ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAHE

Fakulta elektrotechnická
Katedra meraní

Rýchly vstup dát pre Raspberry PI

High Speed Data Input for Raspberry PI

Bakalárska práca

Študijný program: Kybernetika a robotika
Študijný odbor: Senzory a prístrojová technika

Vedúci práce: Doc. Ing. Jan Fischer, CSc.

Matej Ondrička

Praha 2014

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
Katedra měření

Akademický rok **2013-14**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student:	**Matej Ondrička**
Studijní program:	**Kybernetika a robotika**
Obor:	**Senzory a přístrojová technika**
Název tématu česky:	**Rychlý vstup dat pro Raspberry PI**
Název tématu anglicky:	**High speed data input for Raspberry PI**

Pokyny pro vypracování:

Navrhněte metodu rychlého vstupu digitalizovaného obrazu a dalších dat do modulu Raspberry PI a jemu podobných, které využívají procesory ARM a pracují s některou variantou operačního systému Linux. Přednostně se orientujte na použití rozhraní USB 2.0 a řadiče firmy Cypress. Navrženou metodu implementujte a prakticky ověřte pro přenos obrazu z obrazových senzorů CMOS. Vytvořte potřebné programové vybavení tak, aby modul Raspberry PI spolu s dalšími bloky mohl fungovat jako základ měřicí kamery.

Seznam odborné literatury:

[1] Universal Serial Bus Specification, Revision 2.0. April 27, 2000, http://www.usb.org
[2] EZ-USB FX2LPt USB Microcontroller, Cypress, Document #: 38-08032
[3] Skalický P.: Mikroprocesory řady 8051, BEN, Praha 2002

Vedoucí bakalářské práce:	doc. Ing. Jan Fischer, CSc.
Datum zadání bakalářské práce:	25. listopadu 2013
Platnost zadání do[1]:	30. ledna 2015

Prof. Ing. Vladimír Haasz, CSc.
vedoucí katedry

L.S.

Prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 25. 11. 2013

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

ABSTRAKT

Bakalárska práca sa zaoberá rýchlym vstupom dát do linuxového systému Raspberry Pi cez zbernicu USB. Cieľom práce je použitie Raspberry Pi spolu s obrazovým senzorom CMOS Micron MT9M001 ako základ meracej kamery. Komunikáciu medzi CMOS senzorom a Raspberry Pi zabezpečuje obvod CY7C68013A od spoločnosti Cypress. Aplikácia pre Raspberry Pi je písaná v jazyku C a pre komunikáciu s obvodom využíva knižnicu libusb. Práca testuje spoľahlivosť a rýchlosť USB prenosu s využitím čítača 74HC191. Navrhuje metódu pre odber dát jedného snímka z CMOS senzora s využitím vstavaného procesora 8051. Pre kontinuálne zobrazenie prijatých dát v okne programu využíva knižnicu OpenCV.

V závere práce sú uvedené merania snímkovej frekvencie pri rôznych rozlíšeniach obrazového senzora a zhodnotené dosiahnuté výsledky práce.

KĽÚČOVÉ SLOVA

Raspberry Pi. Cypress CY7C68013. USB. Endpoint. Linux. Libusb. Obrazový senzor CMOS. Micron MT9M001. OpenCV.

ABSTRACT

The scope of this Bachelor Thesis deals with High Speed Data Input using USB bus for the Rapsberry Pi Linux system. Main goal of the is to use Rapsberry Pi together with CMOS Image Sensor Micron MT9M001 as a base of the measuring camera. Communication between CMOS Image Sensor and Rapsberry Pi is provided by the CY7C68013A Cypress controller. The application for Rapsberry Pi is created in C programming language and the communitacion with Cypress Controller uses libusb library. In a Thesis the realiability and speed of the USB transfer has been tested using 74HC191 4-bit counter. For simultaneous representation of received data in the application window use OpenCV library.

At the end of the Bachelor thesis there are shown frame rates measurements with different CMOS resolutions and summarize achieved results.

KEYWORDS

Raspberry Pi. Cypress CY7C68013. USB. Endpoint. Linux. Libusb. CMOS Image Sensor. Micron MT9M001. OpenCV.

PREHLÁSENIE

Prehlasujem, že som predloženú bakalársku prácu vypracoval samostatne a že som uviedol informačné zdroje v súlade s Metodickým pokynom o dodržiavaní etických princípov pri príprave vysokoškolských záverečných prác.

Táto práca vznikla v laboratórií videometrie, katedry meraní ČVUT – FEL v Prahe pod vedením doc. Ing. Jana Fischera, CSc. Nadväzuje tiež na výskum v rámci MSM6840770015 – „Výskum metód a systémov pre meranie fyzikálnych veličín a spracovanie nameraných dát“, ktorého niektoré poznatky a výstupy v oblasti optoelektronických senzorov taktiež využíva.

V Prahe dňa …… …….

Matej Ondrička

POĎAKOVANIE

Ďakujem vedúcemu bakalárskej práce doc. Ing. Janovi Fischerovi, CSc. za účinnú metodickú, pedagogickú a odbornú pomoc, za zapožičanie potrebného hardvérového vybavenia k vypracovaniu práce, za umožnenie vstupu do Laboratória videometrie a ďalšie cenné rady pri spracovaní mojej bakalárskej práce. Taktiež chcem poďakovať Bc. Michaele Kavkovej za pomoc pri práci s knižnicou OpenCV.

OBSAH

ÚVOD

Raspberry Pi je malý a lacný počítač (cena cca. 35$) s výkonným procesorom ARM a 512 MB operačnou pamäťou RAM. Využíva vlastný operačný systém na linuxových distribúciách. Raspberry Pi sa teda stáva výkonným a spoľahlivým zariadením a nachádza využitie v rôznych aplikáciách. Preto by toto kompaktné zariadenie mohlo slúžiť ako monitorovacie zariadenie okolitej scény s využitím obrazových senzorov.

Spoločnosť Raspberry Pi ponúka k počítaču kamerový modul s 5 Mpix CMOS senzorom Omni-Vision OV5647. Senzor dokáže v HD rozlíšení generovať 30 fps. K Raspberry Pi sa pripája pomocou CSI konektora, teda sériového rozhrania. To možno považovať za veľkú nevýhodu. Druhou nevýhodou je obmedzená možnosť konfigurácie vnútorných registrov senzora, teda nastavenie senzora. Navyše kúpou tohto senzora sme obmedzení použitím len pre Raspberry Pi.

Ďalším rozhraním, ktoré je súčasťou takmer každého PC je USB. Rovnako je tomu aj v prípade Raspberry Pi, ktoré poskytuje 2 porty USB2.0. Zbernica nachádza hlavné využitie pre pripájanie periférií a rovnako je tomu aj v prípade Raspberry Pi. Práca sa preto orientuje na využitie zbernice USB s využitím obvodu CY7C68013A od spoločnosti Cypress. Táto spoločnosť poskytuje pre obvod softvérové vybavenie a knižnicu umožňujúcu komunikáciu pre systémy Windows. Z tohto dôvodu vzniklo na Katedre meraní mnoho aplikácií práve pre túto platformu.

Ako už bolo spomenuté, Raspberry Pi pracuje s linuxovým OS. Preto dôležitou súčasťou práce bude vývoj aplikácie pre Raspberry Pi, ktorá zabezpečí komunikáciu s obvodom Cypress (USB radič). Po prekonaní tejto bariéry môžu byť dáta získavané z ľubovoľného digitálneho obvodu s paralelným výstupom. Prevažne sa však orientujeme na dáta získané z obrazového CMOS senzora s paralelným rozhraním. Získame tak väčšiu variabilitu použitia obrazových senzorov, čo je obrovská výhoda pre Laboratórium videometrie, ktoré tieto senzory používa. Je nutné zabezpečiť spoľahlivý a bezstratový prenos dát cez zbernicu USB. Preto treba vykonať sadu meraní pre určenie maximálnej dosiahnuteľnej rýchlosti hostiteľského USB radiča modulu Raspberry Pi.

V konečnej fáze bude potrebné vytvoriť programové vybavenie tak, aby Raspberry Pi spolu s obvodom Cypress a CMOS senzorom mohli tvoriť základ meracej kamery.

1 RASPBERRY PI

Raspberry Pi (ďalej len RPi) je lacný počítač pozostávajúci z jedinej dosky (obr. 1). Veľkosť počítača sa prirovnáva veľkosti kreditnej karty a umožňuje k sebe pripojiť periférie ako monitor, klávesnicu a myš. Tým je možné vytvoriť malý plnohodnotný počítač. Navyše umožňuje prehrávať video vo Full HD rozlíšení, a preto sa pomerne často používa ako média server k TV. Časté využitie taktiež nachádza v automatizácií a inteligentných budovách pri zbere dát a riadení procesov, ďalej ako web server a sieťové úložisko. Možností, ktoré RPi ponúka je však podstatne viac.

Obr. 1 RPi model B (prevzaté z [1])

1.1 Technická špecifikácia RPi

Počítač je založený na obvode BCM2835, ktorý v sebe zahŕňa 700 MHz procesor ARM1176JZF-S, VideoCore IV GPU a pôvodne bol navrhnutý s 256 MB RAM (model A), neskôr (model B) bola veľkosť navýšená na 512 MB (čerpané z [2]). Pri práci bude používaný model B, preto sa modelom A nebudeme zaoberať. RPi neobsahuje pevný disk, ale používa SD kartu pre bootovanie a dlhodobé ukladanie dát. RPi je určené pre operačné systémy založené na linuxových jadrách. Ďalej je vybavené 10/100 eternetovým portom, čím umožňuje vzdialený prístup cez internet. Dva porty USB2.0 poskytujú možnosť pre pripojenie klávesnice a myši. Napájanie RPi je veľmi jednoduché a elegantné. RPi nemá žiadny vypínač, preto po pripojení zdroja s micro-USB konektorom začne automaticky bootovať. RPi má 8 vstupne/výstupných pinov pre GPIO, UART, I2C a SPI zbernicu.

1.2 Potrebné periférie pre lokálne použitie RPi

Aby mohlo byť RPi naplno využívané, je nutné pripojiť periférne zariadenia aj v tom prípade, že k RPi chceme v budúcnosti pristupovať vzdialene (pre vzdialený prístup treba systém nakonfigurovať). Ethernetovým pripojením možno pristupovať

k RPi vzdialene. Softvér potrebný pre vzdialený prístup zbytočne vyťažuje procesor, preto je vhodnejšie k RPi pristupovať lokálne. Lokálnym prístupom rozumieme pripojenie potrebných periférnych zariadení, čím vytvoríme osobný počítač. Navyše pri meraní rýchlosti prenosu potrebujeme mať pripojený USB radič k RPi a budú využívané meracie prístroje Laboratória videometrie. To je hlavný dôvod vylúčenia vzdialeného prístupu. Preto budú v nasledujúcich podkapitolách popísané dôležité hardvérové periférie pre lokálny prístup k RPi.

Tab. 1 Technická špecifikácia RPi

	Model B
SoC	Broadcom BCM2835 (CPU, GPU, DSP, SDRAM)
CPU	700 MHz ARM1176JZF-S core (ARM11 family)
GPU	Broadcom VideoCore IV, OpenGL ES 2.0, 1080p30 h.264/MPEG-4 AVC high-profile decoder
Pamäť (SDRAM)	512 MB (zdieľaná s GPU)
USB 2.0 porty	2 (cez integrovaný USB hub)
Video výstup	Kompozitný RCA, HDMI
Audio výstup	3.5 mm jack, HDMI
Úložisko dát	SD, MMC, SDIO slot
Sieťové pripojenie	10/100 Ethernet (RJ45)
Nízkoúrovňové periférie	8 x GPIO, UART, I2C, SPI s dvoma chip selektmi, +3.3V, +5V, Ground
Min. napájací prúd	700 mA (3.5 W)
Napájací zdroj	5V cez MicroUSB alebo GPIO header
Rozmery	85.60 x 53.98 mm
Operačný systém	Debian GNU/Linux, Fedora, Arch Linux

1.2.1 Zdroj pre napájanie RPi

RPi používa pre napájanie micro-USB konektor. Štandardná nabíjačka mobilných telefónov preto poslúži ako vhodný napájací zdroj. Musí však dodávať prúd 700mA pri napätí 5V DC.

1.2.2 Možnosti pripojenia zobrazovacích jednotiek k RPi

RPi má dve výstupné rozhrania pre pripojenie zobrazovacích jednotiek, HDMI pre vysoké rozlíšenie a kompozitný videosignál pre nízke rozlíšenie.

HD televízory a väčšina súčasných monitorov môže byť pripojená použitím HDMI kábla. Použitím tohto kábla je zvuk aj video dostupné cez HDMI. Taktiež je podporovaný aj HDMI vstup a v prípade, že RPi je použité ako média server k TV, je možné s RPi komunikovať diaľkovým ovládačom TV. Pre pripojenie DVI monitorov je nutné použiť HDMI-DVI prevodník.

Pri pripájaní sa k starším typom DVI displejov s využitím HDMI-DVI adaptéra boli spozorované menšie problémy. Užívateľ by očakával, že ak počas bootovacieho procesu pripojí k RPi HDMI-DVI kábel, displej nebude mať problém so zobrazením. Staršie typy monitorov však nerozpoznali žiadny signál a prešli do úsporného režimu. Ako najúčinnejšia metóda proti tomuto javu je súčasné pripojenie RPi a displeja do elektrickej siete.

Druhým, v dnešnej dobe postupne miznúcim rozhraním je kompozitný videosignál

(žltý kábel). Možno tiež použiť prevodník z kompozitného videosignálu na SCART rozhranie. Týmto rozhraním môže byť RPi pripojené k starším typom TV. V tomto prípade je zvuk prenášaný analógovo cez 3.5 mm konektor. Pre pripojenie zvuku k TV je potrebný RCA prevodník. Pripojenie VGA monitora je možné s použitím aktívneho DVI-VGA prevodník.

1.2.3 USB hub pre navýšenie počtu pripojených periférií

Ak chceme k RPi pripojiť viac zariadení, ako je počet dostupných USB portov, je potrebné využiť USB hub. Pre napájanie USB zariadení sa odporúča použiť hub napájaný externým zdrojom 5V DC. Pripojené USB periférie teda nebudú preťažovať obvod stabilizátora napätia na doske RPi určený pre napájanie periférií.

Treba si uvedomiť, že podľa USB špecifikácie [4] je stanovený maximálny prúdový odber na jeden port 500 mA.

1.2.4 Kompatibilita USB klávesníc a myši

Štandardné klávesnice a myši s rozhraním USB pracujú s RPi bez menších problémov. Kompatibilita s multimediálnou klávesnicou nebola testovaná. Vyskytol sa problém, keď USB klávesnica od spoločnosti Dell (typ KB-212B) pripojená k RPi cez hub nekomunikovala s RPI. Pri priamom pripojení (teda bez rozbočovača) klávesnica fungovala bez problémov.

V prípade setu bezdrátovej klávesnice a myši obsadí pripojený rádiofrekvenčný adaptér jediný USB port. Druhý port zostane teda voľný pre iné periférne zariadenie.

2 PRÍPRAVA SD KARTY PRE RPI

Zavedenie OS do RPi je odlišné od spôsobov, na ktoré sme zvyknutí pri inštalácií OS Windows alebo Linux. Spoločnosť Raspberry Pi Fundation poskytuje mnoho operačných systémov, ktoré sú vyvinuté priamo pre RPi. Najvhodnejšou voľbou pre použitie RPi ako klasického stolného PC je operačný systém Raspbian. Raspbian je verzia Debian Linux-u (**Raspbian** = **Rasp**berry + De**bian**), ktorá je špeciálne vytvorená pre RPi.

2.1 Stiahnutie systému Raspbian z webových stránok

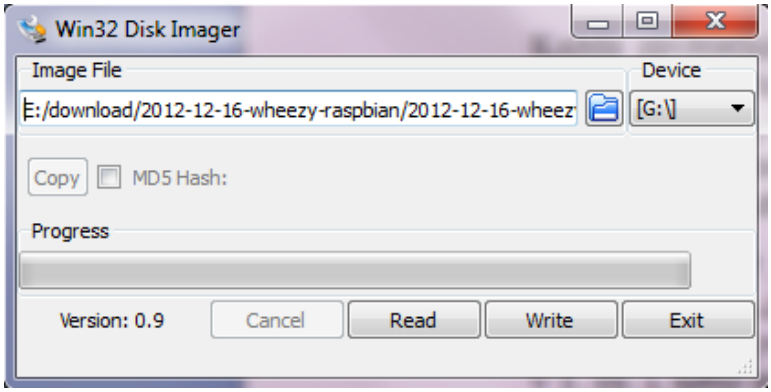
Raspbian je voľne dostupný z webových stránok spoločnosti [2] v sekcii *Downloads*. Existujú dve možnosti stiahnutia OS. Jednou možnosťou je stiahnutie torrent odkazu a následné stiahnutie torrent downloaderom. Druhý spôsob umožňuje stiahnutie priamo z webových stránok. Stiahnutý ZIP súbor rozbalíme a získame jeden *.img* súbor.

