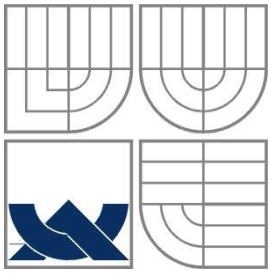
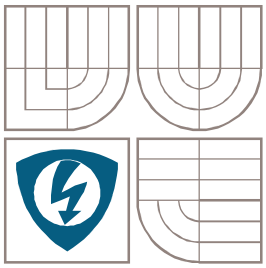


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

KOMUNIKACE OBVODU FPGA S POČÍTAČEM - USB HIGH SPEED

COMMUNICATION BETWEEN FPGA AND COMPUTER - USB HIGH SPEED

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Tomáš Kollár
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Jaromír Kolouch, CSc.
SUPERVISOR

BRNO, 2008

Abstrakt
Bakalařská práce se zaměřuje na možnost propojení čipů FPGA s PC pomocí sběrnice USB 2.0 v režimu high-speed. Tyto programovatelné obvody jsou velice důležité v dnešní době, hlavně kvůli jejich rychlosti a flexibilitě. Jejich vlastnosti se dají využít pro různá aplikace, která využívají malé datové toky, nebo i dosti veliké, které mohou využít celý přenosový potenciál USB 2.0 zběrnice. Práce se zaobírá s možnosti a stručným náčrtem využití takového přenosu a následně návrhem desky plošného spoje, umožňující využít těchto vysokých přenosových rychlostí. Pro komunikaci FPGA s počítačem se používá mikrokontrolér, který má vhodné vybavení pro tuto činnost a v důsledku toho se v této práci podrobněji opíšou její bloky a funkce. Na konci práce je rozebrána komunikace přes rozhraní GPIF, která odpovídá svou rychlostí požadavkům režimu high-speed.

Abstract
This bachelors thesis is about a possible interconnection method of FPGA (Field-Programmable Gate Array) chips with the PC through the USB 2.0 bus. These programmable arrays are very important nowadays mainly because of their high speed and high flexibility. Such properties make possible to use FPGAs in applications that require small data bandwidth and also which can utilize the whole bandwidth offered by the USB 2.0 bus. This thesis deals with possibility to use such transfer rate and its short description followed by the design of PCB layout. As the communication between the FPGA and the PC is made possible with the help of a microcontroller capable of handling such requests, I decided to include a short description of its functionality and working principle description in the thesis. The last part describes the communication through the GPIF interface for high-speed transfers.

Klíčová slova
FPGA, USB 2.0, popis CY7C68013A/CY7C68014A a CY7C68015A/CY7C68016A, mikroprocesor 8051, endpoint RAM, JTAG, GPIF rozhraní.

Keywords
FPGA, USB 2.0, characterization of CY7C68013A/CY7C68014A and CY7C68015A/CY7C68016A, 8051 microprocessor, endpoint RAM, JTAG, GPIF interface.

KOLLÁR, T. *Komunikace obvodu FPGA s počítačem - USB High Speed*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 49 s. Vedoucí bakalářské práce doc. Ing. Jaromír Kolouch, CSc.

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan

Jméno a příjmení:	Tomáš Kollár
Bytem:	Bešeňov č.d. 605, 941 41
Narozen/a (datum a místo):	12.07.1985, Nové Zámky

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1

Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   ☐ disertační práce
   ☐ diplomová práce
   ☒ bakalářská práce
   ☐ jiná práce, jejíž druh je specifikován jako .....
   (dále jen VŠKP nebo dílo)

Název VŠKP:	Komunikace obvodu FPGA s počítačem - USB High Speed
Vedoucí/ školitel VŠKP:	doc. Ing. Jaromír Kolouch, CSc.
Ústav:	Ústav radioelektroniky
Datum obhajoby VŠKP:	_____

   VŠKP odevzdal autor nabyvateli*:

   ☒ v tištěné formě – počet exemplářů: 2
   ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

* hodící se zaškrtněte

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 6. června 2008

……..	……
Nabyvatel	Autor

Obsah

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor
Elektronika a sdělovací technika

Student: Kollár Tomáš
Ročník: 3

ID: 83627
Akademický rok: 2007/08

NÁZEV TÉMATU:

Komunikace obvodu FPGA s počítačem - USB High Speed

POKYNY PRO VYPRACOVÁNÍ:

Na základě studie o komunikaci počítače PC prostřednictvím rozhraní USB v režimu High Speed navrhněte zapojení obvodu pro přenos dat mezi počítačem a připojenou uživatelskou deskou uvedeným rozhraním (předpokládá se použití mikropočítače CY7C68013A s návazností na obvod typu FPGA umístěný na další desce připojené k mikropočítači přes konektor).

Zpracujte návrh plošného spoje desky s mikropočítačem a rámcový návrh jeho komunikace s obvodem FPGA.

DOPORUČENÁ LITERATURA:

[1] MATOUŠEK, D. USB prakticky s obvody FTDI. Praha: BEN, 2003.
[2] EZ-USB FX2TM PCB Design Recommendations (FX2_PCB.PDF). Cypress, 2002.
[3] FX2LP_ATA.ZIP. Cypress Reference Design, 2005.
[4] EZ-USB Technical Reference Manual. Cypress Semiconductor, 2005.

Termín zadání: 11.2.2008 ***Termín odevzdání:*** 6.6.2008

Vedoucí projektu: doc. Ing. Jaromír Kolouch, CSc.

prof. Dr. Ing. Zbyněk Raida
předseda oborové rady

UPOZORNĚNÍ:

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Komunikace obvodu FPGA s počítačem - USB High Speed jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 6. června 2008 .....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce doc. Ing. Jaromírovi Kolouchovi, CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 6. června 2008 .....
podpis autora

1 Úvod

Úkolem této práce bylo prostudování protokolu USB 2.0, zaměření se na vysokorychlostní přenos přes tuto sběrnici, a následně návrh uživatelské desky, podporující přenos dat mezi touto deskou a počítačem. Jako připojená periferie se předpokládá FPGA programovací obvod. Práce se dělí na tři části.

Prví část má za úkol vysvětlení funkčnosti USB. Začína od základů, a vysvětluje princip fungování přenosu, typy komunikací a způsob kódování. Tato část je velice obecná, avšak důležitá. Zahrňujou se sem kapitoly od 1 do 6.

Druhá část projektu se zabývá už s vlastním mikrokontrolérem, který se následně bude použít pro návrh systému. V této kapitole se rozebere vnitřní struktura této součástky, a opíšou hlavní části. Rozsah této kapitoly je od 7 do 9. Po dosažení dostatečných množství informací, se projekt dostáva do třetího stádia.

Třetí část se začíná od kapitoli 10 a tvoří hlavní část projektu. Nejdřív se seznamuje s možnosti spojení a následné naprogramování obvodu FPGA. Samotný vysokorychlostní přenos se řeší v části, která popisuje vlastnosti rozhraní GPIF. Tato část je v použitém mikrokontroléru nejrychlejší přenosovou jednotkou. Na konci projektu zpracuju návrh desky plošného spoje, který by měl zvládat vysokorychlostní přenos dat pomocí GPIF jednotky mikrokontroléru.

2 Historie a vlastnosti USB:

V roce 1996 vznikla první specifikace USB. Jednalo se o verzi 1.0. Tato specifikace však úplně nebyla jednoznačná a proto některá zařízení spolu nespolupracovala. Z tohoto důvodu se během roku 1998 objevila upravená verze této specifikace, která nesla označení 1.1. Maximální přenosová rychlost 12Mb/s však přestala vyhovovat a v roce 2000 vznikla verze 2.0, která podporuje přenosové rychlosti až do 480 Mb/s. Tato poslední verze USB se na trhu výrazně prosadila až na přelomu roku 2001 a 2002 kdy se objevila první zařízení podle této specifikace.

Požadavky na rozhraní :

- možnost připojit více zařízení současně
- možnost připojovat a odpojovat zařízení za chodu k počítači
- při konfiguraci nového zařízení se obejít bez zásahu uživatele
- poskytnout dostatečné přenosové kapacity

2.1 Vlastnosti USB sběrnice a USB zařízení :

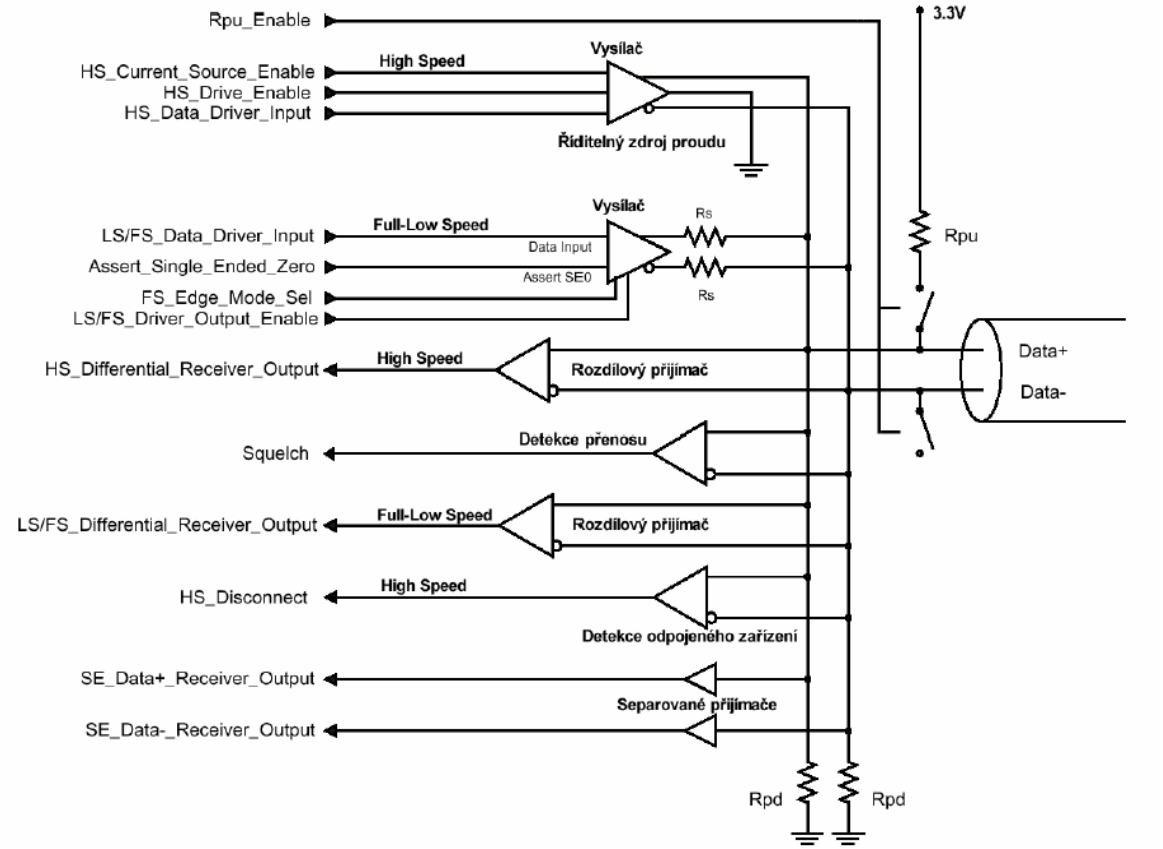
- nezávislost na použité platformě (PC, MacOS)
- garantovaná přenosová kapacita a zpoždění
- podpora pro přenosy dat v reálném čase
- možnost připojit až 127 zařízení současně
- lze připojovat a odpojovat zařízení za chodu
- konfiguraci připojeného zařízení provede hostitelský systém bez zásahu uživatele (konfigurace prostředků zařízení)
- nízká zátěž samotného protokolu včetně volby velikosti datového paketu
- Dostatečná přenosová kapacita :
  - Low Speed zařízení (LS) - maximální rychlost 1,5Mb/s
  - Full Speed zařízení (FS) - maximální rychlost 12 Mb/s
  - High Speed zařízení (HS) - maximální rychlost 480 Mb/s
- mechanismy pro detekci chyb a následné opakované poslání poškozených paketů
- podpora PnP (plug and play) pro detekci zařízení a následnou volbu ovladače (pokud je třeba)
- možnost napájet energeticky nenáročná zařízení přímo ze sběrnice

2.2 Fyzická vrstva

USB sběrnice používá pro propojení zařízení s hostitelem celkem 4 vodiče :

- **červený** – napájecí napětí +5V
- **černý** – napájecí a signálová zem
- **zelený** – signálový vodič D+
- **bílý –** signálový vodič D-

Signály D+ a D- lze považovat za signály komplementární při přenosu dat. V některých okamžicích mají oba stejnou úroveň a slouží k signalizaci specifických požadavků. USB zařízení podle verze 1.1 používají pro přenos dat napěťové úrovně. Naproti tomu USB zařízení podle verze 2.0, které komunikuje na HS používá pro přenos dat proudové smyčky. Pokud toto zařízení komunikuje na FS používá pro signalizaci opět napěťové úrovně. Z tohoto důvodu musí mít HS zařízení přijímač i vysílač, který je schopen zpracovávat data přenášená napěťovými úrovněmi a také proudovou smyčkou. Schéma HS přijímače je zobrazena na následujícím obrázku:



Obrázek 1 - HS přijímač vysílač

(zdroj: literatura [5] *Universal Serial Bus Specification*)

USB specifikace nepoužívá pojmy logická „1“ a logická „0“ ale stavy **„J“ a „K“**. Podle toho na jaké rychlosti zařízení komunikuje (LS, FS, HS) jsou těmto stavům přiřazeny příslušné napěťové úrovně. Tyto urovně mají následující hodnoty napětí:

V_{OH} = 2,8 - 3,6V, V_{OL} = 0 - 0,3V V_{HSOH} = 360 - 440mV, V_{HSOL} = -10 - 10mV

Definice stavů:

	FS/LS		HS	
	D+	D-	D+	D-
High	D+ > $V_{OH\ (min)}$	D- < $V_{OL\ (max)}$	$V_{HSOH\ (min)} \leq D+ \leq V_{HSOH\ (max)}$	$V_{HSOL\ (min)} \leq D- \leq V_{HSOL\ (max)}$
Low	D+ < $V_{OL\ (max)}$	D- > $V_{OH\ (min)}$	$V_{HSOL\ (min)} \leq D+ \leq V_{HSOL\ (max)}$	$V_{HSOH\ (min)} \leq D- \leq V_{HSOH\ (max)}$

Tabulka 1 - Definice napěťových úrovní

	LS	FS	HS
J	Low	High	High
K	High	Low	Low

(zdroj: literatura [4] *Univerzální Sériová Sběrnice*)

Pro připojení USB zařízení se používá kabelů, jejichž impedance je 90Ω. Oba datové vodiče D+ a D- jsou jak na straně rozbočovače tak na straně zařízení zakončeny 45Ω rezistorem pro komunikaci na HS. Po FS a LS komunikaci má vysílač i přijímač impedanci 45Ω. V předchozích tabulkách nejsou

uvedeny číselné hodnoty. Zjednodušeně lze tvrdit, že u LS a FS signalizace odpovídá High +5V a Low 0V. U HS zařízení neslouží pro přenos informací napěťové úrovně přímo. Je použita proudová smyčka s proudem o velikosti +-17,78mA. Vedení je na každé straně zakončeno rezistorem 45Ω proti zemi. Průchodem proudu vznikne na každém rezistoru +400mV nebo -400mV podle směru procházejícího proudu. Tyto 2 rezistory jsou vlastně paralelně zapojeny, a tak na obou z nich teče nějaký proud. Protože jsou si stejný odpory, tak ten proud je poloviční na obou, čili 8,89mA. Z toho plyne, že na konci datových vodičů bude napětí:

$$U = R * I = 45\Omega * 8{,}89mA = +400{,}05mV$$

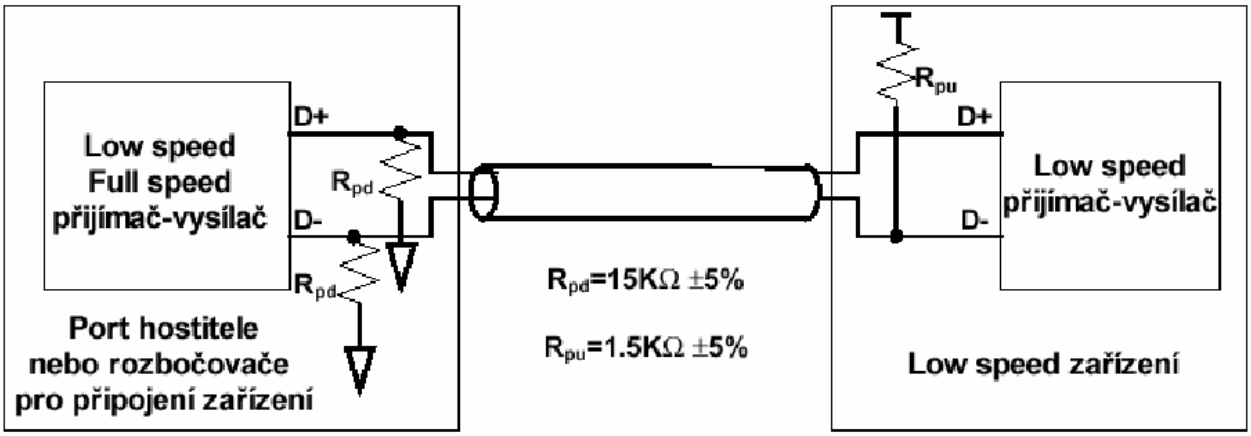
Pro druhý případ, kdy proud teče opačným směrem, by napětí bylo -400,05mV. Z toho plyne, že nominální diferenciální High-Speed napětí (D+ - D-) je 400mV pro stav J a -400mV pro K.

Rychlost	Přenosový výkon	Standard
Low Speed	1,5 Mb/s	USB 1.1/2.0
Full Speed	12Mb/s	USB 1.1/2.0
High Speed	480 Mb/s	USB 2.0

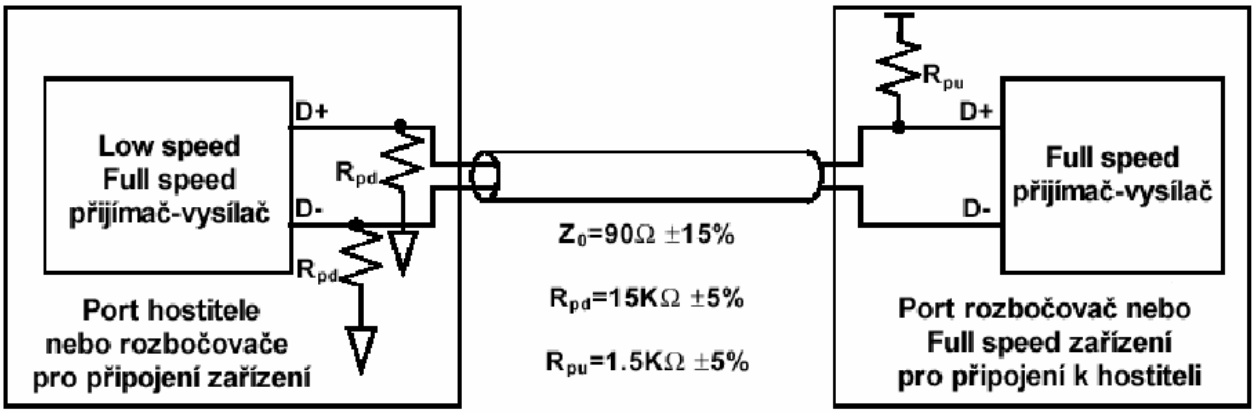
Tabulka 2 - Přenosové rychlosti USB
(zdroj: literatura [4] *Univerzální Sériová Sběrnice*)

2.3 Jak se pozná, že zařízení je LS, FS nebo HS

Podle rezistoru o velikosti 1,5kΩ, který je připojený na signál D+ nebo na D-.



Obrázek 2 - Low Speed zařízení
(zdroj: literatura [5] *Universal Serial Bus Specification*)



Obrázek 3 - Full Speed zařízení
(zdroj: literatura [5] *Universal Serial Bus Specification*)

HS zařízení má k signálu D+ připojený rezistor o velikosti 1,5kΩ proti +3,3V (Obrázek 3 - Full Speed zařízení). Tento rezistor je odpojitelný a používá se pouze při detekci zařízení. Zařízení se detekuje stejně, jako zařízení Full Speed s tím, že změna rychlosti se řeší programově. Jestli je zařízení schopno komunikovat na HS se zjišťuje během BusReset.

2.4 Shrnutí k signalizaci :

- signály D+ a D- jsou navzájem komplementární
- LS a FS zařízení používají pro přenos dat napěťové úrovně
- HS zařízení používá proudové smyčky
- signalizace u FS je invertovaná oproti LS
- rychlost zařízení se pozná podle rezistoru, který je připojený na D+ nebo D-
- signály D+ a D- musí být na každé straně zakončeny rezistorem o velikosti 45Ω, což odpovídá polovině jmenovité impedance kabelu.

3 Model toku dat

V této části jsou vysvětleny funkce vrstev, z kterých mají některé na starosti jednotlivé operace a přes nekteré procházejí data.

3.1 Fyzická vrstva – stará se o fyzický přenos dat. Jde o konkrétní HW, který je většinou implementován na základní desce počítače. Společně s ovladačem pro daný USB obvod je tato část zodpovědná za příjem a vysílání paketů včetně generování a kontrolu CRC. Díky ovladači poskytuje standardní rozhraní pro vyšší vrstvu. Ta je tak nezávislá na konkrétní HW implementaci.

3.2 Komunikační vrstva – na této úrovni probíhá detekce a konfigurace zařízení a následný přenos dat. Skládá se ze dvou částí. Nižší je zodpovědná za správu a řízení zařízení. Je pevnou součásti systému. Vyšší část je reprezentována ovladačem konkrétního zařízení a zodpovídá za příjem požadavků z vyšší vrstvy, jejich transformaci a předávání vrstvě nižší.

3.3 Funkční vrstva – typickým představitelem je konkrétní klientský SW, který chce komunikovat se „svým“ zařízením. Nezajímá se jak přesně je zařízení připojeno, požaduje pouze bezchybný přenos dat.

