UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Využití kamery pro zpracování obrazu v embedded systémech

Jiří Hakl

Bakalářská práce
2014

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2013/2014**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jiří Hakl**
Osobní číslo:	**I09013**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Komunikační a mikroprocesorová technika**
Název tématu:	**Využití kamery pro zpracování obrazu v embedded systémech**
Zadávající katedra:	**Katedra elektrotechniky**

Zásady pro vypracování:

Cílem práce je zprovoznit kameru na embedded zařízení. V teoretické části bude popsáno získávání obrazové informace, možnosti uložení obrazových dat, nejpoužívanější kompresní algoritmy a formáty pro uložení obrazu. Teoretická část bude dále obsahovat rozbor používaných snímacích systémů. V praktické části bude zapotřebí oživit zařízení s vestavěnou kamerou nebo kameru k zařízení připojit. Na získaná obrazová data z kamery následně aplikovat základní algoritmy zpracování obrazu pro úpravu těchto dat do požadované podoby. Zobrazení zpracovaných snímků bude provedeno buď na zařízení s displejem, nebo budou obrazová data vyslána do PC, kde proběhne požadované zpracování.
Zásady pro vypracování:
1.Analýza problematiky spojené se získáváním obrazové informace.
2.Rozbor snímačů obrazu a porovnání jejich vlastností.
3.Zprovoznění kamery na embedded zařízení.
4.Na získaná data z kamery aplikovat základní algoritmy zpracování obrazu.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

[1] KLÍMA, Miloš. Zpracování obrazové informace. 1. vyd. Praha: České vysoké učení technické, 1999, iii, 177 s.
ISBN 80-010-1436-3
[2] FRIBERT, Miroslav. Základy zpracování obrazu. Vyd. 1. Pardubice: Univerzita Pardubice, 2006, 104 s.
ISBN 80-719-4901-9
[3] CASTLEMAN, Kenneth R. Digital image processing. Upper Saddle River: Prentice-Hall, 1996, xvii, 667 s.
ISBN 01-321-1467-4

Vedoucí bakalářské práce: **Ing. Pavel Chmelař**
Katedra elektrotechniky

Datum zadání bakalářské práce: **20. prosince 2013**
Termín odevzdání bakalářské práce: **9. května 2014**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Zdeněk Němec, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2014

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 30. 5. 2014 Jiří Hakl

Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce Ing. Pavlu Chmelařovi, za cenné rady a věcné připomínky, které napomohly k vytvoření této bakalářské práce.

Anotace

Bakalářská práce měla za cíl zprovoznění kameru na embedded systému. Embedded systém komunikuje s kamerou prostřednictvím sériové linky a provádí zpracování dat obdržených kamerou. Úprava zpracovaných dat se provádí v podobě negativu nebo Laplaceova operátoru a výsledek v podobě snímku zobrazuje na LCD displeji. Embedded systém je řízen programem na PC.

Klíčová slova

digitalizace, kódování, embedded systém, kamera, obrazová informace, zpracování obrazu

Title

Application of the camera for image processing in embedded systems

Annotation

The aim of this bachelor thesis is putting a camera on an embedded system into operation. Embedded system communicates with the camera via the serial line and performs the processing of the data received by the camera. The processed data are modificated as negative image or via Laplace operator. The resulting images are shown on the LCD display. The embedded system is controlled by software on PC.

Keywords

digitalization, encoding, embedded system, camera, image information, image processing

Obsah

Seznam zkratek

A/D	Analog/Digital
APS	Active Pixel Sensor
D/A	Digital/Analog
BMP	Windows Bitmap
CCD	Charged Coupled Device
CCITT	Telegraph and Telephone Consultative Committee
CCM	Core Couled Memory
CMOS	Complementary Metal Oxide Semiconductor
DCT	Discrete cosine transform
I^2C	Inter Integrated Circuit
I/O	Input/Output
ISO	Intrenational Organization for Standardization
JPEG	Joint Photographic Experts Group
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MCU	Microcontroller Unit
PNG	Portable Network Graphics
RAM	Random Access Memory
RLE	Run Length Encoding
RX	Receiver
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
TX	Transmitter
UART	Universal Asynchronous Receiver and Transmitter
USART	Universal Synchronous Asynchronous Receiver and Transmitter
USB	Universal Serial Bus
WHT	Walsh-Hadamardova Transform

Seznam obrázků

Seznam tabulek

Úvod

Embedded systém představuje elektronický obvod v nejrozšířenější variantě počítačových systémů řízený především procesorem nebo mikroprocesorem, pro konkrétně řešenou problematiku. V této práci se embeddend systém zaměřuje na digitální zpracování obrazové informace. Bakalářská práce je rozdělena do čtyř kapitol.

První kapitola této práce se zabývá problematikou zpracování obrazové informace po jednotlivých krocích. Na začátku kapitoly se řeší digitalizace obrazové informace z analogové veličiny a je rozdělena na dvě části. Jedna část popisuje vzorkování a druhá se zabývá kvantizací a jakých chyb se dopustíme. Následně se navazuje na kódování obrazové informace, kde se popisují vybrané komprimační metody. Závěr kapitoly pojednává o formátech vhodných k ukládání digitalizované obrazové informace.

Druhá kapitola pojednává o obrazových snímačích, které se v současnosti používají, a je rozdělena na tři hlavní části. První část rozebírá snímače založené na technologii s vázaným nábojem (známé pod zkratkou CCD). Podrobně popsané jednotlivé konstrukční řešení a principy používané pro získání dat ze snímače. Druhá část kapitoly zkoumá snímače vyráběné technologickým postupem CMOS. Stejným způsobem rozebírá technické řešení a kroky získávání hodnot jako u CCD snímačů. Obě části popisují přeměnu fotoelektrického signálu na elektrický. Poslední popisovanou oblastí této kapitoly tvoří stručné porovnání jednotlivých řešení konstrukčních typů obrazových snímačů a výhod či nevýhod.

Kapitola třetí je rozdělena do dvou podkapitol řešící konstrukční a softwarové provedení embedded zařízení. Část zabývající se konstrukcí toho to zařízení popisuje celkové zapojení skládající se z následujících komponentů: kamera C328, LCD displej, převodník pro komunikaci s počítačem a LED diody pro signalizaci. Softwarové řešení je rozděleno na tři části a popisuje komunikaci s kamerou, zpracování a zobrazení i zkompletování do funkčního celku.

Poslední čtvrtá kapitola pojednává o zpracování obdržených dat z kamery C328 a úpravu pomocí použitých metod na snímky. Důležitou částí této kapitoly je separace RGB barev. Popisuje jednotlivé kroky pro získání jednotlivých barev s 8 bitovou barevnou hloubkou. Dále jsou popsány vybrané metody pro úpravu snímků s vyobrazením a vývojovými diagramy. Poslední komentovanou částí kapitoly je řešení ovládání kamery C328 prostřednictvím embedded zřízení přes ovládací program v počítači. Nechybějí ani ukázky zrodových kódů přiložených v přílohách.

1 Zpracování obrazové informace

V současné době je zpracování obrazové informace významnou a samostatnou oblastí z velké skupiny obecného zpracování informací. Jedná se o rozsáhlou problematiku, kterou lze rozdělit do několika skupin, jako např. digitalizace obrazu, rekonstrukce obrazu, kódováni a analýza obrazu.(1)

Následující odstavce popisují, jak se zpracovává obrazová informace a jaké se používají obrazové formáty.

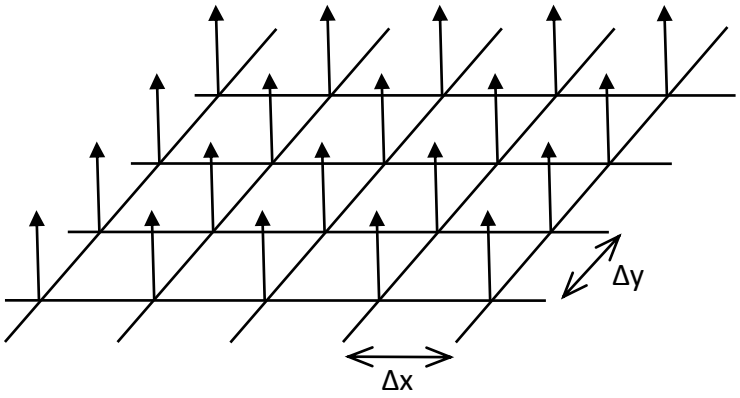
1.1 Digitalizace obrazové informace

Pojmem digitalizace v obrazové technice se rozumí převod spojité obrazové informace na digitální obrazovou informaci, která je reprezentovaná dvojrozměrnou maticí. Digitalizace se skládá ze dvou kroků, které nazýváme vzorkování (rozdělení spojité obrazové informace na stejné časové intervaly) a kvantizace (rozdělení spojité obrazové informace do jednotlivých hladin).(1)

1.1.1 Vzorkování obrazu

Vzorkování je prvním krokem, aby se získala dvojrozměrná matice nesoucí hodnoty obrazové informace a následně mohly aplikovat další kroky. Pochopení teoretického procesu vzorkování obrazu, neboli obrazové informace, se demonstruje na ideálním vzorkování. Ideální vzorkování se provádí polem Diracových impulsů (nekonečné pole impulsů, které vychází z Diracovy funkce) na obrázku (Obrázek 1). Matematicky, lze vyjádřit pole Diracových impulsů následující rovnicí (1.1).(1),(2)

$$s(x,y) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta(x - m\Delta x, y - n\Delta y) \qquad (1.1)$$



Obrázek 1 – Pole Diracových impulsů (1)

Pro získání výsledku (1.2), kterým je navzorkovaný obraz, je zapotřebí provést součin obrazové funkce s funkcí vyjadřující pole Diracových impulsů (1):

$$f_P(x,y) = f_1(x,y) \cdot s(x,y) \tag{1.2}$$

kde: $f_1(x,y)$ – je analogová funkce nesoucí obrazovou informaci,
$s(x,y)$ – je pole Diracových impulsů.

Po dosazení (1.1) a (1.2) dostáváme konečné vyjádření vzorkovací funkce (1.3), která nese hodnotu obrazové informace v daném bodě.(1)

$$f_P(x,y) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} f_1(j\Delta x, k\Delta y) \cdot \delta(x - j\Delta x, y - k\Delta y) \tag{1.3}$$

kde: $f_1(j\Delta x, k\Delta y)$ – obrazová funkce na j-tém intervalu Δx osy x a k-tém intervalu osy y,
$\delta(x - j\Delta x, y - k\Delta y)$ – je Diracův impuls, který je posunut do j-tého intervalu Δx a k-tého intervalu Δy vůči počátečním souřadnicím

Po uplatnění tohoto postupu v reálném vzorkování narážíme na omezující parametry. Ve skutečnosti má vzorkovaný obraz konečnou velikost a použití nekonečného vzorkovacího pole Diracových impulsů je zbytečné. Při realizaci vzorkování stačí uvažovat pole Diracových impulsů o velikosti vzorkovaného obrazu. Dalším omezující parametr je vzorkovací impuls. Diracův impuls je pro realizaci neproveditelný a je nahazován impulsem, který má určitý tvar a šířku (např. obdélník). Reálné vzorkování je po zmíněných podmínkách popsáno následujícím vztahem rovnice (1.2).(1) Vzorkovací funkce $s(x,y)$ není tvořena Diracovými impulsy ale reálným vzorkovacím impulsem. Impuls volíme podle následujícího výrazu (1.4), tak aby platilo:

$$\iint_{-\infty}^{\infty} p(x,y)dxdy = 1 \tag{1.4}$$

Výsledná navzorkovaná reálná obrazová funkce je popsána následující rovnicí (1-5).(1)

$$f_P(x,y) = \sum_{j=-J}^{J} \sum_{k=-K}^{K} f_1(j\Delta x, k\Delta y) \cdot p(x - j\Delta x, y - k\Delta y) \tag{1.5}$$

1.1.2 Kvantování obrazu

Obrazová informace není už analogově spojitá ve smyslu časové oblasti, jelikož je diskretizovaná (rozdělená po impulsech neboli vzorcích), ale stále je analogově spojitá v amplitudě. Aby se obrazová informace mohla číslicově zpracovávat je zapotřebí přiřadit číselnou hodnotu konkrétní amplitudě příslušného vzorku. Tímto procesem se zabývá kvantizace, která má jednorozměrný charakter na rozdíl od vzorkování (dvojrozměrný charakter). Spočívá obecně v přidělení konečného počtu diskrétních (kvantizačních) úrovní každému analogovému plošně diskretizovanému vzorku.(1)

Pro přidělení číselné hodnoty konkrétní amplitudě je zapotřebí určit počet kvantizačních úrovní. Jelikož se hodnoty amplitud vzorků pohybují v rozmezí, které je dáno maximální a minimální hodnotou. Tím je dána oblast, kterou budeme diskretizovat v několika hladinách. K určení hladin použijeme Weber-Fechnerův zákon (1.6), který popisuje vnímání jasových diferencí, proti pozadí s rovnoměrným jasem.(1)

$$\Delta B = k \cdot B_0 \tag{1.6}$$

kde: ΔB je diference jasu, kterou vyjádříme jako $B_{i+1} - B_i$, B_0 je jas pozadí a dosadíme jako B_i,
k je konstanta a je rovna *0,015 až 0,02*. Po dosazení a úpravách dostáváme vztah (1.7).(1)

$$B_{i+1} = B_i \cdot (1 + k) \tag{1.7}$$

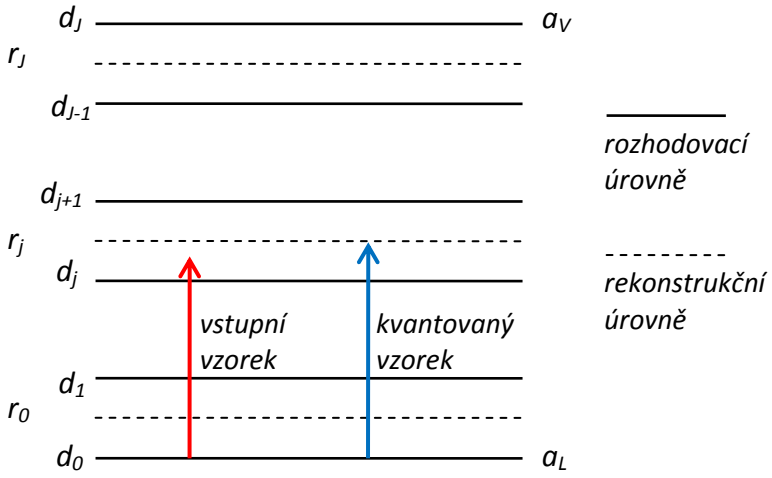
K určení rozlišitelných stupňů je zapotřebí použít, vycházíme z následujícího vzorce (1.8).(1)

$$\frac{B_{max}}{B_{min}} = (1 + k)^n \tag{1.8}$$

Hodnota B_{max} odpovídá maximální hodnotě jasu, B_{min} je minimální hodnotou jasu, n je počet rozlišovacích stupňů (kvantizační hladiny) a zjistíme je podle upraveného vzorce (1.9).(1)

$$n = \frac{log \frac{B_{max}}{B_{min}}}{log(1 + k)} \tag{1.9}$$

Praktickými testy byly potvrzeny uvedené úvahy pro obrazovou informaci a pro realizaci většiny aplikací postačuje vyjádření kvantizačních hladin v osmi bitech. U složitějších aplikací se provádí experimentální výpočet kvantizačních hladin. Důvodem toho to postupu je, že citlivost lidského zraku je závislá na jasu a Weber-Fechnerův zákon je odvozen pro stejnosměrné pozadí jasu.(1)



Obrázek 2 – Princip kvantování (1)

Dalším krokem v kvantizaci obrazové informace je stanovení rekonstrukční a rozhodující úrovně znázorněné na obrázku (Obrázek 2). K určení se využívá zjištěný počet kvantizačních úrovní. Stanovení rekonstrukční úrovně a zároveň i rozhodovací úrovně je využito výrazu kvantizační chyby, která má být co nejmenší. Definovaná je jako střední kvadratická hodnota a pro J kvantizačních úrovní je ve tvaru (1.10).(1)

$$\varepsilon = \int_{a_L}^{a_V} (f - f_r)^2 \cdot p(f) df = \sum_{j=0}^{J-1} \int_{d_j}^{d_{j+1}} (f - r_j)^2 \cdot p(f) df \qquad (1.10)$$

kde f – je původní analogová funkce obrazové informace,
f_r – je funkce rekonstruované obrazové informace vzniklá z r_j,
$p(f)$ – je hustota pravděpodobností výskytu jednotlivých úrovní signálu a pro velký počet kvantizačních úrovní je možné považovat $p(f)$ za konstantu v jednom kvantizačním pásu (lze vytknout před integrál jako $p(r_j)$),
r_j – je rekonstrukční úroveň (odpovídá digitálnímu číslu při kvantování).

Po integraci výrazu (1.10) dostáváme výsledek (1.11),

$$\varepsilon = \frac{1}{3} \sum_{j=0}^{J-1} p(r_j) \cdot \left[(d_{j+1} - r_j)^3 - (d_j - r_j)^3 \right] \qquad (1.11)$$

kde dosazením za r_j dostaneme z podmínky derivace kvantizační chyby (1.10) následující výraz pro dosazení.(1)

$$\frac{d\varepsilon}{dr_j} = 0 \Rightarrow r_j = \frac{d_{j+1} + d_j}{2}$$

Z podmínky vyplývá, že optimální rekonstrukční úroveň se nachází mezi dvěma rozhodujícími úrovněmi.(1) Dosadíme-li výraz z podmínky do rovnice pro kvantizační chybu (1.10) vychází takto:

$$\varepsilon = \frac{1}{12}\sum_{j=0}^{J-1} p(r_j) \cdot \left[d_{j+1} - d_j\right] \tag{1.12}$$

Po drobných úpravách a zavedení diskrétní pravděpodobnosti je výsledkem rovnice pro kvantizační chybu (1.13),(1)

$$\varepsilon = \frac{1}{12}\sum_{j=0}^{J-1} p_j(r_j) \cdot \Delta_{\mathrm{j}}^2 \tag{1.13}$$

kde: $p_j(r_j)$ - je diskrétní pravděpodobnost a to ve tvaru $\int_{d_j}^{d_{j+1}} p(r_j)df = p(r_j)\Delta_j$,

$\Delta_{\mathrm{j}} = d_{j+1} - d_j$

Pokud je ve všech kvantizačních úrovní hustota pravděpodobnosti výskytu stejná lze psát

$$\varepsilon = \frac{1}{12}\Delta^2 \tag{1.14}$$

Víme-li *n* kvantizačních hladin, lze dosadit hodnotu za výraz $\Delta = \frac{1}{2^n}$

1.2 Komprese obrazu

V předešlých dvou podkapitolách je popsána digitalizace obrazové informace. Digitalizovaný obraz reprezentovaný dvojrozměrným polem, potom v každém bodu nese hodnotu, která může představovat vyjádření barvy nebo jasu. Toto pole je někdy označováno za bitovou mapu. Například obrázek s velikostí 640 × 480 obrazových bodů, při osmi bitovém kvantování a pro tři barvy bude téměř zabírat 1MB na paměťové kartě, pevném disku nebo při přenosu. S nároky na kvalitnější obrázky, pak výsledná velikost rapidně roste. Z těchto důvodů se aplikuje na obrazovou informaci komprese, která má zredukovat obrazovou informaci na velikost, s kterou se dá snáze pracovat.(1)

Metody komprese obrazu lze rozdělit na dva principy a to:

- redukce redundance (bezeztrátový),
- redukce irelevance (ztrátový).

