ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA ELEKTROENERGETIKY A EKOLOGIE

DIPLOMOVÁ PRÁCE

Václav Kotora **2018/2019**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA ELEKTROENERGETIKY A EKOLOGIE

DIPLOMOVÁ PRÁCE

Datová komunikace mezi elektronickými zařízeními trakčních vozidel

Václav Kotora **2018/2019**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2018/2019**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Václav KOTORA**
Osobní číslo:	**E16N0012K**
Studijní program:	**N2644 Aplikovaná elektrotechnika**
Studijní obor:	**Aplikovaná elektrotechnika**
Název tématu:	**Datová komunikace mezi elektronickými zařízeními trakčních vozidel**
Zadávající katedra:	**Katedra elektroenergetiky a ekologie**

Zásady pro vypracování:

Věnujte se problematice dvoubodové datové komunikace mezi elektronickými zařízeními trakčních vozidel s možností dálkového programování a diagnostikou.

1. Popište možnosti komunikace mezi zařízeními trakčních vozidel. Pro novou generaci elektronických jednotek v trakčních vozidlech navrhněte koncept zpracování dat s možností přeprogramování SLAVE ze strany MASTER.
2. Navrhněte architekturu systému a nadefinujte činnosti jednotlivých bloků a v závislosti na této architektuře systému zvolte vhodné softwarové nástroje pro realizaci.
3. Vytvořte jednoduchou realizaci dvoubodové datové komunikace splňující zadání včetně kontroly funkčnosti.
4. Zhodnoťte výhody a nevýhody navrhovaného řešení vzhledem k jeho využitelnosti v praxi.

Rozsah grafických prací: **podle doporučení vedoucího**

Rozsah kvalifikační práce: **40 - 60 stran**

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

Seznam odborné literatury:

1. **Student si vhodnou literaturu vyhledá v dostupných pramenech podle doporučení vedoucího práce.**

Vedoucí diplomové práce: **Doc. Ing. Karel Noháč, Ph.D.**
Katedra elektroenergetiky a ekologie

Datum zadání diplomové práce: **5. října 2018**

Termín odevzdání diplomové práce: **30. května 2019**

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

L.S.

Doc. Ing. Karel Noháč, Ph.D.
vedoucí katedry

V Plzni dne 10. října 2018

Abstrakt

První část práce se zabývá teoretickým popisem datových komunikací. Jsou zde popsána komunikační rozhraní, jako jsou UART, RS-232, RS-422, RS-485, SPI, I2C a sběrnice CAN. Druhá část práce je zaměřena na problematiku vzdáleného přehrání konfigurace hradlového pole. Cílem práce bylo navrhnout řešení vzdáleného přehrání konfigurace hradlového pole, které bude možné aplikovat v praxi v problematice trakčních vozidel. Jako zařízení MASTER byl zvolen osobní počítač, který podporuje rozhraní UART a obsahuje vhodný komunikační terminál. Komunikační terminál umožňuje odesílání řídících znaků a nového konfiguračního souboru pro hradlové pole. Zařízení SLAVE je realizováno vývojovým kitem s hradlovým polem řady MAX10 od výrobce Intel. Návrh celého systému probíhá ve vývojovém prostředí Quartus. Přínosem navrhovaného řešení je možnost přehrání konfigurace zařízení, které obsahuje hradlové pole, bez potřeby jakékoli mechanické manipulace se zařízením (zařízení se může nacházet na obtížně přístupném místě). Tím lze značně snížit časové i finanční náklady na servis daného zařízení. Pro účely této práce bylo zvoleno komunikační rozhraní UART, ale v praxi může být použito jakékoli rozhraní, které je popsáno v teoretické části této práce. V práci je detailně popsán postup při návrhu, který má za účel usnadnit orientaci v celém projektu.

Klíčová slova

Komunikační rozhraní, UART, RS-232, RS-485, RS-422, Ethernet, CAN, SPI, I2C, FPGA, hradlové pole, Quartus, NIOS II, Platform Designer, FTDI, JTAG

Abstract

First part of this thesis is focused on describing data communications. UART, RS-232, RS-422, RS-485, SPI, I2C interfaces and CAN bus are described. Second part of this thesis is focused on remote update of programmable gate array. Aim of this thesis was suggested a solution remote update of programmable gate array which will be can applicated in traction vehicles. MASTER device is personal computer with UART interface and communication terminal. Communication terminal allows sending driving characters and new configuration file for programmable gate array. SLAVE device is realized by development kit with programmable gate array MAX10 by manufacturer Intel. Design of this system is designed in development environment Quartus. Benefit of this suggest solution is support remote update device consist of programmable gate array without mechanical manipulation with this device (device can be placed in a hard-to-reach location). Suggested solution reduce time and financial expenses. For the purpose of this thesis was used UART interface but in real application can be used another interface described in theoretical part of this thesis. Design of this system is detail described.

Key words

Communication interface, UART, RS-232, RS-485, RS-422, Ethernet, CAN, SPI, I2C, FPGA, gate array, Quartus, NIOS II, Platform Designer, FTDI, JTAG

Prohlášení:

Prohlašuji, že jsem tuto diplomovou práci vykonal samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této diplomové práce je legální.

…….

podpis

V Chocenickém Újezdě dne 21.5.2019 Václav Kotora

Poděkování

Tímto bych rád poděkoval vedoucímu diplomové práce Doc. Ing. Karlu Noháčovi, Ph.D. za cenné rady a připomínky při zpracovávání práce. Dále bych rád poděkoval Doc. Ing. Martinu Poupovi, Ph.D. za pomoc při oživování systému a v neposlední řadě patří poděkování společnosti Škoda Electric a.s. za zadání práce.

Obsah

Seznam symbolů a zkratek

ISO/OSI	International Organization for Standartion
UART	Universal Asynchronous Reciever and Transmitter
USART	Universal Synchronous and Asynchronous Reciever and Transmitter
CAN	Controller Area Network
SPI	Serial Peripherial Interface
FPGA	Field Programmable Gate Array
RX(D)	Receive Data
TX(D)	Transmit Data
DTE	Data Terminal Equipment
DCE	Data Communication Equipment
PC	Personal Computer
D-SUB	D-subminiature
LCC	Logical Link Control
MAC	Media Access Control
IEEE	Institute of Electrical and Electronics Engineers
CSMA/CD	Carrier Sense Multiple Acces with Collision Detection
PRE	Preambule
SFD	Start Frame Delimiter
DA	Destination Address
SA	Source Address
FCS	Frame Check Sequence
PCS	Physical Conding Sublayer
PMA	Physical Medium Attachment
MII	Media Independent Interface
MDI	Medium Dependent Interface

SOF	Start Of Frame
AR	Arbitration
RTR	Remote Transmit Request
IDE	Identification
CRC	Cyclic Redundancy Check
ACK	Acknowledgement
EOF	End Of Frame
ADR	Adress
R/W	Read/Write
QSPI	Quad Serial Peripheral Interface
VHDL	VHSIC Hardware Description Language
HDL	Hardware Description Language
CFM	Configuration Flash Memory
UFM	User Flash Memory
JTAG	Join Test Action Group
USB	Universal Serial Bus
PLL	Phase Locked Loop
CLK	Clock
MISO	Master Input Slave Output
MOSI	Master Output Slave Input
SCK	SPI Clock
CS	Chip Select
COM	Communication Port
OCR	On Chip RAM
RAM	Random Access Memory
OCF	On Chip Flash

Úvod

Práce je rozdělena na dvě části. První část je zaměřena na teorii datových komunikací se zřetelem na použití v trakčních vozidlech. Je zde popsána základní teorie přenosu dat mezi zařízeními, která je dále rozšířena o popis referenčního modelu ISO/OSI. V teoretické části se nachází popis standardů, rozhraní a sběrnic, které jsou v dnešní době běžně používány pro komunikaci mezi elektronickými zařízeními trakčních vozidel. Jedná se o rozhraní, případně sběrnice, které slouží k přenosu dat na vzdálenosti desítek až stovek metrů. Konkrétně jsou v práci popsána rozhraní typu RS-232, RS-422, RS-485, Ethernet a je zde zmíněna i sběrnice CAN. Následuje popis rozhraní, která slouží k přenosu dat na velmi krátké vzdálenosti, řádově desítek centimetrů. Konkrétně se jedná o rozhraní typu SPI a I2C, která nejčastěji slouží k přenosu dat mezi mikrokontrolérem a jeho periferiemi.

Druhá část práce je zaměřena na návrh systému obsahující hradlové pole, jehož konfiguraci je možné vzdáleně přehrát. Je zde popsán koncept celého systému, který definuje vlastnosti zařízení MASTER, SLAVE a rozhraní, pomocí kterého jsou mezi sebou schopna zařízení komunikovat. Zařízení MASTER je v tomto systému realizováno osobním počítačem s rozhraním UART, obsahující vhodný komunikační terminál. Komunikační terminál musí být schopen vysílat zařízení SLAVE řídící povely, na základě kterých je zařízení SLAVE schopno přepínat mezi referenční konfigurací a aplikační konfigurací. Terminál musí být také schopen odeslat zařízení SLAVE soubor, obsahující novou aplikační konfiguraci, podle které je následně schopno zařízení SLAVE vykonávat svoji funkci. Jako zařízení SLAVE byl vybrán vývojový kit MAX10 FPGA Development kit, obsahující hradlové pole řady MAX10 od výrobce Intel. Důvod výběru vývojového kitu byla schopnost použitého hradlového pole přepínat mezi konfiguračními obrazy, které jsou uloženy v interní flash paměti, přítomnost rozhraní UART a poměrně nízká cena.

V závislosti na zvolených prostředcích pro realizaci systému je v této práci navržena architektura, ve které jsou popsány funkce a činnosti jednotlivých bloků. Současně je popsána i praktická realizace jednotlivých bloků ve vývojovém prostředí Quartus. V dalších kapitolách je velmi podrobně popsán postup pro vygenerování a nahrání výsledných souborů do zařízení SLAVE.

Ke konci práce jsou ověřeny všechny body zadání, zda byly splněny v celém rozsahu. Je zde také otestována schopnost zařízení SLAVE rozeznat poškozený konfigurační obraz, včetně možnosti diagnostiky zařízení.

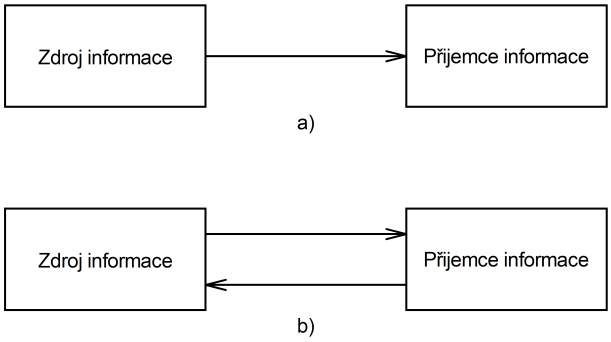
Závěrem jsou shrnuty výhody, nevýhody a možnosti použití navrhovaného řešení v praxi.

1 Datová komunikace

Datovou komunikaci lze chápat jako přenos informace na určitou vzdálenost. Komunikace nejčastěji probíhá dvoubodově mezi dvěma účastníky nebo mnohabodově mezi více účastníky. Při aplikaci v trakčních vozidlech je nejčastěji používána dvoubodová datová komunikace na vzdálenosti jednotek až stovek metrů.

Přenosová cesta v trakčních vozidlech je nejčastěji realizována pomocí optického nebo metalického vedení. V budoucnu lze uvažovat i bezdrátové přenosové cestě.

Na obr. 1.1 a) je schéma simplexní datové komunikace (zařízení komunikují v jednom směru) a na obr. 1.1 b) duplexní datové komunikace (zařízení komunikují v obou směrech). Duplexní datovou komunikaci lze ještě dále rozdělit na plně-duplexní (zařízení jsou schopna vysílat i přijímat v jeden okamžik) a polo-duplexní (zařízení se vzájemně střídají ve vysílání a přijímání). Pokud není vyžadována zpětná informace od příjemce, je používána komunikace podle schématu na obr. 1.1 a). Pokud je vyžadována zpětná informace od příjemce, např. za účelem diagnostiky nebo kontroly správnosti přenosu informace, je používána komunikace podle schématu na obr. 1.1 b). Mezi elektronickými zařízeními trakčních vozidel je nejčastěji používána komunikace podle schématu na obr. 1.1 b), čímž lze zajistit např. bezpečnou funkci přenosu informace, na kterou je v této problematice kladen velký důraz.



Obr. 1.1 Základní schéma datové komunikace a) simplexní, b) duplexní

1.1 Přenos dat

Obecně vzato, dvě a více zařízení jsou spolu schopna komunikovat, pokud jsou připojeny ke společné sběrnici. Sběrnice je skupina řídících, adresových a datových vodičů, pomocí kterých lze uskutečnit datový přenos. Na společnou sběrnici je obvykle připojeno více zařízení, ale v daný okamžik probíhá komunikace pouze mezi dvěmi zařízeními, která mají přidělenou svoji jedinečnou adresu.

Před samotným zahájením přenosu dat dochází k inicializaci. Inicializace spočívá v tom, že zařízení, které chce vysílat data tuto skutečnost oznámí ostatním zařízením. To je z důvodu, že na sběrnici již může probíhat komunikace mezi jinými zařízeními a následná komunikace více zařízení současně by celý přenos dat narušila.

Po inicializaci dochází k adresaci. Všechna zařízení připojená ke společné sběrnici mají svoji jedinečnou adresu. Vysílač dat tuto adresu vyšle a zařízení s touto adresou je následně připraveno k přijetí dat. Přenášená data se skládají z jednotlivých slov. Tato slova obvykle mají na svém začátku tzv. start bit, který signalizuje začátek a na konci tzv. stop bit, který signalizuje konec datového slova. Přenášené slovo může také obsahovat tzv. paritní bit, který slouží k detekci chyb.

Přenos dat může být realizován sériově, paralelně, synchronně nebo asynchronně. Při sériovém přenosu dat dochází k přenosu jednotlivých slov bit po bitu. Při paralelním přenosu je přenášeno celé slovo naráz. Z toho je zřejmé, že při stejných hodinových kmitočtech je paralelní přenos značně rychlejší, oproti sériovému. Paralelní přenos se vyznačuje větším počtem datových vodičů oproti sériovému přenosu. Mezi těmito vodiči může docházet vlivem vzájemných indukčností a kapacit k zarušení sběrnice a celá komunikace se může rozpadnout. U sériového přenosu, kde se využívají obvykle dva datové vodiče tento efekt není. Při synchronním přenosu je zdroj i příjemce dat řízen stejným hodinovým signálem. Při asynchronním přenosu je potřeba provádět synchronizaci pro každé datové slovo, což znamená, že je asynchronní přenos značně pomalejší oproti synchronnímu přenosu.

V dalších kapitolách budou popsány jednotlivé sběrnice a rozhraní, která se nejčastěji využívají při komunikaci mezi elektronickými zařízeními v trakčních vozidlech, ale i mezi jednotlivými součástkami, ze kterých jsou tato zařízení složena.

1.2 Referenční model ISO/OSI

Referenční model ISO/OSI je standard, který vypracovala organizace ISO na počátku 80. let. Model vznikl z důvodu kompatibility zařízení od různých výrobců, které bylo potřeba mezi sebou propojovat pomocí sítí, za účelem vzájemné komunikace. Proto vznikl sedmivrstvý model, ve kterém jsou popsány jednotlivé vrstvy z hlediska jejich funkcí a poskytovaných služeb. V následujících kapitolách budou jednotlivé vrstvy velmi stručně popsány.

1.2.1 Aplikační vrstva

Poskytuje rozhraní k uživatelského softwaru.

1.2.2 Prezentační vrstva

Konverzuje data mezi obecným formátem a formátem definovaným pro koncové zařízení.

1.2.3 Relační vrstva

Navazuje, udržuje a ukončuje spojení mezi koncovými zařízeními. Zajišťuje tak vhodné podmínky pro komunikaci mezi koncovými zařízeními.

1.2.4 Transportní vrstva

Vytváří z dat pakety pro síťovou vrstvu a zajišťuje jejich bezpečný přenos mezi koncovými zařízeními.

1.2.5 Síťová vrstva

Zprostředkovává přenos dat do správných uzlů, tedy zajišťuje směrování a volbu přenosové cesty datových rámců, které se v této vrstvě nenazývají jako datové rámce, ale jako pakety.

1.2.6 Linková vrstva

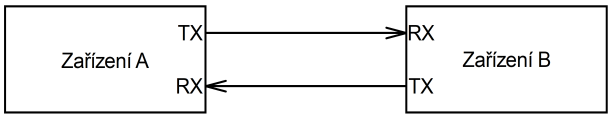
Popisuje pravidla přístupu na sběrnici, zajišťuje bezchybný přenos celých datových rámců mezi dvěma uzly.

1.2.7 Fyzická vrstva

Definuje přenosové médium - předepisuje použití konkrétních konektorů a jejich zapojení, popisuje elektrické úrovně logických signálů.

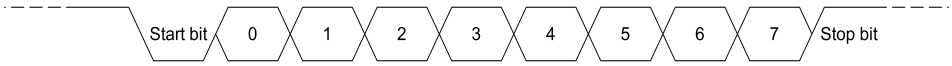
2 UART

UART je zařízení pro sériový asynchronní přenos dat. Jedná se o protokol, ze kterého jsou odvozena rozhraní např. RS-232, RS-422 a RS-485. UART má dva datové vodiče RX a TX. Datový vodič RX slouží pro přijímání a vodič TX pro odesílání dat. Aby spolu dvě zařízení mohla navzájem komunikovat, je potřeba je propojit pomocí datových vodičů RX a TX tak, jak je naznačeno na obr. 2.1.



Obr. 2.1 Propojení dvou zařízení pomocí UART

Důležité je, aby vysílač i přijímač dat měli nastavenou shodnou přenosovou rychlost, která se uvádí v počtu přenesených bitů za sekundu. Nastavení přenosové rychlosti se obvykle provádí pomocí konfiguračních registrů, ve kterých lze také nastavit počet přenášených bitů datového slova, délku stop bitu případně paritu. Na obr. 2.2 je znázorněn osmi bitový asynchronní přenos, bez přítomnosti paritního bitu. Logická úroveň „1“ znamená, že je zařízení v klidovém stavu. Start bit v logické úrovni „0“ signalizuje přijímači, že bude vysílač vysílat data. Poté vysílač vyšle datové slovo o velikosti 5-ti až 8-mi bitů. Pokud bude použit paritní bit, bude vždy umístěn za datové slovo. Nakonec vysílač vyšle stop bit v logické úrovni „1“. Tím signalizuje přijímači, že je datové slovo kompletní. V tomhle okamžiku zařízení zůstává v klidu, nebo přichází nový start bit a celý proces se opakuje. Napěťové úrovně pro UART jsou dány konkrétní použitou technologií daného zařízení (obvykle TTL nebo CMOS).



Obr. 2.2 Osmi bitový asynchronní přenos

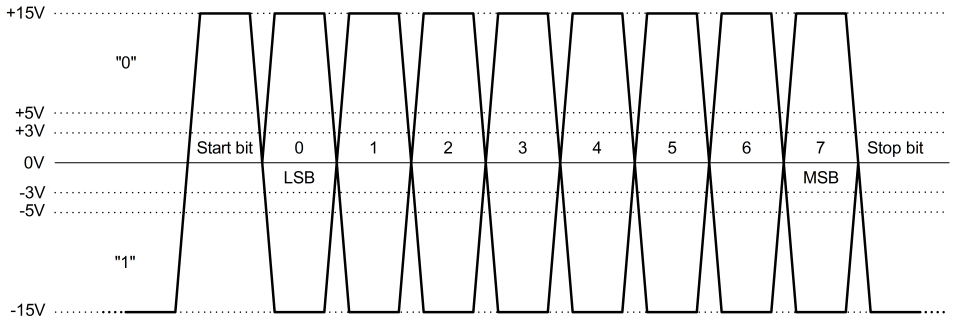
V praxi se lze také setkat s pojmem USART. V tomto případě se jedné o zařízení, které lze nastavit jak pro asynchronní, tak i pro synchronní datový přenos. Rozdíl mezi asynchronním a synchronním přenosem je vysvětlen v kapitole 1.1.

3 RS-232

RS-232 [1] [2] je standard, který byl navržen z důvodu kompatibility různých zařízení. Poslední verze pochází z roku 1967 a označuje se jako RS-232C. Zařízení, která využívají RS-232 lze rozdělit do dvou typů. První typ je označován jako DTE a lze jej chápat jako všechna koncová zařízení. Druhý typ je označován jako DCE a lze jej chápat jako zařízení, která zprostředkovávají komunikaci nebo také poskytují nějakou službu (např. myš k PC). Zařízení DTE a DCE lze od sebe na první pohled vizuálně rozeznat tak, že pro DTE se používají D-SUB konektory s piny a u DCE se používají D-SUB konektory se zdířkami – viz obr. 3.2.

3.1 Napěťové úrovně a časový průběh signálů

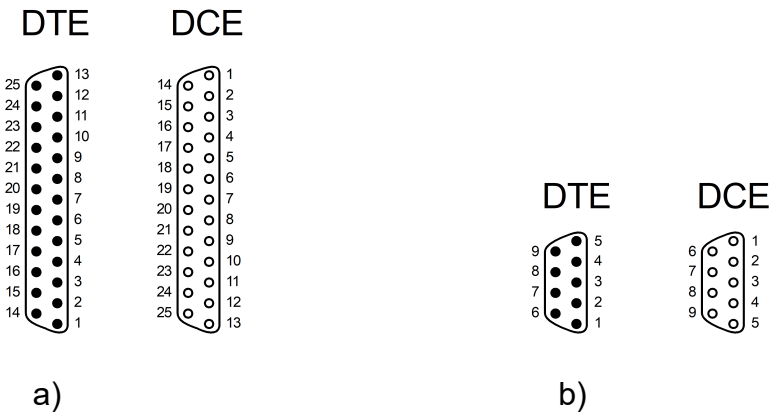
Na obr. 3.1 je časový průběh logického signálu, kde jsou vyznačeny jednotlivé napěťové úrovně. Princip přenosu dat je identický, jako bylo uvedeno v kapitole 2 s tím, že rozhraní RS-232 má jiné napěťové úrovně. Tyto napěťové úrovně jsou shodné jak pro datové, tak pro řídící signály. Pro datové a řídící signály platí, že se napěťové úrovně pro vysílač pohybují pro logickou úroveň „1“ od -5V do -15V a pro logickou úroveň „0“ od +5V do +15V . U přijímače se napěťové úrovně pro logickou úroveň „1“ pohybují v rozsahu od -3V do -15V a pro logickou úroveň „0“ od +3V do +15V .