Na straně USB zařízení nelze jednoznačně určit, kde přesně začíná a končí jednotlivé vrstvy. Vždy bude záležet na konkrétní implementaci. Jednou z možností je kompletní obvodová implementace, kde zpracování všech standardních požadavků ze strany USB hostitele a také požadavků ovladače zařízení obstarává samotný obvod. Jakákoli změna v parametrech USB zařízení znamená zásah do struktury obvodu nebo jeho paměti. Druhou možností je obvod, který je schopen zpracovávat fyzickou vrstvu a komunikační vrstvu musí již implementovat další obvod. Ve většině případů se používá mikrokontrolér. U funkční vrstvy je problém podobný. Funkčnost může realizovat samotný obvod, nebo řidicí mikrokontrolér nebo další obvod, který využívá USB rozhraní ke komunikaci. Fyzické USB zařízení může v sobě integrovat více logických zařízení. Typickým představitelem může být kopírka, tiskárna a scanner v jednom zařízení. Každé zařízení má své rozhraní (interface) a pomocí něj lze využívat služeb daného zařízení. Výsledkem je, že fyzicky je připojeno pouze jedno zařízení avšak v systému se může vyskytnout několik nových logických zařízení.

4 Komunikace s USB

Při komunikaci s USB zařízením se používají komunikační kanály, roury (pipes), které jsou zakončeny na straně zařízení koncovým bodem (EndPoint-EP) a na straně hostitele vyrovnávací pamětí. Koncový bod není nic jiného než vyrovnávací paměť, která má definovány určité parametry a ty jsou vázány na danou rouru. Roura může být pouze jednosměrná. Pokud požadujeme obousměrnou komunikaci musíme použít roury dvě. Terminologie USB rozlišuje dva typy kanálů (pipes) :

Message pipe – přenášena data jsou strukturovaná a jednotlivé položky mají z pohledu USB určitý význam. Tyto informace však nejsou interpretovány koncovým bodem ale až vyšší komunikační vrstvou.

Stream pipe – přenášená data nemají z pohledu USB specifikace žádnou strukturu a jejich skutečný význam zná až funkční vrstva.

4.1 Typy přenosů

USB specifikace rozlišuje celkem čtyři typy koncových bodů a příslušných kanálů a od toho se odvíjející 4 typy přenosů. Typ podporovaného přenosu pro daný EP se nastavuje během konfigurace zařízení a za provozu se nemůže již změnit. U všech těchto přenosů je volitelná velikost paketu. Existuje několik omezení v závislosti na použité revizi USB sběrnice a rychlosti zařízení, na které komunikuje.

Control transfer – tento typ přenosů se používá pro detekci a konfiguraci zařízení, přenášena data mají specifikovanou strukturu a jedná se tedy o Message pipe. Tento typ přenosů se většinou nepoužívá pro data, které chce přenést uživatelská aplikace.

Interrupt transfer – pokud je třeba přenést data ze zařízení do počítače s definovaným maximálním zpožděním, je použit tento typ přenosu. Jeden z parametrů nastavovaných při konfiguraci je právě maximální akceptovatelné zpoždění.

Isochronous transfer – pokud je třeba přenášet data v pravidelných intervalech a s pevnou velikostí (real time data) je použit tento typ přenosu. Pokud dojde k chybnému přenosu paketu, tak se tato chyba ignoruje a data se zahazují.
Oba předchozí typy přenosů mají jisté nároky na časování. Protože veškerou komunikaci na USB řídí hostitel, musí všechny tyto požadavky vzít v úvahu při tvorbě časového schématu sběrnice. V případě Interrupt přenosů se musí hostitel dotazovat zařízení na data nejpozději v okamžiku kdy vyprší čas na maximální zpoždění přenosu dat. V případě Isochronous přenosů se musí zařízení dotazovat v přesně definovaném intervalu a musí vždy vyhradit dostatečný prostor pro přenos dat.

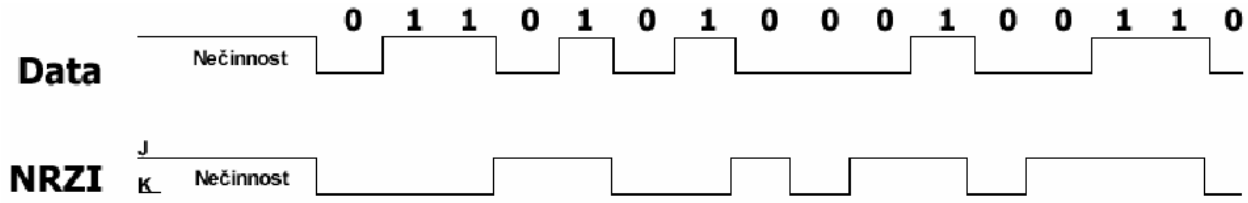
Bulk transfer – tento typ přenosů nemá žádné omezení na časování případně vyhrazenou přenosovou kapacitu. Pokud je na USB více logických zařízení, které používají tento typ přenosu snaží se hostitel zbývající prostor v časovém schématu sběrnice spravedlivě podělit mezi všechny přenosy tohoto typu.

Z charakteristiky výše uvedených přenosů plyne několik poznatků :

- Pro přenosy typu Control si systém vyhradí zhruba 10% přenosové kapacity sběrnice. Tím je zajištěno, že se vždy povede detekovat nově připojené zařízení.
- Pokud je na USB připojeno relativně hodně zařízení nemusí být v časovém schématu sběrnice již prostor pro nové zařízení a nemusí tedy fungovat i když se správně detekuje.
- Chyby u přenosu typu Isochronous se neopravují a chybná data se zahazují.

5 Kódování, časování

Při komunikaci se nevysílají přímo stavy J a K, které reprezentují příslušné bity. Provede se nejdříve NRZI překódování, které je zobrazeno na následujícím obrázku (Obrázek 4 - NRZI kódování).



Obrázek 4 - NRZI kódování
(zdroj: literatura [4] *Univerzální Sériová Sběrnice*)

5.1 Princip NRZI :

- pokud je „0“ změň signál na signál opačný
- pokud je „1“ nedělej nic

Všechna data jsou vysílána sériově. Pokud však nedojde po delší dobu ke změně signálu mohlo, by dojít k rozsynchronizování hodin vysílače a přijímače. To by se mohlo stát v případě, kdy by se vysílaly za sebou samé „1“, protože ty nemění výstupní signál. Používá se metoda vycpávání (bit stuffing). Pokud je za sebou vysláno šest „1“ je automaticky vložena „0“, která se na straně přijímače následně odstraní. Tímto způsobem lze detekovat i některé chyby. Pokud je za sebou víc jak sedm „1“ je určitě něco špatně.

Problém synchronizace hodin se také musí vyřešit při vysílání paketu. Než se začnou přenášet užitečná data jsou vyslány 3 páry KJ a 2 K signály („KJKJKJKK“) u Low Speed a Full Speed zařízení. U High Speed zařízení se používá 15 KJ páru a 2 K signály („KJKJKJKJKJKJKJ KJKJKJKJKJKJKK“).

5.2 Rámec, mikrorámec

Časování sběrnice je u LS a FS komunikace děleno do rámců (frame) kde každý rámec trvá 1ms. Na začátku každého rámce pošle hostitel paket (StartOfFrame-SOF) s číslem rámce. Pro počítání rámců se používá 11 bitový čítač, který přetéká a začíná znovu počítat od nuly. U HS komunikace se každý rámec skládá z 8 mikrorámců (microframe), což odpovídá délce 125ms.

6 Přenos dat a příbuzné pojmy:

USB je jednomasterová sběrnice, a všechny aktivity vycházejí z počítače. Data se vysílají v paketech o délce 8 až 64 (1024 pro izochronní přenos) bajtů. Počítač může požadovat data od zařízení, ale žádné zařízení nemůže začít vysílat samo od sebe.
Veškerý přenos dat se uskutečňuje po rámcích o délce 1 ms. Uvnitř těchto rámců mohou být postupně zpracovávány pakety pro několik zařízení. Spolu se mohou vyskytovat pakety různých rychlostí. Obrací-li se počítač na více zařízení, zajišťuje rozdělení paketů hub (rozbočovač), který také zabraňuje tomu, aby se signály s vyššími rychlostmi předaly na pomalá zařízení.

Slave (podřízené zařízení) se musí zasynchronizovat na datový tok. Protože hodinový signál není přenášen po zvláštní lince, získávají se hodiny přenosu přímo z datového signálu. K tomu se používá metoda NRZI (Non Return To Zero). Nuly v datech vedou k změně úrovně, jedničky nechávají úrovně beze změny.

Kódování a dekódování signálů je čistě hardwarovou záležitostí. Přijímač musí být schopen získat hodinový signál, přijmout a detekovat data. Specielní prostředky zajišťují, aby nedocházelo ke ztrátě synchronizace.

Každý paket obsahuje za účelem synchronizace speciální bajt, tzv. sync-bajt (00000001b). Přijímač pak v důsledku NRZI a vsouvání bitů vidí 8 střídajících se bitových stavů, na které se snadno zasynchronizuje. Během následujícího přenosu musí být synchronizace zachována.

Vysílač i přijímač se v každé součástce objevují společně. Zařízení obsahuje jednotku SIE (Seriál Interface Engine), která přebírá vlastní práci. K výměně dat mezi SIE a zbytkem zařízení slouží buffery (vyrovnávací paměti) FIFO. Architektura FIFO představuje paměti schopné postupně přijímat a vysílat data podobně jako posuvné registry (údaje zapsané vysílačem do FIFO se vysílají v pořadí zápisu, údaje přečtené přijímačem se čtou v pořadí příjmu). FIFO umožňuje vzájemně sladit rozdílné rychlosti USB sběrnice a USB zařízení.

Existují mikrokontroléry, u nichž je SIE jejich přímou součástí (např. AN2131). Existují také speciální obvody (PDIUSBD11,PDIUSBD12), které se k mikrokontroléru připojují například přes sběrnici I^2C.

Zařízení USB má obecně několik pamětí FIFO, jejichž prostřednictvím je možno přenášet data. K adrese zařízení se pak navíc přidává adresa tzv. koncového bodu (endpointu). Tato adresa udává, kam se data mají uložit nebo odkud se mají vyzvednout (udávají použitou FIFO). Například myš má vždy endpoint 0 (používá se při inicializaci) a endpoint 1 (sem mikrokontrolér zapisuje užitková data a počítač si je vybírá).

USB programy tvoří tzv. roury (pipes) k jednotlivým endpointům. Jedna pipe pak představuje logický kanál k jednomu endpointu v jednom zařízení. Rouru si tedy můžeme představit jako datový kanál tvořený jedním vodičem (ve skutečnosti jsou však data z pipe přenášena jako datové pakety v milisekundových rámcích a hardware je směruje do reálné paměti podle jejich endpointu). Jedno zařízení může současně používat více rour, takže přenosová rychlost pak vzroste.

7 Vlastnosti mikrořadičů CY7C68013A/CY7C68014A a CY7C68015A/CY7C68016A

EZ-USB FX2LP (CY7C68013A/14A) je vyráběn firmou Cypress Semiconductor Corporation, a je to malovýkonová verze EZ-USB FX2 (CY7C68013), což je vysoce integrovaný malovýkonový USB 2.0 mikrořadič. Pomocí integrace USB 2.0 vysílače/přijímače, sériového rozhraní (serial interface engine SIE), rozšířeného 8051 mikroprocesoru, a programovatelných periferních rozhraní na jediném čipu, Cypress vytvořil velmi ekonomické řešení, které poskytne větší výhody pro výkonově nenáročné aplikace.

Vynalézavá architektura FX2LP má za následek datovou přenosovou rychlost přes 53Mbytes/s, což je maximální přípustná USB 2.0 šířka pásma, při použití levného 8051 mikrořadiče v pouzdře 56 QFN. Protože obsahuje USB 2.0 vysílač/přijímač, je FX2LP ekonomičtější, a to všechno poskytuje v součástce menší, než kdybychom používali USB 2.0 SIE nebo implementovali externí vysílač/přijímač. Cypress Smart SIE pomocí EZ-USB FX2LP ovládá většinu USB 1.1 a USB 2.0 protokolů hardwarově, a jeho následkem je uvolnění zabudovaného mikrořadiče pro aplikačně-specifické funkce a menší čas pro vývoj pro zajištění USB kompatibility.

General Programmable Interface (Všeobecné programovatelné rozhraní) (GPIF) a Master/Slave koncový bod FIFO (8-nebo 16-bitových sběrnicí dat) poskytuje snadné připojení k populárním rozhraním jako ATA, UTOPIA, EPP, PCMCIA, a většina DSP/procesorů. FX2LP odebírá podstatně menší proud, než FX2 (CY7C68013), má dvojitý čipový kód/data RAM a je tvarově a funkčně kompatibilní s 56-, 100-, a 128- pinovým FX2.

Čtyři patice jsou definované pro rodinu 56 SSOP, 56 QFN, 100 TQFP, a 128 TQFP.

8 Základní charakteristika (CY7C68013A/14A/15A/16A)

- USB 2.0–USB-IF high speed certifikace (TID # 40440111)
- Jednočipový integrovaný USB 2.0 vysílač/přijímač, inteligentní SIE, a vylepšený 8051 mikroprocesor
- Tvar a funkce je kompatibilní s FX2

-Vývodově kompatibilní
-Kompatibilní strojový kód
-Funkčně kompatibilní (FX2LP je superset)

- Extrémně malý odběr: I_{cc} menší než 85mA v jakémkoli režimu

-Ideální pro bateriově a sběrnicově napájené aplikace

- Software: 8051 kód běží z:

-vnitřní RAM -stáhnuto skrz USB
-načteno z EEPROM
-vnější paměťové zařízení (pouzdro 128 pinové)

- 16KByte čipového Kód/Data RAMu
- Čtyři programovatelné BULK/INTERRUPT/ISOCHRONOUS koncové body

-Vyrovnávací možnosti: dvojitý, trojitý a čtyřnásobný

- Přídavný programovatelný (BULK/INTERRUPT) 64-bytový koncový bod
- 8-nebo 16-bitové externí datové rozhraní
- Smart Media Standard (Inteligentní Média Standard) ECC generace
- GPIF (General Programmable Interface) (Všeobecně programovatelné rozhraní)

-umožní přímé spojení s většinou paralelních rozhraní
-programovatelné deskriptory tvarů a konfigurace registrů pro definování tvarů
-podpora vícenásobných Ready (RDY) vstupů a Control (CTL) výstupů

- Integrovaný průmyslový-standard vylepšeno pro 8051

-48-MHz, 24-MHz, anebo 12-MHz CPU operace
-Čtyři hodinové impulsy za jeden instrukční cyklus
-Dva USARTy
-Třikrát čítač/časovač
-Rozšířený systém přerušení
-Dva datové ukazatele

- Napájení 3,3V s max. 5V-ovými řídícími vstupy
- Vektorové USB přerušení a GPIF/FIFO přerušení
- Oddělené vyrovnávací paměti pro nastavovací (Set-up) a datové části (Data portions) z CONTROL přenosu
- Integrovaný I^2C kontrolér běží na 100 anebo 400kHz
- Čtyři integrované FIFO

-Integrované tzv. „glue logic“ (přizpůsobení komunikační sběrnice mezi jednotlivými IO)
-Automatický převod na 16-bitovou sběrnici a zpět
-Funkce v režimech Master anebo Slave
-Používá externí hodinový signál
-Snadné rozhraní pro ASIC a DSP Integrované čipy

9 Funkční přehled CY7C68013A/14A/15A/16A

9.1 USB signalizace rychlosti

FX2LP pracuje ve dvou ze tří rychlostí definovaných v USB Specification Revision 2.0, dated April 27, 2000:

- Full speed s přenosovou rychlostí 12Mbps
- High speed, s přenosovou rychlostí 480 Mbps.

FX2LP nepodporuje low-speed přenosovou rychlost 1,5Mbps.

9.2 8051 Mikroprocesor

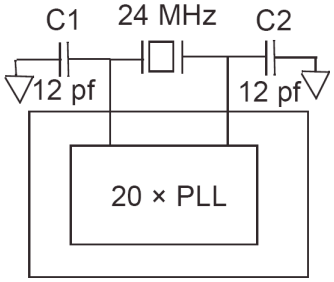
8051 mikroprocesor z rodiny FX2LP má 256 bajtů registrového RAMu, rozšířený systém přerušení, tři časovače/čítače a dvě synchroní a asynchroní sériové rozhraní USART.

8051 Hodinová frekvence

FX2LP má na vlastním čipu oscilátorový obvod (násobič PLL) který používá vnější 24 MHz (+-100-ppm) krystal s následujícími vlastnosti:

- Paralelní rezonátor
- 500uW odběr
- 12pF (5% tolerance) kapacita

Čipový PLL násobí 24MHz oscilátor na 480MHz, jak je požadováno od přenosové jednotky/PHY (physical layer) a vnitřní čítače zas dělí na požadovanou hodnotu pro 8051. Standardně 8051 má hodinový impuls o frekvenci 12MHz. Hodinový impuls 8051 může být změněn pomocí 8051 přes CPUCS registr dynamicky.



Obrázek 5 - Zapojení krystalu
(12pF kondenzátorové hodnoty převezmou kapacitu vedení 3pF za jednu stranu na čtyř-vrstvém PCB typu FR4 PCA)
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

CLKOUT pin, který může být tří stavový a invertovaný pomocí vnitřních řídících bitů, má na výstupu 50%-ní hodinový pracovní cyklus 8051, na vybrané frekvenci 48, 24 anebo 12MHz.

USARTy

FX2LP obsahuje dva standardní 8051 USARTy (Universal asynchronous receiver/transmitter), adresované přes bity Speciálního Funkčního Registru (SFR). Tyto jednotky slouží vlastně k tomu, aby se paralelní komunikace přetransformovala na sériovou, a zpátky. Piny rozhraní USART jsou dostupné na oddělených I/O pinech a nejsou multiplexovány se vstupními piny.

UART0 a UART1 mohou fungovat pomocí vnitřního hodinového generátoru na 230KBaud-ů s maximálně 1%-ní chybou přenosové rychlosti. 230KBaudová funkčnost je dosažena vnitřním sekundárním zdrojem hodinového impulsu, které generuje overflow pulsy v příslušných časech. Vnitřní hodinový generátor přizpůsobí pro 8051 taktovací kmitočet (48 ,24, 12MHz) tak, aby vždycky měla správnou frekvenci pro 230KBaud-ovou funkčnost.

Speciální Funkční Registry

Jisté 8051 SFR adresy jsou obsazeny pro poskytnutí rychlého přístupu ke kritickým FX2LP funkcím. Tyto SFR dodatky jsou ukázané v tabulce 3. Tlustý písmena ukazují nestandardní, rozšířené registry 8051. Dvě řádky SFR které končí s 0 a 8 obsahují bitově adresované registry. Čtyři I/O porty

A-D používají SFR adresy použité v standardním 8051, pro porty 0-3, které nejsou implementovány v FX2LP. Kvůli rychlejšímu a efektivnějšímu SFR adresování, FX2LP I/O porty není možné adresovat v externím RAMu (pomocí instrukce MOVX).

9.3 I^2C Sběrnice

FX2LP podporuje I^2C sběrnici jenom jako master na 100-/400-KHz. SCL a SDA piny mají open-drain výstup a hysterezní vstup. Tyto signály musí být připojeny na 3,3V i tehdy, když žádné I^2C zařízení není připojeno.

9.4 Sběrnice

Všechny pouzdra: 8- nebo 16-bitová FIFO obousměrná datová sběrnice, je multiplexována na I/O portech B a D. 128 pinové pouzdro: přidává pro 8051 16 bitovou, jenom výstupovou adresovou sběrnici, a 8 bitovou obousměrnou datovou sběrnici.

x	8x	9x	Ax	Bx	Cx	Dx	Ex	Fx
0	**IOA**	**IOB**	**IOC**	**IOD**	SCON1	PSW	ACC	B
1	SP	**EXIF**	**INT2CLR**	**IOE**	SBUF1			
2	DPL0	**MPAGE**	**INT4CLR**	**OEA**				
3	DPH0			**OEB**				
4	**DPL1**			**OEC**				
5	**DPH1**			**OED**				
6	**DPS**			**OEE**				
7	PCON							
8	TCON	SCON0	IE	IP	T2CON	**EICON**	**EIE**	**EIP**
9	TMOD	SBUF0						
A	TL0	**AUTOPTRH1**	**EP2468STAT**	**EP01STAT**	RCAP2L			
B	TL1	**AUTOPTRL1**	**EP24FIFOFLGS**	**GPIFTRIG**	RCAP2H			
C	TH0	**reserved**	**EP68FIFOFLGS**		TL2			
D	TH1	**AUTOPTRH2**		**GPIFSGLDATH**	TH2			
E	**CKCON**	**AUTOPTRL2**		**GPIFSGLDATLX**				
F		**reserved**	**AUTOPTRSET-UP**	**GPIFSGLDATLNOX**				

Tabulka 3 - Speciální funkční registry
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

9.5 USB Bootovací metody

Během zapínací sekvence, vnitřní logika zkontroluje I^2C port pro připojení EEPROM, jehož první bajt je buď 0xC0 nebo 0xC2. Když to najde, tak použije VID/PID/DID hodnoty v EEPROM, místo hodnot uložených uvnitř (0xC0), anebo načte obsah EEPROM do vnitřní RAM (0xC2). Když EEPROM není detekována, tak FX2LP se enumeruje pomocí vnitřně uložených řetězců. Standardní ID hodnoty pro FX2LP jsou VID/PID/DID (0x04B4, 0xAxxx kde xxx=Čipová revize).

I^2C piny sběrnice, SCL a SDA musí být staženy nahoru i když EEPROM není zapojeno. Jinak tato detekční metoda nebude správně fungovat.

Default VID/PID/DID		
Vendor ID	0x04B4	Cypress Semiconductor
Product ID	0x8613	EZ-USB FX2LP
Device release	0xAnnn	Depends on chip revision (nnn = chip revision where first silicon = 001)

Tabulka - 4 Standardní ID hodnoty pro FX2LP
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

9.6 ReNumeration™

Protože konfigurace FX2LP je chytrá, jeden čip může vzít identity několik odlišných USB zařízení. Když USB je zapojeno první krát, FX2LP se enumeruje automaticky, a stáhne firmware a USB deskriptorové tabulky přes USB kabel. Následně FX2LP se enumeruje znovu, ale teďka už jako zařízení definováno v stažených informacích. Tento patentovaný dvou krokový proces, zvaný ReNumeration, se vykoná okamžitě, když se připojí zařízení, ve kterém není naznačení o tom, že počáteční krok se objevil. Dvě kontrolní bity v USBCS (USB kontrola a stav) registru kontrolují ReNumeration proces: DISCON a RENUM. Pro simulace že USB byl odpojen, firmware nastaví DISCON na 1. Pro znovu zapojení, firmware vynuluje DISCON.