2.2 Nahratie súboru .img na SD kartu s využitím systému Windows

Postup pri zavádzaní súboru *.img* sa odlišuje od typu OS, na ktorom bude toto nahrávanie uskutočňované. Ako užívateľ systému Windows popisujem spôsob zavedenia pod týmto OS (postup vychádza z oficiálnych stránok spoločnosti RPi [3]).

Minimálna veľkosť SD karty je 2 GB, čo je dostatočná veľkosť pre Raspbian. Je však doporučené použiť kartu s väčšou pamäťou pre prípad, ak by chcel užívateľ doinštalovať aj ďalšie aplikácie. Ja používam pri svojej práci 16 GB (ADATA SDHC class 10) kartu, ktorá by mala postačovať pre ďalšiu prácu.

Pred samotným nahratím súboru *.img* na pamäťovú kartu je vhodné kartu sformátovať. Formátovanie stačí previesť klasickým spôsobom, na ktorý sme zvyknutí. V okne *Tento počítač* vyberieme kartu, ktorú máme v čítačke a pravým kliknutím myši vyvoláme kontextové menu, v ktorom vyberieme položku *Formátovať*. Nastavenie formátovania ponecháme na *FAT32(východzie)*.



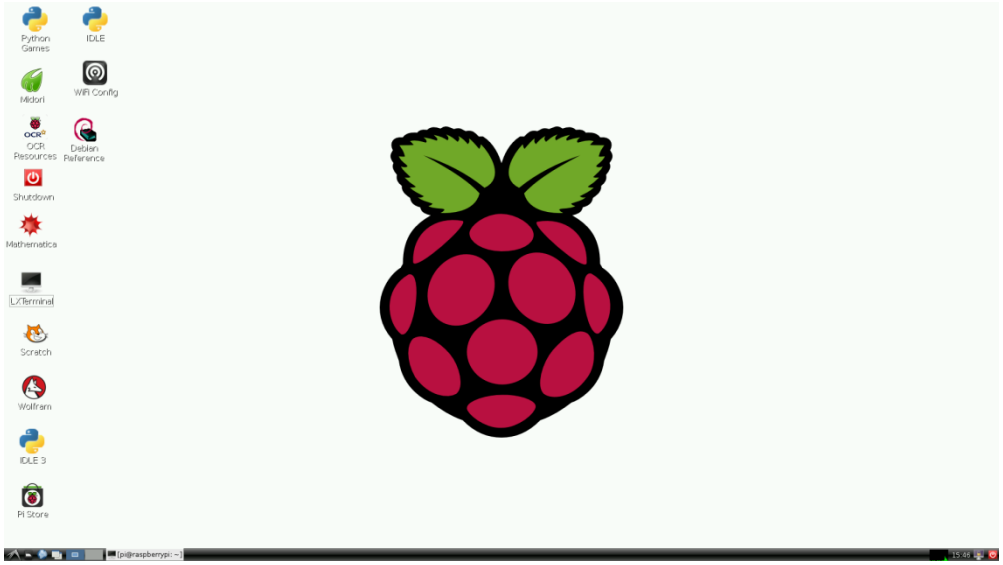
Obr. 2 Win32DiskImager

Na takto pripravenú kartu môže byť zapísaný *.img* súbor. Tento súbor virtuálneho obrazu, nestačí len jednoducho prekopírovať do zložky (koreňového adresára) SD karty. Najjednoduchšou voľbou je použitie programu Win32DiskImager. Program je potrebné po stiahnutí rozbaliť a spustiť ako správca. V programe vyberieme cestu k obrazovému súboru a písmeno diskového zariadenia resp. čítačky kariet, v ktorej je zasunutá SD karta (obr.2). Následným kliknutím na tlačidlo *Write* sa začne zápis dát na kartu.

2.3 Spustenie systému Raspbian na RPi

Po pripojení periférií a vložení SD karty je RPi pripravené na spustenie. Pripojením napájania začne RPi automaticky bootovať. Ako prvá sa zobrazí obrazovka, ktorá slúži pre základnú konfiguráciu systému, ako napr. nastavenie klávesnice, zmena hesla atď. Tieto nastavenia je možné kedykoľvek v budúcnosti zmeniť. Pre dokončenie konfigurácie vyberieme položku *Finish* a systém sa reštartuje.

Po opätovnom bootovaní sa nám zobrazí príkazový riadok, kde sa prihlásime do systému zadaním mena **pi** a hesla **raspberry**. Príkazom **startx** prejdeme do grafického prostredia systému (obr. 3).



Obr. 3 Plocha systému v grafickom prostredí

3 ÚVOD DO PROBLEMATIKY USB

Zbernica USB bola pôvodne navrhnutá pre pripájanie periférnych zariadení k PC. Špecifikácia USB 1.1 podporuje rýchlosti Low Speed (1,5 Mbit/s) a Full Speed (12 Mbit/s), čo je postačujúce pre aplikácie ako napr. klávesnica, myš, audio a mikrofón.

Špecifikácia USB 2.0 prináša so sebou rýchlosť High Speed (480 Mbit/s), pričom hlavné využitie je najmä v prenose veľkých dátových paketov a streamovanie videa. Oproti verzií 1.1 má zariadenie väčšiu veľkosť buffrov endpointov. Rýchlosť 480 Mbit/s platí pri maximálnom využití zbernice, teda súčte všetkých prenosov prebiehajúcich cez zbernicu. Verzia USB 2.0 je spätne kompatibilná s verziou USB 1.1 (kapitola čerpaná z [4]).

Štandard USB umožňuje pripojenie 127 zariadení k hostiteľskému radiču s využitím hubov. Tento radič je súčasťou každého PC, ktoré je definované ako host, teda pripájame k nemu periférie. Úlohou hosta je nadviazanie a zabezpečenie prenosu v oboch smeroch, pričom smer prenosu sa určuje z pohľadu hosta.

V nasledujúcej časti budú vysvetlené základné termíny komunikačného toku USB, ktoré sa budú v práci vyskytovať.

3.1 Deskriptor

Deskriptor je dátová štruktúra s definovaným dátovým formátom, ktorá obsahuje informácie a možnosti konfigurácie daného zariadenia. Každé zariadenie používa viacero deskriptorov (konfiguračný, zariadenia, interface, endpoint,...). Daný typ deskriptora uchováva v sebe informácie ako typ prenosu, počet endpointov a ich veľkosť, informácie o výrobcovi, verzií zariadenia a mnoho ďalších. Tieto informácie sú dôležité pre správnu funkčnosť zariadenia.

3.2 Endpoint

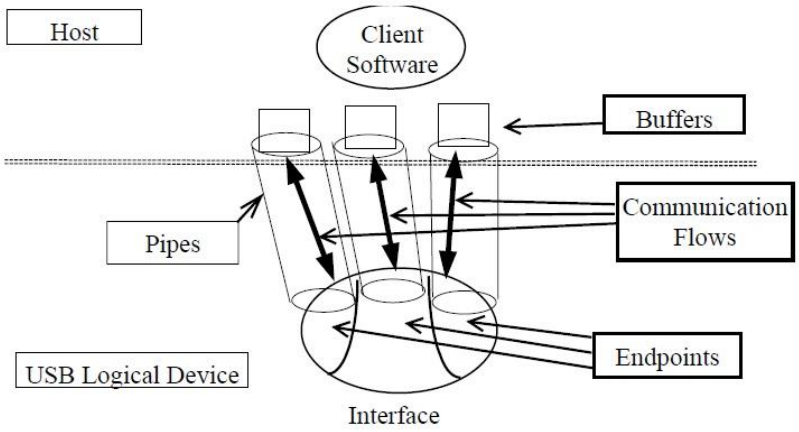
Endpoint je jedinečná časť USB zariadenia, ktorá zabezpečuje prepojenie komunikačných rúr medzi hostom a zariadením (obr. 4). Každé zariadenie má viacero endpointov, kde každý endpoint má svoju adresu. Endpoint môže byť nakonfigurovaný rôznymi spôsobmi pre rôzne typy prenosov. Má definované požiadavky na frekvenciu a latenciu prístupu, maximálnu veľkosť dátového toku, správanie pri obsluhe chýb, svoje číslo a smer.

Endpoint na adrese 0 (EP0) musí byť definovaný v každom zariadení, pretože preberá riadiacu úlohu a enumeráciu zariadenia pri jeho pripojení k PC. Ako jediný zo všetkých je obojsmerný, ostatné endpointy môžu byť nakonfigurované ako IN alebo OUT.

3.3 Rúra

USB rúra (pipe) vytvára schopnosť prenosu dát medzi softvérom hostiteľa cez pamäťový buffer a endpointom zariadenia (obr. 4). Default Control Pipe je rúra

vyhradená pre EP0. Slúži na vyčítanie identifikačných a konfiguračných informácií o pripojenom zariadení a konfiguráciu samotného zariadenia.



Obr. 4 USB komunikačný tok (prevzaté z [4])

Každá rúra udáva svoj stav (nečinná/obsadená). Paketom IRP (I/O Request Packet) sa zadáva požiadavka na využitie rúry. Ak je rúra voľná, zablokuje sa pre ostatných a prenesie sa požadovaný počet bajtov. Nie je problém preniesť väčšie množstvo dát, ako sa zmestí do jedného dátového paketu, pretože obsluha rúr je automatizovaná hostiteľským radičom. Existujú dva typy rúr:

- Rúry správ – používajú sa v riadiacich prenosoch, kde dáta majú definovanú štruktúru.
- Prúdové rúry – používajú sa v ostatných typoch prenosov a nemajú presne definovanú dátovú štruktúru. Tento typ rúry je správ jednosmerný.

3.4 Paket

Hostiteľský radič spracováva príkazy ovládača, na základe ktorých vytvára transakcie. Transakcia je zložená z niekoľkých typov paketov. Transakcie sa vkladajú do úsekov, ktoré nazývame (mikro) rámce. V prípade rámca je jeho dĺžka 1 ms, v prípade mikrorámca 125 μs. Mikrorámce sú definované od štandardu USB2.0 v režime High Speed. Pakety rozdeľujeme na nasledovné typy:

- **Token –** slúži pre nadviazanie komunikácie s konkrétnym zariadením. Určuje smer prenosu, adresu zariadenia a číslo endpointu.
- **Data –** po nadviazaní prenosu sa posielajú užitočné dátové pakety.
- **Handshake –** slúži pre potvrdenie prenosu. Pre kontrolu správnosti odoslaného dátového paketu je v ňom uložený kontrolný súčet CRC.
- **Špeciálne –** hlásenie o chybách, doplňujúce informácie pre dátové prenosy.

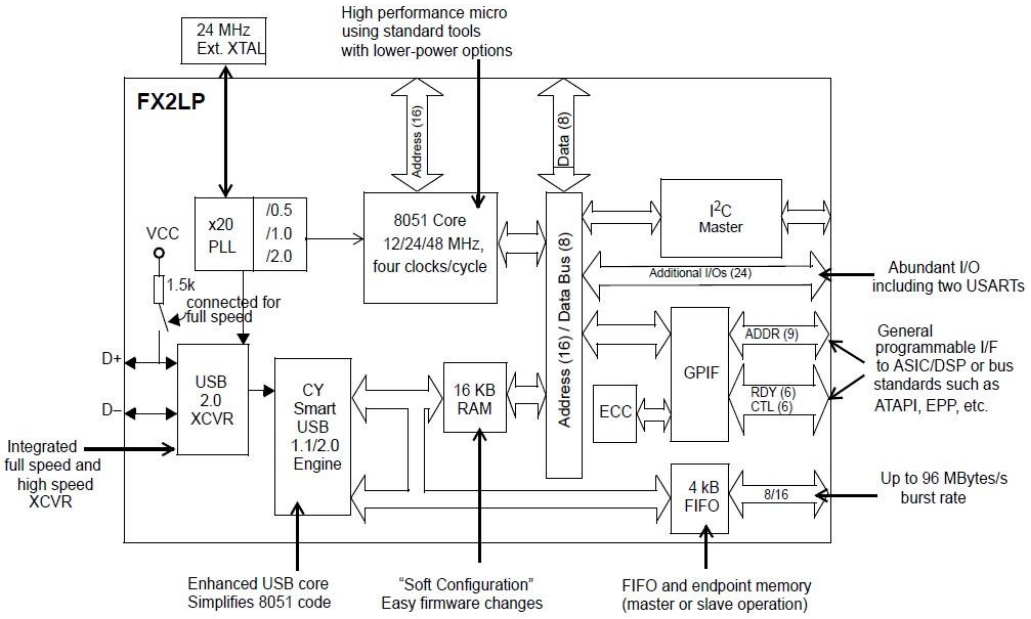
3.5 Typy prenosov

V špecifikácií USB 2.0 sa uvádzajú štyri typy prenosov:

- **Riadiaci –** základný a nevyhnutný typ pre prenos konfiguračných, stavových a riadiacich dát so zariadením. Prenos prebieha prostredníctvom EP0. Maximálna veľkosť paketu je 64 bajtov pre High Speed režim.
- **Izochronný –** slúži pre prenos streamov (audio, video) v reálnom čase. Štandardne sa prevedie jedna transakcia za mikrorámec. Latencia odosielania dát je garantovaná, v prípade chybnej transakcie sa prenos neopakuje. Maximálna veľkosť paketu je 1024 bajtov pre High Speed režim.
- **Bulkový –** slúži pre spoľahlivý prenos dát. Nie je rezervovaná žiadna prenosová kapacita, ale využíva sa zostávajúca. V prípade chyby sa prenos opakuje (CRC zabezpečenie). Maximálna veľkosť paketu je 512 bajtov pre High Speed režim.
- **Prerušovací –** slúži pre bezprostredné obslúženie zariadenia, kde malý dátový paket musí byť spoľahlivo a čo najrýchlejšie prenesený. Využíva sa pri vstupných zariadeniach ako napr. klávesnica a myš. V prípade chyby sa prenos opakuje. Maximálna veľkosť paketu je 1024 bajtov pre High Speed režim.

4 USB RADIČ CY7C68013A

Pre prenos dát cez zbernicu USB do RPi bol vybraný obvod CY7C68013A (ďalej len USB radič) od spoločnosti Cypress. Tento jednočipový obvod je tvorený radičom zbernice USB Serial Interface Engine, vstavaným mikroprocesorom Intel 8051, 8/16 bitovou dátovou zbernicou FIFO s pamäťou 4 kB, 16 kB pamäťou RAM, všeobecne programovateľným rozhraním (GPIF), radičom I2C a UART (viď bloková schéma, obr. 5). Obvod podporuje komunikáciu v režime USB 2.0, teda full speed a high speed. Vyrába sa v troch rôznych prevedeniach s 56, 100 alebo 128 pinovým puzdrom.



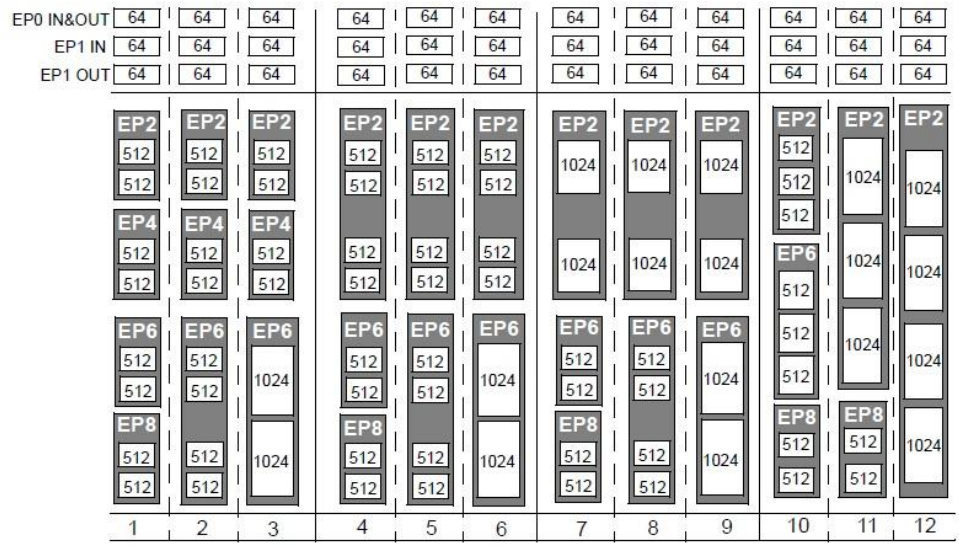
Obr. 5 Bloková schéma obvodu CY7C68013A (prevzaté z [5])

4.1 Endpointy USB radiča

Radič USB zbernice má k dispozícii 6 endpointov (EPx). EP0 je obojsmerný riadiaci endpoint, ktorého veľkosť je 64 B. Do endpointov EP1IN, EP1OUT je možné pristupovať jedine pomocou mikroprocesora 8051 a ich maximálna veľkosť je 64 B pre každý smer.