Bezeztrátový princip komprese, známý také jako entropické kódování, vychází z pojmu entropie. Entropie vyjadřuje množství užitečné informace obsazenou v každém obrazu. Čím větší je hodnota entropie obrazu, tím více bitů je potřebných pro uložení informace. Odečtením entropie obrazu od skutečné velikosti uložení obrazu představuje redundanci. Ve většině případů obraz obsahuje obrazové body, které jsou statistickými vazbami mezi

sebou svázány (korelovány). Vyloučením těchto vazeb v obrazu lze dosáhnout výrazného snížení potřebných bitů pro uložení, aniž bychom snížili informační obsah obrazu.(1)

Druhý ztrátový princip komprese je založený na jistém způsobu vyhodnocování obrazu podle vyhodnocovacího zařízení. Ve většině případů je vyhodnocovacím zařízením pozorovatel a redukce informace je ovlivněna vlastnosti zraku. Tudíž je zbytečné uchovávat či přenášet informace, které pozorovatel není schopen zpracovat. Ztrátová komprese snižuje obsah obrazové informace, tak aby pozorovatel tuto změnu nepoznal.(1)

1.2.1 Kódování běhu (RLE)

Kódování běhu nebo také proudové kódování RLE (Run Length Encoding) je jednoduchou metodou. Spočívá v nahrazení posloupnosti stejných symbolů kódem tzv. ESC sekvencí, který popisuje konkrétní symbol nebo hodnotu a počet opakování. Vhodné využití tohoto principu nastává u často se opakujících znaků za sebou, nebo kde záleží na rychlosti a jednoduchosti algoritmu. Například použitím RLE na sekvenci symbolů přirozených čísel 0 až 7, bude výstupní (zakódovaná) sekvence do tvaru uvedenou v tabulce (Tabulka 1).(1)

Tabulka 1 – Metoda kódování běhu RLE

Vstupní sekvence:	2	4	6	5	5	5	5	6	3	1	0	4	4	4	4	4	4	4	3	2
Výstupní sekvence:	2	4	6	ESC	5	4	6	3	1	0	ESC	4	7	3	2					
Bez ESC sekvence	2	4	6	1	5	4	6	3	1	1	1	0	1	4	7	3	2			

Znak ESC informuje o následujících dvou hodnotách, které jsou součástí ESC sekvence nebo také RLE paketu. První hodnota, hned za znakem ESC, udává opakující se symbol v tabulce např. hodnota 5 a druhé číslo počet opakování (v tabulce reprezentováno hodnotou 4). Součásti RLE paketu je i znak ESC, který je zapotřebí také přenést a přiřadit mu ve většině případů konkrétní hodnotu. Nejvhodnější je vybrání nejméně pravděpodobné hodnoty z uvedeného intervalu a potom je nutné zvolenou hodnotu nebo znak zakódovat do ESC sekvence. V posledním řádku tabulky je dosazena hodnota za ESC = 1 a dále ukázka zakódování zvolené hodnoty do ESC sekvence.(1)

Nastane-li v sekvenci symbolů k opakování některého znaku, které přesáhne maximální hodnotu intervalu, je zapotřebí zopakovat ESC sekvenci jak ukazuje následující tabulka (Tabulka 2).(1)

Tabulka 2 – RLE kódování pro přesáhnutí intervalu

Vstupní sekvence:	3	5	5	5	5	5	5	5	5	5	5	5	5	5	2
Výstupní sekvence:	3	ESC	5	7	ESC	5	6	2							

Metodu RLE lze různě modifikovat (např. vynechání symbolu ESC, nebo zakódování všech znaků do ESC sekvence) a použít i na binární, bytové, pixelové či řádkové úrovni, popřípadě je různě kombinovat.(1)

1.2.2 Huffmanovo a aritmetické kódování

Huffmanovo kódování vymyslel v roce 1957 O. Huffman, které používá celočíselný počet bitů k reprezentování jednoho symbolu. Metoda kódování je založena na nestejné délce kódu, přičemž kratší hodnoty kódu jsou přidělovány více se vyskytujícím symbolům ve zprávě a delší kódy dostanou symboly s menším výskytem ve zprávě. Huffmanův kód, v porovnání s metodami kódování založené na celočíselném počtu bitů, dosahuje nejkratšího kódu.(1)

Princip Huffmanova kódování je ukázán v následující tabulce (Tabulka 3), kde jsou pro příklad uvedeny symboly s_1, s_2, s_3, s_4 a s_5, které chceme zakódovat. Tyto symboly se vyskytují ve zprávě s pravděpodobnostmi $p_1 = 0,5$; $p_2 = 0,25$; $p_3 = 0,125$; $p_4 = 0,0625$; $p_5 = 0,625$. Prvním krokem Huffmanova kódování je seřazení symbolů sestupně podle příslušných pravděpodobností a následním sloučením dvou symbolů s nejmenší pravděpodobností do nového symbolu. Pravděpodobnost nového symbolu, pak udává výskyt jednoho, či druhého sloučeného symbolu. Jednotlivým sloučeným symbolům se přiřadí binární kód odpovídající hodnotě nula nebo jedna, jak ukazuje tabulka (Tabulka 3). Postupným sloučením symbolů se získávají nové symboly s větší pravděpodobností výskytu až do vzniku posledních dvou symbolů s největší pravděpodobností. Pak tyto symboly dostanou nejkratší kód, který je možný získat (nula nebo jedna). Zpětným skládáním binárních kódů, podle slučovaných symbolů, získáme hodnotu kódu, pro zakódování daného symbolu.(1)

Tabulka 3 – Huffmanovo kódování

Znak	**s_1**	**s2**	**s3**	**s4**	**s5**
pravděpodobnost	0,5	0,25	0,125	0,0625	0,0625
binární hodnota				0	1
Znak	**s_1**	**s_2**	**s_3**	**s_{45}**	
pravděpodobnost	0,5	0,25	0,125	0,125	
binární hodnota			0	1	
Znak	**s_1**	**s_2**	**s_{345}**		
pravděpodobnost	0,5	0,25	0,25		
binární hodnota		0	1		
Znak	**s_1**	**s_{2345}**			
pravděpodobnost	0,5	0,5			
binární hodnota	0	1			
Výsledný binární kód znaku:	0	10	110	1110	1111

Pro dosažení lepších výsledků, než s využitím Huffmanova kódu, se používá aritmetického kódování. Na rozdíl od Huffmanova kódu nemá aritmetické kódování omezení v podobě celočíselného počtu bitů pro symbol, ale kódují se najednou celé skupiny symbolů jedním číslem 0 až 1.(1)

Principem aritmetického kódování je rozdělení intervalu $\langle 0, 1)$ na jednotlivé intervaly, které jsou reprezentovány znaky (s_1, s_2,..., s_{n-1}, s_n). Velikosti jednotlivých intervalů jsou závislé na hodnotách pravděpodobností výskytů znaků (p_1, p_2,..., p_{n-1}, p_n), následujícím přepisem (1.15), (3)

$$\langle 0, p_1); \langle p_1, p_1 + p_2); \cdots; \langle p_1 + \cdots + p_{n-1}, p_1 + \cdots + p_n) \tag{1.15}$$

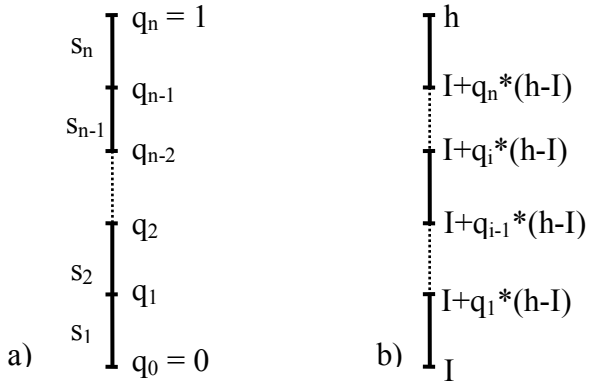
a pro zjednodušení znázornění intervalů použijeme kumulativní pravděpodobnosti (1.16) vyobrazené i na obrázku (Obrázek 3a).(3)

$$q_0 = 0, q_1 = p_1, q_n = p_1 + p_2 + \cdots + p_n \tag{1.16}$$

Po rozdělení intervalu $\langle 0, 1)$ na znaky probíhá kódování. V aritmetickém kódování se přímo nepočítá s intervalem $\langle 0, 1)$, ale postupně se zmenšuje interval, který vymezuje oblast výsledného čísla. Potom tento opakující se interval označený jako I, kterým značíme znak s_i, zapíšeme jako $\langle q_{i-1}, q_i)$ a jeho hodnoty se spočítají dle následujícího vztahu (1.17) a znázorněný na obrázku (Obrázek 3b).(3)

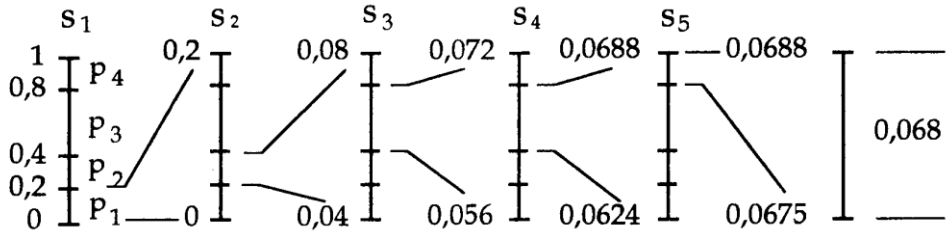
$$I = \langle I + q_{i-1} \cdot (I - h), I + q_i \cdot (I - h)\big) \tag{1.17}$$

kde I, h - hodnoty stávajícího intervalu
q_i, q_{i-1} – kumulativní pravděpodobnosti



Obrázek 3 – Princip aritmetického kódování: a) rozdělení intervalu; b) výpočet nové hodnoty intervalů při kódování znaku

Názorným příkladem postupu aritmetického kódování je následující obrázek (Obrázek 4). Kódovaná zpráva se skládá ze symbolů s_1, s_2, s_3, s_3 *a* s_4 vycházející z množiny čtyř symbolů s_1, s_2, s_3, s_4. Pravděpodobnosti výskytu symbolů jsou $p_1 = 0{,}2$; $p_2 = 0{,}2$; $p_3 = 0{,}4$; $p_4 = 0{,}2$.(1)



Obrázek 4 – Postup aritmetického kódování

Dekódování u aritmetického kódování probíhá podobným způsobem jako opačný proces a znaky ve stejném pořadí, jak byly zakódovány, se dekódují. Postupuje se podle stejných kroků jako v kódování, kde se rozdělí stávající interval *I* podle pravděpodobností výskytu znaků. V každém kroku hledáme interval za pomocí vztahu (1.18) a (1.17), ve kterém se nachází zakódovaná hodnota daného znaku. Znak je následně dekódován a určuje nový interval dle rovnice (1.17).(3)

$$q_{i-1} \leq \frac{c-I}{h-I} < q_i \tag{1.18}$$

kde q_i, q_{i-1} – jsou kumulativní pravděpodobnosti,
c – je kódovaná hodnota znaku,
I, h – jsou hodnoty současného intervalu.

1.2.3 Diskrétní kosinová transformace

Diskrétní kosinová transformace (DCT) patří mezi nejpoužívanější lineární ortogonální transformace v digitálním kódování obrazové informace a je založená na ortogonálním systému diskrétních kosinových funkcí. Diskrétní kosinová transformace vychází z diskrétní Fourierovy transformace.(1)

Nevýhodou Fourierovy transformace je převod reálné matice, která reprezentuje obraz, na komplexní matice. Komplexní jádro diskrétní Fourierovy transformace dle vztahu (1.19) lze rozdělit na sudou (reálnou) složku a lichou (imaginární) složku, které v oboru reálných funkcí nepředstavují systém bázových funkcí.(1)

$$\varphi_{ux} = \cos\left(2\pi\frac{ux}{N}\right) + \mathrm{j}\sin\left(2\pi\frac{ux}{N}\right) \tag{1.19}$$

DCT po použití věty o symetrii a výpočtů normalizačních koeficientů je definovaná vztahem (1.20), a pro zpětnou transformaci je uveden vztah (1.21). Podrobné odvození DCT naleznete v literatuře.(1)

$$F(u,v) = \frac{2}{N}\cdot K(u)K(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\left[\cos\frac{(2x+1)\pi}{2N}u + \cos\frac{(2y+1)\pi}{2N}v\right] \tag{1.20}$$

kde $F(u,v)$ – je spektrum obrazové informace,
$f(x,y)$ – vstupní obrazová informace,
$K(u)K(v)$ – jsou normovací koeficienty $K(0) = \frac{1}{\sqrt{2}}$, $K(n) = 1$ pro n = 1,…,N-1,
u, v – jsou hodnoty v rozmezí $0 \leq x < 2N$, kde x je hodnota u nebo v,
N – je počet bodů rozšíření $f(x,y)$ kolem bodu 0.

$$f(x,y) = \frac{2}{N}\cdot K(u)K(v)\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)\left[\cos\frac{(2x+1)\pi}{2N}u + \cos\frac{(2y+1)\pi}{2N}v\right] \tag{1.21}$$

1.2.4 Walsh-Hadamardova transformace

Walsh-Hadamardova transformace (WHT) představuje nejznámější nesinusovou ortogonální transformaci. Výpočet WHT se vyznačuje tím, že vyžaduje pouze slučovací operace. Transformace je založena na systému Hadamardových matic, které mají příslušné řádky a sloupce navzájem ortogonální. Prvky nabývají pouze hodnot +1 nebo -1.(1)

Hadamardova matice $[H_{2N}]$ řádu 2N je konstruovaná z matic řádu N dle vztahu (1.22).

$$[H_{2N}] = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix} \tag{1.22}$$

Hadamardova matice pro $H_1 = 1$, takže pro H_2 a vyšší platí (1.23) (1).

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{1.23}$$

Dvojrozměrnou WHT je možné definovat pro N^2 obrazových prvků jako vztah (1.24).(1)

$$[W(u,v)] = \frac{1}{N} \cdot [H_N(u,v)] \cdot [f(x,y)] \cdot [H_N(u,v)] \tag{1.24}$$

a zpětná Walsh-Hadamardova transformace je definována v následujícím tvaru (1.25).(1)

$$[f(x,y)] = \frac{1}{N} \cdot [H_N(u,v)] \cdot [W(u,v)] \cdot [H_N(u,v)] \tag{1.25}$$

Použijí-li se pravoúhlé (obdélníkové) kmity s periodou $\frac{1}{N}$ a amplitudami od -1 do +1 pro interpretování jednotlivých řádků Hadamardovy matice, pak získán soubor Walshových funkcí. Použitím Walshových funkcí může být dvojrozměrná WHT zapsaná v následujícím tvaru (1.26).(1)

$$W(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cdot (-1)^{p(u,v,x,y)} \tag{1.26}$$

kde $p(u,v,x,y) = \sum_i^{n-1} (u_i x_i \oplus v_i y_i)$

u_i, v_i, x_i, y_i – i-tý bit čísla u, v, x, y v binární reprezentaci

u, v, x, y – binární čísla např. u = 10 bude $u_3 = 1$, $u_2 = 0$, $u_1 = 1$, $u_0 = 0$

symbol $\oplus$ - značí binární sčítání (mod 2)

$f(x,y)$ – je vstupní obrazová funkce,

$\frac{1}{N}$ – je perioda.

1.3 Obrazové formáty pro ukládání

V současnosti je mnoho obrazových formátu, které se používají pro obrazovou informaci. Hlavní předností těchto formátu, kterých se využívá u digitalizované obrazové informace, je především schopnost redukce dat z důvodů přenosu po sítích a zabírání menšího prostou na úložných medií se zachováním kvality vstupního obrazu. Jednotlivé obrazové formáty se dělí na rastrové či vektorové. Rastrové formáty se dají ještě rozdělit na bezeztrátové a ztrátové. Následující odstavce popisují některé vybrané rastrové formáty používaných pro ukládání obrazové informace.(1),(5)

1.3.1 Formát RAW

Jedná se o obrazový formát využívaný především v profesionálních a poloprofesionálních fotoaparátech. Jedna se o bezeztrátový obrazový formát, který je založený na ukládání dat přímo ze snímače fotoaparátu, aniž by se zpracovával nějakými aritmetickými operacemi způsobující kompresi dat. Hlavní výhodou tohoto formátu je úprava snímku podle představ uživatele a na rozdíl od ostatních obrazových formátů např. JPEG se dosahuje lepší kvality výsledných snímků.(4),(5)

Nevýhodou formátu RAW je datová náročnost, jelikož na paměťovém mediu bude zabírat více místa než obrázek ve formátu JPEG. Data formátu RAW je ukládán ve tvaru: datová hlavička, popis expozice a popis uspořádání snímače. Formát RAW má u výrobců různé označení a proto výrobci fotoaparátů nabízejí i vlastní software pro zpracování dat ve formátu RAW.(4),(5)

1.3.2 Formát BMP

Formát BMP vyvinula společnost Microsoft s názvem Windows Bitmap, jenž je znám pod zkratkou BMP. Jedná se o grafický formát navržený pro program Windows k uschovávání a přenosu grafiky. Windows bitmap je poměrně jednoduchý formát umožňující kompresi pomocí modifikovaného algoritmu RLE popsaný v kapitole (Kódování běhu (RLE)) a reprezentací obrazových bodů paletou nebo složkami RGB (red, green, blue).(1)

Soubor dat grafického formátu BMP se skládá ze čtyř hlavních částí, které na sebe navazují (1):

- BitmapFileHeader
- BitmapInfoHeader
- paleta barev
- vlastní obrazová dat

Obraz ve Windows Bitmap se ukládá od nespodnějšího řádku a zleva doprava, až po horní řádek. Jinak řečeno od levého dolního rohu po pravý horní roh s možností samostatného komprimováni každého řádku. Pro zachování hardwarové přenositelnosti se řádky doplňují nulami, pokud je to zapotřebí. Doplňování se provádí, aby délka řádků v bytech byla celistvým násobkem čtyř. Toto je jednou variantou, kterou Windows Bitmap formát podporuje.(1)

Druhou variantou reprezentace obrazu ve formátu BMP je definován strukturou RGBQuad. Struktura přiděluje každému obrazovému bodu složky typu BYTE. První až třetí složka struktury RGBQuad reprezentuje velikost složky R, G a B. Čtvrtá složka, označena jako rezervovaná, se používá pro zprůhlednění. Taktéž je známý pod názvem alfa kanál. Wndows Bitmap interpretuje barvy podle bitové šířky jednotlivých obrazových bodů a připouští tyto možnosti (1):

- 1 bit – binární obraz, paleta má dvě položky (černá a bílá)
- 4 bity – obraz obsahuje 16 různých barev
- 8 bitů – maximálně 256 různých barev
- 24 bitů – obrazový bod popsán strukturou RGBQuad a max. 16,7 milionu barev

Více informací o tomto formátu a podrobnějším rozboru najdete v literatuře (1).

