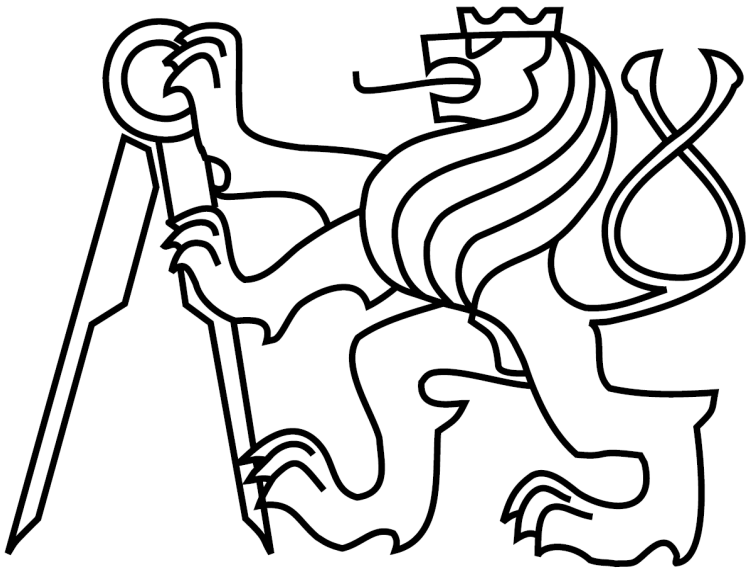
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE

2008 **Ladislav Vávra**

České vysoké učení technické v Praze
Fakulta elektrotechnická



Diplomová práce

Vysoce spolehlivý systém založený na FPGA obvodech

Bc. Ladislav Vávra

Vedoucí práce: Ing. Pavel Kubalík., Ph.D.

Studijní program: Elektrotechnika a informatika, strukturovaný, magisterský

Obor: Výpočetní technika
Květen 2008

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb. , o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne ……. …….

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce Ing. Pavlovi Kubalíkovi., Ph.D. za jeho pomoc a trpělivé vedení během tvorby této práce.

Dále bych rád poděkoval všem blízkým, kteří mě po celou dobu studia na FEL ČVUT podporovali.

Anotace

Tato diplomová práce se zabývá návrhem a realizací vysoce spolehlivého systému založeného na obvodech FPGA. Systém je složen z identických zařízení. Každé zařízení obsahuje obvod FPGA a MCU. Systém je složený z libovolného počtu zařízení. Část analýza se zabývá návrhem propojení zařízení a vhodným výběrem komponentů. Takto navržený systém je odolný proti poruchám. Zařízení bylo úspěšně testováno na jednoduché demonstrační aplikaci, která byla napsána v rámci této diplomové práci.

Abstrakt

This thesis deals with proposal and realization of highly reliable system based on FPGA. The system is built from any count of identical devices. Each device cosists of one FPGA and one MCU. In the analysis part is described proposal of connection of the devices and seletion of components. System designed like that is fault resistant. The system was succesfully tested by the simple demo application.

1. Úvod

Cílem této diplomové práce je navrhnout a zrealizovat vysoce spolehlivý systém založený na obvodech FPGA. Systém se bude skládat z univerzálních modulů (zařízení), které budou v systému plnit tutéž funkci. Takto navržený systém bude odolný proti poruchám. Výkonnou funkci zařízení bude plnit právě obvod FPGA respektive design implementovaný v obvodu FPGA. Výstupní signály budou přeposílány k porovnání s výstupními signály jiného zařízení. Cílem této práce je tedy i navrhnout způsob propojení více zařízení tak, aby bylo univerzální. Bude tedy možné složit různé systémy ze dvou, tří, čtyř atd. zařízení. Takto složené systémy pak budou moci pracovat v různých módech např. za správné hodnoty výstupních signálů se budou považovat hodnoty na kterých se shodnou alespoň tři zařízení ze tří (zkráceně 3ze3) nebo dvě zařízení ze tří (2ze3) atd. Budou-li se výstupní signály lišit, provede se rekonfigurace obvodů FPGA, kvůli zvýšení bezpečnosti proti SEU (Single Event Upset). Implementovaný design bude TSC (Totalty Self-Checking), česky samotestovatelný. Takto napsaný design může odhalit některé vzniklé chyby sám. Bezpečnost systému tedy nebude zvýšena pouze zdvojením nebo ztrojením zařízení, ale také implementovaným designem a případným znovunahráním designu do obvodu FPGA. Pro takto navržený systém bude napsána jednoduchá demonstrační aplikace.

2. Vymezení řešeného problému, cílů a požadavků

- Navrhnout jednoduché zařízení s obvodem FPGA do kterého bude možno implementovat zabezpečený design

- Navrhnout propojení zařízení tak, aby bylo možné sestavit systém o libovolném počtu zařízení, které se budou navzájem kontrolovat, takto navržený systém bude odolný proti poruchám

- Jako zdroj konfiguračních dat pro obvod FPGA a jako úložiště diagnostických dat bude sloužit přenosné paměťové médium

- Obvod FPGA bude konfigurován mikrořadičem ihned po startu a poté kdykoliv to bude nutné

- Bude napsána jednoduchá demonstrační aplikace

- Bude vytvořena knihovna nejdůležitějších funkcí, z kterých bude možnost sestavit obslužný algoritmus obvodu FPGA

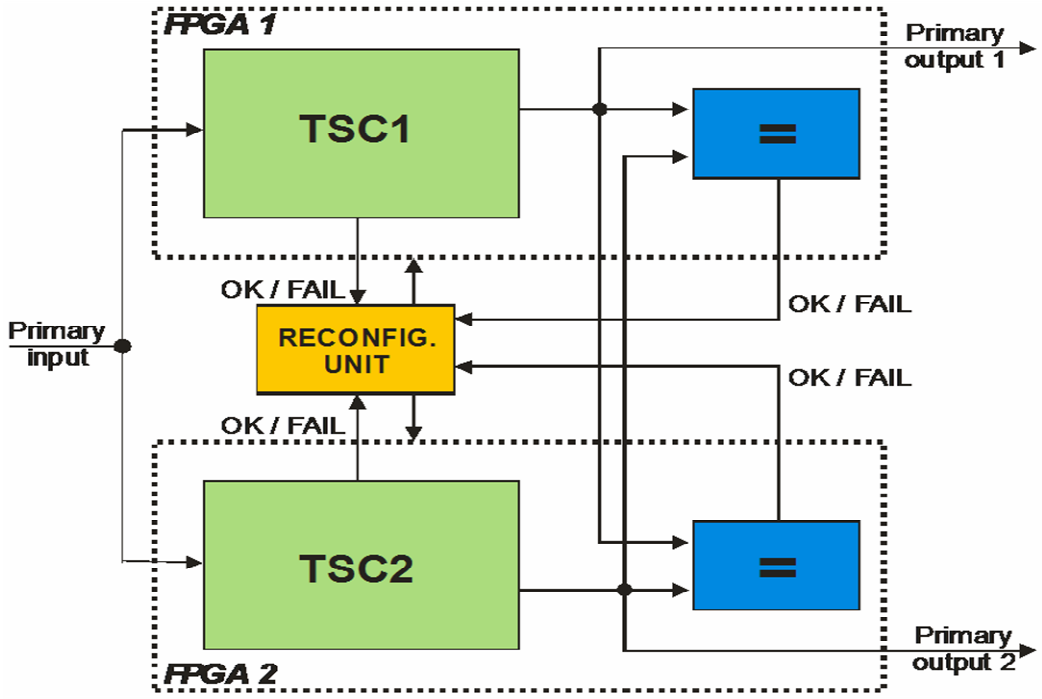
- Návrh synchronizace přeposílání výstupů jednotlivých FPGA na druhé není cílem této diplomové práce, protože je určitým způsobem závislá na implementovaném designu v obvodu FPGA

3. Návrh systému

3.1. Návrh zařízení

Návrh zařízení vychází ze systému uvedeném v disertační práci[1] pana Ing. Pavla Kubalíka., Ph.D. Tento systém je uveden na obrázku Obr. 1. Je vidět, že se jedná o systém s obvodem FPGA, který pro zvýšení bezpečnosti používá zdvojení obvodu FPGA. V systému jsou tedy dva obvody FPGA. Výstupy z jednoho FPGA jsou přivedeny do druhého a jsou v něm porovnávány komparátorem a naopak. Z komparátoru jsou vedeny do rekonfigurační jednotky signály OK a FAIL, které jsou nastaveny v závislosti na výsledku porovnání obou výstupů. Do rekonfigurační jednotky jsou ještě přivedeny signály OK a FAIL z TSC (Totally Self-Checking Circuit). Jde o samotestovatelný obvod nahraný v FPGA. Rekonfigurační jednotka má za úkol přehrát bitstream některého z obvodů FPGA, v případě že dojde k neshodě výstupů z FPGA nebo když TSC hlásí zjištěnou chybu. Účelem takto navrženého systému je ochránit obvod implementovaný v obvodu FPGA proti SEU (Single Event Upset). Toho je docíleno:

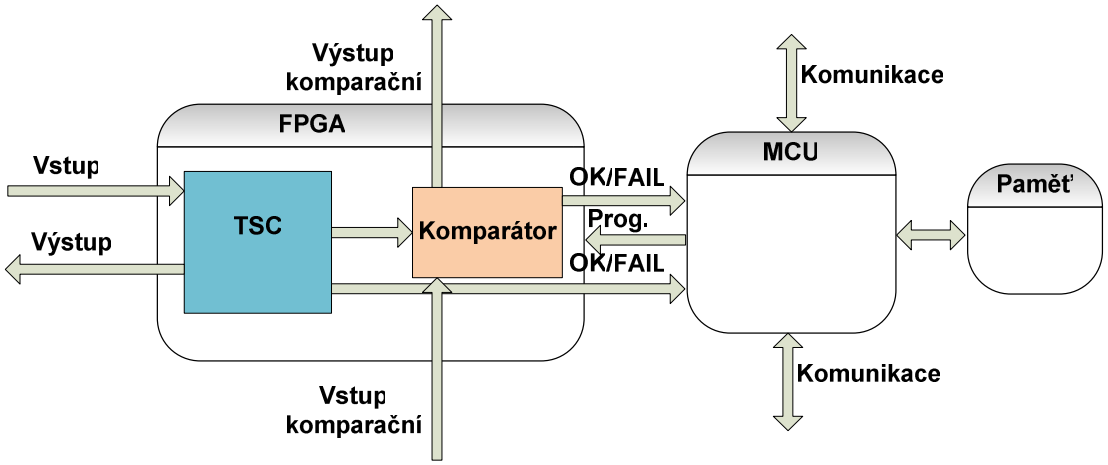
1. Použitím rekonfigurace přímo v systému
2. Zdvojením výkonného jádra (obvod FPGA)
3. Implementací TSC v obvodu FPGA



Obr 1. Spolehlivý systém

Mým úkolem je navrhnout zařízení, které bude mít implementováno výše uvedené požadavky, ale navíc musí být univerzální tak, aby bylo možné sestavit libovolně velký systém. Například systém s pěti obvody FPGA, který pak může pracovat v módu 5z5 nebo 4z5 tzn. výstupy se považují za správné, pokud se na nich shodnou alespoň čtyři zařízení.

V mém zařízení bude funkci rekonfigurační jednotky plnit mikrokontroler. Na jednom zařízení bude pouze jeden obvod FPGA, takže když sestavíme systém s pěti obvody FPGA bude v systému i pět rekonfiguračních jednotek. Jako univerzální propojení je vhodné propojení do kružnice. Na obrázku níže je uveden blokový návrh mého zařízení, které bude možno propojovat do libovolně velkého systému.

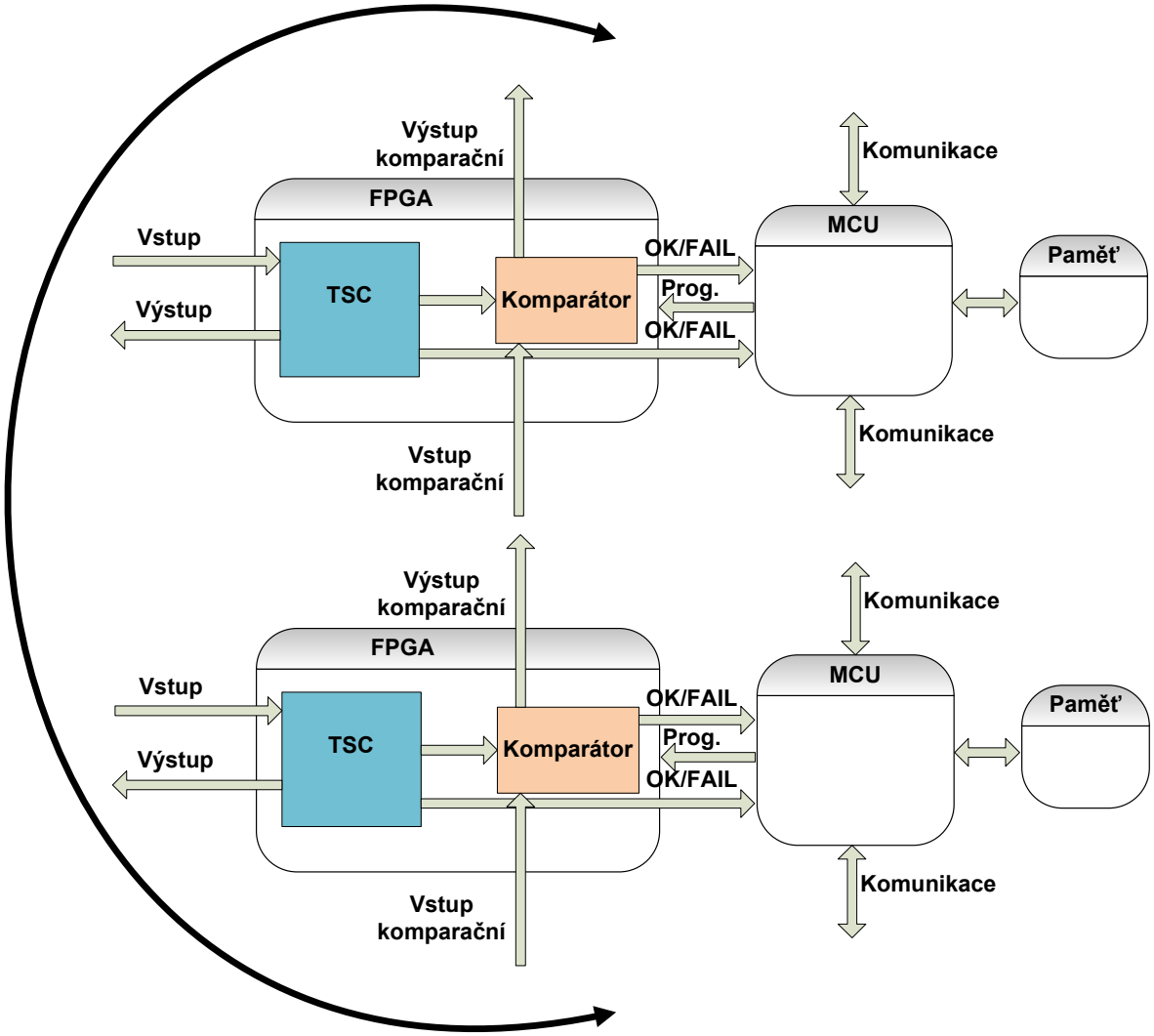


Obr 2. Blokový návrh zařízení

Uvnitř FPGA je nahraný samotestovatelný obvod, ve schématu označen jako (TSC Totally Self-Checking Circuit), který chceme ještě zbezpečnit tím, že přidáme další zařízení. Aby bylo možné výstupy porovnat s druhým zařízením, je v obvodu FPGA ještě umístěn komparátor. Ten porovnává výstupy z vlastního TSC a z TSC druhého zařízení. Zařízení tedy má dvě linky s názvy Výstup komparační a Vstup komparační, sloužící k přenosu výstupů z jednoho zařízení do druhého a naopak. Další dvě linky jsou Vstup a Výstup pro TSC, jimiž zařízení vykonává danou funkci v nějakém systému. Obvod FPGA (Komparátor) také musí komunikovat s mikrokontrolerem a informovat ho o stavu v jakém se nachází. K tomu slouží signály OK/FAIL. Samotestovatelný obvod v FPGA také informuje MCU o svém stavu a rovněž k tomu slouží signály OK/FAIL. Mikrokontroler má dvě komunikační linky pro komunikaci se sousedními zařízeními. K MCU je ještě připojena paměť, která slouží jako zdroj bitstreamu pro FPGA a jako úložiště diagnostických dat.

3.2. Propojení více zařízení

Na obrázku Obr. 3 je zobrazeno propojení dvou zařízení s naznačenou kružnicí, která naznačuje, že tímto způsobem může být propojeno libovolné množství zařízení. Hodnoty výstupů jsou posílány po kružnici do dalšího zařízení, to znamená, že zařízení "pod" posílá své hodnoty zařízení "nad", které je porovná se svými hodnotami atd. Zkráceně, směr toku dat po kružnici je pouze v jednom směru. Komunikační kružnice mezi mikrořadiči je obousměrná. Tok dat probíhá v obou směrech. To znamená, že každé zařízení (MCU) komunikuje se svými sousedy "nad i pod".



Obr 3. Propojení více zařízení

3.3. Analýza propojení více zařízení

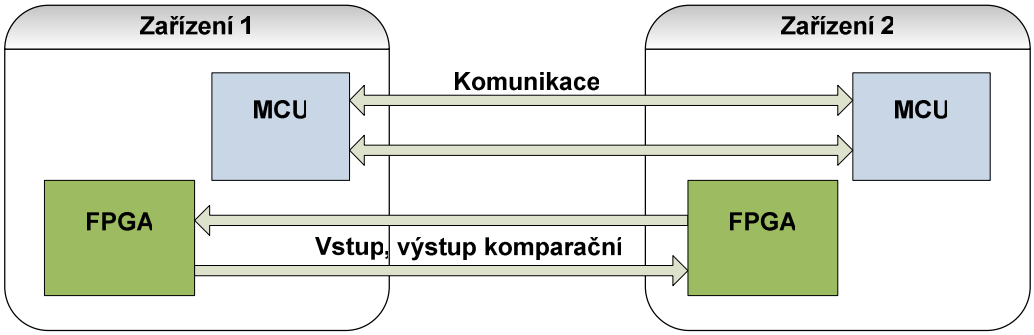
3.3.1. Definování stavů zařízení

V této podkapitole si nadefinuji stavy, ve kterých se může zařízení nacházet.

1) **Stav OK –** samočinně testovatelný obvod (TSC) i komparátor (CP) nehlásí chybu, jsou nastaveny signály OK
2) **Stav TSC FAIL –** samočinně testovatelný obvod hlásí chybu, jsou nastaveny signály FAIL
3) **Stav CP FAIL –** komparátor hlásí chybu, jsou nastaveny signály FAIL
4) **Stav TSC CP FAIL –** samočinně testovatelný obvod i komparátor hlásí chybu, jsou nastaveny od obou signály FAIL
5) **Stav FAIL –** selhání celého propojeného systému, nelze určit které zařízení způsobilo chybu, správnost výstupů není zaručena, tento stav je následkem některého z výše uvedených stavů

3.3.2. Propojení dvou zařízení

Nejzákladnější systém je propojení dvou zařízení. Na obrázku Obr. 4 je uvedeno blokové schéma tohoto propojení. MCU jsou propojeny komunikační linkou. A komparační výstup z FPGA je připojen na komparační vstup FPGA druhého zařízení a naopak. Vznikne tak propojení obvodů FPGA do kružnice. Mikrokontroler ale komunikuje jen se svými sousedy po nezávislých komunikačních linkách. Komunikační sběrnice nebyla použita.



Obr 4. Blokové propojení dvou zařízení

Stav OK

1) Porovnávané signály v komparátoru jsou shodné
2) Komparátor informuje MCU pomocí signálů OK
3) Samočinně testovatelný obvod (TSC) nahraný v FPGA je ve stavu OK a informuje MCU pomocí signálů OK
4) MCU může ukládat diagnostická data, nemá žádný důvod pro rekonfiguraci FPGA, komunikuje s MCU na druhém zařízení

Stav TSC FAIL

1) Samočinně testovatelný obvod hlásí chybu, jsou nastaveny signály FAIL
2) MCU provádí rekonfiguraci FPGA, ukládá diagnostická data a informuje MCU druhého zařízení o tom, že rekonfiguruje

V tomto případě je zřejmé, že rozhodnutí o rekonfiguraci je lokální (tzn. Není potřeba informace z druhého zařízení). Taktéž komparátory obou zařízení nehlásí chybu, výstupy jsou tudíž shodné.