Celková FIFO pamäť pre všetky endpointy (EP2, EP4, EP6 a EP8) je 4 kB. Všetky tieto endpointy môžu byť pripojené k pamäti FIFO. Buffre endpointov možno rozdeliť do jednej z dvanástich konfigurácií. Jeden typ konfigurácie FIFO pamäte je jedným stĺpcom obr. 6.

Endpointy môžu mať rôznu násobnosť buffrovania. Napr. dvojnásobne buffrovaný endpoint môže jeden buffer prenášať, kým do druhého sa zapisujú dáta. Uvediem príklad konfigurácie č. 3 z obr. 6. Veľkosť EP0, EP1IN a EP1OUT je pre všetky konfigurácie rovnaká, teda 64 B. Veľkosť EP2 je 512 B a využíva dvojnásobné buffrovanie, preto celková veľkosť EP2 je 1024 B. EP4 je obdobná ako EP2. Veľkosť EP6 je 1024 B a je dvojnásobne buffrovaný, preto celkovo využíva 2048 B.



Obr. 6 Konfigurácia endpointov (prevzaté z [5])

Z obr. 6 je ďalej vidieť, že veľkosti EP2 a EP6 môžu byť 512 B alebo 1024 B a veľkosti EP4 a EP8 len 512 B. Dvojnásobne môžu byť buffrované všetky endpointy, trojnásobne len EP2 a štvornásobne len EP2 a EP6.

4.2 Serial Interface Engine (SIE)

Každé USB zariadenie má SIE, ktorý je pripojený k dátovým vodičom USB zbernice. Hlavnou úlohou SIE je dekódovanie paketu, kontrola chybných prenosov pomocou CRC kódu, poskytnutie užitočných dát USB zariadeniu.

Veľkou výhodou USB radiča je, že nepotrebuje pamäť ROM alebo inú pevnú pamäť. Obsahuje internú pamäť RAM pre program aj dáta. Do tejto pamäte možno nahrávať cez rozhranie USB, kým je procesor držaný v stave resetu. Tento proces taktiež obsluhuje SIE.

4.3 Vstavaný mikroprocesor 8051

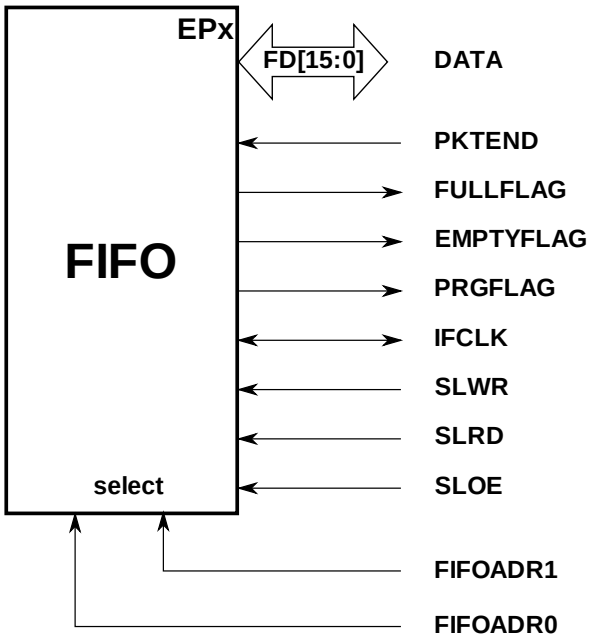
Súčasťou USB radiča je vstavaný mikroprocesor 8051. Procesor komunikuje so SIE sadou registrov, ktoré sú v pamäti RAM. Tieto registre slúžia pre nastavenie endpointov, implementáciu USB protokolov (spracovávanie požiadaviek od daného hosta). Procesor je použitý k riadeniu toku dát a implementácií vlastných funkcií. Oproti klasickej 8051 má trojstavový výstup, vstup je TTL kompatibilný. Pracuje na frekvenciách 12 MHz, 24 MHz alebo 48 MHz so štyrmi cyklami na inštrukciu. Dve z troch I/O brán sú multiplexované s rozhraním FIFO pamäti. Ďalej je procesor vybavený zbernicou I2C a nesmie chýbať ani sériová linka UART.

4.4 Režim FIFO slave

USB radič umožňuje komunikáciu riadenú externým masterom, to znamená, že mikroprocesor 8051 nemusí do komunikácie vôbec zasahovať. Tento typ prenosu je určený pre rýchly prenos dát a je možné prijímať 8 alebo 16 bitové dáta.

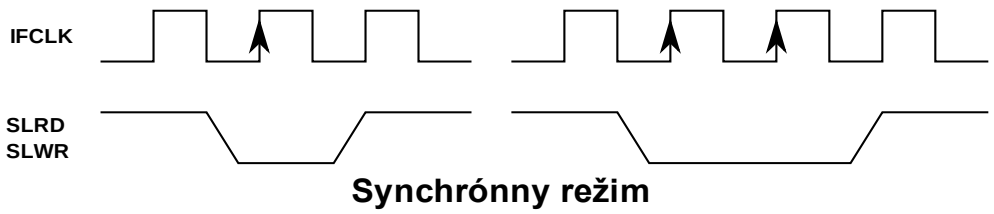
Oblasť FIFO pamäte (čiže endpoint), do ktorej sa bude zapisovať je vyberaná pomocou pinov FIFOADR1 a FIFOADR0. Pomocou príznakov FLAGB (full flag) a FLAGC (empty flag) je možné zistiť stav komunikácie (obr. 7). Príznaky môžu byť nastavené do dvoch módov, indexovaného alebo fixného.

V prípade indexovaného módu sa príznak priraďuje k endpointu, ktorý je vybraný hodnotami FIFOADR. V prípade fixného módu určujú registre PINFLAGSAB a PINFLAGSCD endpointy, ktorým budú príznaky priradené. Taktiež je možné použiť dva programovateľné príznaky, pričom aktívna úroveň všetkých príznakov sa dá nastaviť registrom FIFOPINPOLAR.

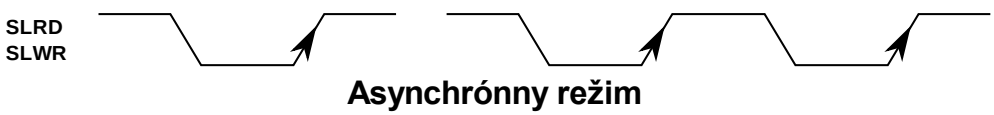


Obr. 7 Režim FIFO slave (podľa [5])

Zápis do FIFO pamäte sa môže riadiť synchrónne alebo asynchrónne. Synchrónny režim znamená, že pri aktívnej úrovni (logická nula) signálu SLRD sa dáta čítajú pri každej nábežnej hrane hodín ICLK. Pri aktívnej úrovni signálu SLWR sa dáta zapisujú pri každej nábežnej hrane hodín IFCLK. Signál SLRD a SLWR slúži len ako kvalifikátor (obr. 8).



Obr. 8 Synchrónny režim (vľavo zápis v jednom takte, vpravo zápis v dvoch taktoch)



Obr. 9 Asynchrónny režim (vľavo zápis v jednom takte, vpravo zápis v dvoch taktoch)

V prípade asynchrónneho režimu sa dáta čítajú/zapisujú z/do FIFO pamäte pri každej nábežnej hrane signálu SLRD/SLWR (obr. 9).

4.5 Dôležité registre USB radiča

Pri tvorbe firmware USB radiča nás budú zaujímať dva dôležité registre z pohľadu správneho fungovania režimu FIFO slave. Jedná sa o register pre konfiguráciu endpointov a register hodín pre zápis do pamäte FIFO. Informácie ku konfigurácií ostatných registrov možno nájsť v [6], kapitola 15.

4.5.1 Register EPxCFG

Register EPxCFG (kde x môže byť 2, 4, 6 alebo 8) slúži pre konfiguráciu daného endpointu. Umožňuje nastaviť násobnosť buffrovania (viď kapitola 4.1 a obr. 6), veľkosť, typ a smer. V prípade EP4 a EP8 nie je možné nastaviť veľkosť, ale je pevne daná na 512 B. Konfiguráciu a význam jednotlivých bitov registra EPxCFG zobrazuje tab. 2.

Tab. 2 Konfigurácia registra EPxCFG

bit	význam
7	0 - zakáž EP, 1 - povoľ EP
6	0 - OUT, 1 - IN (smer)
5&4	01 - izochrónny, 10 - blokový, 11 - prerušovaný
3	0 - 512 B, 1 - 1024 B (veľkosť EP)
2	vždy 0
1&0	00 - štvornásobne, 10 - dvojnásobne, 11 - trojnásobne buffrovaný

4.5.2 Register IFCONFIG

Tento register slúži pre výber typu hodín, ktoré budú riadiť zápis do FIFO pamäte. Dostupné sú interné alebo externé hodinové signály. V prípade interného hodinového signálu je možné vybrať z dvoch dostupných frekvencií, a to 30 MHz a 48 MHz. Externý hodinový signál privedený na pin IFCLK môže byť v rozsahu 5 až 48 MHz. Konfiguráciu a význam jednotlivých bitov registra IFCONFIG zobrazuje tab. 3 a obr. 10.

4.6 Enumerácia USB radiča

Ako už bolo spomenuté v kapitole 4.2, USB radič používa pre program aj dáta pamäť RAM, do ktorej môže byť program zavedený cez zbernicu USB. Tým sa odlišuje od ostatných jadier 8051, ktoré využívajú pre program aj dáta pevnú pamäť (najmä ROM a FLASH).

Radič môže byť enumerovaný rôznymi spôsobmi. Ak nie je pripojená externá pamäť na zbernici I2C, USB radič sa enumeruje ako „Default USB Device“ s hodnotami VID=0x04B4 (Cypress Semiconductor) a PID=0x8613 (EZ-USB FX2LP) a tento stav indikuje bit RENUM, ktorý sa nastaví do nuly.

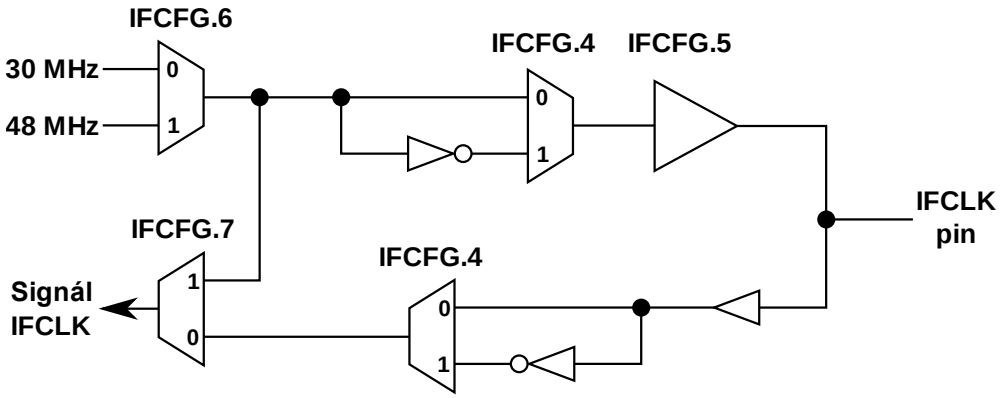
Ak prvý bajt pamäte EEPROM pripojenej na zbernici I2C obsahuje hodnotu 0xC0

a ďalšie bajty hodnoty VID/PID, USB radič sa taktiež inicializuje ako „Default USB Device“. Avšak inicializuje sa s hodnotami VID/PID uloženými v EEPROM. Ak sú tieto hodnoty rôzne od tých, ktoré sú v systémovom ovládači hosta (ovládač bude popísaný v kapitole 4.8), ovládač drží USB radič po enumerácií v resete. Následne je možné nahrať firmware do pamäte RAM, po ktorom systémový ovládač uvoľní obvod z resetu.

Poslednou možnosťou enumerácie USB radiča je stiahnutie celého firmware z pripojenej pamäte EEPROM do RAM. V tomto prípade musí prvý bajt pamäte EEPROM obsahovať hodnotu 0xC2.

Tab. 3 Konfigurácia registra IFCONFIG

bit	význam
7	0 - externý, 1 - interný zdroj hodín
6	0 - 30 MHz, 1 - 48 MHz interné hodiny
5	0 - zakazuje, 1 - povoľuje výstup
4	0 - normálna, 1 - inverzná polarita
3	0 - synchrónny, 1 - asynchrónny mód
2	0 - normálny, 1 - GSTATE
1&0	00 - porty, 11 - FIFO slave



Obr. 10 Konfigurácia registra IFCONFIG pre FIFO slave mód

4.7 Softvér pre vývoj firmware USB radiča

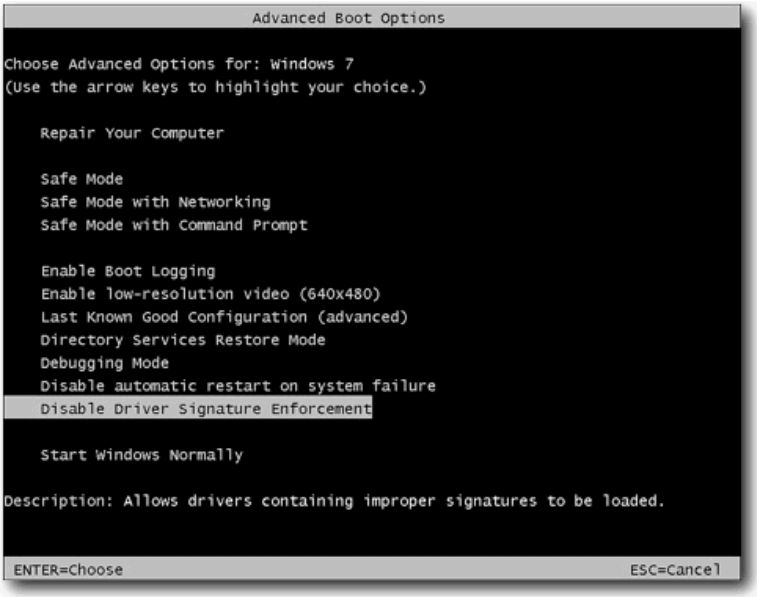
Pri tvorbe firmware bolo vychádzané z ukážkového projektu *bulkloop*, ktorý je súčasťou voľne dostupného (zo stránok spoločnosti Cypress) vývojového balíčka *CySuiteUSB*. Vzhľadom na to, že *IDE Keil uVision* nemá pre unixové systémy žiadnu softvérovú podporu, ďalší vývoj a úpravy firmware budú prebiehať v prostredí *Keil uVision 4* pod OS Windows 7.

Základnou časťou projektu *bulkloop* je zdrojový súbor *bulkloop.c*, v ktorom je jadro aplikácie a funkcie pre inicializáciu obvodu. Ďalším dôležitým súborom je *dscr.a51*, kde sú uložené deskriptory a nastavenia zariadenia.

4.8 Ovládač USB radiča pre systém Windows 7

Ako už bolo spomenuté v kapitole 4.7, firmware pre USB radič bude vytváraný a kompilovaný v *IDE Keil uVision* pod systémom Windows 7. Následne bude firmware nahratý do USB radiča pomocou programu *CyConsole* (súčasť *CySuiteUSB*). Aby bolo možné tento firmware nahrať, systém Windows 7 musí USB radič inicializovať,preto je nutné nainštalovať potrebný ovládač *CyUSB.sys* (súčasť *CySuiteUSB*).

Pred samotnou inštaláciou ovládača pod OS Windows 7 (taktiež Windows Vista a Windows 8) je potreba reštartovať počítač a pri bootovaní systému držať klávesu F8, čím sa dostaneme do rozšíreného nastavenia systému. V tomto nastavení vyberieme možnosť *„Zakázanie vynútenia podpisu ovládača“* (obr. 11). Tým umožníme inštaláciu ovládačov, ktoré obsahujú neplatné podpisy. Treba upozorniť, že táto zmena je dočasná a je nutné ju vykonať pri každom zapnutí počítača.



Obr. 11 Zákaz vynútenia podpisu ovládača

V ďalšom kroku bude upravený súbor *cyusb.inf*, kde pre danú platformu nahradíme symboly XXXX hodnotami VID/ PID. Defaultne má radič VID=04b4 a PID=8613. Ukážkový firmware *bulkloop* používa hodnoty VID=04b4 a PID=1004 (tieto hodnoty budú použité aj pre ostatné verzie firmware), preto vložíme do súboru ešte jeden riadok s týmito hodnotami. Po nahratí nového firmware sa zariadenie inicializuje s novým VID/PID. Z toho užívateľ vie, že nepracuje so zariadením v defaultnej konfigurácií, ale s novým firmware. Na konci súboru *cyusb.inf* v sekcií *[Strings]* taktiež upravíme hodnoty VID/PID a názov zariadenia. Pod týmto názvom sa USB radič zobrazuje v Správcovi zariadení. Zmeny vykonané v súbore *cyusb.inf* sú zobrazené v prílohe A.1.

Po týchto úpravách je možné obvod pripojiť k počítaču. Keď nás OS vyzve, zadáme cestu k tomuto ovládaču.