1.3.3 Formát JPEG

V letech 1989 – 1991 pracovní skupina Joint Photographic Experts Group vytvořena organizacemi ISO (Intrenational Organization for Standardization) a CCITT (Telegraph and Telephone Consultative Committee), která navrhla standard JPEG pojmenovaný podle této skupiny.(1)

Standard JPEG zahrnuje již popsané metody pro kompresi obrazové informace, kromě Walsh-Hadamardovy transformace. Ta je součástí formátu JPEG2000. Dalšími podporovanými kompresními metodami ve standardu JPEG jsou prediktivní, slovníkové, transformační metody a vektorová transformace. Zástupcem transformačních metod ve standardu JPEG je DCT.(1)

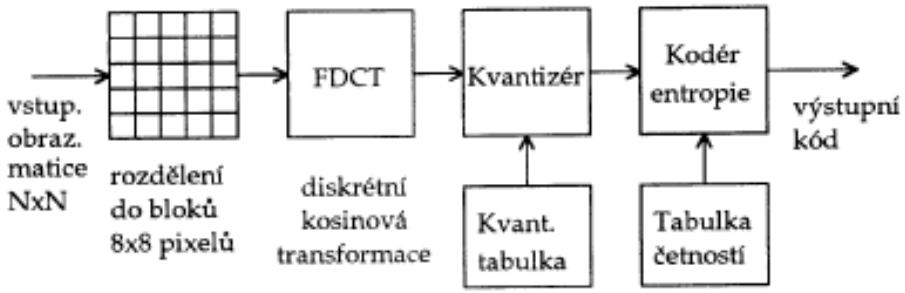
Kompresi ve standardu JPEG lze provést jakoukoliv z uvedených metod:

- Sekvenční kódování
- Bezeztrátová komprese
- Progresivní kódování
- Hierarchické kódování

Následující odstavce popisují sekvenční kódování, jelikož se jedná o hlavní komprimační metodu JPEG.

Sekvenční kódování

Sekvenční kódování vyobrazené obrázkem (Obrázek 5), na následující straně, v podobě blokového schématu, znázorňuje postup komprese obrazové informace v několika navazujících krocích a zároveň je základním typem standardu JPEG.(1)



Obrázek 5 – Blokové schéma JPEG formátu (1)

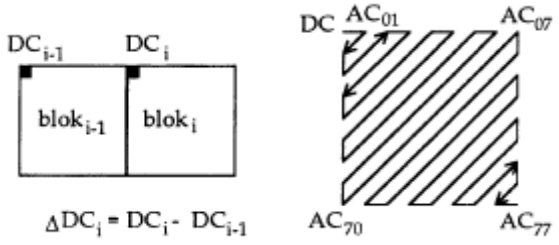
Obrazová informace reprezentovaná maticí se v prvním kroku rozděluje na bloky o rozměrech 8 × 8 obrazových bodů, jak je znázorněno na obrázku (Obrázek 5). Vstupní hodnoty obrazových bodů jsou transponovány z původních hodnot $[0 - 2^{n-1}]$ na rozsah hodnot $[2^{n-1} - 2^{n-1} - 1]$, přičemž hodnota n = 8, nebo 12 bitů v sekvenčním kódování.(1)

Po takto rozdělené obrazové matici se aplikuje na každý blok DCT dle vztahu (1.21), kde hodnotu N reprezentuje konkrétní číslo rovné osmi. Z výsledného výrazu získáme transformaci, jejímž výsledem je 64 koeficientů. První koeficient představuje stejnosměrnou složku obrazu a zbylé hodnoty koeficientů představují prostorové harmonické kmitočtové složky výstupního obrazu. Následně probíhá kvantování podle vztahu (1.27) (1):

$$F^Q = Int\frac{F(u,v)}{Q(u,v)} \tag{1.27}$$

kde $Q(u,v)$ je definováno volitelnými kvantizačními tabulkami.

Posledním krokem sekvenčního kódování je entropické kódování. Ale nejprve než proběhne entropické kódování, je nutné zakódovat stejnosměrnou složku. Ta udává průměrnou hodnotu všech 64 obrazových bodů jednoho bloku obrazové matice. Následně stejnosměrné složky jednotlivých bloků vykazují vysokou korelaci k sousedním blokům, a proto postačuje kódovat stejnosměrnou složku jako rozdíl od hodnoty stejnosměrné složky z předešlého bloku, který ukazuje obrázek (Obrázek 6a). Potom čtení kvantovaných koeficientů se provádí postupem znázorněným na obrázku (Obrázek 6b), kde se může zakódovat podle Huffmanova či aritmetického kodéru.(1)



Obrázek 6 – a) Kódování stejnosměrné složky, b)Postup čtení „cik-cak“ koeficientů DCT

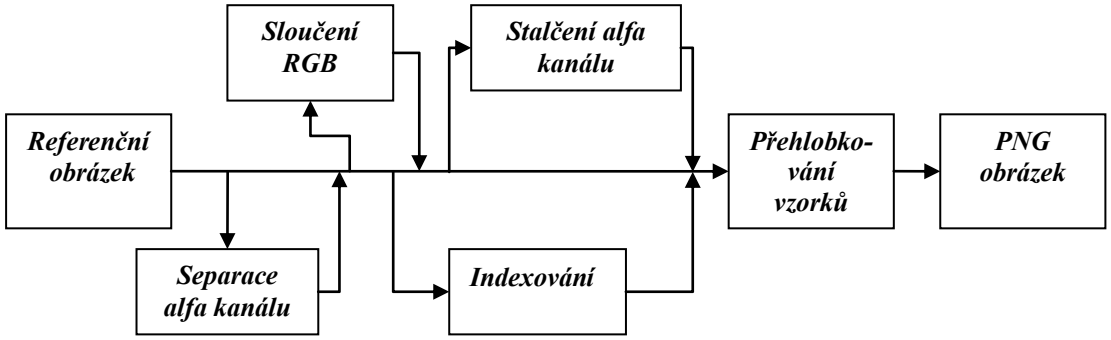
1.3.4 Formát PNG

Formát PNG (Portable Network Graphics) vytvořilo v roce 1996 konsorcium W3C a byl zkonstruován pro interpretaci obrázku na internetu. Jedná se o bezeztrátový formát umožňující zobrazení barev v několika režimech od indexování po Truecolor. Dokáže reprezentovat barevnou hloubku až do 48 bitů a využití alfa kanálu, jinak známé zprůsvitnění.(6)

Formát PNG obsahu i hlavičku souboru, ve které najde následující údaje (6):

- určení formátu PNG,
- délka soubodu,
- typ souboru v rámci PNG,
- kontrolní součet,
- popis obrázku,
- informace o autorovy a další.

Postup převodu snímku do PNG formátu ukazuje následující blokové schéma na obrázku (Obrázek 7).



Obrázek 7 – Transformace referenčního obrázku na PNG

Separace alfa kanálu – pokud referenční obraz obsahuje maximální hodnotu, pak alfa kanál může být vynechán a docílí se následně lepšího kódování.(7)

Indexování – se používá u následujících obrazových bodů (7):

- s hodnotou menší než 255,
- s barevnou hloubkou do 8bitů,
- s 8 bitovým alfa kanálem nebo bez alfa kanálu,
- každý průhledný nebo neprůhledný bod.

Indexování se skládá z palety, což je seznam položek tří barev po osmi bitech (červená, zelená, modrá) a alfa tabulky s osmi bity znázorněno na obrázku (Obrázek 8e.).(7)

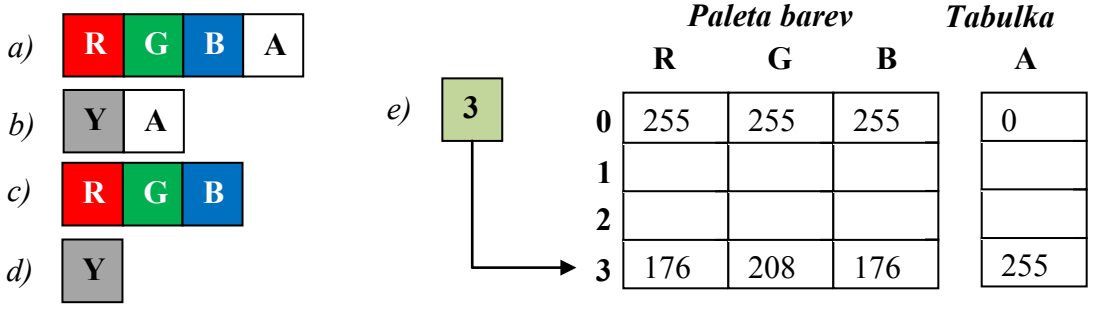
Sloučení RGB – provádí se v případech, kdy jsou vzorky se stejnou barevnou hloubkou v jednotlivých barevných kanálech (červeného, zeleného a modrého) nebo se jednotlivé barvy pixelu rovnají. Nastává sloučení jednotlivých barevných kanálů do jediného kanálu ve stupních šedi (jinak jasová složka).(7)

Stlačení alfa kanálu – je další operací, která se provádí pouze u neindexovaných obrázků. Existuje-li taková hodnota z RGB nebo ve stupni šedi, a všechny pixely se s touto hodnotou shodují a jsou transparentní od ostatních, pak alfa kanál tyto body může svou hodnotou reprezentovat. Tímto způsobem se v PNG formátu dosahuje stlačení snímku (7).

Přehloubkování vzorků – aplikuje se na sjednocení barevné hloubky snímku, jelikož nejsou všechny typy barevných hloubek ve formátu PNG podporovány. Minimální barevná hloubka výsledného obrázku v PNG je dána barevnou hloubkou referenčního snímku a nedochází ke ztrátě dat (7).

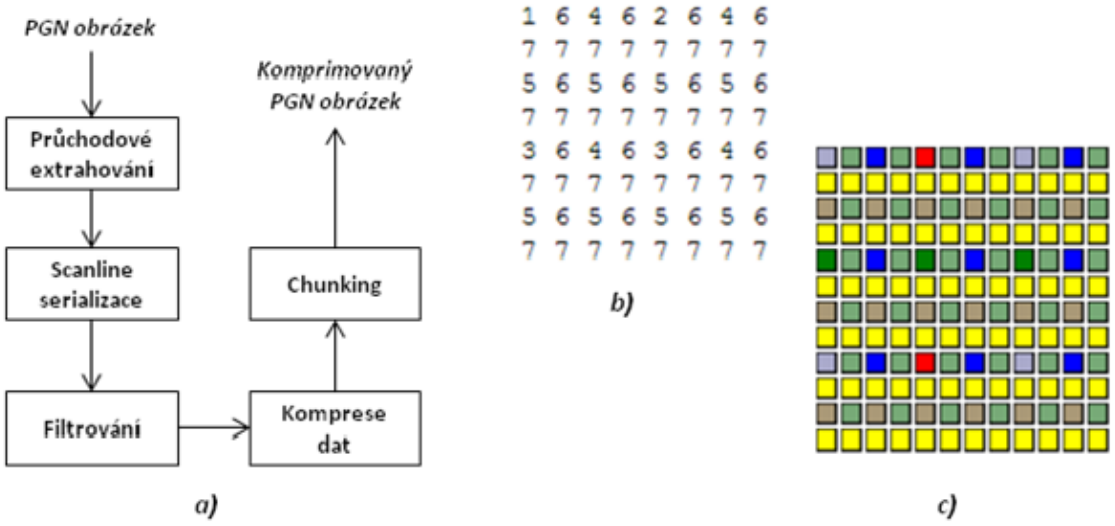
PNG obrázek se může transformovat z referenčního snímku do jedné z následujících typů zobrazených i na obrázku (Obrázek 8).(7)

- Truecolour s alfa kanálem – takto definovaný pixel se skládá ze čtyř vzorků
- Ve stupních šedi – pixel je definován dvěma vzorky
- Truecolour – pixel se skládá ze tří vzorků, i v tomto případě se může aplikovat alfa kanál, který je reprezentován hodnotou jedné dané barvy pixelu a posléze odpovídající barva pixelu je transparentní a zbylé barvy ne. Není-li takto definován alfa kanál, tak jsou všechny barvy neprůhledné.
- V odstínech šedé – pixel je definován jedním vzorkem. V tomto případě se může alfa kanál definovat jako v předešlém bodě, pak jednotlivé pixely budou průhledné. V opačném případě je pixel neprůhledný.
- Indexované barvy – každý pixel je reprezentován indexem, který odkazuje na tabulku s paletou barev a hodnotou alfa je-li k dispozici



Obrázek 8 – Typy transformací PNG obrázků a) Truecolour s alfa, b) ve stupni šedi s alfa, c)Truecolour, d) ve stupních šedi, e) indexované barvy

Kódování ve formátu PNG ukazuje následující obrázek (Obrázek 9a) a skládá se z pěti kroků. Prvním krokem je průchodové extrahování nebo také nazýváno jako prokládání PNG obrázku. Spočívá v rozdělení PNG obrázku na menší části obrazu, kde první tvoří hrubý náhled. Ostatní snímky doplňují hrubý náhled tak dlouho, až poslední část obrazu nevytvoří PNG obrázek. Toto rozdělení se provádí dvěma metodami a zde bude popsána metoda Adam7 vyobrazena na obrázku (Obrázek 9b). Metoda Adam7 je založena na sedmi průchodech očíslovaných podle obrázku (Obrázek 9b), které se v celém PNG obrázku opakují. Opakování všech sedmi průchodů ukazuje obrázek (Obrázek 9c), kde v prvním průchodu jsou vybrány všechny obrazové body šedé barvy. Pixely šedé barvy se pravidelně opakují po osmi obrazových bodech. Druhým průchodem se vybírají pixely označeny červenou barvou, přesně podle obrázku (Obrázek 9b) a v následujících průchodech se pixely vybírají totožným způsobem jako předešlé průběhy podle metody Adam7.(7)



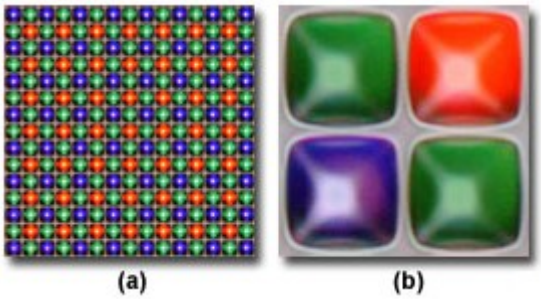
Obrázek 9 – Kódování PNG obrázků (7)

Scanline serializace navazuje na předešlý krok. Spočívá v seřazení jednotlivých pixelů v řádku za sebou a jeden seřazený řádek se nazývá scanline. Jednotlivé scanline se řadí podle průchodu pod sebe od levého horního okraje po spodní. Výsledek by vypadal asi následovně podle obrázku (Obrázek 9c). První s druhým řádkem scanline jsou tvořeny šedou barvou a třetí s čtvrtým červenou barvou. Další řádky scanline budou tvořeny zbylými barvami podle průchodů. Navazujícím krokem v kódování PNG obrázku je filtrování. Filtrují se jednotlivé seřazené scanline podle jednoho z definovaných filtrů a vznikne filtrovaná scanline. Následně tato scanline v kroku komprese dat proběhne podle jedné z definovaných komprimačních metod. Všechny definované komprimační metody jsou bezeztrátové a tvoří výstup nazývaný datastream (datový proud). Posledním krokem je Chunking. V Chunkingu se datastream rozdělí na menší bloky, které obsahují vlastní redundanční kontrolu. Více o tomto formátu naleznete na W3C.(7)

2 Obrazové snímače

V současné době se nejčastěji používají obrazové snímače, nebo jiným pojmenováním jako obrazové čipy, založené na technologiích CCD případně CMOS. Často se uplatňují v různých aplikacích od kamer, fotoaparátu, čtečkách čárových kódů po mobilní telefony, přenosné počítače a dalších zařízení. Použití nacházejí ale i v lékařství nebo v automatizaci a dalších odvětí.

Snímání obrazové informace snímači CCD či CMOS je založeno na stejném principu, který vychází z fotoelektrického jevu (popsaný v kapitole *Vzniku náboje a řízení hradel v CCD snímačích*) a jejich odlišnost je závislá pouze na použité technologii pří výrobě. V obou případech se jedná o obrazové čipy, které vytvářejí pouze černobílé snímky. Vytváření snímků v obrazových čipech je založeno na akumulaci náboje prostřednictvím světla tvořeného fotony, nehledě na jejich vlnovou délku. Pro vytvoření barevného obrázku je součástí každého obrazového snímače barevný filtr, který propouští tři základní barvy červenou, zelenou a modrou. Barvy filtru tvoří mozaiku vyobrazenou na obrázku (Obrázek 10a) rozprostírající se po celé ploše čipu. Kombinací jedné červené, jedné modré a dvou zelených filtrů dostaneme elementární obrazový bod, také označován jako pixel na (Obrázek 10b). Uvedený pixel v následujících kapitolách nemá nic společného s obrazovým bodem, ale slouží k pojmenování fotodiod.