Obr. 3.1 Napěťové úrovně a časový průběh rozhraní RS-232

3.2 Způsob propojení zařízení

Pro rozhraní RS-232 jsou nejčastěji používány D-SUB konektory s 25-ti nebo 9-ti piny s tím, že v praxi jsou nejčastěji používány 9-ti pinové konektory. Na obr. 3.2 a) je zapojení D-SUB konektoru s 25-ti piny pro zařízení DTE a DCE a na obr. 3.2 b) je zapojení D-SUB konektoru s 9-ti piny pro zařízení DTE a DCE. Na obr. 3.3 je tabulka, kde jsou popsány signály pro oba známe typy zařízení. Pokud je potřeba mezi sebou propojit zařízení DTE a DCE, používá se přímý propojovací kabel. Pokud je potřeba propojit dvě zařízení DTE, použije se křížený propojovací kabel. Z výše uvedeného vyplývá, že pomocí rozhraní RS-232 lze propojit pouze dvě zařízení, které spolu mohou komunikovat.



Obr. 3.2 Zapojení D-SUB konektorů a) pro 25 pinů b) pro 9 pinů

Signály RXD a TXD slouží pro datový přenos. Signál SGND je signálová zem. Ostatní signály slouží k hardwarovému řízení toku datového přenosu. Pod pojmem „hardwarové řízení toku dat“ si lze představit, že přijímač vysílá vysílači signály, kterými potvrdí, že byla data přijata, nebo že je zaneprázdněn a nová data ještě není schopen přijmout. Existuje také druhý způsob, který se označuje jako softwarové řízení toku dat. To spočívá v řízení toku dat pomocí znaků, které jsou vysílány datovými signály a následně jsou vysílačem vyhodnocovány pomocí softwaru. Tím je vysílač schopen vyhodnotit, zda je přijímač zaneprázdněn nebo připraven k přijetí nových dat.

Název	D-SUB 9	D-SUB 25	DTE	DCE	Popis
RXD	2	3	Vstup	Výstup	Data z DCE do DTE
TXD	3	2	Výstup	Vstup	Data z DTE do DCE
CTS	8	5	Vstup	Výstup	DCE oznamuje DTE volnou cestu
RTS	7	4	Výstup	Vstup	DTE oznamuje DCE volnou cestu
DSR	6	6	Vstup	Výstup	DCE oznamuje DTE, že je připraven pro komunikaci
DTR	4	20	Výstup	Vstup	DTE oznamuje DCE, že je připraven pro komunikaci
CD	1	8	Vstup	Výstup	DCE oznamuje DTE, že detekoval nosný kmitočet
RI	9	22	Vstup	Výstup	DCE oznamuje DTE, že detekoval signál zvonění
SGND	5	7	-	-	Signálová zem

Obr. 3.3 Přehled signálů rozhraní RS-232 a jejich popis

3.3 Přenosová rychlost

Při komunikaci mezi dvěmi zařízeními pomocí RS-232 se nepřenáší pouze datové bity, ale i synchronizační bity – viz kapitola 3.1. Z tohoto důvodu je potřeba rozlišovat přenosovou rychlost, která je dána počtem všech přenesených bitů za sekundu a datovou přenosovou rychlost, která je dána pouze počtem přenesených datových bitů za sekundu. Maximální přenosová rychlost, kterou jsou mezi sebou dvě zařízení schopna komunikovat pomocí rozhraní RS-232 je 115200 bit/s. Dílčí rychlosti 57600, 38400, 28800, 23040, 19200, 9600, 4800, 2400, 1200, 600, 600, 150 a 110 jsou od

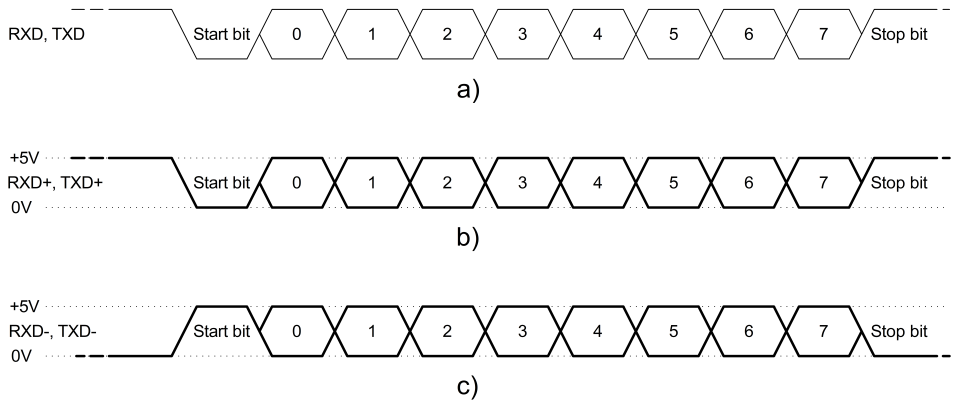
této rychlosti odvozené. Podle specifikace, jsou spolu schopna dvě zařízení komunikovat pomocí RS-232 do vzdálenosti 15 metrů. V praxi je však tato vzdálenost větší a může se pohybovat výjimečně až ve stovkách metrů. Je třeba mít také na paměti, že přenosová rychlost se se vzdáleností značně snižuje.

4 RS-422

RS-422 [1] [3] [4] je standard, který je založen na podobném principu, jako RS-232. Standard byl navržen z důvodu snížení rušení a přenosu dat na větší vzdálenosti. Oproti standardu RS-232 zde nejsou definována zapojení ani použití konkrétních konektorů.

4.1 Napěťové úrovně a časový průběh signálů

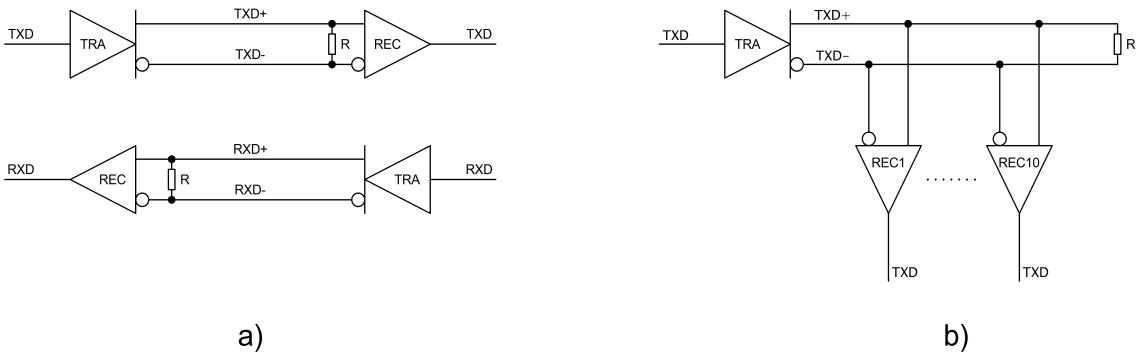
Rozhraní s RS-422 používá pro každý signál RXD a TXD pár vodičů oproti rozhraní s RS-232, který používá pro každý signál pouze jeden vodič. Mezi jednotlivými páry vodičů je rozdíl potenciálů, tedy je zde využíváno vlastností diferenciálního signálu. Diferenciální signál je odolnější vůči rušení, což umožňuje přenos dat na větší vzdálenosti. Na obr. 4.1 b) je časový průběh, nenegovaných signálů RXD+ a TXD+ a na obr. 4.2 c) je časový průběh negovaných signálů RXD- a TXD-. Rozdílové napětí mezi jednotlivými signály se podle standardu RS-422 může pohybovat od -7V do 7V, v praxi se ale nejčastěji setkáme s rozdílovým napětím od -5V do 5V. Logické úrovni „1“ odpovídá rozdíl napětí signálů (U_{RXD+}) - (U_{RXD-}) < 0,2V resp. (U_{TXD+}) - (U_{TXD-}) < 0,2V. Logické úrovni „0“ odpovídá rozdíl napětí signálů (U_{RXD+}) - (U_{RXD-}) > 0,2V resp. (U_{TXD+}) - (U_{TXD-}) > 0,2V.



Obr. 4.1 Napěťové úrovně a časový průběh rozhraní RS-422 a) signálů RXD, TXD b) signálů RXD+, TXD+ c) signálů RXD-, TXD-

4.2 Způsob propojení zařízení

Na obr. 4.2 a) je schéma zapojení čtyřvodičové plně-duplexní datové komunikace dvou zařízení, realizované pomocí rozhraní RS-422. Vysílače, označené jako TRA vysílají diferenciální signál, který je dále šířen nejčastěji kroucenou dvojlinkou. Pro potlačení odrazů a šumů je na koncích vedení použit zakončovací odpor, který impedančně přizpůsobuje celé vedení. Přijímače, označené jako REC převádí diferenciální signál zpět na signál obyčejný. Na obr. 4.2 b) je schéma zapojení dvouvodičové simplexní mnohabodové datové komunikace, které může obsahovat jeden vysílač a až deset přijímačů. Limitujícím parametrem počtu přijímačů je jejich vstupní impedance a impedance zakončovacího odporu vedení.



a) b)

Obr. 4.2 Schéma zapojení datové komunikace pomocí rozhraní RS-422

a) čtyřvodičové plně-duplexní dvoubodové b) dvouvodičové simplexní mnohabodové

4.3 Přenosová rychlost

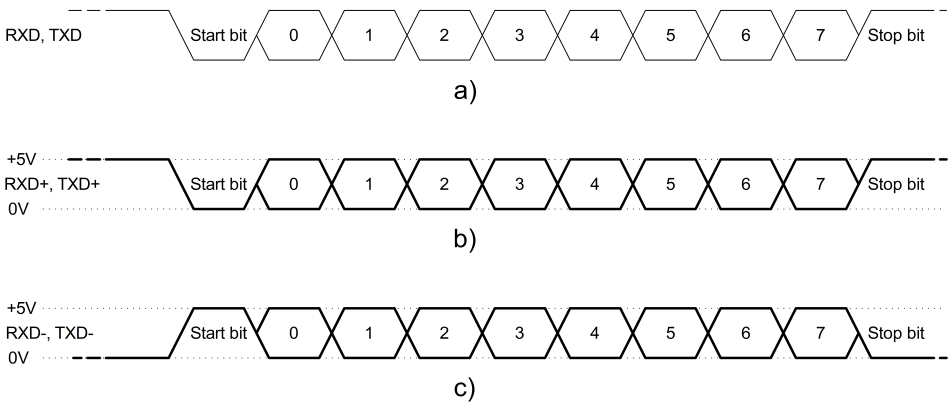
Přenosová rychlost dosahuje u rozhraní RS-422 až až 10 Mbit/s do vzdálenosti 15 metrů. Maximální vzdálenost, na kterou jsou spolu schopna zařízení komunikovat, uvádí standard RS-422 až na 1200 metrů. Jak lze ale očekávat, přenosová rychlost se se zvětšující vzdáleností opět značně snižuje.

5 RS-485

RS-485 [1] [3] [4] je standard, který je založen na podobném principu, jako RS-232. Standard byl navržen z důvodu snížení rušení a přenosu dat na větší vzdálenosti. Stejně jako u standardu RS-422 zde nejsou uvedena zapojení ani použití konkrétních konektorů.

5.1 Napěťové úrovně a časový průběh signálů

Rozhraní s RS-485 používá pro každý signál RXD a TXD pár vodičů oproti rozhraní s RS-232, které používá pro každý signál pouze jeden vodič. Mezi jednotlivými páry vodičů je rozdíl potenciálů, tedy je zde využíváno vlastností diferenciálního signálu. Diferenciální signál je odolnější vůči rušení, což umožňuje přenos dat na větší vzdálenosti. Na obr. 5.1 b) je časový průběh nenegovaných signálů RXD+ a TXD+ a na obr. 5.1 c) je časový průběh nenegovaných signálů RXD- a TXD-. Rozdílové napětí mezi jednotlivými signály se podle standardu RS-485 může pohybovat v rozsahu od -7V do 12V, v praxi se ale nejčastěji setkáme s rozdílovým napětím od -5V do 5V. Logické úrovni „1“ odpovídá rozdíl napětí signálů (U_{RXD+}) - (U_{RXD-}) < 0,2V resp. (U_{TXD+}) - (U_{TXD-}) < 0,2V. Logické úrovni „0“ odpovídá rozdíl napětí signálů (U_{RXD+}) - (U_{RXD-}) > 0,2V resp. (U_{TXD+}) - (U_{TXD-}) > 0,2V.

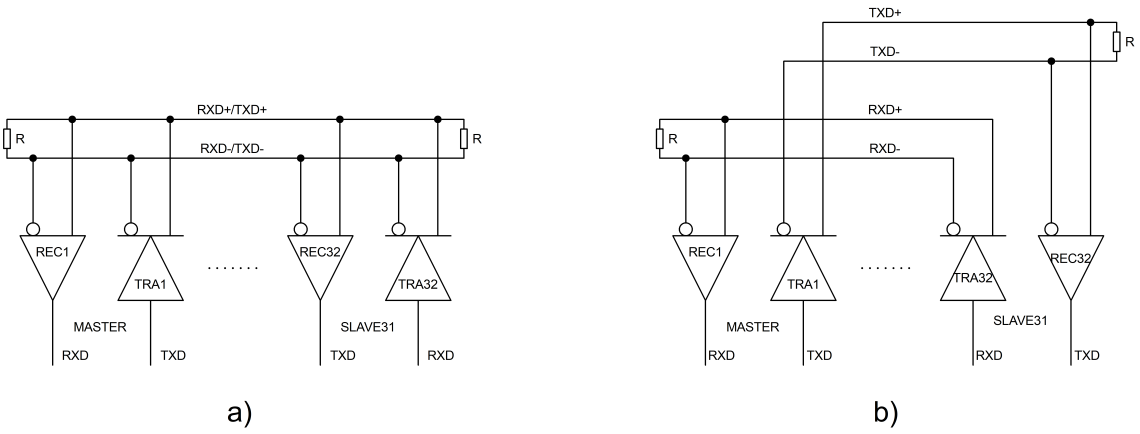


Obr. 5.1 Napěťové úrovně a časový průběh rozhraní RS-485 a) signálů RXD, TXD b) signálů RXD+, TXD+ c) signálů RXD-, TXD-

5.2 Způsob propojení zařízení

Na obr. 5.2 a) je schéma zapojení dvouvodičové polo-duplexní datové komunikace s možností připojení až 32 zařízení komunikujících pomocí rozhraní RS-485. Vysílače, označené jako TRA vysílají diferenciální signál, který je dále šířen nejčastěji kroucenou dvojlinkou. Pro potlačení odrazů a šumů je na koncích vedení použit zakončovací odpor, který slouží jako impedanční přizpůsobení. Přijímače, označené jako REC převádí diferenciální signál zpět na signál obyčejný. V tomto zapojení je obvykle jedno zařízení, které se označuje jako MASTER a až 31 zařízení, která se označují jako SLAVE. Zařízení MASTER vysílá příkazy a zařízení SLAVE na ně odpovídají. Protože není možné, aby vysílalo více zařízení současně, je potřeba aplikovat vhodné řízení datového toku, což ale nespecifikuje standard RS-485, ale konkrétní použitý komunikační protokol

vyšší vrstvy. Na obr. 5.2 b) je schéma zapojení čtyřvodičové plně-duplexní datové komunikace. V tomto zapojení je opět obvykle jedno zařízení, které se označuje jako MASTER a až 31 zařízení, která se označují jako SLAVE. Zařízení MASTER vysílá příkazy a zařízení SLAVE na ně odpovídají.



Obr. 5.2 Schéma zapojení datové komunikace pomocí rozhraní RS-485

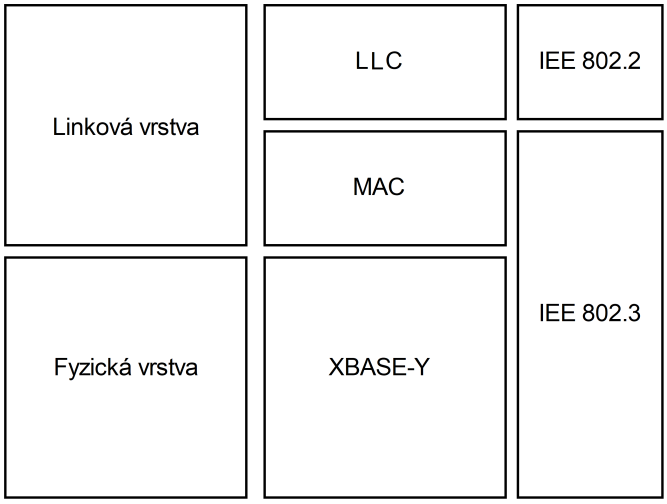
a) dvouvodičové polo-duplexní b) čtyřvodičové plně-duplexní

5.3 Přenosová rychlost

Přenosová rychlost dosahuje u rozhraní RS-485 až 10 Mbit/s do vzdálenosti 15 metrů. Maximální vzdálenost, na kterou jsou spolu schopna zařízení komunikovat, uvádí standard RS-485 až na 1200 metrů. Opět i zde platí, že se přenosová rychlost se zvětšující vzdáleností snižuje.

6 Ethernet

Ethernet [5] je technologie, která umožňuje propojování různých zařízení pomocí sítí za účelem jejich vzájemné komunikace. V dnešní době je Ethernet velmi používaný a to z hlavních dvou důvodů. První z důvodů je jeho poměrně vysoká spolehlivost a druhý z důvodů je jeho příznivá cena. Ethernet vychází z modelu ISO/OSI, kde zaujímá linkovou a fyzickou vrstvu. ISO/OSI model je detailněji popsán v kapitole 1.2. Na obr. 6.1 je základní struktura ethernetových vrstev a standardů které, jej popisují.



Obr. 6.1 Základní struktura ethernetových vrstev z hlediska ISO/OSI modelu

6.1 Linková vrstva

Linková vrstva se skládá z podvrstev LLC (definované standardem IEEE 802.2) a MAC (definované standardem IEEE 802.3). Podvrstva LLC má za úkol zajistit vhodné řízení toku dat tak, aby bylo možné přepínat mezi různými komunikačními protokoly, navazovat a přerušovat spojení, zajistit kontrolu přenášených dat apod..

Podvrstva MAC má za úkol řídit přístup ke společné sběrnici na základě metody CSMA/CD. Metoda CSMA/CD ve zkratce říká, že může v jeden okamžik vysílat pouze jedno zařízení, které je připojené ke společné sběrnici. Ostatní zařízení mezi tím sledují stav, zda na sběrnici probíhá komunikace. Pokud je komunikace všech zařízení připojených ke sběrnici úspěšně dokončena, zařízení, které čekalo na vysílání může v tento okamžik začít vysílat vlastní data. Může také nastat situace, kdy budou vysílat dvě zařízení současně. Tím by vznikl stav, který by celou komunikaci narušil. Metoda CSMA/CD tento problém bere v úvahu a dokáže ho včas detekovat. První zařízení, které problém detekuje, vyšle ostatním zařízením signál, která se následně odmlčí. Po uplynutí určité doby se zařízení pokusí o opakované vysílání.

Data jsou Ethernetem přenášena pomocí rámců, které jsou definovány různými standardy. Na obr. 6.2 je příklad přenosového rámce definovaného standardem IEEE 802.3, který se skládá ze sedmi dílčích částí:

PRE Synchronizační značka přijímacího a vysílacího zařízení

SFD Značí konec synchronizační sekvence

DA MAC adresa cílového zařízení

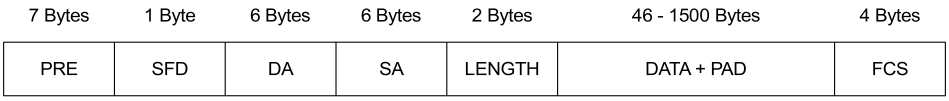
SA MAC adresa zdrojového zařízení

LENGTH Délka rámce

DATA Přenášená datová zpráva

PAD Doplnění minimální délky rámce

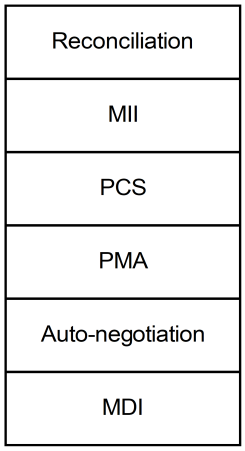
FSC Kontrola pomocí CRC kódu



Obr. 6.2 Rámec definovaný IEEE 802.3 pro Ethernet [5]

6.2 Fyzická vrstva

Na obr. 6.3 je detailnější popis fyzické vrstvy Ethernetu. Podvrstvy Reconciliation a MII zajišťují přechod mezi podvrstvou MAC a nižšími podvrstvami fyzické vrstvy. Podvrstva PCS je zodpovědná za kódování a dekódování dat, PMA jsou obvody příjmu a vysílání dat, Auto-negotiation slouží k automatickému nastavení možností komunikace (např. rychlosti). Poslední podvrstvu MDI lze chápat jako konkrétní fyzické rozhraní (konektor).



Obr. 6.3 Detail fyzické vrstvy Ethernetu [5]

6.2.1 Způsob propojení zařízení

V počátcích Ethernetu se zařízení propojovala pomocí koaxiálních kabelů, které mohli dosahovat vzdáleností až 500m. Na koncích koaxiálního vedení byl vždy umístěn zakončovací odpor, který sloužil jako impedanční přizpůsobení. Technologie, které se nejčastěji používaly k šíření Ethernetu koaxiálními kabely, nesly nejčastěji označení 10BASE2 nebo 10BASE5.

Později se začaly používat k propojování zařízení kabely s kroucenými dvojlinkami. Kabely se dělí na křížené (slouží k přímému propojení dvou zařízení) a přímé (slouží k propojení zařízení pomocí switche). V dnešní době, se jak pro propojení dvou zařízení, tak pro propojování více zařízení pomocí switche používají přímé kabely. Důvodem je, že dnešní síťové karty umožňují automatické překřížení signálů, které jsou poté přivedeny do konektoru. Tyto technologie se označují 10BASE-X, 10BASE-FX, 10BASE-T, 100BASE-TX, 1000BASE-T, 1000BASE-X apod..

Ethernet lze mezi dvěma místy také přenášet pomocí optických kabelů. Výhoda je, že lze přenášet data na poměrně velkou vzdálenost a přenos není ovlivňován elektromagnetickým rušením.