Stav CP FAIL

1) Komparátor hlásí chybu, jsou nastaveny signály FAIL
2) Komparátor druhého zařízení však musí chybu hlásit také a MCU se o tom informují, proto nelze rozhodnout, na kterém zařízení chyba vznikla
3) Obě zařízení provedou rekonfiguraci
4) Pokud problém přetrvává, dostáváme se do stavu **FAIL (selhání celého systému)**, protože nelze určit zařízení správně pracující ani správné výstupy

Stav TSC CP FAIL

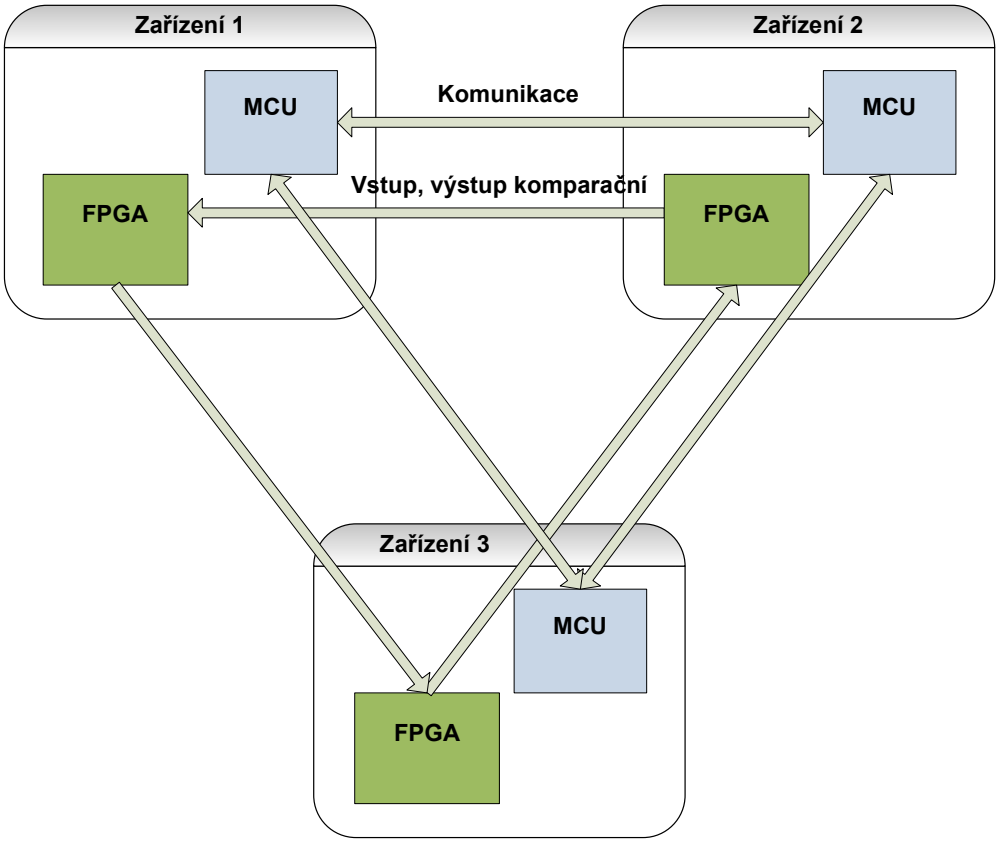
1) Komparátor hlásí chybu, jsou nastaveny signály FAIL
2) Komparátor druhého zařízení však musí chybu hlásit také a MCU se o tom informují, proto nelze prozatím rozhodnout, na kterém zařízení chyba vznikla
3) TSC hlásí chybu, jsou nastaveny signály FAIL
4) TSC nahlášením chyby jednoznačně identifikoval zařízení na kterém vznikla chyba, chybové zařízení je rekonfigurováno, výstupy druhého zařízení jsou dočasně prohlášeny za správné

Stav FAIL

Při takto propojených zařízeních se do tohoto stavu dostáváme jakmile nastane stav CP FAIL a rekonfigurace zařízení nepomohla. Nedá se určit správně pracující zařízení a tudíž ani správné výstupy.

3.3.3. Propojení třech zařízení

Další případ propojení je propojení třech zařízení. Zařízení jsou propojeny do kružnice. Každé MCU je propojeno komunikační linkou jen se svými sousedy. Na obrázku Obr. 5 je celé propojení znázorněno.



Obr 5. Blokové propojení tří zařízení

Stav OK

1) Porovnávané signály v komparátorech jsou shodné, jsou nastaveny signály OK u všech zařízení
2) TSC taktéž nehlásí chybu u žádného zařízení, jsou nastaveny signály OK
3) Každé MCU komunikuje se svými sousedy, a ukládá diagnostické údaje

Stav TSC FAIL

1) TSC hlásí chybu, jsou nastaveny signály FAIL daného zařízení
2) MCU provádí rekonfiguraci a informuje své sousedy o tom, že rekonfiguruje FPGA
3) Správnost výstupů je jednoznačně určena zbývajícími dvěmi zařízeními

Stav CP FAIL

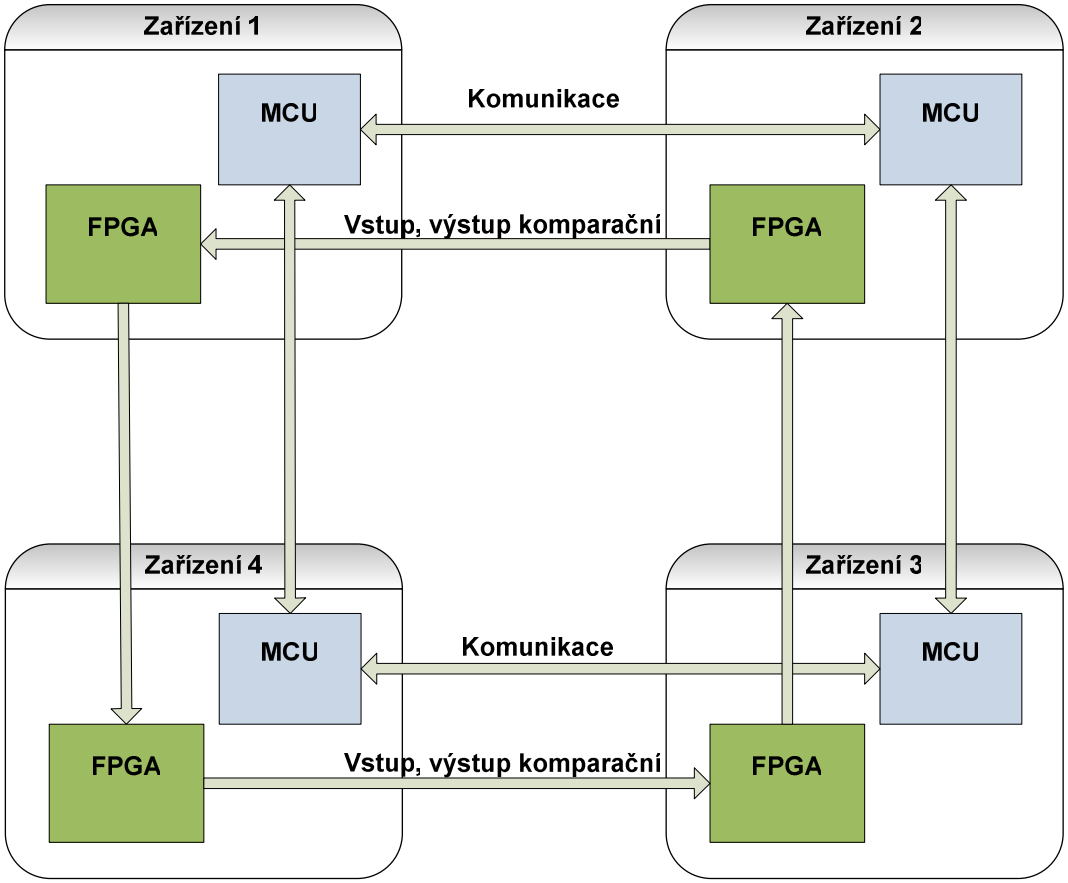
1) Komparátor hlásí chybu, jsou nastaveny signály FAIL
2) Komparátor druhého zařízení hlásí chybu také a MCU se o tom informují, v tomto případě propojení je zde však ještě třetí zařízení, díky kterému je možné určit které z těch zbylých dvou zařízení generuje chybu. V tomto případě je vadné zařízení to, do něhož třetí, správně pracující, zařízení posílá své výstupy.
3) Provede se rekonfigurace zařízení, které generovalo chybu
4) Systém dále poskytuje výstupy generované a ověřené dvěmi zařízeními

Stav TSC CP FAIL

1) Komparátor některého zařízení hlásí chybu, jsou nastaveny signály FAIL, druhé zařízení reaguje a také hlásí chybu a nastavuje signály
2) TSC jednoho z těchto zařízení hlásí chybu a nastavuje signály FAIL
3) TSC nahlášením chyby jednoznačně identifikoval zařízení které generuje chybový výstup, provádí se rekonfigurace, zbylé dvě zařízení poskytují ověřené výstupy

3.3.4. Propojení čtyř zařízení

Poslední propojení, které zde naznačím je propojení čtyř zařízení. Zařízení jsou opět propojeny do kružnice. Každé MCU je propojeno komunikační linkou jen se svými sousedy. Na obrázku Obr. 6 je celé propojení znázorněno.



Obr 6. Blokové propojení čtyř zařízení

Stav OK

1) Porovnávané signály v komparátorech jsou shodné, jsou nastaveny signály OK u všech zařízení
2) TSC taktéž nehlásí chybu u žádného zařízení, jsou nastaveny signály OK
3) Každé MCU komunikuje se svými sousedy, a ukládá diagnostické údaje

Stav TSC FAIL

1) TSC hlásí chybu, jsou nastaveny signály FAIL daného zařízení
2) MCU provádí rekonfiguraci a informuje své sousedy o tom, že rekonfiguruje FPGA
3) Správnost výstupů je jednoznačně určena zbývajícími třemi zařízeními

Stav CP FAIL

1) Komparátor hlásí chybu, jsou nastaveny signály FAIL
2) Komparátor druhého zařízení hlásí chybu také a MCU se o tom informují, v tomto případě propojení jsou zde však ještě dvě další zařízení, díky nimž je možné určit které z těch zbylých dvou zařízení generuje chybu. V tomto případě je vadné zařízení to, do něhož čtvrté, správně pracující, zařízení posílá své výstupy. Obecně lze říci, že špatně pracující zařízení je vždy to, které je druhé proti směru propojení kružnice od prvního správně pracujícího ve směru propojení kružnice od posledního špatně pracujícího.
3) Provede se rekonfigurace zařízení, které generovalo chybu
4) Systém dále poskytuje výstupy generované a ověřené třemi zařízeními

Stav TSC CP FAIL

1) Komparátor některého zařízení hlásí chybu, jsou nastaveny signály FAIL, druhé zařízení reaguje a také hlásí chybu a nastavuje signály
2) TSC jednoho z těchto zařízení hlásí chybu a nastavuje signály FAIL
3) TSC nahlášením chyby jednoznačně identifikoval zařízení které generuje chybový výstup, provádí se rekonfigurace, zbylé tři zařízení poskytují ověřené výstupy

3.3.5. Závěr

Analýzu propojení dalších systémů s pěti a více zařízeními už nemá cenu popisovat, je vidět z analýzy že mezi systémy vzniká jakási analogie. Chování systémů je nezávislé na počtu zařízení, a toho jsem chtěl právě dosáhnout. Jediná analogie je u systému propojení dvou zařízení a to u stavu CP FAIL, kde nelze rozhodnout v určitém případě, které zařízení selhalo.

4. Analýza

4.1. Obvody v návrhu

V tabulce Tab.1. je uveden seznam obvodů použitých v zařízení a kterými se bude zabývat kapitola 4 Analýza. Je v ní uvedena i činnost, kterou v zařízení zastávají. Obvody jsem zvolil na základě jejich parametrů tak, aby splňovali všechny vlastnosti a požadavky zařízení.

Činnost	**Obvod**
Výkonná část, design nahraný v obvodu	FPGA
Obslužná část, rekonfigurace	MCU
Úložiště dat, zdroj bitstreamu	SD paměť
Komunikace s PC přes USB	Převodník rozhraní USB na RS232
Napájení	Spínaný stabilizátor, lineární stabilizátor

Tab.1. Obvody v zařízení a jejich funkce

4.2. Mikrořadič

Na mikrořadič v zařízení jsou kladeny vysoké nároky především na komunikaci. Jde o komunikaci mezi dalšími mikrořadiči jiných zařízení, komunikaci s PC a komunikaci s obvodem FPGA. Dál musí mikrořadič umožňovat komunikaci po rozhraní SPI pro připojení SD paměti. Neméně důležitým kritériem pro výběr mikrořadiče je i jeho pracovní frekvence z důvodu dosažení optimální rychlosti konfigurace obvodu FPGA. Dalším důležitým kritériem je dostatečný počet datových pinů. Odhadovaný počet datových vývodů je uveden v tabulce Tab.2. [2]

Vytížení vývodů	Počet vývodů
Rozhraní USB	5
Rozhraní RS232	4
Připojení k SD paměti	6
Konfigurace FPGA	6
Rozhraní JTAG	4
Signály OK	16
Signály FAIL	16
Celkem	**57**

Tab.2. Odhad datových vývodů mikrořadiče

Z tabulky odhadovaných počtu datových vývodů a z blokového schéma jsem si sestavil základní požadavky, které musí mikrořadič splňovat. Mezi nejdůležitější požadavky, pro jeho plnou funkci v zařízení, patří:

- Pracovní frekvence – min. 16MHz
- Paměť programu – min. 128kB
- Rozhraní JTAG
- Rozhraní SPI
- 3x rozhraní RS232

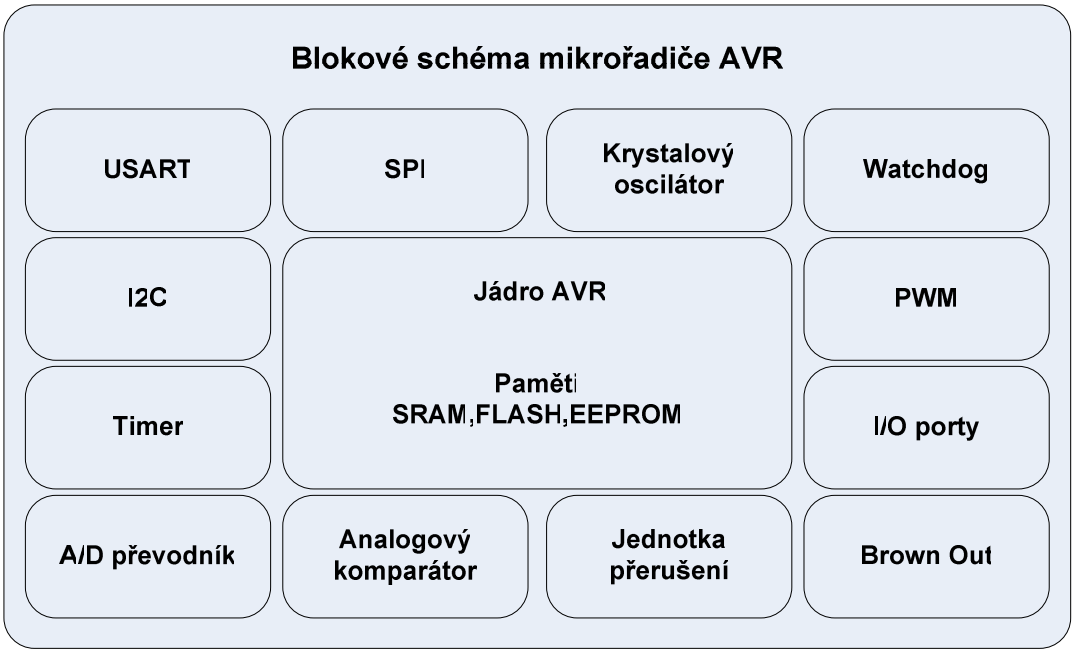
Mikrořadiče jsem vybíral pouze z produktů dvou pro mě nejznámějších výrobců, a to od firem Atmel a Microchip. Po pečlivé úvaze jsem pro realizaci zařízení zvolil mikrořadič ATMEGA 2560 v SMD pouzdře TQFP100. Moje rozhodování také ovlivnila větší praxe s používáním obvodů od firmy Atmel.

ATMEGA 2560 je "low-power" 8-mi bitový mikrořadič vyrobený technologií CMOS. Používá rozšířenou AVR RISC architekturu. V každém taktu dokončí jednu instrukci a dosahuje téměř 1 MIPS za 1 MHz. AVR jádro má k dispozici 32 pracovních registrů. Každý z registrů je přímo propojen s ALU jednotkou. AVR jádro používá za účelem maximalizování výkonu a dosažení co nejvyššího paralelismu harvardskou architekturu tzn. paměť programu a paměť dat jsou odděleny. Další hlavní rysy mikrořadiče ATMEGA 2560 jsou uvedeny v tabulce Tab.3.

I/O porty	11 (86 pinů)
Paměť programu	256kB
Paměť RAM	8kB
Paměť EEPROM	4kB
PWM kanály (16 bitů rozlišení)	12
USART	4
ADC kanály	16
Čítače/časovače	6
JTAG	1
Rozhraní SPI	1

Tab.3. Hlavní rysy mikrořadiče ATMega2560

Blokové schéma mikrořadiče AVR rodiny AT mega, uvedeného na obrázku níže, dává ucelený přehled o funkčních blocích tohoto mikrořadiče.



Obr 7. Uspořádání bloků v mikrořadiči rodiny ATMega

Krátký popis funkčních bloků

- **USART** (Univeral Asynchronous/Synchronous Receiver and Transmitter) Plně duplexní sériové rozhraní. Obsahuje detekci falešného start-bitu, detekci chybného znaku a přetečení datového registru, filtraci šumu.
- **Watchdog** (hlídací pes), je to časovač, který je spuštěn po resetu nebo později z programu a po vypršení časového limitu vyvolá automatické přerušení, pokud ho nějakým podmětem (signálem, hodnotou zapsanou do řídícího registru, speciální instrukcí) nevrátíme do počátečního stavu
- **SPI** (Serial Peripheral Interface) sériové synchronní rozhraní pro přídavná zařízení.

  Přes toto rozhraní se dá obvod programovat metodou ISP (In – System Programming) umožňuje programování MCU přes jednoduché sériové rozhraní
- **Brown Out** podpěťová ochrana
- **I2C** univerzální sériová sběrnice pro komunikaci jednoho obvodu master s obvody slave
- **Timer** čítač
- **Analogový komparátor** může se využít například pro méně přesný A/D převodník
- **A/D převodník** ve většině MCU 8x 10 bitový převodník s možností omezení šumu.
- **Přerušení** (Interrupts) Adresy vektorů přerušení (pomocí kterých se přerušení obsluhuje) jsou uspořádány za sebou od začátku paměťového prostoru. Jejich počet a funkce závisí na dostupných periféríích daného MCU.

4.3. Obvod FPGA

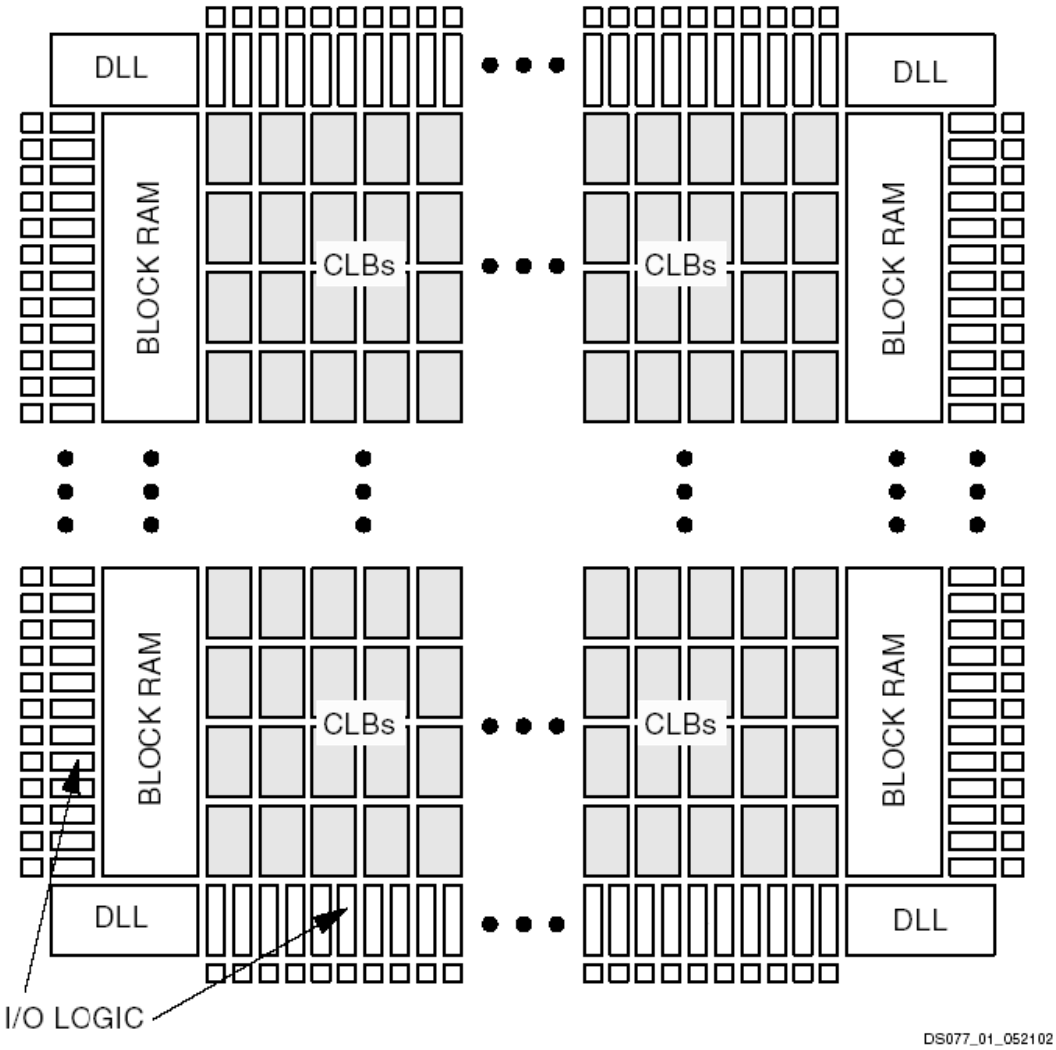
Výkonným srdcem zařízení je obvod FPGA. V tomto obvodu lze sestavit obrovské množství různých systémů a funkčních bloků. Zařízení se tak stává velmi univerzální a může plnit mnoho funkcí v různých odvětvích. Zvolil jsem obvod patřící do rodiny Spatan II od firmy Xilinx. Jde o typ XC2S200E v pouzdře PQ208. Důvody, které mě k tomu vedli jsou dostupnost obvodu (školní zásoby). Dalším hlavním důvodem použití je to, že obvod je dostupný v pouzdře PQ208, takže ho lze osadit na desku plošného spoje vlastními silami. Obvod potřebuje dvě napájení. A to napájení I/O bank napětím 3,3V a napájení výkonného jádra u obvodu XC2S200E napětím 1,8V. V seznamu níže jsou uvedeny hlavní rysy architektury Spartan IIE.

