovávány přímo připojenou platformou a tedy při pomalejší platformě dochází k značnému omezení výkonu. Jednotlivé výstupní piny můžeme vidět na obrázku 14. Na obr. 16. můžeme vidět kamerku bez FIFO bufferu. Kamerku s FIFO bufferem AL422 pak na obr. 17. Vysvětlení pinů je v tab. 2. v následující kapitole. [20]

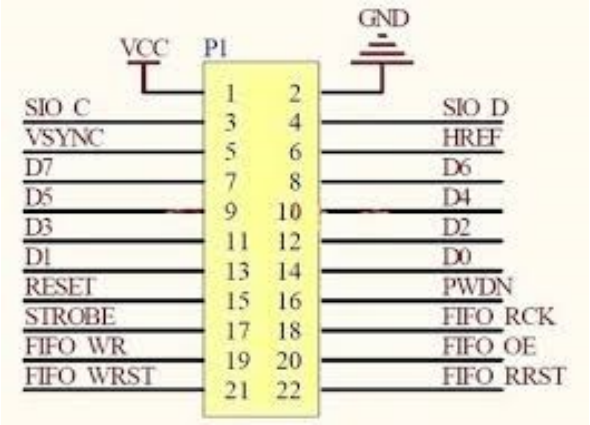


Obr. 14. Pinouts OV7670 without FIFO [22]

3.1.2 OV7670 with FIFO

Jak již bylo psané v předchozí kapitole, kamerka s FIFO bufferem je výhodnější pro pomalejší zařízení, jelikož značnou část obrazových výpočtů probíhá přímo v kameře. Tím pádem je zpracování dat rychlejší a hlavně u pomalejších platforem je značně viditelnější nárůst výkonu kamerky.

Na obr. 16. a obr. 17. můžeme porovnat obě dvě kamery a jednoduše můžeme určit, která z vyfocených kamer obsahuje FIFO buffer. Na zadní straně kamery, která je napravo, můžeme vidět velký čip s nápisem AL422B. To je právě FIFO čip kamery.



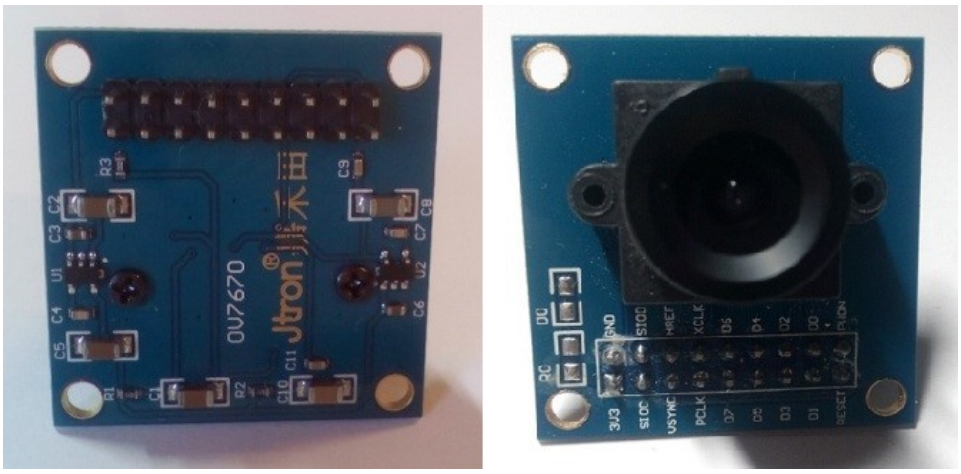
Obr. 15. OV7670 pinouts with FIFO [23]

Tab. 2. Popis významu jednotlivých pinů [20]

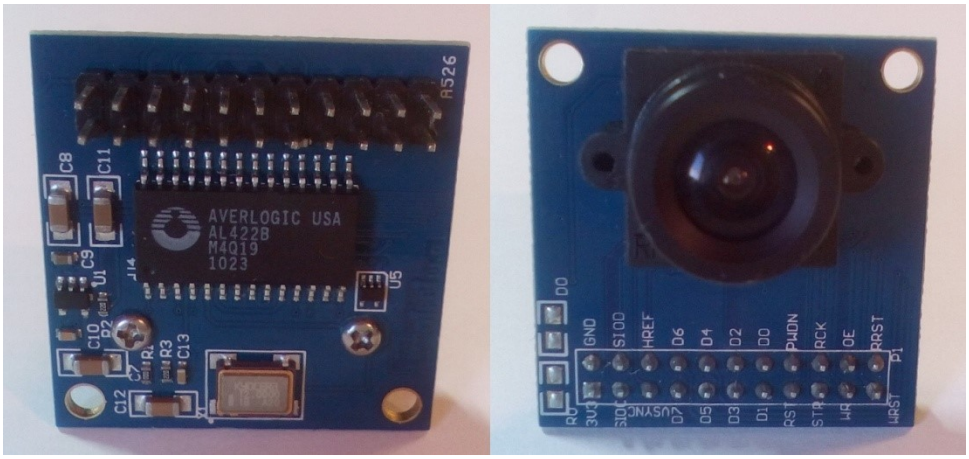
VCC/3V3	Analog power suppy	STROBE	LED/strobe control output
GND	Ground	FIFO RCK	Read clock
SIO_C/SDC	SCCB clock	FIFO WR	Write clock
SIO_D/SDA	SCCB data	FIFO OE	Output enable
VSYNC	Vertical synchronization	FIFO WRST	Write reset
HREF	Horizontal synchronization	FIFO RRST	Read reset
Reset	Reset	XCLK	System Clock
PWDN	Power down mode selection	PCLK	Pixel clock
D0-D1	Digital outputs		

Tab. 3. Obecné parametry OV7670 [21]

Rozlišení	640 × 480
Napájení	Výpočetní logika 1.8 V, Analog 2.45 V – 3 V, I/O 1.7 V – 3.0 V
Výstupní formáty	YUV/YCbCr 4:2:2, RGB565/555/444, GRB 4:2:2, RAW RGB data
Zorný úhel	25°
Šířka čipu	1/6“
Maximum FPS	30 (pro VGA)
Citlivost	1.3 V/(Lux*sec)
Temné napětí	12 mV/s při 60 °C
Obrazová plocha	2.36 mm × 1.76 mm



Obr. 16. OV no FIFO [autor]



Obr. 17. OVFIFO [autor]

3.2 Vývojové platformy

Vývojových platforem je nepřeberné množství. Zde uvedené vývojové platformy jsou často používané a známé. Pro snazší výběr té správné platformy zde budou uvedeny hlavní vlastnosti vývojové platformy, jako je výkon, velikost pamětí, počet a typy vstupních a výstupních pinů, rozměry, komunikační rozhraní a další specifické vlastnosti.

3.2.1 Arduino UNO

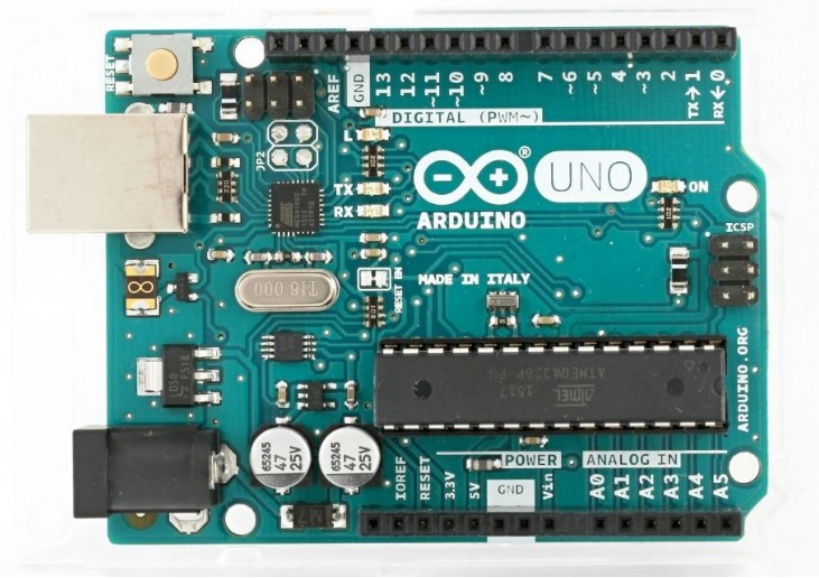
Arduino UNO patří do velké rodiny vývojových platforem od firmy Arduino, která se specializuje právě na vývojové platformy a všechno potřebného okolo nich. Tyto platformy jsou označovány jako open source, tedy lze veškerou dokumentaci a zapojení DPS najít a je volně dostupná. Arduino UNO je oproti ostatním deskám od Arduina malá a má omezené možnosti, co se týče množství pinů a hlavně velikosti paměti, která mnohdy na rozsáhlejší aplikace nestačí. Ovšem na jednodušší aplikace nebo učení se programování pro začátečníky je i díky obrovské podpoře a rozsáhlým výukovým tutoriálům skvělý prostředník. Velkou výhodou jsou také patice pro jednoduché připojení vodičů, na DPS jsou osazené patice typu samice a jakýkoliv vodič pak můžeme do této patice zasunout. Toto všechno dělá z této desky přívětivého „user friendly“ pomocníka.

Připojení k počítači je velmi jednoduché, stačí propojit vývojovou desku konektorem USB k PC, stáhnout program Arduino IDE přímo z oficiálních stránek Arduina, v položce Nástroje ve vrchní liště tohoto programu vybrat příslušný port, kde je vývojová deska připojena, typ desky a můžeme začít programovat. Napájení může probíhat přes USB kabel nebo přes externí napájecí konektor. Na desce také můžeme najít ICSP header pro programování hlavního mikroprocesoru a externí programování USB převodníku. Tyto patice využívají některé shieldy. [24] Více informací o připojení k PC a programování bude v praktické části v kapitolách 4 a 8.

Tab. 4. Parametry – Arduino UNO [24]

Typ mikroprocesoru	ATmega328P
Taktovací frekvence	16 MHz
Paměť Flash	32 kB (0,5 rezervováno pro bootovací sekci)
EEPROM	1 kB
Datová sběrnice	8bitová
Provozní napětí	5 V

Vstupní napětí	7 ÷ 12 V
Maximální proud	20 mA na I/O, 50 mA na 3,3 V
Digitální I/O piny	14 (6 z nich umožňuje PWM)
Analogové piny	6
ICSP rozhraní	2 (pro programování mikroprocesoru a USB)
Rozměry	68,6 × 53,4 mm
Hmotnost	25 g



Obr. 18. Arduino UNO rev3 [24]

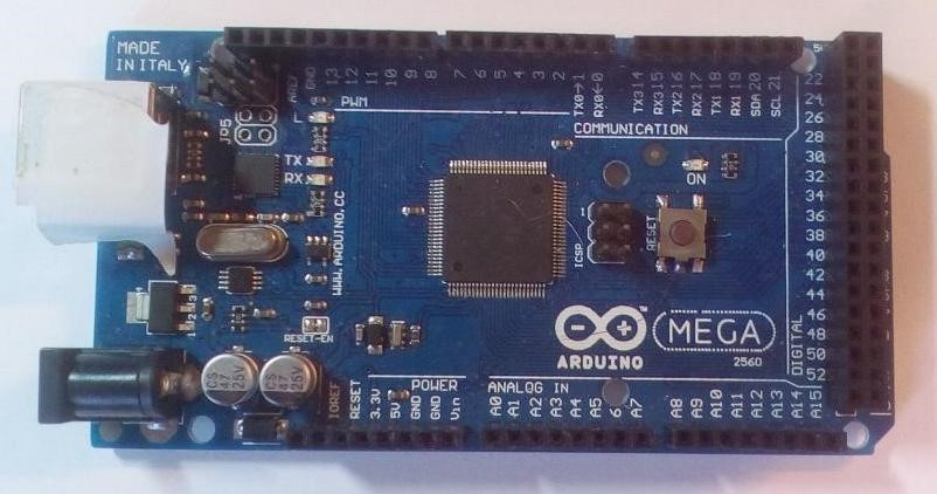
3.2.2 Arduino Mega 2560

Arduino Mega je větší a výkonnější verze Arduina UNO. Obsahuje daleko větší paměťový prostor a více pinů k použití. To je umožněno díky lepšímu procesoru ATmega2560, který je schopen všechny peripetie obsluhovat. Programování, připojení k PC a napájení je stejné jako u arduina UNO.

Tab. 5. Parametry – Arduino Mega 2560 [25]

Typ mikroprocesoru	ATmega2560
Taktovací frekvence	16 MHz
Paměť Flash	256 kB (8 Kb rezervováno pro bootovací sekci)
SRAM	8 kB
EEPROM	4 kB
Datová sběrnice	16bitová

Provozní napětí	5 V
Vstupní napětí	7 ÷ 12 V
Maximální proud	20 mA na I/O, 50 mA na 3,3V
Digitální I/O piny	54 (15 z nich umožňuje PWM a 4 UART)
Analogové piny	16
ICSP rozhraní	2 (pro programování mikroprocesoru a USB)
Rozměry	101,5 × 53,3 mm
Hmotnost	37 g

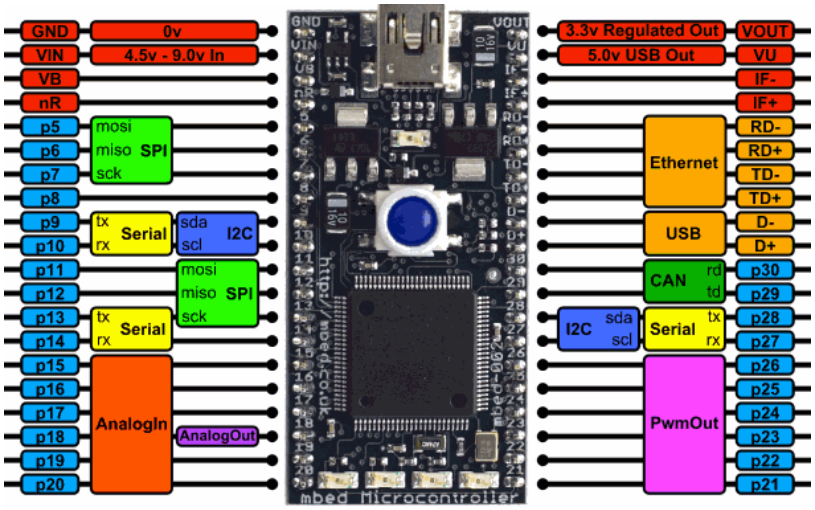


Obr. 19. Arduino Mega 2560 [autor]

3.2.3 Mbed NXP LPC1768

Mbed NXP LPC1768 (dále jen jako Mbed) je malá vývojová platforma, avšak výkonem předčí i mnohé větší platformy. Oproti Arduino platformám nejsou na desce vyvedeny vstupy z vrchní strany ve formě samice, ale tato deska má konektory typu samec ze spodní části. To zaručuje minimalizování rozměrů, avšak pro propojení jsou pak nutné jiné typy vodičů, nebo využití kontaktního nepájivého pole.

Napájení může probíhat přes mikro USB, nebo přes dvojici pinů vyvedených z desky. Piny p5 až p30 se dají použít pro digitální vstupy a výstupy. Tyto piny mají také specifické vlastnosti a podle dokumentace je možné je využít pro následující rozhraní: ethernet, USB, CAN, SPI, I^2C, A/D a D/A převodníky, PWM, USART. Jednotlivé rozložení pinů je na obrázku 20. [26]

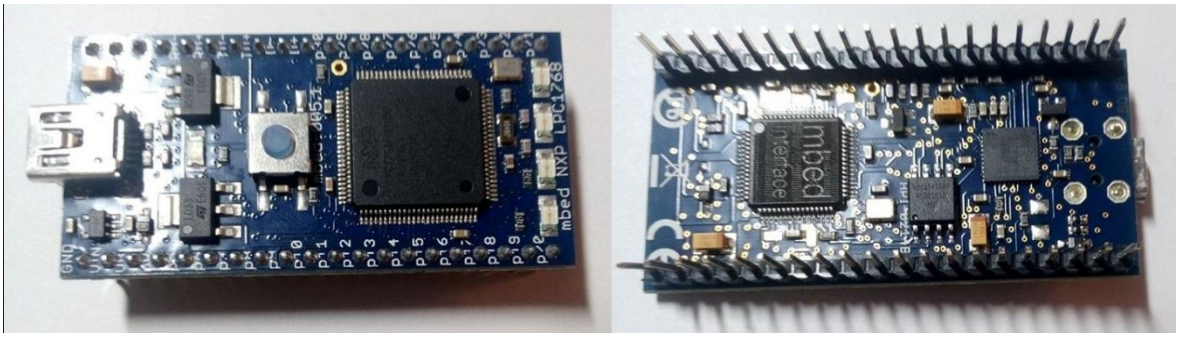


Obr. 20. Rozložení pinů na desce Mbed NXP PLC1768 [26]

Pro propojení s PC využijeme USB port. Dále otevřeme nový adresář s připojeným externím zařízením a přes ikonu MBED.HTM se spojíme se serverem a zaregistrujeme se. Následně můžeme využít online podporu, dokumenty a veškeré návody. Jako programovací jazyk můžeme použít C, C++ nebo využít jednodušší online compiler. [26] Podrobný návod bude uveden v praktické části diplomové práce v kapitole 5.