Před znovu zapojením, firmware nastaví, anebo vynuluje RENUM bit, pro indikace, či firmware anebo Defaultní USB Zařízení chtějí ovládat žádosti zařízení přes endpoint zero: když RENUM=0, defaultní USB zařízení bude ovládat žádosti zařízení, když RENUM=1, firmware bude.

9.7 Sběrnicově napájené aplikace

FX2LP zcela podporuje sběrnicově napájené konstrukce. Díky tomu při enumeraci je jeho proudový odběr menší jak 100mA, jak je požadováno ve specifikaci USB 2.0.

9.8 Přerušovací systém

INT2 Přerušovací požadavky a povolovací registry

FX2LP implementuje autovektorovou vlastnost pro INT2 a INT4. Nachází se tady 27 INT2 (USB) vektorů, a 14 INT4 (FIFO/GPIF) vektorů.

USB-přerušovací autovektory

Hlavní USB přerušení jsou sdělena od 27 přerušovacích zdrojů. Pro ušetření kódového a zpracovacího času, co by bylo normálně potřebné pro identifikace individuálního přerušovacího zdroje USB, FX2LP provede druhý stupeň přerušovacího vektorování, zvaný Autovectoring. Když USB přerušení je potvrzeno, FX2LP posune programový čítač na jeho zásobník, pak skočí na adresu 0x0043, kde očekává, že najde skokovou instrukci (“jump”) k USB přerušovací servisní rutině (Interrupt service routine).

FX2LP skokové instrukce jsou kódovány následovně:

viz. další strana

USB Tabulka přerušení pro INT2			
Priorita	**INT2VEC Hodn.**	**Zdroj**	**Poznámky**
1	00	SUDAV	Set-up Data Available **Nastavovací data dostupné**
2	04	SOF	Start of Frame (or microframe) **Začátek rámce (mikrorámce)**
3	08	SUTOK	Set-up Token Received **Nastavovací znak přijat**
4	0C	SUSPEND	USB Suspend request **USB přerušovací žádost**
5	10	USB RESET	Bus reset **Resetování sběrnice**
6	14	HISPEED	Entered high speed operation **Vstup vysokorychlostní operace**
7	18	EP0ACK	FX2LP ACK'd the CONTROL Handshake
8	1C		reserved **Rezervováno**
9	20	EP0-IN	EP0-IN ready to be loaded with data **Připraven k načítání s daty**
10	24	EP0-OUT	EP0-OUT has USB data **Má USB data**
11	28	EP1-IN	EP1-IN ready to be loaded with data **Připraven k načítání s daty**
12	2C	EP1-OUT	EP1-OUT has USB data **Má USB data**
13	30	EP2	IN: buffer available. OUT: buffer has data **buffer dostupný/má data**
14	34	EP4	IN: buffer available. OUT: buffer has data **buffer dostupný/má data**
15	38	EP6	IN: buffer available. OUT: buffer has data **buffer dostupný/má data**
16	3C	EP8	IN: buffer available. OUT: buffer has data **buffer dostupný/má data**
17	40	IBN	IN-Bulk-NAK (any IN endpoint) **Jakýkoliv IN koncový bod**
18	44		reserved **Rezervováno**
19	48	EP0PING	EP0 OUT was Pinged and it NAK'd
20	4C	EP1PING	EP1 OUT was Pinged and it NAK'd
21	50	EP2PING	EP2 OUT was Pinged and it NAK'd
22	54	EP4PING	EP4 OUT was Pinged and it NAK'd
23	58	EP6PING	EP6 OUT was Pinged and it NAK'd
24	5C	EP8PING	EP8 OUT was Pinged and it NAK'd
25	60	ERRLIMIT	Bus errors exceeded the programmed limit **Sběrnicové chyby překročili nastavenou limitu**
26	64		
27	68		reserved **Rezervováno**
28	6C		reserved **Rezervováno**
29	70	EP2ISOERR	ISO EP2 OUT PID sequence error **Sekvenční chyba**
30	74	EP4ISOERR	ISO EP4 OUT PID sequence error **Sekvenční chyba**
31	78	EP6ISOERR	ISO EP6 OUT PID sequence error **Sekvenční chyba**
32	7C	EP8ISOERR	ISO EP8 OUT PID sequence error **Sekvenční chyba**

Tabulka - 5 INT2 USB Přerušení
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Když je Autovektorizace zapnuta (AV2EN=1 v INTSET-UP registru), FX2LP nahradí svůj INT2VEC byte. Proto když horní byte skokové tabulky je načtena z adresy 0x0044, automaticky vložený INT2VEC byte na 0x0045 bude směrovat skok na korektní adresu z 27 adres uvnitř stránky.

FIFO/GPIF Přerušení (INT4)

Podobně jako USB přerušení je sdíleno mezi 27 individuálními USB přerušovacími zdroji, FIFO/GPIF přerušení jsou sdíleny mezi 14 individuálními FIFO/GPIF zdroji. FIFO/GPIF přerušení, jako USB přerušení, mohou použít autovektorizace. Tabulka - 6 ukazuje priority a INT4VEC hodnoty pro 14 FIFO/GPIF přerušovacích zdrojů.

Priorita	INT4VEC Hodn.	Zdroj	Poznámky
1	80	EP2PF	Endpoint 2 Programmable Flag **Koncový bod 2 Programovatelný indikátor**
2	84	EP4PF	Endpoint 4 Programmable Flag **Koncový bod 4 Programovatelný indikátor**
3	88	EP6PF	Endpoint 6 Programmable Flag **Koncový bod 6 Programovatelný indikátor**
4	8C	EP8PF	Endpoint 8 Programmable Flag **Koncový bod 8 Programovatelný indikátor**
5	90	EP2EF	Endpoint 2 Empty Flag **Koncový bod 2 Prázdný indikátor**
6	94	EP4EF	Endpoint 4 Empty Flag **Koncový bod 4 Prázdný indikátor**
7	98	EP6EF	Endpoint 6 Empty Flag **Koncový bod 6 Prázdný indikátor**
8	9C	EP8EF	Endpoint 8 Empty Flag **Koncový bod 8 Prázdný indikátor**
9	A0	EP2FF	Endpoint 2 Full Flag **Koncový bod 2 Úplný indikátor**
10	A4	EP4FF	Endpoint 4 Full Flag **Koncový bod 4 Úplný indikátor**
11	A8	EP6FF	Endpoint 6 Full Flag **Koncový bod 6 Úplný indikátor**
12	AC	EP8FF	Endpoint 8 Full Flag **Koncový bod 8 Úplný indikátor**
13	B0	GPIFDONE	GPIF Operation Complete **GPIF Operace hotová**
14	B4	GPIFWF	GPIF Waveform **GPIF Tvar**

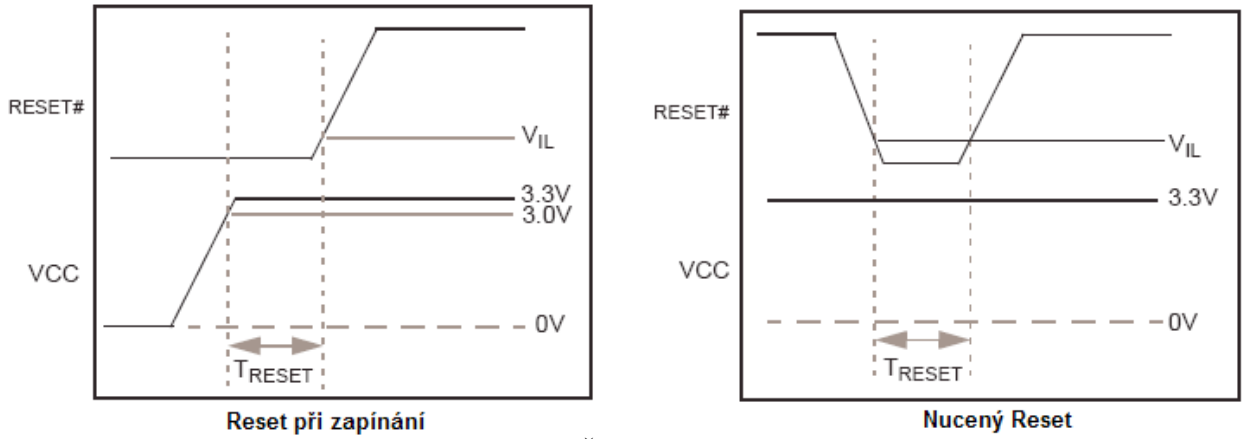
Tabulka 6 - Individuální FIFO/GPIF přerušovací zdroje
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Když je Autovektorizace zapnuta (AV4EN=1 v INTSET-UP registru), FX2LP nahradí svůj INT4VEC byte. Proto když horní byte skokové tabulky je načtena z adresy 0x0055, automaticky vložený INT4VEC byte na 0x0055 bude směrovat skok na korektní adresu ze 14 adres uvnitř stránky. Když ISR nastane, FX2LP posune programový čítač na jeho zásobník a skočí na adresu 0x0053, kde očekává, že najde "jump" instrukci k ISR přerušovací servisní rutině (Interrupt service routine).

9.9 Reset a Wakeup

Reset pin

Vstupním pinem RESET#, se resetuje FX2LP. Tento pin má hysterezi a je aktivní LOW. Když je krystal použit s CY7C680xxA, resetní perioda musí souhlasit pro stabilizaci krystalu a PLL. Tato resetovací perioda musí být přibližně 5ms po tom, jak VCC dosáhla hodnotu 3.0V. Když vstupní pin krystalu je řízen hodinovým signálem, tak interní PLL se ustálí v času 200us po tom, jak VCC dosáhlo 3.0V. Když externí hodinový signál je zapnutý ve stejném okamžiku, jako CY7C680xxA, a má čekací periodu pro ustálení, tak tato hodnota musí být přidána k hodnotě 200us. Obrázek - 6 ukazuje stav resetu při zapnutí, a použití resetu během operace. Nucený reset je definován k vykonání, když FX2LP byl dříve zapnutý a provozován, a RESET# pin byl nastavěn. Cypress poskytuje aplikační poznámku, která popisuje a doporučuje resetní implementaci při zapnutí.



Obrázek 6 - Časování Resetu
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Stav	T_{RESET}
Reset při zapínání s krystalem	5 ms
Reset při zapínání s externím hodinovým generátorem	200 µs + Ustálovací čas hodin. generátoru
Nucený Reset	200 µs

Tabulka 7 - Časovací hodnoty Resetu
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Wakeup piny

8051 vypne sám sebe a zbytek chipu přepnutím do power-down módu s nastavením PCON.0=1. Toto zastaví oscilátor i PLL. Když chce vykonat externí logika WAKEUP (probuzení), oscilátor restartuje, a jakmile se PLL ustálí, tak 8051 přijme wakeup přerušení. Toto se vykoná v každém případě, i tehdy když FX2LP není připojen k USB.

FX2LP vystoupí z power-down stavu (USB zastaven) pomocí následujících metod:

- USB sběrnicová aktivita (když na linkách D+/D- se objeví aktivita, šum signalizuje FX2LP, a spustí wakeup)
- Externí logika uplatní WAKEUP pin.
- Externí logika uplatní PA3/WU2 pin.

Druhý wakeup pin WU2 může být taky nakonfigurován jako generálně zaměřený I/O pin. Toto umožňuje použití jednoduché externí R-C sítě, jako zdroj periodického probuzení. Dejme pozor ale na to, že WAKEUP je aktivní standardně v LOW.

9.10 Program/Data RAM

Velikost

FX2LP má 16Kbajtů vnitřního programového/datového RAMu, kde PSEN#RD# signály jsou vnitřně ORovány pro povolení 8051 přistoupit k RAMu a to jak k programu, tak k datům. Žádné USB kontrolní registry se neobjeví v tomto prostoru.

Dvě rozložení paměti jsou ukázány v následujících diagramech:
Obrázek - 7 Vnitřní kódová paměť, EA=0
Obrázek - 8 Externí kódová paměť, EA=1.

Interní kódová paměť, EA = 0

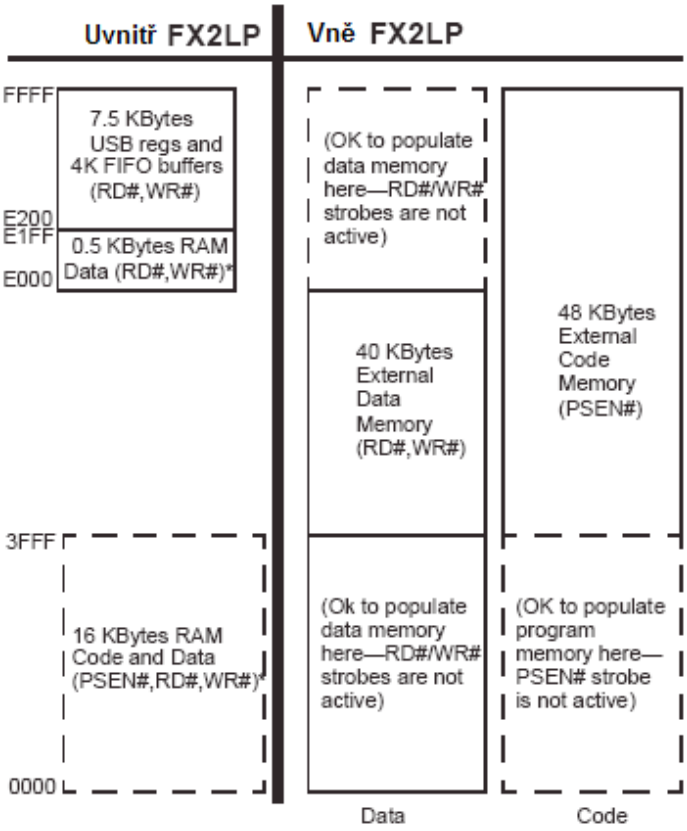
Tento mód implementuje 16KBajtový blok RAMu (začínající se na 0) jako kombinovaná paměť kódu a dat. Když je přidána externí paměť typu RAM anebo ROM, externí čtecí a zapisovací impulzy jsou potlačeny pro paměťové prostory uvnitř čipu. Toto umožňuje uživateli připojit 64KBajtovou paměť bez nutnosti použít adresové dekodéry pro udržení volné paměti uvnitř čipu.

Pouze interní 16KBajtová paměť a pomocná paměť (scratch pad), která má 0,5KBajtový RAMový prostor, mají následující přístupy:

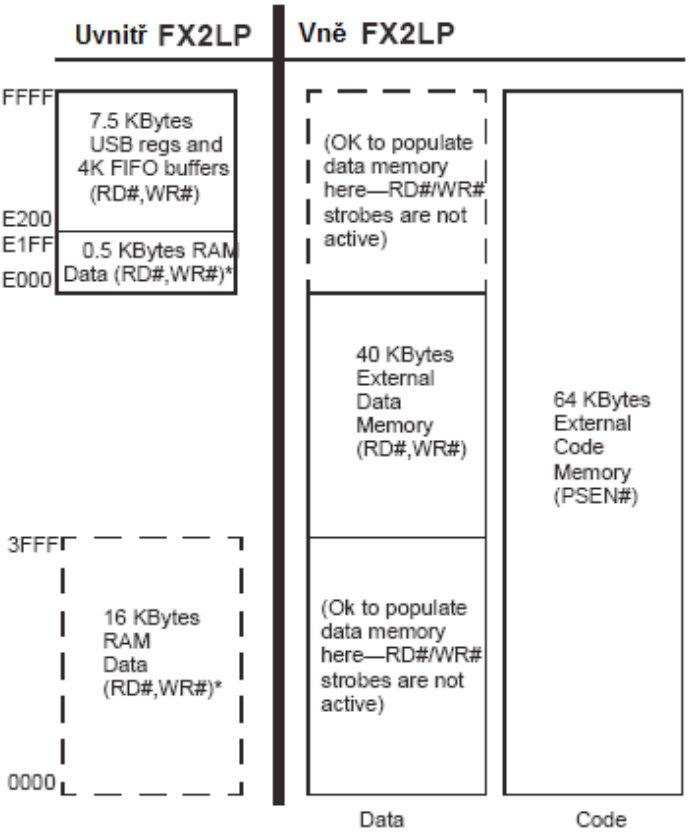
- USB stahování (download)
- USB nahrávání (upload)
- Nastavění datového ukazatele
- Zavádění (boot load) přes I^2C rozhraní

Externí kódová paměť, EA = 1

Dolní část 16KBajtové programové paměti je externí, a proto dolní část 16KBajtové interní RAM je dostupná jenom jako datová paměť.

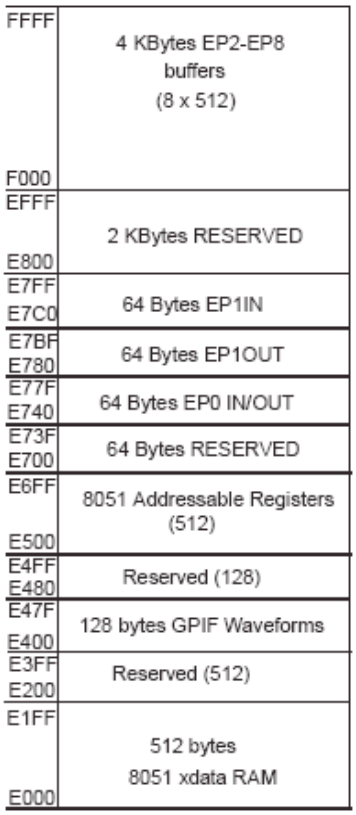


Obrázek 7 - Interní kódová paměť, EA=0
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)



Obrázek 8 - Externí kódová paměť, EA=1
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

9.11 Adresy registrů



Obrázek 9 - Adresy registrů
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

9.12 Endpoint RAM

Velikost

- 3 x 64 bajtů (Endpointy 0 a 1)
- 8 x 512 bajtů (Endpointy 2, 4, 6, 8)

Organizace

- EP0
- Obousměrný endpoint zero, 64 bajtový buffer
- EP1IN, EP1OUT
- 64 bajtový buffery, bulk, nebo interrupt (přerušovací)
- EP2, 4, 6, 8
- Osm 512 bajtových bufferů, bulk, interrupt, nebo isochronní. EP4 a EP8 mohou být double buffered , zatímco EP2 a 6 mohou být double, triple a quad buffered. Pro high-speed endpoint konfigurace viz obrázek 10.

Nastavění datového bufferu

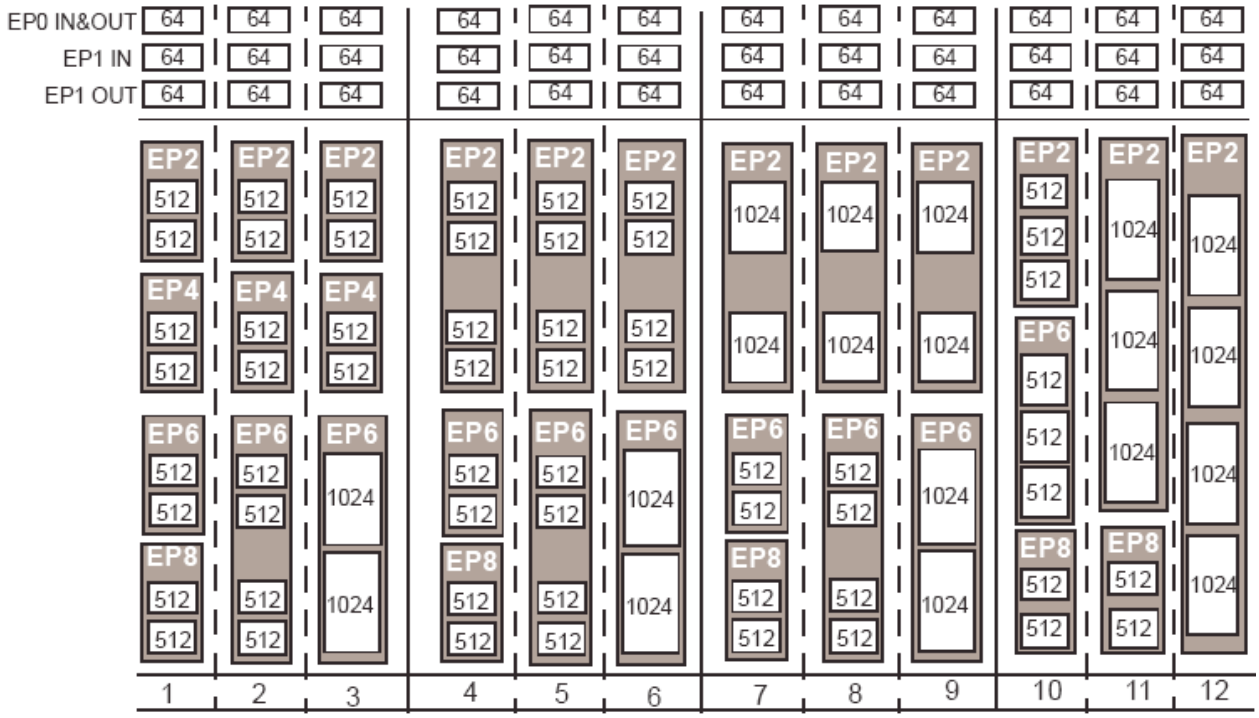
Oddělený 8 bajtový buffer na adrese 0xE6B8-0xE6BF obsahuje nastavovací data pro CONTROL přenos (CONTROL transfer).

Endpoint konfigurace (High-speed mód)

Koncové body (Endpoints) 0 a 1 jsou stejné pro všechny nastavění. Koncový bod 0 může být jenom „CONTROL“ koncový bod, ale EP1 už může být jak „BULK“, tak „INTERRUPT“. Buffery koncových bodů mohou být nastavěny na kterékoliv konfigurace od 1 do 12, jak je ukazováno ve sloupcích. Když se operuje ve full-speed BULK módu, jedině prvních 64 bajtů je používáno z jednotlivých bufferů. Například v high-speedu maximální velikost paketu je 512 bajtů, zatímco ve full-speed módu jenom 64 bajtů. Ačkoli je buffer nastavěn aby byl 512 bajtový buffer, ve full-speedu

se bude používat jenom prvních 64 bajtů. Nepoužitý prostor endpoint bufferu není dostupný pro jiné operace. Příklad konfigurace endpointu by byl:

EP2–1024 double buffered ; EP6–512 quad buffered (sloupec 8).