- až 600000 systémových hradel nebo 15552 logických buněk
- až 514 I/O pinů
- podpora 19 úrovňových standardů
- až 288Kb blokové RAM
- až 221184b distribuované RAM
- interní napájení 1,8V
- nižší příkon
- vstupy tolerantní na 3V signály
- vstupy tolerantní na 5V signály s externím rezistorem
- větší hustota a větší počet I/O oproti **SPARTAN-II**
- větší „výkon“ oproti **SPARTAN- II**
- neomezená přeprogramovatelnost
- hodinový signál až 200MHz
- integrovaná struktura pro konstrukci rychlých sčítaček
- 4 Delay-Locked Loop (DLL) umožňující frekvenční syntézu
- JTAG port pro testování
- plná podpora vývojového prostředí Xilinx ISE

Můj vybraný obvod obsahuje 5292 logických buněk a na 200 000 hradel a je schopen také pracovat až na frekvenci 200MHz. Architektura obvodu je složena ze čtyř základních bloků:

- **IOBs** (Input/Output Blocks) řídí tok dat mezi I/O pinem a vnitřní logikou
- **CLBs** (Configurable Logic Blocks) obsahující LUTs (Look-Up Tables) na principu paměti RAM
- **Block RAM** umožňující ukládaní dat ve formátu 4096b dual-port blocks.
- **DLL** (Delay-Locked Loop) poskytuje autokalibraci, plně digitální řešení distribuce zpoždění, násobení, dělení a fázový posun hodinového signálu

SPARTAN-IIE i SPARTAN-II využívá pro konfiguraci paměťových buněk a při vypnutí napájení je daná konfigurace ztracena. Proto je potřeba zajistit nahrání konfigurace při každém zapnutí FPGA. Konfigurační data jsou automaticky čtena z externího zdroje dat (PROM, JTAG, FPGA) a to buď sériově nebo paralelně. Na obrázku Obr 8. je zobrazeno uspořádání čtyř základních bloků v architektuře Spartan IIE.



Obr 8. Uspořádání základních bloků v FPGA SPARTAN II

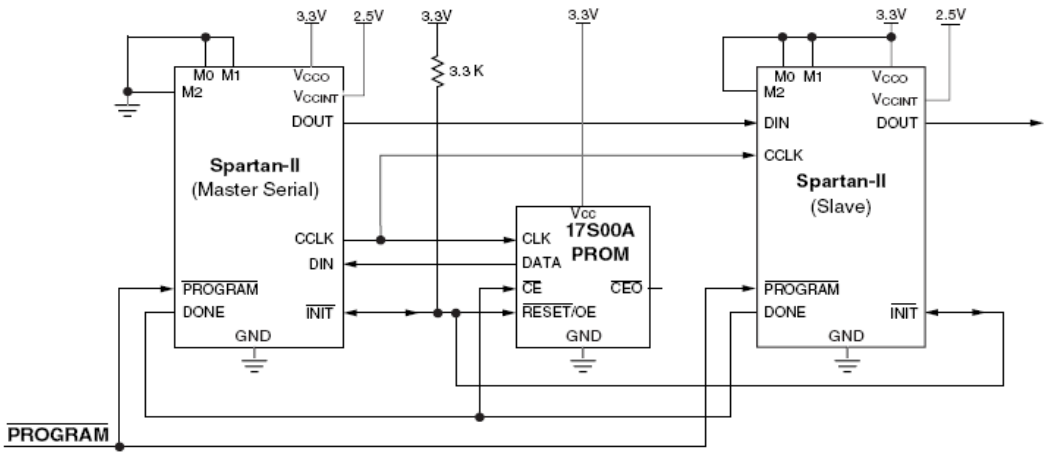
4.4. Konfigurace FPGA

Obvod od firmy Xilinx typu XC2S200E lze konfigurovat více způsoby. Je možné použít sériový mód, paralelní mód nebo přes rozhraní JTAG. Některé módy se ještě dále rozdělují na MASTER nebo SLAVE mód. Tyto módy se nastavují hardwarově pomocí konfiguračních pinů M0, M1 a M2. Nastavení těchto pinů je uvedeno v tabulce Tab. 4.

Configuration Mode	Preconfiguration Pull-ups	M0	M1	M2	CCLK Direction	Data Width	Serial D_{OUT}
Master Serial mode	No	0	0	0	Out	1	Yes
	Yes	0	0	1			
Slave Parallel mode	Yes	0	1	0	In	8	No
	No	0	1	1			
Boundary-Scan mode	Yes	1	0	0	N/A	1	No
	No	1	0	1			
Slave Serial mode	Yes	1	1	0	In	1	Yes
	No	1	1	1			

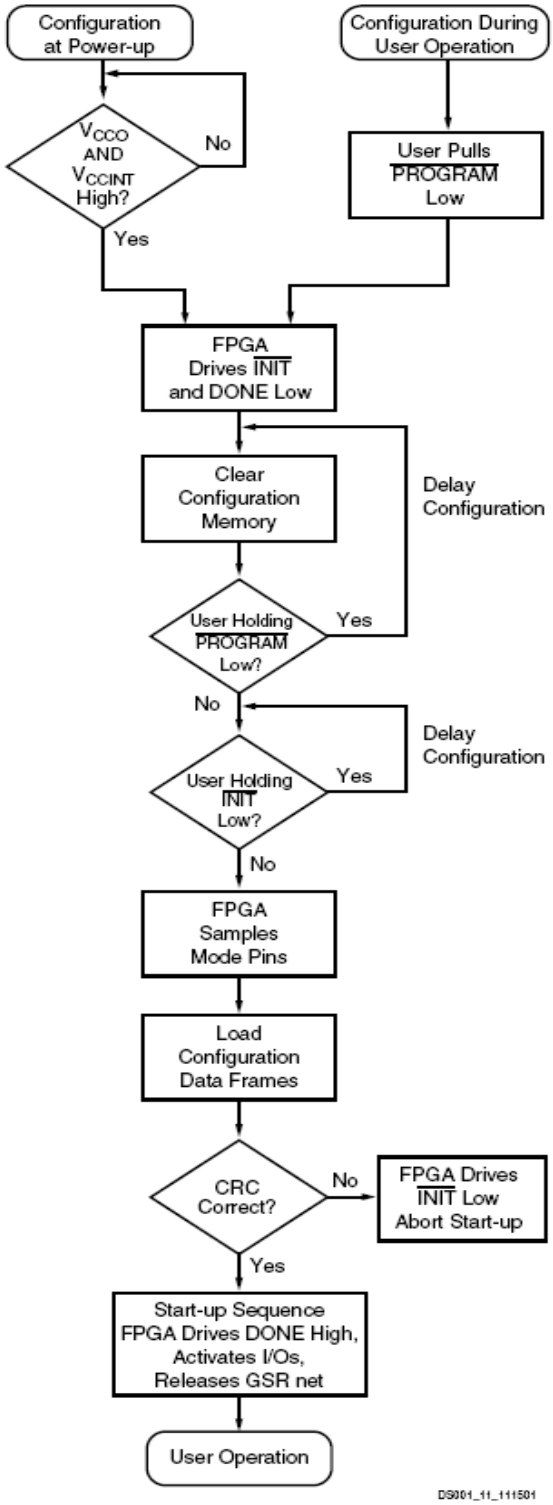
Tab. 4. Konfigurační módy rodiny Spartan II

Já použiji v tomto zařízení sériový mód. Nicméně jsem na desce plošného spoje vyvedl na konektor i rozhraní JTAG a konfigurační piny M0,M1 a M2 jsou nastavitelné pomocí jumperů. Existují dva sériové režimy – master serial mod, FPGA řídí konfigurační proces řízením signálu CCLK (cclk je jako výstup). Slave serial mod, FPGA přijímá hodinový signál CCLK zvenčí (cclk je jako vstup). V obou módech je za jeden hodinový takt nahrán jeden bit bitstreamu. Jako první se zapisuje nejvyšší bit bytu. Další piny obvodu použité v tomto módu jsou #PROGRAM, #INIT, DONE, CCLK, DIN, DOUT. Všechny tyto signály jsou řízeny napětím 3,3V. Na obrázku Obr.9. je zobrazeno schéma přebrané z literatury [5] , zapojení pro oba sériové módy. Zdrojem bitstreamu je paměť PROM.



Obr 9. Zapojení sériového módu

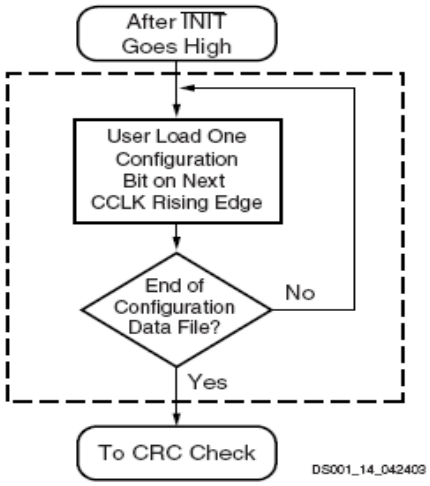
Na obrázku Obr 9. je vidět, že funkci MASTER zastává levý obvod FPGA, který generuje hodinový signál CCLK. Z levého FPGA je připojen signál DOUT na pravé FPGA na signál DIN. Pravé FPGA pracuje v režimu SLAVE. V mém případě zastává funkci MASTER obvod MCU a zdrojem bitstreamu je SD paměť. Algoritmus pro konfiguraci FPGA je pomocí vývojového diagramu znázorněn na obrázku Obr 10.



Obr. 10 Algoritmus konfigurace FPGA v sériovém módu

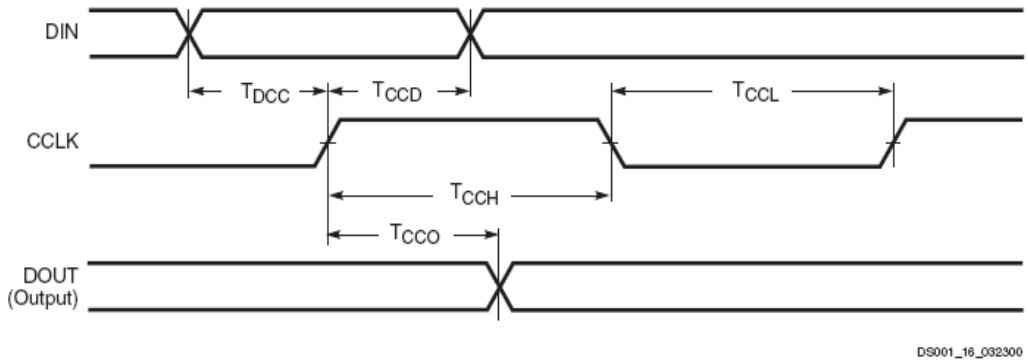
Do konfiguračního módu lze vstoupit dvěmi možnostmi, buď ihned při zapnutí zařízení nebo na vyžádání od uživatele tím, že se nastaví signál PROGRAM na hodnotu low. FPGA potvrdí vstup do konfiguračního módu nastavením signálu INIT na hodnotu low. Poté se signál PROGRAM uvolní z hodnoty low. Uživatel může

v této části konfiguraci pozdržet tím, že bude držet signál PROGRAM nebo INIT na hodnotě low. Poté FPGA vstoupí do fáze mazání konfigurační paměti. Paměť je kompletně smazána, když FPGA nastaví signál INIT na hodnotu high. Následuje stav, kdy FPGA vzorkuje piny M0,M1 a M2 a zjistí tak, v jakém módu bude FPGA konfigurováno. Já používám sériový mód, cyklus probíhající v této části konfigurace je zobrazen na obrázku níže Obr 11.



Obr. 11 Cyklus nahrávání dat do FPGA

Nastaví se hodnota signálu DIN a klikne se hodinovým signálem CCLK, na náběžnou hranu tohoto signálu se hodnota zapíše do konfigurační paměti. Tento cyklus se opakuje, dokud není konec bitstreamu. Časování těchto signálů je uvedeno na obrázku Obr 12. a v tabulce Tab. 5.



Obr. 12 Průběh časování signálu CCLK a DIN

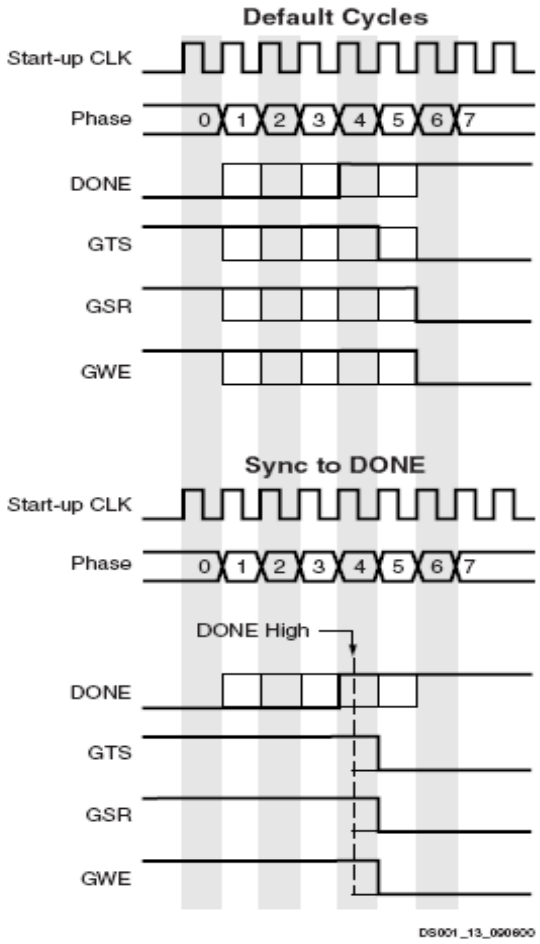
Symbol		Description		Units
T_{DCC}	CCLK	DIN setup	5	ns, min
T_{CCD}		DIN hold	0	ns, min
T_{CCO}		DOUT	12	ns, max
T_{CCH}		High time	5	ns, min
T_{CCL}		Low time	5	ns, min
F_{CC}		Maximum frequency	66	MHz, max

Tab. 5 Tabulka časování k průběhu uvedenému na obrázku

Po nahrání konfigurace do FPGA se porovná hodnota CRC uložená v bitstreamu a hodnota CRC vypočtená v FPGA. Jestliže se hodnoty nerovnají, FPGA nastaví signál INIT na hodnotu low, to indikuje špatný bitstream nebo chybu při nahrávání konfiguračního bitstreamu, konfigurace je ukončena. Znovu je konfigurace spuštěna nastavením signálu PROGRAM na hodnotu low. Jestliže je CRC kontrola v pořádku, vstupuje se do spouštěcí fáze. Spouštěcí sekvence dohlíží na přechod FPGA z konfiguračního módu do plně uživatelského módu. Shoda CRC indikuje úspěšnou konfiguraci signálem INIT a zároveň inicializuje spouštěcí sekvenci. Během této sekvence FPGA vykoná čtyři operace.

1. Potvrzení signálem DONE. Je-li DONE na hodnotě high, konfigurace selhala
2. Uvolnění globální třístavové sítě (GTS)
3. Zrušení globálního resetu (GSR). To umožní všem klopným obvodům typu D změnu stavu
4. Nastavení globálního povolení zápisu (GWE). To dovolí všem obvodům D a paměti RAM změnu stavu

Tyto operace jsou synchronizovány signálem CCLK. Celá spouštěcí sekvence trvá osm taktů a tyto takty se nazývají C0 až C7, načež celý design je plně funkční. Průběh signálů spouštěcí sekvence je zobrazena na obrázku Obr 13.



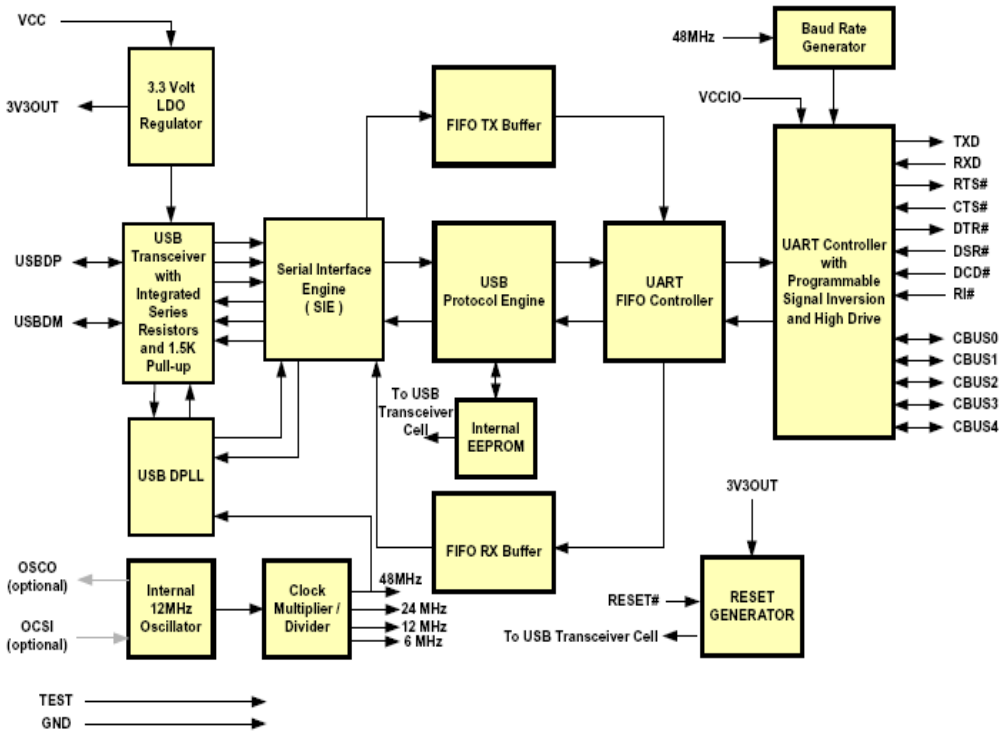
Obr. 13 Spouštěcí sekvence taktů C0 až C7

4.5. Komunikace přes USB

Použitý mikrořadič nedisponuje rozhraním USB, proto je třeba použít již hotový externí modul, kterých je v dnešní době na trhu velké množství. Vybral jsem obvod od firmy Future Technology Device Inc. (FTDI). Jde o modul FT232RL. Je to převodník rozhraní USB na rozhrání RS232. K tomuto obvodu jsou dodávané ovladače pro OS, které zařídí, že se rozhraní USB tváří na PC jako klasické sériové rozhraní RS232. K tomuto obvodu je dostupné i velké množství literatury. V tomto zařízení bude použit i pro konfiguraci mikrořadiče. Konfigurace mikrořadiče a komunikace s PC bude tudíž probíhat pouze po jedno kabelu typu USB připojeným k PC.

Obvod FT232RL dosahuje přenosové rychlosti 300 Bit/s až 3 MBit/s. Obsahuje dvouportovou vyrovnávací paměť velikosti 128 bajtů ve směru od PC k aplikaci a 384 bajtů od aplikace k PC. K dispozici je i plně hardwarové řízení přenosu - signály RTS, CTS, DTR, DSR, DCD a RI. Dále má modul výstupní hodinový signál pro řízení mikrořadiče nebo FPGA 48MHz, 24MHz,12MHz, a 6MHz. Obvod je

kompatibilní s USB 2.0 a má integrovanou EEPROM paměť o velikosti 1024 Bytů pro konfiguraci I/O a uložení USB VID, PID, SN a informací o modulu. Na obrázku Obr 14. je uvedeno blokové schéma modulu.



Obr. 14 Blokové schéma obvodu FT232RL

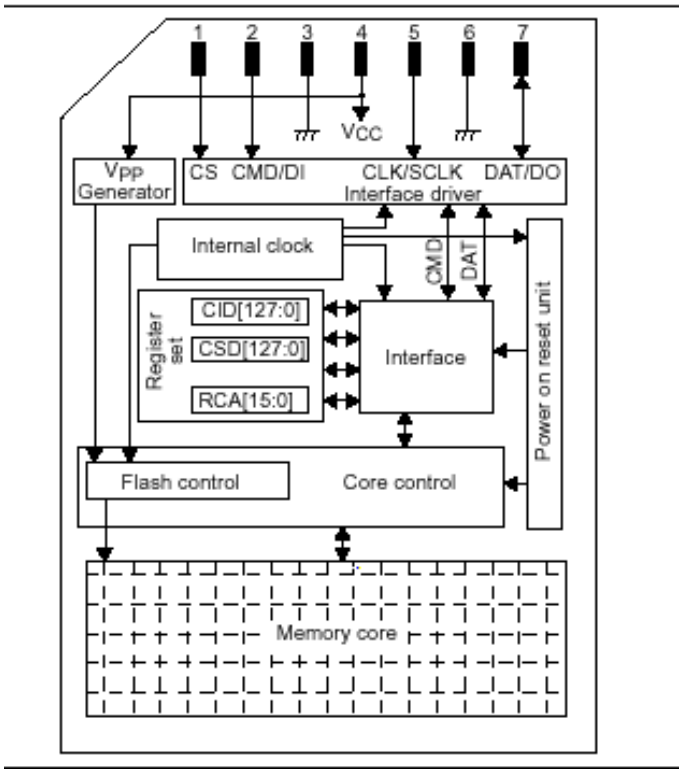
4.6. Zdroj bitstreamu

Jako zdroj bitstreamu pro obvod FPGA slouží paměťové karty typu MMC (MultiMedia Card) nebo SD (Secure Digital). Je to umožněno tím, že čtečka SD karet podporuje i paměťové karty typu MMC.