Tab. 6. Parametry – Mbed NXP LPC1768 [26]

Typ mikroprocesoru	ARM® Cortex™-M3 Core
Taktovací frekvence	96 MHz
Paměť Flash	512 kB (8 Kb rezervováno pro bootovací sekci)
SRAM	32 kB
Datová sběrnice	32-bit
Provozní napětí	3,3 V
Vstupní napětí	4,5 ÷ 9 V
Maximální proud	20 mA na I/O, 50 mA na 3,3V
Digitální I/O piny	26 – z toho 6 analog in, 1 analog out, 6 PWM, dále je možné propojit přes tyto piny rozhraní: ethernet, USB, CAN(2×), SPI(2×), I^2C(2×), UART (3×)
Rozměry	54 × 26 mm
Hmotnost	11 g



Obr. 21. Mbed NXP LPC 1768 [autor]

3.2.4 Raspberry Pi 3 Model B

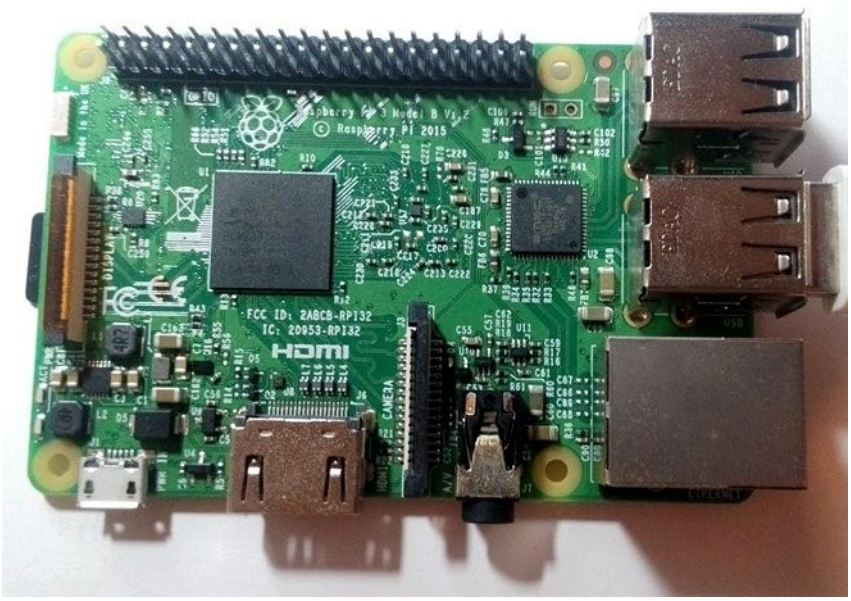
Další z velmi oblíbených vývojových platforem pro výuku programování a využití v nejrůznějších aplikacích. Výkonově se dokáže rovnat slabšímu počítači a díky výstupu HDMI pro monitor, 4 USB porty a zabudovanou síťovou kartou je možné jej zapojit jako pracovní stanici. Je možné na něm provozovat různé distribuce Linuxu a speciálně navržené operační systémy jako je RISC OS a Microsoft Windows 10 Iot Core. [27]

Raspberry Pi lze použít jako přehrávač hudby, videí nebo pro přístup k internetu. Disponuje 17 GPIO piny, kde některé z nich mají speciální funkce. Přistupuje se k nim v systému jako k samostatným speciálním souborům. Lze také využít UART, I2C, SPI. Na desce je 15pinový MIPI konektor kamerového rozhraní (CSI). [27]

Největší výhodou (a také případnou nevýhodou) je práce přímo s operačním systémem. Pro programování a ladění to může být dobré v tom, že po prvotním nastavení už není třeba externě přistupovat a vše se dá programovat přímo na místě. Nevýhodou je ovšem to, že díky složitějšímu softwaru je větší šance, že dojde k nestabilitě na softwarové úrovni. Případně aktualizace mohou zapříčinit nefunkčnost dříve funkčních aplikací. [27]

Tab. 7. Parametry – Raspberry PI 3 Model B [27]

Typ mikroprocesoru	ARM® Cortex™-A53
Taktovací frekvence	1,2 GHz
Video procesor	Broadcom VideoCore IV (250 MHz)
SDRAM	1GB (společná pro video i procesor)
Datová sběrnice	64-bit
Vstupní napětí	5 V
Provozní napětí - GPIO	3,3 V (maximální proud 50 mA)
GPIO	17
Rozměry	85,6 × 56,5 mm
Hmotnost	45 g



Obr. 22. Raspberry PI 3 [autor]

II. PRAKTICKÁ ČÁST

4 VYUŽITÍ ARDUINO MEGA 2560 S KAMEROU OV 7670

Při obou konfiguracích je využita kamera OV 7670 od firmy Omni Vision, aby se daly porovnat rozdíly ve výkonu jednotlivých zařízení a způsobu programování. Budou zde také uvedeny základní příkazy a popsaná struktura programu. Samotný program poté bude přílohou k této diplomové práci.

4.1 Použité komponenty a elektrické zapojení

Vnitřní logika vývojové desky Arduina Mega 2560 a výstupní napětí na digitálních pinech je 5 V. Oproti tomu, napájení a logika kamery OV 7670 je 3,3 V. Z toho důvodu je nutné přidat do obvodu napěťové konvertory, které nám zajistí správný přenos signálu na jedné straně a zabrání zničení kamery přepětím na straně druhé. Dalšími komponenty, které budeme pro zapojení potřebovat, je kontaktní nepájivé pole a množství propojovacích vodičů různých barev a typů. Ideálními vodiči jsou přímo vodiče s koncovkami a to buď typu samec, nebo samice. Rozpis používaných komponentů je v následující tabulce.

Tab. 8. Soupis potřebných komponentů pro Arduino

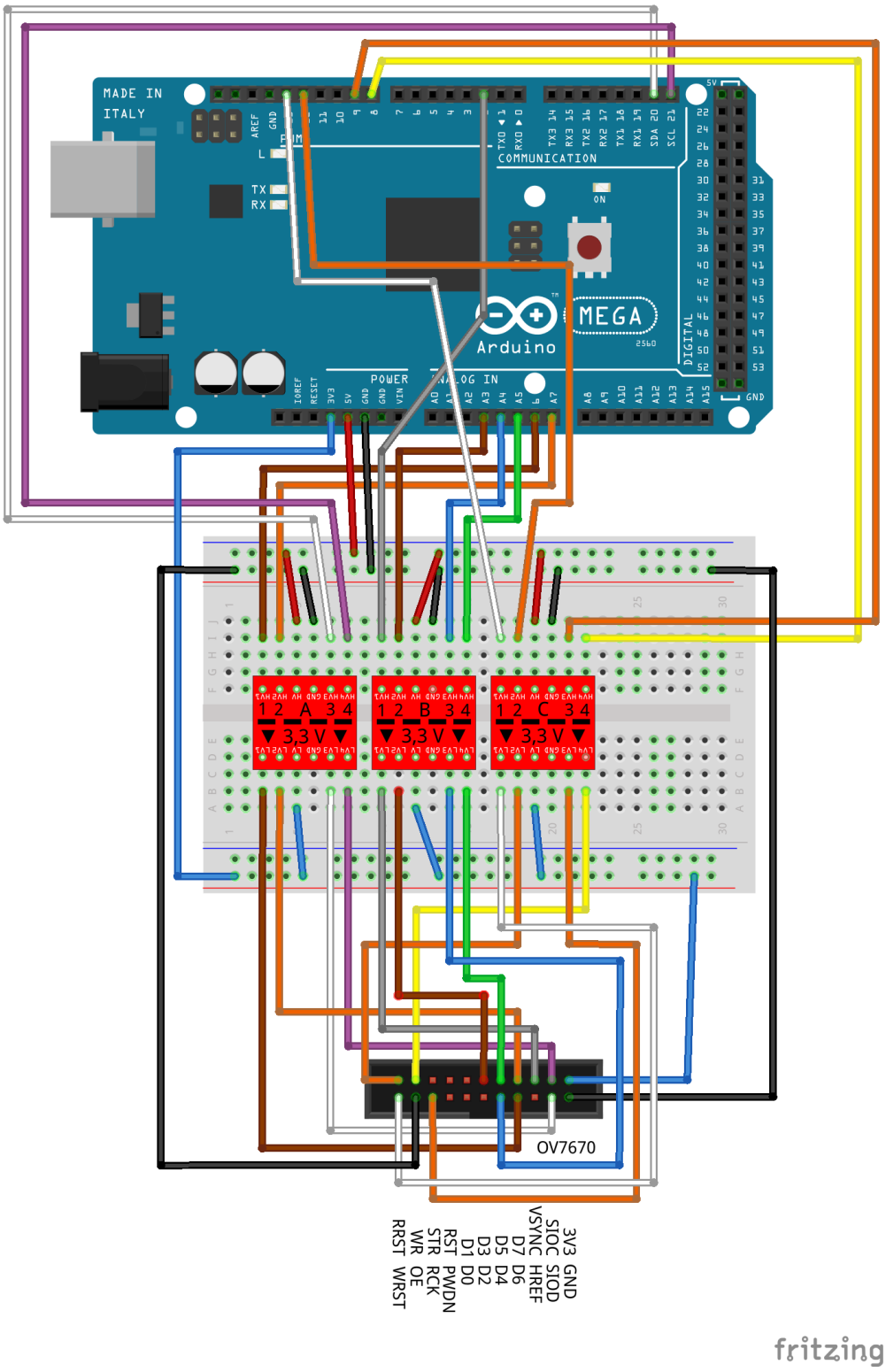
Typ zařízení (komponent)	**Počet**
Arduino Mega 2560 rev.3	1x
OV 7670 V3 with FIFO	1x
Logický napěťový konvertor	3x
Kontaktní nepájivé pole	1x
Vodič typu samec – samec	24x
Vodič typu samec – samice	15x

Na obr. 23. je elektrické zapojení kamery OV 7670 s Arduinem Mega 2560. Používané barvy vodičů jsou orientační pro snazší zapojení a přehlednost. Reálně je možné použít jakékoliv vodiče. Všechny vodiče zapojené přímo na kameru jsou typu samec – samice, jelikož piny na kameře jsou typu samec. Všechny ostatní vodiče jsou typu samec – samec. Zapojení je vytvořené s programem Fritzing, nejedná se tedy o elektrické schéma, ale o pomocné schéma pro snazší orientaci a zapojování.

V tabulce 9 jsou rozepsané všechny použité vodiče, barva a odkud kam se dané vodiče připojují. Pro lepší přehlednost je na obrázku vyobrazeno kontaktní nepájivé pole. Všechny součástky musí mít propojené země (GND), napájení 5 V a 3,3 V. Pro snazší orientaci je napětí i zem vyvedeno na napěťové sběrnice na kontaktním nepájivém poli. Přičemž vyšší potenciál a to 5 V je blíže k Arduinu a nižší potenciál, tedy 3,3 V, je vyveden na opačnou stranu desky, a to ke kameře. U napěťových konvertorů je ve sloupci L/H naznačeno, na kterou stranu součástky se mají dané vodiče zapojit. Pokud je zde L, tak na stranu s nízkým potenciálem, tedy 3,3 V, pro H na stranu s 5 V. Zem (GND) lze zapojit na obě strany, ale pro správnou funkčnost stačí zapojit pouze jednu stranu, země na obou stranách jsou spojené. Pro H – L platí pravidlo, že všechny vodiče jdoucí směrem od Arduina jsou zapojovány na vyšší potenciál, tedy stranu H a všechny vodiče jdoucí směrem ke kamerce jdou zapojeny na stanu L.

Tab. 9. Soupis vodičů pro zapojení s Arduinem Mega 2560

Arduino	**Level Convertor**			**OV 7670**	**Barva vodiče**
Pin	**Konvertor**	**L / H**	**Č. cesty**	**Pin**	
GND	A,B,C	H	GND	GND, OE	Černá
5V	A,B,C	H	HV	-	Červená
3,3V	A,B,C	L	LV	3V3	Modrá
A6	A	H - L	1	D6	Hnědá
A7	A	H - L	2	D7	Oranžová
SCL - D20	A	H - L	3	SIOD	Bílá
SDA - D21	A	H - L	4	SIOC	Fialová
D2	B	H - L	1	VSYNC	Šedá
A3	B	H - L	2	D3	Hnědá
A4	B	H - L	3	D4	Modrá
A5	B	H - L	4	D5	Zelená
D13	C	H - L	1	WRST	Bílá
D12	C	H - L	2	RRST	Oranžová
D9	C	H - L	3	RCK	Oranžová
D8	C	H - L	4	WR	Žlutá



Obr. 23. Zapojení kamery OV 7670 v3 – FIFO s Arduino Mega 2560 [autor]

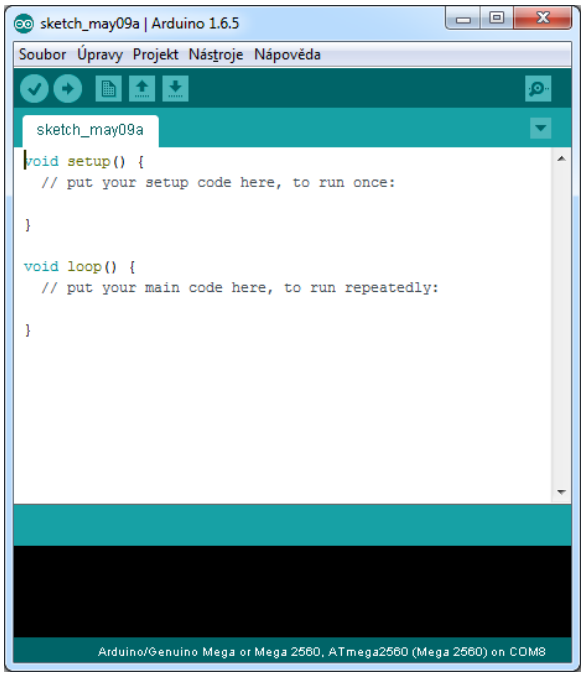
4.2 Potřebný SW a propojení s PC

Využívanými programy pro komunikaci kamery s PC jsou: IDE Arduino a program Processing. Každý z těchto nástrojů bude v následujících kapitolách více rozebrán. Všechny programy běží na Windows 7, 64 bit.

4.2.1 IDE Arduino

Jedná se o integrované vývojové prostředí (IDE znamená Integrated Development Environment). Jedná se tedy o vývojové prostředí, které má v sobě integrovanou spoustu nástrojů a ulehčuje samotné programování. Program se dá stáhnout na oficiálních stránkách *www.arduino.cc/en/Main/Software*, kde lze najít poslední verzi programového prostředí. Distribuce softwaru je open source a tedy je možné se k němu dostat zdarma a snadno. [30]

Na obr. 24. můžeme vidět úvodní obrazovku spuštěného programu Arduino IDE. Ve vrchní části najdeme lištu s různým nastavením programu a práce s kódem samotným. V záložce Soubor je hlavní práce s právě otevřeným projektem, jako je ukládání, zavření, vlastnosti daného projektu a jeho zobrazení, nebo otevření nového projektu. Úpravy, kde se upravuje převážně vzhledová stránka samotného kódu. V záložce Projekt jsou nástroje pro kompilaci programu a jeho nahrávání na desku. Nástroje slouží primárně pro nastavení vlastností připojené desky, portů, na které je připojené a otevření sériového monitoru, kde se vypisuje sériová komunikace. Poslední záložkou je Nápověda.



Obr. 24. IDE Arduino – úvodní obrazovka [autor]

Pod stavovou lištou jsou aktivní tlačítka:

Obr. 25. IDE Arduino – aktivní tlačítka [autor]

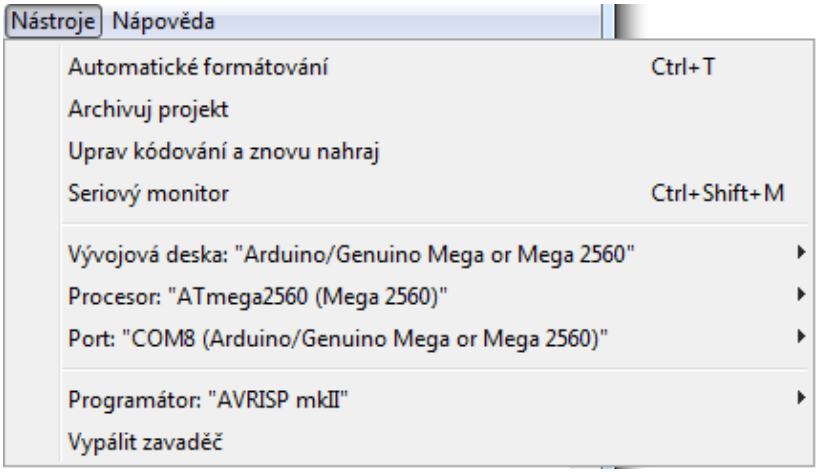
Prvním zleva je tlačítko, které zkompiluje aktuální projekt a zjistí, zda se v něm nevyskytují chyby. Druhé tlačítko je šipka vpravo, to zkompiluje projekt a odešle ho do připojeného zařízení. Třetím je tlačítko pro otevření nového projektu. Ten se otevře jako další záložka. Čtvrtým tlačítkem je šipka nahoru, která otevře konkrétní soubor. Pátým tlačítkem je šipka dolů, toto tlačítko slouží pro uložení aktuálního projektu. Posledním tlačítkem, které je úplně vpravo, je otevření sériového monitoru, který složí pro zachytávání sériové komunikace mezi připojeným zařízením a PC.