Obrázek 10 - Endpoint konfigurace
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Standardní alternativní nastavění pro full-speed

Alternativní nastavění	0	1	2	3
ep0	64	64	64	64
ep1out	0	64 bulk	64 int	64 int
ep1in	0	64 bulk	64 int	64 int
ep2	0	64 bulk out (2×)	64 int out (2×)	64 iso out (2×)
ep4	0	64 bulk out (2×)	64 bulk out (2×)	64 bulk out (2×)
ep6	0	64 bulk in (2×)	64 int in (2×)	64 iso in (2×)
ep8	0	64 bulk in (2×)	64 bulk in (2×)	64 bulk in (2×)

Tabulka 8 - Standardní alternativní nastavění pro full-speed
(0 znamená není implementován, 2x znamená double buffered)
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Standardní alternativní nastavění pro high-speed

Alternativní nastavění	0	1	2	3
ep0	64	64	64	64
ep1out	0	512 bulk *	64 int	64 int
ep1in	0	512 bulk *	64 int	64 int
ep2	0	512 bulk out (2×)	512 int out (2×)	512 iso out (2×)
ep4	0	512 bulk out (2×)	512 bulk out (2×)	512 bulk out (2×)
ep6	0	512 bulk in (2×)	512 int in (2×)	512 iso in (2×)
ep8	0	512 bulk in (2×)	512 bulk in (2×)	512 bulk in (2×)

Tabulka 9 - Standardní alternativní nastavění pro high-speed
(0 znamená není implementován, 2x znamená double buffered)
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

* i když tyto buffery jsou 64 bajtové, jsou označovány jako 512 bajtové pro vyhovění USB 2.0 standardu. Uživatel nesmí nikdy přenášet pakety větší jak 64 bajtů do EP1.

9.13 Externí FIFO rozhraní

Architektura

FX2LP podřízená (slave) FIFO architektura má osm 512 bajtových bloků v endpoint RAMu , které slouží přímo jako FIFO paměti, a jsou řízeny FIFO řídícími signály (jako jsou IFCLK, SLCS #, SLRD, SLWR, SLOE, PKTEND, a příznakové bity).

V provozu, některé z osmi RAM bloků se zaplní, nebo vyprázdní z SIE (serial interface engine), zatímco ostatní jsou připojeny k I/O přenosové logiky. Přenosová logika má dvě formy: GPIF pro vnitřně generované ovládací signály, a podřízené (slave) FIFO rozhraní pro vnějškově ovládané přenosy.

Master/Slave kontrolní signály

FX2LP endpointy FIFO jsou implementovány jako osm fyzikálně rozdílných 256x16 RAM bloků. 8051/SIE umí přepínat kterýkoliv RAM blok mezi dvěma doménami, a to mezi USB (SIE) doménou nebo 8051-I/O Unit doménou. Toto přepínání se dělá virtuálně okamžitě, a v podstatě za nulový přenosový čas mezi "USB FIFO" a "Slave FIFO". Protože fyzicky jsou tyto paměti stejné, žádný bajty se nepřenáší mezi buffery.

V libovolném okamžiku, jisté RAM bloky jsou zaplněny/vyprázdněny s USB daty pod kontrolou SIE, zatímco jiné RAM bloky jsou dostupné pro 8051 a/nebo pro I/O kontrolní jednotku. RAM bloky pracují jako single-porty v USB oblasti, a jako dual-porty v 8051 I/O oblasti. Bloky mohou být nastavěny jako single, double, triple nebo quad buffered.

V Master (M) módu GPIF vnitřně ovládá FIFOADR[1..0] pro vybírání FIFO. Piny RDY mohou být použity jako příznakové vstupy (flag inputs) z externí FIFO, nebo z jiné logiky, když je požadováno. GPIF může běžet pomocí vnitřního hodinového generátoru, nebo z externího (IFCLK), na rychlosti, která přenáší data na 96Megabyte/s (48-MHz IFCLK s 16 bitovým rozhraním).

V Slave módu (S) FX2LP přijímá vnitřní hodinový impulsy i vnější (IFCLK, max. frekvence 48MHz) a SLCS#, SLRD, SLWR, SLOE, PKTEND signály z externí logiky. Když použijeme externí IFCLK, vnější hodinový signál musí být přítomný už před zapnutím s IFCLKSRC bitem. Každý endpoint může být individuálně zvolen pro bytovou nebo word operace interním nastavovacím bitem, přičemž Slave FIFO Output Enable (Podřízený FIFO Povolení Výstupu) signál SLOE povoluje data zvolené šířky. Externí logika se musí ujistit, že výstupní povolovací signál je inaktivní, když zapisujeme data do slave FIFO. Slave rozhraní může operovat taky asynchronně, kde SLRD a SLWR signály pracují přímo jako synchronizace, nebo spíš hodinový kvalifikátor, jako v synchronním módu. Signály SLRD, SLWR, SLOE a PKTEND jsou synchronizovány se signálem SLCS#.

GPIF a FIFO hodinové rychlosti

8051 registrový bit vybere jednu ze dvou frekvencí pro vnitřní rohraní: 30MHz a 48MHz. Drouhou alternativou je, že externí hodinový generátor s 5MHz-48MHz který napájí IFCLK pin, se může používat jako hodinový signál pro interface. IFCLK můžeme nastavit tak, aby fungoval jako výstupní hodinový signál, když GPIF a FIFO jsou vnitřně napájeny hodinovým impulsem. Když

nechceme, aby tento výstup fungoval jako výstup hodinového signálu, tak v registru IFCONFIG s určitým bitem to můžeme vypnout. Další bit uvnitř IFCONFIG registru zinvertuje IFCLK signál, nezávysle na tom zda byl zdrojem vnitřní, nebo vnější hodinový signál.

9.14 GPIF

GPIF je flexibilní 8- nebo 16-bitové paralelní rozhraní řízené používatelem programovaným automatem, s konečnými stavy. To umožňuje CY7C68013A/15A vykonávat lokální sběrnicový mastering, a může implementovat širokou skálu protokolů, jako jsou ATA rozhraní, paralelní port pro tiskárnu a Utopia.

GPIF má šest programovatelných kontrolních výstupů (CTL), devět adresových výstupů (GPIFADRx), a šest univerzálních RDY (ready) vstupů. Šířka datové sběrnice může být 8 nebo 16 bitová. Každý GPIF vektor definuje stav kontrolního výstupu, a určuje jaký stav má mít ready vstup (nebo vícenásobné vstupy) před činností. GPIF vektor může být naprogramován pro napomáhání postupu FIFO na další hodnotu, postupování na adresu, atd. Posloupnost GPIF vektorů tvoří samostatný časový průběh signálů, který bude vykonán a bude sloužit pro posun požadovaných dat mezi FX2LP a externím zařízením.

Šest kontrolních OUT signálů

V 100- a 128-pinovém pouzdru je vyvedeno veškerých 6 kontrolních výstupních pinů (CTL0-CTL5). 8051 naprogramuje GPIF jednotku, aby nadefinoval CTL časové průběhy. V 56 pinovém pouzdru jsou vyvedeny 3 z těchto signálů, CTL0-CTL2. Hrany časového průbehu signálu CTLx mohou být naprogramovány pro převádění rychlostí 1/frekvence (20,8ns při f=48MHz).

Šest Ready IN signálů

V 100- a 128-pinovém pouzdru je vyvedeno veškerých 6 Ready vstupních pinů (RDY0-RDY5). 8051 naprogramuje GPIF jednotku pro testování RDY pinů pro GPIF větvení. V 56 pinovém pouzdru jsou vyvedeny 2 z těchto signálů, RDY0-1.

Devět GPIF adresních OUT signálů

Devět GPIF adresních linek je dostupných v 100- a 128-pinovém pouzdru, GPIFADR[8..0]. GPIF adresové linky umožňují indexování přímo až 512-byteových RAM bloků. Když je vyžadováno více adresních linek, používají se I/O porty.

Přenosový mód Long Transfer

V master módu, 8051 vhodně nastaví GPIF transakční čítací registr (GPIFTCB3, GPIFTCB2, GPIFTCB1 nebo GPIFTCB0) pro neobsluhovaný přenos transakcí až do 2^{32} . GPIF automaticky ubírá datový přenos aby předjel pod- nebo přetečení, dokud nedojde ke kompletní transakci požadavků. GPIF sníží hodnotu v těchto registrech pro reprezentace aktuálního stavu transakce.

9.15 ECC generace

EZ-USB umí vypočítat ECC (Error-Correcting Codes) dat, které prochází přes jeho GPIF, nebo Slave FIFO rozhraní. Existují 2 ECC konfigurace:
-Dva ECC, každý z nich vypočetno pro 256 byte-ů (SmartMedia™ Standard)
-Jeden ECC vypočten pro 512 byte-ů.

ECC umí korigovat jakoukoli jedno-bitovou chybu, anebo detekovat jakoukoli dvou-bitovou chybu.

ECC implementace

Dva ECC konfigurace jsou vybrané pomocí ECCM bitu:

ECCM=0

Dva 3 byte-ové ECC, každý z nich vypočteno pro 256 byte-ový blok dat. Tato konfigurace odpovídá SmartMedia Standard-u.

Se zapsáním jakékoli hodnoty do ECCRESETu se aktivuje výpočet ECC pro data, které jsou poslány přes GPIF nebo Slave FIFO rozhraní. ECC pro prvních 256 byte-ů dat bude vypočteno a uloženo v ECC1. ECC pro následujících 256 byte-ů dat bude uloženo v ECC2. Po vypočtení druhého ECC, hodnoty v ECCx registrech se nebudou měnit, dokud do ECCRESETu se nebude zas psát, i když bylo přenesených více dat za sebou přes rozhraní.

ECCM=1

Jeden 3-byte-ový ECC, vypočtený pro 512 bytový blok dat. Se zapsáním jakékoli hodnoty do ECCRESETu se aktivuje vypočítání ECC pro data, které jsou poslány přes GPIF nebo Slave FIFO rozhraní. ECC pro prvních 512 byte-ů dat bude vypočteno a uloženo v ECC1;ECC2 není používán. Po vypočtení druhého ECC, hodnota v ECC1 se nebude měnit, dokud se do ECCRESETu nebude zas psát, i když bylo přenesených více dat za sebou přes rozhraní.

9.16 USB Upload a Download

Jádro má schopnost přímo editovat datový obsah interního 16KByte-ového RAMu a interní 512byte-ové rychlé paměti (scratch pad) pomocí specifických příkazu. Tato schopnost se používá tehdy, když stahujeme uživatelský kód (soft downloading), a je dostupný jenom z/pro vnitřní RAM, a když 8051 je držen v resetu. Dostupné RAMové prostory jsou 16KByte-ů od 0x0000-0x3FFF (kód/data) a 512 byte-ů od 0xE000-0xE1FF (rychlá RAM paměť na data). Když data byli staženy od hostu, "loader" může spouštět kód z interní RAM za účelem přenášet stažená data do externí paměti.

9.17 Přístup Autopointer Access

FX2LP poskytuje 2 identické auto-ukazatele. Jsou podobné jako interní datové ukazatele 8051, ale s přídavnou vlastností: umí se inkrementovat volitelně po každém přístupu k paměti. Tato schopnost je dostupná do, i z obou externích a interních RAMů. Auto-ukazatelé jsou dostupné v externích FX2LP registrech, pod kontrolou jistého módu bitů (AUTOPTRSET-UP.0). Použitím externího FX2LP auto-ukazatelského přístupu (na 0xE67B-0xE67C) se umožňuje auto-ukazatelům přistupovat ke každé RAM, a to buď externí, nebo interní. Taky auto-ukazatelé mohou ukazovat na jakýkoli registr FX2LP, nebo na prostor v endpoint buffer. Když auto-ukazatelský přístup do externí paměti je povolen, prostor 0xE67B a 0xE67C v XDATA a kódovém prostoru se nemohou používat.

9.18 I^2C Kontrolér

FX2LP má jeden I^2C port, který je řízen pomocí dvou vnitřních kontrolérů, ze kterých jeden automaticky operuje při zaváděcím času, pro načtení VID/PID/DID a konfiguračních informací, a druhý, který 8051 používá k ovládání externích I^2C zařízení. Port I^2C pracuje jenom v master módu.

I^2C port piny

I^2C piny SCL a SDA musí mít externí 2,2Kohm-ové pull-up rezistory i když žádná EEPROM není připojena k FX2LP. Externí EEPROM piny musí být nastavěny správně. V tabulce 10 je vidět, jak se musí nastavit adresy zařízení.

Bajty	Příklad na EEPROM	A2	A1	A0
16	24LC00	N/A	N/A	N/A
128	24LC01	0	0	0
256	24LC02	0	0	0
4K	24LC32	0	0	1
8K	24LC64	0	0	1
16K	24LC128	0	0	1

Tabulka 10 - EEPROM nastavovací adresy
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

I^2C Rozhraní zaváděcího přístupu (Interface Boot Load Access)

Při resetování u zapínání (power-on reset), zavaděč I^2C rozhraní načte VID/PID/DID, a 16KBajtů programu/dat. Dostupné RAM prostory jsou 16KBajtů od 0x0000-0x3FFF a 512 bajtů od 0xE000-0xE1FF. 8051 bude v resetu. I^2C rozhraní se zavádí jenom po power-on resetu.

I2C Rozhraní hlavního účelového přístupu (Interface General-Purpose Access)

8051 umí ovládat periférie připojené k I^2C rozhraní použitím registrů I2CTL a I2DAT. FX2LP poskytuje I^2C jenom v master módu, nikoli v slave.

9.19 Kompatibilní s předešlými verzi EZ-USB FX2

EZ-USB FX2LP je tvarově totožný, a s výjimkami některých drobností funkčně kompatibilní s jeho předchůdcem EZ-USB FX2. Toto umožňuje konstruktérům jednoduché aktualizování jejich systémů z FX2 na FX2LP. Vývody čipu a pouzdru jsou stejné a většina programového vybavení původně navrženo pro FX2 bude fungovat i v FX2LP. FX2LP má hlavně větší vnitřní paměť.

9.20 Rozdíly mezi CY7C68013A/14A a CY7C68015A/16A

CY7C68013A je identický s CY7C68014A ve velikosti, tvaru a funkčnosti. CY7C68015A je identický s CY7C68016A ve velikosti, tvaru a funkčnosti. CY7C68014A a CY7C68016A mají menší přerušovací proud, jako CY7C68013A a CY7C68015A. Z toho důvodu jsou ideální při použití bateriových napájení.

CY7C68015A a CY7C68016A jsou dostupné jenom v 56-pinovém QFN pouzdru. Dva přídavné GPIO signály jsou dostupné v CY7C68015A a CY7C68016A pro poskytnutí větší flexibility, když žádný IFCLK ani CLKOUT nepotřebujeme v 56-pinovém pouzdru. USB vývojářům, který chtějí přestavět své FX2 56-pinové aplikace na sběrnicově napájené systémy, tyto přídavné signály přinášejí výhodu. Tyto dva GPIO dají těmto vývojářům signály, které potřebují k napájecím ovládacím obvodům pro sběrnicově napájené aplikace bez nutnosti použití více pinové verze FX2LP. CY7C68015A je dostupný jenom v 56-pinovém QFN pouzdru.

CY7C68013A/CY7C68014A	CY7C68015A/CY7C68016A
IFCLK	PE0/T0OUT
CLKOUT	PE1/T1OUT

Tabulka 11 - CY7C68013A a CY7C68015A rozdíly pinů
(zdroj: literatura [2] *EZ-USB FX2LP Datasheet*)

Pro zpracování návrhu desky plošného spoje v důsledku menší spotřeby elekrické energie jsem si zvolil mikrokontrolér CY7C68013A-128AXC.

10 FPGA

Programovatelné hradlové pole. FPGA jsou programovatelné digitální logické obvody. To znamená, že se dají naprogramovat na jakoukoli funkci. Práce s FPGA:

- Pro naprogramování funkcí FPGA čipu se používá počítač. Pomocí softwaru, se dá nakreslit požadované schéma, nebo opsát funkci (VHDL, nebo Verilog), kterou chceme naprogramovat do našeho FPGA.
- Po vytvoření konfiguračního souboru, se pomocí výrobcem dodávaného softwaru naprogramuje nově vytvořená funkce do FPGA.
- Připojí se FPGA k počítači pomocí kabelu a nahraje se do ní binární soubor.
- Od této chvíle FPGA obsahuje naši logickou funkci

Výhody jsou následující:

- FPGA se může naprogramovat tolik krát, kolik krát jenom chceme, a to po každé s jinou logikou. Když se udělá chyba v logické funkci, tak se jednoduše opraví, zas se zkompiluje a nahraje se do FPGA. Součástka se nemusí vypájkovat.

- Navržený design poběží mnohem rychleji, jak kdyby zapojení bylo vytvořeno z více diskrétních součástek, protože všechno běží v FPGA na jediné křemíkové desce.

- FPGA stratí svoji funkcionalitu, jakmile se odpojí napájení. V tom případě se zas musí nakonfigurovat.

FPGA čipy vyrábí 5 větších firem na světě, z toho první 2 jsou největší: Xilinx, Altera, Lattice, Actel, Quicklogic.

10.1 FPGA-jak funguje

FPGA je postavěno maximalně až z několika miliónů tzv. logických buněk. Tyto buňky se skládají ze 4 look up table (LUT), a z jedné D flip-flop logiky. Každá logická buňka se může připojit k jakékoli jiné buňce přes propojovací vedení. S touto metodou propojování malých buňek se pak získá komplexní logická jednotka. Tyto propojovací vedení vedou taky na okraj FPGA čipu, kde se mohou přepojit s I/O vsupy/výstupy a tak fungovat, jako vstupní, nebo výstupní buňky. Dnešní FPGA už mají také dedikované bloky statického RAM. K těmto blokům se dají nastavit různé typy přístupů, jako např. single-port RAM –v tomto případě může jenom jedna funkce psát, nebo číst do RAM, anebo dual- nebo quad-port RAM. V tomto případě mají přístup 2 nebo až 4 funkce k RAM, a to tak, že mohou mít i různá hodinové signály. Psaní do RAM se provádí většinou synchrónně, ale čtení může být synchronní, nebo i asynchronní.

FPGA mají mnoha pinů, které spadají do 2 kategorií:
1. Dedikované piny
2. Uživatelské piny

Asi 20%-30% pinů je dedikovaných, co znamená že jejich funkce je pevně dána. Tyto funkce mohou být např.: napájecí, konfigurační, vsup pro hodinový signál. Ostatní piny jsou plně programovatelné a mohou mít funkci vstupu, výstupu nebo obou. Tyto piny se jmenují IO piny. Dělíme je podle toho, jakou mají napěťovou úroveň logických hodnot. Tato skutečnost umožňuje, aby se k FPGA připojovali signály s různymi logickými úrovni. Vnitřek celého FPGA pak běží na určité napěťové hodnotě. FPGA návrhy bývají většinou synchronní, co znamená ,že zapojení je závislé na hodinovém signálu. Při každé rostoucí hraně, D flip-flop jednotky mohou změnit svích stavů. V synchronním návrhu jeden hodinový signál může řídit mnoho flip-flop jednotek, co může způsobit potíže. Proto výrobce většinou definují tzv. global lines. Tyto vodiče jsou uvnitř FPGA rozvedeny tak, aby distribuce hodinového signálu byla co nejideálnější (hrany signálu se dostávají na jednotky ve stejný okamih). Jiná možnost je taková, že se používa více hodinových signálů. Pomocí softwaru se analizují trasy, a zjistí se kde jaká frekvence se může používat. Pro kombinace takových různě rychlých dat se pak používají FIFO pamětě nebo synchronizéry.

FPGA se dají naprogramovat různymy způsoby. Při programování se může nacházet ve 2 stavech: konfigurační a uživatelský. Když se FPGA zapne, je ve stavu konfiguračním. Čeká na nastavění. Jakmile načteme do ní konfigurační soubor, dostane se do uživatelského stavu. Existují 3 typické metody naprogramování FPGA:

- Použití kabelu pro přepojení FPGA s PC, a následné naprogramování pomocí software

- Použitím mikrokontroléru na uživatleské desce, a adekvátního firmwaru se pošlou data do FPGA

- Použití boot-PROM, připojené k FPGA, které nastaví FPGA po zapnutí.

Při vývoji se používa 1. metoda, a v hotových aplikacích 2. nebo 3.

10.2 Konfigurace FPGA (Xilinx a Altera)

Nastavování FPGA obou výrobců je identická. Rozdíly jsou hlavně v pojmenování pinů, ale jejich funkce jsou většinou úplně identická. FPGA používají 3 typy rozhraní pro konfiguraci:

- JTAG rozhraní

- Synchronní sériové rozhraní

JTAG rozhraní

Původně bylo navrženo pro testovací a výrobní účely. Jak se ale elektronické desky časem zmenšily, otestování, jestli je deska dobrá se stávala čím dál, tím složitější. Primárním účelem JTAG je povolení převzetí kontroly nad všemi IO piny obvodu. Toto umožňuje otestování konektivity každého zařízení k druhým zařízením. Standardní JTAG příkazy se k tomuto účelu dají použít. FPGA obvody jsou ale vyspělejší a umožňují ovládání IO pinů pomocí JTAG rozhraní. Pomocí určitých JTAG příkazů se dají naprogramovat FPGA čipy.

JTAG se skládá ze 4 vodičů: TDI, TDO, TMS a TCK. Pátý pin TRST je volitelný. Jeden JTAG port se dá připojit k více zařízením, ovšem když to podporují. Pomocí více zařízení se tak dá vytvořit tzv. JTAG řetězec. TMS a TCK jsou spojeny se zařízeními přímo, ale TDI a TDO se zapojují do řetězce: TDO z jednoho čipu je zapojeno do TDI dalšího čipu v řetězci. Celý řetězec je pak ukončen při hlavním kontroléru (např. PC). TCK je hodinový signál, TMS se používá pro posílání příkazů do zařízení a TDI/TDO se používají pro přijem/odesílání dat. Každé zařízení v řetězci ma své vlastní ID, takže počítač vždycky ví které zařízení jsou dostupná.

V mém zapojení pro desku s FPGA je možná konfigurace pomocí JTAG, protože tyto konfigurační piny jsou vyvedeny na výstupní sběrnici.

Synchronní sériové rozhraní

Je to jednoduché synchronní rozhraní, kde se při jednom hodinovém cyklu přenese jeden bit. Rozpis funkcí pěti nejdůležitějších pinů:

Xilinx jméno	Altera jméno	Směr komunikace	Funkce pinu
data	data0	vstup do FPGA	Konfigurační datový bit
clk	dclk	vstup do FPGA	Konfigurační hodinový signál. Konfigurační datový bit je posunut v FPGA při rostoucí hraně hodinového signálu
prog_b	nConfig	vstup do FPGA	Když se nastaví (do nuly), FPGA dostane reset a ztratí své nastavění. Když se FPGA nacházelo v uživatelském módu, pak zastaví veškerou svoji činnost, a všechny IO piny se dostanou do tři-stavového režimu.
init_b	nStatus	výstup z FPGA	Tento pin signalizuje kdy je FPGA připraven na konfigurační proces, jakmile prog_b ukončí svou činnost.
done	ConfDone	výstup z FPGA	Když se nachází ve stavu high, tak to ukazuje, že FPGA je naprogramován (je v uživatelském módu).