Obrázek 10 – Barevný filtr a) mozaika na snímači, b) jeden pixel (8)

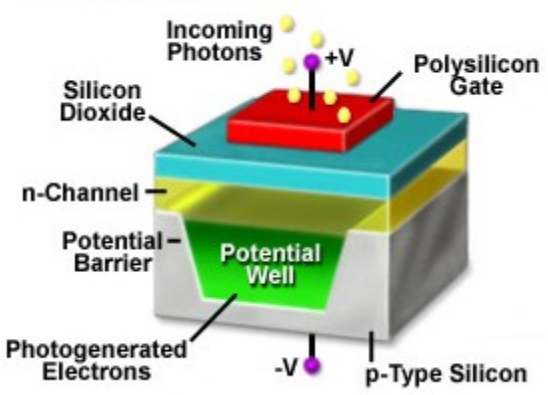
2.1 CCD snímače

Snímací čipy založené na technologii CCD (Charged Coupled Device), v překladu zařízení s vázanými náboji, byly vyvíjeny Dr. Willard Boyle a Dr. George Smith. V roce 1969 snímací čip CCD vznikl v laboratořích firmy Bell Laboratories. Používání CCD snímačů v televizních kamerách začalo roku 1975 a později se objevily i v jiných zařízení.(9)

2.1.1 Struktura CCD snímačů

Obrazové snímače CCD jsou tvořeny velkým počtem buněk, které jsou vyskládány vedle sebe a tvoří tak dvourozměrné pole. Jednotlivé buňky reprezentují detektory světla, které jsou tvořeny MOS kondenzátory (Metal Oxide Semikonductor). Jeden MOS kondenzátor je vyobrazen na obrázku (Obrázek 11). MOS kondenzátor je schopný pracovat jako fotodioda a akumulační zařízení, který uchovává náboj.(10)

Základem MOS kondenzátor na obrázku (Obrázek 11) je křemíková destička z polovodiče s vodivostí P (p-Type Silicon). Na této křemíkové destičce je zabudovaný kanál s vodivostí N polovodiče (n-Channel). Tyto dvě části tvoří PN přechod a reprezentují fotodiodu. Hradlo (Polysilicon) je tvořeno polykrystalickým křemíkem, který má schopnost propouštět viditelnou část světla až do vlnových délek kolem 400nm a tudíž nezabírá plochu pro dopadající fotony na rozdíl od vodivých materiálů jako např. hliník. Zároveň je hradlo oddělené od vzniklé fotodiody vrstvou oxidu křemičitého (Silicon Dioxide). Pod jeho plochou se nachází potenciálová oblast (Potential Well), ve kterém se akumuluje náboj.(10)

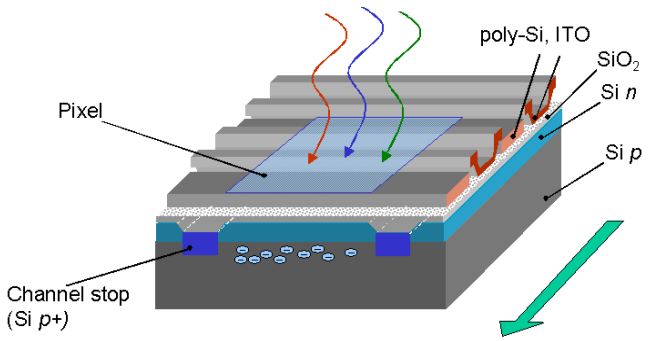


Obrázek 11 – Struktura MOS kondenzátoru (10)

Pole snímače je rozděleno do dvou hlavních částí. Tvoří je sériový a paralelní registr vyobrazené na obrázku (Obrázek 15) v kapitole (Vznik náboje a řízení hradel v CCD snímačích). Jeden registr se skládá z jednotlivých MOS kondenzátorů seřazených sériově a pomocí hradla uchovávají informaci v podobě náboje vytvořeného fotodiodou. Vhodným řízením jednotlivých hradel MOS kondenzátorů se docílí posouvání náboje a vytvoří se tak posuvný registr, který je podstatný pro fungování CCD snímače.(10)

Paralelním seřazením popsaných registrů vytvoříme fotocitlivou část snímače, kterou nazýváme paralelním registrem. Jednotlivé registry jsou odděleny od sebe po celé snímací ploše CCD snímače izolačními bariérami. Na obrázku (Obrázek 12) je vyobrazeno část posuvného paralelního registru vymezený izolačními bariérami (Channel stop). Paralelní posuvný registr je kolmí k sériovému posuvnému registru. Sériový posuvný registr se odlišuje od paralelního posuvného registru stíněním. Stínění má za cíl

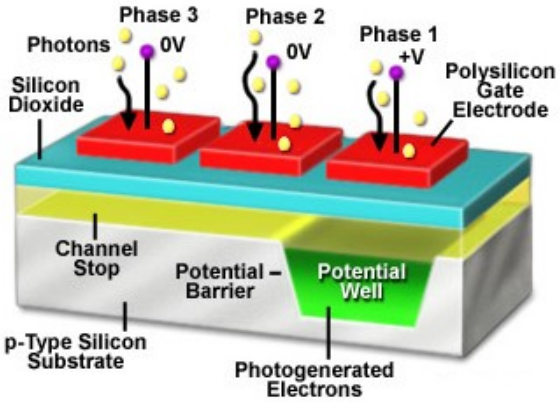
zamezení vzniku nábojů na sériovém registru, který slouží výhradně pro sériový přenos náboje na výstup ze snímače (10).



Obrázek 12 – Část paralelního registru (11)

2.1.2 Vznik náboje a řízení hradel v CCD snímačích

Necháme-li světlo, reprezentované fotony, dopadat na fotodiodu, která je součásti MOS kondenzátoru, začne se v potenciálové oblasti (Potential Well) na obrázku (Obrázek 13) vytvářet náboj. Náboj vzniká pomocí dopadajících fotonů na oblast zabudovaného kanálu (n-Channel žlutou barvou znázorněn). Vlivem velké energie fotonu se v krystalové mřížce polovodiče ve valenční vrstvě uvolní elektron. Přivedením kladného napětí na hradlo MOS kondenzátoru se začnou uvolněné elektrony pohybovat pomocí vytvořeného potenciálového pole. Uvolněný elektron ve vyčerpané oblasti přechodu PN vytvoří pár složený z elektronu a díry. Počet těchto párů elektron-díra je závislý na množství dopadajících fotonů, které určují výslednou velikost náboje.(10)

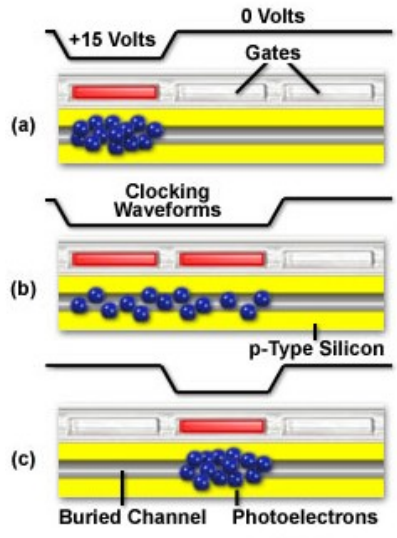


Obrázek 13 – Struktura jedné fotodiody v CCD snímačích (10)

Jak znázorňuje obrázek (Obrázek 13) je jedna fotodioda tvořena třemi hradly a často je toto uspořádání označováno za pixel. Hradla, v tomto uspořádání, jsou nejčastěji ovládána pomocí tří fází (třífázové schéma), kde každé třetí hradlo je připojeno ke stejnému okruhu napětím řídících hodin. Není to pouze jediné řešení ovládání hradel ale nejjednodušší a často používané (10).

V CCD snímačích jsou hradla využívána při vzniku a přemisťování náboje. Při získání náboje již popsaného jsou dvě ze tří hradel fotodiody aktivována větším napětím příslušných řídících hodin. Například první hradlo je nastaveno napětím fáze 1 (Phase 1) a současně druhé hradlo napětím fáze 2 (Phase 2). Třetí hradlo má malé nebo nulové napětí fáze 3 (Phase 3) a plní funkci bariéry (Channel Stop) mezi náboji sousedních fotodiod. Takto jsou nastavená hradla po celou dobu expozice. Dobou expozice se rozumí čas, po který na fotodiodu dopadají fotony. Po skončení expoziční doby se aktivuje pouze jedno hradlo řízené napětím fáze 1 řídících hodin. Zbylá hradla jsou nastavena na malá nebo nulová napětí. Volné elektrony z celé fotodiody se nahromadí pod hradlo řízené napětím fáze 1 řídících hodin. Náboj ovládaný napětím na hradle fáze 1 je na obrázku (Obrázek 13) vyobrazen potenciálovou oblastí (Potencial well).(10)

Druhým využitím hradel je přenos náboje pomocí tří fázového schématu ukázán na obrázku (Obrázek 14). Nahromaděný náboj pod hradlem řízený napětím fáze 1 řídících hodin, popsaný v předešlém odstavci znázorňuje obrázek (Obrázek 14a). Přivedeme-li na sousedící hradlo napětí fáze 2 řídících hodin, vyvolá rozprostření náboje pod obě hradla, které je vyobrazen na obrázku (Obrázek 14b). Uvedením hradla řízeného fázi 1 na nulové napětí, se všechen náboj posune pod hradlo s napětím fáze 2 vyobrazené na obrázku (Obrázek 14c). Stejný postup se opakuje i mezi hradly s fází 2 a fází 3 řídících hodin. Změna napětí na jednotlivých hradlech je postupná, a to z důvodů zamezení ztrát náboje při přenosu mezi jednotlivými hradly fotodiod. Tento mechanismus třífázového schématu je aplikován na sériovém registru, ale také i na paralelním registru.(10)



Obrázek 14 – Třífázové schéma řídících hodin (10)

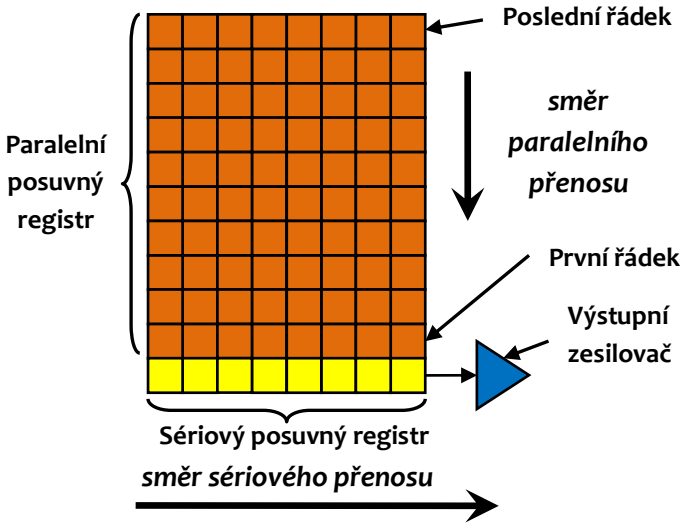
2.1.3 Architektura a princip CCD snímačů

CCD snímače se vyrábějí ve třech architekturách lišící se od sebe způsobem, jakým je náboj přemístěn na výstup. Jsou jimi:

- Full Frame architektura,
- Frame Transfer architektura,
- Interline Transfer architektura.

Všechny uvedené architektury využívají kombinace sériového a paralelního přenosu náboje. Tento koncept přenosu náboje, kterým je dosaženo sériového výstupu z dvourozměrného pole snímače, je charakteristický pro CCD snímače.

Full Frame architektura vyobrazena na obrázku (Obrázek 15) se skládá z paralelního a sériového registru. Celá plocha paralelního registru je citlivá na dopadající světlo a tvoří tak snímací pole na rozdíl od sériového registru. Kombinaci sériového a paralelního přenosu náboje zajišťují právě zmíněné registry, které je velmi snadné ovládat pomocí hradel. Využitím tří fázového schématu řízení hodin popsaný v předešlé kapitole se stávají oba registry posuvnými.(10)



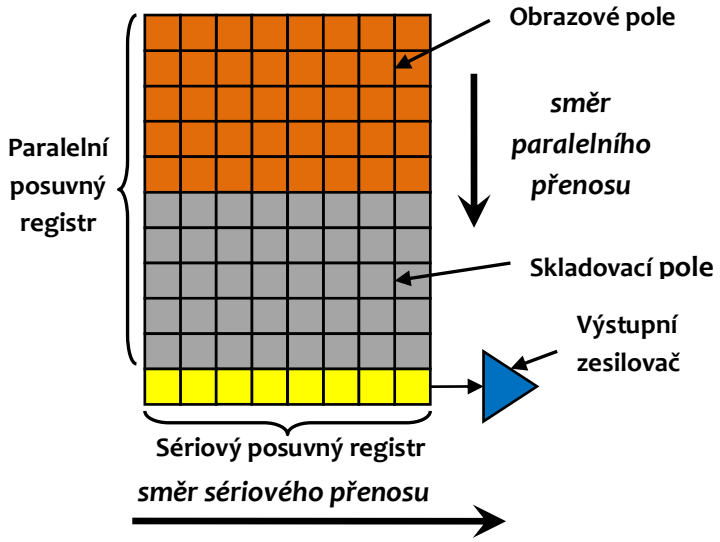
Obrázek 15 – Full-Frame architektura

Na fotodiodách, v obrazovém poli po skončení expoziční doby, se vytvoří náboj. V každém místě snímací plochy nemusí být stejný náboj a je závislý na množství dopadajících fotonů. Aby se tyto náboje dostaly na výstup ze snímače je zapotřebí je získat z paralelního posuvného registru. K tomuto přemístění se použije sériový posuvný registr sousedící v blízkosti paralelního posuvného registru. Začíná se náboji nejblíže k sériovému posuvnému registru, které se naráz jako jeden řádek přemístí do sériového posuvného registru. Uvolněná místa v prvním řádku se zaplní náboji z druhého řádku a stejným postupem se posouvají následující řádky směrem k sériovému posuvnému registru (paralelní přenos). V sériovém posuvném registru se náboje posouvají jednotlivě za sebou směrem k výstupu ze snímače, kde je výstupní zesilovač. Po vyprázdnění registru se opět

přemístí naráz další řádek nábojů a celý postup paralelního a sériového přenosu náboje se opakuje do doby, než v paralelním posuvném registru není žádný náboj.(10)

Frame Transfer architektura na obrázku (Obrázek 16) se podobá Full Frame CCD snímačům a má rozdělený paralelní posuvný registr na dvě přibližně stejné oblasti nazývané obrazovým a skladovacím polem. Obrazové pole v horní části paralelního posuvného registru je citlivé na dopadající světlo. Spodní skladovací pole znázorněné šedou barvou zakrývá stínítko, které brání dopadání světla na zbylou část paralelního posuvného registru. Tímto rozdělením paralelního posuvného registru se urychlí snímání obrazu.(12)

Náboje vytvořené na fotodiodách dopadajícími fotony v obrazovém poli po uplynutí expoziční doby se rychlým paralelním přenosem přemístí do skladovacího pole. Přemístěním nábojů se uvolnilo obrazové pole pro další snímání obrazu reprezentovaný náboji. Ve skladovacím poli se náboje předávají do sériového posuvného registru od nejbližšího řádku paralelního posuvného registru stejně jako ve Full Frame snímačích. V sériovém posuvném registru se náboje přemisťují za sebou k výstupnímu zesilovači totožně s Full Frame architekturou.(12)

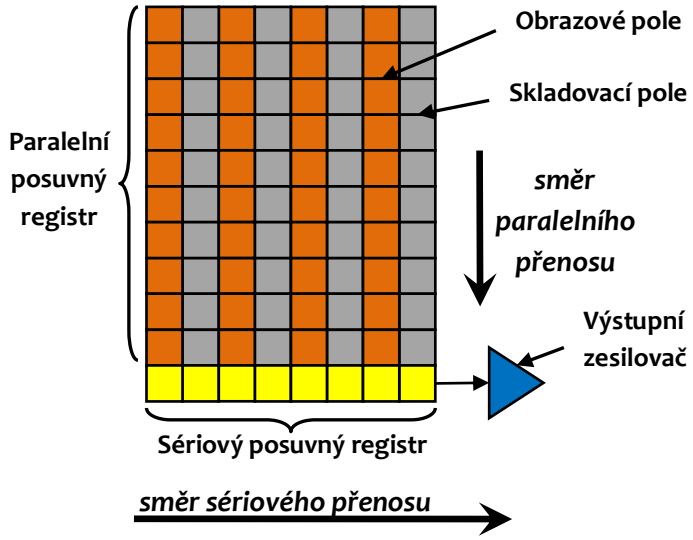


Obrázek 16 – Frame-Tansfer architektura

Poslední vyráběným typem CCD snímačů je *Interline Transfer architektura*, která kompenzuje nevýhody Frame Transfer architektury popsané v předešlém odstavci. Tato zařízení se skládají z hybridního uspořádání obrazového a skladovacího pole, kde každá fotodioda v obrazovém poli má paralelně přidruženou oblast ve skladovacím poli. Obě oblast znázorněné na obrázku (Obrázek 17) se rozprostírají od sériového posuvného registru po okraj paralelního posuvného registru a střídají se po celé ploše snímače. Skladovací oblast, jako u Frame Transfer snímačů, je stejně řešená zastíněním fotodiod přiléhajícím stínítkem.(13)

V jednotlivých obrazových polích se vytvoří náboj stejně jako u předešlých typů snímačů. Po skončení expozice nastává přemístění nábojů do sousedícího skladovacího pole. Náboje se naráz z obrazového pole paralelním přenosem přemístí na stíněné fotodiody. Prázdná obrazová pole je znovu možné použít pro získání nové obrazové informace.(13)

Stejně jako u Full Frame, Frame Transfer i Interline Transfer architektura využívá paralelního přenosu náboje na sériový posuvný registr. Jednotlivé řádky reprezentovány sladovacími poli se postupně posouvají na sériový posuvný registr, kde se jednotlivé náboje za sebou posouvají k výstupnímu zesilovači. Jednotlivé procesy se opakují až do doby, než se náboje ze skladovacích polí paralelního posuvného registru dostanou na výstupní zesilovač.(13)