4.9 Nahratie firmware do USB radiča

Firmware bude vyvíjaný a následne nahratý do USB radiča na PC pod OS Windows. Po nahratí firmware bude USB radič odpojený od tohto PC a pripojený k RPi. Tým dôjde

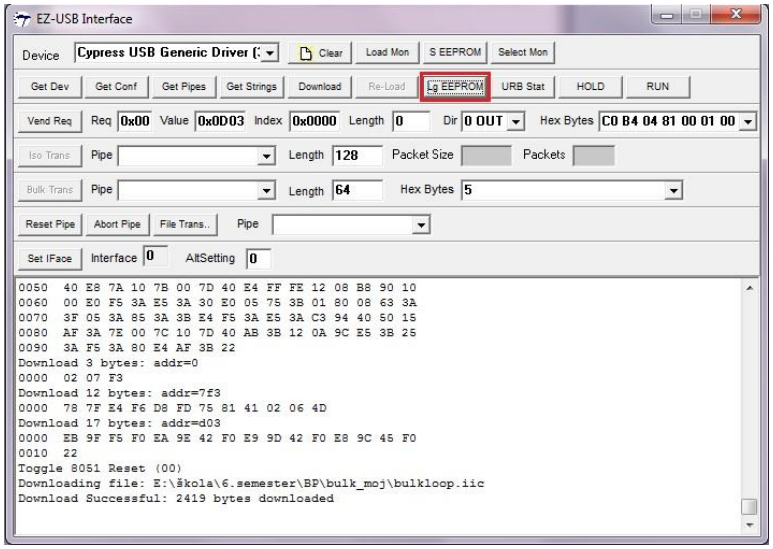
k strate napájania zo zbernice USB a k vymazaniu pamäte RAM. Preto je potrebné nahrávať firmware do pamäte EEPROM.

Výstupný *.hex* súbor prostredia *Keil* je nutné pre zápis do pamäte EEPROM konvertovať na súbor typu *.iic*. Prvý bajt pamäte EEPROM musí obsahovať hodnotu 0xC2 (viď kapitola 4.6). Ku konverzií súboru *.hex* na *.iic* bude použitý program *Hex2bix* (súčasť *CySuiteUSB*). Pre jednoduchosť je vhodné aby bol tento program v zložke projektu. Samotnú konverziu a volanie programu je nutné vykonať v príkazovom riadku (v zložke projektu) spôsobom

```
>Hex2bix.exe -i -f 0xc2 -r -m 0xe200 -v 0x04b4 -p 0x1004 -o
bulkloop.iic bulkloop.hex
```

Parameter *i* znamená, že výstupný súbor bude vo formáte *iic*, parameter *f* určuje hodnotu prvého bajtu, parameter *r* povoluje bootovanie po uvoľnení resetu, prepínač *m* určuje veľkosť pamäte EEPROM, prepínač *v/p* určuje hodnotu VID/PID, parametrom *o* zadávame názov výstupného súboru. Na konci príkazu je názov zdrojového *.hex* súboru.

Nahratie firmware bude vykonané programom *CyConsole*, kde v sekcií *Options* vyberieme možnosť *EZ-USB interface.* Zobrazí sa okno a voľbou *LgEEPROM* (obr. 11) zapíšeme firmware do pamäte EEPROM. Pomocou programu je tiež možné jednoducho skontrolovať konfiguráciu zariadenia.



Obr. 12 Prostredie programu CyConsole

5 KOMUNIKÁCIA RPI S USB RADIČOM

Komunikácia medzi linuxovým systémom a USB radičom nebola doteraz na fakulte realizovaná. Spoločnosť Cypress neposkytuje žiadne linuxové knižnice pre USB radič, preto prvé kroky viedli na diskusné fóra spoločnosti Cypress.

Prvé odpovede odkazovali na použitie EZ-USB FX3 SDK, teda pre vyššiu radu obvodov (USB3.0). Vyskytli sa však problémy pri inštalácií, ktoré mohli byť spôsobené nízkou úrovňou užívateľských znalostí linuxových systémov.

Bádanie teda pokračovalo ďalej a ako priaznivá voľba pre ďalší postup sa ukázalo použitie knižnice *libusb.*

5.1 Základné informácie o knižnici libusb

Knižnica umožňuje aplikáciám jednoduchý prístup k USB zariadeniam, teda vyčítanie deskriptorov, riadenie prenosu dát do a zo zariadenia USB bez potreby ovládačov jadra. Používa sa prevažne na linuxových systémoch (ale aj systémoch Windows) a je písaná v jazyku C.

5.1.1 Inštalácia knižnice libusb

Aplikácia bude používať verziu knižnice *libusb-1.0.* Pomocou linuxového terminálu je inštalácia veľmi jednoduchá (nutné pripojenie na internet)

```
> sudo apt-get install libusb-1.0-0
> sudo apt-get install libusb-1.0-0-dev
```

Po inštalácií knižnice je vhodné urobiť update systému

```
> sudo apt-get update
> sudo apt-get upgrade
```

5.2 Aplikácia na RPi pre komunikáciu s USB radičom

Aplikácia je písaná v jazyku C a pre komunikáciu s USB radičom postačí knižnica *libusb.* Aby aplikácia vyťažovala procesor a pamäť RPi v najmenšej možnej miere, nebude používať žiadne grafické prvky. Bude ovládaná len s využitím príkazového riadku. Preto pre písanie zdrojového kódu postačí ľubovoľný textový editor.

Aby bolo možné v aplikácií pracovať s knižnicou libusb (čerpané z [7]), je nutné pridať na začiatok zdrojového kódu hlavičkový súbor

```
#include <libusb-1.0/libusb.h>
```

Pre prácu s pripojeným USB zariadením je nutné vytvoriť na začiatku programu dve dôležité premenné

```
typdef libusb_device *device;
libusb_device_handle *dev_handle;
```

Knižnica libusb vytvára vlastné dátové typy, kde prvý riadok reprezentuje pripojené

zariadenie a druhý riadok reprezentuje handle, teda pomocný objekt na zariadenie, s ktorým budeme pracovať.

V nasledujúcich bodoch sú popísané dôležité kroky pre správnu komunikáciu s USB radičom.

1. Na začiatku je dôležitá inicializácia knižnice libusb

```
err = libusb_init(NULL);
```

2. Pre výber pripojeného USB radiča musíme získať zoznam všetkých USB zariadení pripojených k hostovi (RPi)

```
cnt = libusb_get_device_list(NULL, &devs);
```

3. Vo funkcií *vypis_info()* vypíšeme zoznam všetkých pripojených USB zariadení spolu s VID/PID, ktoré získame volaním funkcie knižnice

```
libusb_get_device_descriptor(device, &devs);
```

   V tomto kroku nás program vyzve k výberu zariadenia, s ktorým budeme ďalej pracovať. Je možné vybrať akékoľvek zariadenie, avšak komunikácia bude možná len so zariadením s VID=0x04b4 a PID=1004.

4. Pre ďalšie operácie nad USB radičom je mu potrebné priradiť handle

```
err = libusb_open(device, &dev_handle);
```

5. Po skončení prác so zoznamom zariadení, je potrebné zoznam uvoľniť

```
libusb_free_device_list(devs, 1);
```

6. Pre získanie informácií, teda vyčítanie deskriptorov zariadenia a konfiguračných deskriptorov, použijeme nasledujúce volania knižnice

```
libusb_get_device_descriptor(device, &desc);
libusb_get_config_descriptor(device, 0, &config);
```

7. Pre komunikáciu s USB radičom čítaním/zápisom z/do endpointov, je nutné požadovať interface zariadenia. Firmware USB radiča používa len interface číslo nula. Volanie knižnice preto vyzerá nasledovne

```
err = libusb_claim_interface(dev_handle, 0);
```

8. Prenos pre obidva smery (IN a OUT) sa vykonáva volaním rovnakej funkcie. Táto funkcia rozozná o aký smer prenosu ide podľa horného bajtu adresy endpointu (0x00 pre smer OUT, 0x80 pre smer IN)

```
err = libusb_bulk_transfer(dev_handle, EpNum, in_data_buf, paket,
&transferred_bytes, 100);
```

   Táto funkcia je veľmi dôležitá z pohľadu prenosu dát, preto budú popísané aj významy jednotlivých parametrov. Prvý parameter je handle zariadenia, teda handle na USB radič. Druhý parameter určuje smer komunikácie a číslo endpointu, po ktorom sa budú prenášať dáta. Tretí parameter je pole znakov, čiže dátový buffer. Parameter paket určuje, počet bajtov, ktoré majú byť prenesené (veľkosť prenášaného paketu môže byť väčšia ako veľkosť endpointu). Piaty parameter obsahuje aktuálny prenášaný bajt počas prenosu. Posledný parameter udáva dobu čakania (timeout) v milisekundách. Ak sa v tomto čase dáta neprenesú, funkcia skončí chybou.

9. Aby aplikácia predišla chybovým hláseniam, po ukončení prác nad zariadením je potrebné uvoľniť interface, zatvoriť handle zariadenia a taktiež komunikáciu s knižnicou libusb.

```
err = libusb_release_interface(dev_handle, 0);
libusb_close(dev_handle);
libusb_exit(NULL);
```

5.3 Kompilácia a spustenie aplikácie na RPi

Pre kompiláciu zdrojových kódov je v každej z aplikácií priložený *Makefile*, pričom všetky aplikácie budú mať jednotný názov *USBprenos.c*. Kompiláciu a vytvorenie spustiteľného súboru *USBprenosApp* vykonáme volaním príkazu

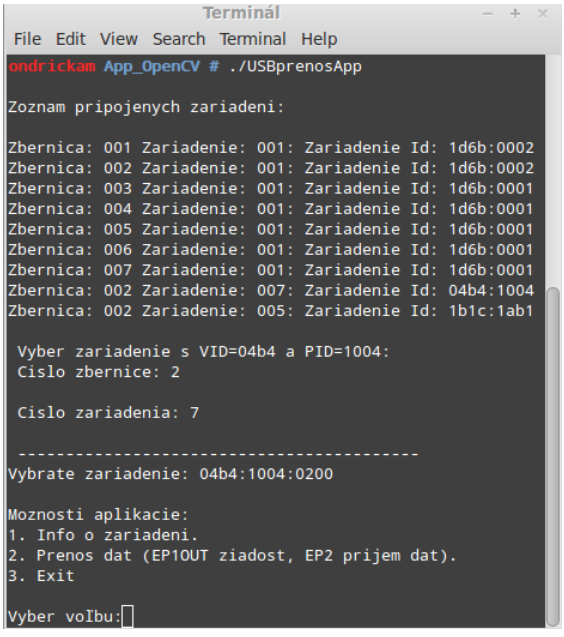
```
>make
```

Aplikácia musí byť spustená s právami roota

```
>sudo ./USBprenosApp
```

Po spustení aplikácia vypíše všetky pripojené USB zariadenia (vrátane hostiteľského radiča). Nás bude zaujímať zariadenie s VID=04b4 a PID=1004 (viď podkapitola 4.8). Podľa týchto hodnôt identifikujeme náš pripojený USB radič. Program nás ďalej navedie pre zadanie zbernice a zariadenia, teda USB radiča. Potom nám ponúkne 3 možnosti:

1. Informácie o pripojenom zariadení.
2. Začiatok prenosu dát.
3. Koniec programu.



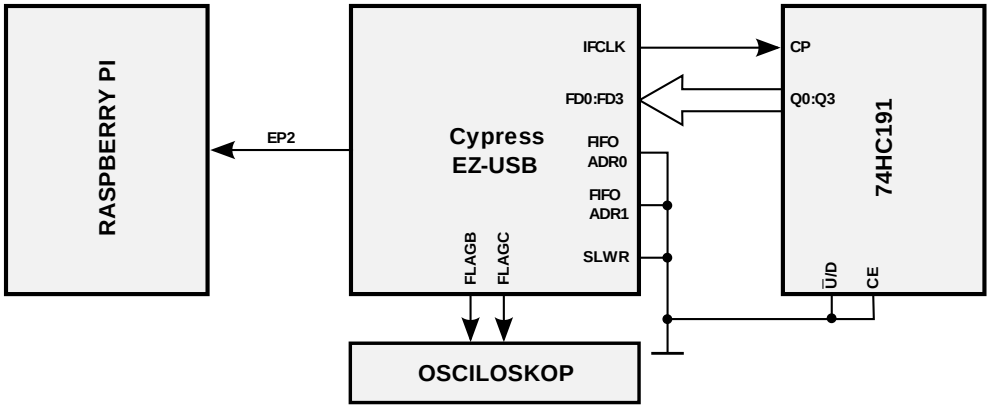
Obr. 13 Základné menu aplikácií

6 TEST RÝCHLOSTI USB VSTUPU DO RPI

Pre prenos dát získaných z obrazového senzora do RPi bude použitý USB radič Cypress. Prenášané dáta sa nemôžu počas prenosu strácať, preto je nutné vykonať meranie spoľahlivosti a maximálnej dosiahnuteľnej rýchlosti USB rozhrania RPi. K tejto strate dát dochádza vtedy, keď rýchlosť zapisovania do FIFO pamäte prevyšuje rýchlosť, ktorou dokáže RPi z FIFO pamäte čítať. Rýchlosť zápisu do FIFO pamäte môžeme ovplyvniť výberom hodinového signálu (interný 30 a 48 MHz, externý 5-48 MHz, viď 4.5.2). Rýchlosť hostiteľského radiča, teda radiča RPi nastavovať nemôžeme. Môžeme ju len negatívne ovplyvniť tým, že na RPi budú spustené aplikácie.

6.1 Hardvérové zapojenie s interným hodinovým signálom (test č. 1)

Pre testovanie rýchlosti USB prenosu bol k USB radiču pripojený 4-bitový synchrónny čítač 74HC191. Výhodou pripojeného čítača je možnosť jednoduchej kontroly prijatých dát a jednoduchosť zapojenia (viď obr 14). Vzhľadom na to, že čítač je 4-bitový, využívajú sa len dolné 4 bity USB radiča. Ostatné sú pripojené na zem. Hodinový signál pre zápis dát do pamäte FIFO bol vybrané s nižšou frekvenciou, teda 30 MHz. Maximálna dosiahnuteľná rýchlosť je 30 MB/s pri 8-bitovej dátovej zbernici. Nižšia frekvencia bola vybraná zámerne, lebo maximálna vstupná frekvencia čítača je 36 MHz [8].



Obr. 14 Test rýchlosti s čítačom

Rýchlosť prenosu bola testovaná postupným zväčšovaním prenášaného paketu v násobkoch dvojky (teda 2^n). Príznakom FULL FLAG a EMPTY FLAG možno osciloskopom pozorovať stav FIFO pamäte. K strate dát dôjde pri zaplnení pamäte FIFO, kedy RPi nie je schopné dáta tak rýchlo čítať. Tento stav sa prejaví zmenou príznaku FULL FLAG z log. 1 do log. 0.

Prvé príznaky zaplnenia pamäte FIFO sa objavili pri dátovom pakete veľkosti 64 kB. To je približne 5-krát menej, ako je potrebné preniesť zo senzoru pri rozlíšení VGA a 8-bitovej hĺbke na pixel. Dôležitý poznatok je, že malé dátové pakety (do 64 kB) je RPi schopné prijímať bez straty dát pri rýchlosti 30 MB/s. Upozorňujem, že na RPi nie je spustená žiadna iná aplikácia, iba tá, ktorá sa stará o komunikáciu s USB radičom.

6.2 Firmware pre test č. 1

Pre uvedenie USB radiča do režimu FIFO slave je treba nastaviť registre vo funkcií *TD_Init()*. Táto funkcia sa volá automaticky pri pripojení obvodu k napájaniu. Za niektoré nastavenia registrov je potrebné vložiť synchronizačné makro SYNCDELAY (detaily v kapitole 15.15 [6]), ktoré slúži k synchronizácií zápisu do registrov. V hlavnom riadiacom cykle, teda funkcií *TD_Pool()* sa procesorom nevykonáva žiadna obsluha prenosu, preto táto funkcia neobsahuje žiaden kód.

Firmware prevedie nastavenie rozhrania FIFO do synchrónneho módu s 8-bitovou dátovou zbernicou a s interným 30 MHz hodinovým signálom IFCLK. Využíva sa iba EP2 v smere IN veľkosti 512 B, ktorý je štvornásobne buffrovaný. Celkovo má teda k dispozícií 2 kB FIFO pamäte.

```
IFCONFIG = 0xA3; SYNCDELAY;      // int. 30MHz hodiny, synchr. mód
REVCTL = 0x03; SYNCDELAY;        // 8051 nespôsobí stratu dát
EP2CFG = 0xE0; SYNCDELAY;        // smer IN, bulk, 512B, 4-nás. buff
FIFORESET = 0x80; SYNCDELAY      // reset pamäte vyhradenej pre EP2
FIFORESET = 0x02; SYNCDELAY;
FIFORESET = 0x00; SYNCDELAY;
EP2FIFOCFG = 0x0C; SYNCDELAY;    // auto in, odoslanie paketu nulovej
                                 dĺžky
EP4FIFOCFG = EP6FIFOCFG = EP8FIFOCFG = 0x00; SYNCDELAY;
PINFLAGSAB = 0x00; SYNCDELAY;    // FLAGA - progr. nast., FLAGB podľa
                                 FIFOADR[1:0]
PINFLAGSCD = 0x00; SYNCDELAY;    // FLAGC ako FLAGB, FLAGD programovo
                                 nastavený
FIFOPINPOLAR = 0x00; SYNCDELAY; // signály aktívne v log. nule
```

6.3 Test č. 2 s externým hodinovým signálom pre spomalenie zápisu do FIFO

Ako ukázal predchádzajúci test, 30 MHz hodinový signál je rýchly pre RPi z pohľadu čítania väčších dátových paketov. Nižší interný hodinový signál módu FIFO slave obvod neponúka, preto treba použiť externý.