Tabulka 12 - Synchronní sériové rozhraní
(zdroj: literatura [16] *fpga4fun.com - where FPGAs are fun*)

11 Vysokorychlostní komunikace pomocí GPIF rozhraní

Toto rozhraní umožňuje použitím zařízení CY7C68013A-128AXC dosáhnout teoretickou největší přenosovou rychlost až 96MB/s. Při přenášení dat se používa tzv. auto-transfer (automatický přenos) architektura, kde pomocí GPIF se přenáší data mezi mikrokontrolérem, externím slave zařízením, a USB host-em. Aby se dosáhla maximální přenosová rychlost, nesmí se do komunikace přímo zahrnout CPU, v našem případě 8051. Důvodem je to, že tato část by byla největší překážkou při dosažení high-speed přenosu. Každopádně procesor 8051 hraje velice důležitou roli i tak, protože on definuje jak má fungovat fyzické rozhraní, nastaví koncové body, spíná GPIF přenos, a pomocí něho se dají také data manuálně přenášet. Může také monitorovat vnější okolí, a podle nutnosti

regulovat přenosovou rychlost. Fyzické rozhraní běží na maximální rychlosti 48MHz, co při sběrnici o šířce max. 16 bitů činí 96MB/s.

Když se GPIF resetuje, pak po resetu I/O piny jsou nastavěny do Ports módu a ne do GPIF Master módu. Aby se piny nastavili do GPIF módu, IFCFG1:0 bity v IFCONFIG registru musí být nastavěny na logickou 1 a 0. Tento register se používa pro konfiguraci na rozhraní.

Jednou velikou výhodou GPIF je to, že umí generovat potřebné signály pro externí periférie, jako jsou např. čtecí/zapisovací cykly, časování. GPIF používá následující piny: nastavitelnou 8- nebo 16-bitovou sběrnici, kontrolní výstupy a ready vstupy. Detailněji jsou to následující (GPIF Master mód):

IFCLK (obousmerný) – je to referenční hodinový signál pro všechny GPIF operace. Může fungovat buď jako vstup, nebo výstup. Závisí na požadavcích systému. Když je řízen vnitřně, pak jeho výstupní frekvence může mít hodnotu buď 30MHz, nebo 48MHz. Když funguje jako vstupní signál, pak jeho rozsah je v rozmezí 5-48MHz.

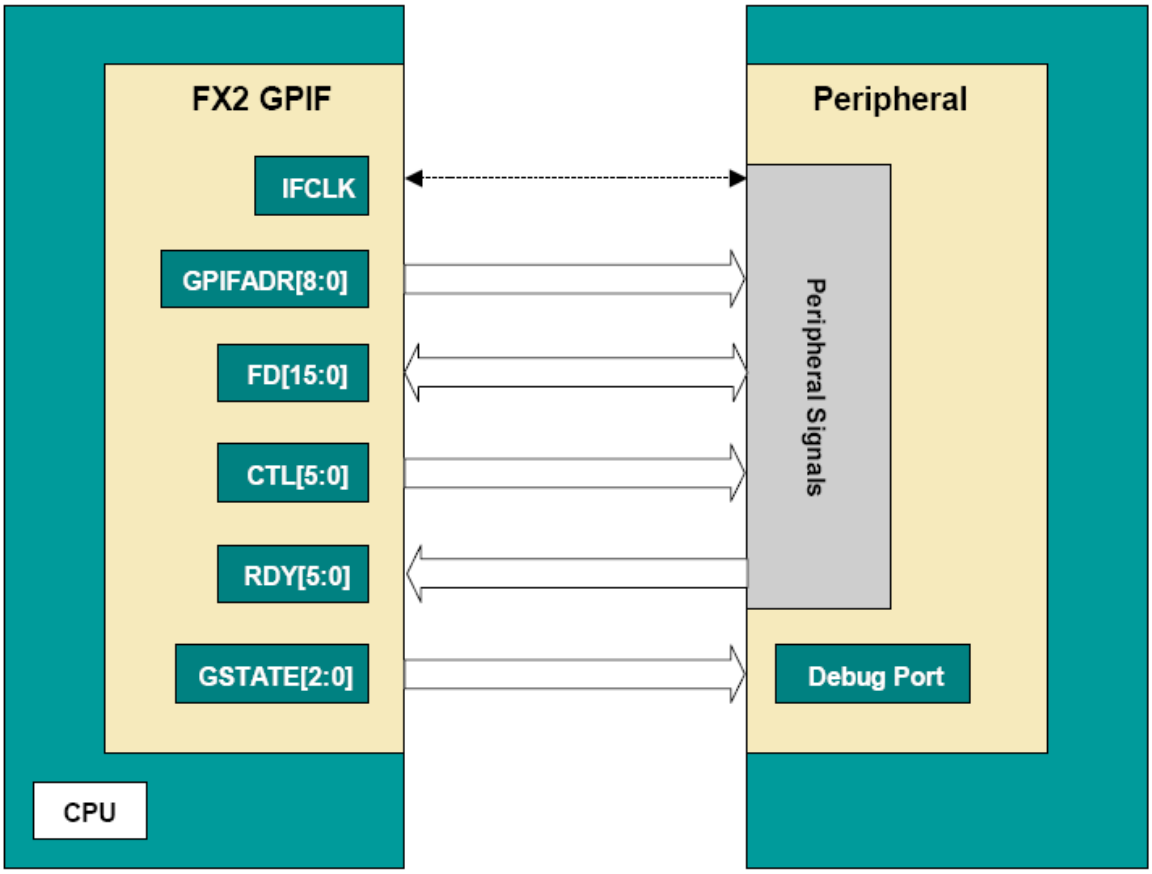
GPIFADDR[8:0] (výstup) – V zapojení PC[7:0] a PE[7]. GPIF rozhraní může využít tyto výstupy jako adresové linky pro externí periferie, které to vyžadují. Fungují pouze jako výstupní signály.

FD[15:0] (obousměrný) – V zapojení PB[7:0] a PD[7:0]. Je to dátová sběrnice použita rozhraním GPIF a je to vlastně přenosový kanál mezi mikrokontrolérem a externím zařízením. Může fungovat jako 8- nebo 16-bitová sběrnice, zavisí jenom na nastevění, a může být i tří statová, když to systém vyžaduje. V 16-bitovém módu FD[7:0] reprezentuje první byte ve FIFO koncovém bodu a FD[15:8] reprezentuje druhý byte ve FIFO koncovém bodu.

CTL[5:0] (výstup) – Jsou to kontrolní signály, které potřebují některé externí periferie. Tyto jsou např.: impulz pro čtení/zápis (RD/RW), povolení.

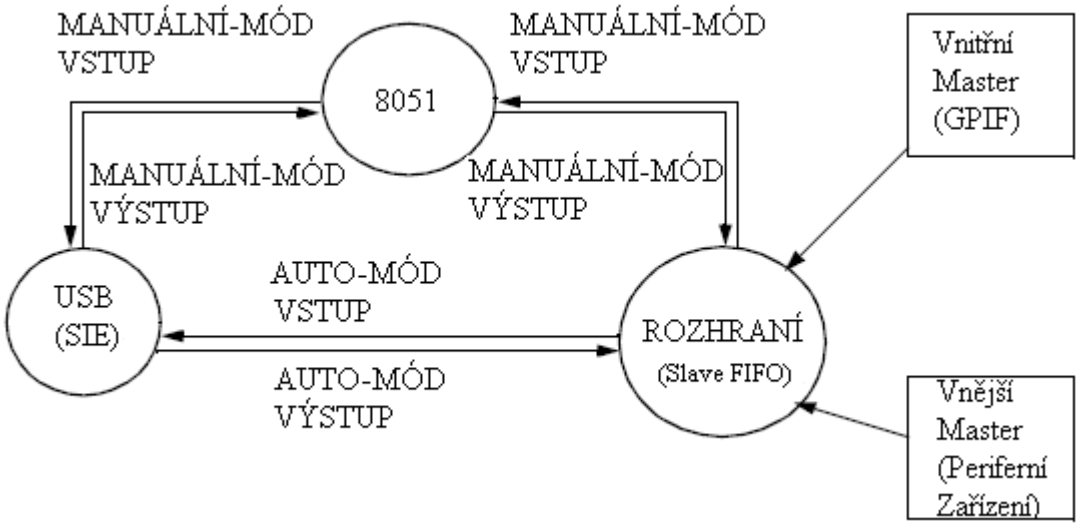
RDY[5:0] (vstup) – Jsou to vstupy pro monitorování stavů z externí periferie, jako jsou FIFO status flag, dostupnost dat, atd. Pomocí těchto signálů má GPIF možnost rozhodovat o jistých věcech.

GSTATE[2:0] (výstup) – Používají se při ladění. Ukazují vykonané GPIF časové průběhy signálů. Typicky se připojují k logickému analyzéru. Tyto piny jsou sdíleny s portem E.



Obrázek 13 - GPIF komunikace
(zdroj: literatura [9] *EZ-USB® FX2™ GPIF Primer*)

Aby se dalo komunikovat s externí periférií, musi se nastavit koncové body. Náš radič má 6 koncových bodů, které jsou: 0, 1, 2, 4, 6, a 8. EP0 je vždy konfigurační koncový bod. V určité době přístup k datům je ovládáno jednou ze tří oblastí: USB oblast, 8051 oblast a oblast rozhraní. Mikroradič poskytuje programátorovi 2 způsoby manipulace s buffery koncových bodů když se zaplní USB daty. Toto závisí na tom, jaký mód nastaví koncovým bodům procesor 8051, pomocí nastavění třetího a čtvrtého bitu v registru EPxFIFOCFG. Při zapínání, EP6 a EP8 jsou nastaveny jako double-buffered vstupní koncové body v manuálním módu, kdežto EP2 a EP4 jsou taky double-buffered, ale jsou výstupy. V tomto manuálním módu má 8051 počáteční přístup k datům v koncovém FIFO buffru. Když výstupní OUT koncový bod je nastavěn, hostitel může posílat data do ní. Při přijímání dat 8051 má přístup k přijatým datům, a může je změnit dle potřeby. Procesor může také posunout data do oblasti rozhraní. Jakmile se data přesunou do oblasti rozhraní, 8051 ztrácí přístup k FIFO buffrům a dostává přístup k datovýmu buffru. Rozhraní je řízeno vnitřním, nebo vnějším mastrem. GPIF je vnitřním mastrem pro 8051, a je také vždycky master pro FIFO (slave). FIFO může nabýt master módu pomocí externí periferie. Jak už bylo zmíněno, aby se dosáhlo high-speed rychlosti, data se můžou posunout přímo z USB oblasti do oblasti rozhraní. Tato možnost nastává, když FIFO jsou nastavěna do auto-módu. Následující obrázek ukazuje teoretické uspořádání všech domén, a jejich vzájemné vztahy.



Obrázek 14 - Vzájemný vztah třech oblastí

(zdroj: literatura [10] *Introduction to the EZ-USB FX2™ GPIF Engine*)

11.1 AUTO-MÓD

Příjem dat

Když se 3. bit (AUTOIN) EPxFIFOCFG registru nastaví pomocí 8051 na logickou 1, data ve FIFO buffru jsou automaticky a okamžitě připojeny ke koncovému FIFO bodu. Příznakové bity FIFO koncových bodů a čítače buffrů současně signalizují změnu ve stavu FIFO. Když hodnota bytů ve FIFO přesáhne nastavěnou maximální hodnotu ve EPxAUTOINLEN registru, jádro automaticky „předá" data z oblasti rozhraní do USB oblasti. V tomto AUTO-MÓDu kdy EOPxAUTOINLEN je specifikovaný, externí master může přenášet data nepřetržitě přes FIFO, skrz USB, až k hostiteli. Když velikost packetu není násobkem velikosti packetu specifikovaném v EPxAUTOINLEN, tak poslední dloužka tohoto packetu bude menší, jak je definováno v EPxAUTOINLEN registru, a z toho důvodu nebude automaticky odevzdán USB oblasti. Aby se předal i tento poslední packet, master může udělat 2 věci:

1. Vyplnit packet neužitečnými daty, aby se dosáhlo velikosti specifikovaném v EPxAUTOINLEN.

2. Zaznamenat tento krátký packet do FIFO a potom nastavit PKTEND pin.

Odesílání dat

Když 8051 nastaví čtvrtý bit (AUTOOUT) registru EPxFIFOCFG, data v buffru koncového bodu jsou automaticky a okamžitě připojeny ke koncovému FIFO bodu, a příznakové bity FIFO a čitače buffrů okamžitě signalizují změnu ve stavu FIFO. V tomto módu jsou data přenášeny z oblasti hostovacího zařízení do oblasti periferní.

11.2 MANUÁLNÍ-MÓD

Když 8051 nastaví 3. bit (AUTOIN pro vsupní koncový bod) anebo 4. bit (AUTOOUT pro výstupní koncový bod) v registru EPxFIFOCFG na logickou 0, 8051 přijme přerušení pro buffer, aby se zaplnil s daty z USB. Připojení buffru ke koncovému bodu se dostane pod kontrolu 8051. Tento mód se nazývá manual-mode. V tomto módu má 8051 na starosti předávání dat, když počet bytů v buffru dosáhne hodnotu nastavěnou v registru EPxAUTOINLEN.

Příjem dat

Pro příjem existujou 3 možnosti předávání dat, jakmile se vstupní buffer koncového bodu zaplní. 8051 může provádět jednu z následujících možností:

1. Zápis do byte count registru.

2. Zápis do INPKTEND registru čísla vstupního koncového bodu.

3. Použití PKTEND pinu. Vnější master nastaví PKTEND pin pro předávání vstupních packetů pro USB bez ohledu na délku packetů. PKTEND se většinou používá tehdy, když master chce posílat „krátký“ packet (menší, jak je specifikován v registru EPxAUTOINLEN).

Použití druhé, nebo třetí metody je nejrychlejší, protože vyžadují psaní do jediného registru. Použití byte count registrů vyžaduje psaní do dvou registrů, co znamená více instrukčních cyklů.

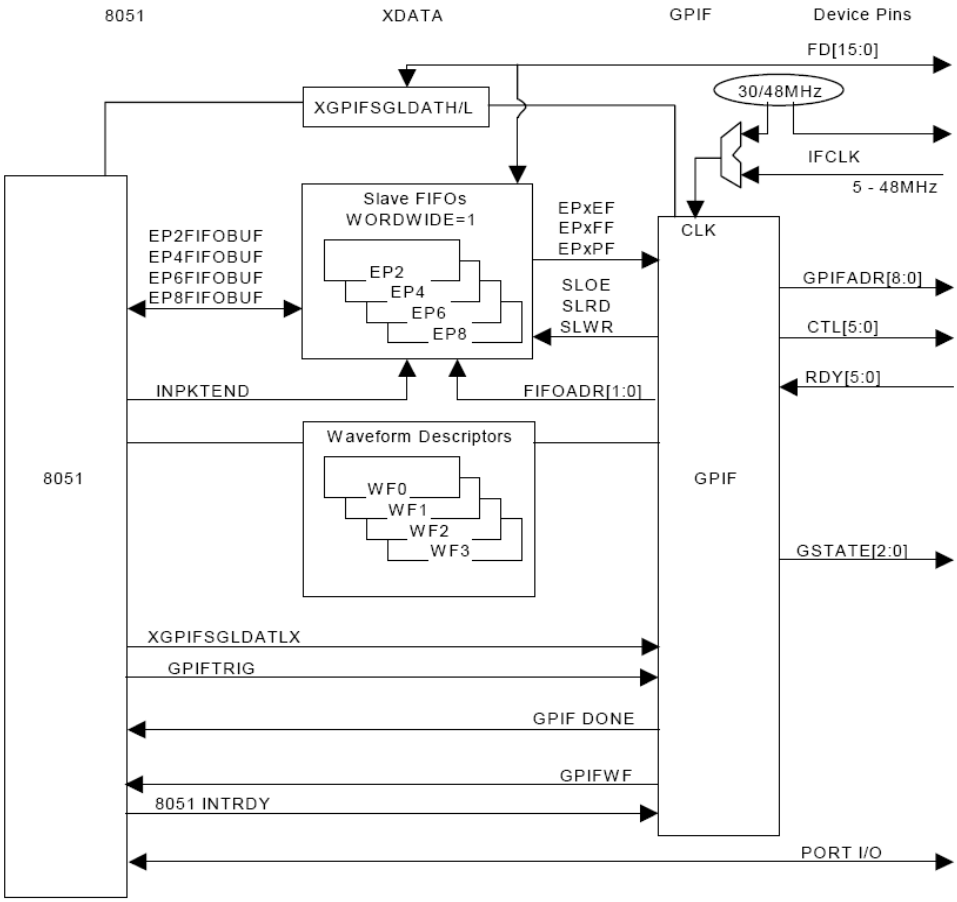
Odesílání dat

V manuálním módu při vysílání dat, 8051 potřebuje psát do byte count registru se SKIP bitem nastavěným na 0, aby mohl předat data pro master. Když SKIP bit je nastavěn do logické 1, data jsou jednoduše ignorovány. Psaním do byte count registru se koncový bod znovu inicializuje.

11.3 Časový průběh signálů

Časové průběhy signálů GPIFu jsou uživatelem programovatelné, a jsou zakódované v sigálových deskriptorech. Signálové deskriptory jsou instrukce pro GPIF rozhraní. Ty „řeknou“ GPIFu co udělat, když je spuštěn signál. Signál se skládá ze sedmi programovatelných intervalů (stavů) a z jednoho IDLE (nečinného) stavu. Každý jeden takový průběhový signál spotřebuje 32 bytů paměti. Čtyři byty jsou použity pro popis každého intervalu. Tyto 4 byty obsahují stavy CTLx výstupních signálů, stavy datové a adresní sběrnice, stavy RDYn vstupních signálů a informaci, jestli je interval rozhodovací bod, nebo ne. Bloková schéma na následujícím obrázku ukazuje kde se ve systému nachází GPIF.

viz. další strana



Obrázek 15 - GPIF v systému

(zdroj: literatura [10] *Introduction to the EZ-USB FX2™ GPIF Engine*)

Seznam registrů přirazeních k GPIF hardware:

GPIFIDLECS	IFCONFIG
GPIFIDLECTL	FIFORESET
GPIFCTLCFG	EPxCFG
PORTCCFG	EPxFIFOCFG
PORTECFG	EPxAUTOINLENH/L
GPIFADRH/L	EPxFIFOPFH/L
GPIFTCB3:0	
GPIFWFSELECT	EPxTRIG
EPxGPIFFLGSEL	GPIFABORT
EPxGPIFPFSTOP	XGPIFSGLDATH/LX/LNOX
GPIFREADYCFG	GPIFSGLDATH/LX/NOX
GPIFREADYSTAT	GPIFTRIG

(písmeno x značí číslo koncových bodů 2,4,6,8. 0 a 1 nejsou přiřazeny k GPIF)

Tabulka 13 - Registry přiřazené k GPIF

(zdroj: literatura[3] *EZ-USB Technical Reference Manual*)

11.4 GPIF přenosový mód

Tato část popisuje jak GPIF používá FIFO architekturu pro příjem a odesílání mezi připojenou periferií a hostovacím zařízením. 8051 umí spouštět 3 typy GPIF přenosů. Součástí nastavovacího procesu GPIFu je, že 8051 inicializuje GPIF registry a deskriptory časových průběhů signálů. Toto se provádí utilitou zvanou GPIFtool.exe, která je dostupná u výrobce mikrokontroléru © Cypress Semiconductor Corporation. Před spouštěním GPIF přenosu je důležité se ujistit, že GPIF je v nečinném stavu a není v činnosti žádný přenos. Hodnota bitu GPIFDONE v registru GPIFTRIG

signalizuje jeho stav, a měla by mít hodnotu 1 před spouštěním datového přenosu. Není dovoleno psát do žádného registru GPIF (ani do registrů popisujících průbehy signálů), dokud je GPIF v zaneprázněném stavu. Jediná výjimka je GPIFABORT, která se používá k násilnému přerušení aktuálního průběhu signálů. Zápis do kteréhokoli GPIF registru během přenosu může způsobit poškození dat. Jsou tady 2 možnosti přenosu, které GPIF umí spouštět. Jsou to následující:

- FIFO burst přenos
  - Periferní FIFO čtení
  - Periferní FIFO zápis
- Jednoduchý přenos
  - Jednotlivé čtení
  - Jednotlivý zápis

Pseudokód pro výše zmíněné GPIF přenosy se nachází v následující části. Tento kód shrnuje logické kroky, které jsou zahrnuty ve vykonávání GPIF transakce.

Periferní FIFO zápis

Zahrňuje zápis jednoho bytu/wordu na GPIF datovou sběrnici počas spuštěného průběžného signálu. Procesor 8051 umí spouštět takový přenos pomocí těchto registrů:

- GPIFTRIG: zápis čísla FIFO koncového budu do registra s R/W bity pro zápis.
- EPxGPIFTRIG (kde x je číslo koncového bodu): zápis jakékoli hodnoty do tohoto registru.

Použití GPIFTRIG registru pro spouštění průběhového signálu je rychlejší než zápis do EPxGPIFTRIG registru, protože GPIFTRIG je SFR (Special Function Register).
Následující pseudo kód slouží k manipulaci s výstupními daty:

```
If endpoint not empty (if data available in the buffer)
        If GPIFidle
                **Address slave**
                If slave not full
                        Transaction count = Endpoint byte count
                        Commit packet to master
                        Re-arm endpoint (If in manual mode)
                        Trigger FIFO write transaction
                        If short packet
                                Wait for GPIF transfer to complete
                                Signal short packet to slave
```

Periferní FIFO čtení

Zahrnuje čtení více jak jednoho bytu/wordu na GPIF datovou sběrnici počas spuštěného průběžného signálu. Procesor 8051 umí spouštět takový přenos pomocí těchto registrů:

- GPIFTRIG: zápis čísla FIFO koncového budu do registra s R/W bity pro čtení.
- EPxGPIFTRIG (kde x je číslo koncového bodu): čtení tohoto registru zahájí FIFO čtecí transakce.

Použití GPIFTRIG registru pro spouštění průběžného signálu je rychlejší než zápis do EPxGPIFTRIG registru, protože GPIFTRIG je SFR.