6.3 Přenosová rychlost

U Ethernetu, šířeným pomocí koaxiálních kabelů dosahuje přenosová rychlost 10 Mbit/s. Pokud je použit k přenosu kabel s kroucenou dvojlinkou, lze dosáhnout přenosové rychlosti až 1 Gbit/s. Pomocí optických kabelů lze dosáhnou přenosové rychlosti až 10 Gbit/s. I jako u předchozích rozhraních, které byly v této práci popsány, i zde platí, že se přenosová rychlost se zvětšující vzdáleností snižuje.

7 CAN

CAN [6] je sběrnice, která umožňuje propojování snímačů a akčních členů s řídící jednotkou v průmyslu, automobilech, trakčních vozidlech apod.. CAN sběrnice je popsána standardem ISO 11898, kde jsou uvedeny i jednotlivé verze. Z hlediska ISO/OSI modelu, standard ISO 11898 popisuje fyzickou a linkovou vrstvu.

7.1 Linková vrstva

Jak bylo řečeno výše, všechna zařízení jsou připojena ke společné sběrnici a vzájemná komunikace probíhá na základě datových rámců. Na obr. 7.1 je obecný datový rámec pro CAN. Rámec se může v jednotlivých polích lišit v počtu bitů, čímž jsou podle ISO 11898 dány jednotlivé verze CAN. Obecný rámec je rozdělen na 7 polí, kde jednotlivé pole mají význam:

SOF Start bit

AR Arbitrační pole

RTR RTR bit

IDE Identifikace formátu zprávy

DATA Přenášená datová zpráva

CRC Kontrola pomocí CRC kódu

ACK Potvrzení přijetí zprávy

EOF Zakončovací pole bitů

Obr. 7.1 Obecný datový rámec pro CAN sběrnici

Aby se zařízení vzájemně neovlivňovala v komunikaci, je potřeba je řídit vhodnou arbitrací pomocí arbitračního pole datového rámce. Arbitrace spočívá v tom, že jednotlivá zařízení mají svoji jedinečnou identifikaci, která je uložena v arbitračním poli datového rámce. Všechna zařízení tedy mohou začít vysílat v libovolný okamžik, kdy je sběrnice v klidovém stavu. Pokud se o přístup na sběrnici snaží dvě nebo více zařízení ve stejný okamžik, dostane přednost to zařízení, které má v arbitračním poli danou největší přednost. Až toto zařízení úspěšně přenos dokončí, je následně umožněno přistoupit na sběrnici i ostatním zařízením s nižší předností. V případě shody arbitračního pole datového rámce u více zařízení, je o přednosti ve vysílání rozhodnuto pomocí bitu RTR.

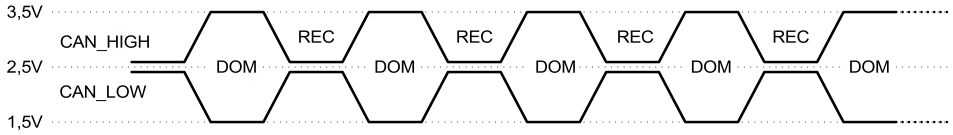
7.2 Fyzická vrstva

Samotná CAN sběrnice je nejčastěji realizována kroucenou dvojlinkou, po které jsou přenášena data. Synchronizace zařízeních, která jsou připojena k CAN sběrnici probíhá pomocí synchronizačních bitů. Pokud vysílací zařízení vyšle 5 bitů po sobě jdoucích, které mají stejnou hodnotu, vysílací zařízení vyšle jeden bit opačné hodnoty. Přijímací zařízení si díky tomuto generují vlastní hodinový signál, který je synchronní se všemi ostatními zařízeními na sběrnici. Přijímací zařízení tento bit následně odstraní.

7.2.1 Napěťové úrovně a časový průběh signálů

Standard ISO 11898, který popisuje CAN sběrnici rozlišuje na sběrnici dvě logické hodnoty, které označuje jako dominantní a recesivní. Pokud všechna zařízení generují recesivní úroveň, pak je na sběrnici také recesivní úroveň. Pokud jedno ze zařízení generuje dominantní úroveň, pak je na sběrnici dominantní úroveň. Na obr. 7.2 jsou napěťové úrovně a časový průběh signálů na CAN sběrnici. V klidovém stavu je na obou signálových vodičích CAN sběrnice napětí o velikosti 2,5V.

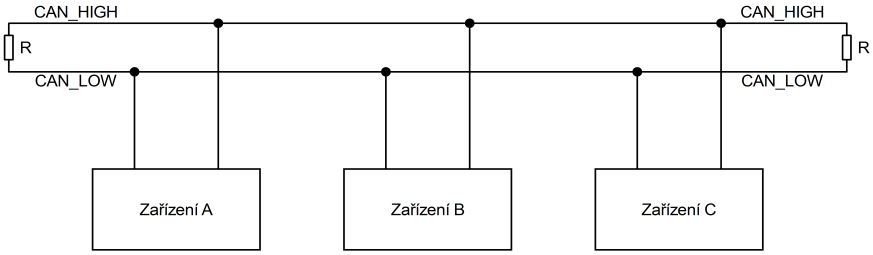
Dominantní úrovni odpovídá rozdíl napětí signálů $U_{CAN\_HIGH} > U_{CAN\_LOW} + 0{,}9V$ a recesivní úrovni odpovídá rozdíl napětí signálů $U_{CAN\_HIGH} < U_{CAN\_LOW} + 0{,}5V$.



Obr. 7.2 Napěťové úrovně a časový průběh rozhraní na CAN sběrnici

7.2.2 Způsob propojení zařízení

Na obr. 7.3 je schéma, kde je znázorněno propojení jednotlivých zařízení pomocí CAN sběrnice. K propojení jednotlivých zařízení je používána kroucená dvojlinka. Pro potlačení odrazů a šumů je na obou koncích použit zakončovací odpor, který impedančně přizpůsobuje celé vedení.



Obr. 7.3 Schéma zapojení komunikace pomocí CAN sběrnice

7.3 Přenosová rychlost

Přenosová rychlost, kterou spolu zařízení mohou komunikovat pomocí sběrnice CAN, je maximálně 10 Mbit/s do vzdálenosti 40-ti metrů. Přenosová rychlost se ovšem se zvětšující vzdáleností snižuje. Standard ISO 11898 udává maximální délku vedení pro CAN na 1000 metrů. Přenosová rychlost se na tuto vzdálenost pohybuje maximálně do 40kbit/s.

8 SPI

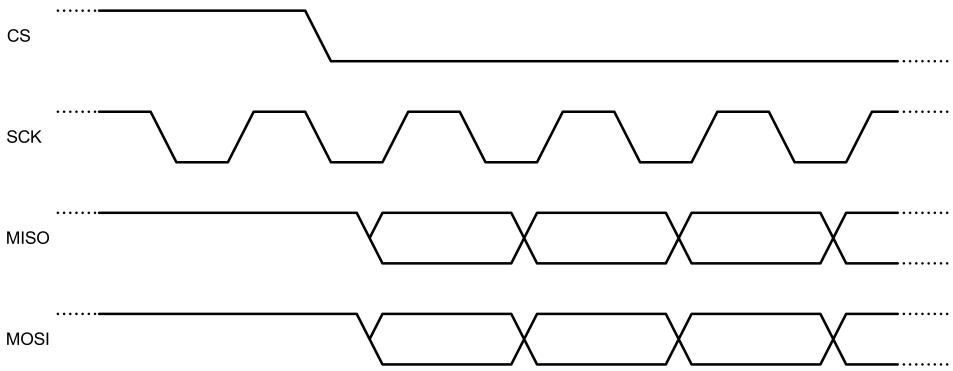
SPI [6] je rozhraní pro sériovou komunikaci, kde je jedno zařízení MASTER a jedno, nebo více zařízení SLAVE. Zařízení MASTER je nejčastěji mikrokontrolér a zařízení SLAVE mohou být různé periferní obvody např. paměti, akcelerometry, převodníky apod..

8.1 Napěťové úrovně a časový průběh signálů

Na obr. 8.1 je časový průběh všech signálů rozhraní SPI. Na obrázku nejsou vyznačeny

napěťové úrovně jednotlivých signálů. Důvodem je, že napěťové úrovně mohou být různé a jsou dány konkrétní použitou technologií.

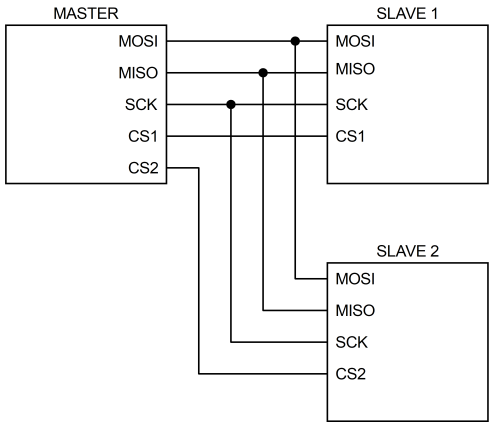
Signál MOSI značí, že pro zařízení MASTER je tento signál výstupem a pro zařízení typu SLAVE vstupem. Signál MISO značí, že pro zařízení MASTER je tento signál vstupem a pro zařízení typu SLAVE výstupem. Signálem CS vybírá zařízení MASTER jedno jediné zařízení typu SLAVE, které bude vysílat, resp přijímat svá data. Tím je zajištěno, že se zařízení nebudou vzájemně ovlivňovat v komunikaci. Posledním signálem SCK je hodinový signál, který je generován zařízením MASTER a slouží k synchronizaci celé komunikace.



Obr. 8.1 Časový průběh signálů rozhraní SPI

8.2 Způsob propojení zařízení

Na obr. 8.2 je schéma zapojení, kde jsou propojena dvě zařízení SLAVE se zařízením MASTER komunikující přes rozhraní SPI.



Obr. 8.2 Schéma zapojení komunikace pomocí SPI rozhraní

8.3 Přenosová rychlost

Jelikož je rozhraní SPI konstruováno pro přenos dat mezi mikrokontrolérem a jeho

periferními obvody, vzdálenost, na kterou je možné přenášet data se pohybuje pouze v řádu desítek centimetrů. Maximální frekvence hodinového kmitočtu se uvádí v řádu desítek MHz, z čehož lze odvodit i maximální možnou přenosovou rychlost.

9 I2C

I2C [6] je stejně jako SPI rozhraní pro sériovou komunikaci. Rozdíl je v tom, že zařízení komunikující přes SPI mohou být zároveň jak MASTER tak SLAVE. Rozhraní opět slouží ke komunikaci mezi mikrokontrolérem a jeho periférními obvody např. pamětmi, akcelerometry, převodníky apod..

9.1 Linková vrstva

Zařízení, připojená ke společné sběrnici mezi sebou komunikují na základě datových rámců. Na obr. 9.1 je datový rámec, který se skládá ze sedmi polí, kde jednotlivá pole mají význam:

SOF Start přenosu

ADR Adresa SLAVE zařízení

R/W Určuje, zda se bude číst (R) nebo zapisovat (W) do SLAVE

ACK Potvrzení od SLAVE zařízení, že přijalo data

DATA Přenášená datová zpráva

EOF Zakončovací pole bitů

Obr. 9.1 Datový rámec pro I2C rozhraní

Aby se zařízení vzájemně neovlivňovala v komunikaci, je opět potřeba řídit přístup ke sběrnici vhodnou arbitrací. Každé zařízení, které je připojeno ke sběrnici, může kdykoli zahájit svůj přenos pod podmínkou, že je sběrnice v klidovém stavu. Pokud jedno ze zařízení v daném okamžiku vysílá data, zároveň je také neustále porovnává s daty na sběrnici. Pokud zařízení zaznamená rozdíl vysílaných dat a dat na sběrnici, je nuceno neprodleně ukončit vysílání.

Samotná arbitrace probíhá pomocí bitového pole v datovém rámci, které je značeno jako ADR. Pole ADR obsahuje adresu zařízení, pro které jsou data určena. Pokud zařízení zachytí datový rámec s jeho adresou, následuje přenos dat mezi MASTER a SLAVE, který je ukončen

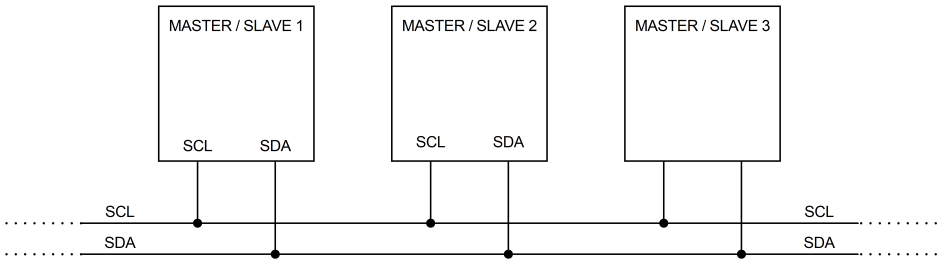
polem ACK a následuje přenos dalších dat nebo ukončení komunikace. Adresy jednotlivých zařízení, jsou dány výrobcem konkrétního integrovaného obvodu podporující komunikaci rozhraním I2C.

9.2 Fyzická vrstva

Jednotlivá zařízení jsou propojena pouze jedním datovým vodičem, který se označuje jako SDA a hodinovým vodičem, který slouží k synchronizaci všech zařízení připojených na sběrnici, který se značí jako SCL.

9.2.1 Způsob propojení zařízení

Na obr. 9.2 je schéma zapojení, kde jsou propojena tři zařízení typu MASTER/SLAVE, komunikující přes rozhraní I2C.



Obr. 9.2 Schéma zapojení komunikace pomocí I2C rozhraní

9.3 Přenosová rychlost

Jelikož je rozhraní I2C stejně jako rozhraní SPI určeno pro přenos dat mezi mikrokontrolérech a jeho periferními obvody, vzdálenost, na kterou je možné přenášet data se pohybuje v opět v řádu desítek centimetrů. Tato vzdálenost je také omezena kapacitou přenosové cesty. Maximální možní frekvence hodinového signálu SCL se udává 400 kHz, z čehož plyne i maximální možná přenosová rychlost.

10 Koncept systému

V následujícím textu je popsán koncept systému formou požadavků, pro které jsou následně navrženy prostředky pro praktickou realizaci a jednoduchou ukázku funkčnosti celého systému.

Systém je složený ze dvou zařízení, která jsou schopna mezi sebou vzájemně komunikovat pomocí vhodného komunikačního rozhraní. Tato zařízení budou dále označovány jako MASTER a

SLAVE.

10.1 Požadavky na zařízení SLAVE

1) Zařízení SLAVE je zařízení, které obsahuje FPGA hradlové pole.
2) Hradlové pole musí umožňovat nahrání minimálně dvou konfiguračních obrazů, mezi kterými je možné na povel zařízení MASTER přepínat.
3) Konfigurační obrazy jsou umístěny ve vnitřní konfigurační paměti hradlového pole.
4) Jeden z obrazů obsahuje aplikační konfiguraci a druhý referenční konfiguraci.
5) V případě chybného vykonávání aplikační konfigurace je zařízení schopné tento stav rozpoznat a přepnout na referenční konfiguraci.
6) Referenční konfigurace umožňuje nahrání nové aplikační konfigurace ze strany zařízení MASTER pomocí komunikačního rozhraní.
7) Zařízení SLAVE obsahuje rozhraní pro prvotní nahrání konfigurace, pomocí kterého je možné v případě problému ve výrobě či vývoji zařízení diagnostikovat.

10.1.1 Volba prostředků pro splnění požadavků

Nejdříve byl proveden výběr vhodného hradlového pole. Hradlové pole, umožňující přepínání konfiguračních obrazů, které jsou uloženy ve vnitřní flash paměti není na trhu příliš. Jako nejvhodnější hradlové pole pro tuto konkrétní aplikaci bylo zvoleno hradlové pole řady MAX 10 od výrobce Intel (dříve Altera). Dále byl zvolen vývojový kit MAX10 FPGA Development kit, který obsahuje již zmíněné hradlové řady MAX10, s konkrétním výrobním číslem: 10M50DAF484C6GES. Toto výrobní číslo nám říká, že hradlové pole podporuje dvojí konfiguraci, že se skládá z 50 000 logických elementů v pouzdře FBGA s 484 piny a je schopno pracovat v teplotním rozsahu od 0°C do 85°C. Vývojový kit dále podporuje komunikační rozhraní UART, externí QSPI paměť, JTAG rozhraní, pomocí kterého je umožněno nahrání prvotní konfigurace ve výrobě a pomocí kterého je možné zařízení diagnostikovat (vyčítat obsah jednotlivých pamětí, registrů apod.).

Důvodem výběru hradlového pole řady MAX 10 od výrobce Intel byl také ten, že je cenově velmi výhodný. Dalším důvodem byla dostupnost dokumentace a referenčních manuálů, které jsou k dispozici zcela zdarma.

Vývojový kit podporuje i mnoho dalších funkcí, které zde nejsou popsány z toho důvodu, že

je zařízení SLAVE nijak nevyužívá.

10.2 Požadavky na zařízení MASTER

1) Zařízení MASTER umožňuje vysílat zařízení SLAVE řídící povely pro přepínání konfiguračních obrazů.
2) Zařízení MASTER umožňuje vysílání souboru s novou aplikační konfigurací do zařízení SLAVE, pomocí zvoleného komunikačního rozhraní.

10.2.1 Volba prostředků pro splnění požadavků

Jako zařízení MASTER byl zvolen osobní počítač, který má funkční rozhraní UART a má pro něj nainstalovaný vhodný komunikační terminál. Tento terminál musí být schopen přijímat a odesílat znaky a podporovat odesílání souborů.

10.3 Požadavky na komunikační rozhraní

1) Komunikační rozhraní musí být odolné proti rušení. Důvodem je použití navrhovaného systému v trakčních vozidlech.
2) Komunikační rozhraní musí být schopno komunikace na vzdálenosti desítek až stovek metrů.

10.3.1 Volba prostředků pro splnění požadavků

Jelikož MASTER a SLAVE podporují rozhraní UART, jako komunikační rozhraní byl tedy zvolen UART. Pro použití v praxi je vyžadována určitá odolnost vůči okolnímu rušení a pro přenos na větší vzdálenosti lze využít převod na rozhraní např. RS-232, RS-422 nebo RS-485. Rozhraní UART, RS-232, RS-422 a RS-485 jsou popsána v kapitolách 2, 3, 4 a 5.

11 Volba softwarových nástrojů

Vzhledem k výběru hradlového pole z řady MAX 10 od výrobce Intel, bylo zvoleno vývojové prostředí Quartus Prime ve verzi 18.0 Lite Edition, které je dostupné zcela zdarma. Quartus Prime podporuje nástroje pro analýzu a syntézu kódu v jazyce VHDL a Verilog apod..

Pro účely této práce byl zvolen nástroj Platform Designer, který automaticky generuje HDL kód v závislosti na propojení použitých komponent, které se nacházejí v tzv. IP katalogu. Jednotlivé

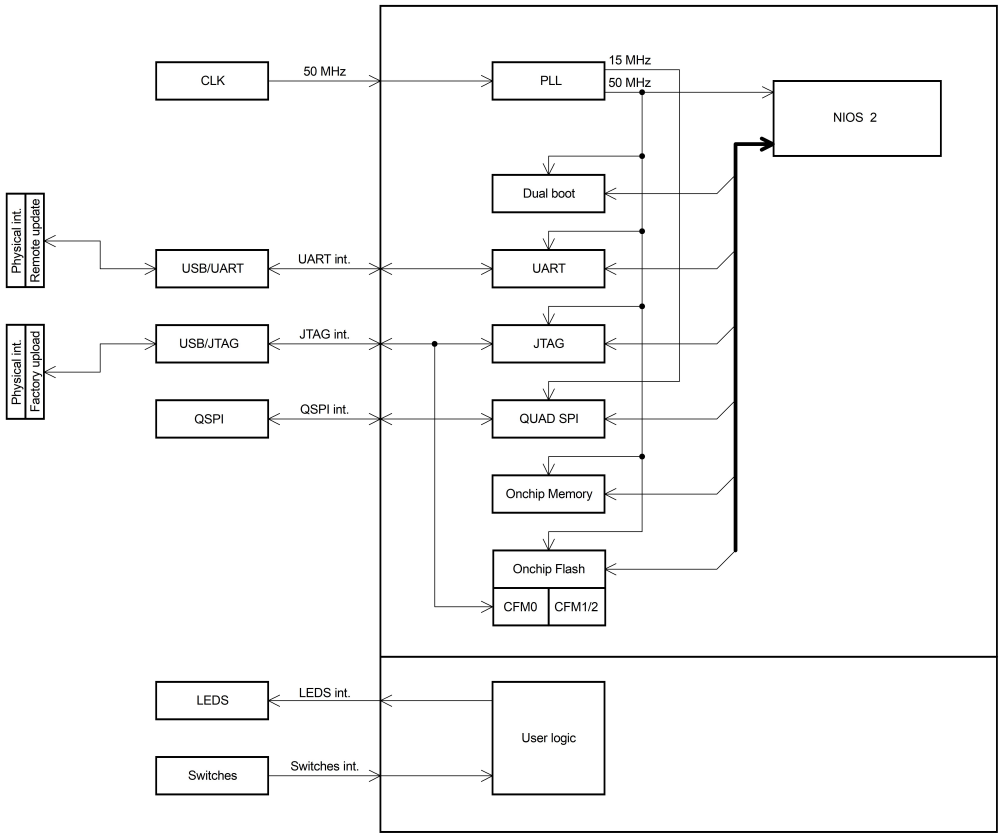
komponenty jsou mezi sebou graficky propojovány pomocí signálů a datových sběrnic, což je z hlediska přehlednosti v konkrétním návrhu velmi výhodné.

V návrhu systému je také použit procesor, který je implementován do výše zvoleného hradlového pole. Procesor slouží k obsluze periférií z IP katalogu, které jsou použity v daném systému. Aby mohl procesor vykonávat požadované instrukce, je pro něj potřeba vytvořit program, podle kterého bude schopný pracovat. K tomu byl zvolen nástroj NIOS II Software Build Tools for Eclipse. V tomto nástroji je vytvářen program pro procesor NIOS II v programovacím jazyce C.

12 Architektura systému

V této kapitole bude popsána architektura zařízení označovaného jako SLAVE, která se skládá z několika bloků. Nejdříve budou popsány činnosti jednotlivých bloků z obecného hlediska a následně bude popsána jejich praktická realizace.

Architektura na obr. 12.1 popisuje dvě základní funkce systému SLAVE. První funkce s Factory upload spočívá v nahrání prvotních konfigurací do interní flash paměti hradlového pole. Konkrétně se jedná o dvě konfigurace. První konfigurace, která slouží jako referenční, je nahrána do části paměti CFM0. Druhá konfigurace, která slouží jako aplikační je nahrána do částí paměti CFM1 a CFM2. Druhá funkce Remote update, slouží ke vzdálenému přehrání aplikační konfigurace, která se nachází v části konfigurační paměti CFM1 a CFM2. Tato konfigurace je vzdáleně přehrána pomocí rozhraní UART.