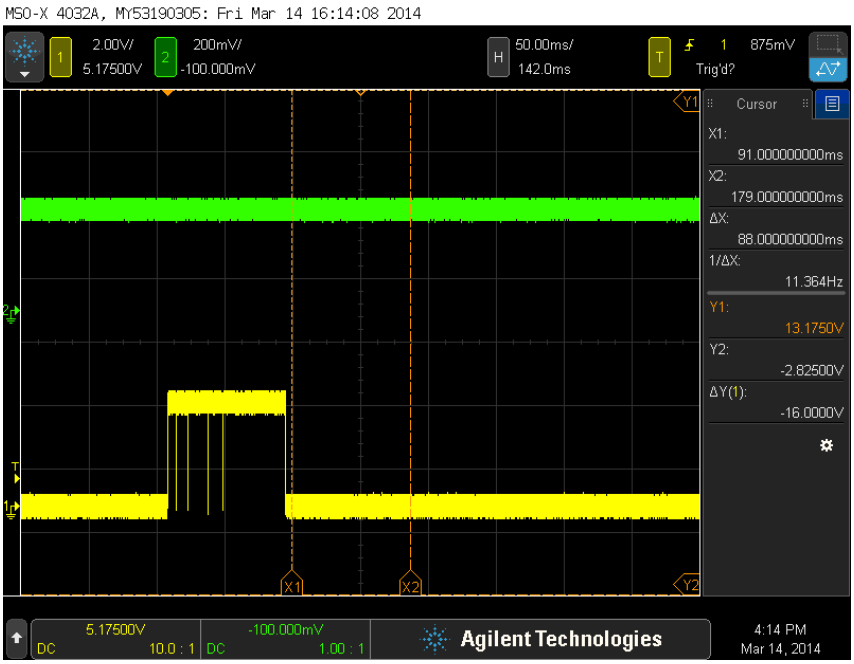
Hodinový signál, ktorý obvod poskytuje je vyvedený na pin CLKOUT. Frekvencia tohto signálu môže byť 12, 24 alebo 48 MHz. Výberom tejto frekvencie udávame aj frekvenciu pre mikroprocesor (viď obr. 5 v kapitole č. 4). To môže mať negatívny dopad v aplikáciách s požiadavkami na rýchlosť. V hardvérovom zapojení z predchádzajúceho testu (obr. 14) stačí priviesť hodinový signál z pinu CLKOUT na pin IFCLK.

Vo firmware vykonáme zmenu v nastavení registra CPUCS a registra IFCONFIG. Registrom CPUCS povolíme výstup CLKOUT a nastavíme frekvenciu na 24 MHz. Registrom IFCONFIG nastavíme pin IFCLK do vstupného módu pre privedenie externých hodín (viď obr. 10).

```
CPUCS = 0x0A; SYNCDELAY:         // 24 MHz, output enable
IFCONFIG = 0x03; SYNCDELAY;      // ext. hodiny, synchr. fifo mód
```

Maximálna dosiahnuteľná rýchlosť v tejto konfigurácií je 24 MB/s pri 8-bitovej dátovej zbernici. Testovanie prenosu neprebiehalo postupným zväčšovaním dátového paketu, lebo veľkosť bola pevne stanovená na 2 MB. Ako ukazuje obr. 15, k zaplneniu FIFO pamäte (FullFlag aktívny v log. 0) dochádza najmä počiatočnej fáze prenášaného paketu. Tu som si uvedomil, že signál SLWR je neustále pripojený na log. úroveň 0

(zem) a dáta sa do FIFO neustále zapisujú (viď obr. 8). RPi teda začne dáta čítať zo zaplnenej FIFO pamäte. Tým však uvoľní miesto pre ďalšie dáta, ktoré sa okamžite zapisujú do FIFO pamäte. Preto môžeme povedať, že pamäť FIFO je neustále na hranici zaplnenia a vôbec neplní účel 2 kB vyrovnávacej pamäte.



Obr. 15 Príznaky EmptyFlag (horný zelený) a FullFlag (dolný žltý)

6.4 Hardvérové zapojenie s externým hodinovým signálom pre FIFO a riadením zápisu SLWR (test č. 3)

Hlavný cieľ, ktorý chceme v novom zapojení dosiahnuť je vyprázdnenie pamäte FIFO pred začatím prenosu. Z hľadiska USB komunikácie pribudne jeden endpoint EP1, ktorým bude RPi zasielať žiadosť o dáta. USB radič túto žiadosť prijme, spracuje a povolí zápis (výstup procesora ovláda SLWR) do FIFO pamäte. V tom momente už je EP2 pripravený prijímať nové dáta. Zapojenie externých hodín z testu č. 2 zostáva nezmenené. Schému zapojenia znázorňuje obr. 16.

6.5 Firmware pre test č. 3

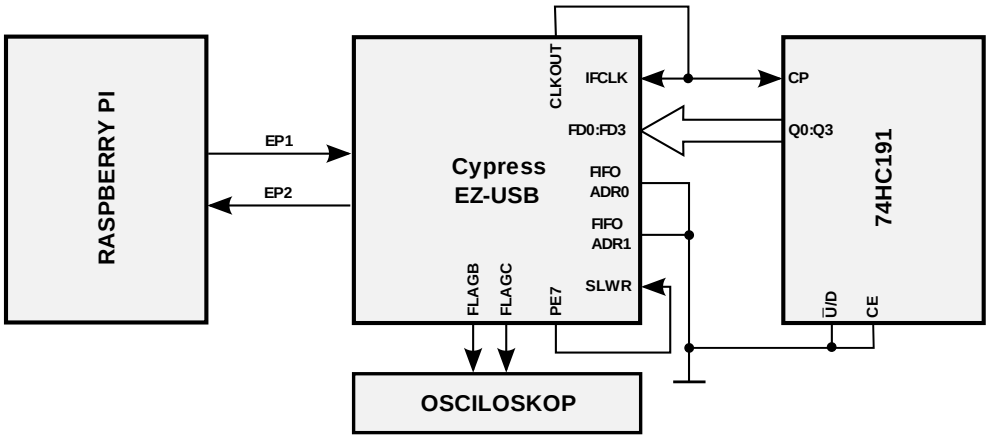
Vo firmware pribudne konfigurácia EP1OUT. Tento endpoint bude RPi využívať pre odoslanie žiadosti o dáta. Zmeny vo funkcií *TD_Init()* sú vykonané nastavením registra EP1OUTCFG a povolením prerušenia na tento endpoint.

```
EP1OUTCFG = 0xA0;   SYNCDELAY;  // typ bulk, OUT, 64 B
OEE = 0xC0;                     // povolí bránu E, bit 7 a bit 6 (LED)
EPIE |= bmBIT3;                 // povolí prerušenie na EP1OUT
```

Vo funkcií *ISR_Ep1out*, ktorá sa vyvolá ako prerušenie po prijatí dát cez EP1OUT, dekóduje procesor prijatý znak. Týmto spôsobom je možné s RPi pomocou EP1OUT zasielať rôzne znaky, a tak ovládať USB radič. Do riadenia komunikácie sa teda začne zapájať aj vstavaný mikroprocesor. Po dekódovaní znaku sa vymaže FIFO pamäť

a následne sa pinom procesora povolí zápis.

```
switch (EP1OUTBUF[0]){
  case 0x07:{
  IOE = 0x80; SYNCDELAY;  // PE7(SLWR)=1 (zakáz zápisu do FIFO)
  FIFORESET = 0x80; SYNCDELAY;        //reset FIFO pamäte
  FIFORESET = 0x02; SYNCDELAY;
  FIFORESET = 0x00; SYNCDELAY;
  IOE = 0xC0;              //rozsvieť LED
  IOE = 0x40;              //povolenie zápisu (SLWR = 0)
  cakaj(100000);
  IOE = 0x80;              //zhasnutie LED, zákaz zápisu
}}
```



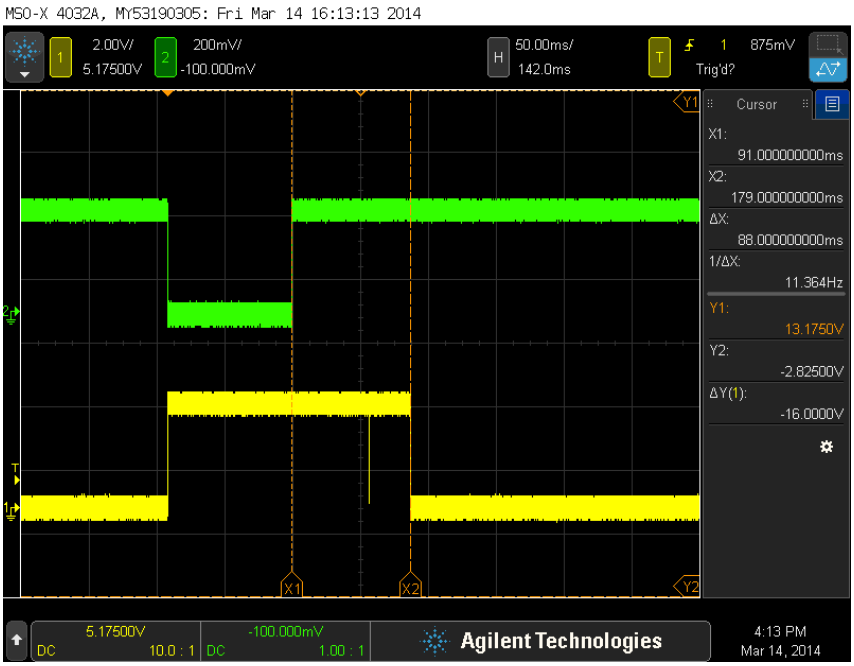
Obr. 16 Test s využitím čítača a mazaním FIFO pamäte

6.6 Dosiahnuté výsledky testu č. 3

Veľkosť dátového paketu a frekvencia hodín zostali nezmenené (2 MB, 24 MHz). Úpravou firmware a hardvérového zapojenia sa podarilo minimalizovať stratu dát (obr. 17), hlavne v počiatočnej fáze prenosu. Príznak EMPTY FLAG zobrazuje, že dáta sa začnú zapisovať do prázdnej FIFO pamäte. Pri tejto konfigurácií sa objavili aj dátové pakety, ktoré sa podarilo preniesť bezchybne (približne 1 bezchybný z 10 celkových).

V ďalšej fáze viedli úpravy firmware k navýšeniu endpointu z 512 B na 1024 B, a tým získanie 4 kB FIFO pamäte. Aj napriek minimálnym zásahom do pôvodného firmware sa ho nepodarilo uviesť do funkčného stavu.

Posledná možnosť, ktorú obvod ponúka je použitie 12 MHz (teda 12 MB/s pri 8-bitovej zbernici) hodinového signálu vyvedeného na pin CLKOUT. V tomto prípade RPi stíhalo čítať dáta z pamäte FIFO a nedochádzalo k ich strate (aj pri dátových paketoch väčších ako 2 MB). Taktiež boli na RPi spustené aj iné aplikácie (ako napr. internetový prehliadač, predinštalované hry) pre zaťaženie procesora, avšak na stratu dát to nemalo žiadny vplyv. Prenos sa teda dá považovať za spoľahlivý, preto na prvé pokusy s obrazovým senzorom bude využívaný 12 MHz hodinový signál.



Obr. 17 Príznaky EmptyFlag (horný zelený) a FullFlag (dolný žltý) s mazaním FIFO

7 OBRAZOVÝ SENZOR MICRON MT9M001

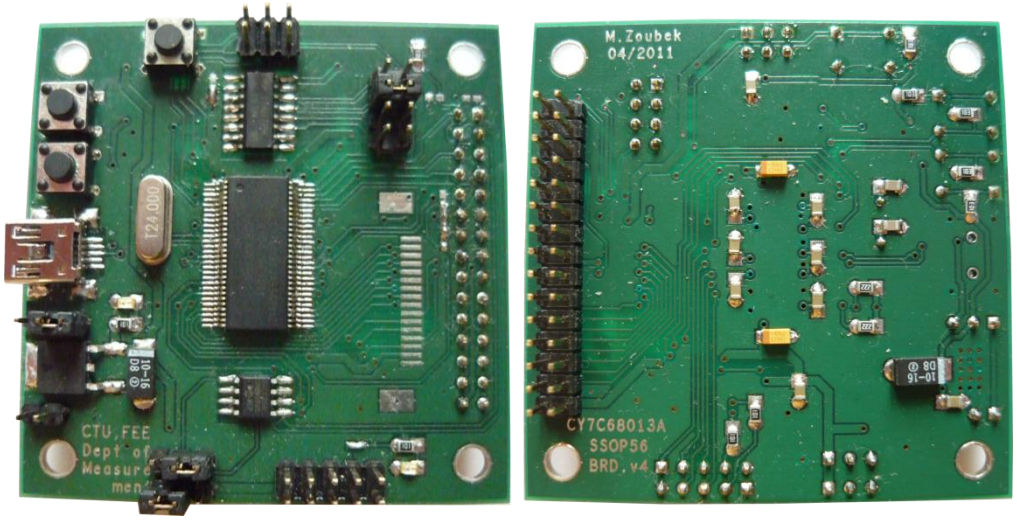
MT9M001 je monochromatický SXGA CMOS senzor so snímacou plochou veľkosti jedného palca [9]. Rozlíšenie senzora je 1312x1048 pixelov, pričom aktívnych je 1280x1024. V tomto rozlíšení je senzor schopný generovať 30 obrázkov za sekundu. Po pripojení hodinového signálu k senzoru začne pracovať v defaultnom režime, teda v plnom rozlíšení. Dôležité parametre senzora (teda registre), ako napr. rozlíšenie a počet snímok za sekundu sú nastaviteľné cez zbernicu I2C. Adresa senzora pre komunikáciu cez zbernicu I2C je pevná (0xBA), teda nemožno ju zmeniť. Výstupy senzora spolu s platnými dátovými bitmi začnú byť aktívne v momente, keď je pripojený hodinový signál (max. 48 MHz) na CLKIN a je povolený výstup senzora OE (aktívny v log. nule).

7.1 Doska plošného spoja pre CMOS senzor

Autorom návrhu dosky plošného spoja obrazového senzora je Ing. Jan Šedivý. Doska je univerzálna pre viacero CMOS senzorov od spoločnosti Micron. Informácie o osadení a oživení dosky uvádza Ing. Martin Zoubek vo svojej diplomovej práci [10].

7.2 Doska plošného spoja pre USB radič

Pre elegantné pripojenie CMOS senzora k USB radiču bola v rámci DP Ing. Zoubka [10] vyvinutá DPS pre USB radič (BRD.V4). V laboratórií videometrie mi boli doc. Fischerom poskytnuté súčiastky a vyrobená DPS. Túto dosku som osadil a zaspájkoval podľa postupu uvedeného v DP práci Ing. Zoubka (obr. 18).



Obr. 18 Osadená doska s USB radičom Cypress

Na doske sú umiestnené spájkovacie prepoje a mechanické prepoje (tzv. jumpre), ktorými doska poskytuje variabilitu konfigurácie (napr. voľba hodinového signálu pre senzor a režim FIFO slave, typ napájania a i.). V nasledujúcej časti bude popísaná konfigurácia dosky použitá v práci. Táto konfigurácia je dôležitá pre správnu funkčnosť s priloženým firmware a zavádzanie programu do pamäte EEPROM.

7.2.1 Nastavenie dosky pomocou spájkovacích prepojov PPx

Nastavenia dosky, ktoré sa v priebehu ďalšej práce nebudú výrazne meniť zvolil autor spájkovacie prepojky. Pin prepoja, ktorým sa začína číslovanie je na schéme osadenia súčiastok (obr. 19) označený modrým štvorčekom. U každého prepoja je v prvej odrážke uvedený význam a v druhej odrážke piny, ktoré treba prepojiť.