Následující pseudo kód slouží k manipulaci se vstupními daty:

```
If GPIF idle
        **Address slave**
        If slave not empty ** or slave fifo buffer not empty **
                If fifo buffer available
                        Trigger FIFO read transaction
                        Wait for transfer to complete
                        If endpoint buffer available
                                Commit packet to host
```

Jednotlivý zápis

Zahrňuje zápis jednoho bytu/wordu na GPIF datovou sběrnici počas spuštěného průběžného signálu. Procesor 8051 umí spouštět takový přenos pomocí těchto registrů:

- XGPIFSGLDATLX: zápis do tohoto registru působí, že tyto zapsaná data se dostanou na výstupní sběrnici GPIF (FD[0..7] pro přenos bytu, 8-bitový mód). Pro přenos wordu (16-

bitový mód), tento zápis musí být vykonán tak, že vyšší byte se musí zapsát do XGPIFSGLDATH.

Následující pseudo kód slouží k manipulaci s výstupními daty při jednotlivém zápisu:

```
If GPIFidle
        Trigger single write ( write to the XGPIFSGLDATL)
        Wait for transfer to complete
```

Jednotlivé čtení

Zahrňuje čtení jednoho bytu/wordu z externí periferie (přes GPIF datovou sběrnici) počas spuštěného GPIF signálu. V 16-bitovém módu MSB je získán z XGPIFSGLDATH registru. LSB je získán z XGPIFSGLDATLX a/nebo z XGPIFDSGLDATLNOX.
Procesor 8051 umí spouštět takový přenos pomocí těchto registrů:

- XGPIFSGLDATLX: výsledkem toho je přečtení LSB, a zahájení dalšího jednotlivého čtení.
- XGPIFDATLNOX: výsledkem je přečtení LSB, a ukončení čtení.

Následující pseudo kód slouží k manipulaci se vstupními daty při jednotlivém čtení:

```
If GPIF idle
        Trigger Single read (reading of the XGPIFSGLDATLX)
        Wait for transfer to complete
```

Implementace

Firmwareový soubor, který potřebujeme ke komunikaci se dá buď to vytvořit, nebo použít již vytvořenej. Existují hotové základy návrženého firmware, které jsou dostupné v Keil uVision2 projektu. Bohužel software od Keil je placený. Použití těchto hotových základů usnadňuje, a hlavně urychlý další vývoj systému. Rozdělujeme 2 hlavní části pro zhotovení GPIF aplikačního řešení:

1. firmware vyššího stupně, který nastaví GPIF a spustí přenos
2. GPIF deskriptor pro průběhy signálů, který provádí časování fyzické sběrnice

První část se skládá normálně z 5 souborů (fw.c, periph.c, dscr.a51, ezusb.lib, usbjmptb.obj), které se nachází v Keil uVision 2 Firmware Frameworks projektu. Druhá část, kterou tvoří samostatný gpif.c soubor, obsahuje GPIF deskriptory signálů a je přidáván ke Keil projektu. Tento soubor se generuje pomocí grafické aplikace, zvané GPIF Designer [18]. Aplikace umožňuje implementaci různě nastavěných signálů, které se pak dají přidělovat ke 4 typům průběhových signálů. Tyto jsou: Single Write, Single Read, FIFO Write, a FIFO Read. Přidělování se nastavuje pomocí registru GPIFWFSELECT. Každý takový popisovač signálů má velikost 32 bytů, a nachází se v paměťi čipu, když se načte procesorem. Když jsou tyto GPIF signálové deskriptory připraveny, tak speciální GPIF registr vybere který ze 4 typů má spouštět GPIF jednotka. Jako první krok by se mělo implementovat jednoduché čtení a zápis, z toho důvodu, že je koncepčne jednodušší jeho návrch, a můžeme se díky tomu snadno přesvědčit o funkčnosti fyzického kontaktu dvou systémů.

Při implementaci FIFO přenosů je rozumné postupovat tak, že nejdřív se vytvoří část FIFO Write, a pak se zkontroluje, jestli správně funguje přenos. Pak jako krok 2 se vytvoří FIFO Read. Tento postup je důležitý z toho důvodu, že kdybychom všechno implementovali naráz, tak když se vyskytne chyba, těžko zjistíme v které části chyba vznikla.

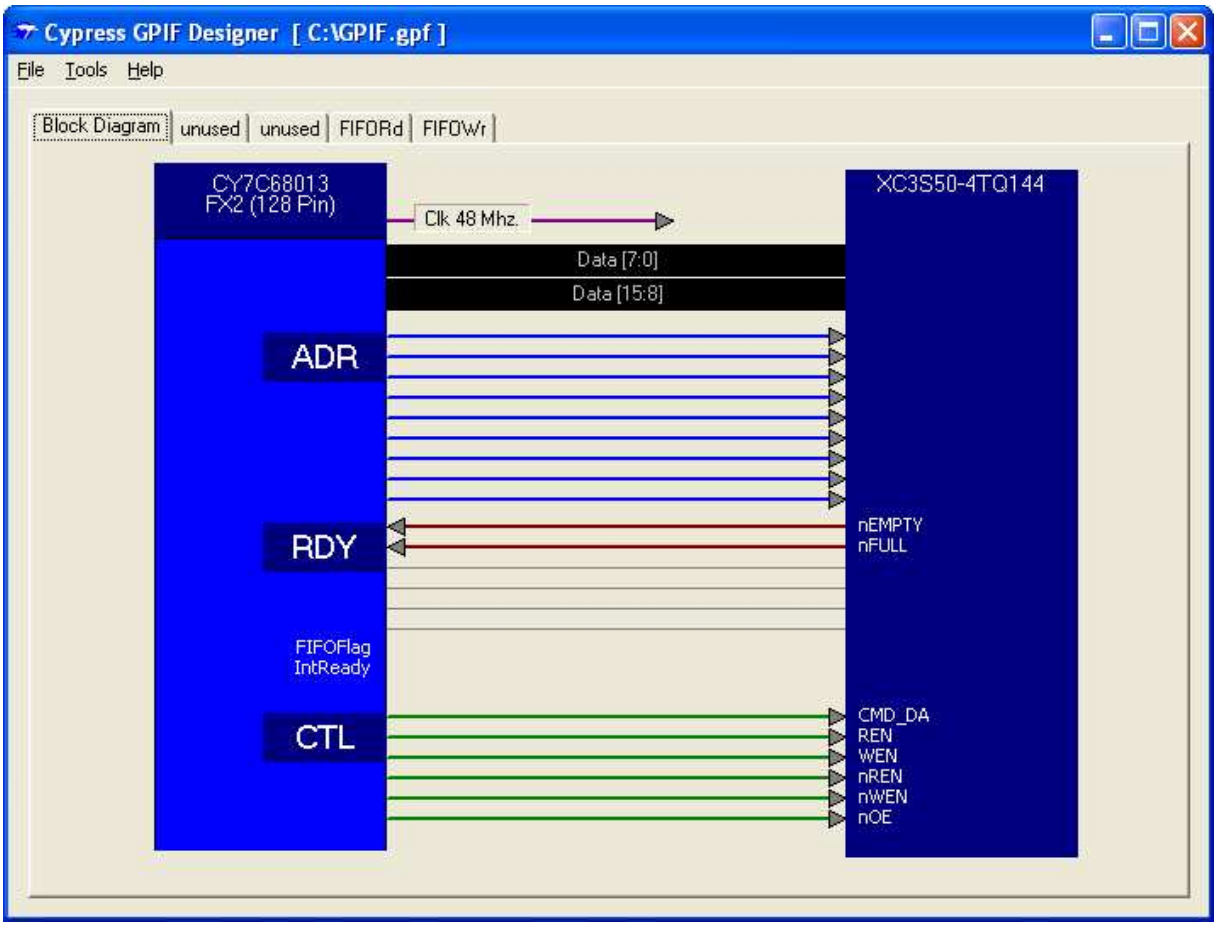
Pro komunikaci mezi GPIF a FPGA by se muselo nadefinovat v GPIF Designeru několik vstupů a výstupů. Tyto jsou následující: IFCLK, GPIFADR, PB[7:0] a PD[7:0], CTL[5:0], a RDY[1:0]. Podrobněji:

IFCLK je hodinový signál pro synchronní přenos.
GPIFADR [8:0] (PC0..7, PE7) je adresovací výstup pro periferii, když to potřebuje.
PB[7:0] a **PD[7:0]** je 16 bitová datová sběrnice.
nEMPTY (RDY0) ukazuje prázdný stav FIFO, vybratého pomocí FIFOADR[1:0].
nFULL (RDY1) ukazuje zaplněný stav FIFO, vybratého pomocí FIFOADR[1:0].
CMD_DATA (CTL0) je Command Enable výstup. Pomocí něj se indikuje, jestli se bude přenášet data, nebo se budou přenášet příkazy. Používá se pro nastavování registrů v připojené periferii, jako např. FPGA.
REN nebo **nREN** (CTL1 a CTL3) dokud tento výstup je ve stavu high (low), data se čtou z externí periferie.

WEN nebo **nWEN** (CTL2 a CTL4) dokud tento výstup je ve stavu high (low), data se zapíšou do externí periferie.

nOE (CTL5) Output Enable signál, pro připojenou periferii.

Následující obrázek ukazuje jak vypadá celá záležitost v GPIF Designeru:



Obrázek 16 - GPIF konfigurace

(zdroj: literatura [18] *GPIF Designer*)

Po nastavění těchto základních věcí se pak musí nastavit ještě FIFORd a FIFOWr. Ty je možné nastavit pomocí příkladů v literatuře [9], nebo vyvíjet vlastní řešení. Po nastavění všech důležitých parametrů se vytvoří pomocí GPIF Designeru požadovaný soubor gpif.c. Tento soubor spolu s ostatními zdrojovými soubory se pak pomocí Keil uVisionu zkompiluje na soubor s příponou .hex. K poslednímu kroku, čili download hex programu do zařízení se použije EZ-USB Development Kit, který je volně stažitelný ze stránek Cypress Semiconductors. Pomocí této aplikace je možné odladit firmware.

12 Zapojení obvodu

Zapojení vychází z různích zdrojů, které jsem prostudoval, a poznatky použil při návrhu systému. Seznam těchto zdrojů se nachází v seznamu zdrojů pod čísly: [12], [13], [14]. Zapojení by bylo možné zhotovit na 4 vrsvé PCB, jak doporučuje výrobce. Pro dosažení impedance 90ohm mezi datovými vodiči USB jsem použil údaje dostupné na webu [16]. Bohužel stránka neuvádí materiál desky plošného spoje, ale doporučené nastavění které stanoví v plné míře dodržuju. Pod vrchní vrstvou se nachází vrsva GND, která je 0,3mm pod touto tzv. signálovou vrstvou. Pod vrstou GND se nachází VCC vrstva, která slouží jako napájení pro různá části obvodu. Vzdálenost mezi těmito vrstvami je 0,8mm. Pak už následuje spodní vrstva, která se nazývá signálová, a vzdálenost mezi ní a

VCC je taktéž 0,3mm jak v případě prvních dvou vrstev. Popis výstupní sběrnice se nachází v příloze A.

Napájení a zem

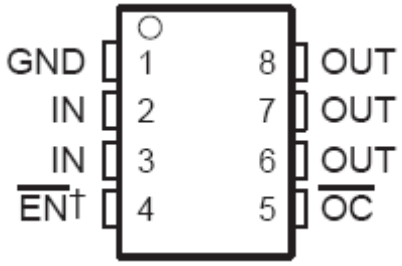
Tato deska může dodat při +5V maximálně 400mA proudu z VBUS pro napájení připojené desky s FPGA, nebo jiné zařízení. Pro spínání napětí se používá FET tranzistor. Napájení je standardně vypnuto a zapíná se až po tom, jak HOST počítač nastaví modul. Tato reakce je vyžadována od USB specifikace. Modul připojuje limitovač proudu, který odpojí VBUS napájecí vodiče, jakmile nastane situace proudového přetížení. V tomto případě, když modul překročí maximální přidělenou hodotu proudu 500mA, celý modul vypne buď HOST počítač, nebo USB HUB.

V případě že připojená deska odebírá míň, jak 400mA, může se napájet z neregulovaného 5V z pinů VBUS. Jakmile připojené zařízení bude odebírat o 50mA víc, musí se přidat spínač napájení do zapojení, jako např. TPS2051A, nebo high enable logikou přepínaný regulátor napětí. Enable signál se připojuje na pin SW_PG. Tento signál pak aktivuje regulátor napětí připojené desky, když se připojí VBUS na zařízení, a výstupní napětí se ustálí na hodnotě ≥93% napětí z USB.

Když připojená deska odebírá více jak 400mA, nesmí se napájet z VBUSu. V tomto případě by měla být připojena na externí zdroj napětí, a digitální zem by se měla spojit s GND připojeného zařízení.

Deska s čipem FPGA

Jako přílohu ke své práci přidávam zapojení desky s čipem FPGA, konkrétně model od firmi Xilinx XC3S50-4TQ144. Zapojení vychází ze schémy, která je dostupná na webu jistého projektu (literatura [12]). Toto zapojení bere napájecí napětí ze sběrnice USB, ovšem pomocí několik vylepšení by bylo možné udělat napájení ze samostatného zdroje, které by automaticky přeplo na externí zdroj, kdyby odběr přesáhl dovolené meze. Důvodem, proč není na této desce v základu možnost připojení vnějšího napájení je asi to, že by napájecí proud nepřesahoval celkových 400mA pro FPGA. Každopádně kdybychom chtěli větší napájení, pak by se do zapojení musela přidat např. součástka od Texas Instruments TPS2051A. Součástka má funci proudovýho limitovače, a přepínače napájení. Umí dodávat sice jenom 500mA, ale existují i přepínače se 4 různými hradly, které by mohli dodávat i 2A. Pro pochopení funkce postačuje tento jednoduchý model. Součástka vypadá následovně:



Obrázek 17 – TPS2051A

(zdroj: literatura [17] *CURRENT-LIMITED POWER-DISTRIBUTION SWITCHES*)

Má dva vstupy, na které se může připojit napětí v rozmezí 2,7V-5,5V. V mém případě bych potřeboval stabilizované napětí 5V. Na výstupech, na které se přepíná zdrojové napětí se pak může dodávat proud 500mA. Ovšem u této součástky jsou výstupy jenom rozděleny, takže při spojení všech třech stále zůstává maximální proud 500mA. Přepínače, které spínají proud mají prudový limit nastavěno na 0,9A. Na desce s USB mikrořadičem, pomocí výstupu SW_PG by bylo možné spínat napájení z externího zdroje. Pin SW_PG by se v tomto případě měl připojit ke vstupu nEN. Je to aktivní low, jak výstup SW_PG, takže bez dalších součástek by bylo možné tyto 2 vsupy/výstupy propojit. Externí napájení by se připojovalo tehdy, kdyby úroveň napětí z USB kleslo pod 88%. Výstupy řady TPS20xx je také možné spojit, a tak můžeme dostat jeden výstup s maximálním proudem až 2A, např. u modelu TPS2054A. U tohoto typu jsou 4 různé nEN vstupy, na které by bylo potřeba připojit SW_PG. Pin nOC (Over Current) slouží k signalizaci stavu, když se součástka přehřeje, nebo když je odebýrán příliž veliký proud, případně zkrat. Po vychlazení, nebo odpojení spotřebiče se však součástka dostane do normálního funkčního stavu.

Sběrnicový opakovač

Tato součástka se jmenuje PCA9515A I^2C Bus Repeater. Je přidána do zapojení z toho důvodu, aby se nemusela přidávat na připojenou desku, kde by se použila součástka používající sběnici I^2C, která není napájena z USB VBUS. Komunikace přes tuto sběrnici je zahrnuta do zapojení, jenom jako přídavek, pro případné použití.

13 Závěr:

Projekt měl za úkol zpracovat návrh obvodu umožňujícího komunikaci počítače PC rozhraním USB 2.0 v režimu high-speed s obvodem FPGA s využitím mikrořadiče CY7C6801xA. Písmeno x může být 3,4,5,6 co značí o jakou součástku přesně jde. Na připojené desce se předpokádá použití programovatelného obvodu FPGA a to na rychlosti využívající přenosovou kapacitu USB 2.0 sběrnice, čili 480Mbit/s.

Nejdřív se projekt zaměří na samostatnou USB sběrnici. Popisuje se tady její funkčnost, kódování dat, způsoby přenosů, atp. Tato část slouží jenom pro lepší pochopení protokolu USB 2.0.

V té druhé části se práce zaměřuje na specifickou řadu mikrořadičů, které jsou určeny pro vysokorychlostní datový přenos. Po přečtení různych literatur a uvažování jsem si následně zvolil CY7C68013A-128AXC. Ta se vyznačuje nízkou spotřebou energie, a 128 pinová verze poskytuje dostatečné množství různých vsupů a výstupů. Pro lepší pochopení vnitřní struktury proto rozeberu nejdůležitější části této součástky.

Poslední část se zaobírá přesnějším opisem části mikrořadiče, která má za úkol zabezpečit komunikaci mezi počítačem a připojenou uživatelskou deskou. Pak už zůstává jenom vlastní návrh desky. Většinou není veliký problém najít způsob jak přepojit FPGA obvod s počítačem, ale když se vyžaduje tak veliký datový tok, tak se musí hledět už na více detailů. Tyto detaily jsou hlavně problémem kolem mikrořadiče, protože tam běží přenos dat sériově, a to pomocí vysokých kmitočtů, při kterých se už při návrhu musí dát pozor. Sice 480MHz ještě nepatří typicky do kategorie mikrovln, ale musí se dodržet doporučení, které poskytuje výrobce, nebo je otestována třetími stranami. Já jsem zvolil druhou možnost, která se nachází v literatuře [15]. Při návrhu se také doporučuje použití čtyřech vrstev plošného spoje, ale připadá do alternativy taky použití dvoustranního plošného spoje, což by ale vyžadovalo určité množství výpočtů, a/nebo experimentů. Z tohoto důvodu jsem zvolil výrobcem doporučenou metodu, která by cenově vycházela dráž, ale s velikou pravděpodobností by také fungovala. Přepojení navržené desky s FPGA by tak kritické už nebylo. Při tom by se použila sběrnice, která je velice flexibilní, a hlavně rychlá, GPIF. Tam se může použít maximální frekvence při přenosu až 48MHz. Takový kmitočet činí při 16 bitové maximální šířce sběrnice propustnost 768Mbit/s. Takovou přenosovou rychlost ovšem samotný sériový přenosový kanál USB 2.0 směrem k počítači nezvládá. Vývody pro připojenou desku tvoří řada pinů, ke kterým se připojuje vhodně navržený obsloužený systém. Popis vývodů se nachází v příloze A. Na výstup desky lze připojit různá FPGA od firem Xilinx, nebo Altera. Důvodem je to, že firmy vyrábějí součástky s identickými vstupními a výstupními vývody. Software pro komunikaci (firmware) lze napsát pomocí nástrojů výrobce Cypress Semiconductor Corporation, anebo také lze použít volně dostupné, nebo třetími stranami vyvinuté softwary. Zapojení by se pak dalo využít pro přenos videa ve vysoké kvalitě, pro vzorkování a zpracování signálů, atd. Jako další krok jsem vyvinul desku s čipem FPGA, které vychází ze zapojení z literatury [12]. Návrh desky plošného spoje se nachází v přílohách.

14 Literatura:

[1] Matoušek, D. *USB prakticky s obvody FTDI.* BEN, Praha, 2003.

[2] Cypress Semiconductor Corporation, *EZ-USB FX2LP Datasheet*, Cypress Semiconductor Corporation, Dostupný z WWW: <http://download.cypress.com.edgesuite.net/design_resources/reference_designs/contents/cy4611b___usb_2_0_usb_to_ata_reference_design_19.zip> , 2005.

[3] Cypress Semiconductor Corporation, *EZ-USB Technical Reference Manual*. Cypress Semiconductor Corporation, Dostupný z WWW: <http://www.keil.com/dd/docs/datashts/cypress/fx2_trm.pdf > , 2000.

[4] URBIŠ, Hynek. *USB - Univerzální Sériová Sběrnice*, 27.4.2000.

[5] Compaq Computer Corporation, Hewlett-Packard Company, Intel Corporation, Lucent Technologies Inc, Microsoft Corporation, NEC Corporation, Koninklijke Philips Electronics N.V., *Universal Serial Bus Specification,* Dostupný z WWW: <http://www.usb.org/developers/docs/usb_20_040908.zip>, 27.4.2000.

[6] Intel Corporation, High *Speed USB Platform Design Guidelines Rev. 1.0*, Dostupný z WWW: <http://www.usb.org/developers/docs/hs_usb_pdg_r1_0.pdf>, 7.12.2000.

[7] Ing. FRÝZA, Tomáš, Ph.D., *Mikroprocesorová technika* – studijní opora Brno, 2006.

[8] *Wikipedie otevřená encyklopedie*, Dostupný z WWW: <http://en.wikipedia.org>, 29.4.2008.

[9] Cypress Semiconductor Corporation, *EZ-USB® FX2™ GPIF Primer*, Cypress Semiconductor Corporation, Dostupný z WWW: <http://pages.cpsc.ucalgary.ca/~walpole/525/FRIESS%20and%20MCNEIL/datasheets/FX2/FX2_GPIF_Primer.pdf>, 2003.

[10] Cypress Semiconductor Corporation, *Introduction to the EZ-USB FX2™ GPIF Engine*, Cypress Semiconductor Corporation, Dostupný z WWW: <http://pages.cpsc.ucalgary.ca/~walpole/525/FRIESS%20and%20MCNEIL/datasheets/FX2/FX2_IntroToGPIF.pdf>, 2002.

[11] Cypress Semiconductor Corporation, *EZ-USB FX2 Technical Reference Manual*, Cypress Semiconductor Corporation, Dostupný z WWW: <http://www.keil.com/dd/docs/datashts/cypress/fx2_trm.pdf>, 2001.

[12] *FPGAz Wiki - USB FPGA Hardware - Schematics*, 2007 Dostupný z WWW: <http://www.fpgaz.com/usbp/usbp_fpga.pdf>, 2007.

[13] WIESER, Wolfgang, *Electronics -- USB-FX2 Interface Board (USB-2.0)*, Dostupný z WWW: <http://www.triplespark.net/elec/periph/USB-FX2/circuit.html>, 2006

[14] Bitwise Systems, *QuickUSB Module*, Dostupný z WWW: <http://www.quickusb.com/store/media/quickusb_schematic_rev_b1.pdf>, 2008.