Největší část okna zabírá bílé místo pro psaní samotného kódu. Zde se dělí kód na tři hlavní oblasti. Prvním je deklarace proměnných, knihoven a dalších věcí, které se nastavují prvotně, aby program věděl, s čím pracuje a co musí připojit. Dalším je tělo setup(), kde se nastavuje rychlost komunikace a nastavení vstupů a výstupů. Třetím je tělo loop(), které slouží jako hlavní tělo programu. Více bude popsáno a vysvětleno v kapitole 8 Praktické příklady a výukový tutoriál.

Oblast mezi bílou plochou pro psaní kódu a černou (zde se vypisují informace ohledně paměti, chyb a dalších věcí), slouží pro stavové informace.

Pro správné připojení zařízení k PC vývojovou desku Arduino Mega 2560 připojíme pomocí USB do libovolného portu v PC. Otevřeme si stažené prostředí IDE Arduino a v záložce Nástroje nastavíme jako vývojovou desku „Arduino/Genuino Mega or Mega 2560“. Procesor nastavíme na „ATmega2560 (Mega 2560)“ a port vybereme ten, kde máme připojené své zařízení. Je možné, že toto už bude přednastavené, ale je lepší si to vždy před kompilací programu zkontrolovat, hlavně port. Občas se může stát, že bude vybrán jiný port, než ke kterému je zařízení reálně připojené. Správné nastavení můžeme vidět na obr. 26.

Spustíme soubor „ov_fifo_test_mega.ino“ v prostředí Arduino IDE a tento spuštěný soubor nahrajeme na vývojovou desku Arduina pomocí tlačítka Nahrát.



Obr. 26. IDE Arduino – nastavení připojeného zařízení [autor]

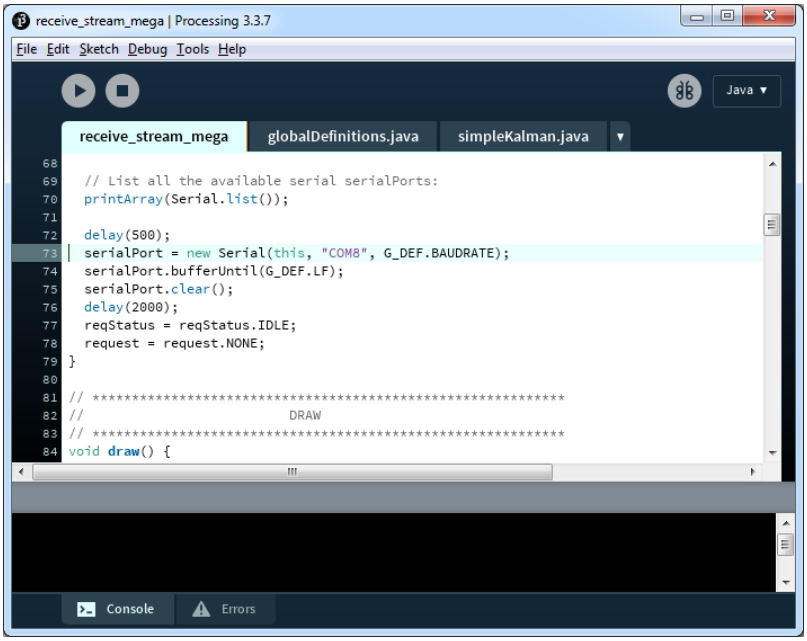
4.2.2 Processing 3.3.7

Jedná se o PDE (Processing Development Environment), tedy obdobu IDE programového prostředí. Program je vzhledově podobný jako IDE Arduino a mají mnoho společného. Nepoužívá se na programování hardware, ale programování v něm má zjednodušit a přiblížit programování i neprofesionálům. Programovacím jazykem je Processing, který vychází z jazyku Java. Hlavní oblastí, v které dané prostředí operuje, je kreslení 2D a 3D struktur a práce s obrazem. Pro náš projekt budeme používat právě práci s obrazem, kde budeme propojovat výstup z kamery přes Arduino do PC, kde obraz budeme pomocí tohoto programu zobrazovat. [31]

Po spuštění programu „receive_stream_mega.pde“ se zobrazí úvodní obrazovka programu Processing s třemi otevřenými okny. Hlavní část programu je „receive_stream_mega“ a další dva listy „globalDefinition.java“ sloužící pro definování proměnných použitých v programu a „simpleKalman.java“, kde je nadeklarován Kálmánův filtr. To můžeme vidět na obr. 27. Ve vrchní části programu jsou jednotlivá nastavení programu samotného a otevřeného projektu. V nabídce File můžeme otevřít a zavřít soubor, spustit příkladové programy a také exportovat hotovou aplikaci.

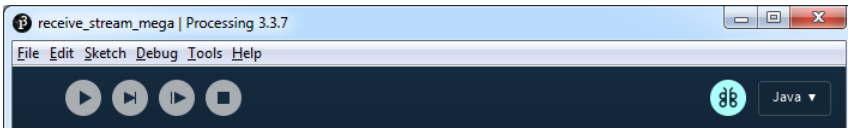
Jakmile budeme mít spuštěný soubor a otevřené okno stejně jako na obr. 27., přepneme se v okně „receive_stream_mega“ na řádek 73:
serialPort = new Serial(this, "COM8", G_DEF.BAUDRATE);

V tomto řádku změníme číslo portu na port, který se nám zobrazuje v programu Arduino IDE úplně dole a je to číslo portu, kam máme připojené Arduino Mega s připojenou kamerkou. Zde je COM nastavený na COM8, avšak u každého počítače se tento port mění.



Obr. 27. Processing – úvodní obrazovka [autor]

Pokud je připojené Arduino s kamerkou k počítači a je v něm nahraný program, můžeme spustit program v Processingu stiskem na tlačítko Run – kulaté šedé tlačítko se šipkou ve vrchní části programu, případně program zastavit pomocí Stop – kulaté šedé tlačítko se čtvercem. Po stisknutí tlačítka Debug – kulaté šedé tlačítko vypadající jako motýl na pravé straně programu se spustí režim Debug, kde můžeme program step-by-step krokovat a sledovat hodnoty jednotlivých proměnných.



Obr. 28. Processing – aktivní tlačítka [autor]

Program pro vyčítání obrazu přes sériovou linku „receive_stream_mega.pde“ napsaný v programu Processing byl použit jako volně přístupný kód použitý v projektu Arduvision od David Sanz Kirbis. [28]

4.3 Struktura programu

Pro značně rozsáhlý kód nebude zde uveden celý kód, ale bude tu popis, co která část programu dělá a k čemu která knihovna slouží. Knihovny mají koncovku .h nebo .cpp. Hlavní program má pak vždy koncovku .ino.

ov_fifo_test_mega.ino

Hlavní část programu se skládá z větší části z podprogramů, které se vykonávají postupně. Na obr. 29. můžeme vidět schéma, jak je program postupně vykonáván a jaké má části.

Nejdříve se načtou externí knihovny, které jsou pro tento program využívány, a to „IO_config,h“, „fifo.h“ a „sensor.h“. Následně dojde k nastavení přenosové rychlosti komunikace po sériové lince (baudrate) a definici obrazových formátů QQVGA a QQQVGA.

Při zahájení komunikace přes sériovou linku se kdykoliv může spustit podprogram *void serialEvent()*, který zjistí, jaká data jsou přes linku posílána. Pokud je zde žádost o obrazová data, spustí se podprogram *void parseSerialBuffer()*, který nastaví parametry z instrukcí na sériové lince pro obrazová data a nastaví hodnotu očekávaného přenosu *bRequestPending* na true. Následně se podprogram vrací na místo, kde došlo k přerušení a vyvolání podprogramu *void serialEvent()*.

Následně se spustí podprogram *void setup()*, ve kterém se nastaví porty pro OV7670 a jejich přenosové režimy, inicializuje se senzor a vypíše se stav inicializace na sériovou linku. Vyvolá se podprogram void vsyncIntFunc(), který zastaví zápis na FIFO buffer a otestuje, zda jsou splněné podmínky *new_frame* = true a *bRequestPending* = true. V případě že ne, resetuje se časovač, obnoví se zápis na FIFO a nastaví se hodnota new_frame na true. Program se vrací zpátky na *void setup()* a v případě, že bude požadavek o obraz (*bRequestPending* = true), zavolá se podprogram *void processRequest()*.

Zjistí se zde, co se má dále s daty udělat. Při požadavku o změnu komunikačního formátu se tento formát nastaví a v tomto formátu se odešlou obrazová data po sériové lince. Případně zde může být požadavek na výpočet FPS, tedy by se vyvolala funkce *void calcFPS(unsigned int &currentFPS)*, která odešle vypočítané FPS na sériovou linku. Následně se pro všechny možnosti popsané v tomto odstavci nastavuje *new_frame* = false a *bRequestPending* = false a program se vrací zpátky na podprogram *void setup()*.

IO_config.h

Tato knihovna slouží pro definování pinů připojených ke kameře OV7670 s FIFO bufferem AL422. Nastavuje, zda jsou tyto vstupy vstupní nebo výstupní a také jejich režim.

delay.h

Knihovna vychází z originální Arduino knihovny „delay.h“. Nastavuje přesně stanovenou čekací dobu v milisekundách, která je optimalizována pro běh uživatelského programu.

fifo.cpp, fifo.h

Knihovna „fifo.cpp“ se odkazuje se na knihovnu „fifo.h“ a slouží pro zpřehlednění výstupů z programu a volání jen potřebných funkcí. Společně ovládají komunikaci FIFO bufferu a jeho nastavování a vyčítání obrazu.

ov7670_regs.h

Nastavování registrů kamery pro obrazové formáty a jejich transformaci mezi sebou. Pro kameru OV7670

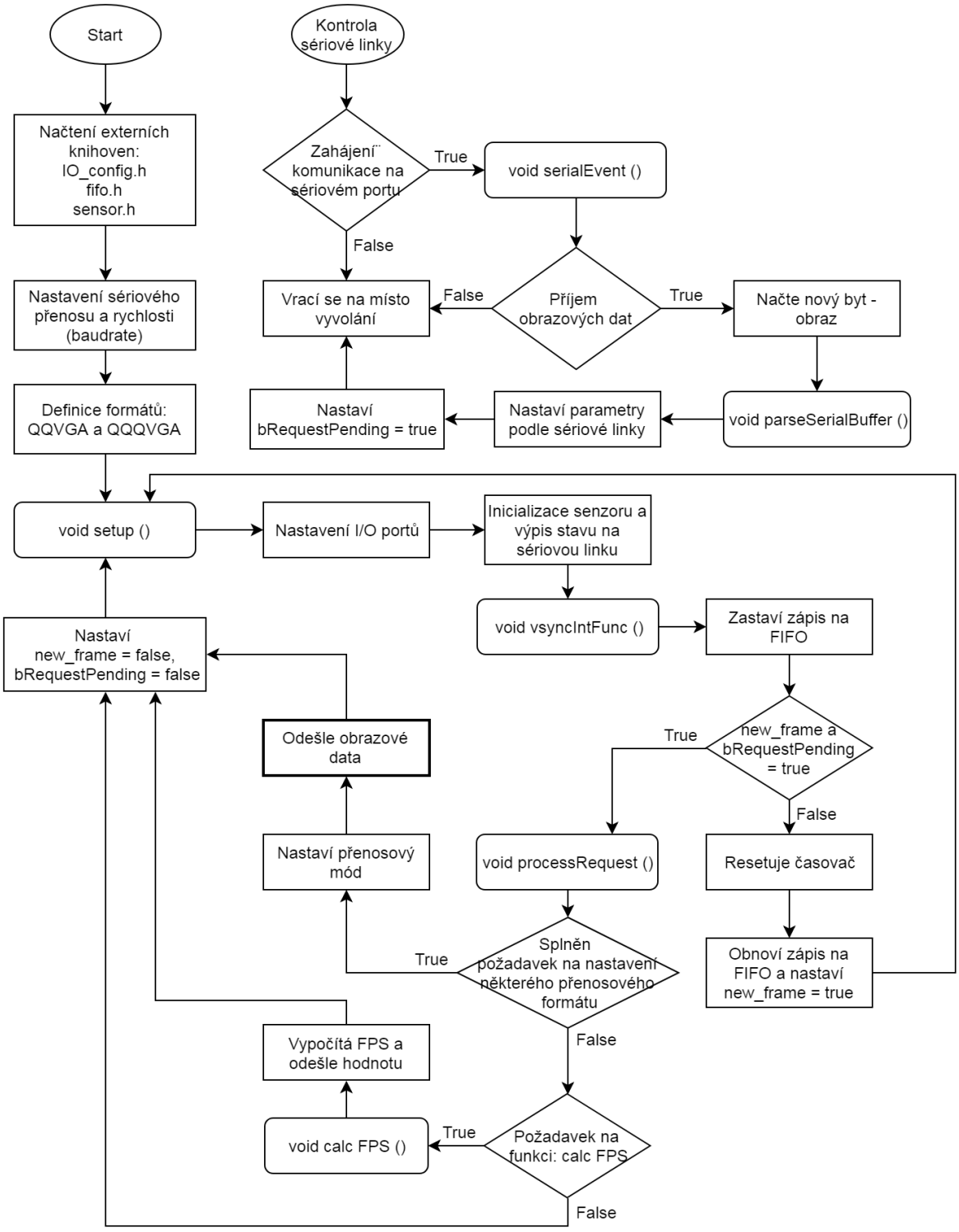
ov772x_regs.h

Nastavování registrů kamery pro obrazové formáty a jejich transformaci mezi sebou. Pro kamery z řady OV772x

sensor.cpp, sensor.h

Knihovna „sensor.cpp“ se odkazuje na knihovny: „sensor.h“, „OV7670_regs.h“ a „ov772x_regs.h“. Společně zajišťují nastavení formátu kamery, rozlišení a dalších obrazových nastavení, včetně výpisu informací z registrů.