4.6.1. MMC karta

Tento paměťový standart byl oznámen v listopadu 1997 firmami Siemens a SanDisk. Tyto karty jsou vzdáleně příbuzné ke Smart Media, jsou ale nepatrně menší a odolnější, jejich rozměry jsou 32 x 24 x 1,4 mm. Obsahují v sobě speciální rozhraní, které umožňuje celkem rychlý sériový zápis dat do paměti (200kB/s zápis, 2MB/s čtení). Standard paměťových karet MMC je propagován jako otevřená

platforma. Standardní MMC karty lze ve většině případů použít v SD zařízeních, nikoliv však naopak. Tento formát byl vyvinut především pro oblast digitálního záznamu zvuku a obrazu, nicméně používají se pro svůj malý rozměr v mobilních komunikačních systémech, jako jsou mobilní telefony nebo MP3 přehrávače. Pozdní nástup a výhradní výroba určena pouze pro dvě firmy však nechali realitu za očekáváním a tak toto médium není zdaleka tolik rozšířené, jak se zprvu očekávalo. Schéma zapojení paměťové karty MMC na obrázku níže bylo použito z dokumentace SLA F0016 H CA firmy Infineon Technologies AG. [7]



Obr. 15 Schéma zapojení paměťové karty MMC

Rozhraní této karty komunikuje pomocí sedmi pinů. Karta může být připojena pomocí dvou módů. Jde o módy SPI a MultiMedia Card. Popis pinů a jejich funkce v těchto módech jsou uvedeny v tabulkách Tab. 6 a Tab. 7.

Pin	Signál	Popis
1	RSV	Nepoužitý pin nebo vždy log. 1
2	CMD	Příkaz, požadavek
3	PWR	Ground
4	PWR	Napájení +3,3V
5	CLK	Hodinový signál
6	PWR	Ground
7	DAT	Data

Tab. 6 Použití pinů v módu MMC

Pin	Signál	Popis
1	CS	Výběr zařízení (chip select)
2	DI	Data in
3	PWR	Ground
4	PWR	Napájení +3,3V
5	CLK	Hodinový signál
6	PWR	Ground
7	DO	Data out

Tab. 7 Použití pinů v módu SPI

Při použití standardního protokolu MultiMediaCard jsou používány tři vodiče pro komunikaci. Zbytek jsou vodiče napájecí. Signál CMD se používá obousměrně na zasílání příkazů a také na přijímání odpovědí. Signál DAT se používá také obousměrně. Hodinový signál CLK je řízen masterem a je zasílán směrem ke kartě.

V módu SPI jsou používány také tři komunikační vodiče. Je zde ale navíc ještě signál CS, je to zkratka Chip Select. Jak název napovídá tento signál slouží pro výběr aktivního zařízení. Obousměrné signály CMD a DAT v módu MMC jsou v tomto módu nahrazeny jednosměrnými signály DI (Data In) a DO (Data Out). Signál CLK je kartou opět jen přijímán.

Životnost paměťových karet MMC je vyjádřena pomocí počtu zápisů do ní, pohybuje se okolo 100 000 zápisů. Velkou výhodou karet tohoto typu je i automatický přechod karty do režimu spánku. Jakmile karta vyřídí všechny požadavky od hostitele přejde do režimu spánku. Když přijde nový požadavek ihned z režimu spánku zase vystoupí. Má to příznivý vliv na spotřebu karty. Spotřeba v režimu spánku se pohybuje okolo 150uA, kdežto v režimech čtení nebo zápis je to do 60mA. Při správně navržené čtečce karet je možno i připojit kartu za běhu zařízení.

4.6.2. SD karta

Tato paměť se fyzickou velikostí příliš neliší od karet MMC, je o 0,7mm silnější. Hlavní rozdíl je hlavně ve vnitřním uspořádání karty. Rozhraní paměti je tvořeno devíti kontakty (MMC má kontaktů sedm) a vnitřní architektura umožňuje vysokou ochranu proti zkopírování obsahu. SD je dnes velice oblíbené médium pro svoji velikost, odolnost a jednoduchou výrobu. Karty SD mají ochranu dat zvýšenou také pomocí implementované technologii SDMI pro ochranu copyright-ových zvukových nahrávek a také mechanickým přepínačem pro ochranu zápisu.

Karta může být připojena pomocí tří módů. Jde o módy SPI, 1-BIT a 4-BIT. Funkce jednotlivých pinů těchto módů jsou uvedeny v tabulce Tab.8. [3]

Pin	**SPI**		**1-BIT**		**4-BIT**	
1	CS	Výběr karty	NC	Nepoužitý pin	CD/DAT[3]	Datový vodič 3
2	DI	Data vstup	CMD	Příkazy	CMD	Příkazy
3	PWR	Ground	PWR	Ground	PWR	Ground
4	PWR	+3,3V	PWR	+3,3V	PWR	+3,3V
5	CLK	Hodinový signál	CLK	Hodinový signál	CLK	Hodinový signál
6	PWR	Ground	PWR	Ground	PWR	Ground
7	DO	Data výstup	DAT	Datový vodič	DAT[0]	Datový vodič 0
8	IRQ	Interrupt	IRQ	Interrupt	DAT[1]	Datový vodič 1
9	NC	Nepoužitý pin	RW	Read Wait	DAT[2]	Datový vodič 2

Tab. 8 Módy SD karet

Mód SPI komunikuje pomocí čtyř vodičů, Jde o signály DI (data in), DO (data out), CLK a signál IRQ pro přerušení. Zbytek vodičů je napájení. 1-BITový mód komunikuje také po čtyřech vodičích. Rozdíl oproti SPI je, že pin 2 je obousměrný signál CMD, po kterém jsou hostitelem posílány příkazy a odpovědi kartou. Na pin 7 je připojen obousměrný signál DAT, po kterém jsou posílány data. 4-BITový mód má na druhém pinu také signál CMD, ale data jsou posílány v šířce čtyř bitů po čtyřech datových signálech.

Já ve svém zařízení použiji pro připojení SD karty mód SPI, který je kompatibilní i s MMC kartou.

4.7. Návrh zdroje zařízení

V této části návrhu již máme všechno potřebné k návrhu zdroje zařízení. Zdroj bude navrhnut podle spotřeby všech součástek v zařízení. Nejprve je třeba si rozdělit obvody podle napájecího napětí. Celé zařízení vyžaduje tři napájecí napětí. A to 1,8V a 3,3V, to jsou napětí pro obvod FPGA. Dále napájecí napětí 5V pro mikrořadič a další součástky.

Napětím 1,8V je napájeno jádro FPGA, jeho spotřeba dosahuje maximálně 300mA, typicky je však 10mA.

Součástka	Typ	Spotřeba
FPGA	XC2S200E	Typicky 10mA, Max. 300mA(spotřeba jádra)
		Celkem 300mA

Tab. 9 Součástky napájené napětím 1,8V a jejich spotřeba

Na napájecí napětí pro jádro FPGA nejsou kladeny žádné vysoké nároky. Toto napětí může být maximálně 2V a minimální mez je 1,6V. Jako dostačující pro tento účel jsem zvolil obvod LM317HVT, je to lineární stabilizátor s nastavitelným výstupním napětím pomocí odporového děliče. Tento stabilizátor může v pouzdře TO220 a s přídavným pasivním chladičem dodávat proud až 1,5A. V mém případě z něj budu odebírat maximálně proud 300mA, proto nechám obvod bez pasivního chladiče a ve vertikální poloze. Základní parametry tohoto stabilizátoru jsou uvedeny v tabulce Tab. 10.

Značka	Parametr	Hodnota
Uout	Výstupní napětí	1,2 až 57V
Iout	Max. výstupní proud	1,5A
Uin	Max. vstupní napětí	60V
	Pouzdro	TO220
	Cena	35Kč

Tab. 10 Parametry stabilizátoru LM317HVT

Napájení 3,3V napájí tři součástky. Jednou z nich je SD paměť, největší odběr elektrického proudu má v režimu zápis, a to přibližně 60mA. Spotřeba v režimu čtení se pohybuje okolo 50mA. V režimu “stanby“ je pak spotřeba okolo 6mA. Další

součástky napájené napětím 3,3V jsou pak FPGA (I/O banky) a krystalový oscilátor, který slouží jako zdroj hodinového signálu pro FPGA. Maximální spotřeba tohoto typu krystalového oscilátoru dosahuje hodnoty 35mA.

Součástka	**Typ**	**Spotřeba**
SD Paměť		Režim zápis: 56.7mA Režim čtení: 47.3mA
Krystalový oscilátor	SG531HVT	35mA
FPGA	XC2S200E	(spotřeba I/O bank)
		Celkem 600mA

Tab. 12 Součástky napájené napětím 3,3V a jejich spotřeba

Pro realizaci tohoto zdroje jsem zvolil opět nastavitelný, tentokrát však spínaný stabilizátor. Jde o obvod LM2575T-ADJ v pouzdře TO 220-5. K tomuto stabilizátoru stačí přidat pouze malé množství dalších součástek a zdroj je hotov. Zdroj s tímto stabilizátorem je schopen dodávat elektrický proud až 1A. Výstupní napětí je nastavitelné v rozmezí 1,23V až 37V. Odchylka výstupního napětí od nastaveného napětí dosahuje maximálně 4%. Vysoká účinnost 82 procent tohoto stabilizátoru zaručuje nízký ztrátový výkon a tedy i nižší požadavky na chlazení obvodu. Základní parametry tohoto spínaného stabilizátoru jsou uvedeny v tabulce níže Tab. 13.

Značka	**Parametr**	**Hodnota**
Uout	Výstupní napětí	1,23 až 37V
Iout	Max. výstupní proud	1A
Uin	Max. vstupní napětí	40V
N	Účinnost	82%
	Pouzdro	TO220-5
	Cena	53Kč

Tab. 13 Parametry stabilizátoru LM2575T-ADJ

K napájecímu napětí 5V jsou připojeny dvě důležité součástky, jsou to mikrořadič ATMega2560 od firmy Atmel a převodník USB RS232 od firmy Future Technology Devices International Ltd., jde o obvod FT232RL. Mikrořadič má spotřebu 400mA. Obvod FT232RL má maximální spotřebu 15mA. Pěti Voltami jsou

napájené ještě dvě LED diody, jejich spotřeba se pohybuje okolo 5mA. Tabulka Tab. 14se součástkami a jejich spotřeba je uvedena níže.

Součástka	**Typ**	**Spotřeba**
Mikrořadič	ATMega2560	400mA
Převodník USB RS232	FTD232RL	15mA
2x LED		5mA
		Celkem 420mA

Tab. 14 Součástky napájené napětím 5V a jejich spotřeba

Celková spotřeba součástek připojených k napájení 5V se pohybuje okolo 420mA. Pro tento zdroj jsem vybral spínaný stabilizátor s pevným výstupným napětím 5V. Tento stabilizátor je schopen dodávat elektrický proud až 1A, což je podle odhadované celkové spotřeby dostačující. U tohoto stabilizátoru dosahuje výstupní napětí odchylky maximálně 4%, jeho účinnost je 77 procent a pracuje na frekvenci 52kHz. Parametry jsou uvedeny v tabulce Tab. 15.

Značka	**Parametr**	**Hodnota**
Uout	Výstupní napětí	5V
Iout	Max. výstupní proud	1A
Uin	Max. vstupní napětí	40V
N	Účinnost	77%
	Pouzdro	TO220-5
	Cena	33Kč

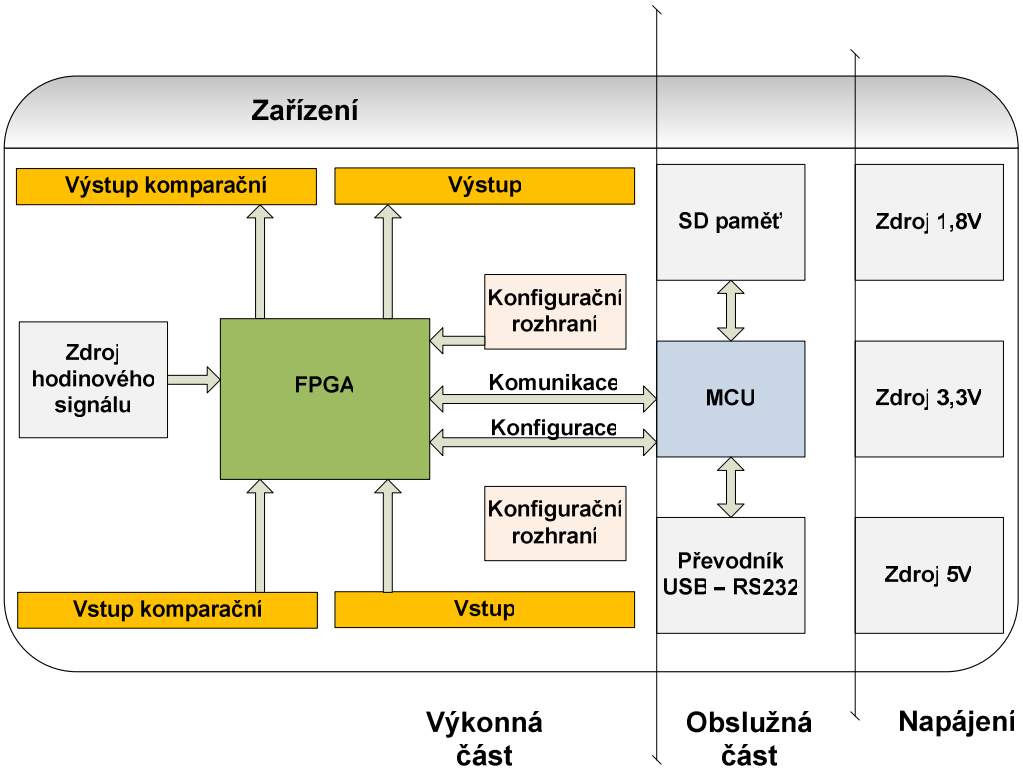
Tab. 15 Parametry stabilizátoru LM2575T-05

5. Návrh řešení

Zařízení lze obecně rozdělit na dvě části, část výkonnou a část obslužnou. Srdcem výkonné části bude obvod FPGA. Pro komunikaci tohoto obvodu s okolním světem budou ze zařízení vyvedeny vstupní a výstupní porty. Další dva vstupní a výstupní porty budou připraveny pro komunikaci s okolními zařízeními.

Srdcem obslužné části bude MCU. Tato část má za úkol obsluhovat část výkonnou, respektive obvod FPGA, který informuje MCU o svém stavu a MCU na ně bude reagovat. Převodník USB – RS232 bude zajišťovat komunikaci mezi zařízením a počítačem. SD paměť bude sloužit v zařízení jako úložiště dat i jako zdroj dat, především na ní bude uložen bitstream pro obvod FPGA. Budou se do ní ukládat i diagnostická data a informace o chodu zařízení. Tzn. počet konfigurací, jaká chyba to způsobila atd.

Zařízení vyžaduje tři úrovně napájecího napětí, jsou tedy v zařízení tři zdroje těchto napájecích napětí. Jedná se o napětí 1,8V, pro jádro obvodu FPGA. Napětí 3,3V pro I/O banky obvodu FPGA, napájení SD paměti a zdroje hodinového signálu pro FPGA. Napětí 5V napájí MCU a převodník USB – RS232. Blokové schéma zařízení je uvedeno na obrázku Obr.16.



Obr. 16 blokové schéma zařízení

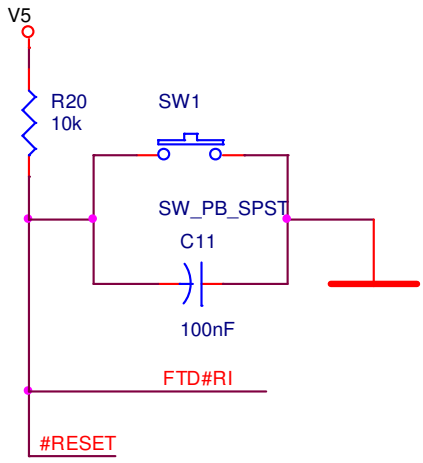
6. Řešení

V této kapitole provedu rozbor řešení. Vyberu vždy nějakou část schématu a detailně ji popíši a odůvodním.

6.1. Mikrořadič

V této části detailně rozeberu schéma zapojení mikrořadiče. Mikrořadič je napájen napětím 5V. Tento mikrořadič umožňuje napájení v rozsahu 2,5V až 5,5V, ale pro pracovní frekvenci 16MHz je vyžadováno napájecí napětí v rozmezí 4,5V až 5,5V. Napájení je do obvodu přivedeno po čtyřech pinech VCC.

Reset tohoto obvodu je aktivní v logické nule, proto je na tento pin přivedeno napětí 5V přes pull-up rezistor velikosti 10kOhmů. Za tímto rezistorem je připojeno resetovaní tlačítko a k němu je paralelně připojen kondenzátor velikosti 100nF sloužící jako ochrana proti zákmitům vznikajícím při stisku nebo uvolnění tlačítka. Při stisku tlačítka je napětí za rezistorem svedeno k zemi a reset se stává aktivním, zmíněný rezistor nastavuje protékající elektrický proud. Schéma je uvedeno na obrázku Obr 16.



Obr. 16 Schéma resetu

Jelikož je v obvodu nepoužitý analog/digital převodník jsou důležité piny AVCC (analogové napájení) a AREF (vnější referenční napětí) připojené taktéž na napájecí napětí 5V. Programování je zajištěno metodou ISP (In – System Programming). Tato metoda umožňuje programování mikrořadiče přímo v zařízení pomocí vyhrazených

vodičů. Metodou ISP nelze nahrávat (aktualizovat) firmware zařízení bez toho, aby nedošlo k přerušení běžícího programu tzn. programovaný obvod je nutné přepnout do režimu programování. Pro tuto metodu se běžně používají rozhraní SPI (Serial Peripheral Interface), JTAG (Joint Test Actoin Group) nebo také USART (Universal Synchronous/Asynchronous Receive and Transmitter). Já jsem použil jednoduché sériové synchronní rozhraní SPI. Používá tří vodičů a je plně duplexní.

- **MOSI** (Master Out Slave In)
- **MISO** (Master In Slave Out)
- **SCK (**Clock**)**

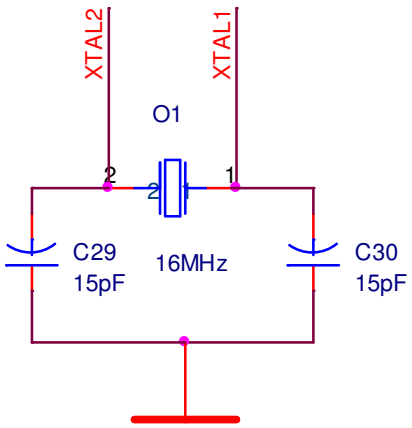
Dva vodiče jsou datové vstup/výstup a jeden vodič hodinový. Na programovací konektor jsem ještě vyvedl reset, který je pro metodu ISP potřebný. V tabulce Tab.16 jsou uvedeny funkce a označení pinů programovatelného rozhraní ISP.

Označení	**Funkce**
VCC	Napájecí napětí 5V
GND	Nulový potenciál GND
SCK	Vstup hodinového signálu
MISO	Sériový datový vstup
MOSI	Sériový datový výstup
RESET\	Resetovací signál

Tab.16 Označení a funkce pinů programovatelného rozhraní ISP

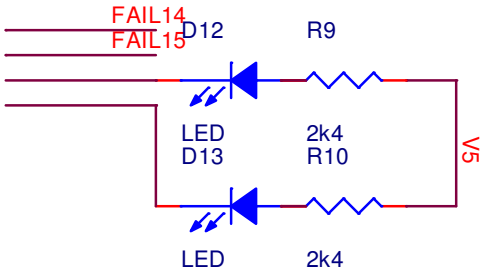
K sériovému rozhraní SPI je ještě připojen obvod FT232RL, jde o převodník USB na RS232. Tento převodník je připojen ještě k jednomu ze čtyř USARTů, které má mikrořadič dostupné. Přes USART může zařízení komunikovat s PC a zároveň lze přes rozhraní SPI konfigurovat mikrořadič.

Externí hodinový signál pro mikrořadič zajišťuje krystal připojený na piny XTAL1 a XTAL2. Mezi nožičky krystalu a zem jsem připojeny dva kondenzátory velikosti 15pF. Krystal kmitá frekvencí 16MHz. Je třeba mít na paměti, že při oživení mikrořadiče pracuje mikrořadič s hodinovým signálem který generuje interní oscilátor. Teprve po nastavení fuse bitů je jako hodinový signál použit signál z externího krystalu. Schéma zapojení je uvedeno na obrázku Obr 17.