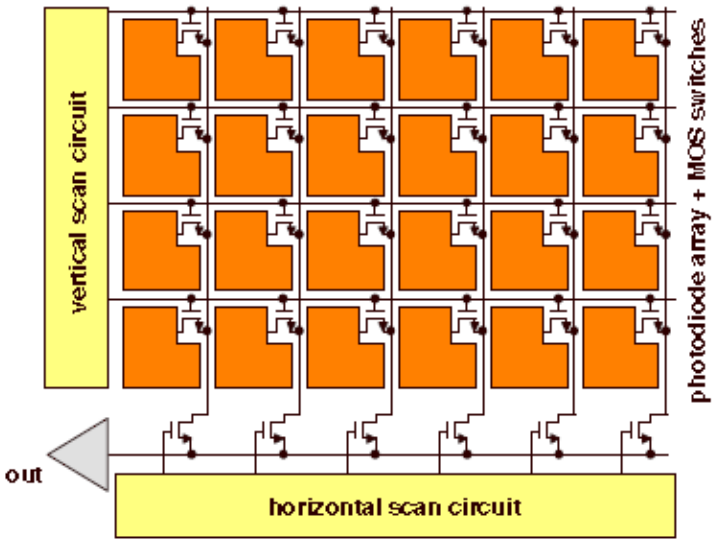
Obrázek 17 – Interline-Transfer architerkura

2.2 CMOS snímače

Snímače založené na technologii CMOS byly vyvíjeny v 70. letech 20. století, kdy byly využívaný ve snímací technice především CCD snímače a nástup vývoje snímačů pro běžné použití nastal až v 90. letech 20. Století (9). Technologie CMOS (Complementary Metal Oxide Semiconductor), využívaná nejen pro snímače, je hojně používána i u dalších polovodičových součástek (např. integrované obvody, unipolární tranzistory, atd.).(8)

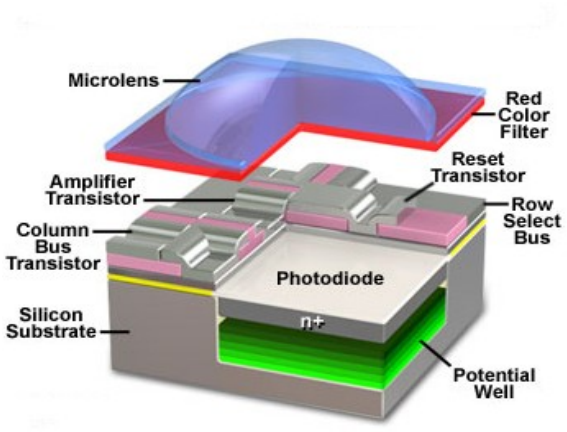
2.2.1 Struktura CMOS snímačů

Snímací pole CMOS snímačů je tvořeno mnoha obrazovými boby uspořádaných do pravoúhlé mřížky, jak je vidět na obrázku (Obrázek 18). Důležitou částí, obrazového bodu, je fotodioda vytvářející náboj, jehož velikost je ovlivňována dopadajícími fotony, stejně jako u snímačů CCD je často považována za pixel. Nejčastěji používané konstrukce obrazových bodů je spojením fotodiody a výstupního zesilovače do jednoho obrazového bodu. Tato konstrukce, znázorněna na obrázku (Obrázek 19), je označována za aktivní pixel snímače (APS).(8)



Obrázek 18 – Uspořádání dvourozměrného pole CMOS snímačů (14)

Malou část obrazového bodu z obrázku (Obrázek 19) zabírá fotodioda, přibližně 30 % plochy. Zbylá oblast je osazena podpůrnými tranzistory, řádkovými a sloupcovými vodiči využívané při odečtu náboje z fotodiody. Celý snímač je vyroben na křemíkové destičce, znázorněn v obrázku (Obrázek 19) jako Silicon Substrate. Na tomto nosném materiálu jsou vyrobeny tři tranzistory, které jsou součástí každé APS. První tranzistor je zesilovač (Amplifier Transistor), druhý tranzistor plní funkci resetu (Reset Tranzistor), třetím je výběrový tranzistor, znázorněný na výběrové sběrnici řádku pojmenovanou v obrázku (Obrázek 19) Row Select Bus. Posledním důležitým tranzistorem v CMOS snímačích je sloupcový tranzistor (Column Bus Transistor), spojující obrazové body jednotlivých řádků pod sebou do jednoho sloupce a spojuje ho s dalšími obvody pro zpracování obrazové informace. Součástí APS je i miniaturní objektiv známý jako mikročočka, pod kterou se nachází barevný filtr červené, modré nebo zelené barvy. Mikročočka (Microlens), označená na obrázku (Obrázek 19), má soustředit dopadající fotony na fotodiodu, pod kterou se nachází potenciálová oblast (Potential Well), ve které se soustředí vzniklý náboj.(8)



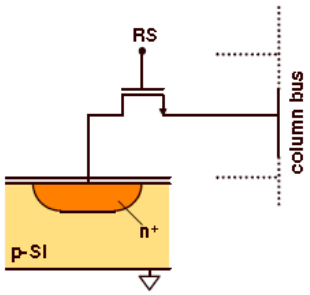
Obrázek 19 – Výřez jednoho APS obrazového bodu (8)

2.2.2 Architektura a princip CMOS snímačů

CMOS snímače používají dva typy provedení architektury a to:

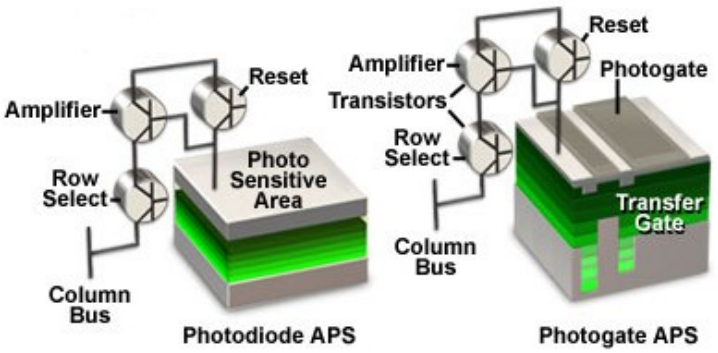
- pasivní,
- aktivní.

Obě provedení zachovávají totožné uspořádání na obrázku (Obrázek 18) lišící se pouze provedení obrazového bodu. Jednodušší variantou snímačů je pasivní architektura, na obrázku (Obrázek 20), využívající kombinaci zapojení fotodiody a adresovacího tranzistoru pracujícího v režimu spínače. Tato koncepce pracuje v několika krocích, kdy na začátku expoziční doby je vystavena fotodioda vysokému napětí (např. hodnoty v rozmezí 3,3V až 5V na sloupcové sběrnici), které jí orientuje do závěrného směru. Po celou expoziční dobu dopadající fotony v potenciálové oblasti obrazového bodu vytváří náboj a zmenšují ochuzenou oblast přechodu způsobenou závěrným napětím. Po skončení expozice, na sloupcové sběrnici, klesne napětí sepnutím tranzistoru řídící sloupec a náboj vytvoří pomocí výběrového tranzistoru řádku RS napětí na sloupcové sběrnici, kde je zpracováno následujícími obvody. Posledním krokem je resetování fotodiody pro další opakující se cyklus sběru fotonů. Takto se postupuje i u zbylých obrazových bodů, dokud se nevytvoří výsledný obraz.(8),(14)



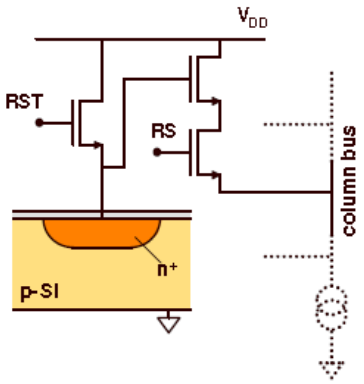
Obrázek 20 – Pasivní architektura CMOS snímačů (14)

Nejpoužívanější architekturou je aktivní, využívá dvou koncepcí fotocitlivých prvků, kterými jsou fotodiody a photogates neboli fotohradla. Výhodou použití fotodiody v obrazovém bodu je dobrá citlivost na viditelné světlo a to především v oblasti modrého spektra. Fotohradla v této oblasti spektra jsou horší, mají menší kvantovou účinnost, ale jejich výhodou je dosažení většího náboje, větší snímací plochy a použití dvojího vzorkování pro korelaci šumu. Uvedené fotocitlivé prvky jsou zobrazeny na obrázku (Obrázek 21).(8)



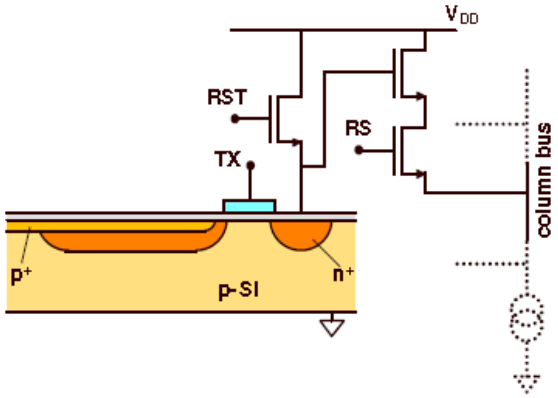
Obrázek 21 – Architektura APS fotodiody (vlevo) a APS fotohradla (vpravo) (8)

Pro odečtení náboje v CMOS snímačích, s fotodiodami, se používá několik kroků. Prvním krokem k zachycení obrazu je resetování fotodiody za účelem odstranění přebytečného náboje pomocí tranzistoru RST znázorněného na obrázku (Obrázek 22). Zároveň nastavuje závěrné napětí, stejně jako u pasívní architektury. Světlo dopadající na fotodiodu po expoziční dobu vytváří náboj, který se akumuluje v potenciálové oblasti pod fotocitlivou plochou vyobrazenou obrázkem (Obrázek 21) a zmenšuje závěrné napětí způsobem popsaným u pasivní architektury. Skončením expozice se sepne výběrový tranzistor na řádkové sběrnici označen RS v obrázku (Obrázek 22) a připojí ke sloupcové sběrnici zesilovač reprezentován zbylým tranzistorem. Tranzistor RST je rozepnut a náboj pomocí zesilovače, vytvoří napětí na sloupcové sběrnici, která je připojena pomocí sloupcového tranzistoru k výstupnímu snímači. Napětí na sloupcové sběrnici je závislé na zesilovači tvořený unipolárním tranzistorem, který zesílí napětí na úroveň odpovídající příslušnému řídícímu napětí tvořený velikosti nahromaděného náboje na fotodiodě. Zde dostáváme část výsledného obrazu. Postup se opakuje i u zbylých obrazových bodů v celém snímači.(8),(14)



Obrázek 22 – Koncepce APS s fotodiodou (14)

Fotohradlo se příliš neliší od koncepce s fotodiodou. Skládá se z již popsaného konceptu s fotodiodou a připojením další fotodiody přes hradlo TX znázorněné na obrázku (Obrázek 23). Tato konstrukce, aktivního snímacího bodu, umožňuje dvojí vzorkování. Nejprve se na fotodiodě oddělené hradlem TX pomocí dopadajících fotonů vytvoří náboj za expoziční dobu. Ke konci expoziční doby nastává první vzorkování, kdy je z fotodiody, připojené k tranzistorům, zjištěno napětí vytvořené po resetu. Napětí na sloupcové sběrnici se objeví stejným postupem uvedeným v předešlém odstavci. Po odebrání prvního vzorku se přes aktivované hradlo TX přemístí náboj z jedné fotodiody do potenciální oblasti druhé fotodiody řízené tranzistory. Totožným způsobem podle koncepce APS s fotodiodou je zjištěné napětí použito pro druhý vzorek. Odečtením prvního vzorku od druhého je možno za použití korelace odstranit velkou část šumu, který se projevuje zhoršenou kvalitou výsledného obrazu.(8),(13)



Obrázek 23 – Koncepce APS s fotohradlem (13)

2.3 Srovnání obrazových snímačů

V tabulce (Tabulka 4) jsou vypsané společné srovnávací parametry pro obě popisované technologie obrazových snímačů. Vyplývá z ní hlavní přednosti uvedených typů snímačů.

Tabulka 4 – porovnání CCD a CMOS snímačů

Snímače	CCD	CMOS
Výroba	Složitá	Jednoduchá
Spotřeba	Vysoká	Nízká
Ovládání snímače	Složité	Jednoduché
Fill faktor	Vysoký	Nízký až střední
Dynamický rozsah	Vysoký	Nižší
Digitální šum	Malý	Vysoký
Použití výřezu (windowing)	Ne	Ano

V tabulce (Tabulka 4) není uvedený parametr tíkající se kvality obrazu. Tento parametr CCD snímače mají vysoký díky řešení pomocí MOS kondenzátorů. U CMOS snímačů řešené popsanou aktivní architekturou je kvalita obrazu nižší, ale tento problém kvality obrazu řeší technologie Back Illuminated (9). Výhodou této technologie je vystavení světlu

zadní stranu CMOS snímače, kde celou plochu tvoří fotodioda bez stínicích překážek v podobě kovových vodičů a tranzistorů.(14)

Fill faktor vyjadřuje poměr plochy fotodiody a celkové plochy pixelu, využitou během detekce fotonů. Například u Full Frame snímačů je fill faktor roven 100%.

Parametr *windowing* popisuje schopnost snímače snímat jenom část obrazu. Výřez u CMOS snímačů je dosažen pomocí řádkových a sloupcových sběrnic. Jednoduchým výběrem kteréhokoliv řádku a příslušného sloupce lze i zpracovat jeden bod z celého obrazu, na rozdíl od CCD snímačů. V CCD snímačích se vždy zpracovává celý obraz.

Dynamický rozsah vyjadřuje maximální změnu intenzity signálu, který lze určit senzorem.

Následující tabulka (Tabulka 5) srovnává jednotlivé architektury CCD snímačů, dle vybraných kritérií včetně jejich dodatečných odůvodnění pod uvedenou tabulkou.

Tabulka 5 – Porovnání architektur CCD snímačů

	Full Frame	**Frame Transfer**	**Interline Transfer**
Výroba	Jednoduchá	Složitá	složitá
Nutnost clony	Ano	Ne	Ne
Snímkovací frekvence	Omezeno rychlostí clony	Vyšší než full frame	Vyšší, díky elektronické kontrole expoziční doby
Vady obrazu	Žádné	Stěr	Dosvit a osvit

Stěr vzniká u Frame Transfer snímačů při přenosu náboje z obrazové části pole snímače do skladovacího pole. Projevuje se šmouhou ve snímku v době paralelního přenosu při absenci clony, kdy neustále dopadají fotony na fotodiody. Tímto způsobem dochází i k *dosvitu* v Interline Transfer architektuře. *Osvit* je druhým nežádoucím procesem, který vniká částečným osvícením fotodiod pod stínítkem ve skladovací oblasti u Interline Transfer snímačů.

Podobné srovnání CMOS architektur jako v případě CCD snímačů nemá přínosný význam. Jedinou výhodou pasivní CMOS snímače je velká snímací plocha tvořená fotodiodami a nevýhodou příliš velký šum. Proto se stala nejpoužívanější aktivní architektura založena na provedení s fotohradlem, které má následující výhody:

- dvojí vzorkování, které je hlavní výhodou a umožňuje provedení korelace těchto vzorků k eliminaci šumu,
- korelace dvojího vzorkování má pozitivní vliv i na zbytkový off-set ,
- větší dynamický rozsah než samotná fotodioda,
- větší světelná citlivost.

3 Zprovoznění embedded zařízeni s kamerou

Tato práce používá pro snímání kameru C328, od společnosti COMedia Ltd., která přenáší data po sériové lince na embedded zařízení. Kromě kamery C328 se zařízením komunikuje ještě počítač. Pří výběru embedded zařízení bylo zapotřebí brát ohled na možnosti komunikace a zejména na dostatečnou velikost paměti zařízení. Velikost paměti významným způsobem ovlivní rychlost zpracování dat a zjednoduší se práce se zařízením. Vhodným zařízením pro tyto požadavky je použití kitu STM32F4Discovery vyrobené společností ST Microelectronics. Kit je osazen mikroprocesorem ARM Cortex-M4 STM32F407VGT6 podporující 32 bitovou instrukční sadu a běžící na maximálním kmitočtu 168 MHz.

Vybrané parametry mikroprocesoru STM32F407VGT6 (16):

- Jádro: ARM 32bit Cortex-M4,
- Pracovní kmitočet maximálně do 168 MHz.
- Napájení od 1,8 V do 3,6 V,
- 1 MB Flash paměť,
- 192 Kb SRAM,
- 64 Kb data RAM z CCM (core coupled memory),
- Paralelní rozhraní pro LCD,
- Paralelní rozhraní pro kameru,
- Interní 32 kHz kalibrovaný RC oscilátor,
- 3 × 12 bitový A/D převodník a 2 × 12 bitový D/A převodník,
- 3 × I^2C rozhraní,
- 4 × USART, 2 × UART,
- 3 × SPI,
- 2 × USB 2.0.

Parametry kitu STM32F4Discovery (17):

- Mikroprocesor STM32F407VGT6,
- Součásti desky je ST-Link s výběrem režimu jako samotný ST-Link (s SWD konektorem pro ladění a programování),
- Napájení pro externí zařízení 3 V a 5 V,
- Dvě tlačítka (uživatelské a reset),
- Čtyři uživatelské LED diody (oranžová, zelena, červená a modrá),
- USB s Mirko AB konektorem.