Program pro komunikaci Arduina Mega 2560 s kamerou OV7670 napsaný pro program Arduino IDE byl použit jako volně přístupný kód použitý v projektu Arduvision od David Sanz Kirbis. [28]



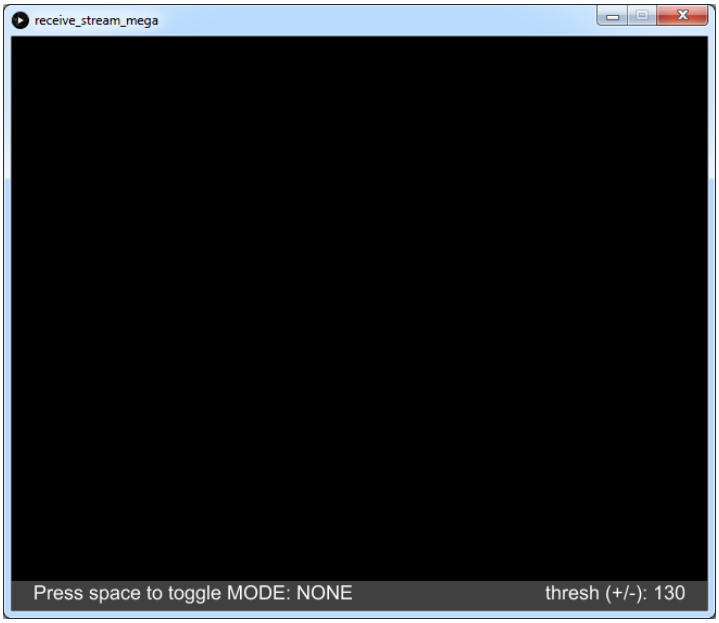
Obr. 29. Hlavní program pro Arduino – logika a hlavní části [autor]

4.4 Běh programu a výsledný obraz

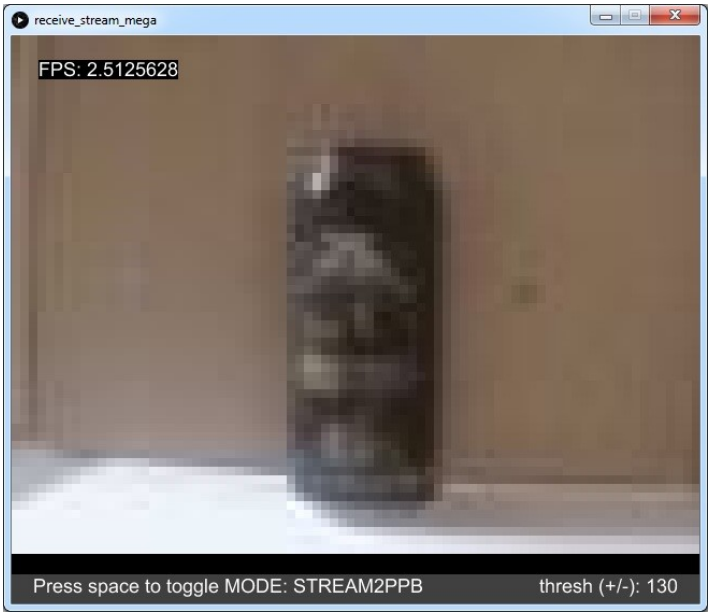
Po nahrání programového vybavení na Arduino Mega 2560, připojení kamery OV7670 a spuštění programu „receive_stream_mega.pde“ dojde ke spuštění programového rozhraní, kde se automaticky načte video nahrávané z Arduina (obr. 30.). Přepínání módů probíhá tak, že se při otevřeném okně programu po zmáčknutí mezerníku přepínají režimy obrazu. Ty jsou: NONE, TRACKDARK, TRANKBRIG, STREAM8PPB (4, 2, 1, 0 PPB). NONE je základní, kde nezobrazuje žádný obraz. TRACKDARK a TRACKBRIG jsou podobné módy, kdy načtou do paměti úroveň osvětlení pro další zpracování. Následují módy STREAM8PPB až 0 PPB. PPB je označení bitové hloubky obrazu. Funkce je potom následující:

$$počet\ barev = \left(\frac{8}{číslo\ před\ PPB}\right)^2 \tag{11}$$

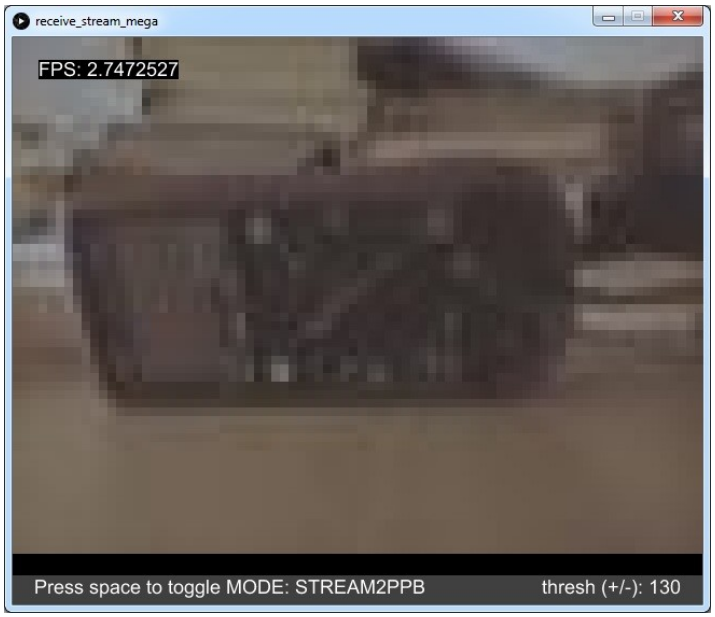
platí přitom, že hodnota pro 0PPB = 16^2, tedy neuvažujeme dělení 0.



Obr. 30. Úvodní obrazovka programu pro nahrávání [autor]



Obr. 31. Stream obrazu v módu 2PPB – láhev [autor]



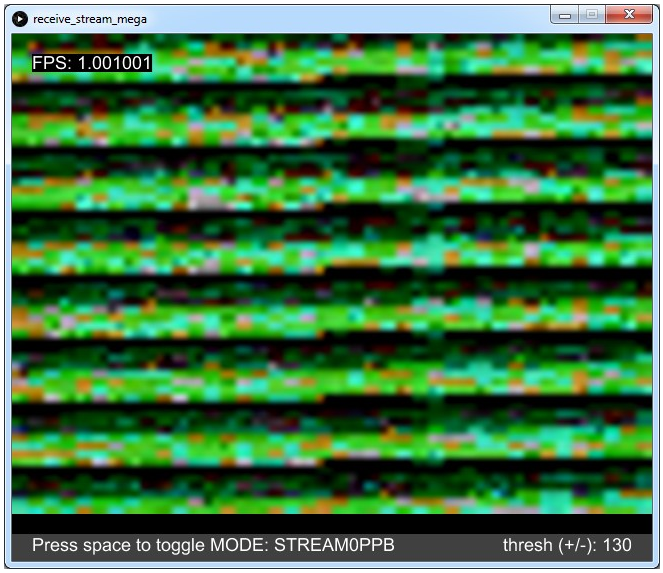
Obr. 32. Stream obrazu v módu 2PPB – multimetr [autor]

Na obr. 31. a obr. 32. vidíme, jaká je výstupní kvalita obrazu při rozlišení 80 × 60 pixelů při počtu 16 barev. Dokážeme rozlišit obrys, ale nedokážeme už určit přesné detaily. Při módu 2PPB se rychlost snímkování průměrně dostane na hodnotu 2,7 snímku za sekundu. Takto pomalý přenos pro tak špatnou kvalitu je dán převážně hardwarovým omezením výkonu Arduina, který nezvládá data zpracovávat rychleji.

4.4.1 Troubleshoting

Kamera OV7670 je citlivá na rušení, které je všude okolo nás, v běžném provozu při správném zapojení to ovšem nemá až takový vliv na přenos. Zásadním problémem je přepětí na vstupně výstupních pinech. Při nepozornosti a špatném přepojení některého z datových vodičů na + 5 V může nastat situace, kterou vidíte na obr. 33 a obr. 34. V tomto případě došlo k elektronickému poškození některých obvodů pro zpracování obrazu. Kamera se stala díky chvilce nepozornosti nefunkční.

Obr. 33. Stream obrazu v módu 2PPB – šum [autor]



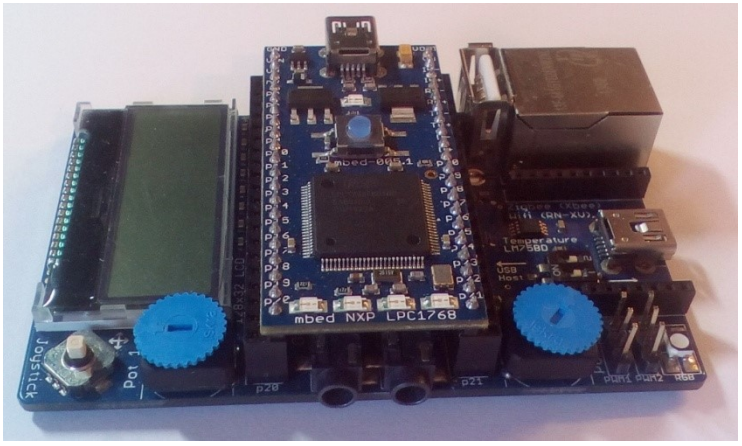
Obr. 34. Stream obrazu v módu 0PPB – šum [autor]

5 VYUŽITÍ MBED NXP LPC1768 S KAMERKOU OV7670

Pro tuto konfiguraci bude použita platforma od firmy Mbed a kamerka OV7670 stejná, jako v předchozí kapitole. Rozepsané bude elektrické zapojení, potřebný SW a programové vybavení pro ovládání kamery.

5.1 Použité komponenty a elektrické zapojení

Jelikož vývojová platforma Mbed NXP LPC1768 má vnitřní logiku a napájení na 3,3 V, nepotřebujeme do obvodu přidávat externí součásti pro změnu napájecích hladin (logic convertor), jelikož logika OV7670 je taky na 3,3 V. To nám velmi ušetří zapojení a tím pádem je zapojení daleko jednodušší a je méně náchylné na poškození z důvodu špatného zapojení vodičů. Pro snazší zapojení a také kvůli zkoušení programování této platformy je samostatná platforma připojena k shieldu Mbed Aplication Board (obr. 35.).



Obr. 35. Zasunutý Mbed do Aplication Board [autor]

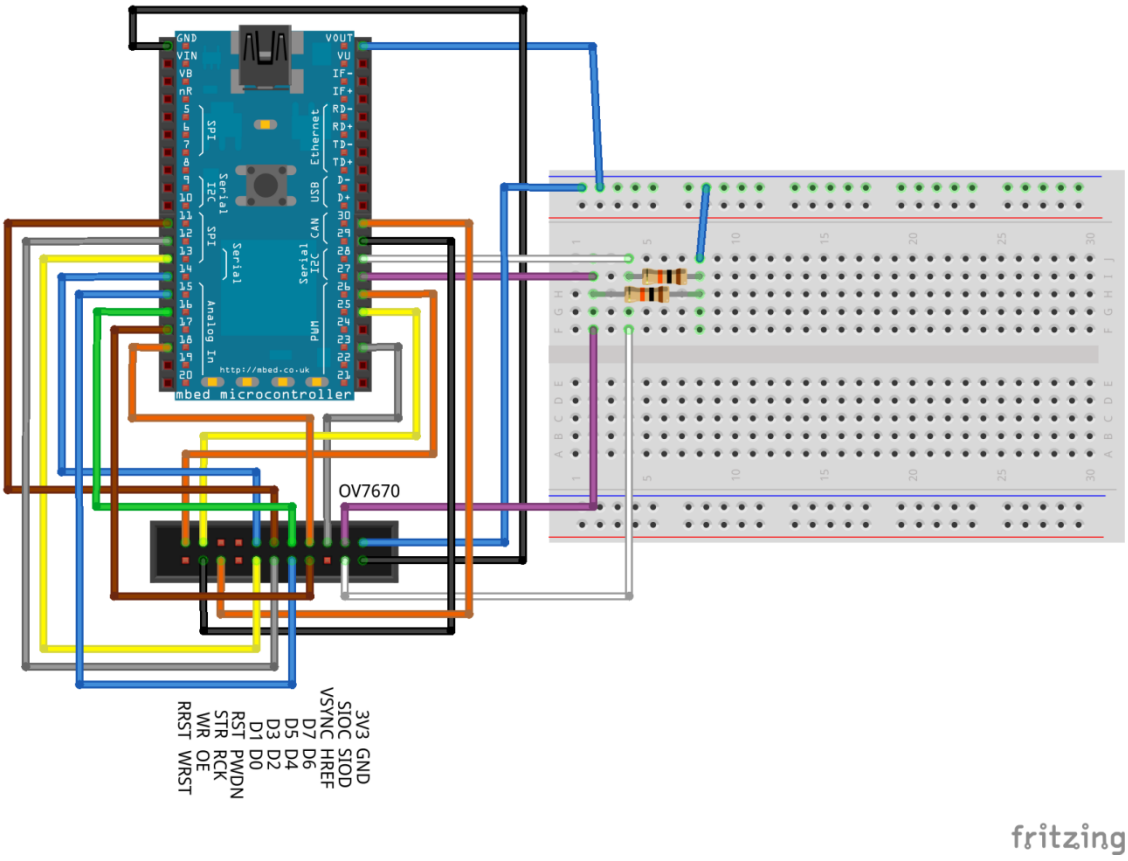
Tab. 10. Soupis potřebných komponentů pro Mbed

Typ zařízení (komponent)	**Počet**
Mbed NXP LPC1768	1x
OV 7670 V3 with FIFO	1x
Kontaktní nepájivé pole	1x
Vodič typu samec – samec	24x
Vodič typu samec – samice	15x
Rezistor 10 kΩ	2x

Tab. 11. Soupis vodičů pro zapojení s Mbed NXP LPC1768

Mbed Pin	**OV 7670 Pin**	**Barva vodiče**
GND	GND	Černá
3,3V	3V3	Modrá
28	SIOD	Bílá
27	SIOC	Fialová
23	VSYNC	Šedá
30	RCK	Oranžová
29	OE	Černá
25	WR	Žlutá
26	RRST	Oranžová
13	D0	Žlutá
14	D1	Modrá
12	D2	Šedá
11	D3	Hnědá
15	D4	Modrá
16	D5	Zelená
17	D6	Hnědá
18	D7	Oranžová

Pro snazší orientaci v zapojení je na obrázku 36 schéma zapojení kamerky s Mbedem. Piny SIOD a SIOC jsou připojeny k desce přes pull-up rezistory, které udržují potřebné napětí na těchto pinech, aby v důsledku okolního rušení nedošlo k neočekávaným stavům na příslušných pinech a komunikace probíhala správně. Zapojení těchto pull-up rezistorů je vyobrazeno na následujícím obrázku.



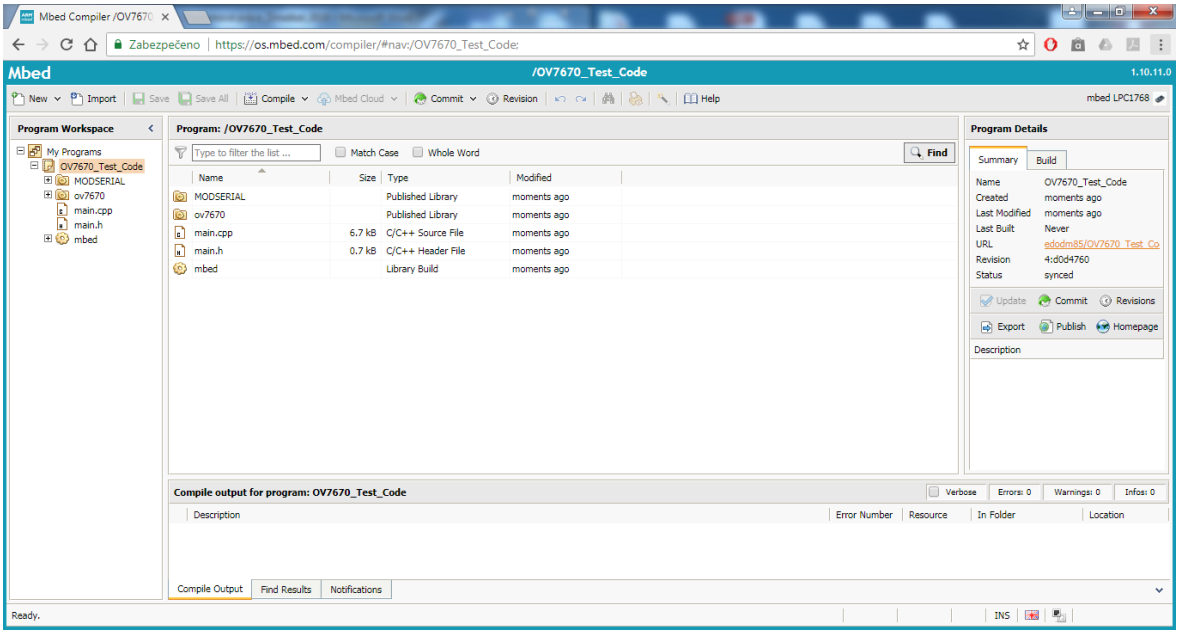
Obr. 36. Zapojení kamery OV 7670 v3 – FIFO s Mbed NXP LPC1768 [autor]

5.2 Propojení s PC a potřebný SW

Pro připojení s PC potřebujeme mít připojený Mbed pomocí USB konektoru. Při prvotním připojení se automaticky nainstalují ovladače pro komunikaci a zobrazí se nám Mbed připojený jako datové úložiště. Následně spustíme soubor „MBED.HTM“, který otevře internetový prohlížeč a požádá nás o přihlášení.

Pro tuto možnost byly vytvořeny přihlašovací údaje: **login: MbedProfile**, **password: Mbedprofile**.

Po přihlášení pomocí těchto údajů se zobrazí webové stránky na serveru: *http://os.mbed.com*, kde ve vrchní části stránky klikneme na oranžovou ikonu Compiler. Následně se nám zobrazí online verze kompilátoru, kde budou uloženy všechny projekty, které budou vytvářeny pro vývojovou desku Mbed NXP LPC1768. Výhodou online kompilátoru je ta, že ke kódu se můžeme dostat kdykoliv a kdekoliv na jakémkoliv zařízení. Také zde budou nahrané všechny programy pro Mbed využívané v této diplomové práci.



Obr. 37. Online kompilátor Mbed [autor]

Na obr. 37. můžeme vidět online prostředí kompilátoru pro vývojové desky Mbed. V horní části programu je lišta s akčními ikonami. Popíšeme si zde ty nejdůležitější. Z levé strany to jsou New - vytvoření nového projektu. Import – nahrání programu z počítače či jiného online uložiště. Save pro uložení aktuálního otevřeného okna. Save all – pro uložení všech otevřených oken. Compile – zkompiluje program a v případě bezchybného programu uloží do PC program s koncovkou .bat.