Obr. 17 Schéma zapojení krystalu

K mikrořadiči jsou připojeny dvě ladící LED diody. Jsou aktivní v logické nule. Elektrický proud je nastaven rezistory R9 a R10 velikosti 2,4kOhmů. LED diody jsou napájeny napětím 5V. Schéma zapojení LED diod je uvedeno na obrázku níže.



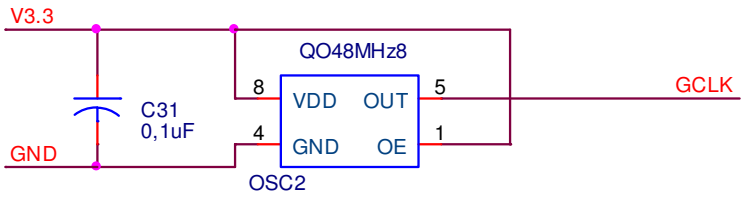
Obr. 18 Schéma zapojení diod LED

Pro komunikaci mezi mikrořadičem a obvodem FPGA slouží 32 vodičů. Jde o 16 signálů OK a 16 signálů FAIL. Dále je k mikrořadiči připojena SD Paměť. Jelikož je už rozhraní SPI použito pro programování přes obvod FT232RL, který slouží také ke komunikaci a komunikace musí být aktivní také při čtení z SD paměti, nechtěl jsem už k rozhraní SPI připojit SD paměť. Proto je SD paměť připojena pouze k obyčejným I/O portům a rozhraní SPI pro paměť bude řešeno softwarově.

6.2. FPGA

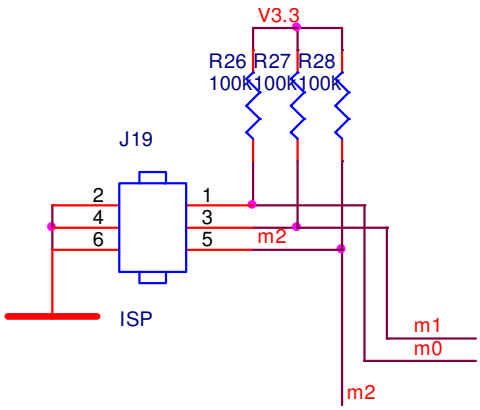
Obvod FPGA je připojen na dvě napájecí napětí. Napětí 3,3V je přivedeno na piny Vcco, toto napětí slouží jako napájení I/O bank. Jádro obvodu je napájeno u tohoto obvodu XC2S200E napětím 1,8V a je přivedeno na piny Vccint.

Jako zdroj hodinového signálu jsem použil krystalový oscilátor SG531PH generující hodinový signál o frekvenci 48MHz. Tento obvod je napájen napětím 3,3V a má u sebe umístěn blokující kondenzátor o velikosti 100nF. Hodinový signál je přiveden do FPGA přes pin GCLK0. Není to ale jediný pin, po kterém lze přivést hodinový signál do obvodu FPGA, jsou celkem čtyři GCLK0 až GCLK3. V mém případě zůstaly zbylé piny GCLK1 až GCLK3 nezapojené. Schéma zapojení krystalového oscilátoru je uvedeno níže Obr.19.



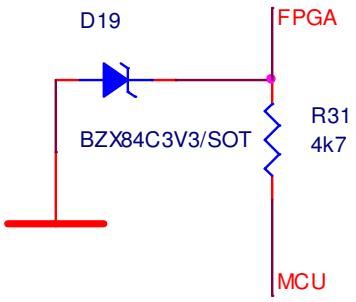
Obr. 19 Schéma zapojení krystalového oscilátoru

Mód jakým se bude obvod FPGA konfigurovat se nastavuje piny M2, M1, M0, jak je uvedeno v kapitole analýza. Protože používám sériovou konfiguraci, ale také záložní rozhraní JTAG, připojil jsem konfigurační piny na napětí 3,3V přes pull-up rezistory velikosti 100kOhmů. K pinům jsou ještě připojeny přepínače pomocí kterých je možno nastavit logické hodnoty na těchto pinech a tím i vybrat konfigurační mód. Nejsou-li přepínače nastaveny je vybrán sériový konfigurační mód. Na schématu níže je uvedeno celé zapojení Obr.20.



Obr. 20 Schéma zapojení mód pinů

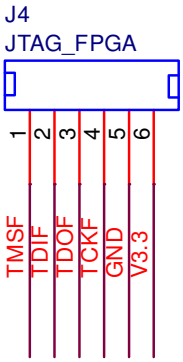
Sériové konfigurační rozhraní používá piny DONE, PROGRAM, INIT, DIN, DOUT. Všechny tyto piny pracují s napětím 3,3V. Jelikož toto konfigurační rozhraní není tolerantní k pěti voltové logice a musí být připojeno k MCU, který pracuje s napětím 5V, musel jsem všechny signály tohoto rozhraní směrem od MCU k FPGA upravit. Pro úpravu napětí jsem použil zapojení se zenerovou diodou a rezistorem uvedené na obrázku Obr.21.



Obr. 21 Schéma zapojení obvodu pro úpravu napětí

MCU ATMega2560 je tolerantní k logice pracující s napětím 3,3V, proto není třeba upravovat napětí směrem od FPGA k MCU. Směrem od MCU je v cestě před Zenerovou diodou připojen rezistor 4k7. Zenerova dioda se otvírá při napětí vyšším než 3,3V a zbylé napětí svádí k zemi. Elektrický proud tekoucí diodou k zemi nastavuje právě zmíněný rezistor. Ve směru od FPGA nemusí být před diodou připojen žádný rezistor, protože při napětí 3,3V zůstává dioda uzavřena => neteče žádný elektrický proud k zemi => není potřeba omezit elektrický proud.

Jako záložní a testovací rozhraní pro obvod FPGA jsem použil rozhraní JTAG. Signály tohoto rozhraní jsem vyvedl na konektor tak, aby uspořádání signálů bylo kompatibilní s programovacím konektorem a kabelem od firmy Xilinx. Zapojení konektoru je uvedeno na obrázku Obr.22.



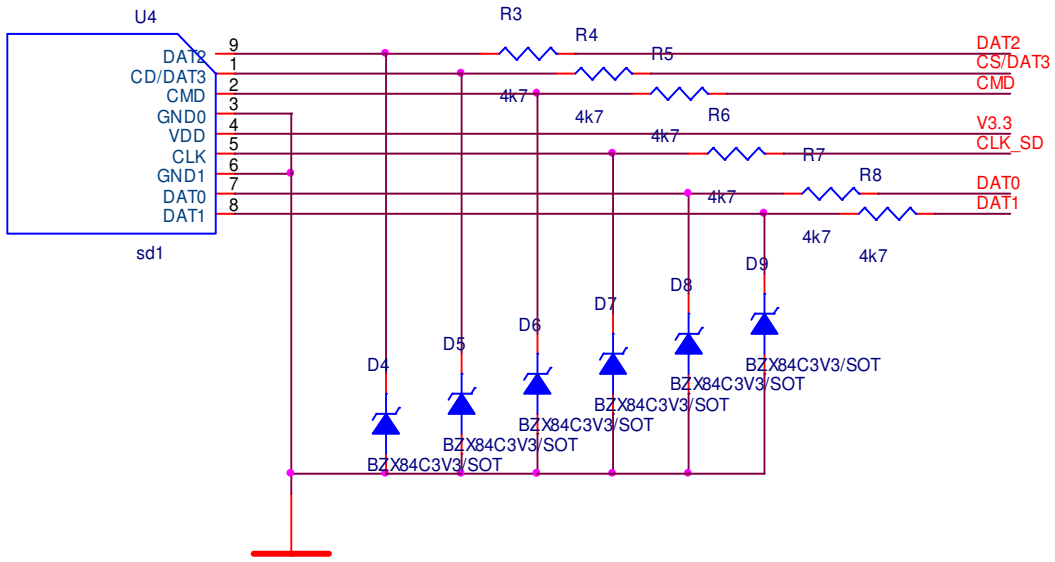
Obr. 22 Schéma zapojení JTAG konektoru

Dále je obvod FPGA propojen s MCU signály OK a FAIL. Jde o 16 signálů OK a 16 signálů FAIL. Žádný z těchto signálů není napěťově upravován, jako tomu bylo u spojení MCU a FPGA u konfiguračního rozhraní, protože I/O porty obvodu FPGA jsou tolerantní k napětí 5V. Pro komunikaci mezi zařízeními (obvody FPGA) slouží dva komunikační porty (komparační výstup, komparační vstup). Každý tento port je tvořen také 16 signály. Další dva porty jsou vstupní port a výstupní port zařízení, každý opět po 16 signálech. Detailnější popis všech těchto signálů a propojení zařízení je uveden v kapitole ANALýZA. K obvodu FPGA jsou ještě připojeny dvě ladící LED diody, schéma zapojení je stejné jako u LED diod u MCU uvedeného v odstavci výše.

6.3. SD paměť

SD paměť, která slouží v zařízení jako zdroj dat, je možno připojit k MCU více způsoby jak je uvedeno v kapitole analýza. Já jsem použil rozhraní SPI, nicméně paměť není připojena k hardware-ově řešenému rozhraní v MCU nýbrž ke klasickým I/O portům MCU. Rozhraní SPI pro SD paměť je řešeno software-ově. Je to z toho důvodu, že ke klasickému rozhraní SPI je už připojen převodník rozhraní USB – RS232, jímž je možné MCU konfigurovat a je vyžadována současná funkce SD paměti a převodníku USB - RS232.

Paměťová karta tohoto typu pracuje s napětím 3,3V a bylo tedy nutné opět upravit napětí směrem od MCU k paměti. Použil jsem stejné zapojení s rezistorem a Zenerovou diodou jaké je popsáno výše v podkapitole FPGA.

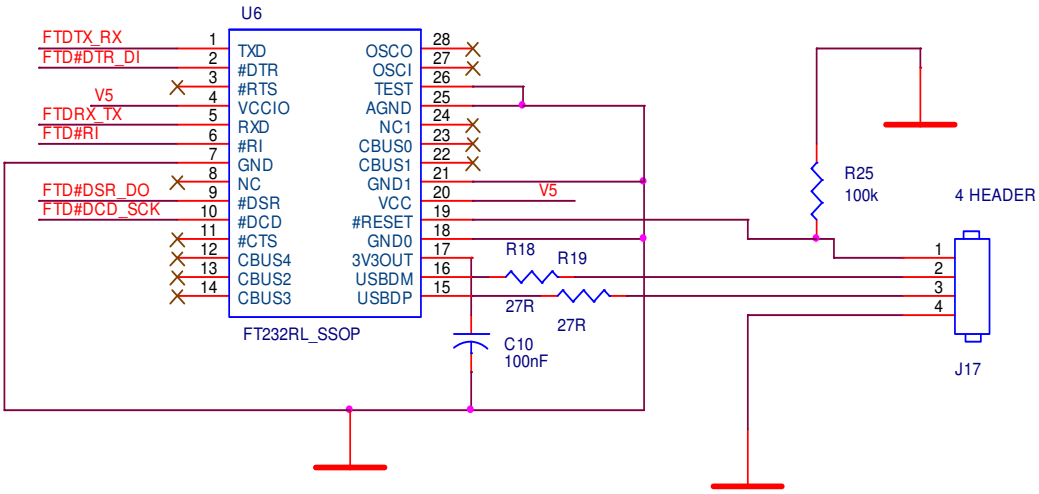


Obr. 23 Schéma zapojení SD paměti

Všechny signály paměťové karty jsem připojil k MCU, i když nebudou při komunikaci přes rozhraní SPI využity. Nicméně takto připojená SD paměť dovoluje v budoucnu, bude-li třeba, použití kteréhokoliv z komunikačních módů, které karta podporuje.

6.4. Převodník USB – RS232

Obvod FT232RL detailněji popsaný v kapitole analýza plní v zařízení dvě funkce. Jednak funkci komunikační, je možné komunikovat s PC přes sériovou linku. Jeho další funkcí je konfigurace MCU přes sériové programovací rozhraní. Způsob připojení převodníku k sériovému programovacímu rozhraní a aplikace v PC, potřebná pro konfiguraci, je součástí bakalářské práce studenta Romana Nesvadby.



Obr. 24 Schéma zapojení SD paměti

Signály převodníku TXD A RXD jsou přivedeny na jeden z USARTů mikrořadiče a slouží pro komunikaci s PC. Ostatní signály sériového rozhraní RS232 jsou připojeny právě k sériovému programovacímu rozhraní MCU. Plní tedy jinou funkci, danou aplikací v PC, která jimi hýbe tak, jak to vyžaduje konfigurační sekvence. Signál #DCD zajišťuje hodinový signál, signál #DTR je datový vstup (DI data in), signál #DSR datový výstup (DO data out) a signál #RI je resetovaní signál pro MCU. Signály rozhraní USB (USBDM, USBDP) jsou přes rezistor 27 ohmů připojeny na konektor USB B dle schématu výše Obr 24. Převodník není napájen z PC přes USB kabel, je napájen vlastním napětím 5V ze zařízení. Napětí 5V přivedené přes USB kabel je použito pro resetování převodníku. Je připojeno přes pull-down rezistor velikosti 100 kOhmů na signál #RESET tak, že při odpojeném USB kabelu se převodník nachází ve stavu RESET. Naopak po jeho připojení kabelem USB se signál #RESET stává neaktivním, převodník vystoupí ze stavu RESET, a je připraven k použití.

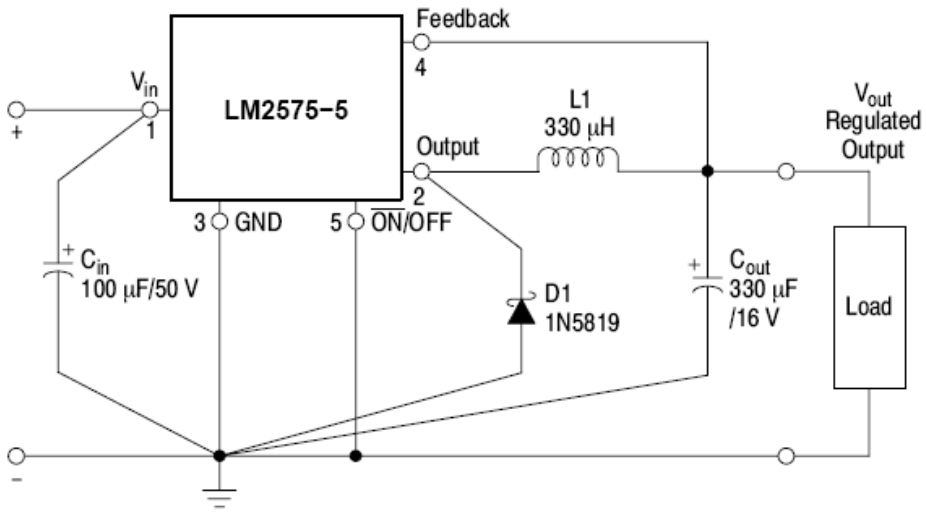
6.5. Komunikace mezi MCU

Každé zařízení (MCU) komunikuje se svými sousedy. Pro každého souseda existuje vlastní komunikační linka nezávislá na dalším sousedovi. Použil jsem sériových rozhraní USART. Jakékoli zařízení v systému složeného z více zařízení má při propojení do kružnice dva sousedy ("NAD a POD"). Signály RX a TX jednoho z USARTů jsou přivedeny na druhý USART sousedního zařízení křížem tzn RX --> TX a TX --> RX. Druhý USART je připojen stejným způsobem na druhého souseda. Tyto signály jsem připojil na konektor sloužící k přenosu dat pro porovnávání, takže zařízení jsou propojeny pouze jedním kabelem. Kabel je možné připojit i lokálně na zařízení z konektoru FPGA_KOMPARACNI_OUT na konektor FPGA_KOMPARACNI_IN, takže zařízení je možno odladit pouze s jedním kusem, což také zaručuje univerzálnost zařízení pro stavbu jakkoli velkého systému.

6.6. Realizace napájecích zdrojů

Zdroj 5V

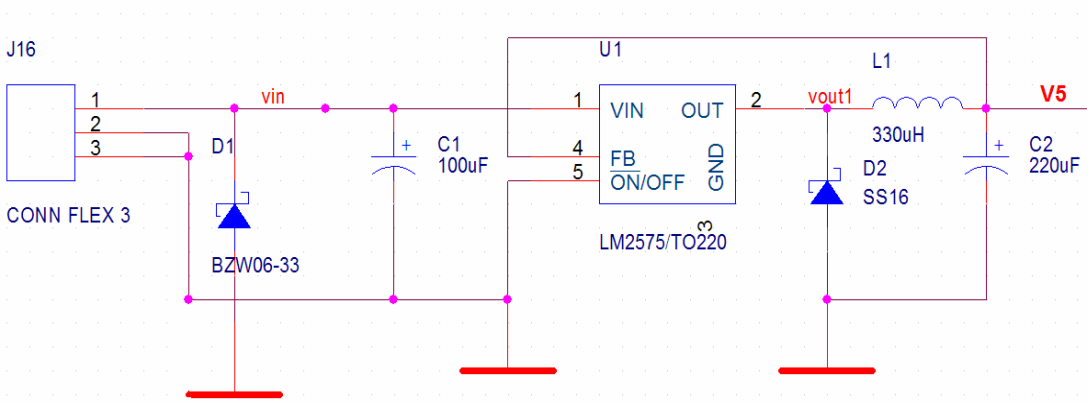
Na napájecí zdroj 5V byly kladeny následující požadavky. Vstupní napětí v rozsahu 0 až 30V, výstupní napětí 5V s malým zvlněním, výstupní proud až 1A a přijatelně velká plocha zdroje na desce plošného spoje. Proto byl zvolen jako srdce zdroje spínaný stabilizátor typu LM2575T-05. Tento obvod potřebuje ke své činnosti jen minimum součástek a zdroj je v podstatě hotov. Jeho vysoká účinnost zaručuje nízký ztrátový výkon, takže nepotřebuje nijak velký přídavný chladič. To je velká úspora místa na desce plošného spoje. Doporučené zapojení uvedené v literatuře [4] je zobrazeno na obrázku Obr.25.



Obr. 25 Doporučené zapojení obvodu LM2575T-05

Z obrázku Obr.26. je vidět, že je pro plnou funkci obvodu potřeba přidat pouze čtyři součástky. Jsou to vstupní a výstupní kondenzátory, tlumivka a dioda.

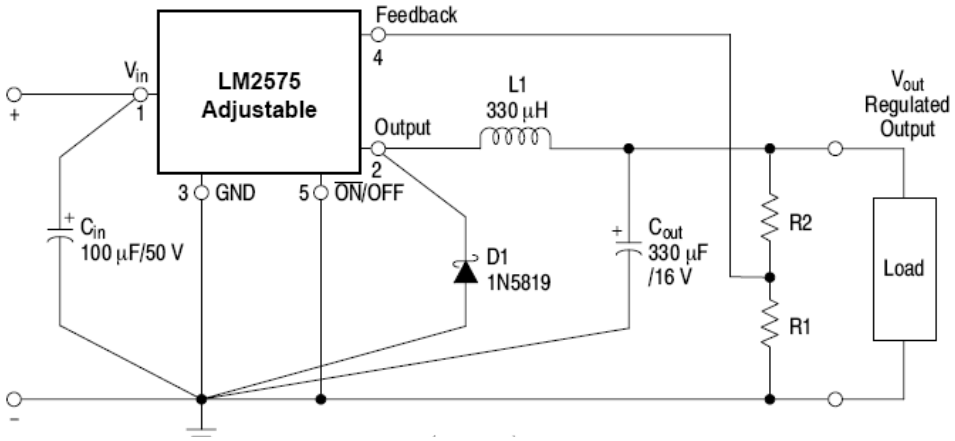
Při návrhu mého zdroje jsem se řídil doporučeními uvedenými v literatuře [4]. Spínaný stabilizátor je typu step-down se spínacím tranzistorem integrovaným na čipu. Vstupní napětí je přivedeno přes konektor J16 na vstupní pin VIN stabilizátoru. Ke vstupnímu napětí je připojen transil, který slouží jako ochrana proti napěťovým špičkám. Dále je k němu připojen vstupní tantalový kondenzátor o velikosti 100uF. Na výstupním pinu OUT je připojen už vlastní energetický obvod složený z tlumivky L1 a rychlé shottkyho diody. Na výstupu tlumivky je připojen ještě elektrolitický kondenzátor 220uF, který slouží jako dočasná zásoba elektrické energie než zareaguje zpětná vazba obvodu, která je připojena také k tomuto místu, na případný nárůst odběru elektrické energie. Takto navržený zdroj, jak je uveden na obrázku Obr.26, je schopen dodávat elektrický proud ž 1A.



Obr. 26 Schéma zdroje 5V se stabilizátorem LM2575T-05

Zdroj 3,3V

Na napájecí zdroj 3,3V byly kladeny obdobné požadavky jako na napájecí zdroj 5V. V podstatě jen s tím rozdílem, že výstupní napětí je 3,3V. Jako srdce zdroje byl opět zvolen spínací stabilizátor LM 2575T, tentokrát však s příponou ADJ, což znamená, že výstupní napětí je nastavitelné pomocí odporového děliče v rozmezí 1,23V až 37V, jak je uvedeno na obrázku Obr.27.