[15] Michael M. Abraham and David Luke, *USB 2.0 Printed Circuit Board Design*, Dostupný z WWW: <http://www.cqpub.co.jp/DWM/article_english/dwm050/USB2_01.htm>, 2001.

[16] Jean P. Nicolle, fpga4fun.com - where FPGAs are fun, Dostupný z WWW: <http://www.fpga4fun.com>, 2008.

[17] Texas Instruments Incorporated, *CURRENT-LIMITED POWER-DISTRIBUTION SWITCHES*, Dostupný z WWW: <http://www.datasheetcatalog.org/datasheet/texasinstruments/tps2042a.pdf>, 2000.

[18] Cypress Semiconductor Corporation , *GPIF Designer*, Dostupný z WWW: <http://download.cypress.com.edgesuite.net/design_resources/software_and_drivers/contents/gpif_designer_13.zip>, 2003

A. Tabulka 14: Cílový interface (vývody č. 1)

Pin	Jméno	Alternativní funkce	Směr komunikace	Popis
1	GND		N/A	Zem
3	PA0	nSS2 / nINT0	I/O	Port A, Bit 0
5	PA1	nSS3 / nINT1	I/O	Port A, Bit 1
7	PA2	nSS4 / SLOE	I/O	Port A, Bit 2
9	PA3	nSS5	I/O	Port A, Bit 3
11	PA4	nSS6 / FIFOADR0	I/O	Port A, Bit 4
13	PA5	nSS7 / FIFOADR1	I/O	Port A, Bit 5
15	PA6	nSS8 / PKTEND	I/O	Port A, Bit 6
17	PA7	nSS9 / FLAGD / nSLCS	I/O	Port A, Bit 7
19	GND		N/A	Zem
21	PB0	FD0	I/O	Port B, Bit 0
23	PB1	FD1	I/O	Port B, Bit 1
25	PB2	FD2	I/O	Port B, Bit 2
27	PB3	FD3	I/O	Port B, Bit 3
29	PB4	FD4	I/O	Port B, Bit 4
31	PB5	FD5	I/O	Port B, Bit 5
33	PB6	FD6	I/O	Port B, Bit 6
35	PB7	FD7	I/O	Port B, Bit 7
37	GND		N/A	Zem
39	PC0	GPIFADR0	I/O	Port C, Bit 0

Pin	Jméno	Alternativní funkce	Směr komunikace	Popis
2	5V		N/A	Neregulovaných +5V z USB sběrnice (400mA max)
4	RESET_B		OD	FX2 reset, Active low.
6	CLKOUT		Output	48MHz CPU clock
8	IFCLK		Output	48MHz GPIO clock
10	INT4		Input	8051 INT4 IRQ. Active high, edge sensitive
12				
14				
16				
18				
20	5V		N/A	Neregulovaných +5V z USB sběrnice (400mA max)
22	CTL0	CMD_DATA / FLAGA	Output	GPIF control output 0
24	CTL1	REN / FLAGB	Output	GPIF control output 1
26	CTL2	WEN / FLAGC	Output	GPIF control output 2
28	CTL3		Output	GPIF control output 3
30	CTL4		Output	GPIF control output 4
32	CTL5		Output	GPIF control output 5
34				
36				
38	5V		N/A	Neregulovaných +5V z USB sběrnice (400mA max)
40	RDY0	EF / ACK / SLRD	Input	GPIF input signal 0

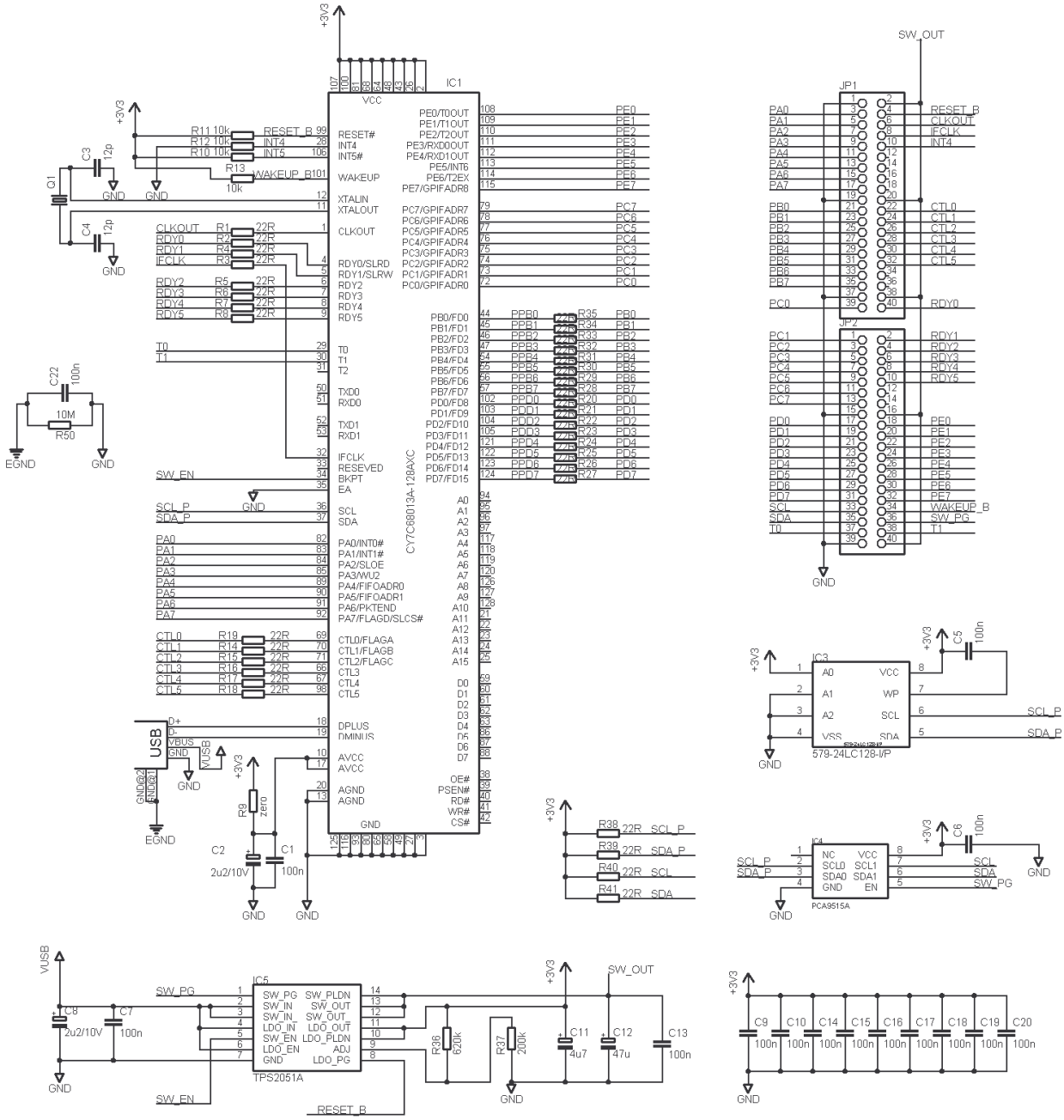
A. Tabulka 15: Cílový interface (vývody č. 2)

Pin	Jméno	Alternativní funkce	Směr komunikace	Popis
1	PC1	GPIFADR1	I/O	Port C, Bit 1
3	PC2	GPIFADR2	I/O	Port C, Bit 2
5	PC3	GPIFADR3	I/O	Port C, Bit 3
7	PC4	GPIFADR4	I/O	Port C, Bit 4
9	PC5	GPIFADR5	I/O	Port C, Bit 5
11	PC6	GPIFADR6	I/O	Port C, Bit 6
13	PC7	GPIFADR7	I/O	Port C, Bit 7
15	GND			Zem
17	PD0	FD8	I/O	Port D, Bit 0
19	PD1	FD9	I/O	Port D, Bit 1
21	PD2	FD10	I/O	Port D, Bit 2
23	PD3	FD11	I/O	Port D, Bit 3
25	PD4	FD12	I/O	Port D, Bit 4
27	PD5	FD13	I/O	Port D, Bit 5
29	PD6	FD14	I/O	Port D, Bit 6
31	PD7	FD15	I/O	Port D, Bit 7
33	SCL		OD	Clock for I2C interface
35	SDA		OD	Data for I2C interface
37	T0		Input	Input for Timer0
39	GND		N/A	Zem

Pin	Jméno	Alternativní funkce	Směr komunikace	Popis
2	RDY1	FF / SLWR	Input	GPIF input signal 1
4	RDY2		Input	GPIF input signal 2
6	RDY3		Input	GPIF input signal 3
8	RDY4		Input	GPIF input signal 4
10	RDY5		Input	GPIF input signal 5
12				
14				
16	5V		N/A	Neregulovaných +5V z USB sběrnice (400mA max)
18	PE0	DATA0 / MOSI	I/O	Port E, Bit 0
20	PE1	DCLK / SCK	I/O	Port E, Bit 1
22	PE2	nCE	I/O	Port E, Bit 2
24	PE3	nCONFIG	I/O	Port E, Bit 3
26	PE4	nSTATUS	I/O	Port E, Bit 4
28	PE5	CONF_DONE / MISO	I/O	Port E, Bit 5
30	PE6	nSS0	I/O	Port E, Bit 6
32	PE7	GPIFADR8 / nSS1	I/O	Port E, Bit 7
34	WAKEUP_B		Input	USB Wakeup. Active low. (not used)
36	SW_PG		Output	Power good signal, Active high when +5V is good
38	T1		Input	Input for Timer1
40	5V		N/A	Neregulovaných +5V z USB sběrnice (400mA max)

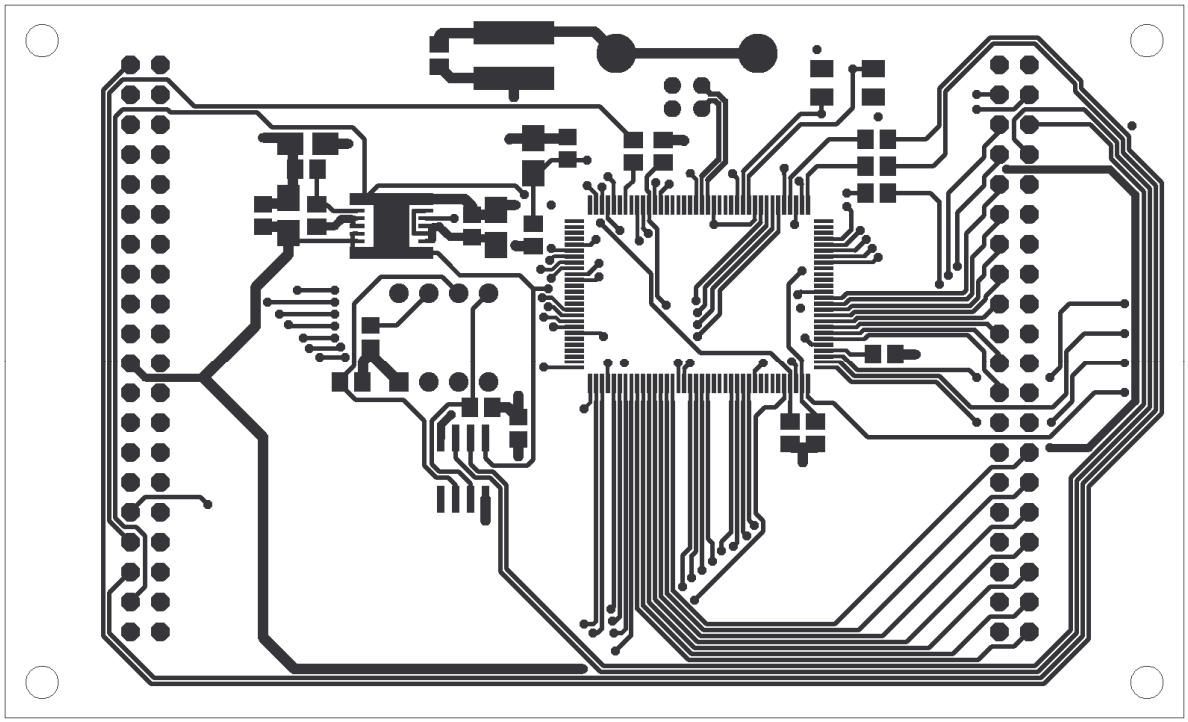
Výkresová dokumentace

B.1 Schéma zapojení desky

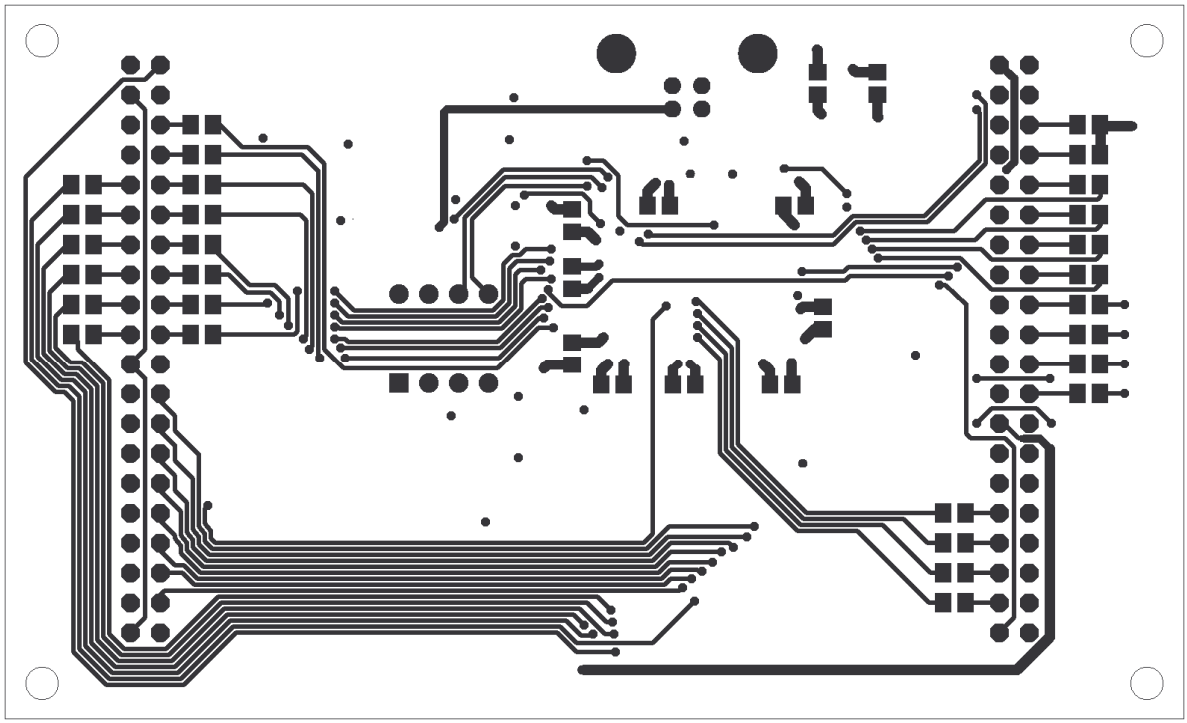


Obrázek B.1: Schéma zapojení desky

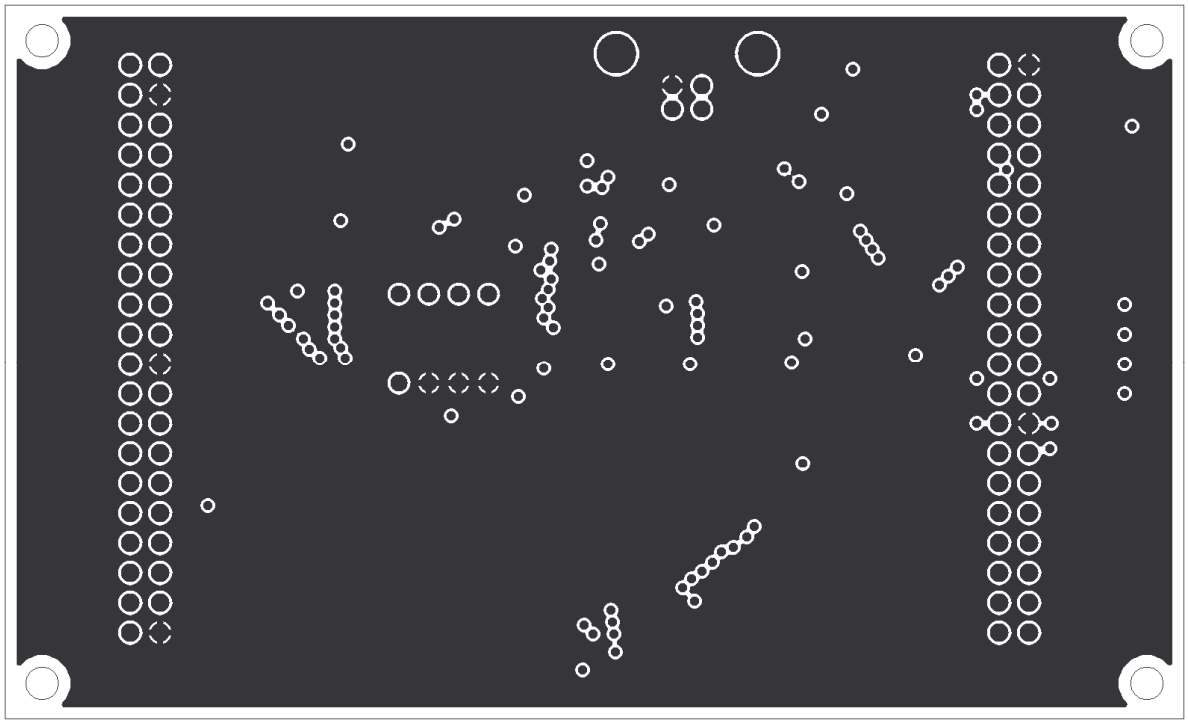
B.2 Předloha DPS



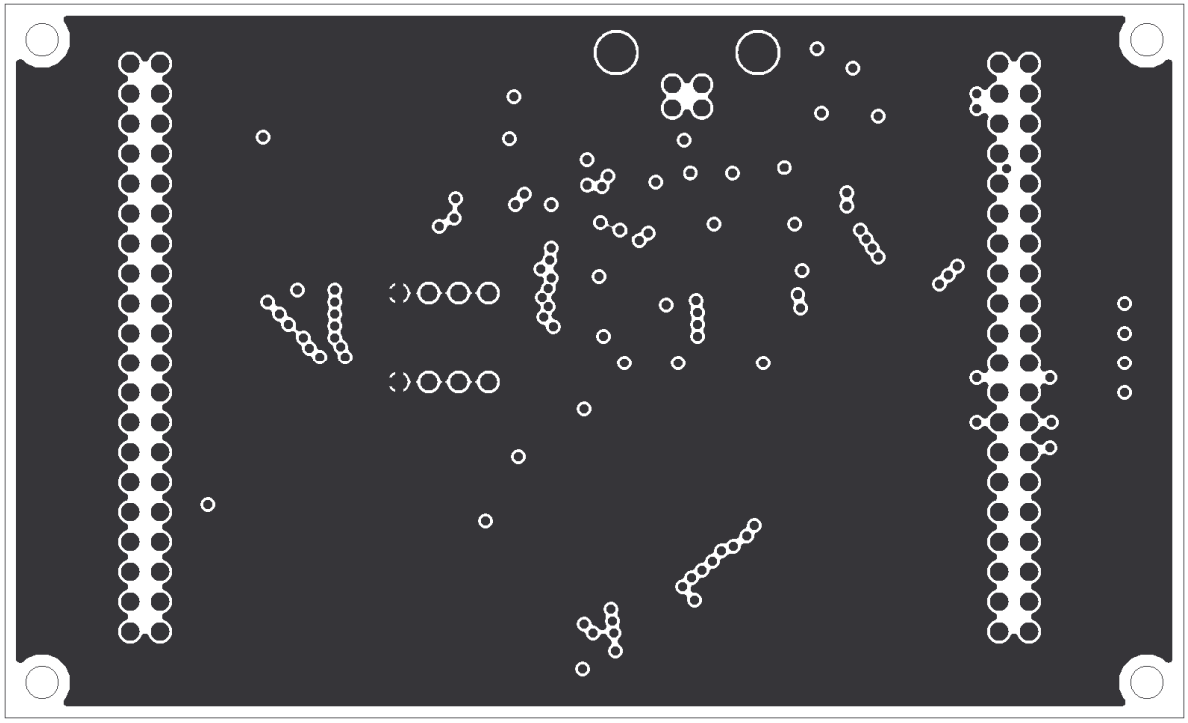
Obrázek B.2: DPS, vrchní strana součástek (TOP)



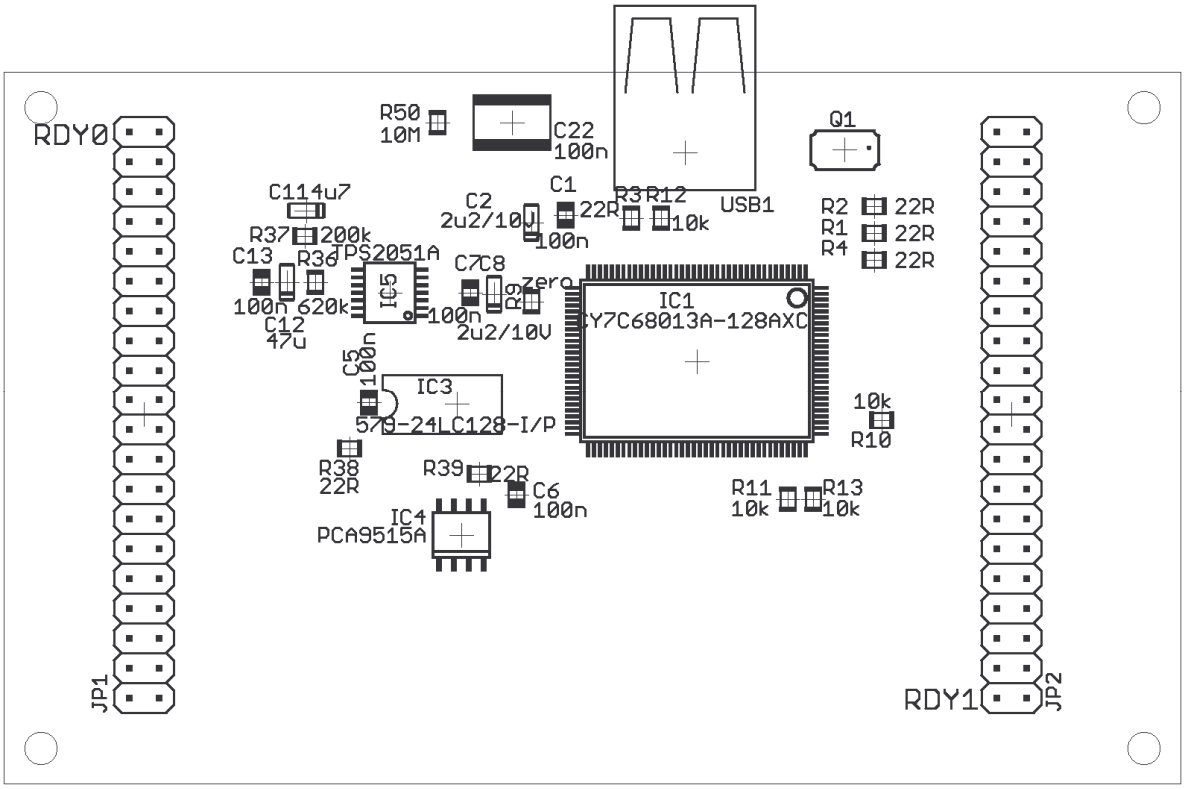
Obrázek B.2: DPS, spodní strana součástek (BOTTOM)