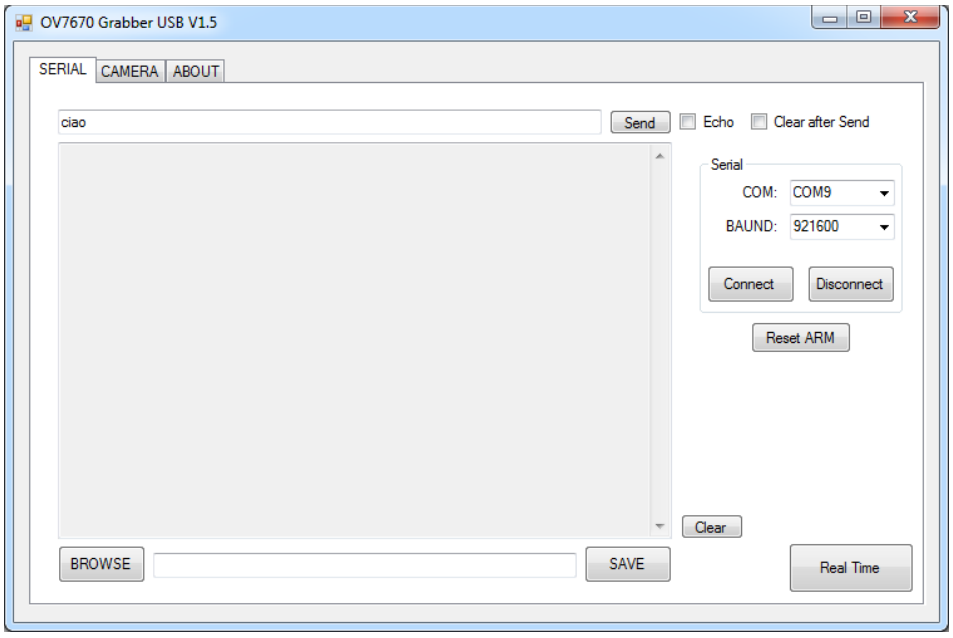
V online kompilátoru si otevřeme projekt OV7670_Test_Code. Po kliknutí na ikonu Compile se projekt zkompiluje a uloží na PC jako „OV7670_Test_Code_LPC1768.bin“. Tento program se poté nahraje na připojený Mbed a po jeho restartování se program spustí. Je důležité, aby byl na desce nahraný pouze jeden program, aby nedocházelo ke kolizím ve spuštění programu.

Následně si spustíme program „OV7670 Grabber USB V1.5.exe“, který nám zajistí vyčítání obrazu ze sériové linky pomocí USB. Po jeho otevření se zobrazí úvodní obrazovka, kterou můžeme vidět na obrázku 38. Zde si nastavíme na pravé straně port COM takový, na kterém máme připojený Mbed.

V této chvíli se může stát, že nenajdeme port, ke kterému je Mbed připojený, jelikož se primárně připojil jako datové uložiště a při nenainstalovaném driveru pro sériovou komunikaci nemusí fungovat. Nejdříve se podíváme do správce zařízení, zda se zde nenachází připojený Mbed na jiném portu, nebo zda zde není „neznámé zařízení“. Kdyby se jednalo o

neznámé zařízení, stačí kliknout pravým tlačítkem myši a doinstalovat ovladače. Kdyby se zde nenacházel, je třeba si ze stránek: *https://os.mbed.com/handbook/Windows-serial-configuration* stáhnout aktuální ovladače. Do počítače se uloží soubor „mbedWinSerial_16466.exe“ a je nutné ovladač doinstalovat. Je možné, že budou aktualizované ovladače a soubor bude mít jinou verzi. Následně Mbed dostane přidělený COM port a můžeme se na něj pomocí programu OV7670 Grabber USB V1.5 připojit.

Program „OV7670 Grabber USB V1.5.exe“ byl použit jako volně přístupný program pro načítání obrazu ze sériové linky pro kameru OV 7670 a vývojovou platformu Mbed NXP PLC1768 nahraný od Edoardo De Marchi. [29]



Obr. 38. Úvodní obrazovka programu OV7670 Grabber USB V1.5 [autor]

5.3 Struktura programu

Program využívá tří knihoven a to MODSERIAL, ov7670 a knihovny mbed. Dále využívá vlastní program „main.cpp“ a knihovnu „main.h“. Oproti předchozí kapitole, kde bylo popsáno Arduino, jsou knihovny v mbedu s koncovkou .h, ale hlavní programy jsou s koncovkou .cpp.

main.cpp

Hlavní program odkazující se na knihovnu „main.h“. Struktura programu a jeho fungování je popsána na vývojovém diagramu na obrázku 39. Cílem programu je zajištění komunikace mezi kamerou a počítačem, nastavování přenosových protokolů a odesílání dat pomocí sériové linky do PC. Program reaguje na příkazy posílané pomocí sériové linky USB do Mbedu a podle příkazů vykonává program.

Na začátku programu se načte používaná knihovna „main.h“ a nastaví se počet pixelů, které obsahují jednotlivé obrazové formáty. Vyčítání obrazu probíhá vždy posíláním dat za sebou, a tedy jsou jednotlivé byty pixelů posílány v datových posloupnostech.

Následně se začne vykonávat podprogram *int main ()*, ve kterém se program zacyklí a čeká na komunikaci po sérově lince. V případě, že po sériové lince dojde jakýkoliv znak, spustí se automaticky podprogram *void rXCallback(MODSERIAL_IRQ_INFO *q)*, který nastaví parametr *new_send* na hodnotu true. Následně se při dalším cyklu začne vykonávat podprogram *int main ()* dál a dojde k načtení celého přenosového slova a uložení tohoto slova do paměti.

V dalším kroku se vyvolá podprogram *void parse_cmd()*, který ovládá komunikaci s PC a s programem pro vyčítání obrazu. Ten nastaví parametr *new_send* na hodnotu false a následně porovnává přijatá slova a podle nich vykonává další instrukce podprogramu. Pokud dojde požadavek na nastavení formátu obrazu „init_*“, kde * značí požadovaný formát, se nastaví přenosový formát a následně se program vrací zpátky do hlavní části programu, kterou je *int main ()*. Při žádosti o výpis časových informací z posledního odeslaného obrazu „time“ se vypíšou přes sériovou linku časové informace. Při žádosti „reset“ se vyvolá program *void mbed_reset()*, který nám resetuje celý mbed a program se začne vykonávat od začátku, ve vývojovém diagramu označeno jako start.

Poslední možná žádost je žádost „snap“, která vyvolá podprogram *void CameraSnap()*. Tento podprogram zajišťuje načítání snímku z kamery OV 7670 do paměti Mbedu a ná-

sledně tato data přeposílá do PC. První se rozsvítí LED 4 na desce Mbedu a připraví se paměť pro načtení nového snímku (začne přenos až s novým snímkem, zbytek se zahodí). Následně vyčítání obrazu a odesílání do PC probíhá ve dvou krocích, prvně se načte první polovina snímku a následně druhá polovina snímku. Na fotce to nejde poznat, ale při videu by to zlepšilo snímkování na úkor kvality v prokládání obrazu. Následně se ukončí odesílání dat a pošlou se informace ohledně obrazu a zhasne se LED 4. Tím je ukončen přenos a program se opět vrací do své výchozí pozice na podprogram *int main ()*, ve kterém čeká na další instrukce z PC.

main.h

Knihovna definující nastavení TX a RX sériového přenosu, časovač a připojené piny kamery OV7670. Tato knihovna se dále odkazuje na zbývající knihovny, jejichž příkazy jsou využívané v programu „main.cpp“.

MODSERIAL

Knihovna řešící komunikaci pomocí sériové linky, definici kanálů a módů pro přenos a jednotlivé přenosové protokoly.

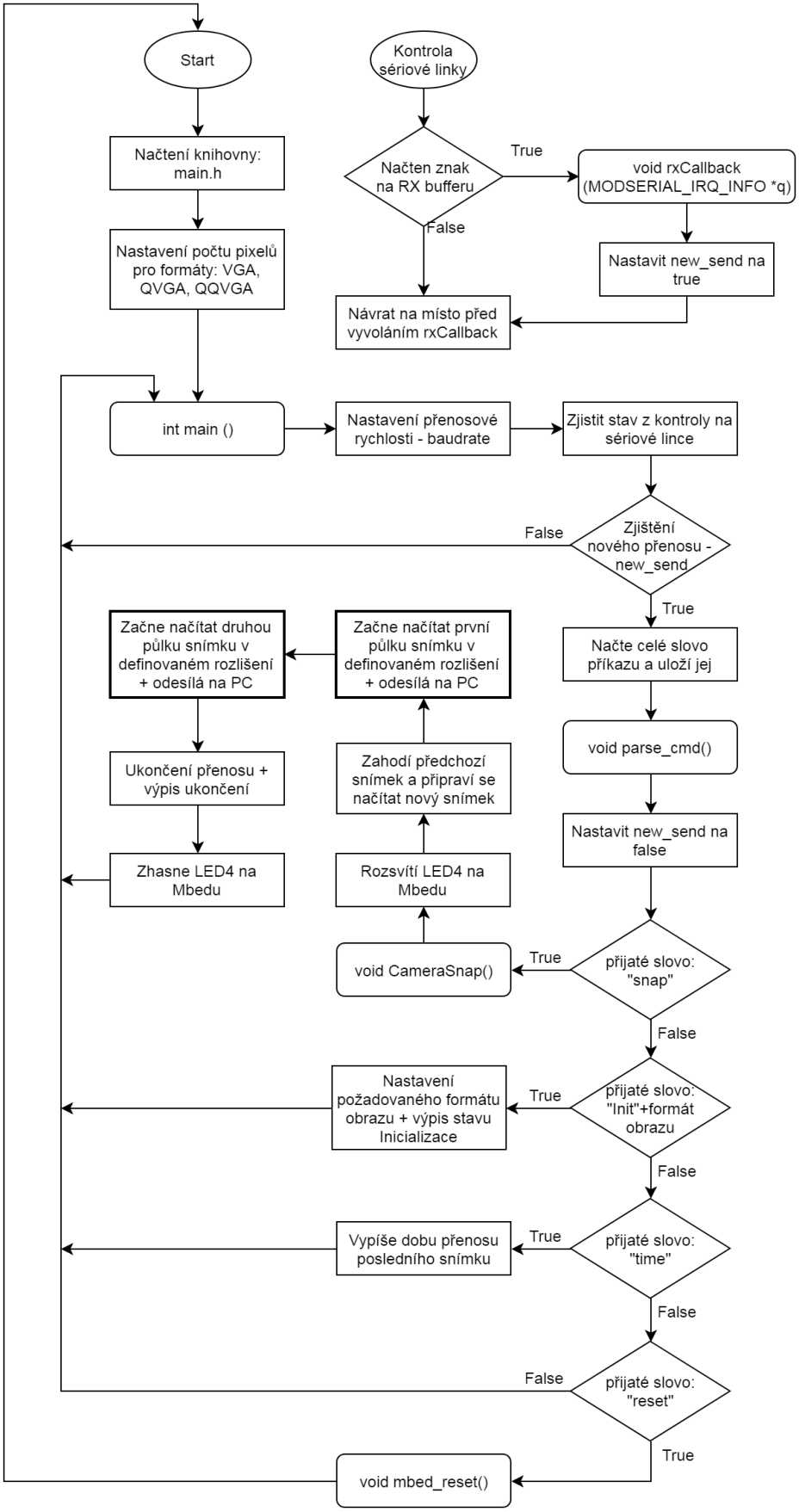
ov7670

Knihovna pro ovládání kamery ov7670 s FIFO bufferem. Řeší vyčítání obrazu z kamery, komunikaci dat v kameře, jednotlivé barevné kanály a nastavení této kamery.

mbed

Knihovna pro vývojovou desku Mbed s definovanými příkazy využívanými pro tuto vývojovou platformu.

Program pro komunikaci Mbed NXP LPC1768 s kamerou OV7670 napsaný pro tuto platformu sloužící k testování kamery byl použit jako volně přístupný kód od Edoardo De Marchi. [29]

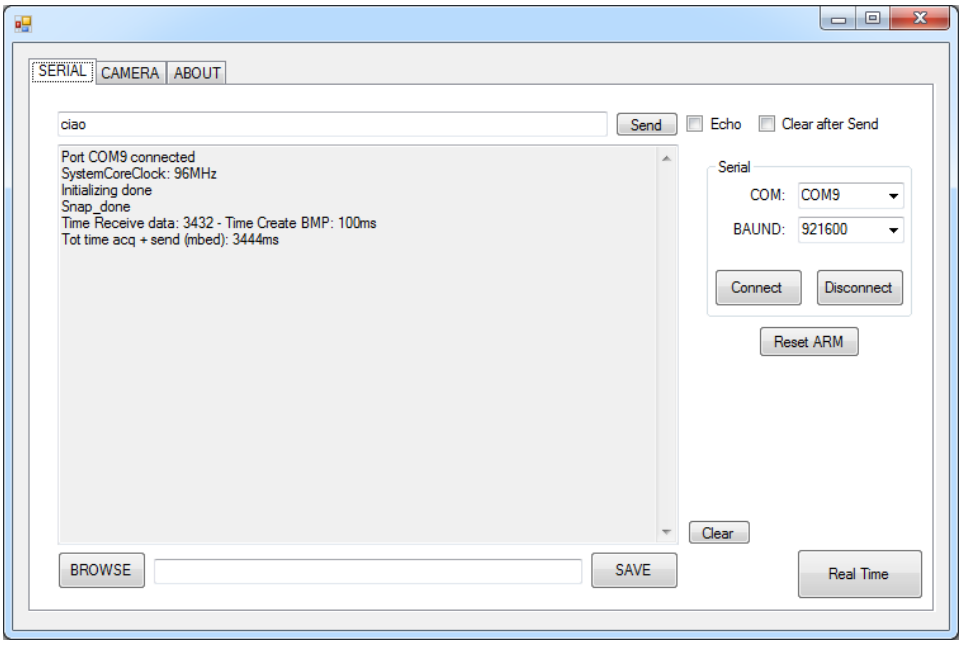


Obr. 39. Hlavní program pro Mbed – logika a hlavní části [autor]

5.4 Běh programu a výsledný obraz

V této kapitole budou ukázány výsledné obrazy z kamery OV7670 nasnímané pomocí vývojové platformy Mbed NXP LPC1768.

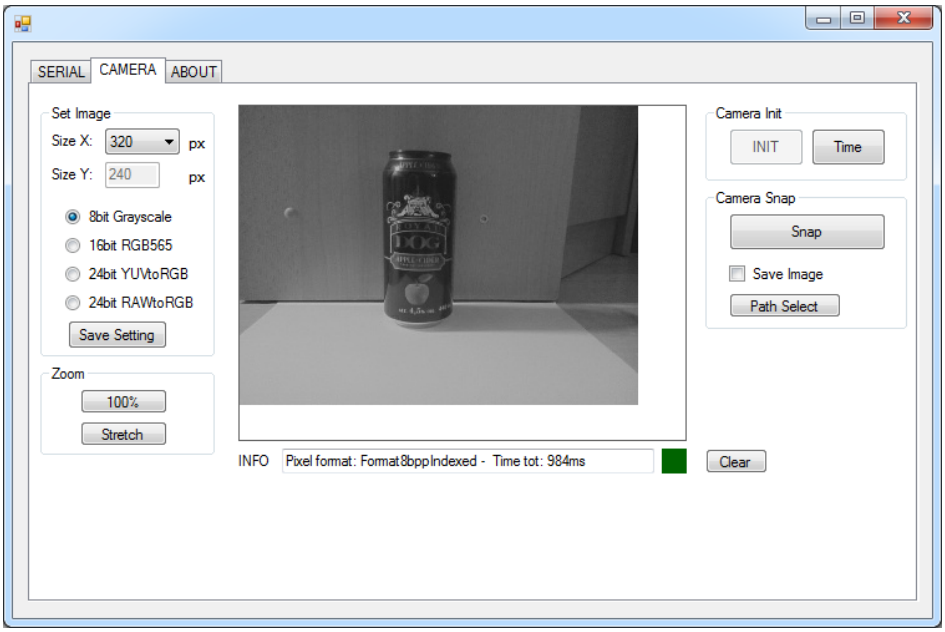
Ve spuštěné aplikaci OV7670 Grabber USB V1.5 se nejdříve nastavilo připojené zařízení Mbed na port COM9. Následně se zařízení připojilo pomocí tlačítka Connect, a otestovala se funkčnost připojení tlačítkem Reset ARM. Tím došlo k resetování Mbedu a k výpisu: SystemCoreClock: 96MHz. Dalším krokem byla změna okna z nabídky SERIAL do nabídky CAMERA a nastavily se následující parametry: Size X: 640 px, 24bit RAWtoRGB a následně se nastavení uložilo pomocí tlačítka Save Setting. Dále bylo stisknuto tlačítko INIT, které zajišťuje ověření připojení kamery k modulu, a tlačítkem Snap se nasnímal obraz a odeslal se na PC. Na obrázcích 40 a 43 vidíme programové okno s nastavením, přijatým obrazem a výpisem informací odeslaných na PC.