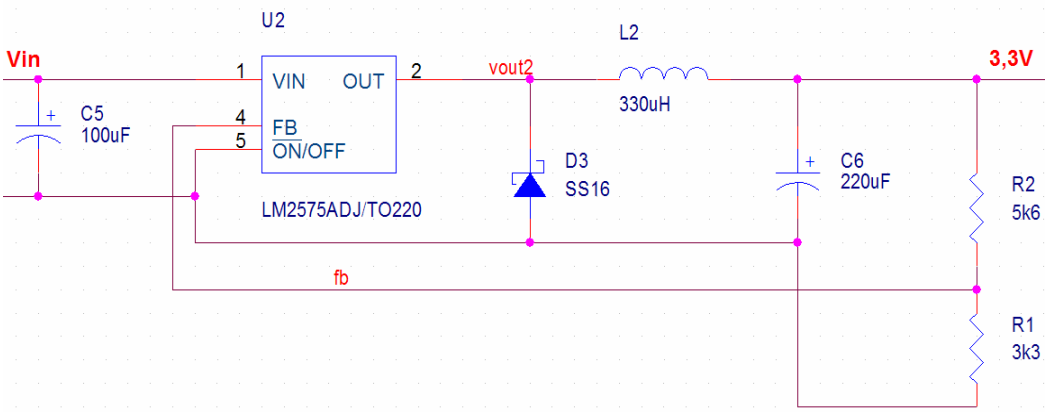


Obr. 27 Doporučené zapojení obvodu LM2575T-ADJ

Hodnoty odporů odporového děliče se vypočtou pomocí vztahu uvedeného níže. Kde Vref je 1,23V a hodnota odporu R1 se musí pohybovat v rozmezí 1kOhm až 5kOhmů.

$$V_{out} = V_{ref}\left(1 + \frac{R2}{R1}\right)$$ nebo lze z něho odvodit vztah $$R2 = R1\left(\frac{V_{out}}{V_{ref}} - 1\right)$$

Moje realizace zdroje 3,3V je uvedena na obrázku Obr.28. Hodnoty odporů R2 a R1 jsem volil tak, aby odpovídaly vyráběným řadám odporů. Jinak jsou schéma i ostatní součástky shodné se zdrojem napětí 5V. Takto navržený zdroj je opět schopen dodávat elektrický proud až 1A.



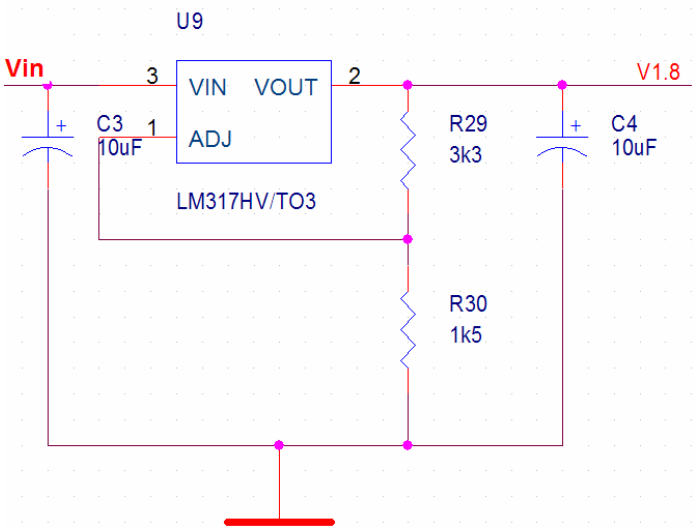
Obr. 28 Schéma zdroje 3,3V se stabilizátorem LM2575T-ADJ

Zdroj 1,8V

Na zdroj napětí 1,8V jsou kladeny některé požadavky nižší, než u obou předchozích. Především jde o maximální odebíraný proud ze zdroje. Ten v tomto případě dosahuje pouze 300mA a to jen nárazově, typicky se odebíraný proud bude pohybovat kolem 10mA. Tímto zdrojem je napájeno pouze jádro obvodu FPGA. Jako srdce zdroje byl tentokrát zvolen pouze lineární stabilizátor LM317 ADJ, který bude výše uvedeným požadavkům dostatečně vyhovovat. K jeho plné funkci je třeba připojit ještě menší počet součástek, než u spínaných zdrojů výše. Odpadne hlavně nutnost mít na výstupu připojenou tlumivku, která je dosti velká a může být zdrojem elektromagnetického rušení. Zmíněný stabilizátor je opět ve verzi s nastavitelným výstupním napětím. Napětí je nastaveno pomocí odporového děliče, kde hodnoty odporů jsou dány vztahem uvedeným níže.

$$V_{OUT} = 1.25V\left(1 + \frac{R2}{R1}\right) + I_{ADJ}R2$$

Hodnota odporu R1 musí být v rozmezí 1kOhm – 5kOhm. Hodnoty jsem volil tak, aby odpovídali vyráběným řadám odporů. Mezi vstup stabilizátoru a zem je připojen vstupní tantalový kondenzátor 10uF. Stejný kondenzátor je připojen i na výstup stabilizátoru. Schéma zdroje je uvedeno na obrázku Obr.29.



Obr. 28 Schéma zdroje 1,8V

6.7. Návrh plošného spoje

Plošný spoj je navržen na podkladech vygenerovaných pomocí schématického editoru. Jde o seznam součástek a vodičů a jejich propojení. Takto vygenerovaný soubor (netlist) jsem načetl návrhovým editorem. Použil jsem systém orcad od firmy cadence.

Při návrhu jsem se snažil respektovat pravidla pro rozmísťování a propojování součástek tak, aby byla dodržena kritéria kladená na zařízení z hlediska elektromagnetické kompatibility (EMC). To znamená, že jsem se snažil jednak, aby zařízení bylo odolné proti elektromagnetickému rušení a také, aby nebylo jeho zdrojem. Mezi nejdůležitější pravidla tohoto návrhu patří. [6]

1) Minimalizace hodnot proudu = vhodně zvolit typy součástek, tak aby byl ve všech smyčkách co nejmenší elektrický proud
2) Minimalizace proudových smyček a délek spojů = vhodné rozmístění součástek, vedení spojů, správné blokování pomocí kondenzátorů, správná konfigurace napájení, vstupní a výstupní kabeláže atd.
3) Minimalizace kmitočtového spektra = zbytečně nepoužívat obvody pracující na vysoké frekvenci není-li to nezbytně nutné, zbytečně nepoužívat rychlou datovou komunikaci
4) Stínění = snažit se pomocí stínění potlačit vyzařování obvodů

Pro návrh jsem použil 4-vrstvou desku plošného spoje. Vnější vrstvy jsou použity pro vedení spojů. Vnitřní vrstvy jsou napájecí, první z nich je zemnící a druhá slouží pro rozvod napájecího napětí. Tyto vrstvy jsou celoplošně rozlité. Jak již víme, zařízení vyžaduje tři napájecí napětí, ale napájecí vrstvu jsem rozdělil pouze na dvě části a to na část 5V a na část 3,3V. Napětí 1,8 voltů pro jádro obvodu FPGA je vedeno ve vnější vrstvě (TOP). Součástky napájené napětím 5V jsou umístěny nad částí napájecí vrstvy 5V. Taktéž i pro napětí 3,3V. Tato nutná skutečnost omezila rozmístění součástek do nejvhodnějších pozic, jelikož součástky některého z napětí musely být pohromadě na jedné straně desky. Všechny součástky jsem se snažil umístit do vrchní vrstvy (TOP), nicméně některé blokovací kondenzátory jsem umístil do spodní vrstvy (BOTTOM), tak aby byly co nejblíže svým obvodům. Šlo jen o součástky typu SMD.

Deska je navržena v konstrukční třídě 5. Tato třída definuje parametry základních objektů na plošném spoji. Mezi ně patří minimální šířka spojů, izolační vzdálenost, minimální průměr vrtaného otvoru a minimální rozměr pájecí vrtané plošky. Hodnoty pro konstrukční třídu 5 jsou uvedeny v tabulce níže.

Třída přesnosti	5
Šířka spoje	8 milů
Min. izolační vzdálenost	8 milů
Min. Průměr vrtáku	20 milů
Min. průměr pájecí plošky	Min. průměr vrtáku + 16 milů
Min. průměr nepájivé masky	Min. průměr pájecí plošky + 8 milů

Tab.17 Konstrukční třída 5

Při návrhu je nutné znát tyto rozměry a snažit se je dodržovat ve vybrané třídě přesnosti, tomu odpovídají i ceníky různých výrobců. Čím jsou rozměry menší (vyšší třída přesnosti), tím je výroba plošného spoje dražší.

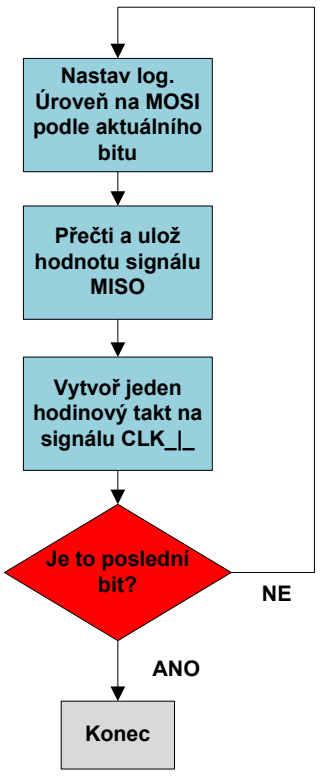
Po natažení všech spojů ve vnějších vrstvách jsem do volných míst rozlil zemnící vodič. To výrazně zlepší odolnost proti elektromagnetickému rušení. Po obvodu celé desky jsem umístil prokovy propojující zemní vodič přes všechny vrstvy, to vytvoří jakýsi rámeček okolo celé desky. Tento rámeček výrazně omezí vyzařování elektromagnetického rušení v horizontálním směru. V rozích desky plošného spoje jsou umístěny montážní otvory pro šroubek M3, jimiž je možno umístit plošný spoj do krabičky nebo třeba jen na nožičky.

7. Firmware

V této kapitole popíši firmware mikrořadiče. Mikrořadič nevykonává žádnou funkci bez nějakého designu implementovaného v FPGA. Důležité je, že pro každý design je třeba naprogramovat mikrořadič na míru pro daný design, protože různě rozsáhlé designy mohou používat různý počet komunikačních signálů OK/FAIL. Z toho důvodu jsem tedy napsal jen jakousi knihovnu funkcí, ze kterých se firmware pro obsluhu designu v FPGA poskládá tak, aby vyhovoval danému designu.

7.1. Komunikace s SD pamětí

Jedna z nejdůležitějších funkcí je komunikace s SD pamětí. Ta komunikuje s MCU přes rozhraní SPI, které jsem vytvořil software-ově. Protokol SPI komunikuje po 8mi bitech, tedy po jednom bytu. Vytvořil jsem metodu **BYTE spi_send_read_byte(BYTE send_byte)**, která má jako parametr jeden byte dat, která mají být odeslána a vrací jeden byte dat, která byla přijata. Při odesílání dat po signálu MOSI se totiž ihned přijímají data po signálu MISO. Algoritmus metody je znázorněn na diagramu níže.



Obr. 29 Algoritmus metody send_read_byte

S takto navrženou metodou jsem schopen vyslat i přijmout jakákoliv data. Veškerá komunikace SPI protokolu je zarovnána na 8 bitů a skládá se z:

- Příkazů (commands)
- Odpovědí (responses)
- Bloků dat (data blocks)

SD paměť vždy odpovídá na příkazy speciálním response tokenem. V tabulce Tab.18. jsou uvedeny základní příkazy pro SD paměť.

Příkaz	**Argument**	**Odpověď**	**Popis**
CMD0	Žádný	R1	Reset zařízení
CMD1	Žádný	R1	Aktivuj inicializační proces
CMD9	Žádný	R1	Zašli svůj CSD registr
CMD10	Žádný	R1	Zašli svůj CID registr
CMD16	[31:0] délka bloku	R1	Nastaví délku komunikačního bloku pro další operace
CMD17	[31:0] adresa dat	R1	Načte data z adresy, počet bytů nastavuje CMD16
CMD24	[31:0] adresa dat	R1	Zapíše data na adresu, počet bytů nastavuje CMD16

Tab.18 Základní příkazy pro SD paměť

Odpověď R1 je speciální příznakový byte jehož každý bit něco znamená. Jeho vysvětlení je uvedeno v tabulce Tab.19.

Bit	**Chyba**	**Popis**
0	Idle stav	Karta je v idle stavu a provádí inicializaci
1	Reset mazání	Mazací sekvence nebyla dokončena
2	Chybný příkaz	Neexistující kód příkazu
3	Chyba CRC při komunikaci	Součet CRC posledního příkazu selhal
4	Chyby při mazání	Nastala chyba při příkazu mazání
5	Chyba adresy	Byla použita nezarovnaná adresa vzhledem k délce nastaveného bloku
6	Chyba parametru	Parametr příkazu je mimo meze
7	Nepoužitý	Vždy nulový

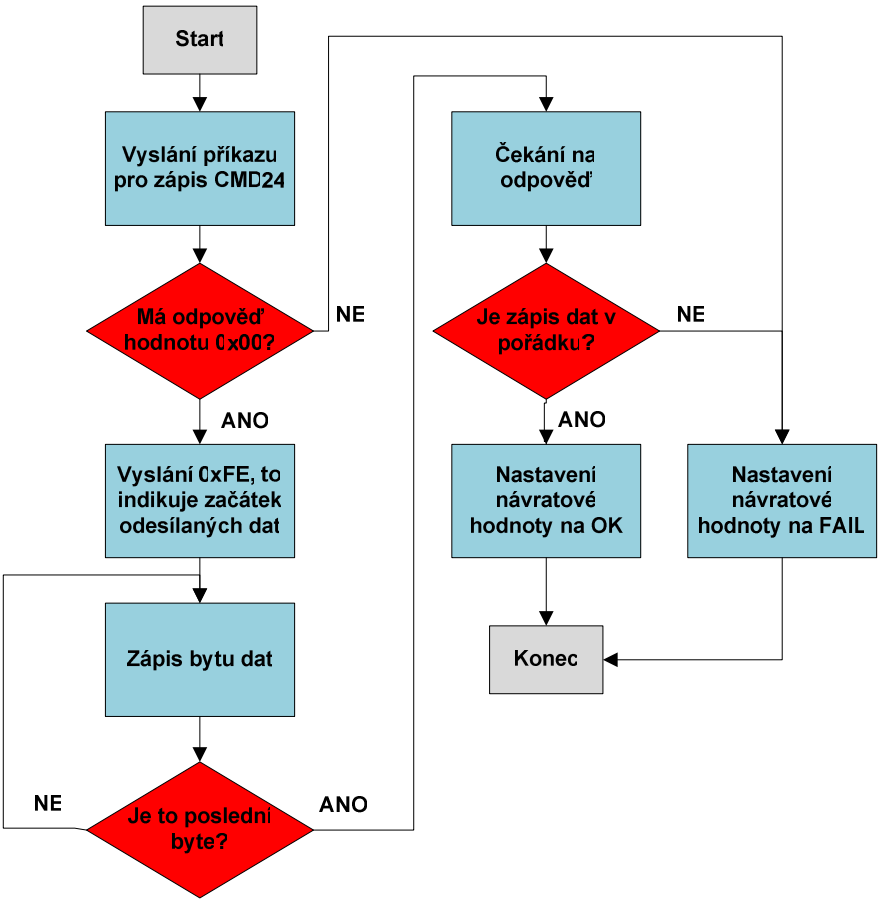
Tab.19 Odpověď R1

Po vložení SD paměti do konektoru je třeba paměť inicializovat. Je třeba donutit paměť, aby vstoupila do SPI módu. Postup je následující:

- Na výběrový signál CS přivést neaktivní signál
- Provést 80 hodinových taktů
- Na signál CS přivést aktivní úroveň a zaslat kartě příkaz CMD0
- Musíme čekat na odpověď příkazu CMD0 a ta musí mít hodnotu 0x01
- Zaslat příkaz CMD1 a čekat na odpověď 0x00, pokud není CMD1 se opakuje
- Po přijetí správné odpovědi je karta připravena

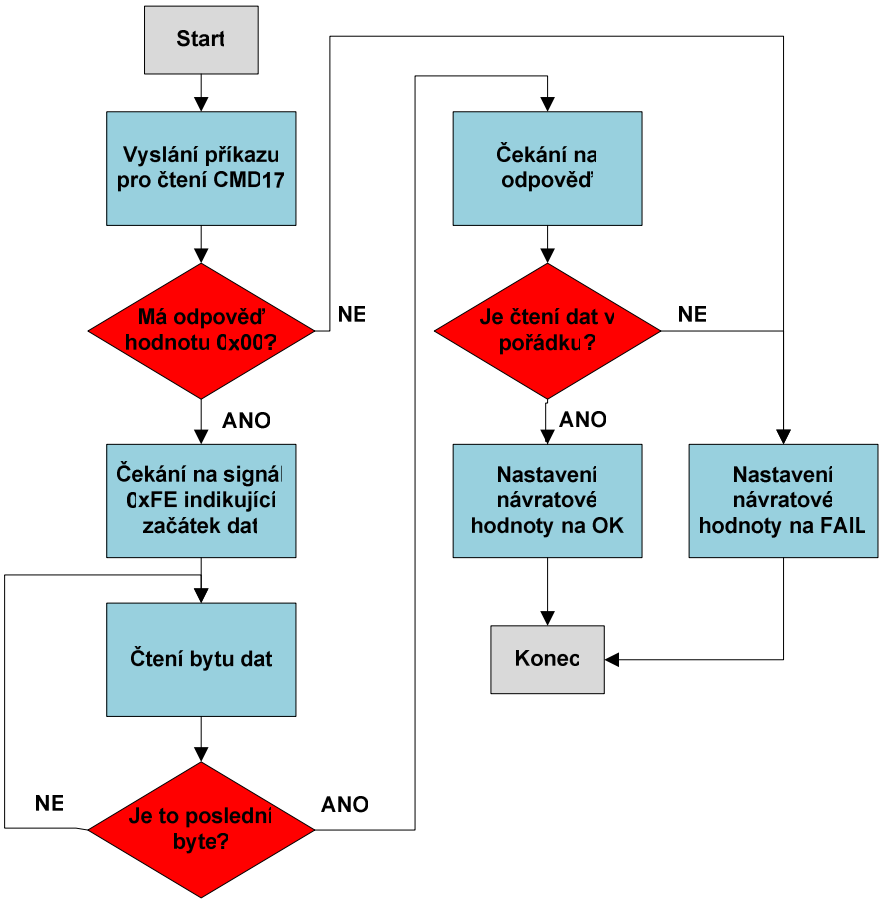
Dokud není inicializace karty dokončena, neměla by přenosová rychlost přesáhnout rychlost 400kHz. Potom už je možno přenosovou rychlost zvýšit.

Metoda s názvem **BYTE sd_write(unsigned long adresa_zapisu)** je určena k zápisu dat na paměť. Data zapisuje v blocích jejichž velikost je nastavena po inicializaci karty. Zde je to velikost 512 bytů. Adresa na kterou budou data uložena musí být při takto nastavené velikosti bloků zarovnána na 512 bytů. Návratovou hodnotou je výsledek zápisu, jestli zápis proběhl v pořádku či nikoli. Algoritmus této metody je na obrázku Obr.30.



Obr. 30 Algoritmus metody sd_write

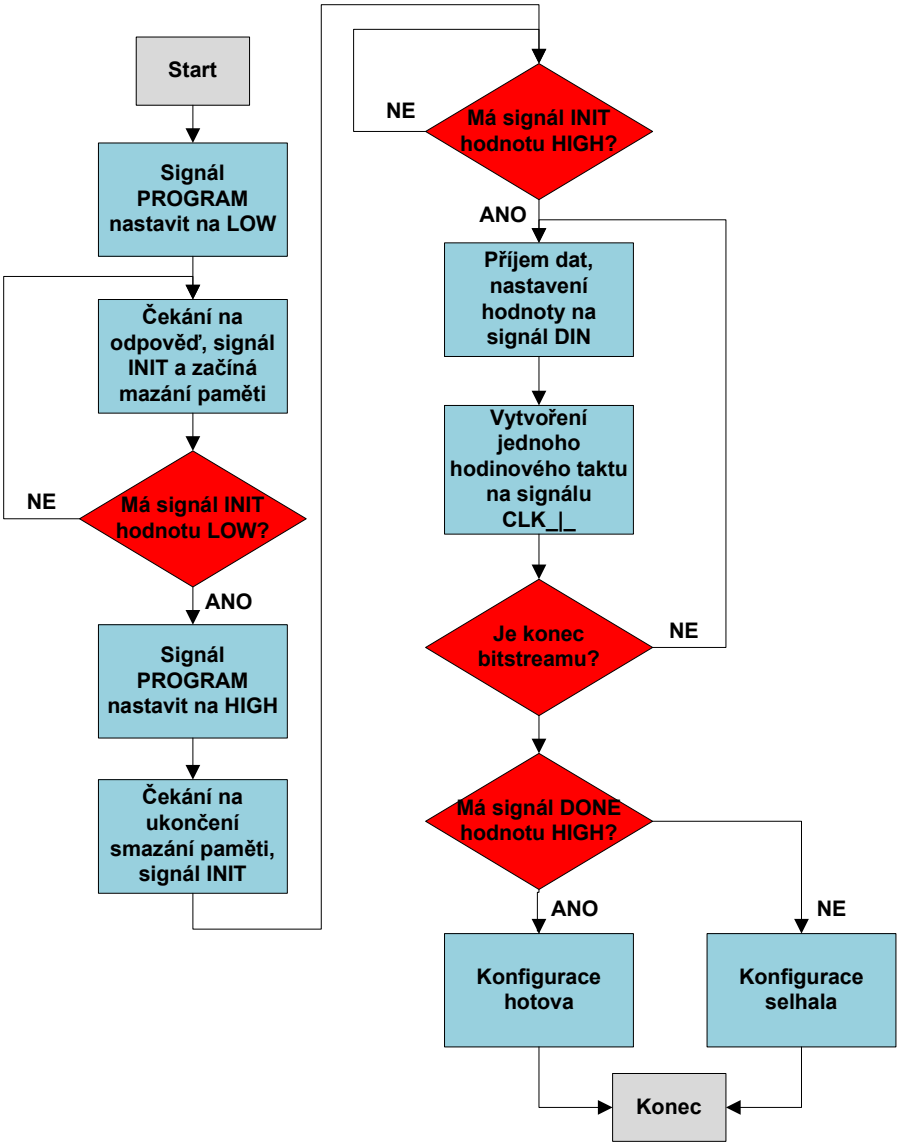
Další obdobnou metodou je **BYTE sd_read(unsigned long adresa_cteni)** která načte data velikosti nastaveného bloku, zde opět 512 bytů. Návratovou hodnotou této metody je opět její výsledek zda čtení proběhlo v pořádku či nikoliv.