Obrázek 24 – Kit STM32F4Discovery (17)

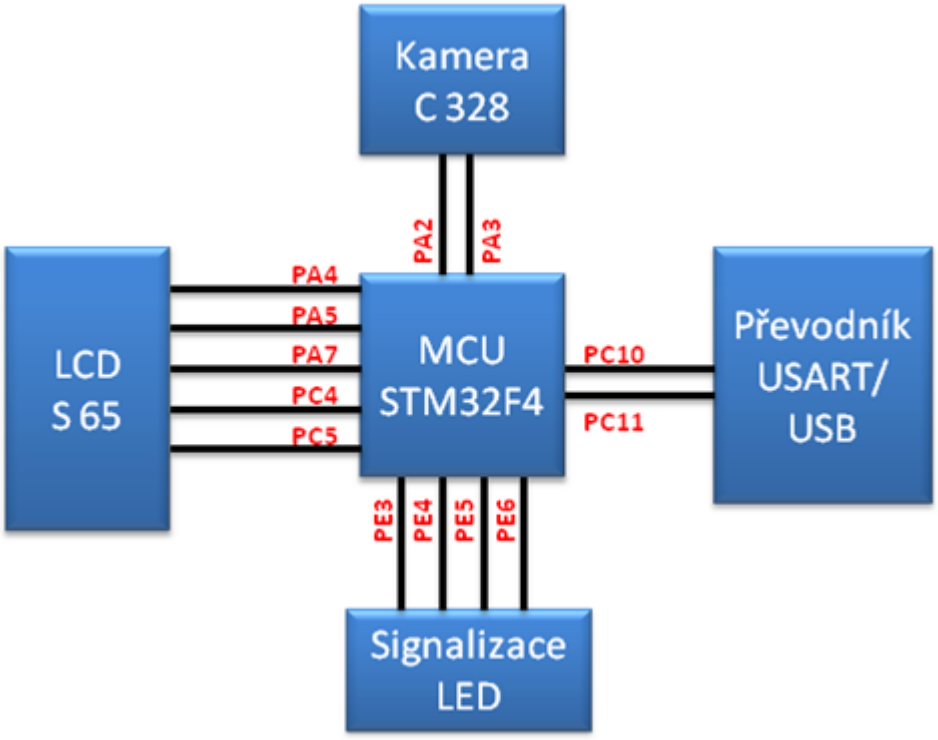
3.1 Hardwarové řešení

S použitím kitu STM32F4Discovery na obrázku (Obrázek 24) se realizace obvodu stává velice jednoduchou. Spočívá v propojení jednotlivých komponentových zařízení pomocí propojovacích kablíků. Připojovanými komponentami ke kitu STM32F4Discovery jsou:

- Kamera C328
- LCD displej S65
- USB převodník
- Panel LED diod (tvořen řadou LED diod)

Zapojení popsaných zařízení vyobrazuje blokové schéma (Obrázek 25), na kterém je panel LED diod reprezentován blokem signalizace LED.

3.1.1 Připojení zařízení k MCU



Obrázek 25 – Blokové schéma zapojení MCU STM32F4

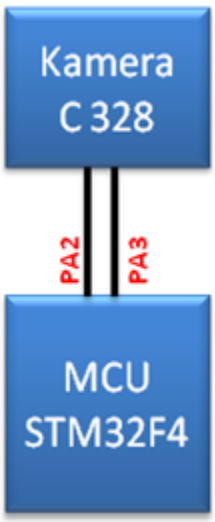
Blokové schéma (Obrázek 25) znázorňuje realizaci embedded systému s použitím mikroprocesoru STM32F407VGT6 na kitu STM32F4Discovery. V blokovém schématu jsou vyznačeny použité vývody při realizaci, jejíž pracovní funkci popisuje následující tabulka (Tabulka 6). Z té vyplívají konkrétní komunikační rozhraní použitých v tomto řešení. Jednotky USART spojují s mikroprocesorem kameru C328 a převodník USB ke komunikaci s počítačem. Mikroprocesor komunikuje s displejem prostřednictvím SPI rozhraní. K signalizaci LED postačuje základní nastavení vývodů označenou I/O, jako vstupně výstupní. Vývody I/O mikroprocesoru nastavují pouze hodnoty na svém výstupu do hodnot odpovídající logickým úrovním 1 nebo 0 nebo přijímají logické hodnoty.

V příloze E je uvedený snímek celého zapojení.

Tabulka 6 - Funkce použitých pinů MCU

Pin MCU	Funkce pinu	Pin MCU	Funkce pinu
PA2	USART2 TX	PC10	USART3 TX
PA3	USART2 RX	PC11	USART3 RX
PA4	SPI1 NSS	PE3	I/O
PA5	SPI1 SCK	PE4	I/O
PA7	SPI1 MOSI	PE5	I/O
PC4	I/O	PE6	I/O
PC5	I/O		

3.1.2 Připojení kamery C328 k MCU

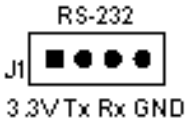


Obrázek 26 – Blokové schéma zapojení kamery C328

Obrázek (Obrázek 26) znázorňuje zapojení kamery s mikroprocesorem pomocí USART. V tabulce (Tabulka 7) jsou přesně definované vývody k přesnému propojení mezi oběma zařízeními. Sériová linka tvořena vývody TX, RX na kameře a piny PA2, PA3 ze strany mikroprocesoru, reprezentují dva spojené páry TX (vysílač) a RX (přijímač). Rozložení konektoru na kameře C328 ukazuje obrázek (Obrázek 27), kde piny s označením 3,3 V a GND slouží pro napájení kamery. Barevné označení pro napájecí kablíky, které jsou součásti kamery, mají standardní červenou barvu pro kladné napájení a černou uzemnění Zbylé vývody konektoru používá komunikační sériová linka s rozdílným barevným značením.

Tabulka 7 – Zapojení pinů kamery C328 k MCU

Pin kamery	Pin MCU
VCC	3V
TX	PA3
RX	PA2
GND	GND

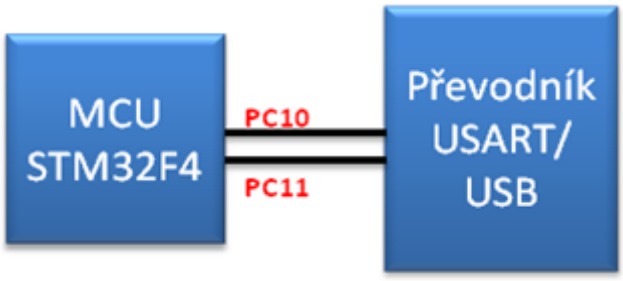


Obrázek 27 – Zapojení konektoru na kameře C328 (18)

3.1.3 Připojení USB převodníku k MCU

Komunikace mezi počítačem a mikroprocesorem nemá přímou kompatibilitu mezi oběma zařízeními prostřednictvím sériové linky. Tento problém řeší použití převodníku USART/USB, které zaručí požadovanou kompatibilitu komunikace mezi zařízeními. Převodníkem je zde použit integrovaný obvod PL2303 vyráběný firmou Prolific. Použití tohoto převodníku je dáno kamerou C328, jejíž součásti byl i tento převodník ke komunikaci s počítačem. Správné fungováni převodníku USART/USB zajistí na počítači instalace ovladače pro převodník. Tento krok zajistí správné fungování převodníku při zasunutí do USB portu ze strany počítače a vytvoří prostřednictvím sériové linky komunikační cestu obou zařízení. Funkčnost vytvořené sérové linky na počítači se projeví označením např. COM 5 ve Správci zařízení sekce porty.

Není nutností používat převodníku PL2303, a lze využít i alternativy k tomuto integrovanému obvodu. Dostupný a hojně využívaný převodník USART/USB pro tyto účely je i obvod FT232RL od společností FTDI Chip. Zde je taktéž nutností instalace ovladače pro správné fungování.



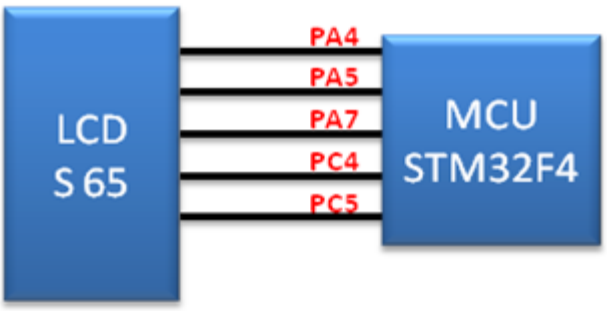
Obrázek 28 – Blokové schéma zapojení převodníku

Zapojení převodníku USART/USB s mikroprocesorem demonstruje blokové schéma (Obrázek 28) a přesný popis zapojených vývodů ukazuje tabulka (Tabulka 8). Vývody s označením 5 V a 3 V na převodníku nejsou zapojeny, to z důvodu nepotřeby napájet kit s MCU. STM32F4Discovery už v samotné konstrukci má zabudované napájení externích zařízení. Piny PC10, PC11 podle tabulky (Tabulka 6) tvoří USART jednotku MCU a tvoří spojení s vývody TX, RX na převodníku dva páry TX, RX jako na předešlé straně.

Tabulka 8 – Zapojení pinů převodníku USART/USB k MCU

Pin převodník	Pin MCU
5V	Nezapojeno
3V	Nezapojeno
TX	PC11
RX	PC10
GND	GND

3.1.4 Připojení LCD displeje k MCU



Obrázek 29 – Blokové schéma zapojení LCD S 65

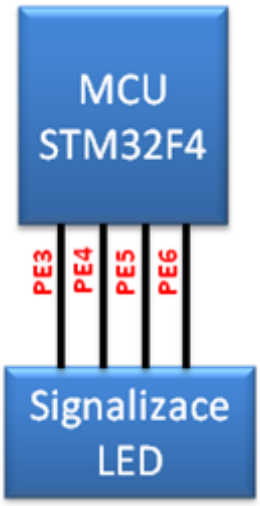
Připojení displeje k mikroprocesoru znázorňuje obrázek blokového schématu (Obrázek 29). Jedná se o displej vyrobený společností Sharp a podrobný popis o tomto displeji je v diplomové práci „Stabilizace polohy létajícího objektu pomocí inerciálních senzorů“ (19). Označení S 65 je podle mobilního telefonu, ve kterém se používal.

Přesné zapojení s mikroprocesorem je v tabulce (Tabulka 9), kde jsou tři vývody nezapojené. První vývod s označením 1V8 se nezapojují vůbec v tomto zapojení embedded zařízení. Zbylé dva nezapojené piny slouží k posvícení displeje S 65 a připojují se ke vlastnímu zdroji napětí.

Tabulka 9 – Zapojení vývodů LCD displeje k MCU

Pin LCD	Pin MCU	Pin LCD	Pin MCU
2V9	3V	CLK	PA5
1V8	Nezapojeno	CS	PA4
RS	PC4	LED+	Nezapojeno
RST	PC5	LED GND	Nezapojeno
DAT	PA7	GND	GND

3.1.5 Zapojení signalizačních LED diod k MCU



Obrázek 30 – Blokové schéma zapojení LED diod

Nejjednodušší připojenou částí tohoto embedded systému znázorňuje blokové schéma na obrázku (Obrázek 30). Blok Signalizace LED je tvořen malou deskou plošných spojů osazenou v řadě několika LED diodami. Funkcí LED diod je vizuální kontrola jednotlivých kroků a detekování nastalých problémů. Tabulka (Tabulka 10) popisuje přesné zapojení led diod na patřičné vývody MCU jejíž význam popisuje předešlá kapitola (Připojení zařízení k MCU).

Tabulka 10 – Zapojení LED diod k MCU

Led dioda	Pin MCU
LED 1	PE3
LED 2	PE4
LED 3	PE5
LED 4	PE6
ZEM	GND

3.2 Softwarové řešení

Řešení se skládá z několika knihoven vytvořených ve vývojovém prostředí Atollic TrueStudio od firmy Atollic. Atollic TrueStudio je konkrétně navrženo pro vývoj embedded systému založených na architektůře mikroprocesorů ARM a programovacím jazyce C a C++.

Program pro zprovoznění kamery C328 obsahuje následující knihovny:

- main.c,
- stm32f4xx_conf.h,
- stm32f4xx_it.c, stm32f4xx_it.h,
- system_stm32f4xx.c,
- tiny_printf.c,
- main.h,
- led.c, led.h,
- usart.c, usart.h,
- kamera.c, kamera.h,
- složka lcd.

První pět odrážek se implicitně vytvářejí při založení projektu a zbylé odrážky tvoří nutnou část pro zprovoznění kitu STM32F4Discovery připojenými periferiemi včetně algoritmů zpracování obrazu. Složka *lcd* tvoří soubor knihoven pro zprovoznění LCD displeje S 65 využité a popsané v diplomové práci (19).

Rozděleni programu podle významu jednotlivých knihoven je následující částech:

- hlavní část programu,
- komunikační část programu,
- zpracování a zobrazení obrazu.

3.2.1 Hlavní část programu

Samotný program začíná ve funkci main deklarací proměnné *operace*. Proměnná *operace* slouží pro výběru algoritmu pro úpravu obrázku. Následně probíhá inicializace následujících částí mikroprocesoru:

- Nastavení delay,
- Inicializace portů pro LED,
- Inicializace portů USART,
- Nastavení vývodů pro LCD,
- Inicializace samotného kitu STM32F4Discovery.

Přesně zapsaný zdrojový kód je v přiložené příloze B.

Po tomto nastavení programu pokračuje nekonečná smyčka, ve které se nachází celý program řídící ovládání periferií a výpočetní operace. V samotném úvodu nekonečné smyčky jsou dvě podmínky reagující na příkazy obdržené od počítače. První podmínka reaguje na příkaz s označením A, který slouží pro synchronizaci a následné hrubé nastavení kamery. Druhá podmínka kontroluje příkaz B pro získání snímku z kamery.

Navazující částí programu na předešlé řádky tvoří podmínka s indikací *ackRX2*. Proměnná informuje o příchozím potvrzovacím příkazu z kamery C328. V těle této podmínky se nachází příkaz *switch* s těmito možnostmi:

- INIT,
- PACKET,
- TAKÉ,
- GET.

Hlavním úkolem vypsaných příkazů je nastavení dalšího kroku pro nastavování volání parametrů a pořízení snímků z kamery. Příklad zdrojového kódu je v příloze B. Program pokračuje podmínkou na *nckRX2*, která informuje o nastalé chybě neprovedení požadovaného příkazu ze strany kamery.

Poslední části kódu nacházející se v nekonečné smyčce je hlavním jádro programu tvořící příkaz *switch* s následujícími možnostmi:

- INIT,
- PACKET,
- TAKE,
- GET,
- ZPRACOVAT,
- OK,
- NE.

Uvedené možnosti v odrážkách obsahují příkazy pro nastavení kamery C328 (první čtyři) a zpracování obrazu (ZPRACOVAT). Zbylé možnosti obstarávají zpětnou vazbu aplikaci v počítači (kladnou nebo zápornou). Kompletní ukázku celé funkce *main* obsahuje již zmíněná příloha B.

3.2.2 Komunikační část programu

Tuto část programu tvoří knihovny *usart.c, stm32f4xx_it.c, kamera.c* a přidružené hlavičkové soubory. Jednotlivé knihovny tvoří důležité příkazy a inicializace portů určených ke komunikaci.

Knihovna s označením usart je důležitou součástí tohoto programu a zprostředkovává nastavení jednotlivých USART jednotek mikroprocesoru STM32F407VGT6. Soubor příkazů obsažených v knihovně pro nastavené USART jsou uvedeny v odrážkách.

- `void USART3_Inicializace(void);`
- `void USART2_Inicializace(void);`

Pod označenou knihovnou *stm32f4xx_it* se skrývá přerušení mikroprocesoru, které umožňuje pozastavit aktuálně prováděnou operaci v hlavní části programu a provést instrukce v přerušení. Zajišťuje samostatné přijímání dat bez nutnosti zápisu funkce pro příjem dat. V odrážkách jsou uvedené funkce pro přerušení obou USART jednotek. Krom těchto knihovna obsahu ještě funkce vygenerované při založení projektu.

- `void USART3_IRQHandler(void);`
- `void USART2_IRQHandler(void);`

Obě funkce mají za úkol ukládat veškeré data do tzv. bufferu, přesně řečeno polí o určité velikosti. Dalšími prvky knihovny jsou proměnné indikující stav komunikace s kamerou a pole pro data.

- ackRX2 – pozitivní odpověď na příkaz
- nckRX2 – negativní odpověď na příkaz
- dataRX2 – pro příjem dat
- syncRX2 – indikuje příkaz sync
- delkaDat – velikost dat obdržená od kamery
- poleDat – slouží pro příjem

Poslední knihovnou pro komunikaci nese označení *kamera*. Obsahuje příkazy pro nastavení kamery C328 v následujících tabulkách (Tabulka 11, Tabulka 12, Tabulka 13, Tabulka 14, Tabulka 15, Tabulka 16) s opsanými parametry.

Tabulka 11 – Příkaz Initial

Byte	1	2	3	4	5	6
Hodnota	AAh	01h	00h	ct	raw	jpeg

Hodnota na pozici *ct* určuje barvou hloubku snímaní kamery s má následujícím výběrem:

- 01h – šedý RAW ve 2 bitech,
- 02h – šedý RAW ve 4 bitech,
- 03h – šedý RAW v 8 bitech,
- 05h – barevný RAW v 12 bitech,
- 06h – barevný RAW v 16 bitech,
- 07h – JPEG.

Do pozice *raw* se dosazují tyto hodnoty:

- 01h – představuje velikost snímku 80 × 60 pixelů,
- 03h – představuje velikost snímku 160 × 120 pixelů.

Posledním parametrem je *jpeg* a nabízí tyto hodnoty:

- 01h – velikost JPEG snímku 80 × 64 pixelů,
- 03h – velikost JPEG snímku 160 × 128 pixelů,
- 05h – velikost JPEG snímku 320 × 240 pixelů,
- 07h – velikost JPEG snímku 640 × 480 pixelů.

Tabulka 12 – Příkaz Get Picture

Byte	1	2	3	4	5	6
Hodnota	AAh	04h	pt	00h	00h	00h

Hodnota *pt* reprezentuje tyto parametry:

- 01h – pořízení snímku,
- 02h - nepřetržité snímání,
- 05h – nepřetržité snímání JPEG.

Tabulka 13 – Příkaz Snapshot

Byte	1	2	3	4	5	6
Hodnota	AAh	05h	st	sf	sf	00h

Hodnota *st* nastavuje tyto parametry:

- 00h – komprimovaný obrázek,
- 01h – nekomprimovaný obrázek (RAW).

Parametr *sf* určuje počet vynechaných snímků před kompresí.

Tabulka 14 – Set Package Size

Byte	1	2	3	4	5	6
Hodnota	AAh	06h	08h	ps	ps	00h

Parametr *ps* zadává velikost posílaného paket s daty a je v Rozmezí od 64 bytu po 512 bytu.

Tabulka 15 – Příkaz Sync

Byte	1	2	3	4	5	6
Hodnota	AAh	0Dh	00h	00h	00h	00h

Tabulka 16 – Příkaz Ack

Byte	1	2	3	4	5	6
Hodnota	AAh	0Eh	cID	ACK c	p	p

Parametr *cID* vrací hodnoty odpovídající umístění na pozici druhého bytu v tabulce.