Obr. 12.1 Architektura systému zařízení SLAVE

12.1 Bloky Physical interface

Jak v případě funkce Remote update, tak v případě funkce Factory upload, tyto bloky znázorňují fyzické rozhraní, které slouží k propojení zařízení MASTER a SLAVE.

12.1.1 Praktická realizace

Jako fyzické rozhraní bylo pro funkci Remote update, tak pro funkci Factory upload byl na vývojovém kitu použit pětipinový MiniUSB konektor.

12.2 Blok CLK

Blok s názvem CLK symbolizuje zdroj hodinového signálu o frekvenci 50MHz, který je přiveden do hradlového pole.

12.2.1 Praktická realizace

Zdroj hodinového signálu je na vývojovém kitu realizován čtyřkanálovým

programovatelným oscilátorem s výrobním označením Si5338. Výstupní frekvence oscilátoru [7] jsou 25 MHz, 50 MHz, 100 MHz a 125 MHz. Pro účely této práce byl zvolen kanál oscilátoru s frekvencí 50 MHz. Parametry a číslo pinu pro připojení signálu k hradlovému poli jsou shrnuty v tabulce na obr. 12.2.

Název signálu vrcholové entity	Napěťová hladina	Číslo pinu na FPGA	Popis
clk_clk	2,5V	M9	50,000 MHz

Obr. 12.2 Parametry a číslo signálu pro připojení bloku CLK k hradlovému poli [7]

12.3 Blok USB/UART

Tento blok symbolizuje převodník z USB na UART. Důvod použití tohoto bloku je ten, že dnešní moderní počítače již neobsahují sériové rozhraní (označované jako COM). Místo toho ale obsahují rozhraní USB. Díky tomuto bloku lze v PC vytvořit tzv. virtuální COM port, díky kterému lze zajistit bezproblémovou komunikaci mezi PC a hradlovým polem.

12.3.1 Praktická realizace

Na vývojovém kitu je tento převodník realizován velmi známým obvodem FT232RQ od výrobce FTDI chip. Výstupem z tohoto obvodu jsou signály RXD a TXD, jejichž parametry a čísla pinů pro připojení k hradlovému jsou shrnuty v tabulce na obr. 12.3. Popis komunikace mezi dvěmi zařízeními přes rozhraní UART je popsáno v kapitole 2.

Pro správnou funkci obvodu FT232RQ je nutné mít v PC nainstalován aktuální ovladač. Ten se nainstaluje automaticky při prvním připojení k PC. Pokud instalace ovladače proběhne v pořádku, zobrazí se ve správci zařízení nové zařízení USB Serial Port s číslem konkrétního portu. Pokud se při instalaci ovladače vyskytne problém, je nutné postupovat podle pokynů na stránkách výrobce (www.ftdichip.com) daného obvodu.

Název signálu vrcholové entity	Napěťová hladina	Číslo pinu na FPGA	Popis
uart_external_connection_rxd	2,5V	Y19	Příjem asynchronních dat
uart_external_connection_txd	2,5V	W18	Vysílání asynchronních dat

Obr. 12.3 Parametry a čísla signálů pro připojení bloku USB/UART k hradlovému poli [7]

12.4 Blok USB/JTAG

Tento blok symbolizuje převodník z USB na JTAG, pomocí kterého je umožněno nahrání konfiguračních obrazů do hradlového pole. V tomto případě se jedná o obrazy prvotní konfigurace

při výrobě. JTAG také umožňuje diagnostiku připojených periférií (včetně použitých komponent z IP katalogu nástroje Platform Designer). JTAG také může být velice užitečný při vývoji aplikace, protože umožňuje vyčítání nebo zapisování obsahu pamětí, registrů použitých komponent a v případě potřeby umožňuje i debug režim.

12.4.1 Praktická realizace

Na vývojovém kitu je tato funkce realizována hradlovým polem řady MAX II, který je nakonfigurován již z výroby. Pro správnou funkci je nutné mít v PC nainstalován aktuální ovladač. Ovladač se nachází ve složce, kde je nainstalováno vývojové prostředí Quartus. Pokud instalace ovladače proběhne v pořádku, zobrazí se ve správci zařízení dvě nová zařízení s názvem Altera USB-Blaster II (JTAG interface), pomocí kterého je možné nahrávání konfiguračních obrazů do hradlového pole a Altera USB-Blaster II (System Console interface), pomocí kterého je možné vyčítání nebo zapisování obsahu pamětí, registrů apod..

12.5 Blok QSPI

Tento blok symbolizuje externí flash paměť, která v tomto případě slouží k uložení programu procesoru NIOS II. Externí paměť je použita v případě, kdy svojí velikostí pro uložení programu nepostačuje vnitřní paměť (CFM) hradlového pole.

Pomocí funkce alt_load(), jsou z QSPI paměti nakopírovány datové sekce, jako jsou .rodata, .rwdata nebo .exceptions do paměti OCR. Jakmile je kopírování dokončeno, procesor začne vykonávat svoji funkci. Datová sekce .text se nekopíruje (důvodem je již zmíněná velikost) a zůstává v QSPI paměti pouze pro čtení.

12.5.1 Praktická realizace

Na vývojovém kitu je externí konfigurační paměť realizována pamětí s výrobním číslem N25Q512A83GSF40F od výrobce Micron [8]. Jedná se o paměť o velikosti 512 Mbit s adresním rozsahem od 0x0000 0000 do 0x03FF FFFF. Jelikož jsou ve vnitřní flash paměti (OCF) uloženy dva konfigurační obrazy, je nutné mít v paměti QSPI uloženy dva programy pro procesor NIOS II. Z toho důvodu je paměť rozdělena na dvě části. První část začíná adresou 0x0000 0000 (obsahuje program pro NIOS II, uložený v CFM0 – referenční konfigurace) a druhá část začíná adresou 0x0200 0000 (obsahuje program pro NIOS II, uložený v CFM1 a CFM2 – aplikační konfigurace).

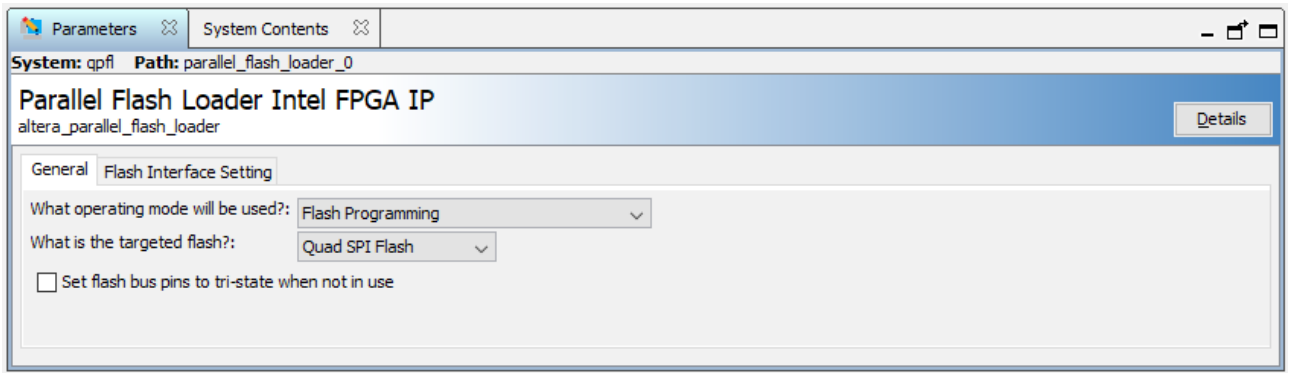
Paměť je k hradlovému poli připojena rozhraním QSPI (Quad SPI). Názvy všech signálů,

jejich parametry, popis a čísla pinů pro připojení k hradlovému jsou shrnuty v tabulce na obr. 12.4.

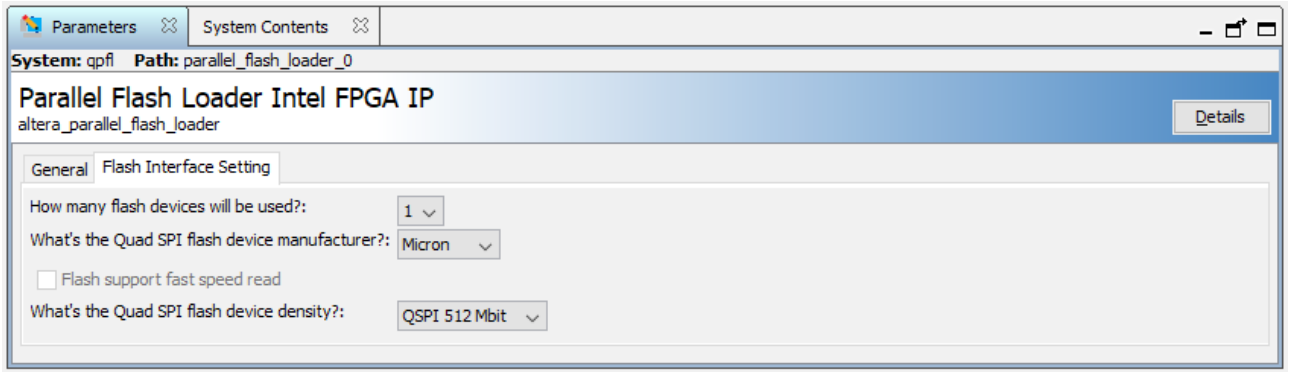
Název signálu vrcholové entity	Napěťová hladina	Číslo pinu na FPGA	Popis
quad_spi_flash_ncs_conduit_ncs[0]	3,3V LVCMOS	C2	Chip select
quad_spi_flash_dclk_out_conduit_dclk_out	3,3V LVCMOS	B2	Clock
quad_spi_flash_dataout_conduit_dataout[0]	3,3V LVCMOS	C6	Adress bus
quad_spi_flash_dataout_conduit_dataout[1]	3,3V LVCMOS	C3	Adress bus
quad_spi_flash_dataout_conduit_dataout[2]	3,3V LVCMOS	C5	Adress bus
quad_spi_flash_dataout_conduit_dataout[3]	3,3V LVCMOS	B1	Adress bus

Obr. 12.4 Parametry a čísla signálů pro připojení bloku QSPI k hradlovému poli [7]

Program pro procesor je možné nahrát externím programátorem nebo pomocí rozhraní JTAG. Z důvodu, že není možné jednoduché vyjmutí paměti z vývojového kitu, bylo zvoleno nahrání programu do flash paměti pomocí rozhraní JTAG. K tomu je ale nutné vytvořit vhodný zavaděč. Pro tento účel je výhodné využít komponentu z IP katalogu v nástroji Platform Designer s názvem Parallel Flash Loader Intel FPGA IP [9]. Komponenta je vytvořena přímo pro tyto účely a nevyžaduje připojení dalších komponent, jako je např. zdroj hodinového signálu, procesor apod.. Jediné, co je potřeba, je nastavení pracovního režimu na „Flash programing“ a typ komunikace na „Quad SPI Flash“ - viz obr. 12.5. Dále je nutné nastavit na záložce „Flash Interface Setting“ počet pamětí, které mají být inicializovány (v tomto případě se jedná pouze o jednu paměť), výrobce a velikost paměti – viz Obr 12.6.



Obr. 12.5 Nastavení parametrů komponenty Parallel Flash Loader - General Setting



Obr. 12.6 Nastavení parametrů komponenty Parallel Flash Loader - Flash Interface Setting

12.6 Bloky LEDS a Switches

Bloky LEDS a Switches nemají na funkci Factory upload a Remote update žádný vliv. Tyto bloky slouží pouze k indikaci přepnutí na daný konfigurační obraz, který je uložen v části paměti CFM0 resp. CFM1 a CFM2. Bloky dále slouží k indikaci úspěšného vzdáleného přehrání aplikačního obrazu, který je uložen v částech OCF paměti CFM1 a CFM2.

12.6.1 Praktická realizace

Na vývojovém kitu je blok LEDs realizován čtveřicí LED diod a blok Switches čtveřicí spínačů. LED diody se rozsvicují a zhasínají v závislosti na stisku spínačů a logice, která je znázorněna v architektuře systému jako blok User logic. Názvy všech signálů, jejich parametry, popis a čísla pinů pro připojení k hradlovému jsou shrnuty v tabulce na obr. 12.7.

Název signálu vrcholové entity	Napěťová hladina	Číslo pinu na FPGA	Popis
PB0	1,5V	L22	Button 0
PB1	1,5V	M21	Button 1
PB2	1,5V	M22	Button 2
PB3	1,5V	N21	Button 3
D0	1,5V	T20	LED Diode 0
D1	1,5V	U22	LED Diode 1
D2	1,5V	U21	LED Diode 2
D3	1,5V	AA21	LED Diode 3

Obr. 12.7 Parametry a čísla signálů pro připojení bloků LEDS a Switches k hradlovému poli [7]

12.7 Blok PLL

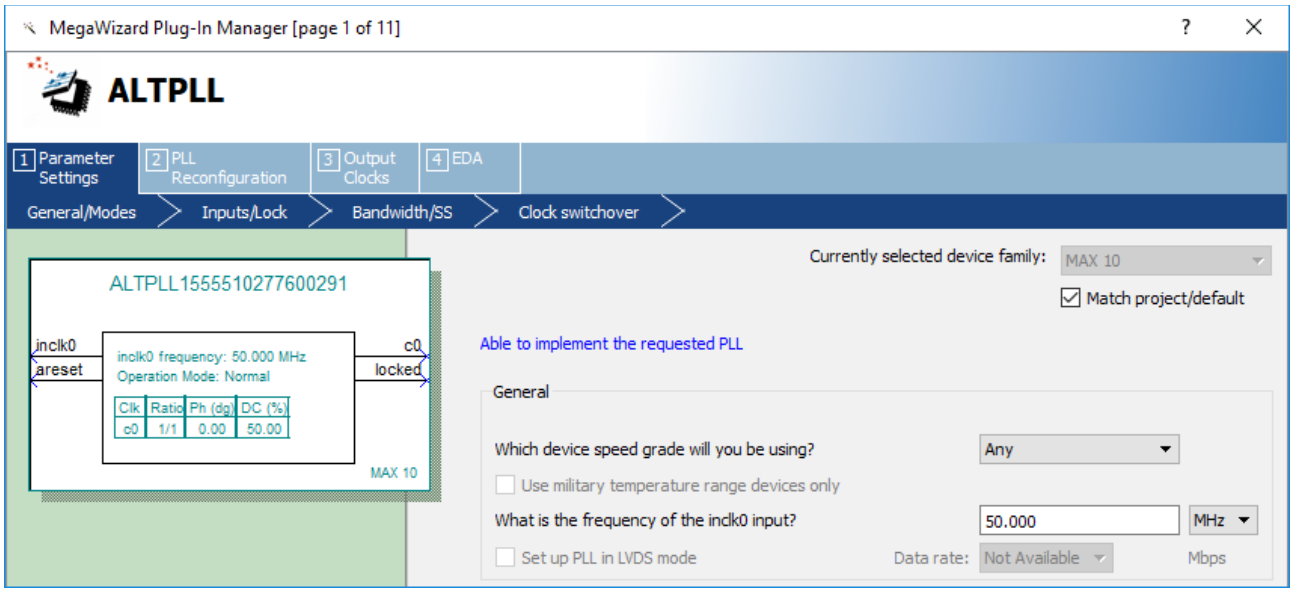
Blok PLL symbolizuje v architektuře systému fázový závěs. Fázový závěs v tomto systému slouží k úpravě frekvence hodinového signálu, který je dále rozveden do ostatních bloků. Ze vstupní frekvence 50 MHz je fázovým závěsem produkována výstupní frekvence 50 MHz a 15 MHz.

Hodinový signál o frekvenci 50 MHz je přiveden do bloků Dual boot, UART, JTAG, Onchip Memory, Onchip Flash a NIOS 2. Tyto bloky jsou schopny pracovat s vyšší hodinovou frekvencí, ale pro účely této práce je frekvence 50 MHz zcela postačující.

Hodinový signál o frekvenci 15 MHz je přiveden pouze do bloku QUAD SPI. Důvodem je, že pro tento blok je doporučena maximální frekvence hodinového signálu 25 MHz [10] (z důvodu určité rezervy bylo zvoleno 15 MHz).

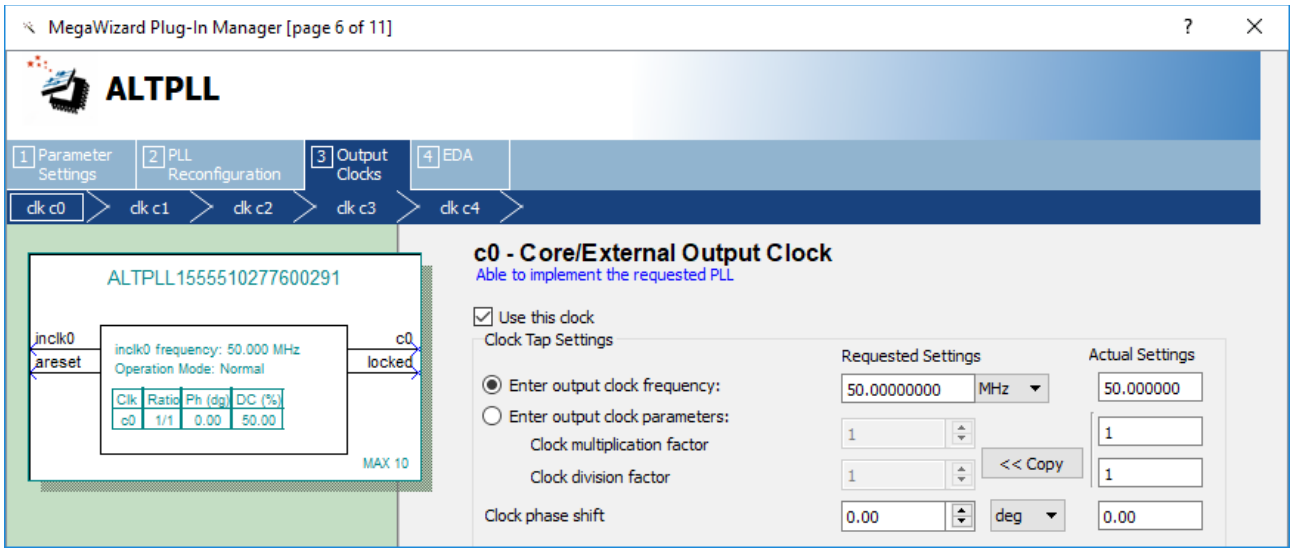
12.7.1 Praktická realizace

Blok PLL je realizován komponentou z IP katalogu nástroje Platform Designer s názvem ALTPLL Intel FPGA IP [11]. Pro správnou funkci je nutné nastavit všechny potřebné parametry. Nejdříve je nutné na záložce „Parameter Settings“ nastavit vstupní frekvenci do fázového závěsu na hodnotu 50 MHz. To je znázorněno na obr. 12.8.



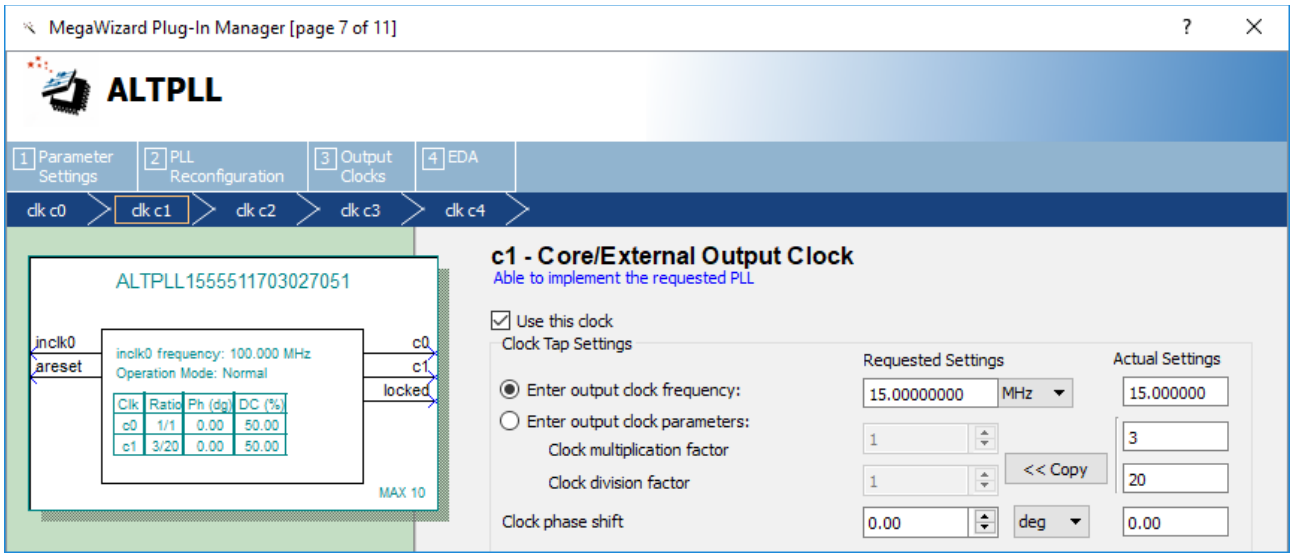
Obr. 12.8 Nastavení parametrů komponenty ALTPLL Intel FPGA IP – Parameter Settings

Dále je nutné na záložce „Output Clocks“ nadefinovat pro výstup c0 výstupní frekvenci fázového závěsu na frekvenci 50 MHz. Toho je dosaženo přepsáním parametru „Enter output clock freguency“ na 50 MHz tak, jak je na obr. 12.9.



Obr. 12.9 Nastavení parametrů komponenty ALTPLL Intel FPGA IP – Output Clocks c0

To samé je nutné provést i pro výstup c1 s tím rozdílem, že výstupní frekvence fázového závěsu je v tomto případě 15 MHz. Toho je dosaženo opět přepsáním parametru „Enter output clock frequency“ tak, jak je na obr. 12.10. Všechny ostatní parametry fázového závěsu pro účely této práce není nutné nijak dále editovat.