PP1 -nastavenie vstupného signálu deliča frekvencie 74AC161
-vstupným signálom je CLKOUT, prepojený pin 1 a 2

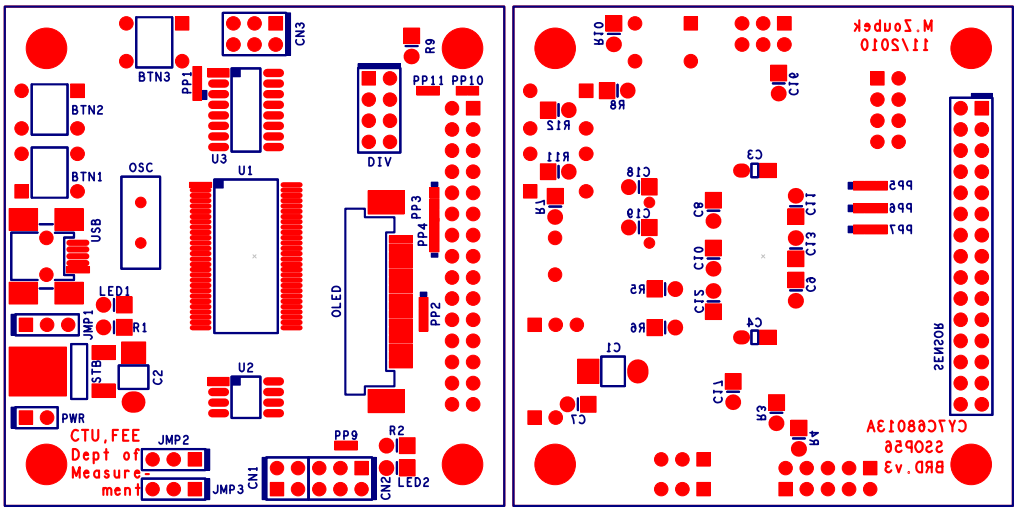
PP3 -voľba hodín pre zápis do pamäte FIFO
-vstupným hodinovým signálom je CLKOUT/X (generované deličom frekvencie), prepojený pin 2 a 3

PP4 -voľba hodín pre CMOS senzor
-vstupným hodinovým signálom je CLKOUT/X (generované deličom frekvencie), prepojený pin 2 a 3

PP9 -pripojenie indikačnej LED na pin procesoru 8051
-LED pripojená na pin PA0/INT0, prepojený pin 1 a 2

PP10 -signál STROBE CMOS senzoru
-signál nie je využívaný a pin PA0/INT0 je možné použiť pre iné účely, neprepojené piny

PP11 -signál SNAPSH CMOS senzoru
-signál nie je využívaný a pin PA1/INT1 je možné použiť pre iné účely, neprepojené piny



Obr. 19 Montážna schéma - vľavo TOP, vpravo BOTTOM (prevzaté z [10])

7.2.2 Nastavenie dosky pomocou jumprov JMPx

Typ napájania a povolenie zápisu do EEPROM môže byť nastavené pomocou jumperov. Pin jumpra, ktorým začína číslovanie má tvar štvorca (obr. 19). V prvej odrážke je význam jumpera a v druhej piny, ktoré treba prepojiť.

JMP1 -typ napájacieho zdroja
-napájanie z USB, prepojené piny 2 a 3

JMP2 -voľba adresy EEPROM
-pamäť EEPROM bude využívaná a program sa v nej začne po štarte

vyhľadávať (A2:A0 = 0b001), prepojené piny 1 a 2

JMP3 -ochrana zápisu do pamäte EEPROM
-do EEPROM je možné zapisovať, prepojené piny 2 a 3

7.3 Podložky pre uchytenie objektívu typu C/CS

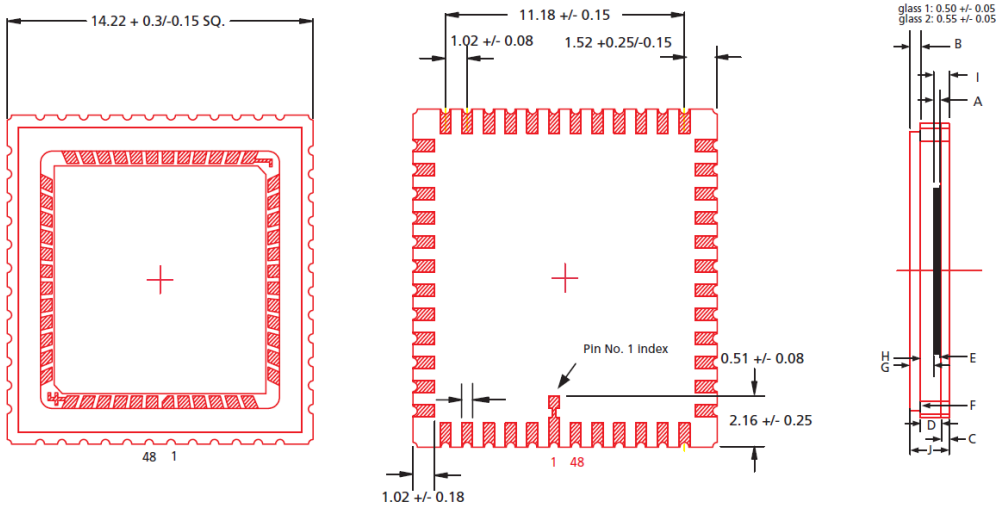
V Laboratóriu videometrie sa vo veľkej miere používajú dva typy výmenných objektívov. Tieto objektívy sa dodávajú so závitom C alebo CS a líšia sa vo vzdialenosti medzi zadnou šošovkou objektívu a aktívnou plochou snímacieho senzora. Pri objektíve typu C je táto vzdialenosť 17,526 mm a typu CS 12,5 mm. Použitím redukčného krúžku môže byť objektív s C závitom použitý na kameru s CS závitom. Objektív s CS závitom nemožno použiť na kameru s C závitom.

V Laboratóriu videometrie sa často používajú senzory Micron MT9M001. V súvislosti s touto prácou som vypočítal vzdialenosť pre objektív typu CS od aktívnej plochy tohto senzora. Z datasheetu senzora [9] je určíme vzdialenosť aktívnej plochy, ktorá je vystúpená od spodnej strany senzora (obr. 20, parameter I). Táto vzdialenosť sa v datasheete uvádza ako „Sensor array to seating plane“ a je 1,27 mm. Podložka a senzor sú umiestnené na rovnakej „sedacej ploche“, preto výšku podložky vypočítame vzťahom

$$h_P = I + C_V, \tag{7.3}$$

kde h_P je výška podložky, I je vzdialenosť aktívnej plochy senzora od sedacej plochy a C_V je vzdialenosť zadnej šošovky objektívu od aktívnej plochy senzora. Dosadením známych hodnôt do vzťahu 7.3 získame požadovanú výšku podložky

$$h_P = 1{,}27 + 12{,}5 = 13{,}77\ mm. \tag{7.4}$$



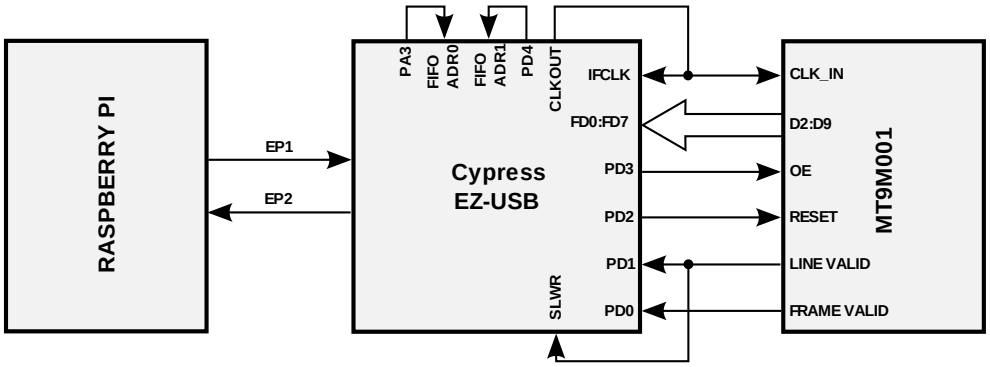
Obr. 20 Rozmery senzora Micron MT9M001; TOP, BOTTOM, SIDE (prevzaté z [9])

S vypočítanou výškou som oslovil rodinného príslušníka o výrobu tejto podložky. V spoločnosti Continental Matador Rubber bola žiadosť o výrobu vyhovená, a tak vzniklo 10 kusov podložiek z hliníkového materiálu pre potreby Laboratória videometrie. Podložku je teda možné použiť pre objektív typu CS a s využitím redukčného krúžku aj pre objektívy typu C (obr. 21).

Obr. 21 Podložka pod objektív, zľava TOP, BOTTOM, SIDE

7.4 Hardvérové zapojenie CMOS senzoru pripojeného k USB radiču

Na obr. č. 22 je zjednodušená bloková schéma zapojenia s dôležitými signálmi pre riadenie zápisu dát zo senzora. Vstupný hodinový signál pre senzor a zápis do pamäte FIFO je vyvedený z výstupu CLKOUT cez delič frekvencie 74AC161. Frekvencia signálu na výstupe CLKOUT je 48 MHz. Na tejto frekvencií teda pracuje aj samotný mikroprocesor 8051. Pri 48 MHz hodinovom signáli na vstupe deliča možno získať na jeho výstupoch frekvencie 24, 12, 6 a 3 MHz. Frekvenciu z deliča možno vybrať jumperom.



Obr. 22 Bloková schéma pripojenia senzora k USB radiču

Výstupy PA3 a PD4 slúžia pre adresovanie pamäte FIFO (teda endpointu), do ktorej sa budú zapisovať platné dáta. Zápis do FIFO (SLWR) je ovládaný výstupným signálom LINE VALID z CMOS senzora, ktorý je spolu so signálom FRAME VALID privedený aj na brány mikroprocesora. CMOS senzor poskytuje 10-bitové dáta, avšak k USB radiču je pripojených iba 8 horných dátových bitov.

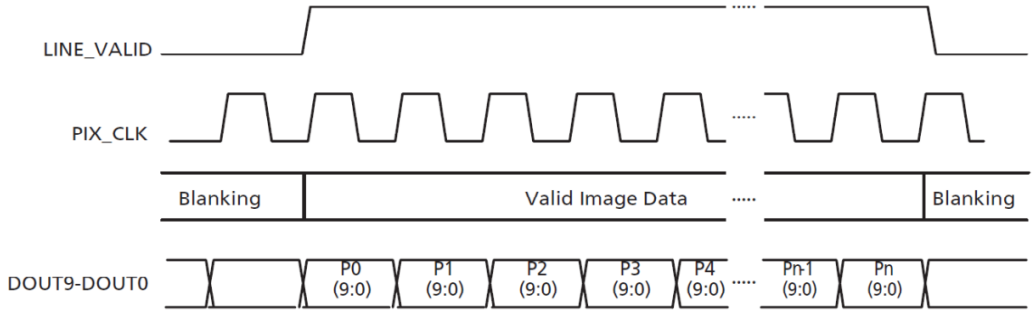
7.5 Synchronizácia pre odber jedného snímka

Obrazový senzor CMOS je schopný po privedení hodinového signálu pracovať v defaultnom režime. Pre odber snímka je nutné správne načasovanie pre odber dát. Pri

nesprávnom načasovaní dôjde k odberu dát práve snímaného obrazu. Tento nežiaduci jav sa prejaví pri vykresľovaní získaných dát, kde výsledný obrázok môže pozostávať z dvoch snímok. Dôležitú úlohu pri tejto synchronizácií zohrávajú signály FRAME VALID a LINE VALID.

Jednou z možností pre odber dát jedného platného snímka je využitie CPLD. Na to potrebujeme ďalšie zariadenie a znalosti k jeho programovaniu.

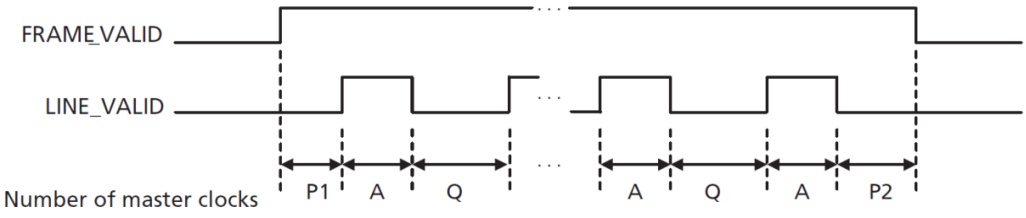
Jednoduchšia možnosť pre synchronizáciu je využitie vstavaného procesora. Najskôr si treba uvedomiť a poznať význam jednotlivých signálov. Začiatok nového snímka signalizuje zmena FRAME VALID z log. 0 do log. 1. Potom nasleduje vyčítanie prvého riadka, čo signalizuje zmena LINE VALID z log. 0 do log. 1. Platné dáta sú na výstupe senzora s každou nábežnou hranou hodín PIX_CLK (jeden pixel predstavuje 10-bitové dáta). Postupne sa vyčítajú dáta pre všetky pixely v riadku (obr. 23). Po vyčítaní prvého riadka (zmena LINE VALID z log 1 do log. 0) nasleduje horizontálne zatemnenie, teda doba medzi dvoma riadkami. Počas tejto doby nie sú na výstupe senzora platné dáta. Po zatemnení nasleduje vyčítanie druhého riadka a proces sa opakuje ako v prípade prvého riadka. Po vyčítaní dát posledného riadka je ukončený prenos jedného snímka, teda zmena FRAME VALID z log. 1 do log. 0.



Obr. 23 Vyčítanie dát jedného riadka CMOS senzora (prevzaté z [9])

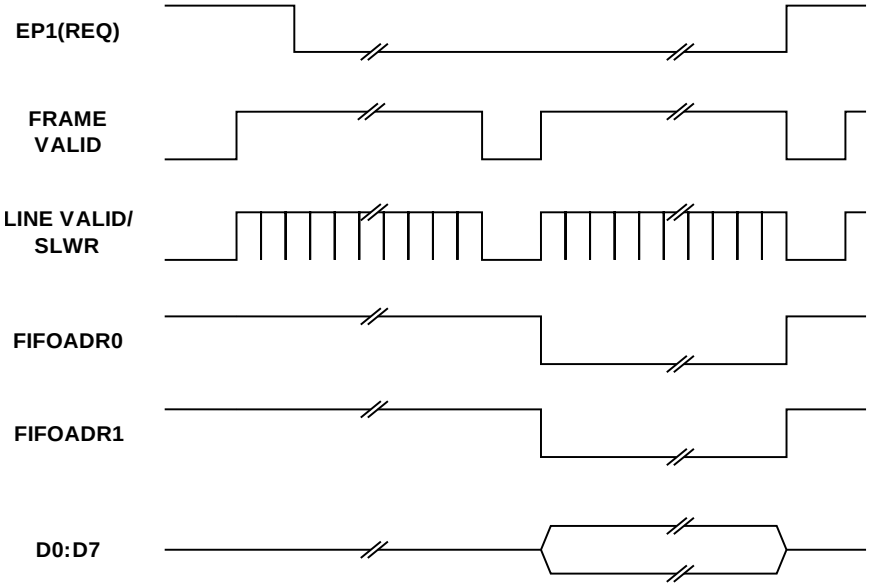
Samotná realizácia pre odber jedného snímka vyzerá nasledovne. Ako doprovod k textu slúži pre lepšie pochopenie obr. 24 a obr. 25. Výstupom procesora PD3 do log. 0 povolíme výstupy CMOS senzora (OUTPUT ENABLE). Žiadosť o dáta prichádza podobne, ako v prípade s čítačom po EP1OUT, čím sa vyvolá prerušenie na endpoint. V tejto fáze začne do prenosu zasahovať mikroprocesor 8051. Žiadosť môže prísť v čase, keď už sa sníma obrázok (FRAME VALID v log. 1), preto je nutné počkať na ďalší snímok. Počas doby čakania sa dáta zapisujú do nenakonfigurovanej pamäte (FIFOADR1 a FIFOADR0 v log.1, teda EP8). Môžeme teda povedať, že dáta sa zahadzujú. V momente, keď sa začne snímať ďalší obrázok (zmena stavu FRAME VALID z log. 0 do log. 1), je potrebné zapisovať dáta do nakonfigurovanej FIFO pamäte EP2. Výber pamäte EP2 je vykonaný zmenou FIFOADR1 a FIFOADR0 z log. 1 do log. 0. Signál LINE VALID je za signálom FRAME VALID oneskorený o dobu štartovacieho zatemnenia (doba P1, obr. 24). Táto doba trvá 242 hodinových cyklov, teda 20,16 µs (senzor s frekv. 12 MHz). To je dostatok času na vykonanie dvoch inštrukcií CLR (pre zmenu FIFOADR), ktoré spolu zaberú 4 hodinové cykly, teda 83,33 ns (procesor s frekvenciou 48 MHz).

Zápis do FIFO pamäte EP2 (SLWR) ovláda signál LINE VALID (zápis v log. 1). Tu už je pripravený buffer aplikácie na RPi prijímať dáta cez komunikačnú rúru. Počas prenosu dát jedného snímka nie je potrebné počítať počet prenesených riadkov. Stačí, ak budeme sledovať koniec snímka (FRAME VALID v log. 0).