Obr. 31 Algoritmus metody sd_read

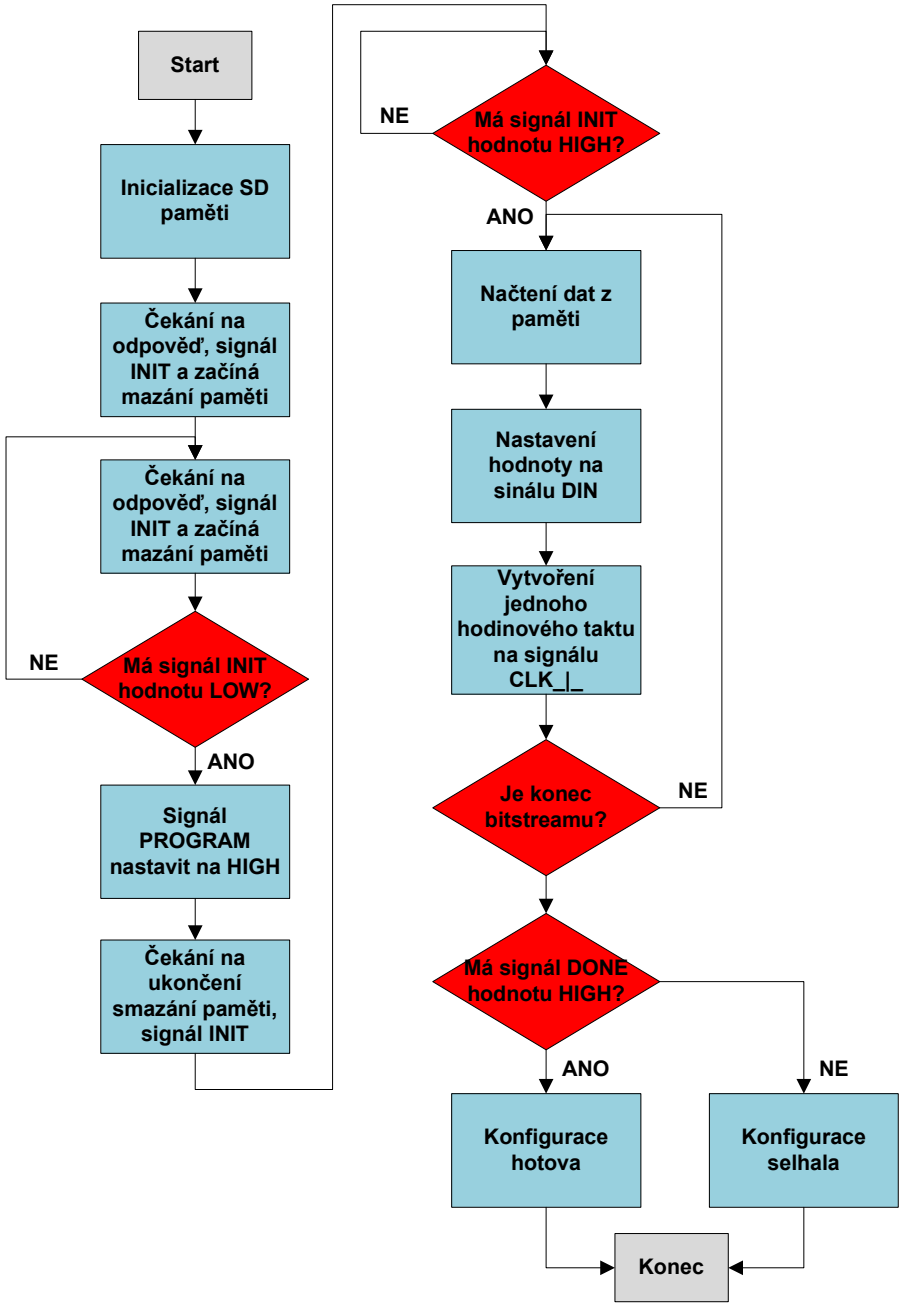
7.2. Konfigurace a rekonfigurace FPGA

Jedna z nejdůležitějších funkcí je konfigurace FPGA. Jde o metodu **void konfigurace_fpga (unsigned long size_bitstream)** jejímž parametrem je velikost bitstreamu. Tato metoda je napsána tak, že bitstream je přijímán z PC přes sériovou linku. Odesílání bitstreamu přes sériovou linku nesmí mít rychlost přenosu vyšší než 19200 baudů, protože příjem dat není přes přerušení a s vyšší přenosovou rychlostí by se nestíhala přijatá data obsloužit a docházelo by ke ztrátě dat. Tato metoda byla nejvíce použita při testování zařízení.



Obr. 32 Algoritmus metody konfigurace_fpga

Nejdůležitější metoda je konfigurace FPGA bitstreamem uloženým v SD paměti **BYTE read_bitstream_konf(unsigned long size_bitstream)**. Tato metoda kombinuje dvě výše uvedené metody **sd_read** a **konfigurace_fpga**. Detailnější popis konfigurace FPGA je uveden v kapitole analýza. Můj algoritmus je uveden na obrázku Obr.33.



Obr. 33 Algoritmus metody read_bitstream_konf

Konfigurace FPGA trvá při velikosti bitstreamu 180 kB přibližně 12 sekund, není to problém software-ový. Tato doba není příliš optimální. Návrhy na zlepšení tohoto problému a podrobnější analýza je uvedena v kapitole zhodnocení a návrhy na zlepšení.

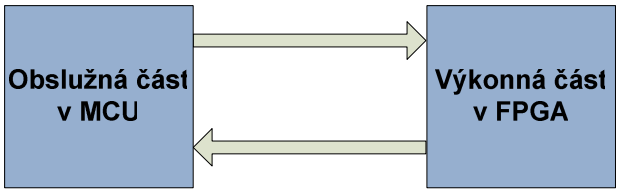
7.3. Uložení bitstreamu na SD paměť

Bitstream je potřeba mít na SD paměti uložen bez nějakého file systému. Používám tedy SD paměť v podstatě jenom jako flash paměť. Bitstream je třeba mít uložen od adresy 0 nebo přepsat metodu **BYTE read_bitstream_konf(unsigned long size_bitstream)**, která předpokládá, že bitstream začíná na adrese právě 0.

8. Aplikace

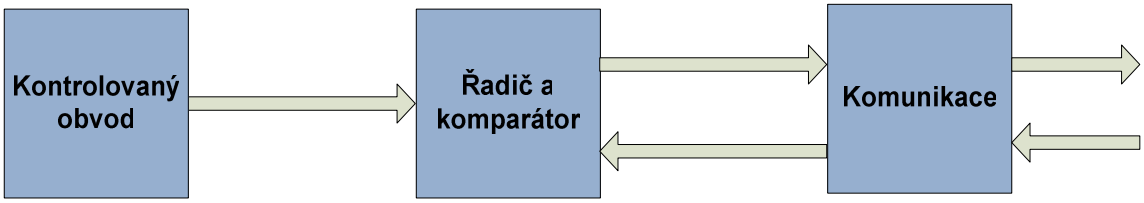
8.1. Obecný popis

Protože je potřeba vždy napsat i firmware pro MCU, lze obecně aplikaci rozdělit na dvě části. Na část v obvodu FPGA a na část v MCU. V obvodu FPGA je nahrána výkonná část aplikace a v MCU je část, která tu v FPGA obsluhuje.



Obr. 34 Obecné rozdělení aplikace

Část v FPGA se dá dále rozdělit na další tři části. Kontrolovaný obvod (logika nebo automat), řadič a komparátor, komunikační část mezi dalšími zařízeními.

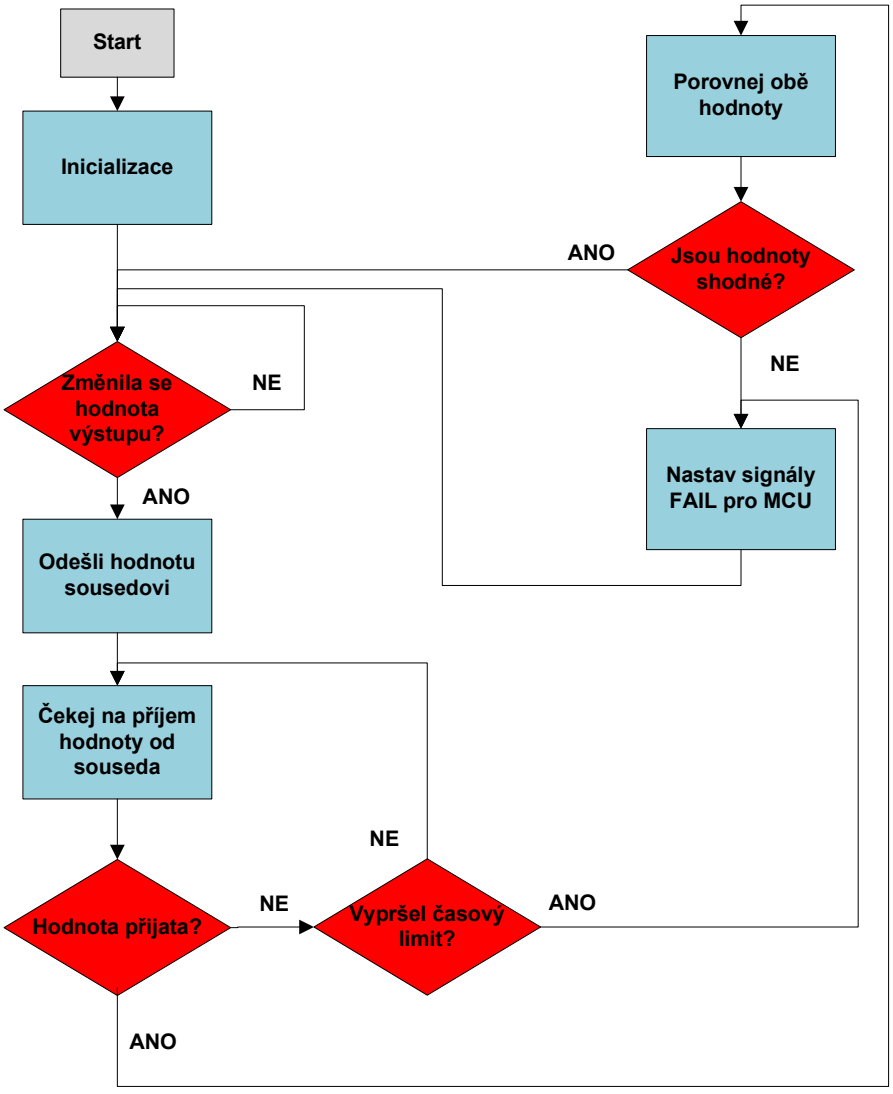


Obr. 35 Obecné rozdělení části v FPGA

Kontrolovaný obvod poskytuje své výstupy vnějšímu světu i komparátoru s řadičem, který kontroluje hodnoty těchto výstupů s hodnotami výstupů druhého zařízení. Řadič obsluhuje komparátor a komunikaci a v závislosti na nich nastavuje hodnoty signálů vedoucích do MCU (obslužná část). Komunikační část zajišťuje přenos signálů do a z komparátoru s druhým zařízením.

8.2. Demonstrační aplikace

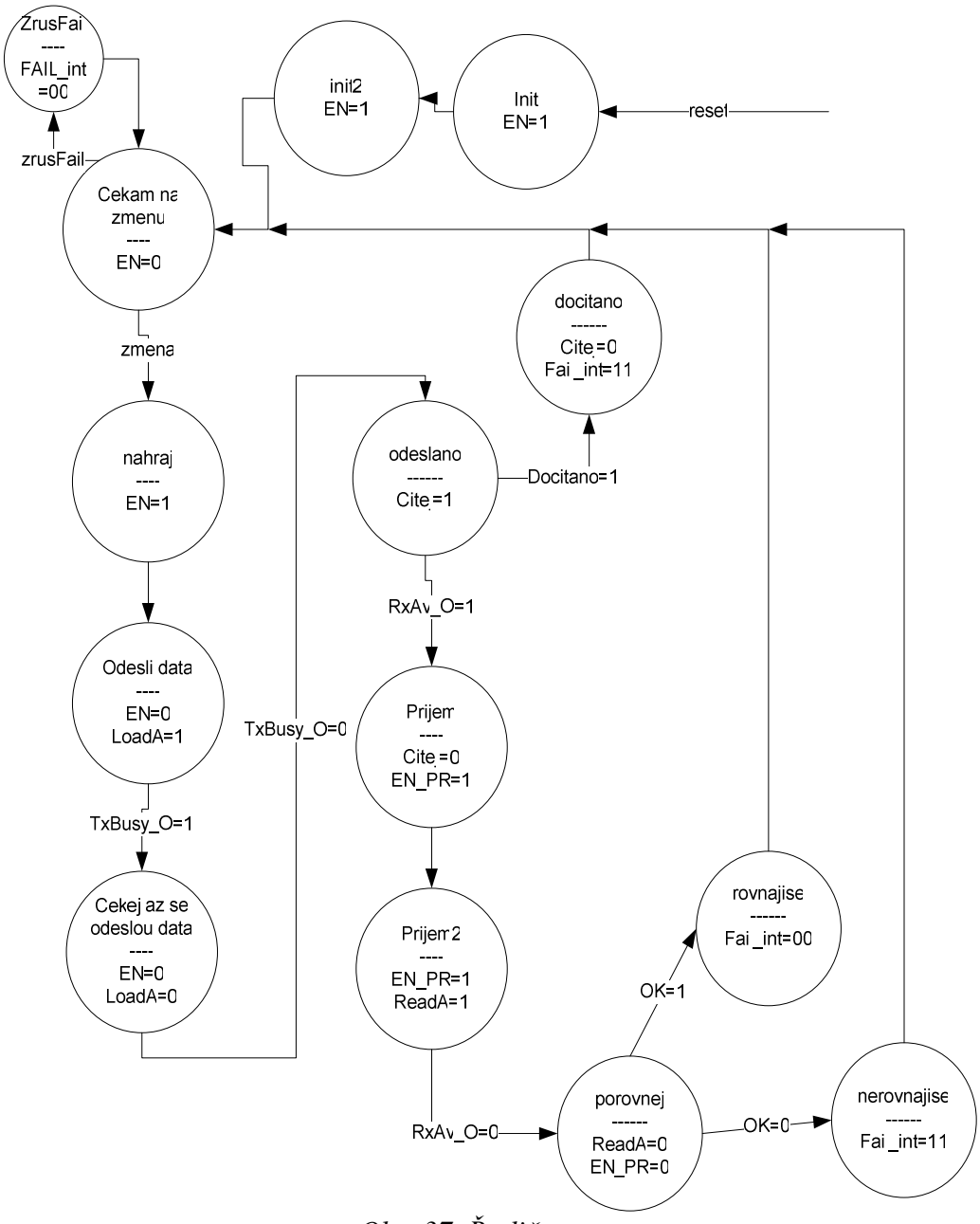
Mojí demonstrační aplikaci jsem navrhoval přesně podle blokového složení uvedeného v podkapitole výše. Nejprve jsem si zvolil kontrolovaný obvod, pro jednoduchost jsem si zvolil kombinační obvod – jedno hradlo typu XOR. Pro rozsáhlejší kombinační obvod by byl princip úplně stejný, takže pro demonstraci mi jedno hradlo stačí. Dále jsem si jako komunikační rozhraní zvolil sériovou linku, kód ve VHDL je stažený z www.opencores.org . Řadič je napsaný v jazyce VHDL a automat je realizovaný metodou tří procesů. Zjednodušený diagram pro řadič je na obrázku Obr.36.



Obr. 36 Zjednodušený diagram řadiče

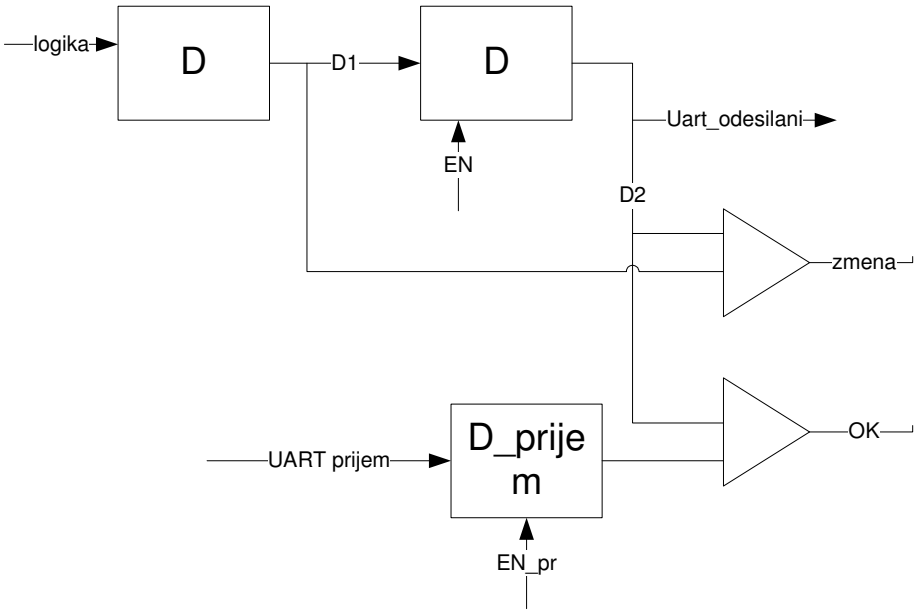
Po spuštění aplikace se provede inicializace signálů automatu, přečte se hodnota výstupu testovaného obvodu a čeká se na její změnu. Nastane-li změna hodnoty

výstupu odešle se tato hodnota sousedovi a čeká se na hodnotu přijatou od druhého souseda. Je-li hodnota přijata, porovná se s druhou a jsou-li shodné celý cyklus se opakuje. Nastane-li porucha někde uvnitř kontrolovaného obvodu, žádná hodnota výstupu od souseda přijata není, protože on žádnou změnu výstupu nezaznamenal. Umístil jsem ještě k příjmu dat čekací smyčku, kdyby byla data o něco zpožděna. Po vypršení tohoto limitu, jsou nastaveny signály FAIL pro MCU, poté se opět celý cyklus opakuje. Signály FAIL jsou nastaveny i v případě, že se přijaté hodnoty nerovnají. Nezjednodušený automat i s vnitřními signály je uveden na obrázku Obr.37. Jsou v něm zahrnuty i stavy sériové linky a proto jsem radši pro lepší vysvětlení uvedl automat na obrázku Obr.36.