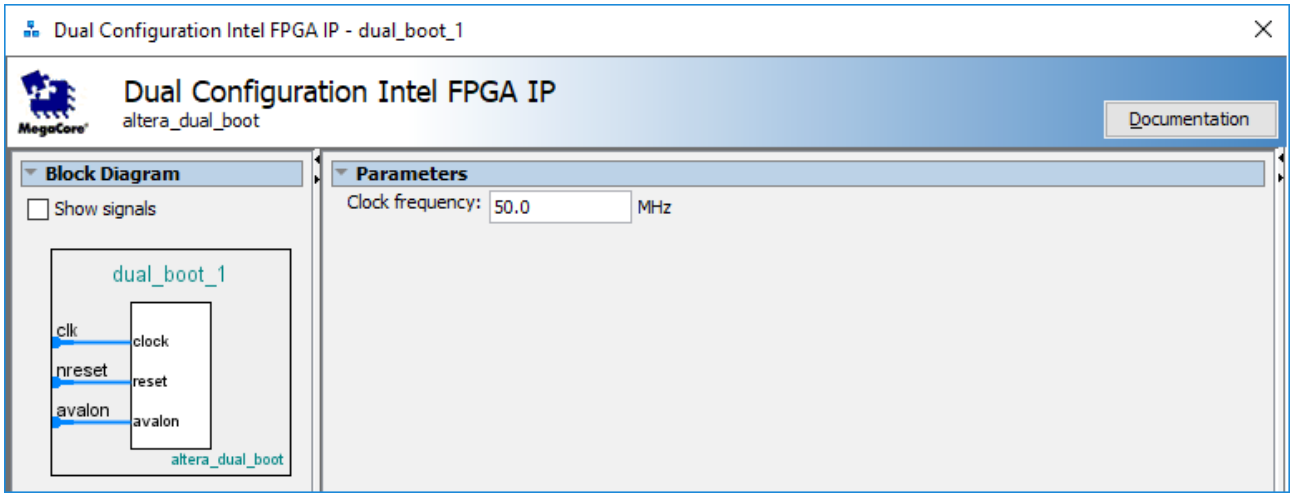
Obr. 12.10 Nastavení parametrů komponenty ALTPLL Intel FPGA IP – Output Clocks c1

12.8 Blok Dual boot

Blok Dual boot umožňuje přepínání mezi obrazy aplikační konfigurace a referenční konfigurace na povel, který vysílá zařízení MASTER prostřednictvím rozhraní UART. Blok také obsahuje funkci User Watchdog Timer, která detekuje chyby při načítání obrazu aplikační resp. referenční konfigurace. Jedná se o časový čítač, který v případě zaznamenání chyby v načítání obrazu aplikační konfigurace začne čítat a po přetečení přepne na referenční konfiguraci, pomocí které je následně možné poškozenou aplikační konfiguraci přehrát. Tento přístup funguje i opačným směrem (v případě zaznamenání chyby v obrazu referenční konfigurace proběhne pokus o přepnutí na aplikační konfiguraci), ale obvykle funkce User Watchdog Timer zaznamená chybu pouze v obrazu aplikační konfigurace, který je uložen v částech pamětí CFM1 a CFM2. Tato chyba bude vždy s největší pravděpodobností způsobena nějakým rušivým elementem při vzdáleném přehrávání aplikačního obrazu.

12.8.1 Praktická realizace

Blok Dual boot je realizován komponentou z IP katalogu nástroje Platform Designer s názvem Dual Configuration Intel FPGA IP [12]. Pro správnou funkci je nutné nastavit pouze hodnotu vstupního hodinového kmitočtu na frekvenci 50 MHz tak, jak je znázorněno na obr. 12.11.



Obr. 12.11 Nastavení hodinového kmitočtu komponenty Dual Configuration Intel FPGA IP

Obsluha přepínání konfigurací se provádí zápisem do registru komponenty Dual Configuration Intel FPGA IP. Do registru se zapisují hodnoty v závislosti na tom, na jaký konkrétní konfigurační obraz je požadavek ze strany zařízení MASTER přepnout. Popis registru je v tabulce na obr. 12.12. Zápisem logické hodnoty „1“ do registru je aktivována funkce uvedená v tabulce ve sloupci popis. Komponenta poskytuje mnohem více možností, které ale vzhledem k použití v této aplikaci není v tabulce nutné uvádět.

Offset	R/W	Velikost [Bit]	Popis
0	W	32	Bit 0 – Přepnutí konfigurace
1	W	32	Bit 0 – Nastaví registr na přepsání Bit 1 – Zapíše do registru 0 (pro přepnutí na obraz aplikační konfigurace) nebo 1 (pro přepnutí na obraz referenční konfigurace)

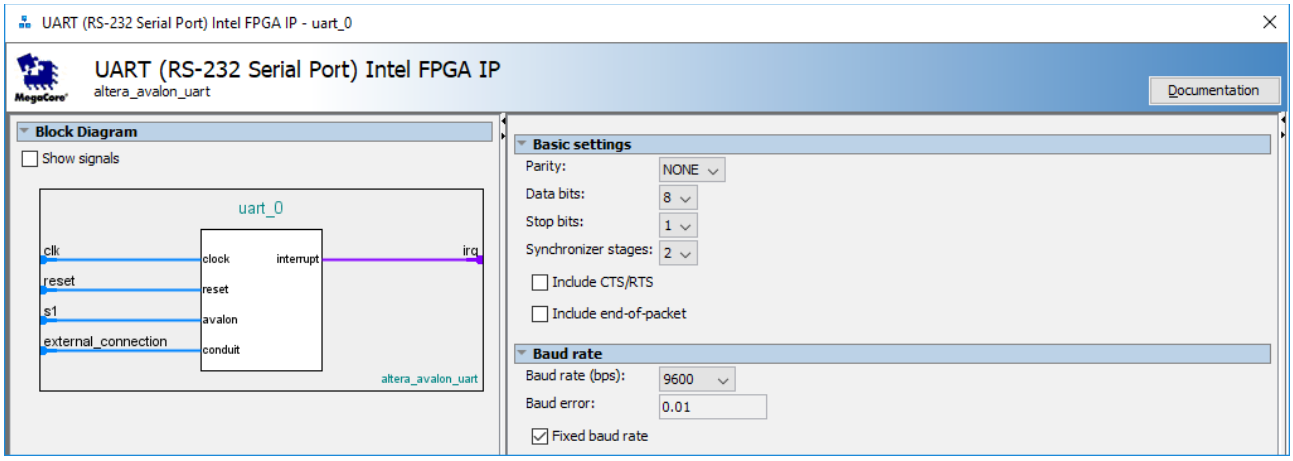
Obr. 12.12 Nastavení registru komponenty Dual Configuration Intel FPGA IP [12]

12.9 Blok UART

Díky tomuto bloku je umožněna obousměrná datová komunikace mezi převodníkem USB/UART, který je popsán v kapitole 12.3 a mezi procesorem NIOS II, ke kterému jsou připojeny další periférie systému.

12.9.1 Praktická realizace

Blok UART je realizován komponentou z IP katalogu nástroje Platform Designer s názvem UART (RS-232 Serial Port) Intel FPGA IP. Pro správnou funkci je nutné nastavit pouze komunikační rychlost, která byla zvolena na 9600 bit/s. Nastavení komunikační rychlosti komponenty je na obr. 12.13. Všechny ostatní parametry komponenty není nutné dále editovat.



Obr. 12.13 Nastavení komunikační rychlosti komponenty UART (RS-232 Serial Port) Intel FPGA IP

12.10 Blok JTAG

Tento blok umožňuje přistupovat přes rozhraní JTAG k registrům procesoru NIOS a k registrům všech komponent, které jsou k rozhraní JTAG připojeny. Tyto registry je možné vyčítat nebo do nich přímo zapisovat, což je velmi výhodné při vývoji nebo diagnostice zařízení. K těmto účelům slouží nástroj ve vývojovém prostředí Quartus 18.0 s názvem System Console [13].

12.10.1 Praktická realizace

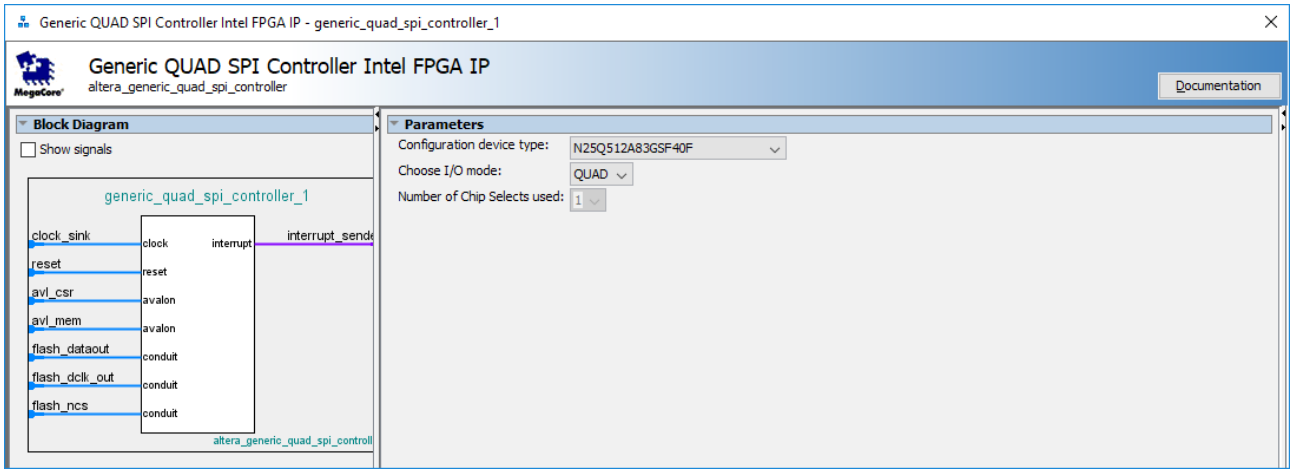
Blok JTAG je realizován komponentou z IP katalogu nástroje Platform Designer s názvem JTAG to Avalon Master Bridge. Pro správnou funkci komponenty v systému není nutné nastavovat žádné parametry.

12.11 Blok QUAD SPI

Jak již bylo řečeno výše, pokud svojí velikostí pro uložení programu pro procesor NIOS II nepostačuje vnitřní OCF paměť hradlového pole, lze použít externí QSPI flash paměť, která je součástí vybraného vývojového kitu – viz kapitola 12.5. Blok QUAD SPI slouží jako rozhraní mezi externí pamětí a procesorem NIOS II, který je v hradlovém poli integrován.

12.11.1 Praktická realizace

Blok QUAD SPI je realizován komponentou z IP katalogu nástroje Platform Designer s názvem Generic QUAD SPI Controller Intel FPGA IP [14]. Pro správnou funkci komponenty v systému je nutné vybrat konkrétní paměť, která je osazena na vývojovém kitu. Jak bylo uvedeno v kapitole 12.5, na vývojovém kitu je osazena paměť s označením N25Q512A83GSF40F od výrobce Micron. Na obr. 12.14 je nastavení komponenty s názvem Generic QUAD SPI Controller Intel FPGA IP, která poté slouží jako rozhraní mezi pamětí N25Q512A83GSF40F a procesorem NIOS II.



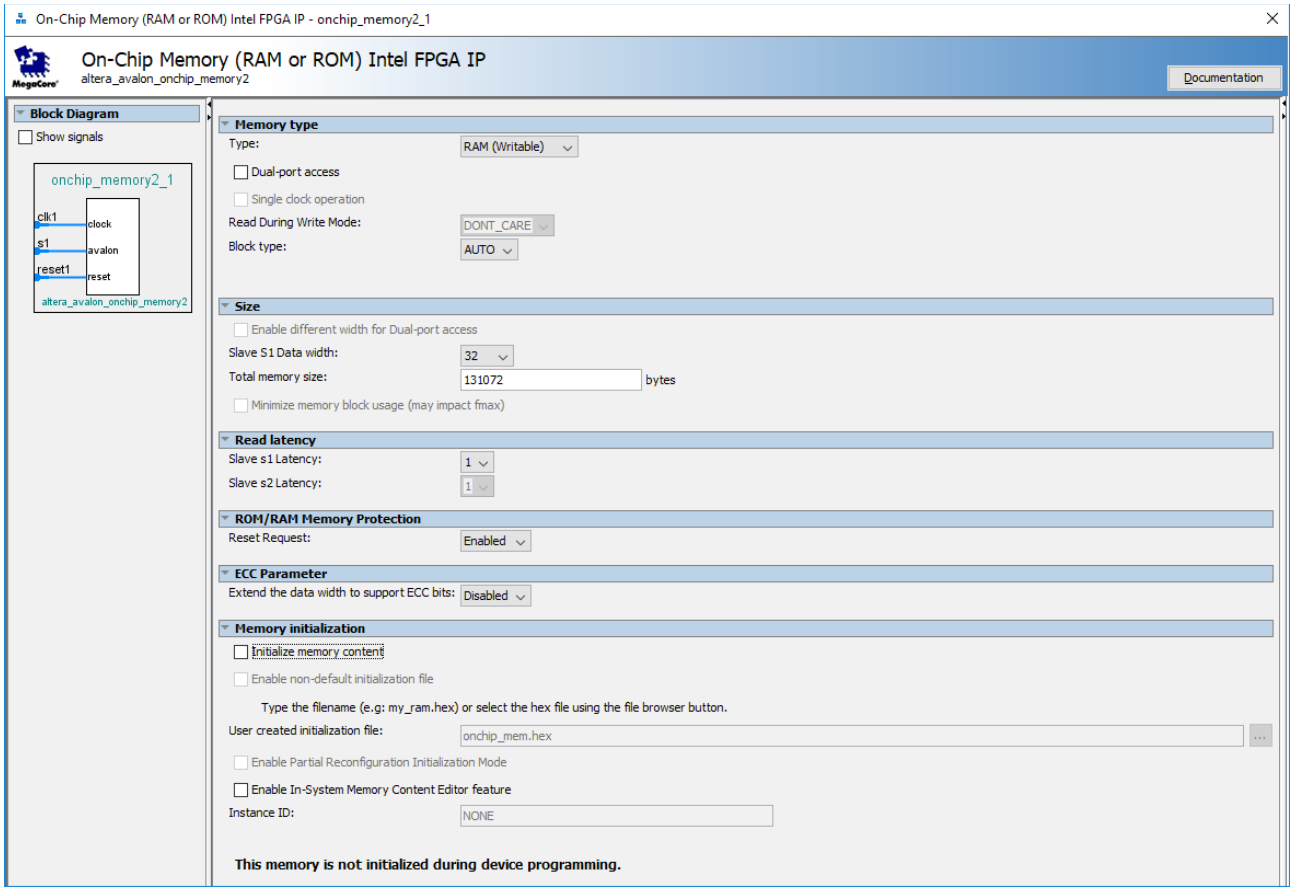
Obr. 12.14 Nastavení komponenty Generic QUAD SPI Controller Intel FPGA IP

12.12 Blok Onchip Memory

Blok znázorňuje paměť typu RAM, kterou ke své činnosti využívá procesor NIOS II. Do této paměti jsou pomocí funkce alt_load() nakopírovány datové sekce z QSPI paměti, jako jsou .rodata, .rwdata nebo .exceptions. Jakmile je kopírování úspěšně dokončeno, procesor začne vykonávat svoji funkci podle daného programu. Datová sekce .text se nekopíruje a zůstává v QSPI paměti pouze pro čtení.

12.12.1 Praktická realizace

Blok Onchip Memory je realizován komponentoru z IP katalogu nástroje Platform Designer s názvem On-Chip Memory (RAM or ROM) Intel FPGA IP [15]. Pro správnou funkci komponenty v systému byla zvolena a nastavena velikost paměti na 131072 bytů (128 kbitů) a byla vypnuta možnost inicializace obsahu paměti (Initialize memory content). Důvodem vypnutí inicializace paměti je ten, že zdrojový kód pro procesor NIOS není uložen v této paměti, ale je uložen v paměti QSPI. Nastavení komponenty je znázorněno na obr. 12.15.



Obr. 12.15 Nastavení komponentyn On-Chip Memory (RAM or ROM) Intel FPGA IP

12.13 Blok Onchip Flash

Blok Onchip Flash v architektuře znázorňuje vnitřní flash paměť hradlového pole MAX10. Pro použití v této konkrétní aplikaci jsou v paměti uloženy dva konfigurační obrazy (referenční a aplikační). Aplikační obraz je možné pomocí referenční aplikace a prostřednictvím zařízení MASTER přes rozhraním UART vzdáleně přehrát. V paměti lze mít také uložen program pro procesor NIOS II, nicméně pro uložení programu pro NIOS II není tato paměť použita a je použita externí QSPI paměť. Důvodem je velikost tohoto programu, který svojí velikostí přesahuje možnosti použití interní flash paměti hradlového pole.

12.13.1 Praktická realizace

V tabulce na obrázku 12.16 je znázorněno vnitřní uspořádání paměti. Jak je vidět, paměť je rozdělena celkem na 5 sektorů. Sektory s označením 1 a 2, které obvykle slouží pro uložení programu pro procesor NIOS II nejsou použity. Důvodem je, že program procesoru NIOS II je uložen v paměti QSPI. Vysvětlení viz výše.

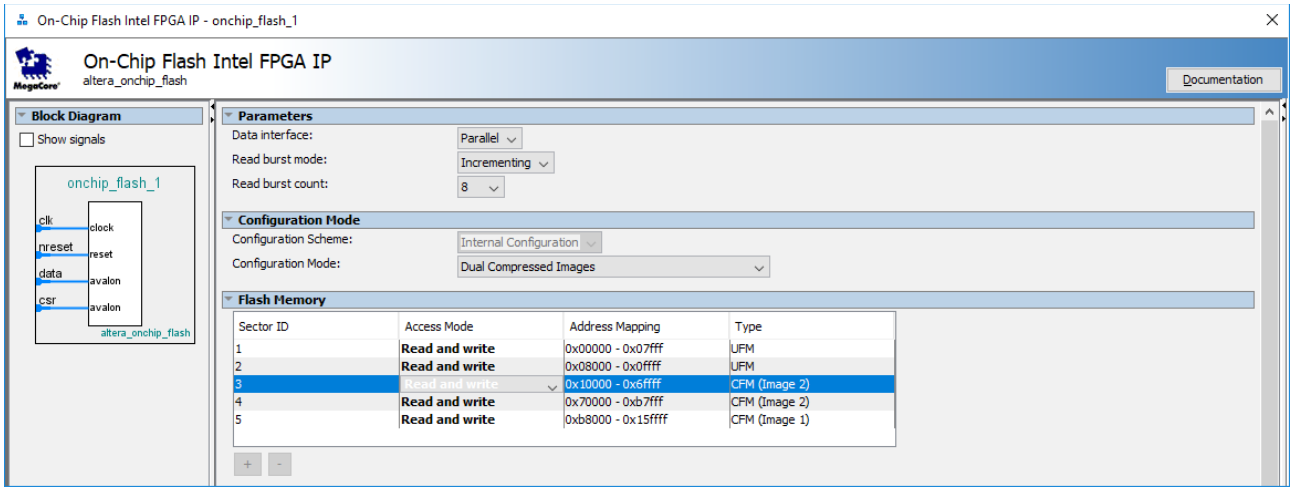
Sektory 3(CFM2) a 4(CFM1) slouží k uložení obrazu aplikační konfigurace. Tato konfigurace je umístěna v části paměti začínající adresou 0x10000 a končící adresou 0xB7FFF. Tuto část paměti je pomocí referenční konfigurace možné vzdáleně přehrát, resp. lze přehrát stávající aplikační konfigurace za novou aplikační konfiguraci.

Sektor 5(CFM0) slouží k uložení obrazu referenční konfigurace. Tato konfigurace je uložena v části paměti začínající adresou 0xB8000 a končící adresou 0x15FFFF. Pomocí referenční konfigurace je možné vzdáleně přehrát stávající aplikační konfiguraci, která se nachází v sektoru paměti 3 a 4.

Sektor	Adresa	Typ	Popis
1	0x00000 - 0x07FFF	UFM	Nepoužito
2	0x08000 - 0x0FFFF	UFM	Nepoužito
3	0x10000 - 0x6FFFF	CFM2	Obraz aplikační konfigurace
4	0x70000 - 0xB7FFF	CFM1	Obraz aplikační konfigurace
5	0xB8000 - 0x15FFFF	CFM0	Obraz referenční konfigurace

Obr. 12.16 Vnitřní uspořádání paměti On-Chip Flash

Blok Onchip Flash je realizován komponentou z IP katalogu nástroje Platform Designer s názvem On-Chip Flash Intel FPGA IP [16]. Pro správnou funkci komponenty v systému je nutné nastavit, že bude paměť provozována v duálním režimu, který umožňuje nahrání do paměti dvou konfiguračních obrazů. Toto nastavení je provedeno vybráním možnosti na rozbalovací kartě „Dual Compressed Images“ u parametru „Configuration Mode“. Dále je nutné u všech sektorů nastavit režim přístupu na čtení a zápis. Toto nastavení je provedeno vybráním možnosti „Read and write“ na rozbalovací kartě ve sloupci „Access Mode“. Ostatní parametry komponenty není nutné dále editovat. Nastavení komponenty je znázorněno na obr. 12.17.



Obr. 12.17 Nastavení komponenty On-Chip Flash Intel FPGA IP

Obsluha OCF paměti je prováděna zápisem do registru s názvem „Control Register“ komponenty On-Chip Flash Intel FPGA IP. Nastavení registru komponenty, vzhledem ke konkrétnímu použití v této aplikaci je shrnuto v tabulce na obr. 12.18. Registr se nachází na adrese 0x01 a jeho výchozí nastavení je 0xFFFF FFFF. Komponenta poskytuje mnohem více možností, které ale vzhledem k použití v této aplikaci není v tabulce nutné uvádět.

Offset	Výchozí nastavení	Velikost [Bit]	Popis
22-20	W	32	Nastavením '011' je vymazán sektor 3 (CFM2)
22-20	W	32	Nastavením '100' je vymazán sektor 4 (CFM1)
25	W	32	Nastavením '0' je odstraněna ochrana proti zápisu u sektoru 3 (CFM2)
26	W	32	Nastavením '0' je odstraněna ochrana proti zápisu u sektoru 4 (CFM1)

Obr. 12.18 Nastavení registru komponenty On-Chip Flash Intel FPGA IP [16]

12.14 Blok User logic

Tento blok slouží k indikaci přepnutí konfigurace na aplikační konfigurační obraz, resp. na referenční konfigurační obraz. V praxi tento blok bude obsahovat potřebné logické funkce, pro konkrétní aplikaci v praxi.

12.14.1 Praktická realizace

Blok obsahuje velmi jednoduché logické funkce, které rozsvěcují a zhasínají LED diody bloku LEDS, v závislosti na stisknutí tlačítek bloku Switches. Díky tomu je velice jednoduché rozeznat, jaká konfigurace je v daný okamžik používána.