Obr. 24 Časovanie FRAME/LINE VALID (prevzaté z [9])

Aplikácia na RPi nestihne za dobu medzi dvoma snímkami (doba vertikálneho zatemnenia; pri rozlíšení 1280x1024 2,54 ms; pri rozlíšení 640x480 1,47 ms) spracovať a vykresliť získané dáta. Preto po prenesení jedného snímka je nutné zakázať zápis do FIFO pamäte EP2 (zmena FIFOADR1 a FIFOADR0 do log. 1). Pre získanie dát ďalšieho snímka musí aplikácia na RPi opätovne zaslať žiadosť cez EP1OUT.



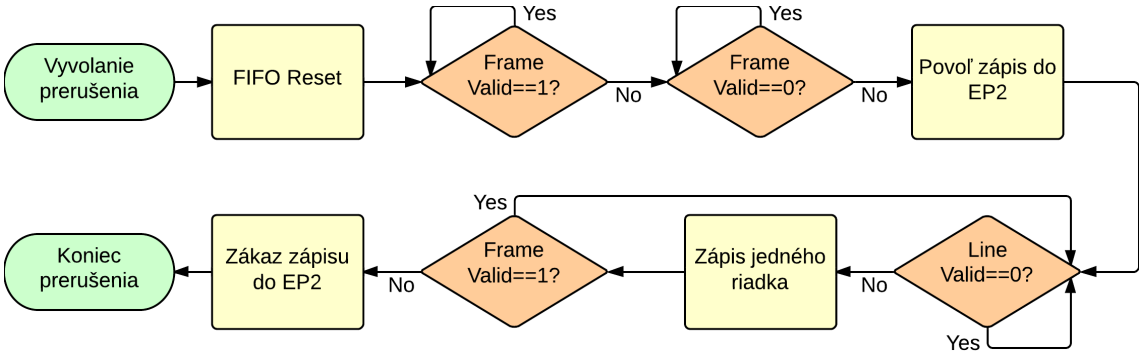
Obr. 25 Proces synchronizácie odberu jedného obrázka

7.6 Firmware so synchronizáciou pre odber snímka z CMOS senzora

Pre odber dát snímka z CMOS senzora je potrebné vykonať zmeny vo firmware USB radiča, ktoré sa riadia postupom uvedeným v kapitole 7.5. Úpravy vychádzajú z firmware použitého v kapitole 6.5 Hlavné zmeny sú uskutočnené v prerušení na EP1OUT, ktoré sa vyvolá po prijatí dát. Tieto úpravy v prerušení zobrazuje vývojový diagram na obr. 26.

```
case 0x07:{
BUS_B = 0x00;                    //povolenie brány B
SEN_OE = 0;                      //aktivácia výstupov senzoru
FIFORESET = 0x80; SYNCDELAY;     //reset FIFO pamäte
FIFORESET = 0x02; SYNCDELAY;
FIFORESET = 0x00; SYNCDELAY;

#pragma asm                      //začiatok programu v ASM
FRAME_VALID_1: JB CCD_PORT.0,FRAME_VALID_1
                                 //ak F_V=1, čakám až F_V=0
```

```
FRAME_VALID_0: JNB CCD_PORT.0,FRAME_VALID_0
                                //ak F_V=0, čakám až F_V=1
CLR FIFOADR0                    //povolenie zápisu
CLR FIFOADR1
FRAME_VALID:
LINE_VALID: JNB CCD_PORT.1,LINE_VALID       //čaká na L_V=1
JB CCD_PORT.0,FRAME_VALID       //ak F_V=1, skok na FRAME_VALID
SETB FIFOADR0                   //zákaz zápisu do FIFO EP2
SETB FIFOADR1
#pragma endasm                  // koniec programu v ASM
}}
```



Obr. 26 Vývojový diagram odberu dát jedného snímka

7.7 Nastavenie registrov senzora cez zbernicu I2C

Pomocou zbernice I2C je možné nastaviť parametre CMOS senzora, kde nadriadeným zariadením (master) je mikroprocesor USB radiča. Ako už bolo spomenuté v úvode kapitoly, adresa senzora je pevná (0xBA) a nemožno ju meniť. Parametre sa nastavujú zápisom hodnoty do 16-bitových registrov.

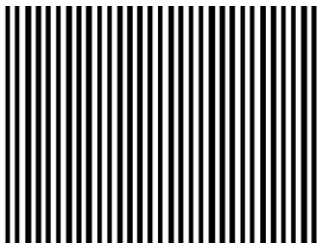
Pre častý zápis do jednotlivých registrov je vhodné vytvoriť funkciu. Funkcia má tri parametre, kde prvým parametrom je adresa registra, do ktorého budeme zapisovať, druhým resp. tretím parametrom je vyšší resp. nižší dátový bajt.

```
void sendI2C(unsigned char REGADR, unsigned char REGDATAH,
unsigned char REGDATAL)
{
I2CS |= bmSTART;                    //začiatok prenosu
I2DAT = adresaSenzor;               //odošle sa adresa senzoru
while((I2CS&(bmDONE))!=bmDONE);     //čaká na odoslanie

I2DAT = REGADR;                     //odošle sa adresa registra
while((I2CS&(bmDONE))!=bmDONE);     //čaká na odoslanie

I2DAT = REGDATAH;                   //odošle sa horný bajt
while((I2CS&(bmDONE))!=bmDONE);     //čaká na odoslanie

I2DAT = REGDATAL;                   //odošle sa dolný bajt
while((I2CS&(bmDONE)) !=bmDONE);    //čaká na odoslanie

I2CS |= bmSTOP;                     //ukončenie prenosu
while((I2CS&(bmSTOP))==bmSTOP);     //čaká na ukončenie
}
```

7.8 Generovanie testovacích dát CMOS senzorom

Defaultne CMOS senzor sníma obraz v rozlíšení 1280x1024. V tomto kroku nie je kladený dôraz na rýchlosť prenosu, ale na overenie funkčnosti prenosu. Senzor umožňuje generovanie testovacích dát vnútornou logikou a ich vyčítanie na dátovej zbernici. Prvý platný stĺpec bude obsahovať dáta testovacieho registra (register 0x32), druhý platný stĺpec bude obsahovať invertované testovacie dáta. Tretí stĺpec bude opäť neinvertovaný, štvrtý invertovaný, atď (obr. 27). Zápisom hodnoty 0xFFFF do registra 0x32 získame maximálny kontrast medzi párnym a nepárnym stĺpcom, teda čierne a biele vertikálne pruhy. Taktiež je nutné nastaviť šiesty bit registra 0x07 na hodnotu 1, čím povolíme na výstup testovacie dáta.

```
sendI2C(0x32, 0xFF, 0xFF);
sendI2C(0x07, 0x00, 0x42);
```



Obr. 27 Výrez testovacieho obrázka

7.9 Aplikácia na RPi

Aplikácia zabezpečujúca prenos dát vychádza zo štruktúry funkcií knižnice libusb (viď kapitola 5). Pre testovacie účely zobrazenia dát reálneho obrázka bolo zvolené ukladanie do súboru. Použitý kód vychádza zo stránok rossetacode.org [11]. Tento typ zobrazenia nevyžaduje inštaláciu ďalšej knižnice, čo je hlavnou výhodou. Má aj obrovskú nevýhodou, ktorou je nutnosť manuálne otvoriť každý snímok pre prezretie. To môže byť z užívateľského hľadiska veľmi nepríjemné. Reálny prenesený snímok pri rozlíšení 1280x1024 pixelov zobrazuje obr 28.

```
FILE *imageFile;
int cisloPixel, sirka, vyska;
imageFile=fopen("image.png","wb");
if(imageFile==NULL){
  perror("ERROR: Subor sa neda otvorit.");
  return(-1);}

fprintf(imageFile, "P5\n");
fprintf(imageFile, "%d %d\n", sirka, vyska);
fprintf(imageFile, "255\n");
for(x=0; x < vyska; x++){
        for(y=0; y < sirka; y++){
              pixel = (int)in_data_buf[cisloPixel];
              fputc(pixel,imageFile);
              cisloPixel++;
}}
fclose(imageFile);
```

Obr. 28 Reálny obrázok získaný z CMOS senzora

7.10 OpenCV pre kontinuálne zobrazenie

OpenCV je multiplatformová knižnica a slúži prevažne pre spracovanie obrazu v reálnom čase a prácu s ním. Je písaná v jazyku C/C++ a nachádza využitie v rôznych aplikáciách ako napr. rozpoznávanie tváre, identifikáciu objektov, sledovanie pohybu a mnohých ďalších.

V tejto práci bude knižnica OpenCV slúžiť pre zobrazenie získaných dát zo senzora v okne programu. Tým získame jednoduchú variantu kontinuálneho zobrazovania získaných snímok.

Knižnica je voľne dostupná a pre inštaláciu verzie OpenCV 2.4.5 a OpenCV 2.4.6.1 bol použitý skript zo stránok Jay Rambhia [12]. Inštalácia na RPi je pomerne zdĺhavá a vyžaduje pripojenie k internetu. Skript, ktorý vykoná kompletnú inštaláciu spustíme príkazmi

```
> chmod +x opencv2_4_5.sh
> ./opencv2_4_5.sh
> chmod +x opencv2_4_6_1.sh
> ./opencv2_4_6_1.sh
```

Po inštalácií knižnice je vhodné previesť aktualizáciu systém

```
>sudo apt-get update
>sudo apt-get upgrade
```

7.10.1 Dôležité funkcie knižnice OpenCV pre zobrazenie snímka

Knižnica ponúka nespočetné množstvo funkcií pre prácu s obrazom (čerpané z [13]). V nasledujúcej časti budú vypísané len tie najnutnejšie, a to pre prevedenie dát zo

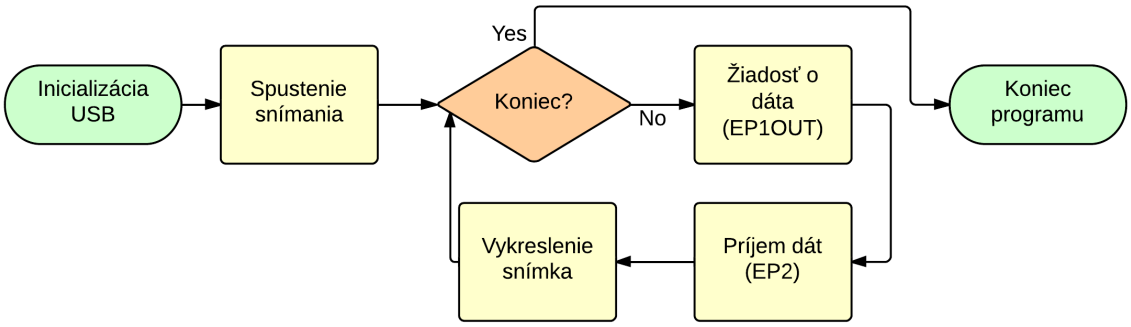
senzora na požadovaný formát, vytvorenie okna pre zobrazenie a vykreslenie v tomto okne.

Taktiež bude uvedená funkcia, ktorej volaním môžeme uložiť obrázok do súboru. Funkcia je uvedená najmä pre svoju jednoduchosť a porovnanie s kódom uvedeným v kapitole 7.9.

```
Mat mat(int rows, int cols, int type, void* data, size_t
step=AUTO_STEP)
            //z buffra dát vytvorí maticu pre ďalšiu prácu s dátami

void namedWindow(const string* winname, int flags=WINDOW_NORMAL)
            //vytvorí nové okno

void imshow(const string& winname, InputArray mat)
            //zobrazí obrázok v okne

void destroyWindow(const string& winname)
            //zatvorí okno
```

7.11 Aplikácia pre kontinuálne zobrazenie

Pre vytvorenie kamery snímajúcej okolitú scénu musíme neustále získavať nové dáta zo senzora. Navrhnuté riešenie synchronizácie dát zo senzora (podkapitola 7.6) zaistí správny odber dát len pre jeden snímok. Preto ak chceme zobrazovať snímky neustále za sebou, musíme o každý snímok zažiadať cez EP1OUT. Po prijatí dát jedného snímka cez EP2 dáta spracujeme a vykreslíme funkciami knižnice OpenCV (podkapitola 7.10). Umiestnením týchto troch funkcií do cyklu *while* (obr. 29) získame veľmi jednoduchú kameru s kontinuálnym zobrazovaním snímaného obrazu.



Obr. 29 Vývojový diagram kontinuálneho snímania

7.12 Testovanie snímkovej frekvencie

Kontinuálne zobrazenie úspešne funguje, preto nás zaujíma rýchlosť zobrazovania. Pri zobrazovacích zariadeniach je dôležitým údajom snímková frekvencia. Táto frekvencia sa udáva v jednotkách fps (z angl. Frames Per Second) a odpovedá počtu snímok za sekundu. Test sa vykonával priemerovaním celkového času potrebného na prenos úspešne prenesených snímok, teda

$$FPS_N = {}^{t}/_{n}\,, \tag{7.11}$$

kde t je celkový čas pre spracovanie a n je počet úspešne prenesených snímok. Pri všetkých testoch bude n=1000.

Taktiež dokážeme určiť snímkovú frekvenciu obrazového senzora CMOS. Môžeme potom určiť, či je RPi schopné čítať všetky snímky, ktoré generuje obrazový senzor. Výpočet snímkovej frekvencie obrazového CMOS senzora vychádza z počtu hodinových cyklov F_C potrebných pre jeden snímok [9]

$$F_c = (n_r + b_v) * (n_s + F_{sb} + F_{eb}), \tag{7.12}$$

kde F_c je počet hodinových cyklov jedného obrázka, n_r je počet riadkov (teda výška snímka), b_v je vertikálne zatemnenie, n_s je počet stĺpcov (teda šírka snímka), F_{sb} je hodnota horizontálneho zatemnenia na začiatku snímka (Frame Start Blanking) a F_{eb} je hodnota horizontálneho zatemnenia na konci snímka (Frame End Blanking). Všetky hodnoty okrem F_{sb} je možné nastaviť registrami obrazového CMOS senzora. Stále však pracujeme so senzorom v defaultnej konfigurácií, teda v rozlíšení 1280x1024 pixelov. Ostatné hodnoty možno nájsť v datasheete senzora [9]. Dosadením do 7.12 dostaneme

$$F_c = (1024 + 20) * (1280 + 242 + 2) = 1\,600\,200\ hod.cyklov/snímok \tag{7.13}$$

Snímkovú frekvenciu senzora FPS_S vypočítame vzťahom

$$FPS_S = {}^{f}/_{F_c}, \tag{7.14}$$

kde f je frekvencia hodinového signálu a F_C počet hodinových cyklov na vyčítanie jedného snímka. Obrazový CMOS senzor je pripojený na 12 MHz hodinový signál z deliča frekvencie (obr. 20). Snímkovú frekvenciu senzora FPS_S vypočítame dosadením frekvencie f a počtu hodinových cyklov F_C do 7.14, teda

$$FPS_S = {}^{12 * 10^6}/_{1\,600\,200} = 7{,}5\ fps \tag{7.15}$$

Testovanie snímkovej frekvencie bolo vykonané so zobrazovaním aj bez zobrazovania snímok v programovom okne. Výsledky merania zobrazuje tab. 4. Dosiahnuté výsledky hovoria, že RPi dokáže čítať dáta rovnakou rýchlosťou, akou ich generuje senzor ak snímky nezobrazuje v programovom okne. Pri zobrazovaní snímok je vyťažovaný procesor a pamäť RAM RPi. Procesorový čas pre spracovanie dát a vykreslenie snímka je dlhší ako doba vertikálneho zatemnenie, dokonca dlhší ako čas ďalších troch snímok generovaných senzorom. Až v priebehu tejto doby zašle RPi žiadosť o ďalší snímok cez EP1OUT. V plnom rozlíšení obrazového senzora s 12 MHz hodinovým signálom je RPi schopné čítať približne každý štvrtý snímok.

Pri zápise dát zo senzora do FIFO pamäte je zápis (SLWR) ovládaný signálom LINE VALID. Preto by som chcel pripomenúť test č. 3 (podkapitola 6.4 až 6.6), kde bol prenášaný dátový paket veľkosti 2 MB s 24 MHz hodinovým signálom. V teste nebol zápis do FIFO pamäte prerušovaný, preto sa dátový paket zapisoval ako celok. Aj

napriek tomu sa podarilo niektoré dátové pakety preniesť bez straty. S rozlíšením senzora 1280x1024 má dátový paket veľkosť približne 1,3 MB a navyše po každom riadku (1280 B) nasleduje horizontálne zatemnenie (približne 10,16 μs pri 24 MHz). Počas tohto horizontálneho zatemnenia po každom riadku sa dáta do FIFO pamäte nezapisujú. Práve naopak, FIFO má možnosť sa čo najviac vyprázdniť. Preto som sa rozhodol otestovať prenos dát zo senzora aj s 24 MHz hodinovým signálom. Test ukázal, že prenos je možný avšak vyskytovali sa aj chybné snímky (v priemere každá dvadsiata snímka chybná). Z tohto dôvodu nemožno stanoviť presnú snímkovaciu frekvenciu, ktorú som počítal len s úspešne prenesených snímok.