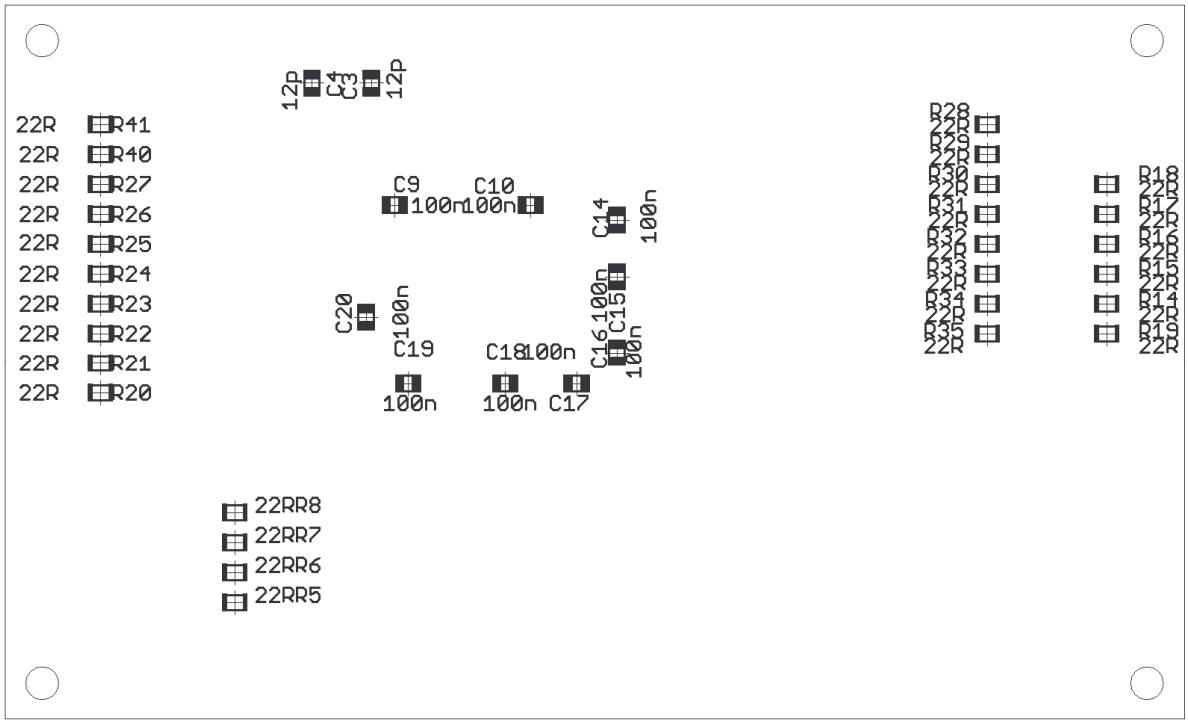
Obrázek B.3: DPS, prostřední vrstva, zem (GND)



Obrázek B.4: DPS, prostřední vrstva, napájení (VCC)



Obrázek B.5: Osazovací plán, vrchní strana (TOP)



Obrázek B.5: Osazovací plán, spodní strana (BOTTOM)

B.3 Seznam součástek

Součástka	Hodnota	Pouzdro
C1	100n	C0805
C2	2u2/10V	A/3216-18W
C3	12p	C0805
C4	12p	C0805
C5	100n	C0805
C6	100n	C0805
C7	100n	C0805
C8	2u2/10V	A/3216-18W
C9	100n	C0805
C10	100n	C0805
C11	4u7	A/3216-18W
C12	47u	A/3216-18W
C13	100n	C0805
C14	100n	C0805
C15	100n	C0805
C16	100n	C0805
C17	100n	C0805
C18	100n	C0805
C19	100n	C0805
C20	100n	C0805
C22	100n	C4564
IC1	CY7C68013A-128AXC	SQFP-R-14X20-128
IC3	579-24LC128-I/P	DIP-8
IC4	PCA9515A	SO08
IC5	TPS2051A	HTSSOP14PWP
JP1		2X20
JP2		2X20
Q1	24MHz	CTS40
R1	22R	M0805
R2	22R	M0805
R3	22R	M0805
R4	22R	M0805
R5	22R	M0805
R6	22R	M0805
R7	22R	M0805
R8	22R	M0805
R9	zero	M0805
R10	10k	M0805
R11	10k	M0805
R12	10k	M0805

R13	10k	M0805
R14	22R	M0805
R15	22R	M0805
R16	22R	M0805
R17	22R	M0805
R18	22R	M0805
R19	22R	M0805
R20	22R	M0805
R21	22R	M0805
R22	22R	M0805
R23	22R	M0805
R24	22R	M0805
R25	22R	M0805
R26	22R	M0805
R27	22R	M0805
R28	22R	M0805
R29	22R	M0805
R30	22R	M0805
R31	22R	M0805
R32	22R	M0805
R33	22R	M0805
R34	22R	M0805
R35	22R	M0805
R36	620k	M0805
R37	200k	M0805
R38	22R	M0805
R39	22R	M0805
R40	22R	M0805
R41	22R	M0805
R50	10M	M0805
USB1	USB-B-H	USB-B-H

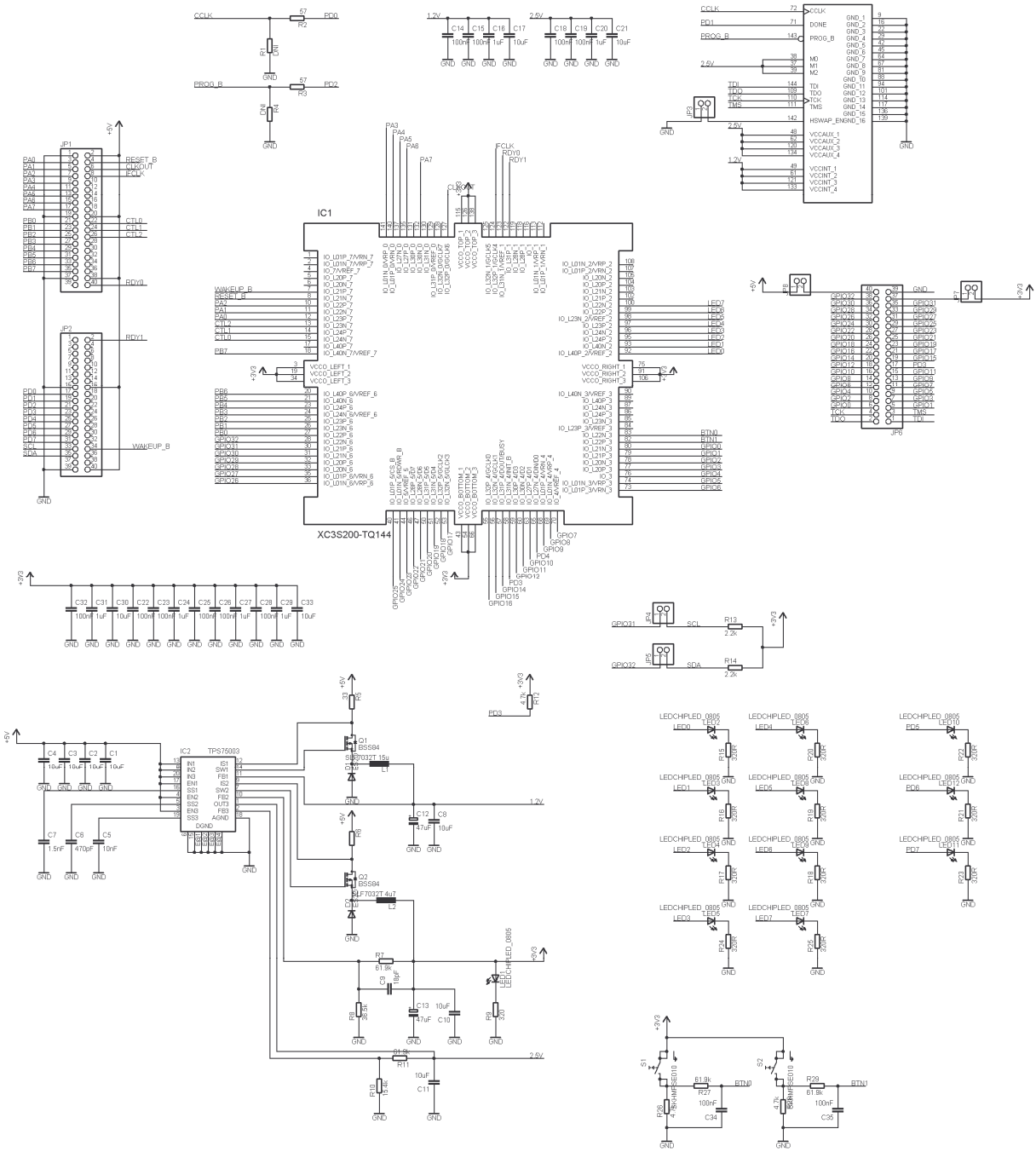
C Seznam příloh

Příloha 1. CD/DVD
Příloha 2. Schéma+DPS obvodu s FPGA

D POZNÁMKY

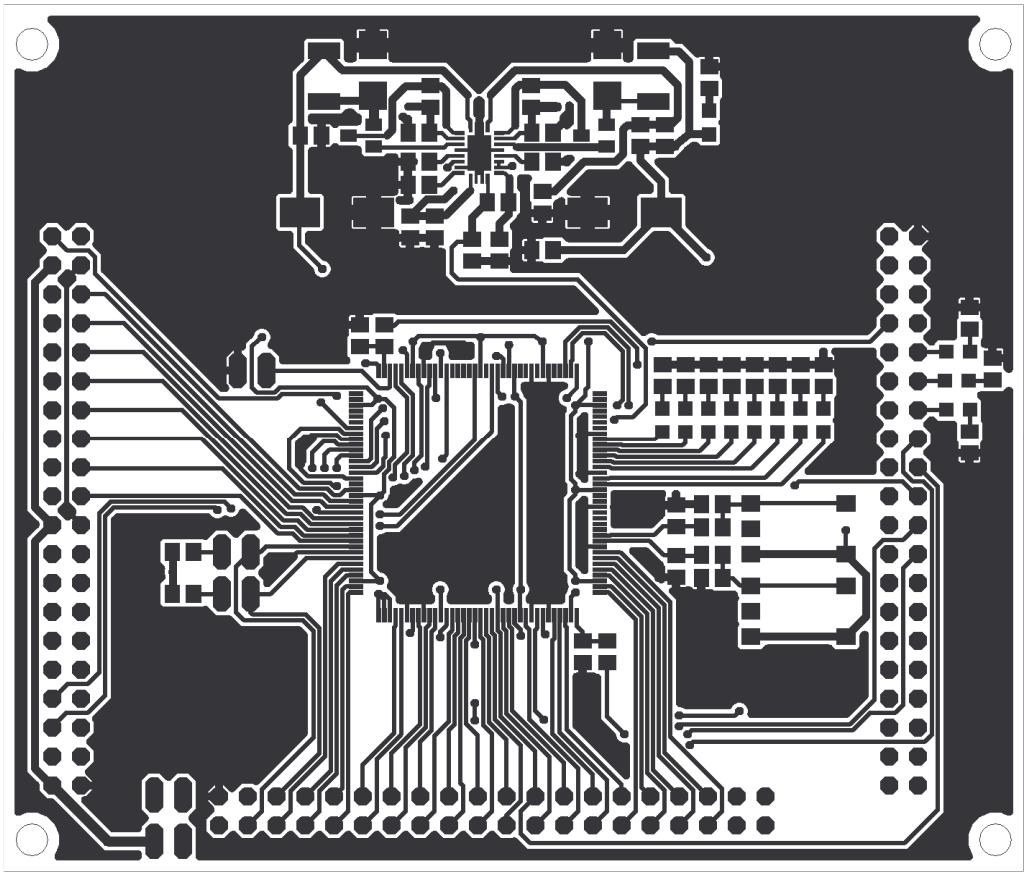
Schémata a desky plošných spojů jsou kresleny v programu EAGLE v. 4.16r2 (CadSoft).

1. Schéma zapojení desky

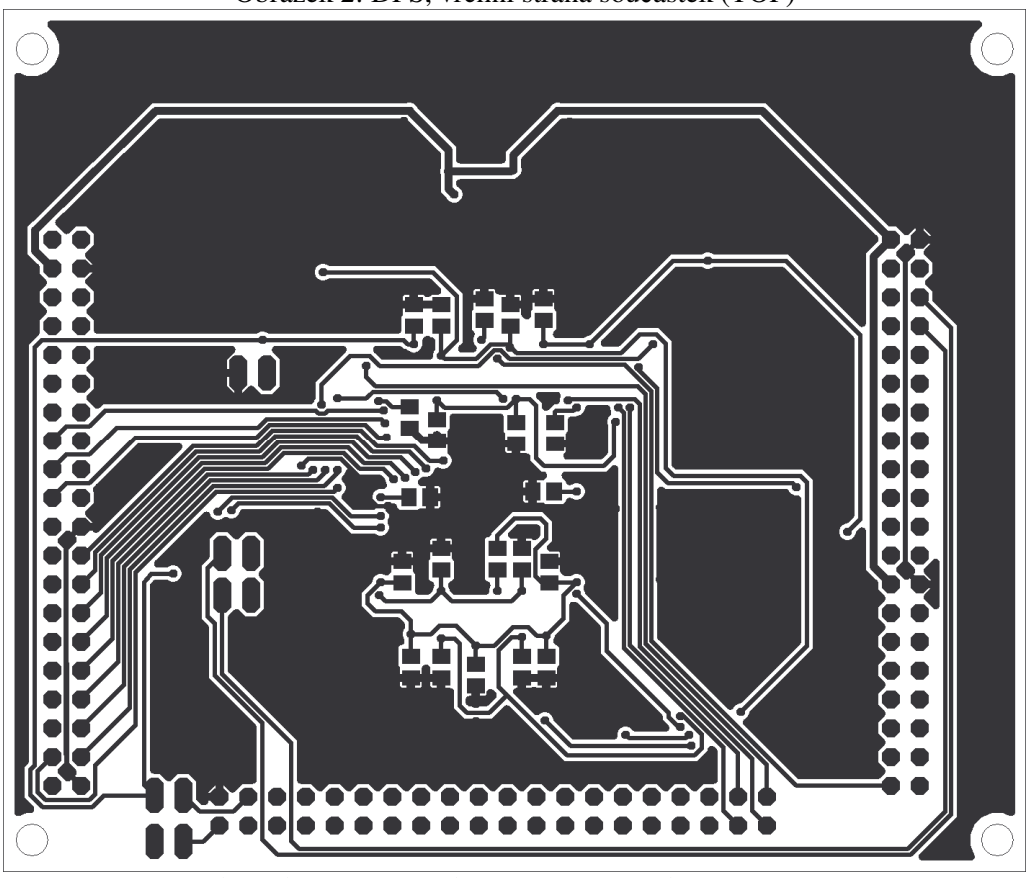


Obrázek 1: Schéma zapojení desky

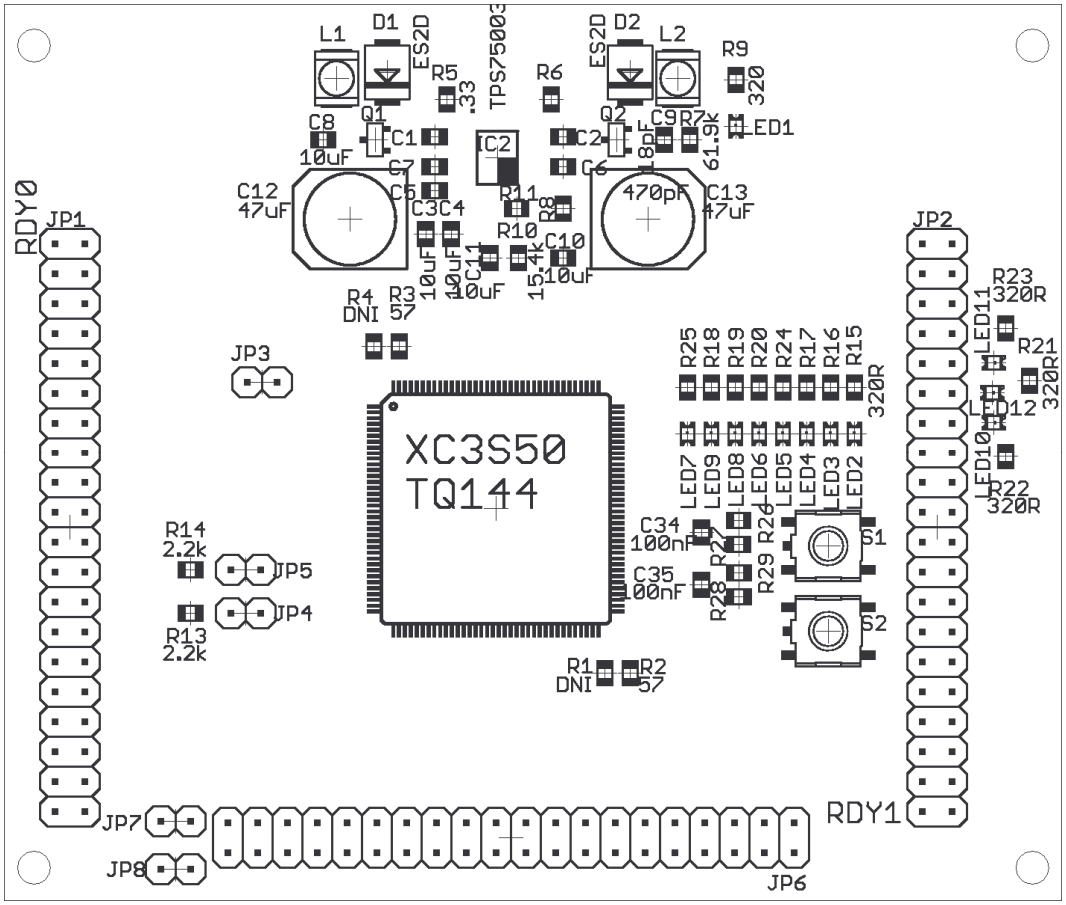
2. Předloha DPS



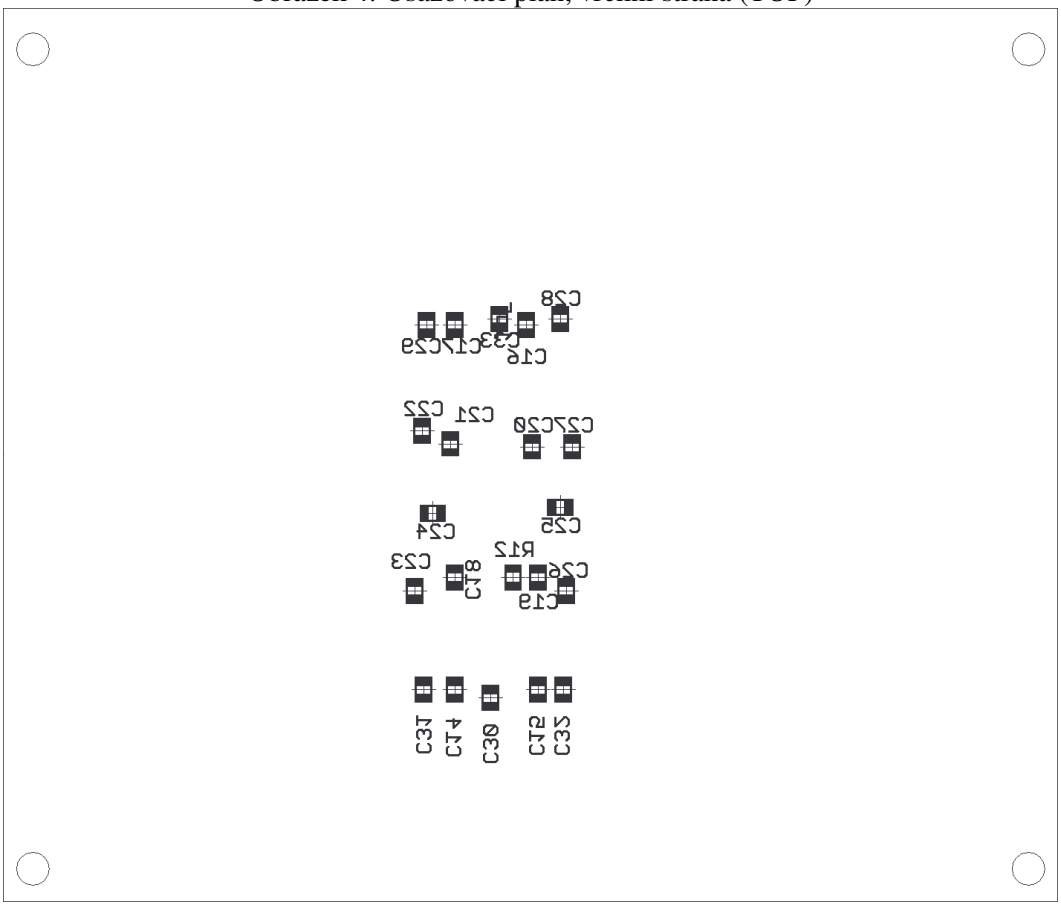
Obrázek 2: DPS, vrchní strana součástek (TOP)



Obrázek 3: DPS, spodní strana součástek (BOTTOM)



Obrázek 4: Osazovací plán, vrchní strana (TOP)


Obrázek 5: Osazovací plán, spodní strana (BOTTOM)