Na obrázku 12.19 je pravdivostní tabulka, popisující funkci, která pomocí rozsvěcování a zhasínání LED diod v závislosti na stisku spínačů indikuje přepnutí na referenční konfiguraci. Hodnota v tabulce u LED diody „1“ znamená, že svítí a „0“ nesvítí. Hodnota v tabulce u spínače „0“ znamená, že je tlačítko stisknuté a „1“ nestisknuté.

Referenční konfigurace							
PB0	**D0**	**PB1**	**D1**	**PB2**	**D2**	**PB3**	**D3**
0	0	0	1	0	0	0	1
1	1	1	0	1	1	1	0

Obr. 12.19 Pravdivostní tabulka funkce indikující přepnutí na referenční konfiguraci (Dx – LED, PBx - Spínač)

Na obrázku 12.20 je pravdivostní tabulka popisující funkci, která pomocí rozsvěcování a

zhasínání LED diod v závislosti na stisku spínačů indikuje přepnutí na aplikační konfiguraci 1 (aplikační konfigurace je nahrána do části paměti CFM1 a CFM2 pomocí funkce Factory upload). Hodnota v tabulce u LED diody „1“ znamená, že svítí a „0“ nesvítí. Hodnota v tabulce u spínače „0“ znamená, že je tlačítko stisknuté a „1“ nestisknuté.

Aplikační konfigurace 1							
PB0	**D0**	**PB1**	**D1**	**PB2**	**D2**	**PB3**	**D3**
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0

Obr. 12.20 Pravdivostní tabulka funkce indikující přepnutí na aplikační konfiguraci 1 (Dx – LED, PBx - Spínač)

Na obrázku 12.21 je pravdivostní tabulka popisující funkci, která pomocí rozsvěcování a zhasínání LED diod v závislosti na stisku spínačů indikuje přepnutí na aplikační konfiguraci 2 (aplikační konfigurace je nahrána do části paměti CFM1 a CFM2 pomocí funkce Remote update). Hodnota v tabulce u LED diody „1“ znamená, že svítí a „0“ nesvítí. Hodnota v tabulce u spínače „0“ znamená, že je tlačítko stisknuté a „1“ nestisknuté.

Aplikační konfigurace 2							
PB0	**D0**	**PB1**	**D1**	**PB2**	**D2**	**PB3**	**D3**
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1

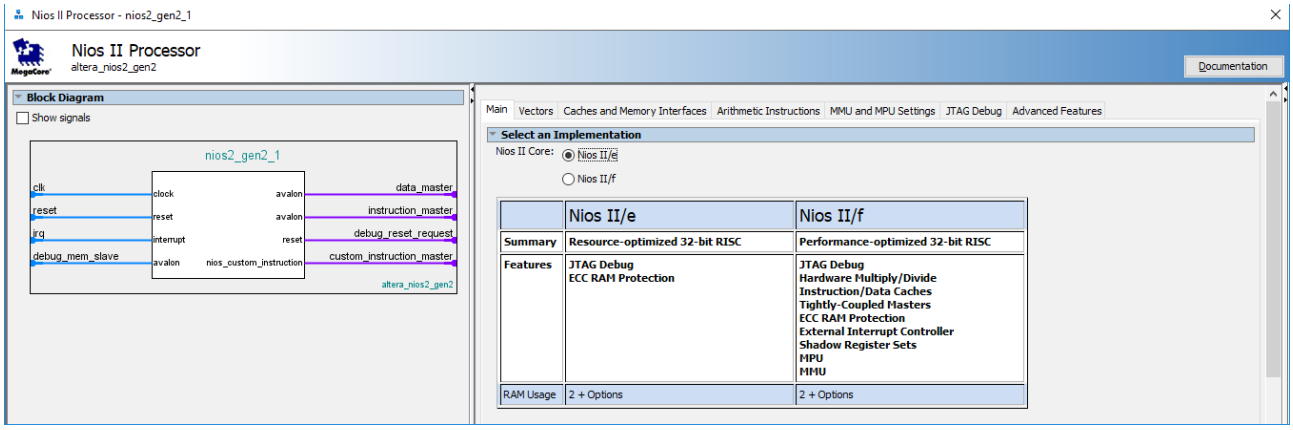
Obr. 12.21 Pravdivostní tabulka funkce indikující přepnutí na aplikační konfiguraci 2 (Dx – LED, PBx – Spínač)

12.15 Blok NIOS 2

Procesor slouží k obsluze použitých periférií z IP katalogu nástroje Platform Designer, které jsou v tomto systému použity. Jedná se o procesor, který byl navržen pro integrování do hradlového pole.

12.15.1 Praktická realizace

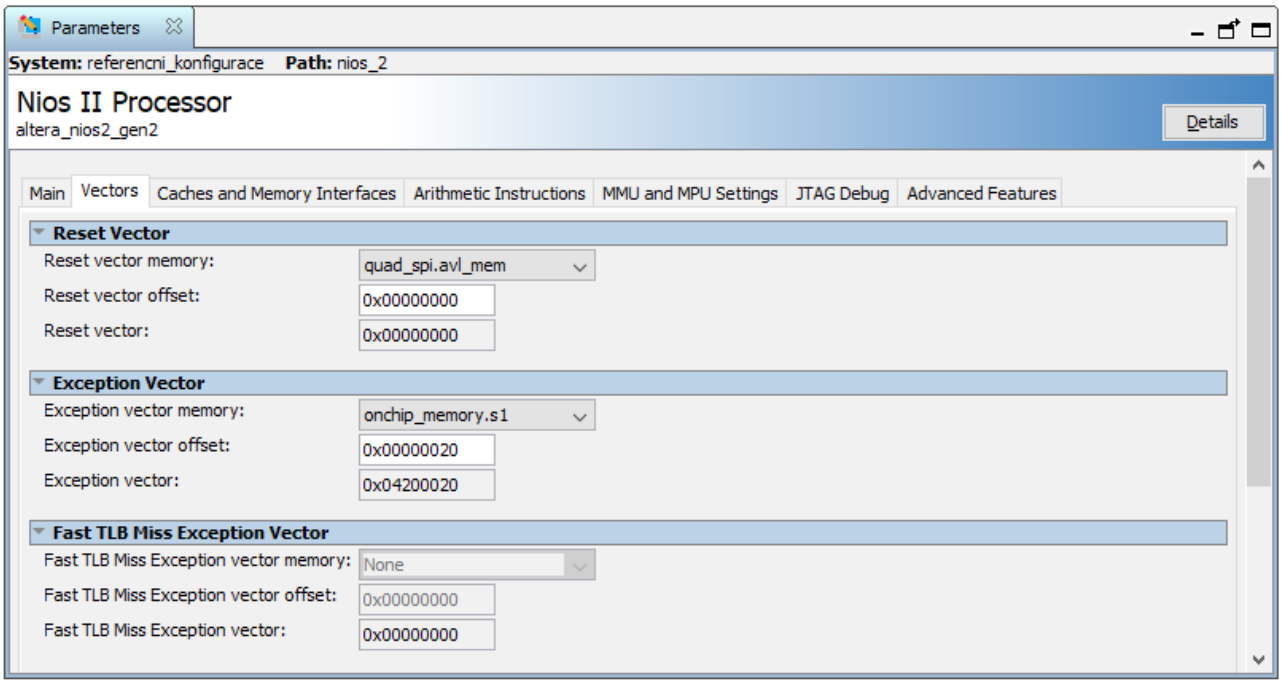
Blok NIOS 2 je realizován komponentou z IP katalogu nástroje Platform Designer s názvem Nios II Processor [17]. Pro správné nastavení komponenty byla použita verze procesoru „e“, která není nijak zpoplatněna a pro účely názorné ukázky vzdáleného přehrání zcela postačuje. Nastavení verze „e“ je znázorněno na obr. 12.22.



Obr. 12.22 Nastavení komponenty NIOS II Processor – verze „e“

Dále byl použit způsob pro bootování popsaný v [10], který se v tomto dokumentu označuje jako Boot Option 4a. Způsob bootování zde uvedený spočívá v nastavení parametrů v záložce „Vectors“, kde je nutné nastavit umístění resetovacího vektoru na QSPI paměť a vektoru vyjímky na OCR paměť. Reset vektoru udává místo, kde se v paměti vyskytuje tzv. bootloader a vektor vyjímky udává místo, kde se nachází tzv. vektor vyjímek.

Nastavení dané paměti a jejích vektorů je možné až po umístění komponenty do systému. Reset vektor pro referenční konfiguraci programu pro procesor NIOS II se nachází na adrese 0x0000 0000 QSPI paměti – viz obr. 12.23. Reset vektor pro aplikační konfiguraci číslo 1 a aplikační konfiguraci číslo 2 programu procesoru NIOS II se nachází na adrese 0x0200 0000 QSPI paměti. Vektor vyjímek je v případě referenční konfigurace, aplikační konfigurace 1 a aplikační konfigurace 2 na adrese 0x0000 0020 OCR paměti.



Obr. 12.23 Nastavení vektorů komponenty NIOS II Processor pro referenční konfiguraci

13 Praktická realizace systému

V této kapitole je popsán postup, jakým byly vytvořeny obrazy referenční konfigurace, aplikační konfigurace číslo 1 a aplikační konfigurace číslo 2. V popisu u výše uvedených konfigurací je také uveden postup pro vytvoření programů procesoru NIOS II. V závěru této kapitoly je popsáno generování finálních souborů a jejich následné nahrání do hradlového pole prostřednictvím rozhraní JTAG (funkce Factory upload).

13.1 Návrh referenční konfigurace

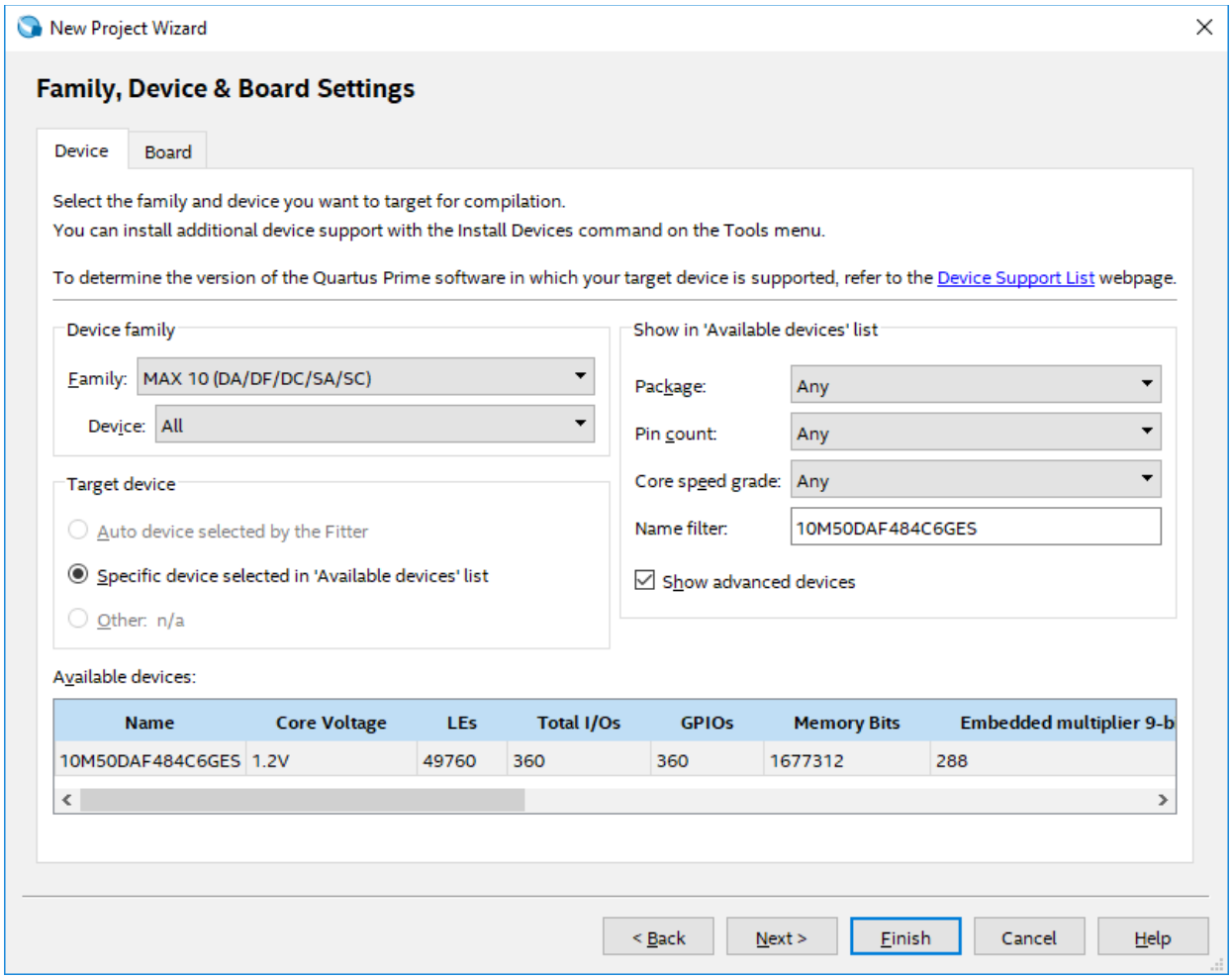
Referenční konfigurace v systému zajišťuje přehrání aplikační konfigurace prostřednictvím rozhraní UART. Potřeba přehrání aplikační konfigurace může mít důvody jako jsou doplnění nové funkcionality, objevení chyby při vývoji apod.. Referenční konfigurace také umožňuje na povel zařízení MASTER přepnout na aplikační konfiguraci (a naopak). Referenční konfigurace je do zařízení nahrána při výrobě a na rozdíl od aplikační konfigurace ji není možné prostřednictvím UART vzdáleně přehrát.

13.1.1 Založení nového projektu v prostředí Quartus

Pro založení nového projektu ve vývojovém prostředí Quartus je postupováno následujícím způsobem.

1) Kliknutím na tlačítko „New Project Wizard“, ve vývojovém prostředí Quartus, je spuštěn průvodce vytvořením nového projektu.
2) Kliknutím na tlačítko „Next“ je zobrazen formulář, kde je nutné nastavit název projektu a adresář, ve kterém bude projekt uložen. Název tohoto projektu byl zvolen jako „refererencni_konfigurace“ a je uložen v adresáři se shodným názvem.
3) Kliknutím na „Next“ a následným vybráním možnosti „Empty project“, je zvolen nový prázdný projekt.
4) Kliknutím na tlačítko „Next“ je zobrazen formulář, který slouží k přidání dalších souborů projektu. Jelikož zatím žádné takové soubory nejsou k dispozici, je tento krok přeskočen.
5) Kliknutím na tlačítko „Next“ je zobrazen formulář, který slouží k vybrání konkrétního hradlového pole, které je pro projekt použito. V rozbalovací kartě s názvem „Family“ je nutné vybrat hradlové pole řady MAX 10 a vyplněním výrobního čísla „10M50DAF484C6GES“ do pole s názvem „Name filter“ je určeno konkrétní hradlové

pole, které je v projektu použito - viz obr. 13.1. Tím je dokončeno základní nastavení projektu a kliknutím na tlačítko „Finish“ lze průvodce vytvořením nového projektu zavřít.



Obr. 13.1 Nastavení nového projektu ve vývojovém prostředí Quartus

13.1.2 Realizace systému na základě architektury

Část architektury, která je realizována hradlovým polem je vytvořena nástrojem Platform Designer, který je součástí vývojového prostředí Quartus. Pro realizaci architektury nástrojem Platform Designer je postupováno následujícím způsobem.

1) Kliknutím tlačítko ve vývojovém prostředí Quartus je s názvem „Tools“ a následným vybráním možnosti „Platform Designer“ je spuštěn nástroj Platform Designer.
2) Pomocí IP katalogu jsou vyhledány a následně vloženy do projektu všechny komponenty, které jsou uvedeny v kapitole 12.
3) Všechny použité komponenty a jejich signály jsou pojmenovány pod shodným názvem, jakým jsou pojmenovány odpovídající bloky a názvy signálů v architektuře systému.

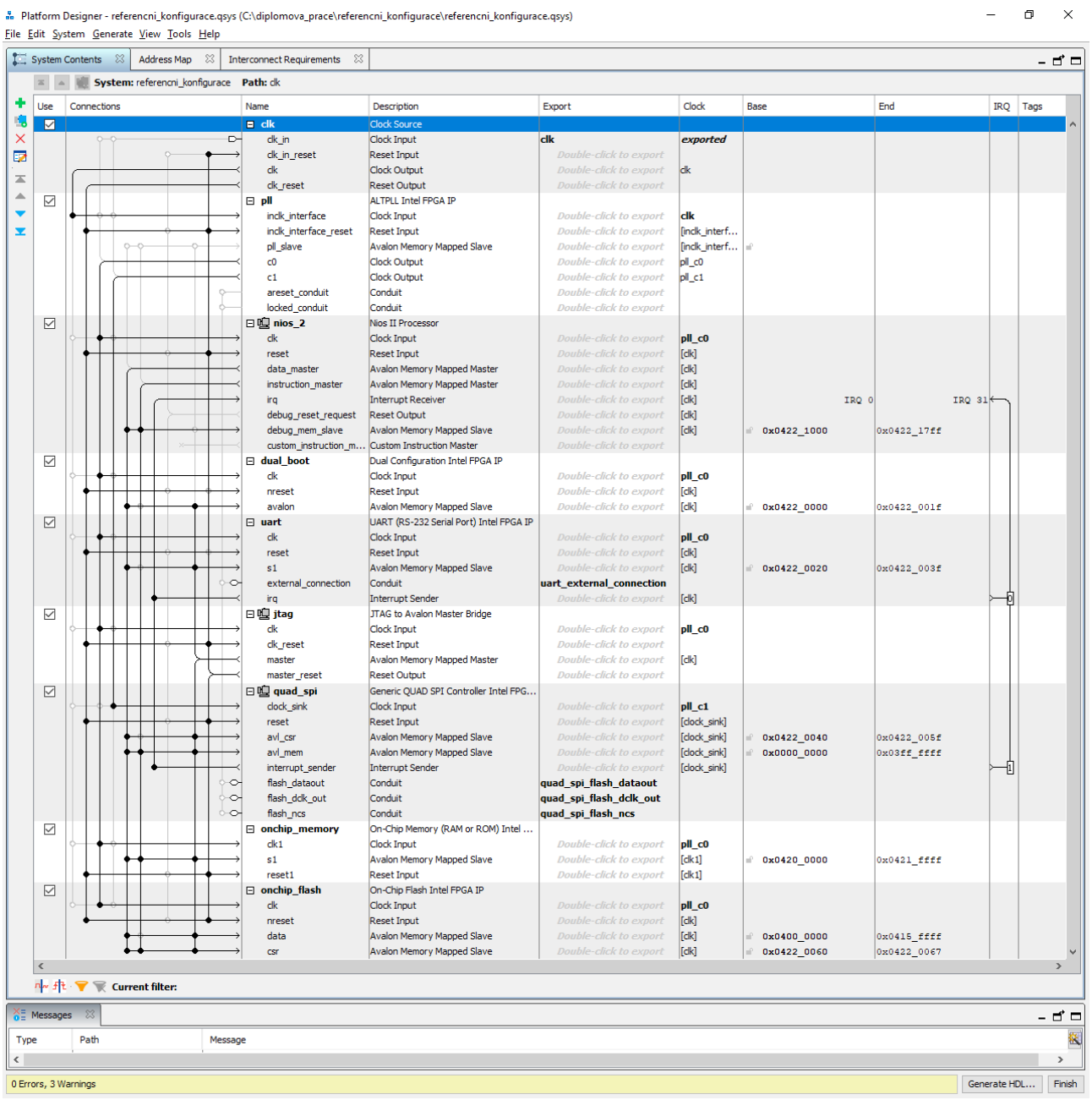
4) Všechny použité komponenty jsou mezi sebou vhodně propojeny pomocí signálů a datových sběrnic. Komponentám jsou nastaveny parametry, které jsou jsou vztaženy k odpovídajícím blokům architektury. Parametry jednotlivých komponent jsou uvedeny v části „Praktická realizace“ v kapitole 12.

5) Kliknutím na tlačítko „System“ a následně na „Assign Base Adresses“ je jednotlivým komponentám přiřazena adresa, která je v celém systému jedinečná.

6) Kliknutím na tlačítko „File“ a následně na „Save As“ je projekt v nástroji Platform Designer uložen pod názvem „referencni_konfigurace“ opět v adresáři se shodným názvem.

7) Kliknutím na tlačítko „Generate“ a následně na „Generate HDL“ je zobrazen formulář s nastavením generování výstupných souborů. Není potřeba žádného dalšího nastavení a kliknutím na tlačítko „Generate“, proběhne generování výstupních HDL souborů projektu.

Pokud generování výstupních souborů proběhlo bezchybně, je možné nástroj Platform Designer opustit. Schéma zapojení všech komponent pomocí datových sběrnic a signálů v tomto nástroji je znázorněno na obr. 13.2.