Tab. 4 Výsledky testu snímkovej frekvencie na RPi (1280x1024)

1280x1024	12 MHz; FPS_S=7,5	
	FPS_N	FPS_S-FPS_N
so zobrazením	1,77	5,73
bez zobrazenia	7,50	0,00

Na úkor rozlíšenia senzora je možné zvýšiť rýchlosť prenosu. Upravíme rozlíšenie na VGA, teda 640x480 pixelov. Zmenu vykonáme cez zbernicu I2C úpravou hodnoty registra 0x03 a 0x04.

```
sendI2C(0x03, 0x01, 0xDF);      //počet riadkov - 1
sendI2C(0x04, 0x02, 0x7F);      //počet stípcov - 1
```

V rozlíšení 640x480 pixelov bol vykonaný rovnaký test s 12 a 24 MHz hodinovým signálom. Hodnoty FPS_S pre rozlíšenie 640x480 pixelov a hodinové signály 12 a 24 MHz vypočítame rovnako dosadením do 7.12 a 7.14. Počet hodinových cyklov F_C je v oboch prípadoch rovnaký, teda

$$F_c = (480 + 20) * (640 + 242 + 2) = 442\,000\ hod.cyklov/snímok. \tag{7.16}$$

Snímková frekvencia FPS_S s 12 MHz hodinovým signálom je

$$FPS_S = {12 * 10^6}/{442\,000} = 27{,}15\ fps \tag{7.17}$$

a s 24 MHz hodinovým signálom

$$FPS_S = {24 * 10^6}/{442\,000} = 54{,}30\ fps. \tag{7.18}$$

Tab. 5 Výsledky testu snímkovej frekvencie na RPi (640x480)

640x480	12 MHz; FPS_S=27,15		24 MHz; FPS_S=54,30	
	FPS_N	FPS_S-FPS_N	FPS_N	FPS_S-FPS_N
so zobrazením	12,62	14,53	12,83	41,47
bez zobrazenia	12,92	14,23	25,86	28,44

Výsledky testu zobrazuje tab. 5. Rýchlosť snímkovej frekvencie pri tomto rozlíšení sa podarilo zvýšiť približne 7-krát v porovnaní s maximálnym rozlíšením. Prenos dát zo senzor spoľahlivo fungoval aj s 24 MHz hodinovým signálom. Z tab.5 vidieť, že zvýšením frekvencie na 24 MHz nemožno zvýšiť počet snímok za sekundu so zapnutým zobrazovaním v okne programu. S touto frekvenciou je RPi schopné čítať až 25,86 fps pri vypnutom zobrazovaní. Z toho jasne vidieť ako spracovanie dát a ich vykreslenie ovplyvňuje výkon procesora a pamäte RAM RPi. RPi nedokáže ani pri jednom z hodinových signálov prijímať snímky tou rýchlosťou, ktorou ich generuje obrazový senzor. Dokáže prijímať približne každý druhý snímok. To je spôsobené zmenou rozlíšenia, a teda zmenšením doby vertikálneho zatemnenia. Počas tohto zatemnenia nestihne RPi zaslať žiadosť o nový snímok cez EP1OUT. Žiadosť je USB radičom prijatá v čase snímania nového obrázka, preto sa čaká na jeho ukončenie. Až potom sa začne prenos ďalšieho snímka (viď obr. 25). Procesor RPi sa teda nevyužíva nie po dobu vertikálneho zatemnenia, ale po dobu celého snímka.

ZÁVER

Cieľom práce bolo navrhnúť rýchly vstup dát z obrazového senzora do RPi. Spoločnosť RPi Fundation ponúka k RPi 5 Mpix senzor, ktorý sa pripája pomocou rozhrania CSI. My sme sa v práci orientovali na využitie zbernice USB s využitím USB radiča CY7C68013A od spoločnosti Cypress. Spoločnosť pre obvod neposkytuje knižnicu, ktorá by bola schopná komunikovať s linuxovými operačnými systémami. Napriek tomu sa túto komunikáciu podarilo vytvoriť s využitím voľne dostupnej knižnice libusb.

Pri testovaní rýchlosti a spoľahlivosti USB prenosu medzi obvodom Cypress a hostiteľským radičom (radičom RPi) boli zistené dôležite poznatky. Pri rýchlosti 30 MB/s je RPi schopné nepretržite prijímať dátové pakety do veľkosti 64 kB. Trvalý a spoľahlivý prísun dát do RPi je možný pri rýchlosti 12 MB/s. V oboch prípadoch sa využíva len polovica vyrovnávacej FIFO pamäte, teda 2 kB. Pri maximálnom využití 4 kB pamäte by tieto čísla mohli vzrásť. Mne sa firmware s plným využitím FIFO pamäte nepodarilo uviesť do funkčného stavu.

Ďalej sa podarilo navrhnúť a implementovať metódu pre načasovanie odberu dát z obrazového senzora CMOS. K riadeniu tejto synchronizácie nie je potrebné použiť CPLD, ale využíva sa vstavaný mikroprocesor 8051 USB radiča. Následne bolo vytvorené jednorazové zobrazenie resp. uloženie snímka do súboru, ktoré pre svoju funkčnosť nepotrebuje ďalšie knižnice. Pre zobrazenie snímok v programovom okne bola použitá voľne dostupná knižnica OpenCV, čo viedlo k vytvoreniu kontinuálneho zobrazovania. Získali sme tak základ videokamery. Malými zmenami v hardvérovom zapojení dokážeme k USB radiču (zároveň aj k RPi) pripojiť akýkoľvek obrazový senzor s paralelným výstupom.

S využitím 12 a 24 MHz hodinového signálu pre obrazový CMOS senzor a FIFO slave mód boli vykonané testy snímkovej frekvencie. Pri maximálnom rozlíšení senzora (1280x1024 pixelov) s 12 MHz hodinovým signálom senzor generuje 7,5 fps a RPi dokáže rovnakou rýchlosťou dáta (snímky) čítať. Táto rýchlosť neplatí pri zobrazovaní snímok v programovom okne. So zapnutým zobrazovaním je RPi výkonovo schopné spracovať a vykresliť 1,77 fps. V nižšom VGA rozlíšení je RPi schopné čítať 25,86 fps, teda približne každý druhý snímok generovaný obrazovým senzorom. S týmto rozlíšením a zapnutým zobrazovaním je RPi výkonovo schopné spracovať a vykresliť 12,83 fps. Z toho vidieť, že zobrazovanie snímok knižnicou OpenCV v programovom okne má výrazný vplyv na výkon RPi, teda snímkovaciu frekvenciu.

Ďalším dôležitým záverom práce je, že práca môže slúžiť ako návod komunikácie s USB radičom pre ľubovoľnú linuxovú platformu a to vďaka použitej knižnici libusb. Preto som sa rozhodol vykonať testy z podkapitoly 7.11 na staršom notebooku s operačným systémom Linux Mint 13 Cinnamon, 1,73 GHz procesorom Intel Celeron a 2 GB pamäte RAM. Výsledky testu s rozlíšením senzora 1280x1024 pixelov zobrazuje tab. 6 a s rozlíšením 640x480 tab. 7. Pri maximálnom rozlíšení obrazového senzora s vypnutým zobrazovaním dokáže notebook čítať dáta rovnakou rýchlosťou ako ich generuje obrazový senzor. S týmto rozlíšením a zapnutým zobrazovaním dokáže spracovať a vykresliť každý druhý snímok. Maximálna snímková frekvencia so zobrazovaním snímok v programovom okne je 7,5 fps. Spôsobené je to tým, že nedokáže vykresliť obrázok v čase vertikálneho zatemnenia. S rozlíšením senzora 640x480 pixelov dokáže notebook spracovať a vykresliť až 25,83 fps. S týmto rozlíšením sa skráti čas

vertikálneho zatemnenia a napriek vypnutému zobrazovaniu sa nestihne odoslať žiadosť o nový snímok.

Tab. 6 Výsledky testu snímkovej frekvencie na notebooku (1280x1024)

1280x1024	12 MHz; FPS_S=7,5		24 MHz; FPS_S=15,00	
	FPS_N	FPS_S-FPS_N	FPS_N	FPS_S-FPS_N
so zobrazením	3,75	3,75	7,50	7,50
bez zobrazenia	7,50	0,00	15,00	0,00

Tab. 7 Výsledky testu snímkovej frekvencie na notebooku (640x480)

640x480	12 MHz; FPS_S=27,15		24 MHz; FPS_S=54,30	
	FPS_N	FPS_S-FPS_N	FPS_N	FPS_S-FPS_N
so zobrazením	12,91	14,24	25,83	28,47
bez zobrazenia	12,93	14,22	25,86	28,44

Moje ciele do budúcna budú viesť k úpravám firmware tak, aby USB radič využíval celú FIFO pamäť a nájsť spôsob vykresľovania snímok, ktorý by menej vyťažoval výkon RPi. Taktiež by som chcel upraviť aplikáciu tak, aby sekvenciu príkazov „žiadosť o dáta, príjem dát, vykreslenie snímka“ optimalizovala (napr. využitím viacerých vlákien a endpointov pre prenos dát).

Pri znalostiach nadobudnutých v tejto práci plánujem k USB radiču pripojiť A/D prevodník a tak vytvoriť lacný osciloskop pre domáce použitie.

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM SYMBOLOV A SKRATIEK

B	Jednotka množstva dát, 1 B = 8 bitov
RAM	Z angl. „*Random Access Memory*“, teda pamäť s náhodným prístupom
CMOS	Z angl. „*Complementary Metal-Oxid-Semiconductor*“, teda doplňujúci sa kov-oxid-polovodič; technológia výroby polovodičových súčiastok
HD	Z angl. „*High Definition*“, teda vysoké rozlíšenie
fps	Z angl. „Frames Per Second“, teda počet snímok za sekundu
CSI	Z angl. „*Camera Serial Interface*“, teda sériové rozhranie pre pripájanie kamier
PC	Z angl. „Personal Computer“, teda osobný počítač
USB	Z angl. „Universal Serial Bus“, sériová zbernica pre pripájanie periférií k počítačom
TV	Z angl. „*Television*“, teda televízor, televízia
Hz	Jednotka frekvencie v sústave SI
GPU	Z angl. „*Graphic Processing Unit*“, teda grafický procesor
SD	Z angl. „*Secure Digital*“, typ pamäťovej karty
GPIO	Z angl. „*General-Purpose Input/Output*“, programovateľné vstupno-výstupné piny obvodu
USART	Z angl. „*Universal Synchronous/Asynchronous Receiver and Transmitter*“, sériová komunikácia
I2C	Z angl. „*Inter-Integrated Circuit*“, multi-masterová počítačová sériová zbernica
SPI	Z angl. „*Serial Peripheral Interface*“, teda sériové periférne rozhranie
DC	Z angl. „*Direct Current*“, jednosmerný prúd
HDMI	Z angl. „*High-Definition Multimedia Interface*“, audiovizuálne rozhranie
DVI	Z angl. „*Digital Visual Interface*“, rozhranie pre vidozariadenia
SCART	Z franc. „*Syndicat des Constructeurs d’Appareils Radiorécepteurs et Téléviseurs*“), 21 pinové audiovizuálne rozhranie
RCA	Z angl. „*Radio Corporation of America*“, prevodník z 3,5 mm audio jacku na dva cinche
VGA	Z angl. „*Video Graphics Array*“, starší video štandard pre zobrazovaciu techniku
OS	Operačný systém
SDHC	Z angl. „*Secure Digital High Capacity*“, nástupca výrobnej technológie SD
FIFO	Z angl. „*First In, First Out*“, spôsob zápisu a čítania dát z pamäte; dáta

zapísané do pamäte ako prvé sa aj prvé čítajú

CRC Z angl. „*Cyclic Redundancy Check*“, funkcia pre detekciu chýb počas prenosu

ROM Z angl. „*Read-Only Memory*“,typ pamäte určenej len pre čítanie z nej

VID Z angl. „*Vendor ID*“, kód označujúci spoločnosť v špecifikácií USB (určuje USB organizácia)

PID Z angl. „*Product ID*“, kód označujúci produkt spoločnosti v špecifikácií USB (určuje výrobca)

EEPROM Z angl. „*Electrically Erasable Programmable Read Only Memory*“, elektricky zmazateľná pamäť ROM

CPLD Z angl. „*Complex Programmable Logic Device*“, teda programovateľné logické zariadenie

OpenCV Z angl. „*Open Source Computer Vision*“, knižnica počítačové spracovanie obrazu

LITERATÚRA

[1] Using the Raspberry Pi in the primary classroom. ScratchMyPi [online]. [cit. 2014-05-06]. Dostupné z: http://www.scratchmypi.co.uk/courses/using-the-raspberry-pi-in-the-primary-classroom/

[2] Raspberry Pi [online]. [cit. 2014-05-06]. Dostupné z: http://www.raspberrypi.org/

[3] INSTALLING OPERATING SYSTEM IMAGES. Raspberry Pi [online]. [cit. 2014-05-06]. Dostupné z: http://www.raspberrypi.org/documentation/installation/installing-images/README.md

[4] Universal Serial Bus Specification, Revision 2.0. April 27, 2000. Dostupné z http://www.usb.org

[5] EZ-USB FX2LPt USB Microcontroller, Cypress Semiconductor, Document # 38-08032. Dostupné z http://www.cypress.com

[6] EZ-USB Technical Reference Manual, Cypress Semiconductor, Document # 001-13670 Rev. *D. Dostupné z http://www.cypress.com

[7] Libusb-1.0 API Reference [online]. [cit. 2014-05-06]. Dostupné z: http://libusb.sourceforge.net/api-1.0/

[8] PHILIPS SEMICONDUCTORS. 74HC/HCT191: Presettable Synchronous 4-bit Counter [online]. December 1990 [cit. 2014-05-18]. Dostupné z: https://www.futurlec.com/74HC/74HC191pr.shtml

[9] ½-INCH CMOS ACTIVE-PIXEL DIGITAL IMAGE SENSOR, MT9M001, Micron Technology. Dostupné z http://www.micron.com

[10] ZOUBEK, Martin. Určování polohy pomocí řádkových obrazů. Praha, 2011. Bakalárska práca. ČVUT, Fakulta elektrotechnická, Katedra meraní. Vedúci práce doc. Ing. Jan Fischer, CSc.

[11] Bitmap/Write a PPM file. Rosetta Code [online]. [cit. 2014-05-06]. Dostupné z: http://rosettacode.org/wiki/Bitmap/Write_a_PPM_file#C

[12] RAMBHIA, Jay. Jay Rambhia main page [online]. [cit. 2014-05-06]. Dostupné z: https://github.com/jayrambhia

[13] Open Source Computer Vision main page [online]. [cit. 2014-05-06]. Dostupné z: http://www.opencv.org

OBSAH PRILOŽENÉHO DVD

Priložené DVD obsahuje zložky:

Aplikácie -aplikácie pre RPi

-*Snimok_do_suboru* -aplikácia, ktorá ukladá snímky do súboru; nevyžaduje inštaláciu OpenCV

-*Zobraz_OpenCV* -aplikácia pre kontinuálne zobrazenie; nutná inštalácia OpenCV

Datasheets -použité datasheety

Firmware -firmware obvodu Cypress pre prenos snímok

Obrázky -obrázky použité v práci plus ďalšie

Práca -táto práca vo formáte *.docx* a *.pdf*

Schéma -kompletná schéma zapojenia obvodu CY7C68013A so senzorom MT9M001

Skripty -skripty pre inštaláciu knižnice OpenCV na OS linux

Software -použitý softvér

-*Wheezy-Raspbian* -OS pre Raspberry Pi

-*CySuiteUSB_3_4_7* -programový balíček pre OS Windows obvodu Cypress; po nainštalovaní obsahuje driver obvodu Cypress, program *CyConsole* a program *Hex2bix*

-*Win32DiskImager* -program pre nahratie *.iso* súboru na pamäťovú SD kartu

ZOZNAM PRÍLOH

A SYSTÉMOVÝ OVLÁDAČ USB RADIČA

A.1 Zmeny v súbore cyusb.inf

Vykonané zmeny sú vyznačené **tučným** písmom:

```
;for x64 platforms
[Device.NTamd64]
%VID_04B4&PID_8613.DeviceDesc%=CyUsb, USB\VID_04B4&PID_8613
%VID_04B4&PID_1004.DeviceDesc%=CyUsb, USB\VID_04B4&PID_1004

[Strings]
CYUSB_Provider    = "Cypress"
CYUSB_Company     = "Cypress Semiconductor Corporation"
CYUSB_Description = "Cypress Generic USB Driver"
CYUSB_DisplayName = "Cypress USB Generic"
CYUSB_Install     = "Cypress CYUSB Driver Installation Disk"
VID_04B4&PID_8613.DeviceDesc="Cypress     USB     Generic     Driver
(3.4.7.000)"
VID_04B4&PID_1004.DeviceDesc="Bulkloop Device"
CYUSB.GUID="{AE18AA60-7F6A-11d4-97DD-00010229B959}"
CYUSB_Unused      = "."
```

B FOTOGRAFIE

B.1 Meracia kamera (pohľad predný)

B.2 Meracia kamera (pohľad bočný)