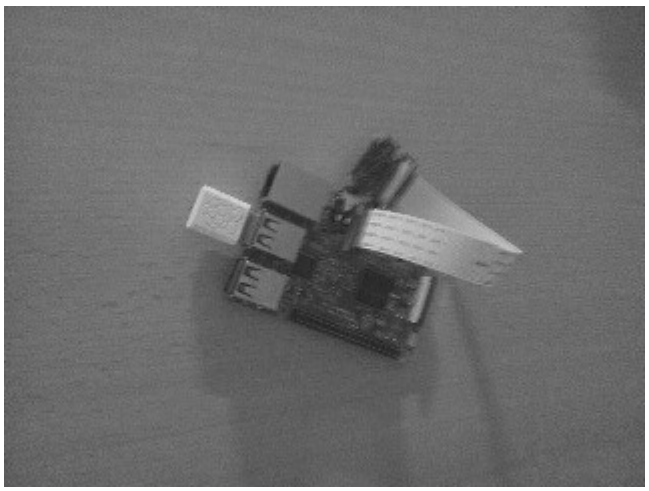
Obr. 40. Úvodní obrazovka se zachycenou komunikací [autor]

5.4.1 8 bit Grayscale

Tento formát obrazu je černobílý, s 256 odstíny šedé barvy. Obrázky snímáme v rozlišení 320 × 240 pixelů. Na obrázku 41 vidíme výstup z programu tak, jak ho zobrazuje program. Když se podíváme na celkový čas pořízení, zjistíme, že se obraz pořizoval a posílal celých 984 milisekund. Z obrázku 42 můžeme vidět kvalitu pořízené fotografie v relativně tmavém prostředí. Kvalita obrazu je tedy celkem dobrá a lze rozpoznat, co bylo vyfocené. Neposkytuje ovšem velké detaily.



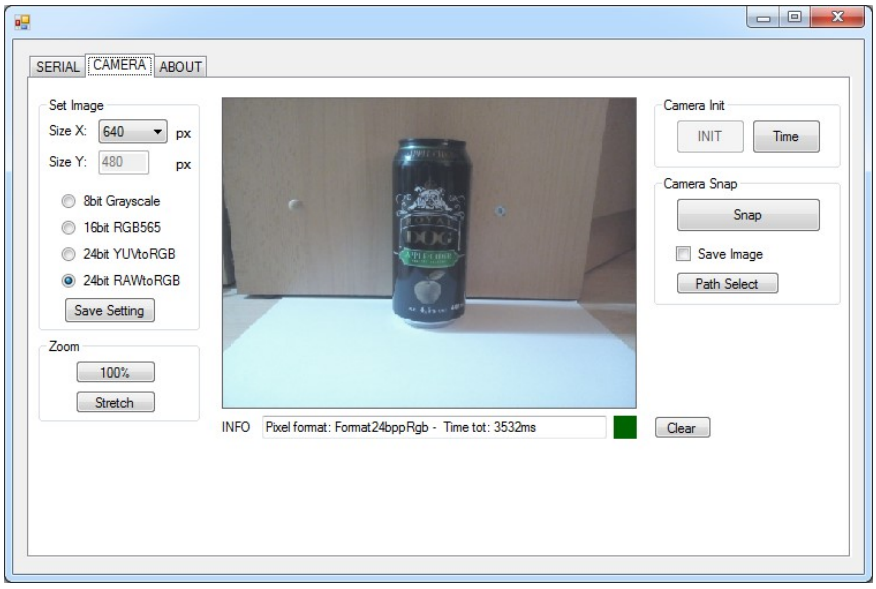
Obr. 41 Výstup z programu při nastavení: 8bit Grayscale 320 × 240 pixelů [autor]



Obr. 42. Raspberry Pie 3 – formát 8bit Grayscale [autor]

5.4.2 24bit RAW to RGB

Tento formát obrazu pracuje s barevnou hloubkou 24bitů pro všechny barvy, tedy 256 možností pro každou barevnou složku. Vyčítané rozlišení je 640 × 480 pixelů. Z celkové doby přenosu zjistíme, že se obraz přenášel 3532 milisekund, tedy něco málo přes 3 a půl sekundy. Výsledný obraz je tedy použitelný pro focení statických objektů, ale pro video je naprosto nevhodný. Co se týče detailů v tomto formátu, vše záleží na ostrosti kamery a tento formát už je použitelný pro focení i větších detailů.



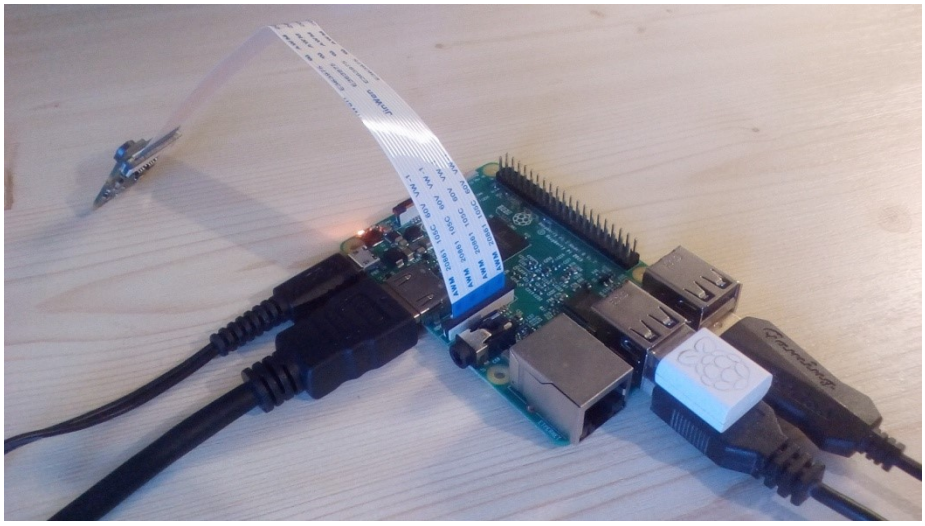
Obr. 43. Výstup z programu při nastavení: 24bit RAW to RGB [autor]

Obr. 44. Raspberry Pie 3 – formát 24bit RAW to RGB [autor]

6 RASPBERRY PI 3

Pro možnost porovnání kvality kamery a ukázání i další možnosti, jak lze připojit kamera k vývojovým platformám, zde bude uvedena vývojová deska Raspberry Pi 3 s originální kamerou pro tuto platformu Camera V2.1. Výhodou této vývojové desky je to, že pracuje jako samostatný PC, a není tedy třeba kameru a zařízení připojovat k dalšímu PC.

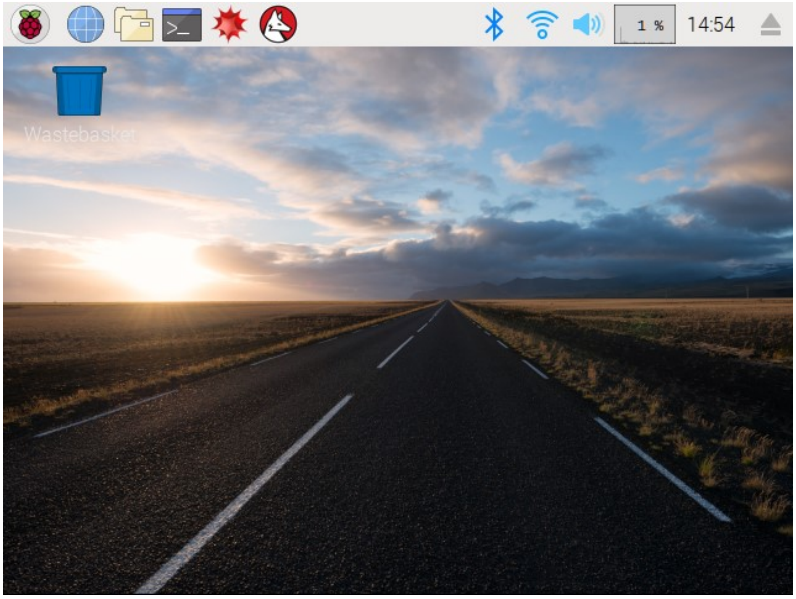
Pro oživení této platformy je potřeba externí klávesnice a myši, které můžeme připojit pomocí vestavěných portů USB. Dále potřebujeme monitor s podporou HDMI, nebo propojovací redukci pro datový kabel k monitoru. Dále potřebujeme toto zařízení napájet a to pomocí micro USB konektoru, tedy takového, jaký využívají mobilní telefony. Na obrázku 45 Můžeme vidět Raspberry PI s připojenou klávesnicí a myší, USB Wi-Fi, napájecím konektorem, HDMI kabelem pro připojení k monitoru a kamerou.



Obr. 45. Raspberry PI 3 se zapojenými zařízeními [autor]

6.1 Připojení k PC a oživení

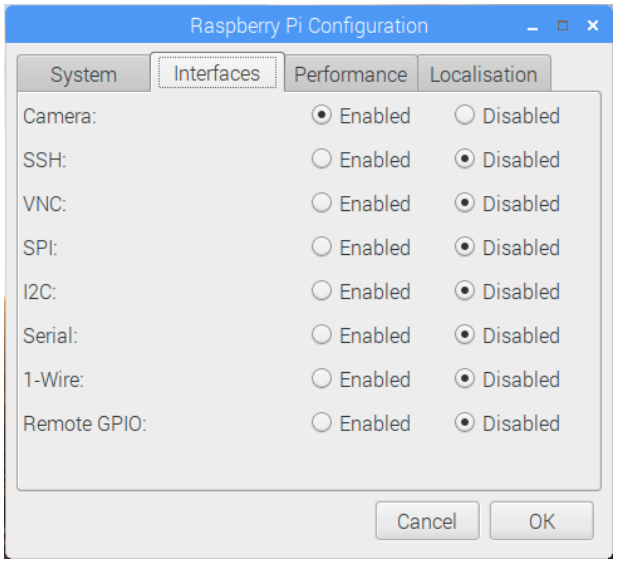
Ze zadní strany je zasunutá 16 GB SSD karta, na které je nahrán operační systém Raspbian, (distribuce Linuxu upravená pro toto zařízení). V případě nového zařízení nebo poškození stávajícího OS je možné Raspbian volně stáhnout z internetu z oficiálních stránek Raspberry. Při prvotním spuštění se načte obrazovka (obr. 46). Ve vrchní části obrazovky jsou ovládací ikony a menu pro nastavení zařízení. Jedná se o téměř totožnou lištu, kterou známe z Windows. Pro úpravu konfigurace systému si z menu (malina) vybereme možnost Raspberry PI Configuration a zde můžeme změnit vlastnosti systému.



Obr. 46. Úvodní obrazovka Raspberry PI [autor]

6.2 Zprovoznění kamery

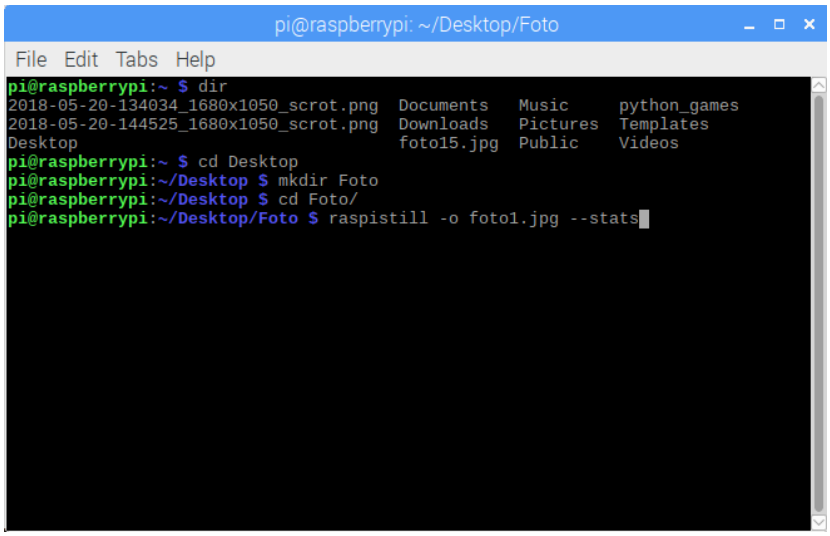
Ve druhém okně Raspberry PI Configuration si vybereme nabídku Interfaces a povolíme položku Camera (obrázek 47). Po stisknutí tlačítka OK a potvrzení se nám systém restartuje a připojí kameru.



Obr. 47. Nastavení pro aktivaci rozhraní [autor]

Následně se přepneme do příkazového řádku a zadáme zde 2 příkazy pro aktualizaci ovladačů. Jedná se o příkaz „*sudo apt-get update*“ a příkaz „*sudo apt-get upgrade*“, po jejich vykonání restartujeme Raspberry a můžeme začít snímat obraz.

Snímání probíhá pomocí příkazů v příkazovém řádku. Lze také využít vestavěného Pythonu a vytvořit skript, který by obraz například periodicky snímal a ukládal jej na určené místo. Pro názornou ukázku to ovšem není třeba. Spustíme si znovu příkazový řádek a přepneme se v něm na úroveň souborů, kde chceme, aby se soubory ukládaly. Ty se ukládají vždy na místo, odkud byl kód vyvolaný. Základní příkazy pro vytvoření složky a přepínání mezi složkami vidíme na obrázku 48. Také zde vidíme příkaz, který slouží pro nasnímání obrazu a to: „*raspistill –o foto.jpg --stats*“. Tento příkaz vyvolá vestavěnou funkci pro nasnímání a uložení obrazu. Prvních 5 sekund je obraz pouze zobrazen na obrazovce a následně je uložen do složky, odkud se tato funkce vyvolává. V našem případě pak nalezneme fotku foto1.jpg ve složce Desktop/Foto.



Obr. 48. Příkazový řádek s kódem pro snímání obrazu [autor]

Obrázky 49 a 50 na následující straně ukazují kvalitu vyfocených fotek pomocí této kamery. Fotografie jsou pořízené ve formátu 3280 × 2464 pixelů o 24bitové hloubce. Jedinou větší nevýhodou je to, že kamera je pevně zaostřena na 50 cm. Hrubým násilím za použití kleští jde ostřícím ústrojím pootočit a vylepšit výsledný obraz, ale hrozí poškození čočky a zničení celého modulu. Tím pádem nedokážeme dostat naprosto ostrý obraz na delší vzdálenosti a pro bližší objekty je obraz rozmazán ještě více. To jde částečně vidět na obrázku 49.

Obr. 49. Foto pomocí Raspberry PI – indoor [autor]

Obr. 50. Foto pomocí Raspberry PI – outdoor [autor]

7 POROVNÁNÍ VÝVOJOVÝCH PLATFOREM PRO PROPOJENÍ KAMERY A VYUŽITÍ PRO VÝUKU

Jednoznačně se nedá říct, která vývojová platforma je nejvhodnější. Každá platforma je vhodná pro určitou oblast a pokulhává v oblasti jiné. V této kapitole bude zhodnoceno využití platforem pro různé účely.

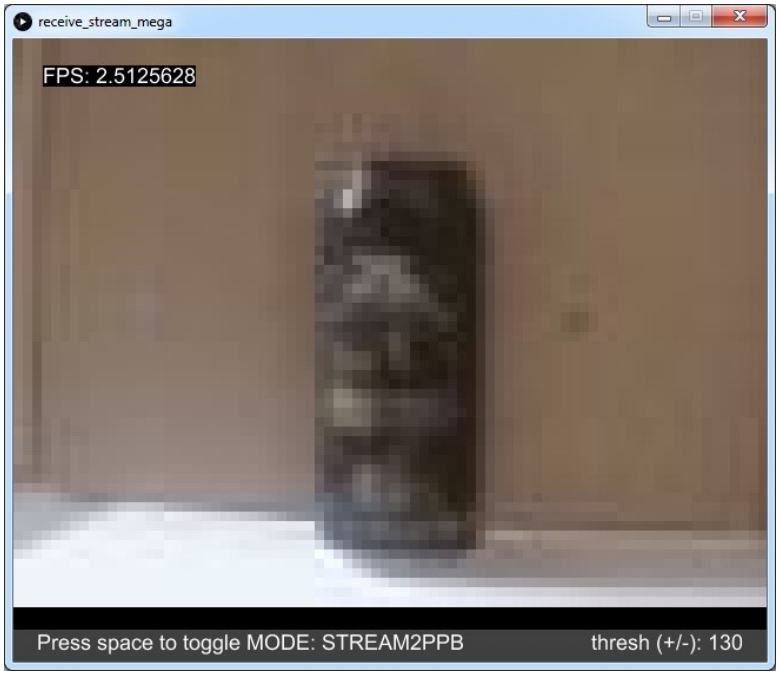
7.1 Propojení kamery, její zprovoznění a výsledný obraz.