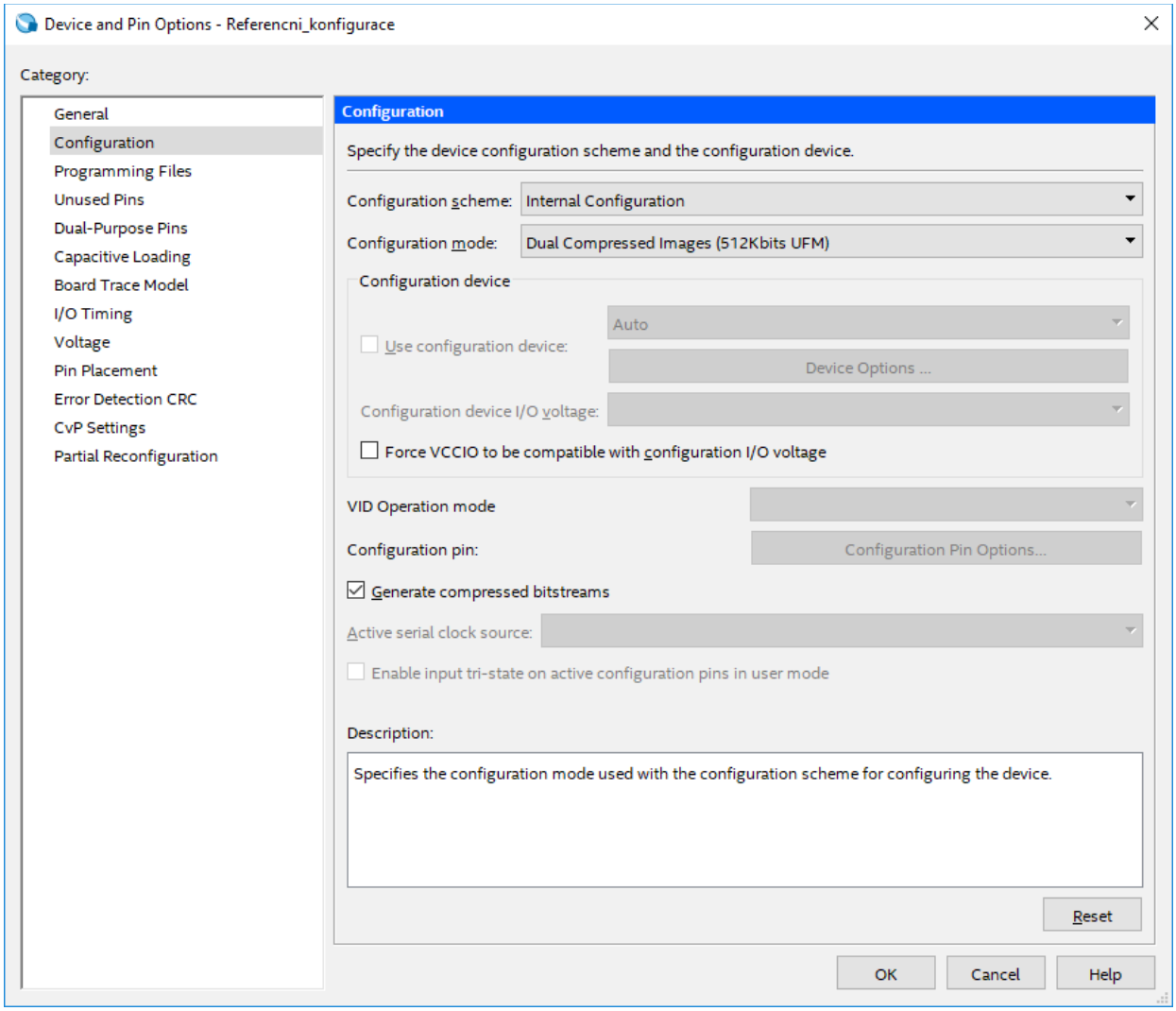


Obr. 13.2 Schéma zapojení všech komponent systému v nástroji Platform Designer

13.1.3 Návrh projektu v prostředí Quartus

Část architektury, vytvořená v nástroji Platform Designer byla úspěšně přeložena. Tím vznikly HDL soubory, které vycházejí z výše uvedené části architektury. Vzniklé soubory je nyní potřeba vložit do projektu ve vývojovém prostředí Quartus a spustit nad nimi analýzu a syntézu. Aby analýza a syntéza proběhla v bezchybně, je nutné nastavit další parametry projektu. Tato nastavení jsou závislá na architektuře systému. K výše uvedenému nastavení slouží následující postup.

1) Kliknutím na rozbalovací pole v okně „Project Navigator“ ve vývojovém prostředí Quartus a následným vybráním možnosti „Files“ jsou zobrazeny zdrojové soubory projektu. Jelikož zatím nebyly vloženy žádné soubory, bude okno prázdné.
2) Kliknutím na tlačítko „Assignments“ a následným vybráním možnosti „Settings“ je otevřen průvodce nastavením projektu.
3) V kategorii „Files“ otevřeného formuláře je potřeba vyplnit cestu k souboru s názvem „referencni_konfigurace.qip“. Tento soubor obsahuje odkazy na všechny HDL soubory, které byly vytvořeny nástrojem Platform Designer. Soubor se nachází ve složce s názvem „sysnthesis“, která se nachází ve složce s projektem, vytvořeného nástrojem Platform Designer.
4) Vybráním souboru a následným kliknutím na tlačítko „Open“ je soubor vložen do otevřeného formuláře. Následným kliknutím na tlačítko „OK“ jsou všechny HDL soubory vloženy do projektu ve vývojovém prostředí Quartus.
5) V okně s názvem „Project Navigator“ je nyní potřeba nastavit HDL soubor, který reprezentuje vrcholovou entitu. V tomto případě se jedná o soubor s názvem „referencni_konfigurace.v“. Kliknutím pravým tlačítkem myši nad tímto souborem a následným vybráním možnosti „Set as Top-Level Entity“ je daná entita určena jako vrcholová.
6) Kliknutím na tlačítko „Assignments“ a následně vybráním možnosti „Device“ je otevřen formulář, který informuje o tom, jaké hradlové pole je aktuálně v projektu používáno.
7) Kliknutím na tlačítko „Device and Pin Options“ je zobrazen formulář, kde lze nastavit režim, v jakém je hradlové pole používáno.
8) Jelikož byla zvolena architektura systému, která je založena na základě přepínání dvou konfiguračních obrazů, je potřeba v kategotii „Configuration“ v rozbalovacím poli s názvem „Configuration mode“ nastavit „Dual Compressed Images (512 Kbits UFM)“ tak, jak je znázorněno na obrázku 13.3.



Obr. 13.3 Nastavení režimu konfigurace hradlového pole

9) Kliknutím na tlačítko „OK“ ve formuláři s názvem „Device and Pin Options“ a opětovným kliknutím na tlačítko „OK“ ve formuláře s názvem „Device“ je projekt připraven na analýzu a syntézu.

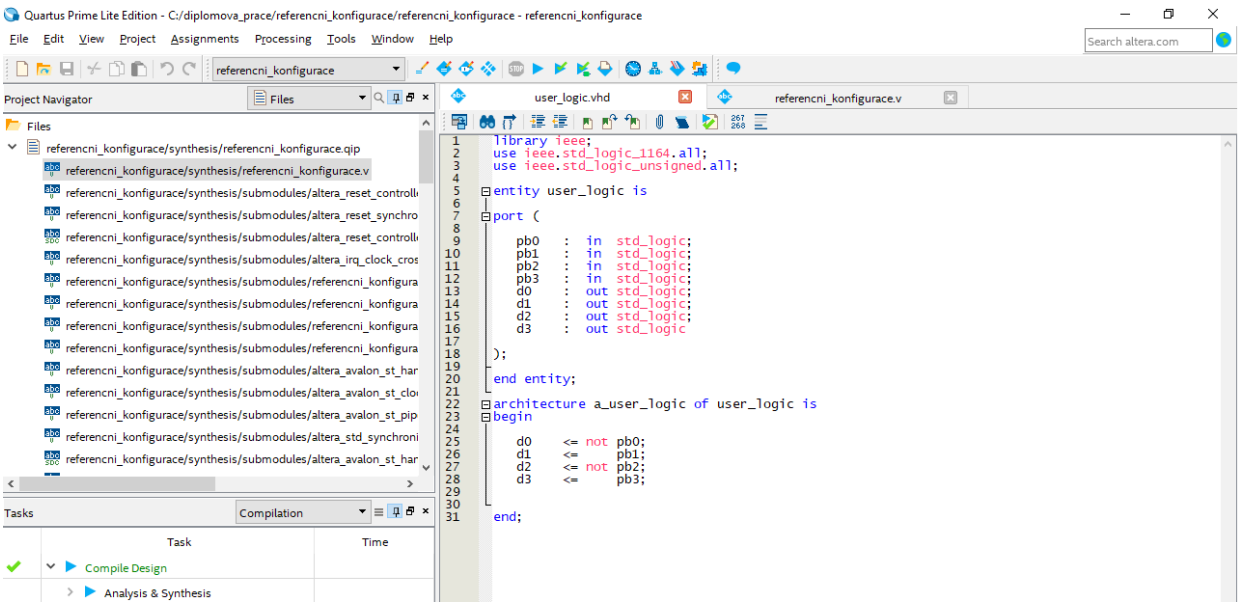
10) Stisknutím klávesové zkratky „Ctrl+K“ je spuštěna analýza a syntéza celého projektu.

13.1.3.1 Návrh entity bloku User logic

Jak již bylo řečeno v kapitole 12.14, blok s názvem User logic v architektuře slouží v tomto případě k indikaci přepnutí na referenční konfigurační obraz. Jedná se o entitu obsahující logickou funkci, která je dána pravdivostní tabulkou na obr. 12.19 v kapitole 12.14.1. Entitu s názvem „user_logic“ je poté nutné vložit do vrcholové entity projektu a opět spustit analýzu a syntézu nad celým projektem. K výše uvedenému slouží následující postup.

1) Stisknutím klávesové zkratky „CTRL+N“ ve vývojovém prostředí Quartus a následným vybráním možnosti „VHDL File“ je vytvořen nový soubor s názvem „Vhdl1.vhd“.

2) V tomto souboru je pomocí jazyka VHDL popsána funkce „user_logic“. Funkce je dána pravdivostní tabulkou na obr. 12.19 v kapitole 12.14.1. VHDL kód funkce je znázorněn na obr. 13.4.

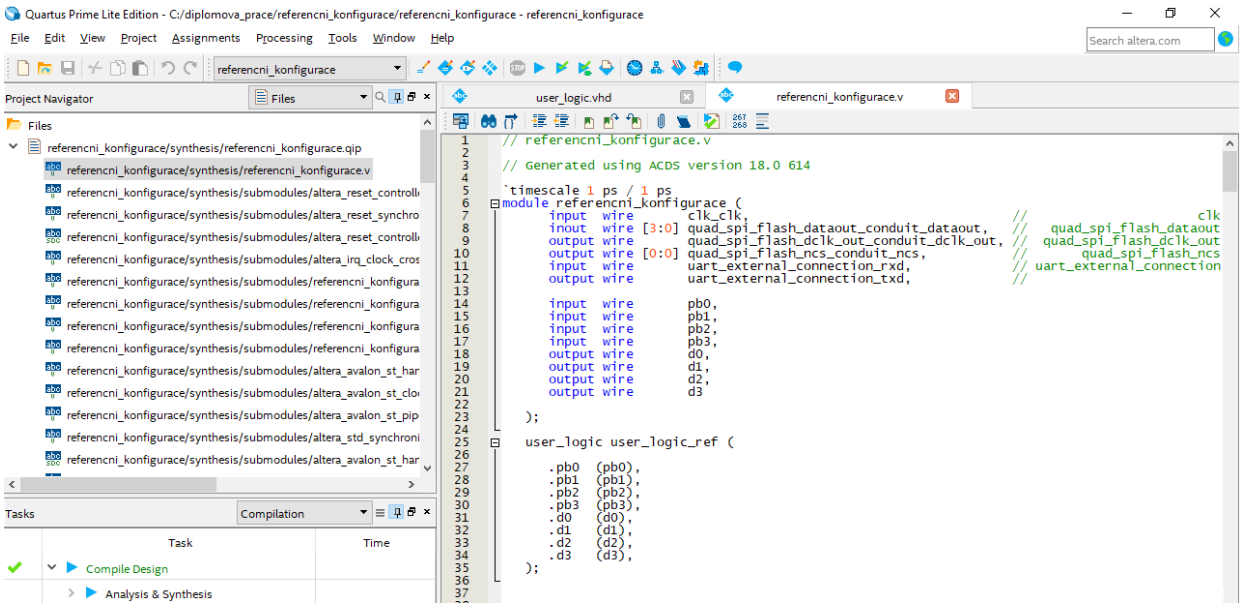


Obr. 13.4 Entita s názvem user_logic referenční konfigurace

3) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Save As“ je soubor uložen do pracovní složky projektu. Název souboru pro uložení byl zvolen jako „User_logic.vhd“.

4) Kliknutím na tlačítko „Save“ je soubor vložen do projektu.

5) Entitu s názvem „User_logic“ je nutné vložit do vrcholové entity projektu. To bylo provedeno jednoduchou úpravou kódu (řádek číslo 14 až číslo 35) vrcholové entity s názvem „Referencni_konfigurace“ – viz obr. 13.5. Vrcholová entita s názvem „Referencni_konfigurace“ je na rozdíl od entity s názvem „user_logic“ napsána v jazyce Verilog.



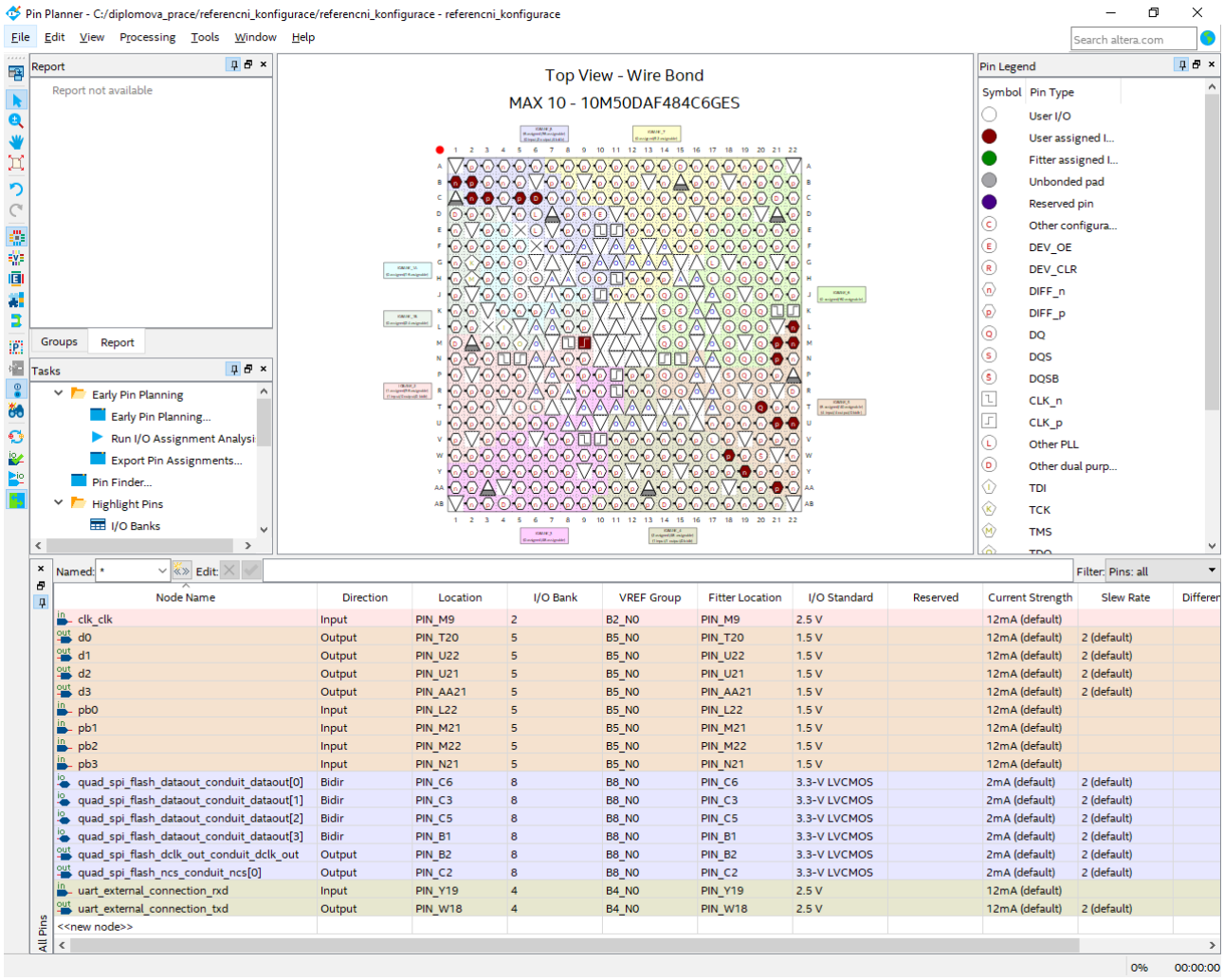
Obr. 13.5 Vložení entity s názvem user_logic do vrcholové entity projektu

6) Stisknutím klávesové zkratky „CTRL+K“ je opět spuštěna analýza a syntéza celého projektu.

13.1.3.2 Ošetření hodinových domén

V architektuře systému jsou pro komponenty použity dvě hodinové domény o frekvenci 15 MHz a 50 MHz. Aby u komponent nedocházelo k hazardům z hlediska časování, je nutné tyto hodinové domény vhodně ošetřit. K tomu složí tzv. SDC soubor, který je založen na skryptovacím jazyce TCL a je umístěn mezi ostatní zdrojové soubory projektu. K ošetření hodinových domén slouží následující postup.

1) Kliknutím na tlačítko ve vývojovém prostředí Quartus s názvem „Assignments“ a následným vybráním možnosti „Pin Planner“ se spustí nástroj, pomocí kterého jsou jednotlivým signálům, které náleží daných bloků architektury přiřazeny fyzické piny hradlového pole. Čísla pinů a napěťové hladiny jednotlivých signálů jsou uvedeny u odpovídajících bloků architektury v části „Praktická realizace“ v kapitole 12. Na obr. 13.6 je znázorněno přiřazení fyzických pinů hradlového pole nástrojem Pin Planner.



Obr. 13.6 Fyzické přiřazení pinů hradlového pole

2) Stisknutím klávesové zkratky „CTRL+L“ je spuštěna kompilace celého projektu.

3) Kliknutím na tlačítko „Tools“ a následným vybráním možnosti „Timing Analyzer“ je spuštěn nástroj pro časovou analýzu celého projektu.

4) Dvojitým kliknutím na tlačítko „Create Timing Netlist“ v okně s názvem „Tasks“, nástroje Timing analyzer je spuštěna analýza, kde výsledkem je seznam všech použitých signálů v projektu.

5) Kliknutím na tlačítko „File“ a následným vybráním možnosti „New SDC File“ v nástroji Timing Analyzer je otevřen textový editor s nově vytvořeným SDC souborem.

6) Kliknutím pravým tlačítkem myši do pracovní plochy textového editoru je zobrazeno výběrové menu. Vybráním možnosti „Instert Constraint“ a následným kliknutím na tlačítko „Create Clock“ je otevřen průvodce, kde je určen zdroj hodinového signálu.

7) Pole s názvem „Clock name“ určuje název zdroje hodinového signálu. Název hodinového

signálu byl zvolen jako „clk“.

8) Kliknutím na tlačítko v poli s názvem „Targets“ je otevřen průvodce „Name Finder“. Průvodce slouží k určení signálu, který je zdrojem hodinového signálu pro celý systém. Následným kliknutím na tlačítko „List“ v tomto průvodci jsou zobrazeny všechny signály vrcholové entity systému.

9) Dvojitým kliknutím na signál s názvem „clk_clk“ je určen zdroj hodinového signálu, který vstupuje do systému z bloku architektury s názvem CLK.

10) Kliknutím na tlačítko „OK“ je průvodce „Name Finder“ zavřen.

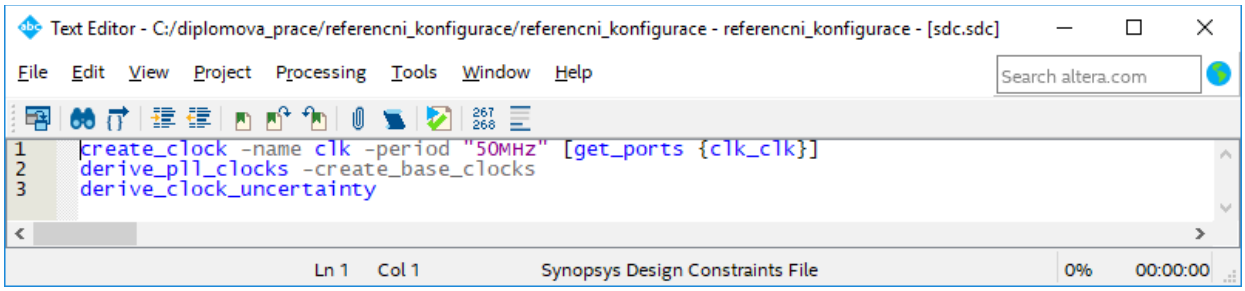
11) Kliknutím na tlačítko „Insert“ v průvodci s názvem „Create Clock“ je vložen do textového editoru SDC příkaz, definující zdroj hodinového signálu systému.

12) V textovém editoru je nutné přepsat hodnotu periody hodinového signálu, z hodnoty 10.000, která je dána jako výchozí, na hodnotu "50MHz", která je dána blokem architektury s názvem CLK – viz obr. 12.7.

13) Opětovným kliknutím pravým tlačítkem myši do pracovní plochy textového editoru je zobrazeno výběrové menu. Vybráním možnosti „Instert Constraint“ a následným kliknutím na tlačítko „Derive PLL Clocks“ je otevřen průvodce, kde je SDC příkaz definující signály, které vystupují ven z fázového závěsu.

14) Kliknutím na tlačítko „Insert“ v průvodci „Derive PLL Clocks“ je SDC příkaz vložen do textového editoru – viz obr. 13.7.

15) Opětovným kliknutím pravým tlačítkem myši do pracovní plochy textového editoru je zobrazeno výběrové menu. Vybráním možnosti „Instert Constraint“ a následným kliknutím na tlačítko „Derive Clocks Uncertainty“ je otevřen průvodce, kde je SDC příkaz ošetřující negativní vlastnosti fázového závěsu (kolísání šířky pulzu apod.).



Obr. 13.7 Ošetření hodinových domén

16) Kliknutím na tlačítko „Insert“ v průvodci s názvem „Derive Clocks Uncertainty“ je SDC

příkaz vložen do textového editoru – viz obr. 13.7.

17) Tímto jsou ošetřeny nejdůležitější signály systému. Zavřením textového editoru a následným potvrzením o uložení změn, je SDC soubor vložen do projektu.