Parametr *p* vrací následující hodnoty:

- číslo přijatého paketu při posílání komprimovaných dat,
- hodnotu 00h pro ostatní příkazy,

Kompletní tabulku s příkazy pro kameru C328 je dostupná v (18) i s popisem příkazů.

3.2.3 Zpracování a zobrazení obrazu

Knihovna *kamera a složka lcd* tvoří velkou část programu s obsahem mnoha funkcí. Mezi důležité funkce z knihovny patří funkce pro *VypocetBarev*, která přepočítává pořízená data kamerou do podoby RGB. Dalšími funkcemi v knihovně *kamera* jsou algoritmy pro zpracování obrazu. Jednotlivé algoritmy a přepočet barev budou podrobněji popsány v další kapitole této práce.

Ze složky lcd je využita funkce pro zobrazení snímku pro displej. Jedná se o parametrickou funkci *lcd_fillRectagle*, která umožňuje vykreslit pixel v RGB složkách. Tato funkce je aplikovaná v dalších funkcích *vykreslitObrazek()* a *vykreslitOrazekZmena()*. Kompletní seznam příkazů pro tuto část programu uvádějí odrážky.

- `void vykreslitObrazek();`
- `void vykreslitObrazekZmena();`
- `void VypocetBarev();`
- `void Negativ();`
- `void Laplace();`

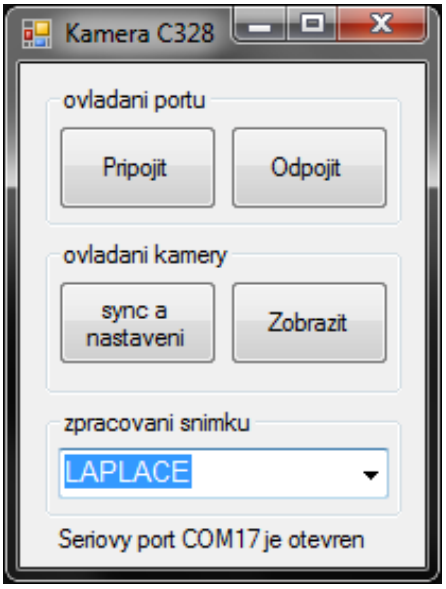
4 Zpracování a zobrazení kamerových dat

Správné zpracování obrazových dat z kamery C328 je důležitým krokem k věrohodnému zobrazování snímků. Této problematice je věnována následující část bakalářské práce.

4.1 Ovládací program

Ovládací program zobrazený na obrázku (Obrázek 31) v této práci řídí operace prováděné na embedded zařízení reprezentované kitem STM32F4Discovery. Jedná se o velice jednoduché řízení pomocí tří kroků. Pro komunikaci se zařízením slouží blok tlačítek s názvem *ovládání portu,* a jak je zřejmé z názvů i jejich účel (výsledkem je vybrání správného sériového portu ke komunikaci a následně odpojení). Panel *ovládání kamery* obsahuje tlačítka s příkazy pro řízení kamery. Tlačítko *sync a nastavení* posílá příkaz pro synchronizaci zařízení s kamerou a následné nastavení s parametry pořizovaného snímku. Druhé tlačítko zajišťuje pořízení snímku, zpracování a zobrazení na LCD displeji. Posledním krokem nastavení je možnost výběru úpravy obrázku a na výběr jsou následující možnosti:

- snímek RAW (nezpracovaný),
- snímek NEGATIV,
- snímek LAPLACE.

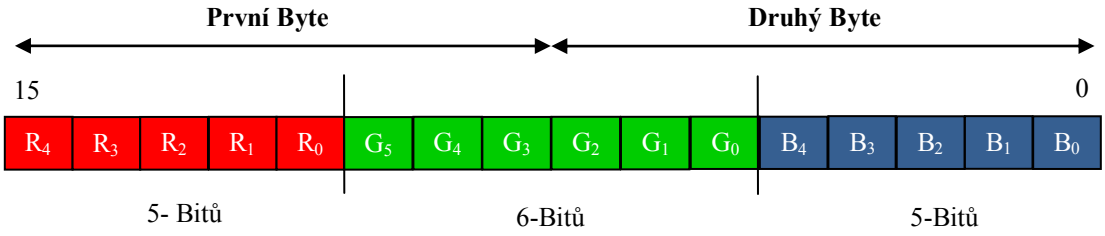


Obrázek 31 – Okno ovládacího programu

4.2 Přepočet barev z kamerových dat

Kamera C328 podporuje mnoho formátů k reprezentaci barevného vyjádření pořízeného snímku. Všechny formáty byly uvedeny v předešlé kapitole a konkrétně v příkazu *Initial* pro kameru. Přesný barevný formát reprezentován daty znázorňuje obrázek (Obrázek 32)

a zároveň ukazuje řazení barev formátu 16 bit RGB použitý při nastavení kamery. Pro získání jednotlivých barev podle obrázku (Obrázek 32) je nutné srovnávat data z kamery tak, jak postupně přicházely. Popsaný postup znázorňuje zdrojový kód v příloze C.



Obrázek 32 – Formát 16-Bit RGB

Po seřazení jednotlivých bytů do formátu na obrázku nastává separace jednotlivých barev. Pro získání barev se aplikují definované masky odpovídající přesnému rozložení barev v následujících odrážkách.

- F800h – maska pro červenou barvu
- 7E0h – maska pro zelenou barvu
- 1Fh – maska pro modrou barvu

Uvedené masky jsou zapsané v hexadecimálním tvaru. Aplikací masek se ještě nezískaly jednotlivé barvy a je nutné provést pravý bitový posun na velikost odpovídající jednoho bytu. Jednotlivé posuny barev demonstrují následující odrážky:

- Červená – posun o 11 bitů doprava,
- Zelená – posun o 5bitů doprava,
- Modrá – neposouvá se.

Získané barvy se ještě musejí upravit pro správné zobrazení na displeji. Zobrazením stávajících barev by vedlo k vzniku tmavého snímku (v lepším případě) nebo pouze černé barvy. K řešení tohoto problému je k dispozici několik variant. Jedna metoda aplikuje levý bitový posun do maximální velikosti jednoho bytu a jednotlivé barvy mají následující posuny:

- Červená – posun o 3 bity doleva,
- Zelená – posun o 2 bity doleva,
- Modrá – posun o 3 bity doleva.

Nebo se může aplikovat přepočet pro jednotlivé bary pomocí následujících výrazů.

- Červena barva = (hodnota červené * 255) / 31,
- Zelená barva = (hodnota zelené * 255) / 63,
- Modrá barva = (hodnota modré * 255) / 31.

Hodnoty červená, zelená, modrá odpovídají barvám po aplikování masek a následným pravým bitovým posunem. Přepočtem dostaneme barvy s bitovou hloubkou 8 bitů, který v současnosti reprezentuje standard pro intenzitu barev. Výsledek separace barev podle popsaného postupu znázorňuje následující obrázek (Obrázek 33).

Obrázek 33 – Reprodukce barevného snímku ve formátu RAW

4.3 Použité algoritmy zpracování obrazu

V této bakalářské práci jsou použité dvě metody pro zpracování obrazu. První aplikovaným algoritmem je zástupce jasové transformace, kterým je negativ. Druhou požitou metodou je Laplaceův operátor, který patří do skupiny metod zabývající se detekcí hran.

4.3.1 Negativ

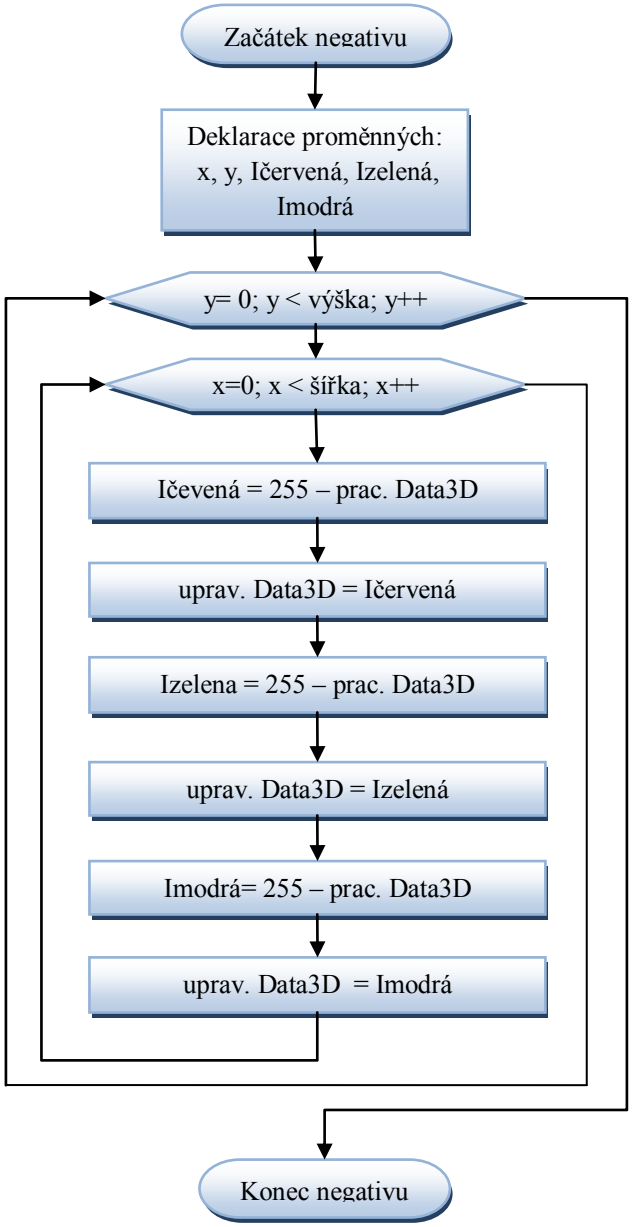
Negativ představuje nejjednodušší obrazovou transformaci a reprezentuje metodu z jasových transformací. Metoda negativu se dá použít na transformaci i bez přešlého procházení za účelem analýzy objektu. Metoda je dána předpisem (4.1) aplikovatelným v jakémkoliv bodě (5):

$$g = L - 1 - f \tag{4.1}$$

kde g je transformovaná funkce,
f je původní funkce,
L je počet úrovní.

Počtem úrovní se rozumí maximální hodnota, kterou nabývá jeden obrazový bod při definovaném počtu bitů. Tvoří-li jeden obrazový bod osm bitů, tak maximální hodnota je 2^8, tedy 256 úrovní.

Konkrétní řešení použité v této práci pomocí transformace negativem ukazuje vývojový diagram (Obrázek 34). Diagram ukazuje provedení nejjednodušší transformace na barevný negativ. Deklarací proměnných se objevují názvy *Ičervená, Izelená, Imodrá* a představují hodnoty intenzit obrazových bodů. Podle výrazu (4.1) se v diagramu (Obrázek 34) vypočítávají hodnoty intenzit barev z polí *prac. Data3D*. Pole *prac. Data3D* je tvořeno třemi dvojrozměrnými poli jednotlivých barev v pořadí červené, zelené a modré. Konkrétní řešení transformace negativem ukazuje zdrojový kód v příloze C.



Obrázek 34 - Vývojový diagram transformace negativ

Obrázek (Obrázek 35) ukazuje použití negativu pro snímek.

Obrázek 35 – Snímek upravená postupem negativu

4.3.2 Laplaceův operátor

Laplaceův operátor definován vztahem (4.2) pro dvojrozměrnou funkci $f(x, y)$ je derivací druhého řádu (2).

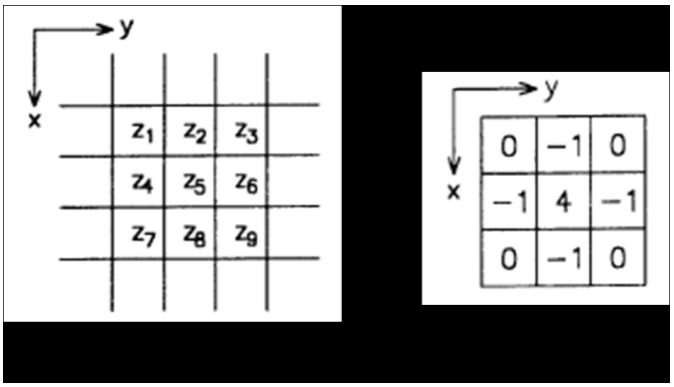
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{4.2}$$

kde $\nabla^2 f$ je druhá derivace gradientu funkce a rovná se Δ (Laplaceův operátor).

Laplaceův operátor pro oblasti 3 × 3 je nejčastěji vyjadřován v číslicové podobě vztahem (4.3) a výslednou masku znázorňuje obrázek (Obrázek 36b).(2)

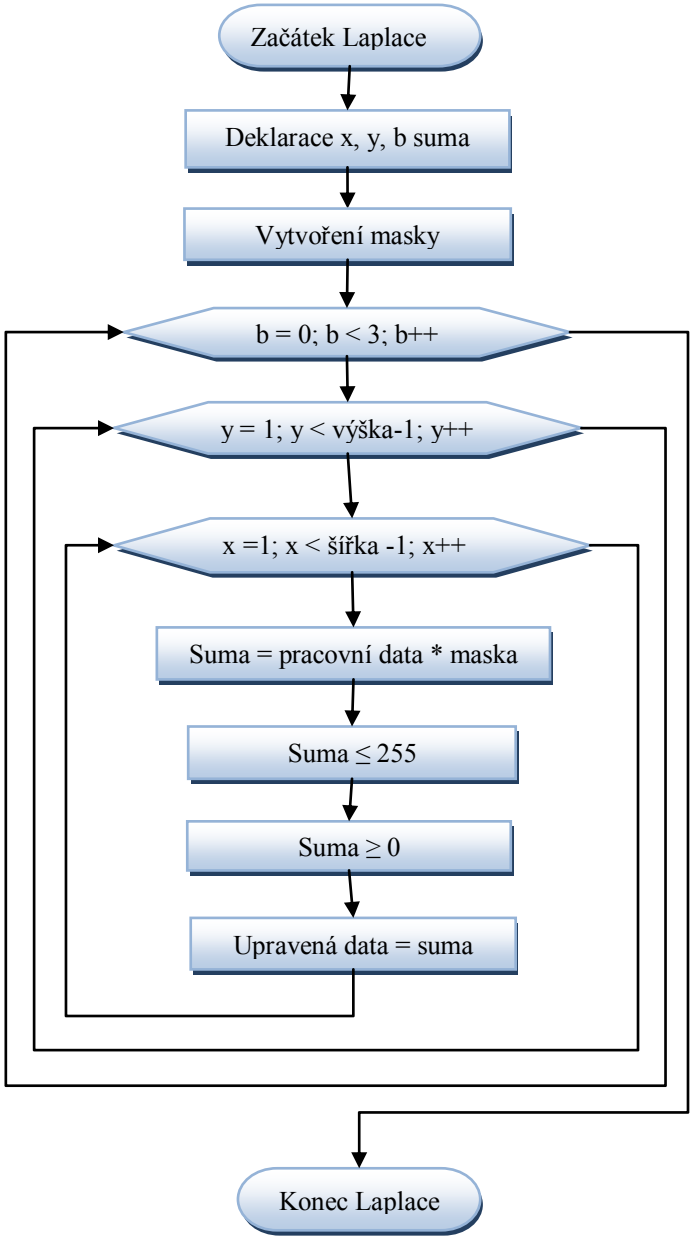
$$\nabla^2 f = 4 \cdot z_5 - (z_2 + z_4 + z_6 + z_8) \tag{4.3}$$

kde z_i jsou intenzity obrazových bodů a jejich uspořádání v matici 3 × 3 ukazuje obrázek (Obrázek 36a).



Obrázek 36 – a) jednotlivé uspořádání intenzit obrazových bodů, b) maska 3×3 pro Laplaceův operátor (2)

Programové řešení Laplaceova operátoru demonstruje vývojový diagram (Obrázek 37). Po deklaraci potřebných proměnných se vytváří maska podle obrázku (Obrázek 36b) reprezentována jednorozměrným polem. Pro každý obrazový bod v poli *Pracovní data* se aplikuje přesně maska podle obrázku (Obrázek 36b). Výsledek *Suma* je korigován do rozsahu 0 až 255 a následně uložen na odpovídající pozici v poli *Upravená data*. Zdrojový kód k Laplaceovu operátoru ukazuje příloha C.



Obrázek 37 – Vývojový diagram Laplaceův operátor

Obrázek (Obrázek 38) ukazuje použití negativu pro snímek.

Obrázek 38 – Snímek upravený Lalaceovým operátorem

Závěr

Cílem této bakalářské práce bylo zprovoznit kameru, v tomto případě model C328 společnosti COMedia, na embedded zařízení. Toto zařízení reprezentuje vývojový kit STM32F4Discovery společnosti ST Mikroelectronics, který je se svými technickými parametry vhodný pro použití ke kameře na zpracování obrazové informace. Práce s tímto kitem po hardwarové stránce byla jednoduchá a bezproblémová. Kromě embedded zařízení bylo nutné vymyslet i funkční program.

Při realizaci tohoto programu, se vyskytovala řada potíží při synchronizaci a zpracování dat. I přes tyto potíže se podařilo vyvinout funkční program pro embedded zařízení.

Časté potíže se synchronizací kamery způsoboval úsporný režim konstrukčního řešení kamery C328. I přes postupy uvedené v manuálu kamery bylo nutné časté odpojování kamery od embedded zařízení. Toto odpojování nastávalo až po několikanásobném opakování příkazu *sync*, což vedlo k restartování kamery. K odstranění tohoto problému stačilo prodloužení doby poslání potvrzovacího příkazu ACK na dostatečně dlouhou dobu pro nastavení obvodu v kameře. V ostatních příkazech používaných pro nastavování kamery nedocházelo k dalším potížím při dodržování parametrů podle manuálu. Při zpracování dat se vybíral z mnoha dostupných variant přesný formát RGB obrazové informace přijaté z kamery. Výsledek řešení byl zaznamenán v kapitole čtyři.