Co se týče propojení kamery OV7670, jednoznačně nejlepší možností je vývojová platforma Mbed NXP LPC 1678. Propojení je relativně jednoduché a je malá šance možnosti špatného zapojení kamery. Platforma Arduino Mega 2560 kvůli vyššímu napětí desky musí využívat napěťové konvertory, které snižují napětí z 5 V na 3,3 V, avšak zapojení také není nějak zásadně složité.

Co se týče oživení samotné kamery, obě platformy na tom byly podobně. Při zapojení Arduina se program dal spustit a po změně pár příkazů, například změna náběžné hrany funkce pro vertikální synchronizaci, byl program víceméně funkční. Občas ještě došlo k zamrznutí programu Processing při načítání obrazu, a bylo nutné restartovat Arduino. Oživení kamery při použití Mbedu naráželo na problém na straně samotného PC pro komunikaci po sériové lince. Nejprve se musely aktualizovat všechny ovladače pro sériovou komunikaci a aktualizovat program (položka update v online kompilátoru). Následně se program dal spustit a oživit přenos dat z kamery.

Na následujících obrázcích můžeme vidět kvalitu obrazu z obou použitých platforem. Limitujícím kritériem je v našem případě rozlišení 80 × 60 pixelů. To je v aplikaci s Arduinem velmi malé a kvalita obrazu je nedostatečná. Dalším je pouze 4bitová barevná hloubka obrazu (obr. 51.). Oproti tomu je výstup pomocí platformy Mbed kvalitnější a používaný formát je zde 640 × 480 pixelů, s 8bitovou hloubkou pro každou z barev RGB (obr 52.).

Pro porovnání kvality kamery používané v této diplomové práci byla přidána třetí platforma, a to Raspberry PI 3 s kamerou verze 2.1 od stejného výrobce. Kvalita této kamery je oproti používané kameře OV7670 několikanásobně větší. Co se týče rozlišení, fotky jsou pořizovány ve formátu 3280 × 2464 pixelů o 24bitové hloubce. Podání obrazu je také mnohonásobně lepší a nedochází k částečnému zešedivění barev, jsou živější. Vyfocená fotka je na obrázku 53.



Obr. 51. Arduino + OV7670 – plechovka [autor]

Obr. 52. Mbed + OV7670 – plechovka [autor]

Obr. 53. Raspberry PI + Camera 2.1 [autor]

7.2 Využití platforem pro výuku

V první řadě začneme od platformy Arduino, které je na prvním místě co se týče využití jednoduššího hardwaru a výuky programování. Programovací jazyk Wiring je dost zjednodušený oproti jazyku C, případně C++, který se používá pro programování platformy Mbed. Také pro pochopení, jak vlastně základní součástky a zařízení fungují, se lépe píše pro Arduino. Velkou nevýhodou je velikost paměti a procesorová rychlost. Při komplikovanějších úlohách, jako je snímání obrazu, bylo velmi poznat rozdíl v době snímání snímku oproti platformě Mbed. Zde snímky v daleko větší velikosti byly snímány při přepočtu velikosti snímku přibližně 20 × rychleji. To je zapříčiněno hlavně daleko výkonnějším procesorem, na kterém Mbed funguje.

Jak již bylo řečeno, programovacím jazykem pro Mbed je jazyk C (C++). Z tohoto pohledu je daleko vhodnější do budoucna se učit programovat v tomto jazyce, jelikož tento programovací jazyk je více univerzální a jeho znalost najde daleko větší uplatnění v životě. Pro úplně začátečníky je ovšem strukturou komplikovanější než jazyk Wiring a tedy jeho pochopení může trvat déle.

Co se týče platformy Raspberry PI, tato platforma je dobrá pro mnohé profesionálnější aplikace, ale práce s PWM výstupy je značně složitá a pro začátečnické aplikace naprosto nevhodná. Její výhodou je samostatný operační systém a velké množství již zabudovaných funkcí, jako například program Python, Mathematica a další software pro psaní programů.

8 PRAKTICKÉ PŘÍKLADY A VÝUKOVÝ TUTORIÁL

Tato kapitola se bude věnovat základům programování ve dvou platformách, a to pro platformy Arduino Mega 2560 a vývojovou platformu Mbed NXP LPC1768. Následující úkoly budou sloužit pro základní seznámení s těmito platformami a osvojení si prostředí, ve kterých se tyto platformy programují. Závěrečným úkolem pro každou platformu bude připojení a oživení kamery OV7670.

8.1 Arduino Mega 2560

Obecné informace se nacházejí v kapitole 3.2.1 Arduino UNO a v kapitole 3.2.2 Arduino Mega 2560. Pro následující aplikace bude použito výkonnější Arduino Mega 2560 (ve všech příkladových úlohách značeno jen jako Arduino). Popsaný program pro psaní programů pro prostředí Arduino je popsaný v kapitole 4.2.1 IDE Arduino. Všechny příkladové programy jsou nahrané jako příloha k této diplomové práci.

Struktura programu

Následně je popsána obecná struktura programu pro Arduino.

```
int LED = 13;                    //V této části programu dochází k deklaracím
#include ‘‘knihovna.h‘‘           //globálních proměnných, nastavování hodnot pro
                                 //proměnné a připojování používaných knihoven.

void setup() {                   //Tělo programu, kde dochází k prvotnímu
pinMode(LED,OUTPUT);             //nastavování vstupně výstupních portů, rychlosti
}                                //komunikace a dalšího nastavování. V tomto těle
                                 //kód probíhá až na výjimky pouze jednou.

void loop() {                    //Hlavní tělo programu, kde se vykonává hlavní kód
digitalWrite(LED, HIGH);         //programu. Při dokončení kódu se vrací na začátek
                                 //tohoto těla a program vykonává kód znovu.
}
```

8.1.1 Úloha č. 1 – Rozblikání LED diody

Pro pochopení programování a přípravu na složitější úkoly budou vyzkoušeny základní příkazy pro komunikaci programu s PC a způsob nahrávání napsaného programu na desku Arduina. Název programu: „Ukol1_rozblikani_LED.ino“

Zadání

Vytvořte program pro blikání vestavěné diody na pinu 13. Způsob blikání by měl být řešen alespoň dvěma způsoby a vytvořený program nahrajte na Arduino.

Použité pomůcky

Arduino Mega 2560, USB kabel

Zapojení

V tomto příkladu nemusíme nic zapojovat k desce Arduino

Řešení

```
int LED = 13;                   //nadefinování proměnné LED na pin 13
int pauza = 1000;               //nastavení proměnné na hodnotu 1000

void setup() {
   pinMode(LED, OUTPUT);        //nastavení LED jako výstupní funkce
}

void loop() {
   digitalWrite(LED, HIGH);     //přivedení LOG 1 na pin LED (13)
   delay (500);                 //nastavení zpoždění na 500ms
   digitalWrite(LED, LOW);      //přivedení LOG 0 na pin LED (13)
   delay (pauza);               //nastavení zpoždění podle proměnné pauza = 1000ms
   digitalWrite(LED, HIGH);     //přivedení LOG 1 na pin LED (13)
   delay (2000);                //nastavení zpoždění na 2000ms
   digitalWrite(LED, LOW);      //přivedení LOG 0 na pin LED (13)
   delay (pauza);               //nastavení zpoždění podle proměnné pauza = 1000ms
}
```

Bonusový úkol

Nastavte blikání diody tak, aby se ve 3 krocích blikání diody zrychlovalo a pak se program opakoval. Využijte k tomu příkaz if().

8.1.2 Úloha č. 2 – Snímání zmáčknutí tlačítka a snímání fotorezistoru

V této úloze bude za úkol seznámit se s tlačítky, tedy půjde o práci s načítáním dat do Arduina. Dále se bude pracovat s fotocitlivým rezistorem, který reaguje na světlo a podle úrovně osvětlení se mu mění odpor. Název programu: „Ukol2_tlacitka_a_fotorezistor.ino“

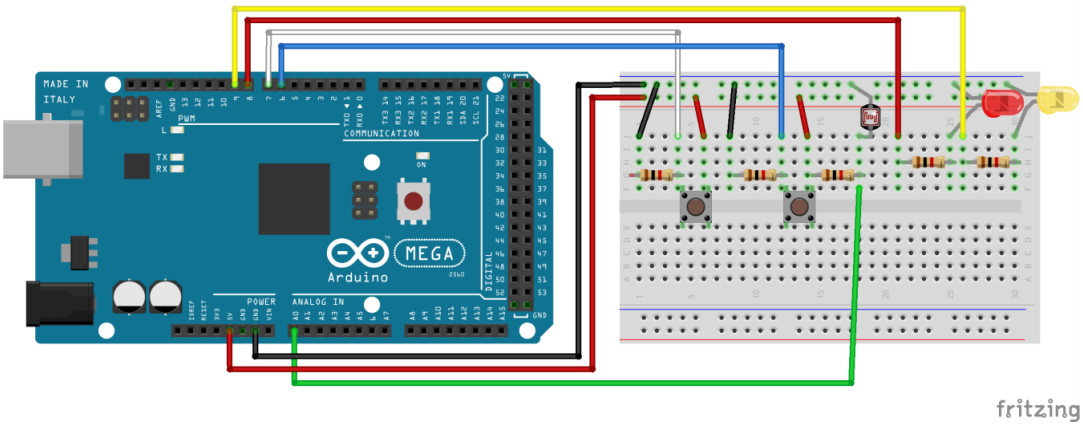
Zadání

Vytvořte program, ve kterém bude za úkol při držení tlačítka rozsvícení jedné diody a při držení druhého tlačítka blikání diody druhé podle úrovně osvícení fotorezistoru.

Použité pomůcky

Arduino Mega 2560, USB kabel, 2x LED, 5x rezistor 1kΩ, 1x fotorezistor, 2x tlačítko, propojovací vodiče.

Zapojení



Obr. 54. Zapojení pro úkol č. 2 [autor]

Řešení

```
int LED1 = 8;                    //deklarace proměnných
int LED2 = 9;
int tlac1 = 7;
int tlac2 = 6;
int svetlo = A0;

int stisk1 = 0;                  //nastavení prvotních hodnot na hodnotu 0
int stisk2 = 0;
int osvit = 0;
```

```
void setup() {
  pinMode(LED1,OUTPUT);          //nastavení výstupních a vstupních portů
  pinMode(LED2,OUTPUT);
  pinMode(tlac1,INPUT);
  pinMode(tlac2,INPUT);
  }

void loop() {
    stisk1 = digitalRead(tlac1);          //snímání stisknutí tlačítka 1
    if (stisk1==HIGH) {                   //pokud stisknuto tlačítko 1, rozsvítí LED1
       digitalWrite(LED1,HIGH);
       }
       else {                             //pokud není stisknuto, zhasne LED2
          digitalWrite(LED1,LOW);
          }

    stisk2 = digitalRead(tlac2);          //snímání stisknutí tlačítka 2
    if (stisk2==HIGH) {                   //pokud stisknuto tlačítko
       osvit = analogRead(svetlo);        //načte hodnotu z fotorezistoru
       digitalWrite(LED2,HIGH);           //rychlost blikání se mění podle
       delay(osvit);                      //úrovně osvícení rezistoru
       digitalWrite(LED2,LOW);
       delay(osvit);
       }
}
```

Bonusový úkol

Zařiďte, aby se program vykonával po stisknutí tlačítka a nebylo nutné tlačítka držet.

8.1.3 Úloha č. 3 – Práce se sériovou komunikací

V předchozích úlohách jsme si vysvětlili, jak probíhá zapojování obvodů a základní příkazy. V této úloze bude za úkol se seznámit se sériovým monitorem a práce s výpisem dat na sériový kanál a vyčítání dat z něj. Využijte připraveného programu: „Ukol3_seriova_komunikace.ino“

Zadání

Vytvořte program, který bude komunikovat po sériovém portu a bude rozsvěcovat dvě LED diody na dobu definovanou z počítače. Ovládat se může jedna nebo druhá dioda a blikat mohou rozdílně.

Použité pomůcky

Arduino Mega 2560, USB kabel, 2x LED, 2x rezistor 1kΩ, propojovací vodiče.

Zapojení

Zapojení zvolte podle uvážení za pomoci obrázku v předchozí úloze.

Řešení

Nadeklarujeme potřebné proměnné a nastavíme vstupně výstupní porty. Do těla podprogramu void setup() přidáme definici rychlosti komunikace po sériové lince

```
Serial.begin(9600);          //Nastaví rychlost komunikace se sériovou linkou
```

Následně v příkladu použijeme následující příkazy:

```
Serial.print("Ahoj");        //Vypíše na sériovou linku text „Ahoj“
Serial.println("Ahoj");      //Vypíše „Ahoj“ a vloží prázdný řádek
Serial.print(stav + prog()); //Vypíše hodnotu proměnné stav zvýšenou o hodnotu
                             //vrácenou z vyvolaného podprogramu prog(),
                             // například Serial.available()
Serial.read();               //Načte znak ze zásobníku sériové linky odeslané z PC
Serial.available();          //Vrátí počet znaků čekajících ve frontě na sériové lince z PC
while (podmínka) {}          //Vykonává program, dokud je splněná podmínka – pro
                             //čekání na zadání znaku na PC
if (podmínka) {}             //Vykoná se co je v závorkách, pokud je splněna podmínka
else if (podmínka) {}        //Vykoná se, když není splněna podmínka pro if, ale zde ano
```

```
else {}                          //Pokud nejsou předchozí podmínky splněny, vykoná se


void loop() {                    //hlavní tělo programu, do kterého se vždy vracíme
 if (Serial.available()>0)       //pokud je přijat znak, počet přijatých znaků je větší než 0
 {
  Serial.print("Prijato: ");
  delay(10);
  vypisznaku();                  //vyvolání podprogramu vypisznaku()
  zapniled();                    //vyvolání podprogramu zapniled()
  Serial.println();
  Serial.println("Pro LED1 odesli znak A, pro LED2 odesli znak B");
  }
}

void vypisznaku(){
  while (Serial.available()!=0)      //vykonávej, dokud je nějaký znak v zásobníku
    {
    prijem = Serial.read();          //načítá data ze sériové linky
    nastav = prijem;                 //uloží znaky do pomocné proměnné
    Serial.println(prijem);          //vypisuje přijaté znaky na sériovou linku
    delay(5);
    }
 }

void zapniled(){                 //podprogram pro porovnávání hodnot znaků a spouštění
                                 //konkrétní LED diody
 if (nastav == 65)               //Znak A v ASCII = 65, znak A = LED1
  {
   i=5;                          //Počet cyklů bliknutí
   rychlost();
   while (i!=0)                  //Zahájení cyklu blikání
   {
    digitalWrite(LED1,HIGH);
    delay(blik);
    digitalWrite(LED1,LOW);
    delay(blik);
    i--;
    }
  }
  else if ()                     //Stejným způsobem doplňte kód pro znak B = LED2

void rychlost(){                 //podprogram pro výpočet rychlosti blikání a jeho nastavení
}                                //využijte příkazu while () {} a použijte hodnoty ASCII pro
                                 //znaky 0 = 48 a pro znak 9 = 57
```

Bonusový úkol

Upravte program tak, aby se dal nastavit počet bliknutí u LED diod.

8.1.4 Připojení kamery OV7670

V předchozích úlohách byly vysvětlené základy propojování a jednodušší příkazy. Tato úloha je zaměřena na vyzkoušení si připojení kamery OV7670 k vývojové desce Arduino Mega2560 a její oživení.

Zadání

Připojte kameru OV7670 k vývojové desce Arduino a vyzkoušejte si funkčnost kamery a kvalitu výstupu obrazu. Využijte k tomu dostupných souborů: „receive_stream_mega.pde“ a „ov_fifo_test_mega.ino“.

Použité pomůcky

Arduino Mega 2560, OV7670 FIFO AL422, USB kabel, 3x logický napěťový konvertor, kontaktní nepájivé pole, propojovací vodiče.

Zapojení

Rozpis vodičů pro připojení naleznete v kapitole **4.1. Použité komponenty a elektrické zapojení** této diplomové práce. Pro snazší orientaci v zapojení můžete využít obr. 23. ve stejné kapitole, kde jsou obrazově znázorněny trasy jednotlivých vodičů.

Řešení

Nejprve si podle návodu připojte kameru k vývojové desce Arduino. Dejte pozor na to, abyste nepřipojili na piny kamery napětí 5 V z Arduina. Mohlo by snadno dojít ke zničení kamery. Následně si otevřete soubor „ov_fifo_test_mega.ino“ v programu Arduino IDE a tento program zkompilujte a nahrajte na vývojovou desku Arduina. Po tomto kroku si spusťte v programu Processing soubor „receive_stream_mega.pde“. V tomto souboru přepiště v řádku 73 číslo COM portu podle toho, na který COM port máte připojeno zařízení Arduino Mega 2560. To můžete zjistit na pravém spodním rohu programu Arduino IDE.

Vycházejte v tomto úkolu z kapitoly **4 – Využití Arduino Mega 2560 s kamerou OV 7670** v této diplomové práci, která je celá věnována této problematice.