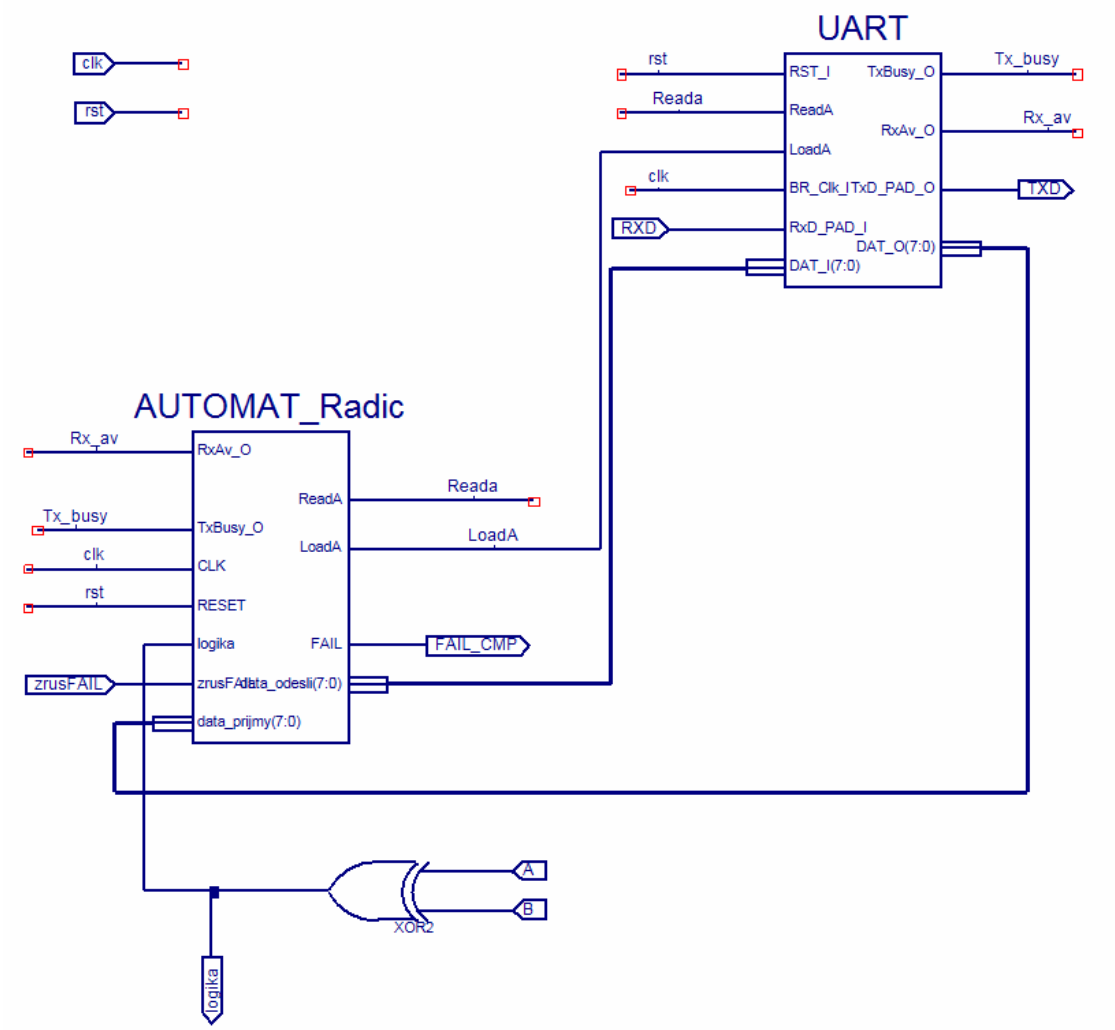
Obr. 37 Řadič

Systém porovnání výstupních hodnot je znázorněn na obrázku Obr.38. Hodnota výstupu je přivedena na klopný obvod D a s každou hodinovou hranou se do něho hodnota zapíše. Následuje druhý klopný obvod D do kterého se hodnota zapíše je-li nastaven signál enable. Komparátor, který odhalí změnu výstupu porovnává právě hodnoty z těchto dvou klopných obvodů a nastavuje signál změna. Druhý komparátor porovnává přijatou hodnotu od souseda a aktuální hodnotu výstupu.



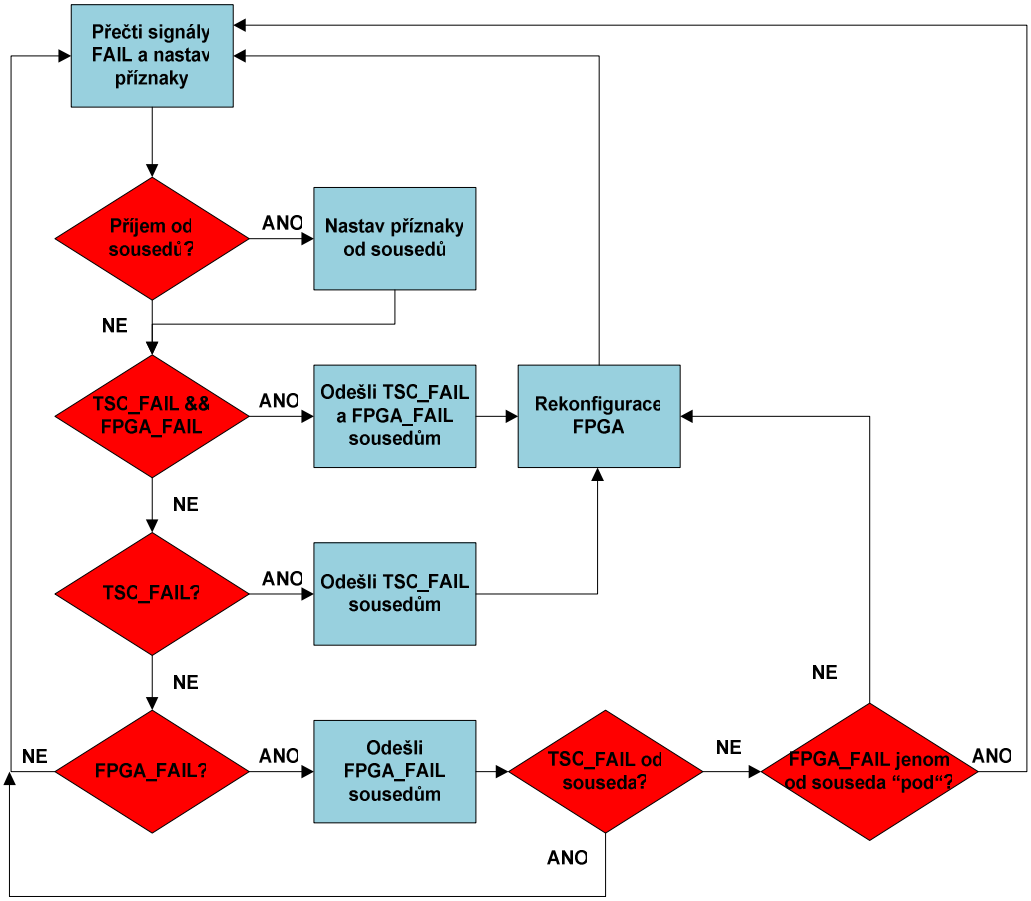
Obr. 38 Systém porovnání výstupních hodnot

Na obrázku Obr.39 je znázorněn design blokově, tak jak vypadá v systému ISE. Je vidět, že je rozdělen na tři části, tak jak jsem popsal v podkapitole výše obecná aplikace. Je tam část logika, reprezentována hradlem XOR. Dále řadič a část komunikace zajišťuje sérová linka.



Obr. 39 Blokové schéma aplikace v systému ISE

Tím by tedy byla vysvětlena část v obvodu FPGA, je třeba objasnit obslužnou část v MCU. Je to jednoduchý automat reagující na vstupní signály od řadiče v FPGA. V závislosti na těchto signálech jsou mikrořadičem prováděny různé akce. Chování tohoto řadiče vychází z kapitoly Analýza propojení. Automat je diagramem znázorněn na obrázku Obr. 40.



Obr. 40 Algoritmus části v MCU

Hlavní cyklus algoritmu začíná tím, že přečte signály od obvodu FPGA, nastaví globální proměnné pro další zpracování. Potom jsou-li přijata nějaká data od sousedů, tak se nastaví další proměnné. Začíná kontrola těchto příznaků. Nejprve je třeba zjistit, jestli jsou nastaveny příznaky TSC_FAIL a FPGA_FAIL. Je-li to tak, pošlou se tyto příznaky sousedům a následuje rekonfigurace. Nejsou-li nastaveny oba tyto příznaky, jde se dál na kontrolu jestli nejsou nastaveny jednotlivě. Je-li nastaven příznak TSC_FAIL odešle se tento příznak sousedům a provede se rekonfigurace. Jestli tento signál nastaven není, kontrola pokračuje. Je-li nastaven příznak FPGA_FAIL odešle se sousedům. Nastala jediná chvíle v této aplikaci, kdy nás zajímá i hodnota souseda, protože v předchozích případech vždy byl nastaven příznak TSC_FAIL. Je-li ale nastaven příznak FPGA_FAIL je jasné, že další zařízení hlásí stejnou chybu, protože chybné výstupy jsou přeposílány do dalšího zařízení, které je porovná se svými. Takže v této chvíli si zjistím, jestli nějaký můj soused nemá nastaven příznak TSC_FAIL. Jestli ano, vím, že chyba není u mě, ale u mého souseda, vracím se na začátek algoritmu bez rekonfigurace. Není-li TSC_FAIL od souseda nastaven kontroluji příznaky FPGA_FAIL od sousedů a nerekonfiguruji sám sebe, pouze tehdy, je-li nastaven příznak jenom od mého souseda "pod". Je to takto

z důvodu univerzálnosti tohoto algoritmu pro dvě a více zařízení. Je-li totiž systém složen pouze ze dvou zařízení, signály FPGA_FAIL budou nastaveny u obou zařízení a musí se rekonfigurovat obě. Systém složený ze tří zařízení je na tom lépe, díky třetímu zařízení lze rozpoznat chybné zařízení jednoznačně. Je-li nastaven příznak od souseda "nad", znamená to, že chyba je u mě a rekonfiguruji se. Nic se neděje jen tehdy, je-li nastaven příznak FPGA_FAIL pouze od souseda "pod". Stejný postup je i pro spojení více zařízení, proto je algoritmus obsluhy univerzální. Časová analýza

8.3. Závěr

Z popsaných algoritmů pro MCU a FPGA je vidět, že algoritmy běží ve stavu bez chyb nezávisle, nejsou nijak synchronizovány. Automat v FPGA začne postupovat do dalších stavů až když se změní hodnota hlídaných výstupů. V případě změny výstupů jsou hodnoty přeposílány a zde je nutná synchronizace obvodů FPGA. V kombinační logice je tato synchronizace jednodušší, než kdyby hlídaný obvod byl nějaký složitý automat. V mém případě mi ovšem stačí zmíněná čekací smyčka. Je-li změna výstupů způsobena vstupní hodnotou, obě FPGA reagují okamžitě. Rychlost bezchybných změn výstupů, na které jsem automatem v FPGA schopen reagovat a přenášet je sousedům je v podstatě omezena rychlostí sériové linky. Upozorňuji, že jako komunikační linka nemusí být použita sériová linka, může být signál přiveden k sousedům přímo a odezva bude okamžitá → rychlost přenesení změn nebude omezena sériovou linkou. Komunikačních rozhraních mezi FPGA je na výběr velké množství a návrhář má k použití připraveno 16 signálů. Sériová linka v mém případě tvoří jakousi bariéru pro synchronizaci komparačních výstupů obvodů FPGA.

V případě, že nastane chybná změna výstupu, čeká se až vyprší čekací smyčka a poté se nastaví signály FAIL pro FPGA. V době čekání na vypršení čekací smyčky jsou další případné změny výstupů ignorovány. Případná rekonfigurace obvodu FPGA trvá 12 sekund, což je dlouhá doba. Případné řešení tohoto problému je popsáno v kapitole návrhy na zlepšení. Časová analýza celé opravy designu v FPGA (rekonfigurace) v podstatě stanový celkový čas. Protože jak automat v MCU tak řadič v FPGA do této doby výrazně nepromluví, jejich reakce je v podstatě okamžitá. Jejich časová analýza by do této doby výrazně nepromluvila, ani kdyby rekonfigurace trvala pod 1 sekundu, což by byl optimální čas rekonfigurace. Tzn. doba trvání celého algoritmu v případě chyby v nejhorším případě = doba rekonfigurace obvodu FPGA.

9. Testování

9.1. Proces oživení zařízení a testování

Testování zařízení nebo jeho částí jsem prováděl už při oživování. Oživení zařízení je velmi důležitá část celé realizace zařízení, je-li oživení prováděno po fázích s určitou systematikou, pomůže to snadněji odhalit případné chyby v návrhu schéma nebo případné výrobní vady na desce plošného spoje. Tento systém oživení se ukázal jako velmi účinný. Požadavky jsem si pro proces oživení zařízení stanovil takto, jejich pořadí odpovídá pořadí, jak bylo celé zařízení oživováno.

- Req1. Proměření oddělení všech napájecích větví od protipólu GND
- Req2. Proměření všech napájecích větví mezi sebou
- Req3. Osazení zdroje napětí 5V, změření výstupního napětí
- Req4. Osazení zdroje napětí 3,3V, změření výstupního napětí
- Req5. Osazení zdroje napětí 1,8V, změření výstupního napětí
- Req6. Osazení MCU a nezbytných součástek k jeho správné funkci
- Req7. Testovací naprogramování MCU
- Req8. Osazení krystalu k MCU
- Req9. Testovací naprogramování MCU a nastavení fuse bitů pro externí krystal
- Req10. Osazení testovacích LED diod k MCU a jejich rozblikání, ověřit pracovní frekvenci MCU
- Req11. Osazení převodníku FTD232RL a jeho komponent, otestovat vysláním a příjmem dat přes hyperterminál
- Req12. Osazení čtečky pro SD paměť, proměření napětí po úpravě na hodnotu 3,3V z 5V
- Req13. Testovací zapsání a čtení dat z SD paměti
- Req14. Osazení součástek pro úpravu napětí pro signály konfigurace FPGA
- Req15. Osazení obvodu FPGA, testovací zapnutí zařízení
- Req15. Osazení krystalového oscilátoru k FPGA, zkušební start
- Req16. Konfigurace FPGA
- Req17. Osazení vstupních a výstupních konektorů a zbylých součástek, zkušební start

Výsledky procesu oživení a ověření funkce jsou uvedeny v tabulce Tab.20.

Požadavek	Výsledek	Problém a řešení
Req1	OK	
Req2	OK	
Req3	OK	
Req4	OK	
Req5	FAIL,OK	Po osazení zdroje bylo naměřeno výstupní napětí 8,4V oproti požadovaným 1,8V, výstupní napětí se u zdroje tohoto typu nastavuje rezistory a právě ty byly zvoleny špatně --> špatný výpočet hodnot, přehlédnuta jedna vlastnost v datasheetu, po změně rezistorů vše OK
Req6	OK	
Req7	FAIL,OK	Zkušební naprogramování se nepodařilo, bylo zjištěno, že byla špatně připájena jedna programovací nožička MCU, znovu připájena a pak vše OK
Req8	OK	
Req9	OK	
Req10	OK	
Req11	OK	
Req12	FAIL,OK	Selhala úprava napětí, výstupní napětí mělo hodnotu 4,3V místo napětí 3,3V, musel se použít jiný typ zenerových diod, po záměně vše OK
Req13	OK	
Req14	FAIL,OK	Stejný problém jako v Req12. po záměně vše OK
Req15	OK	
Req16	OK	
Req17	OK	

Tab.20 Výsledky testování a řešení problémů v procesu oživení zařízení

9.2. Testování hotového zařízení

Ověření správné funkce již hotového zařízení probíhalo ve sledu podle uvedených požadavků na funkčnost. V této fázi testování bylo testováno pouze jedno zařízení, s lokálně umístěným propojovacím kabelem.

- Req1. Naprogramování mikrořadiče
- Req2. Příjem dat přes USART (převodník FT232RL)
- Req3. Odeslání dat přes USART (převodník FT232RL)
- Req4. Inicializace SD paměti
- Req5. Zápis bytu dat na SD paměť
- Req6. Čtení bytu dat z SD paměti
- Req7. Nastavení velikosti bloku dat SD paměti
- Req8. Zápis bloku dat na SD paměť
- Req9. Čtení bloku dat z SD paměti
- Req10. Konfigurace FPGA
- Req11. Ověření pracovní frekvence FPGA
- Req12. Ověření všech výstupních a vstupních signálů obvodu FPGA
- Req13. Příjem dat od souseda vpravo
- Req14. Odeslání dat sousedovi vpravo
- Req15. Příjem dat od souseda vlevo
- Req16. Odeslání dat sousedovi vlevo
- Req17. Rekonfigurace FPGA bitstreamem přijatým po sériové lince
- Req18. Rekonfigurace FPGA bitstreamem z SD paměti

Výsledky testování již hotového zařízení jsou uvedeny v tabulce Tab.21.

Požadavek	Výsledek	Problém a řešení
Req1	OK	
Req2	OK	
Req3	OK	
Req4	OK	
Req5	OK	
Req6	OK	
Req7	OK	
Req8	OK	
Req9	OK	
Req10	OK	
Req11	OK	
Req12	OK	
Req13	OK	
Req14	OK	
Req15	OK	
Req16	OK	
Req17	OK	
Req18	FAIL,OK	Konfigurace bitstreamem o velikosti 180kB trvá přibližně 12 sekund, což je dost dlouho, přenosová rychlost odpovídá 15kB/s, je to způsobeno obvodem pro úpravu napětí z 5V na 3,3V

Tab.21 Výsledky testování hotového zařízení

9.3. Testování systému dvou zařízení

Testování systému dvou zařízení, ve kterých byla nahrána moje demonstrační aplikace, probíhalo podle níže uvedených požadavků. Obě zařízení komunikovaly s PC přes hyperterminál, takže bylo možno sledovat, v jaké fázi se každé zařízení nacházelo.

- Req1. Bezchybná funkce systému bez změn výstupů
- Req2. Bezchybná funkce systému s měnícími se výstupy
- Req3. Zavedena chyba výstupu prvního zařízení CP_FAIL
- Req4. Zavedena chyba výstupu druhého zařízení CP_FAIL

- Req5. Zavedena vnitřní chyba prvního zařízení jako simulace TSC_FAIL
- Req6. Zavedena vnitřní chyba druhého zařízení jako simulace TSC_FAIL
- Req7. Obě chyby zavedeny do prvního zařízení TSC_FAIL i CP_FAIL
- Req8. Obě chyby zavedeny do druhého zařízení TSC_FAIL i CP_FAIL

Výsledky testování systému dvou zařízení jsou uvedeny v tabulce Tab.22.

Požadavek	Výsledek	Problém a řešení
Req1	OK	
Req2	OK	
Req3	OK	
Req4	OK	
Req5	OK	
Req6	OK	
Req7	OK	
Req8	OK	

Tab.22 Výsledky testování systému dvou zařízení

9.4. Závěr

Při oživování zařízení se pomocí testování odhalily drobné chyby, které se po odhalení vyřešily a funkce testovaného bloku byla poté v pořádku. Při takto rozfázovaném procesu oživení zařízení bylo odhalení chyb velice snadné a rychlé. Všechny testy výše uvedených požadavků proběhly v pořádku, u všech je výsledek OK. Jediná poznámka je uvedena u požadavku 18, jde problém s pomalou rekonfigurací obvodu FPGA. Návrh na řešení tohoto problému je uvedeno v kapitole zhodnocení a návrhy na zlepšení.

10. Zhodnocení a návrhy na zlepšení

Cílem této diplomové práce bylo navrhnout a zrealizovat vysoce spolehlivý systém založený na FPGA obvodech. Navrhnout univerzální zařízení tak, aby bylo možné sestavit systém z libovolného počtu těchto zařízení odolný proti poruchám. Pro tento systém vytvořit jednoduchou demonstrační aplikaci. Těchto cílů bylo dosaženo navržením univerzálního zařízení s obvodem FPGA a mikrořadičem, který řídí obslužnou část obvodu FPGA a řeší případnou rekonfiguraci a komunikaci z dalšími zařízeními. Univerzálnost zařízení byla demonstrována v kapitole 3. Návrh systému.

Největší problém je příliš dlouhá doba rekonfigurace. Pohybuje se okolo 12 sekund. Je to způsobené obvodem pro úpravu napětí z 5v na 3,3V. Případná řešení připadající v úvahu:

1) Použít jiný typ zenerových diod
2) Použít jiný typ mikrořadiče pracující s napětím 3,3V, takže by byla úprava napětí zbytečná
3) Snížit pracovní frekvenci mikrořadiče na 8MHz, tak aby mohl pracovat s napětím 3,3V, takže úprava napětí by byla zbytečná
4) Použít místo obvodu se zenerovou diodou obyčejného rezistorového děliče

Tento problém byl nakonec vyřešen použitím jednoho z výše uvedených návrhů, bez nějakého vážného zásahu do stávajícího zařízení. Doba rekonfigurace byla stažena pod 4 sekundy.

11. Závěr

Výsledkem této diplomové práce je návrh a realizace univerzálního zařízení založeného na obvodu FPGA. Zařízení bylo navrženo tak, aby bylo možné propojit libovolný počet zařízení. Z libovolného počtu těchto zařízení je tedy možné sestavit systém, který je bezpečný proti poruchám. Je toho dosaženo použitím designu TSC v FPGA a znásobením zařízení, které si navzájem kontrolují výstupní hodnoty. Každé zařízení je ještě v případě nutnosti možno rekonfigurovat přímo v systému. Bylo tedy nutné navrhnout schéma zařízení, navrhnout desku plošného spoje a poté oživení zařízení. Jako zdroj bitstreamu pro FPGA je použita SD paměť, takže lze pomocí tohoto média bitstream snadno (i za chodu zařízení) zaměnit. Dále byla v rámci této práce vytvořena jednoduchá demonstrační aplikace pro zabezpečení jednoduchého kombinačního obvodu. Na této aplikaci bylo provedeno testování.

Použitá literatura a zdroje

Uvedené soubory jsou na přiloženém CD.

[1] Ing. Pavla Kubalíka., Ph.D. *Design of Self Checking Circuits Based on FPGAs,* 2006

[2] Atmel. *Datasheet ATMega2560*

[3] SanDisk. *Datasheet SD Paměť*
SD Association *Secure Digital Input/Output (SDIO) CardSpecification* 2000, 2001

[4] *Datasheet LM2575T-05*
Datasheet LM2575T-05ADJ
National Semiconductor Corporation *Datasheet LM317HVT* 2000

[5] Xilinx. *Datasheet Rodina Spartan II*

[6] *http://www.digiplus.cz/dokumenty/informace-15/*
V. Záhlava, *skriptum Metodika návrhu plošných spojů*

[7] Infineon Technologies AG.. *Datasheet SLA F0016 H CA*

Seznam příloh

1. Rejstřík obrázků
2. Rejstřík tabulek
3. Obsah přiloženého CD

Rejstřík obrázků

Rejstřík tabulek

Obsah přiloženého CD

1) Text diplomové práce ve formátu *.doc
2) Text diplomové práce ve formátu *.pdf
3) Použitá literatura
4) Projekt z návrhového prostředí
5) Projekt z vývojového programovacího prostředí
6) Projekt demonstrační aplikace v systému ISE