Výsledkem zpracování obrazových dat z kamery pro formát RAW představuje v závěru podkapitoly *Přepočet barev z kamerových dat* prořízený snímek (Obrázek 33). Kvalita uvedených snímků není příliš velká, což je dáno rozměry 80 x 60 pixelu a barevné vyjádření snímků, které se projevuje zabarvení do světlomodrých odstínů (dáno konstrukcí kamery). Při aplikování výpočtu negativu na snímek (Obrázek 35) je použité rozlišení dostačující a dá se tvrdit, že obraz odpovídá skutečnému barevnému negativu. Při použití Laplaceova operátoru na snímku (Obrázek 38) vyobrazený v závěru kapitoly *Laplaceův operátor* je zřejmá vlastnost, kterou je detekce hran s učením bodu ležícího na tmavé či světlé straně. Přednosti metody by byly vidět na snímcích v jasové úrovni. Zajímavým srovnání by bylo použití maximálního rozlišení kamery 160 x 120 pixelu pro použité metody zpracování obrazové informace.

Problematika zpracování obrazových informací by mohla být vyřešena použitím vhodnějších algoritmů určených pro zpracování obrazu. Jsou jimi např. deformace obrazu, různé způsoby vyhlazování nebo konvolučních metod pro ostření a zvětšování snímků. Toto by vedlo ke stávající úpravě programu. Další úpravy programu by spočívaly v možnostech jednotlivých nastavení kamery. Využití LCD displeje by nemuselo být pouze pro zobrazování snímků, ale zařízení by mohlo obsahovat i menu s parametry pro nastavení kamery, nebo úpravy stávajícího ovládacího programu. Současná verze ovládacího programu představuje defaultní nastavení kamery prostřednictvím dvou příkazů. Rozšířením o přesnější nastavení v možnostech zobrazení a ukládání snímků by byl ovládací program mnohem zajímavější než doposud.

Literatura

(1) KLÍMA, Miloš, M. BERNAS, HOZMAN a P. DVOŘÁK. *Zpracování obrazové informace*. 1. vyd. Praha: České vysoké učení technické, 1999, 177 s. ISBN 80-010-1436-3.

(2) FRIBERT, Miroslav. *Základy zpracování obrazu*. Vyd. 1. Pardubice: Univerzita Pardubice, 2006, 104 s. ISBN 80-719-4901-9.

(3) VEČERKA, Arnošt. *Komprese dat* [online]. Olomouc, 2008, 65 s. [cit. 2013-04-06]. Dostupné z: http://phoenix.inf.upol.cz/esf/ucebni/komprese.pdf

(4) ŠVEC, Petr. V čem je lepší formát RAW. *Amos Sofware* [online]. © 2009 [cit. 2013-06-25]. Dostupné z: http://www.amsoft.cz/Produkty/Adobe/lightroom/raw.html

(5) DOBEŠ, Michal. *Zpracování obrazu a algoritmy v C#*. 1. vyd. Praha: BEN - technická literatura, 2008, 143 s. ISBN 978-80-7300-233-6.

(6) KYNCL. Formát PNG - podívejme se na něj podrobněji. *NET-MAG.cz* [online]. 28. 9. 2002 [cit. 2014-05-21]. Dostupné z: http://web.net-mag.cz/?action=art&num=220

(7) W3C: Portable Network Graphics (PNG) Specifikace (Second Edition). BOUTELL, Thomas, Glenn RANDERS-PEHRSON a Mark ADLER. *W3C: Portable Network Graphics (PNG) Specifikace (Second Edition)* [online]. 10.11.2003 [cit. 2014-05-19]. Dostupné z: http://www.w3.org/TR/PNG/

(8) Introduction to CMOS Image Sensors. *Molecular Expressions: Optical Microscopy Primer* [online]. © 1998-2013 [cit. 2013-02-26]. Dostupné z: http://micro.magnet.fsu.edu/primer/digitalimaging/cmosimagesensors.html

(9) ŠURKALA, Milan. Fotomobily: snímací čipy CMOS vs. CCD. *Digimanie* [online]. 5.10.2009 [cit. 2013-03-5]. Dostupné z: http://www.digimanie.cz/art_doc-67BCCD2DF7A9F53EC125763F0044663D.html

(10) Introduction to Charge-Coupled Devices (CCDs). *Microscopy U: The source for microscopy education* [online]. © 2000-2013 [cit. 2013-02-7]. Dostupné z: http://www.microscopyu.com/articles/digitalimaging/ccdintro.html

(11) CCD astronomy. *Amateur Astronomy: The universe as seen through a Celestron EdgeHD 14* [online]. c 2013 [cit. 2013-03-13]. Dostupné z: http://www.sergepetiot.com/?page_id=173

(12) Concepts in Digital Imaging Technology: Interline Transfer CCD Architecture. *Molecular Expressions: Optical Microscopy Primer* [online]. © 1998-2013 [cit. 2013-02-10]. Dostupné z: http://micro.magnet.fsu.edu/primer/digitalimaging/concepts/interline.html

(13) Concepts in Digital Imaging Technology: Frame-Transfer CCD Architecture. *Molecular Expressions: Optical Microscopy Primer* [online]. © 1998-2013 [cit. 2013-02-10]. Dostupné z: http://micro.magnet.fsu.edu/primer/digitalimaging/concepts/frametransfer.html

(14) THEUWISSEN, Albert. *CMOS image sensors: State-of-the-art* [online]. 21. 5. 2008, 6 s. [cit. 2013-02-27]. Dostupné z: http://harvestimaging.com/pubdocs/125_2008_SSE_CMOS_overview.pdf
(15) Technology. SONY. *Sony* [online]. © 2013 [cit. 2013-03-27]. Dostupné z: http://www.sony.net/SonyInfo/technology/technology/theme/exmor_r_01.html
(16) *Datasheet STM32F405xx* [online]. 2013 [cit. 24.5.2014]. Dostupné z: http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00037051.pdf
(17) STM32F4DISCOVERY. STMICROELECTRONICS. *Www.st.com* [online]. ©2014 [cit. 2014-05-26]. Dostupné z: http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419
(18) COMEDIA LTD. *C328-7640 User Manual* [online]. 19.8.2005, 13 s. [cit. 24.5.2014]. Dostupné z: https://www.sparkfun.com/datasheets/Sensors/Imaging/C328_UM.pdf
(19) CHMELAŘ, Pavel. *Stabilizace polohy létajícího objektu pomocí inerciálních senzorů* [online]. Pardubice, 2011 [cit. 2014-05-26]. Dostupné z: http://dspace.upce.cz/handle/10195/33732. Diplomová práce. Univerzita Pardubice.

Příloha A – Přiložené CD

HaklJ_VyuzitiKamery_PCh_2014.pdf

Složka EMBEDDED obsahuje:

- **kameraC328** – software pro ovládání embedded systém s kamerou včetně zdrojových kódů
- **Kamera_C328** – obsahuje program pro MCU kitu STM32F4Discovery se zdrojovými kódy
- **C328-7640** – manuál pro kameru v pdf

Příloha B – Zdrojový kód funkce main

```
int main(void) {

	int i;
	uint8_t operace = 0;
	/* TODO - Add your application code here */
	RCC_Configuration(); //pro nastavení delay
	inicializacePortuLED();
	USART_Inicializace();
	LCDinit();
	vykresleniNazvu();
	//inicializace ledek kitu a uživatelského tlačítka
	STM32F4Discovery_Init();
	/* Infinite loop */
	while (1) {
		if (RX1_prijato == 1) {
			RX1_prijato = 0;
			if ((RXbuffer[0] == 0xAA) && (RXbuffer[1] == 'P')
					&& (RXbuffer[2] == 'A')) { //prikaz pro sync a
nastaveni kamery po take

				Sync();
			}
			if ((RXbuffer[0] == 0xAA) && (RXbuffer[1] == 'P')
					&& (RXbuffer[2] == 'B')) {

				operace = RXbuffer[3];
				ovladaniLED(2, nesviti);
				postup = TAKE;

			}
		}
		if (STM_EVAL_PBGetState(BUTTON_USER) != 0) {
	//vykreslitObrazekZmena();

			delay(500);
		}
		if (ackRX2 == 1) { //ack
			ackRX2 = 0;
			ovladaniLED(0, sviti);
			switch (postup) {
			case INIT:
				postup = PACKET;
				break;
			case PACKET:
				postup = CEKAT;
				break;
			case TAKE:
				postup = GET;
				break;
			case GET:
				postup = CEKAT;;
				break;

			}
			delay(100);
			ovladaniLED(0, nesviti);
		}
		if (nckRX2 == 1) { //chybové hlášení
			nckRX2 = 0;
```

```
                        ovladaniLED(3, sviti);
                        delay(100);
                        ovladaniLED(3, nesviti);
                        postup = NE;
                        //PrikazSync();
                }
                switch (postup) { //posílání příkazů
                case INIT:
                        delay(500);
                        PrikazInit();
                        break;
                case PACKET:
                        delay(500);
                        PrikazPackageSize();
                        break;
                case TAKE:
                        delay(500);
                        PrikazSnapshopt();
                        break;
                case GET:
                        //PrikazSnapshopt();
                        //delay(10);
                        ovladaniLED(1, sviti);
                        delay(10);
                        ovladaniLED(1, nesviti);
                        PrikazGetPicture();
                        break;
                case ZPRACOVAT:
                        VypocetBarev();
                        //delay(100);
                        if (operace == 11) {// provede se zpracovani obrazku na negativ
                                Negativ();
                                vykreslitObrazekZmena();
                                delay(500);
                        }
                        else if (operace == 111) {//provede se zpracovani obrazu
prahovani
                                Laplace();
                                vykreslitObrazekZmena();
                                delay(500);
                        }
                        else{
                                vykreslitObrazek();// vykresli se obrazek z kamery
                                delay(500);
                        }
                        postup = OK;
                        break;
                case OK:
                        PrikazOK();
                        postup = CEKAT;
                        break;
                case NE:
                        PrikazNe();
                        postup = CEKAT;
                        break;
                }
        }
}
```

Příloha C – Zdrojové kódy operací s obrazem

```
void Laplace() {
	// maska pro Laplace
	int velikost = 9;
	int maska[velikost];
	maska[0] = 0;
	maska[1] = -1;
	maska[2] = 0;
	maska[3] = -1;
	maska[4] = 4;
	maska[5] = -1;
	maska[6] = 0;
	maska[7] = -1;
	maska[8] = 0;
	// deklarace a inicializace promennych
	int b, x, y, barva = 3;//, vysledek = 0
	double suma = 0;
	//int stred = 3 / 2; //pocet prvku v dimenzi( v radku) = 3 z masky!!!!!!
vysledkem bude 1
	for (b = 0; b < barva; ++b) {
		for (y = 1; y < vyska-1; ++y) {
			for (x = 1; x < sirka-1; ++x) {
				suma = pracovniData3D[x-1][y-1][b]*maska[0] +
pracovniData3D[x][y-1][b]*maska[1] + pracovniData3D[x+1][y-1][b]*maska[2] +
					pracovniData3D[x-1][y][b]*maska[3] +
pracovniData3D[x][y][b]*maska[4] + pracovniData3D[x+1][y][b]*maska[5] +
					pracovniData3D[x-1][y-1][b]*maska[6] +
pracovniData3D[x][y+1][b]*maska[7] + pracovniData3D[x+1][y+1][b]*maska[8];
				suma = fmin(255,suma);
				suma = fmax(0,suma);
				upraveneData3D[x][y][b] = (uint8_t)suma;
			}
		}
	}


void Negativ() {
	int x, y;
	uint8_t intenzitaCervena = 0, intenzitaZelena = 0, intenzitaModra = 0;
	for (y = 0; y < vyska; ++y) {
		for (x = 0; x < sirka; ++x) {
			// vypocty intezit jednotlivych barev
			intenzitaCervena = (uint8_t) (255 - pracovniData3D[x][y][0]);
			upraveneData3D[x][y][0] = intenzitaCervena;
			intenzitaZelena = (uint8_t) (255 - pracovniData3D[x][y][1]);
			upraveneData3D[x][y][1] = intenzitaZelena;
			intenzitaModra = (uint8_t) (255 - pracovniData3D[x][y][2]);
			upraveneData3D[x][y][2] = intenzitaModra;
		}
	}
}
```

```
void VypocetBarev() {
int pom = 0, x, y, cervena, zelena, modra;
int maskaCervena = 0xF800, maskaZelena = 0x7E0, maskaModra = 0x1F;
for (y = 0; y < vyska; ++y) {
	for (x = 0; x < sirka; ++x) {
		int pozice = y * (sirka * 2) + (x * 2);
		pom = (poleDat[pozice + 0] << 8) | (poleDat[pozice + 1]);
		cervena = (pom & maskaCervena) >> 11; //vymaskovani zelene a modra
barvy
		cervena = (cervena * 255) / 31; //prepocet z 5bitu na 1byte
		pracovniData3D[x][y][0] = (uint8_t) cervena; // ulozeni do pole
		zelena = (pom & maskaZelena) >> 5; //vymaskovani cervene a modre barvy
		zelena = (zelena * 255) / 63; //prepocet z 6bitu na 1byte
		pracovniData3D[x][y][1] = (uint8_t) zelena; // ulozeni do pole
		modra = (pom & maskaModra); //vymaskovani cervene a zelene barvy
		modra = (modra * 255) / 31; //prepocet z 5bitu na 1byte
		pracovniData3D[x][y][2] = (uint8_t) modra; // ulozeni do pole
	}
}
}
```

Příloha D – Zdrojové kódy ostatní důležité

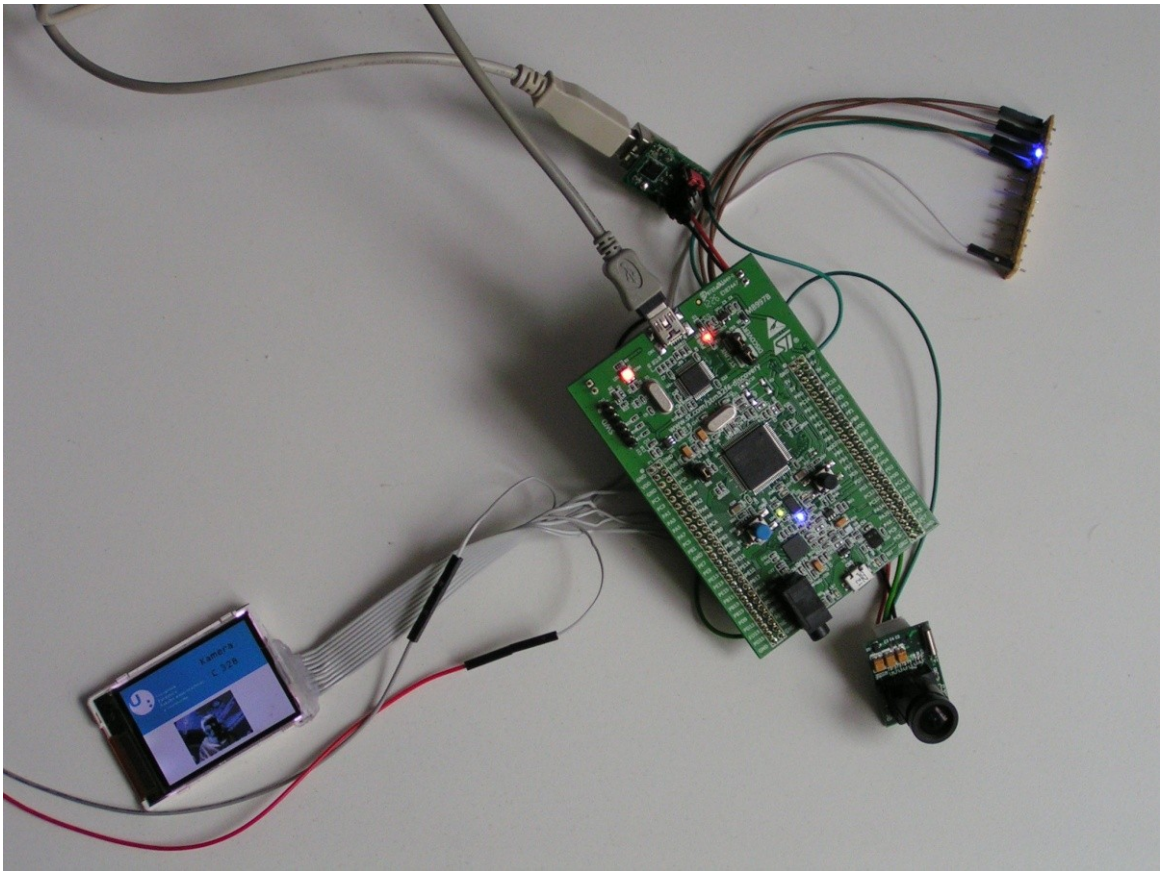
```
void Sync() {
	int i;
	for (i = 0; i < maximum; ++i) { // start kamery
		PrikazSync();
		//	bezErroru = AckNeboNck(pole);
		if (ackRX2 == 1 && syncRX2 == 1) {
			STM_EVAL_LEDOn(LED6);
			PrikazACK(0);
			delay(2000);
			ackRX2 = 0;
			syncRX2 = 0;
			postup = INIT;
			break;
		}
		delay(50);
	}
}


if (USART_GetITStatus(USART2, USART_IT_RXNE ) != RESET) {
		pom = USART_ReceiveData(USART2);
		if (delkaDat == POCET_BITU_DAT && dataRX2 == 1) {
			poleDat[indexDat++]=pom;
			if (indexDat == POCET_BITU_DAT) {
				delkaDat = 0;
				dataRX2 = 0;
				indexDat = 0;
				PrikazACK(10);
				postup = ZPRACOVAT;

				ovladaniLED(2, sviti);
			}
		} else {
			R_Buff_CMD[RX_Index_CMD++] = pom;
		}
		if (RX_Index_CMD == 6 && dataRX2 != 1) {
			//zprávy
			//RX2_prijato = 1;
			if ((R_Buff_CMD[0] == 0xAA) && (R_Buff_CMD[1] == 0x0E)) { //ack
				ackRX2 = 1;
			}
			if ((R_Buff_CMD[0] == 0xAA) && (R_Buff_CMD[1] == 0x0D)) {
//sync
				syncRX2 = 1;
			}
			if ((R_Buff_CMD[0] == 0xAA) && (R_Buff_CMD[1] == 0x0F)) {
//error
				nckRX2 = 1;
			}
			if ((R_Buff_CMD[0] == 0xAA) && (R_Buff_CMD[1] == 0x0A)) {
//data
				delkaDat = R_Buff_CMD[5] << 16 | R_Buff_CMD[4] << 8
					| R_Buff_CMD[3];
				dataRX2 = 1;
			}
			RX_Index_CMD = 0;
		}
```

Příloha E – Obrazová dokumentace

Zapojení obvodu



Zobrazení na displeji – RAW

Zobrazení na displeji – Laplace

Zobrazení na displeji – Negativ