8.2 Mbed NXP LPC1768

V předchozích úkolech bylo studentem vyzkoušeno zapojování, základní příkazy pro diody, sériovou komunikaci a logiku programu. U Mbedu je to částečně obdobné, ale program je psaný přímo v jazyce C++, tudíž jsou využívány jiné zápisy funkcí. Pro názornější ukázku bude programování řešeno v následujících úlohách.

8.2.1 Úloha č. 1 – Rozblikání LED diod

Zadání

Vytvořte program pro rozblikání dvou vestavěných diod na desce Mbed. Využijte k tomu funkci while(). Zjistěte, co se stane, když nepoužijete funkci while() v těle programu pro opakování blikání. Název programu „Uloha1_rozblikani_diod_LPC1768.bin“. Použijte přihlašovací údaje do online kompilátoru: login: MbedProfile, password: Mbedprofile.

Použité pomůcky

Mbed NXP LPC1768, USB kabel

Zapojení

LED diody jsou již vestavěny přímo na desce, není potřeba nic zapojovat.

Řešení

Nadeklarování používaných proměnných a knihovny pomocí příkazu *#include "mbed.h"*. Nadefinování výstupního portu pro led1 a led2 pomocí příkazu DigitalOut led1(LED1);. Oproti Arduinu se nastavení vstupně výstupních portů deklaruje v úvodní části souboru.

Následně v těle *int main()* vytvoříme cyklus *while()* stejně jako v kapitole 7.1.3.

Změna oproti Arduinu je v zápise rozsvícení LED, kdy použijeme tento zápis:

```
led1 = !led1;          //rozsvítí led1
wait(0.2);             //alternativa k delay(), nastavení zpoždění na 0.2 sekundy
led1 = 0;              //zhasne led1
wait(0.2);
```

Bonusový úkol

Vymyslete program pro blikání všech 4 diod, kdy blikají všechny zároveň a frekvence blikání se u každé mění.

8.2.2 Úloha č. 2 – Práce se sériovou komunikací

Pro porovnání rozdílu v programování mezi Mbed NXP LPC1768 a Arduinem Mega 2560 si zkusíme naprogramovat ovládání sériové komunikace i pro platformu Mbed.

Zadání

Naprogramujte pomocí sériové komunikace blikání diod vestavěných na vývojové desce Mbedu. Program bude načítat znaky a podle zvolených znaků se rozbliká buď dioda LED1 nebo LED2. Pro inspiraci můžete použít program v online kompilátoru XXX, použijte přihlašovací údaje: login: MbedProfile, password: Mbedprofile.

Použité pomůcky

Mbed NXP LPC1768, USB kabel

Zapojení

LED diody jsou již vestavěny přímo na desce, není potřeba nic zapojovat.

Řešení

Nadeklarujeme potřebné proměnné a nastavíme módy diod na výstupní. Dále tu přidáme nastavení pro komunikaci s PC a to: *Serial pc(USBTX, USBRX);*. V těle *int main()* nastavíme cyklus while tak, aby se vykonával pořád do kola. Z něj se pak bude vyvolávat podprogram, který bude sloužit pro ověřování stisknutí tlačítka a nastavení zapnutí konkrétních LED. Podprogram se vyvolá následujícím příkazem: vyvolejprg(); a tělo podprogramu se zapíše: *void vyvolejprg() {instrukce}*. Díky těmto příkazům se můžeme přepínat mezi podprogramy. V podprogramu využijeme příkazu *char c = pc.getc();*, který příchozí znak (jen jeden) ze zásobníku na sériové lince uloží do proměnné *c*. Tím dojde k vymazání znaku ze zásobníku a můžeme načítat znak nový. Porovnání znaku můžeme udělat následovně: if (c == ‘A‘) {} = pokud je znak uložený v proměnné c roven znaku A, vykonej.

Oproti Arduinu, znak načtený se neukládá jako hodnota ASCII, ale jako znak. Pro výpis informací z Mbedu na PC využijte příkaz:

```
pc.printf((char*)c);          //vypíše hodnotu uloženou v proměnné c
pc.printf("A\n");             //Vypíše na obrazovku znak „A“ a odřádkuje
```

Bonusový úkol

Upravte program tak, aby se po sériové lince dala ovládat rychlost blikání LED diod.

8.2.3 Připojení kamery OV7670

V předchozích úlohách byly vysvětlené základy propojování a jednodušší příkazy. Tato úloha je zaměřena na vyzkoušení si připojení kamery OV7670 k vývojové desce Mbed NXP LPC1768.

Zadání

Připojte kameru OV7670 k vývojové desce Mbed a vyzkoušejte si funkčnost kamery a kvalitu výstupu obrazu. Využijte k tomu online kompilátoru s přihlašovacími údaji login: MbedProfile, password: Mbedprofile na webové stránce *http://os.mbed.com* a programu „OV7670 Grabber USB V1.5.exe“.

Použité pomůcky

Mbed NXP LPC1768, OV7670 FIFO AL422, USB kabel, 2x rezistor 10 kΩ, kontaktní nepájivé pole, propojovací vodiče.

Zapojení

Rozpis vodičů pro připojení naleznete v kapitole **5.1. Použité komponenty a elektrické zapojení** této diplomové práce. Pro snazší orientaci v zapojení můžete využít obr. 36. ve stejné kapitole, kde jsou obrazově znázorněny trasy jednotlivých vodičů.

Řešení

Připojte kameru OV7670 k vývojové desce Mbed NXP LPC1768, následně se přihlaste na internetové stránky os.mbed.com a spusťte si online kompilátor. V online kompilátoru si vyberte složku OV7670_Test_Code a otevřete si soubor „main.cpp“. Následně zkompilujte soubor, který se uloží do počítače a ten nahrajte na vývojovou desku. Poté si spusťte program „OV7670 Grabber USB V1.5.exe“. Nastavte si port COM podle toho, kam je připojeno zařízení Mbed. To se dá zjistit v ovládacích panelech v nabídce „správce zařízení“. Klikněte na tlačítko INIT a tlačítkem SNAP vyfotíte fotku. Vyzkoušejte si různé nastavení obrazu a podívejte se, jak se mění výsledný obraz na zvoleném formátu obrazu.

Vycházejte v tomto úkolu z kapitoly **5 – Využití Mbed NXP LPC1768 s kamerou OV 7670** v této diplomové práci, která je celá věnována této problematice.

ZÁVĚR

Cílem této diplomové práce bylo zhodnocení komerčně dostupných prostředků pro výrobu bezpečnostní kamery, návrh a realizace této kamery včetně programového vybavení a otestování její funkčnosti. Dále bylo za úkol vytvoření studijního materiálu do předmětu Kamerové systémy.

V teoretické části práce bylo nastíněno fungování kamer z pohledu optiky a detekce obrazu. Jsou zde popsány principy průchodu světla optickou částí kamery včetně úpravy obrazu. Další věcí bylo zhodnocení nejčastěji používaných elektronických částí pro snímání obrazu a převedení světelné podstaty na podstatu elektrickou. Nejčastějšími prvky jsou polovodičové snímací čipy typu CCD a CMOS. Řešené jsou i elektronické prvky pro úpravu samotného datového toku uvnitř kamery. Prostudovány byly elektronické součásti využívané ve vývojových platformách. U vybraných vývojových platforem byly uvedeny technické specifikace.

Pro výrobu kamery byly vybrány dvě vývojové platformy pro připojení kamery OV7670 s FIFO bufferem AL422 a to Arduino Mega 2560 a Mbed NXP LPC 1768. Dále byla přidána vývojová platforma Raspberry PI 3 s vlastní kamerou verze 2.1 od stejné firmy. Praktická část je zaměřena na vytvoření těchto kamer a popsání principu jejich fungování. Jsou tu uvedeny postupy pro připojení těchto platforem k počítači a jejich programové výbavy. Propojení kamer je řešené formou soustavy vodičů, rezistorů a logických převodníků napětí. Vše je řešeno přehledně pomocí schémat zapojení. Programové vybavení je popsáno formou vývojových diagramů, kde jsou vysvětleny logické operace s daty a průchod signálu v těchto programech. Programové vybavení je poté přílohou k této diplomové práci.

Výstupná kvalita obrazu je porovnána podle výstupů jednotlivých vývojových platforem. Zjistilo se, že nejlepší kvalita výstupu je z kamery od Raspberry PI, která značně převyšuje maximální kvalitu kamery OV7670. Důvodem je větší rozlišovací schopnost kamery a také kvalitnější snímací prvek. Dále bylo zjištěno, že datový přenos z kamery připojené pomocí Mbed NXP LPC1768 je daleko vyšší než u platformy Arduino Mega 2560 při použití stejné kamery a to OV7670. To je převážně dáno výkonností daných platforem.

V závěru práce jsou vytvořené podklady pro přiblížení programování vývojových platforem Arduino a Mbed, jejich komunikace se základními hardwarovými prostředky a způsob komunikace těchto platforem s počítačem pomocí sériové linky. Vytvořené jsou úkoly pomáhající pochopení programování těchto platforem vytvořené pro studijní účely v předmětu Kamerové systémy. Vývojové platformy a kamery budou poskytnuty pro studentské účely. Všechny programy využité v této části jsou uvedeny v příloze k diplomové práci.

SEZNAM POUŽITÉ LITERATURY

[1] ADÁMEK, Milan. *Základní prvky kamery* [prezentace]. Zlín, 2012 Univerzita Tomáše Bati.

[2] *Zobrazení čočkami* [online]. [cit. 2018-02-12]. Dostupné z: http://www.gymhol.cz/projekt/fyzika/05_cocky/05_cocky.htm

[3] SMEJKAL, Janko. Clona. *Hľbka ostrosti poľa – Ako na ňu?* [online]. 2015 [cit. 2018-02-13]. Dostupné z: http://jankosmejkal.sk/fotografia/hlbka-ostrosti-pola.html

[4] PIHAN, Roman. *Použití filtrů v digitálním věku* [online]. 2007 [cit. 2018-02-13]. Dostupné z: https://www.digimanie.cz/pouziti-filtru-v-digitalnim-veku/1956

[5] ALEŠ, Bezděk. *Spektrum - přepočty vlnových délek, frekvencí, teplot* [online]. [cit. 2018-02-20]. Dostupné z: http://www.asu.cas.cz/~bezdek/fyzika/vlnDelka_frekvence_teplota.php

[6] ALOIS, Vávra. *CCD kamery* [online prezentace]. [cit. 2018-03-5]. Dostupné z: http://slideplayer.cz/slide/3058830/

[7] PETR, Schmid. *Snímání obrazu* [online]. Blatná, 2011 [cit. 2018-03-5]. Dostupné z: https://www.soublatna.cz/dokumenty/opvk11/dum/ovy-elt.pdf

[8] JAKUB, Jirsa. ALDEBARAN GROUP FOR ASTROPHYSICS. *Nová technologie kamerových systémů na obzoru!* [online]. 2017, **15**(21) [cit. 2018-03-6]. ISSN 1214-1674. Dostupné z: https://www.aldebaran.cz/bulletin/2017_21_kub.php

[9] *CMYK to RGB* [online]. 2016 [cit. 2018-05-15]. Dostupné z: http://www.ginifab.com/feeds/pms/cmyk_to_rgb.php

[10] CRAZDUDE. *CMYK vs RGB – What is the difference and why does it matter?* [online]. 2017 [cit. 2018-05-15]. Dostupné z: https://crazdude.com/2017/08/17/cmyk-vs-rgb/

[11] KRATOCHVÍL, Tomáš. *Digitální video: Využití při výuce Televizní techniky a Videotechniky* [online]. Brno, 2002 FEKT, Vysoké učení technické v Brně [cit. 2018-03-06]. Dostupné z: http://www.elektrorevue.cz/clanky/02062/index.html

[12] DOLEJŠÍ, Tomáš. *Gamma korekce polopatě* [online]. 2017 [cit. 2018-03-06]. Dostupné z: https://www.fotoradce.cz/gamma-korekce-polopate

[13] *Stavebnice robota RP6 V2 C-Control* [online]. [cit. 2018-03-12]. Dostupné z: https://www.conrad.cz/stavebnice-robota-rp6-v2-c-control.k191584?icc=category-carousel-2level&icn=toprate-roboti-stavebnice

[14] *GCC and Make - A Tutorial on how to compile, link and build C/C++ applications* [online]. [cit. 2018-03-12]. Dostupné z: https://www3.ntu.edu.sg/home/ehchua/programming/cpp/gcc_make.html

[15] HIENZSCH, Dan. *Arduino from Scratch Part 15 – Design Requirements* [online]. 2015 [cit. 2018-03-12]. Dostupné z: https://rheingoldheavy.com/arduino-from-scratch-part-15-design-requirements/

[16] TRÁVNÍČEK, Samuel. *[PIC programování] – Úvod - Allcomp Blog* [online]. 2016 [cit. 2018-03-15]. Dostupné z: https://www.allcomp.cz/blog/?p=233

[17] PINKER, Jiří. *Mikroprocesory a mikropočítače*. Praha: BEN - technická literatura, 2004. ISBN 80-730-0110-1.

[18] TOUŠEK, Dominik. *USB 1.0, 1.1, 2.0, 3.0 - MediaWiki SPŠ a VOŠ Písek* [online]. 2010 [cit. 2018-04-02]. Dostupné z: http://wiki.sps-pi.cz/index.php/USB_1.0,_1.1,_2.0,_3.0

[19] GANESH. *USB 3.2 Update to Bring 20 Gbps Bandwidth: USB 3.1 Type-C Cables Compulsory* [online]. 2017 [cit. 2018-04-02]. Dostupné z: https://www.anandtech.com/show/11667/usb32-update

[20] ARDUCAM. *OV7670_CMOS_Camera_Module_REVB_DS.doc* [online]. [cit. 2018-03-20]. Dostupné z: https://www.openhacks.com/uploadsproductos/ov7670_cmos_camera_module_revc_ds.pdf

[21] OMNIVISION. *OV7670_DS (1.01).fm* [online]. 2005 [cit. 2018-03-20]. Dostupné z: https://www.voti.nl/docs/OV7670.pdf

[22] *OV 7670 Camera Module / OV7670 Kamera Module For Arduino...* [online]. [cit. 2018-03-20]. Dostupné z: https://www.tokoarduino.com/ov-7670-camera-module/

[23] *[SOLVED] OV7670 pin out and STM32 port?* [online]. 2014 [cit. 2018-03-20]. Dostupné z: https://www.edaboard.com/showthread.php?313404-OV7670-pin-out-and-STM32-port

[24] *Arduino UNO Rev3 - HW Kitchen* [online]. [cit. 2018-02-28]. Dostupné z: https://www.hwkitchen.cz/arduino-uno-rev3/

[25] *Arduino Mega 2560 Rev3* [online]. [cit. 2018-02-28]. Dostupné z: https://store.arduino.cc/arduino-mega-2560-rev3

[26] *Mbed NXP LPC1768 Getting Started - Handbook* [online]. [cit. 2018-02-28]. Dostupné z: https://os.mbed.com/handbook/mbed-NXP-LPC1768-Getting-Started

[27] *Raspberry Pi 3 Model B 64-bit 1GB RAM - RPishop.cz* [online]. [cit. 2018-02-28]. Dostupné z: http://rpishop.cz/raspberry-pi-3b/283-raspberry-pi-3-model-b-64-bit.html

[28] KIRBIZ, David Sanz. *Arduvision 02 Mega* [online]. 2016 [cit. 2018-03-17]. Dostupné z: https://github.com/dasaki/arduvision/tree/master/arduvision_02_mega

[29] DE MARCHI, Edoardo. *OV7670_TestCode - Test Code for OV7670 Camera module with FIFO AL422*[online]. [cit. 2018-03-21]. Dostupné z: https://os.mbed.com/users/edodm85/code/OV7670_Test_Code/

[30] *Arduino - Software* [online]. [cit. 2018-04-16]. Dostupné z: https://www.arduino.cc/en/Main/Software

[31] *Environment (IDE) \ Processing.org* [online]. [cit. 2018-04-15]. Dostupné z: https://processing.org/reference/environment/

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ALU	Aritmetic Logic Unit
A/D	Analogově digitální převodník
CCD	Charge-coupled Device
CISC	Complete Instruction Set Computer
CMOS	Complementary Metal-Oxide Semiconductor
CPU	Central Processing Unit
D/A	Digitálně analogový převodník
EPROM	Erasable Programmable Read-Only Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
FIFO	First in first out
GND	Zem
HDMI	High-Definition Multimedia Interface
IR	Infra red
LED	Light-Emitting Diode
PC	Osobní počítač
PROM	Programmable Read-Only Memory
PWM	Pulse Width Modulation
RAM	Random Acces Memory
RISC	Reduced Instruction Set Computer
ROM	Read-Only Memory
USART	Universal Synchronous / Asynchronous Receiver and Transmitter
USB	Univerzal Serial Bus
VCC	Napájení

SEZNAM OBRÁZKŮ

SEZNAM TABULEK