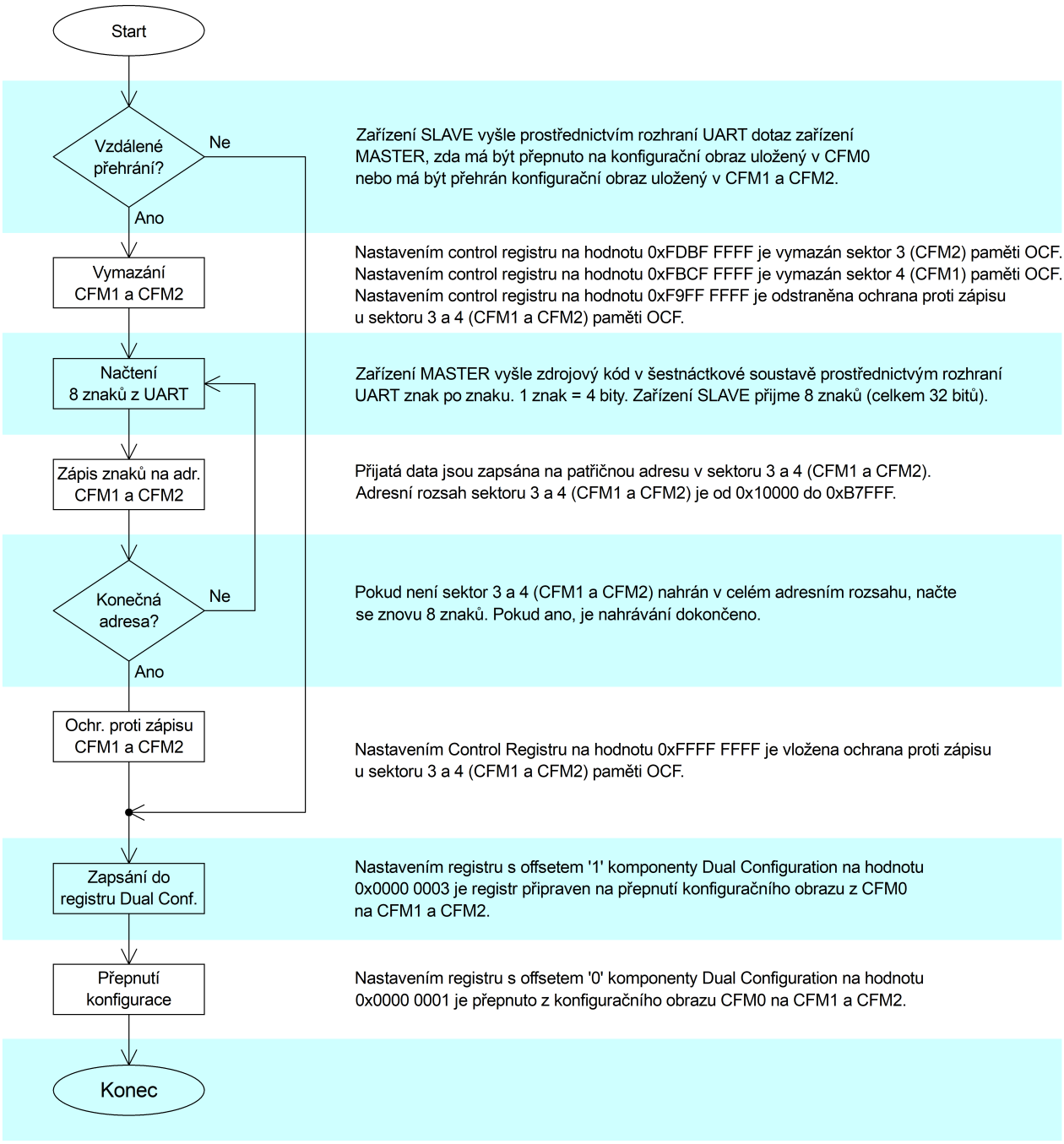
18) Stisknutím klávesové zkratky „CTRL+L“ je spuštěna kompilace celého projektu.

13.1.4 Návrh programu pro NIOS II

V této kapitole je popsán program pro procesor NIOS II. Nejdříve bude program popsán teoreticky pomocí vývojového diagramu a následně bude popsána jeho praktická realizace v nástroji NIOS II Software Build Tools for Eclipse.

13.1.4.1 Popis funkce programu vývojovým diagramem

Na obr. 13.8 je vývojový diagram, který popisuje funkci programu procesoru NIOS II pro referenční konfiguraci. Jednotlivé bloky vývojového diagramu jsou v tomto obrázku popsány a zároveň vysvětleny. Na základě vývojového diagramu je algoritmus v následující kapitole naprogramován v nástroji NIOS II Software Build Tools for Eclipse v programovacím jazyce C.



Obr. 13.8 Vývojový diagram programu procesoru NIOS II pro referenční konfiguraci

13.1.4.2 Praktická realizace

V této kapitole je na základě vývojového diagramu z předchozí kapitoly vytvořen program pro procesor NIOS II v programovacím jazyce C pro referenční konfiguraci. Program je vytvořen ve vývojovém prostředí NIOS II Software Build Tools for Eclipse. K vytvoření programu pro procesor NIOS II slouží následující postup.

1) Kliknutím na tlačítko ve vývojovém prostředí Quartus s názvem „Tools“ a následným

vybráním možnosti „NIOS II Software Build Tools for Eclipse“ je spuštěn nástroj pro vytvoření programu procesoru NIOS II.

2) Kliknutím na tlačítko „Browse“ v otevřeném průvodci, je vybrán hlavní adresář projektu s názvem „referencni_konfigurace“.

3) Kliknutím na tlačítko v prostředí Nios II - Eclipse s názvem „File“, následným kliknutím na tlačítko „New“ a vybráním možnosti „Nios II Application and BSP from Template“, je spuštěn průvodce vytvořením nového projektu. V tomto projektu bude následně vytvořen program procesoru NIOS II.

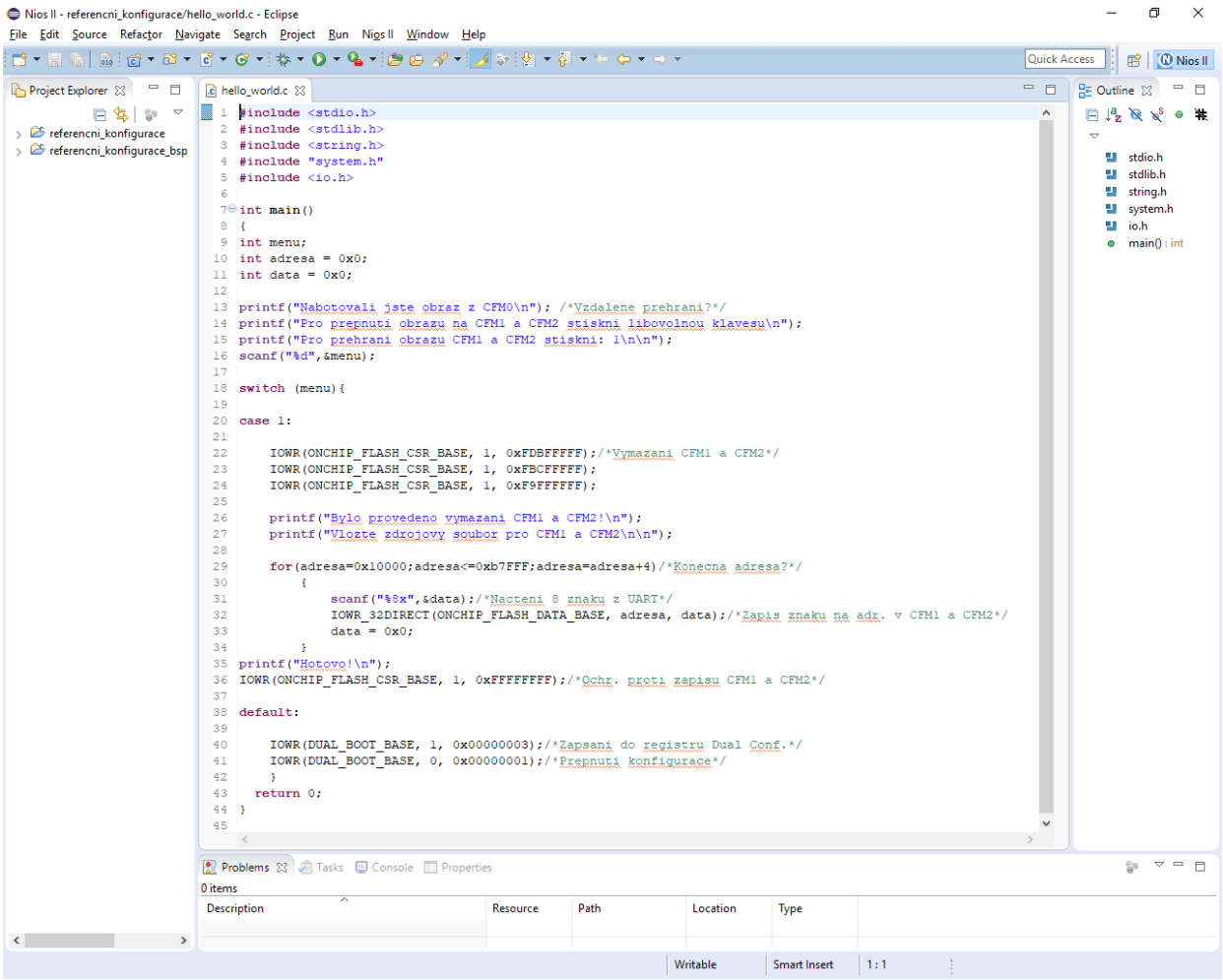
4) Kliknutím na tlačítko v poli s názvem „SOPC Information File name“ je nutné zadat umístění souboru s příponou .SOPCINFO, který obsahuje důležité informace o projektu, vytvořeného nástrojem Platform Designer. Soubor, jehož název je „referencni_konfigurace.sopcinfo“ je umístěn v hlavním adresáři projektu v podadresáři s názvem „referencni_konfigurace“. Kliknutím na tlačítko „Open“ je soubor vložen do průvodce vytvořením nového projektu.

5) V poli s názvem „Project name“ je nutné zadat název nového projektu. Název nového projektu byl zvolen jako „refererencni_konfigurace“.

6) Kliknutím na tlačítko „Finish“ v průvodci vytvořením nového projektu je vytvoření nového projektu úspěšně dokončeno.

7) Dvojitým kliknutí na soubor s názvem „hello_world.c“ v okně „Project Explorer“ je otevřen soubor s jednoduchou ukázkou programu procesoru NIOS II v programovacím jazyce C. Tento program je možné vymazat nahradit vlastním programem v jazyce C.

8) Na základě vývojového diagramu na obr. 13.8 v kapitole 13.1.4.1 byl vytvořen program procesoru NIOS II. Na obr. 13.9 je znázorněna realizace programu ve vývojovém prostředí včetně komentářů, pomocí kterých je možné přiřadit daný blok ve vývojovém diagramu k části příslušného kódu v programovacím jazyce C.

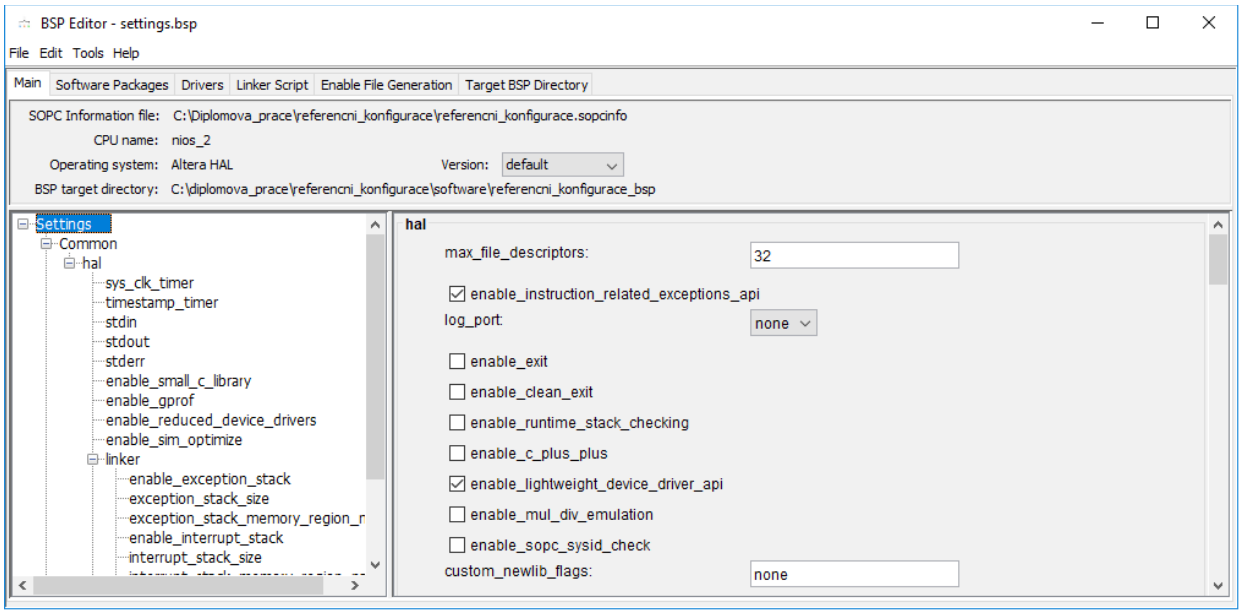


Obr. 13.9 Kód v programovacím jazyce C procesoru NIOS II pro referenční konfiguraci

9) Stisknutím klávesové zkratky „CTRL+S“ je projekt uložen.

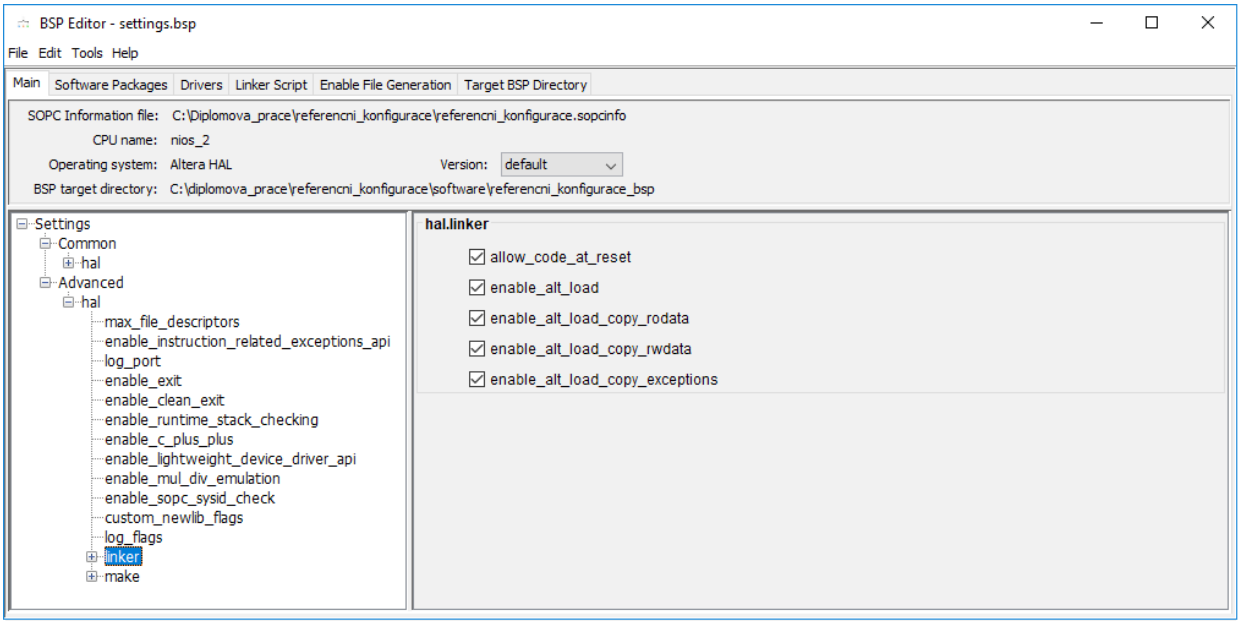
10) V této části je nutné nastavit další parametry projektu, jako je způsob bootování, možnosti použití funkcí a maker z knihoven apod.. K tomu slouží nástroj BSP Editor. Kliknutím pravým tlačítkem na projekt v okně „Project Explorer“, vybráním možnosti „NIOS II“ a následným kliknutím na tlačítko „BSP Editor“ je spuštěn nástroj BSP Editor.

11) Na záložce „Main“ v nastavení „Settings“ je nutné vybrat v části „hal“ možnost „enable_instruction_related_exceptions_api“ (umožní použití funkcí v hlavičkových souborech s příponou *.h*) a „enable_lightweight_device_driver_api“ (umožní nastavení pro komunikaci pomocí rozhraní UART). Ostatní možnosti v této části zůstávají nevybrané viz obr. 13.10.



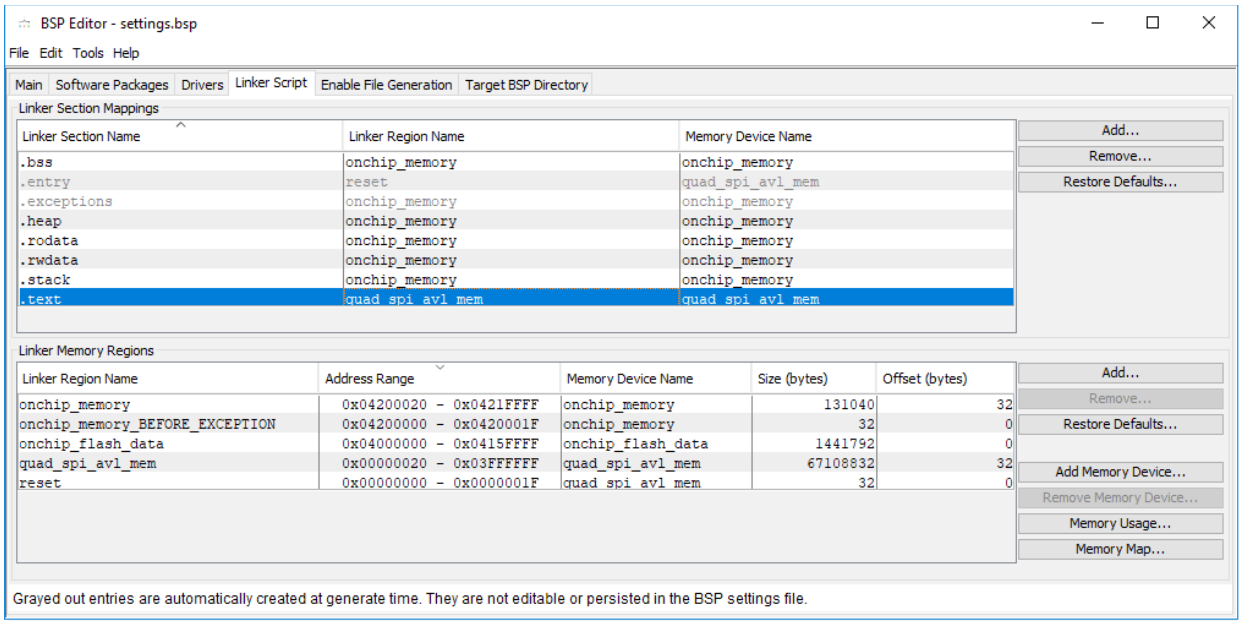
Obr. 13.10 Nastavení parametrů pomocí BSP Editou pro referenční konfiguraci

12) Dále je nutné nastavit možnosti bootování procesoru NIOS II. V záložce „Main“ v nastavení „Advanced“, rozbalením části „hal“ a následně „linker“ je vybráním všech dostupných možností nastaveno bootování podle schématu, doporučeného dokumentem [10]. Uvedené nastavení je v tomto dokumentu označováno jako 4a - viz obr. 13.11. Nastavení umožňuje použití funkce „alt_load()“, která slouží jako jako zavaděč (nazývaný také jako bootloader) – viz kapitola 12.5.



Obr. 13.11 Nastavení zavaděče procesoru NIOS II pro referenční konfiguraci

13) Poslední nastavení spočívá v přiřazení jednotlivých datových sekcí ke konkrétní použité paměti. Datová sekce .entry a .text se nacházejí v externí QSPI paměti (quad_spi_avl_mem) a ostatní datové sekce se nacházejí v interní paměti hradlového pole OCR (onchip_memory) viz kapitola 12.5. Toto nastavení je znázorněno na obr. 13.12.



Obr. 13.12 Nastavení datových sekcí procesoru NIOS II pro referenční konfiguraci

14) Kliknutím na tlačítko „Generate“ je nastavení uloženo a následným kliknutím na tlačítko „Exit“ je BSP editor zavřen.

15) Stisknutím klávesové zkratky „CTRL+B“ je spuštěn překlad projektu.

16) Stisknutím klávesové zkratky „Shift+F9“ a následným dvojitým kliknutím na možnost „mem_init_generate“ je vygenerován zdrojový soubor procesoru NIOS II, který bude uložen do QSPI flash paměti.

17) Nyní je možné vývojové prostředí Nios II – Eclipse opustit.

18) Ve vývojovém prostředí Quartus je potřeba vložit inicializační soubor pamětí s názvem „meminit.qip“, který vznikl při generování zdrojových souborů ve vývojovém prostředí Nios II – Eclipse. Soubor je vložen kliknutím na tlačítko v hlavním menu s názvem „Assignments“ a následným vybráním možnosti „Settings“.

19) V kategorii „Files“ otevřeného formuláře je potřeba vyplnit cestu k souboru s názvem „meminit.qip“. Soubor je uložen v hlavním adresáři projektu *referencni_konfigurace => software => refererencni_konfigurace => mem_init.*

20) Vybráním souboru a kliknutím na tlačítko „Open“ je soubor vložen do otevřeného

formuláře. Následným kliknutím na tlačítko „OK“ je soubor vložen do projektu ve vývojovém prostředí Quartus.

21) Stisknutím klávesové zkratky „CTRL+L“ je spuštěna kompilace celého projektu.

22) Tímto je celý projekt s názvem „referencni_konfigurace“ dokončen a je možné vývojové prostředí Quartus opustit.

13.2 Návrh aplikační konfigurace číslo 1

Aplikační konfigurace systému zajišťuje na povel zařízení MASTER přepnutí na referenční konfiguraci. Narozdíl od referenční konfigurace neposkytuje aplikační konfigurace možnost vzdáleného přehrání referenční konfigurace prostřednictvím rozhraní UART.

Vzhledem k tomu, že se aplikační konfigurace a referenční konfigurace velmi podobají, budou v této kapitole popsány pouze rozdílné části projektu oproti referenční konfiguraci.

13.2.1 Založení nového projektu v prostředí Quartus

Postup pro založení nového projektu ve vývojovém prostředí Quartus je shodný s postupem v kapitole 13.1.1. Rozdíl je pouze v názvu projektu a v názvu adresáře projektu. Název tohoto projektu byl zvolen jako „aplikacni_konfigurace_1“ a je uložen v adresáři se shodným názvem.

13.2.2 Realizace systému na základě architektury

Část architektury, která je realizována hradlovým polem je vytvořena nástrojem Platform Designer, který je součástí vývojového prostředí Quartus.

Pro realizaci architektury nástrojem Platform Designer je postupováno stejným způsobem, jako v kapitole 13.1.2. Rozdíl je pouze v názvu projektu, názvu adresáře projektu a nastavením umístění Reset vektoru u komponenty, reprezentující blok v architektuře s názvem NIOS 2 - viz kapitola 12.15. Název tohoto projektu byl zvolen jako „aplikacni_konfigurace_1“ a je uložen v adresáři se shodným názvem.

13.2.3 Návrh projektu v prostředí Quartus

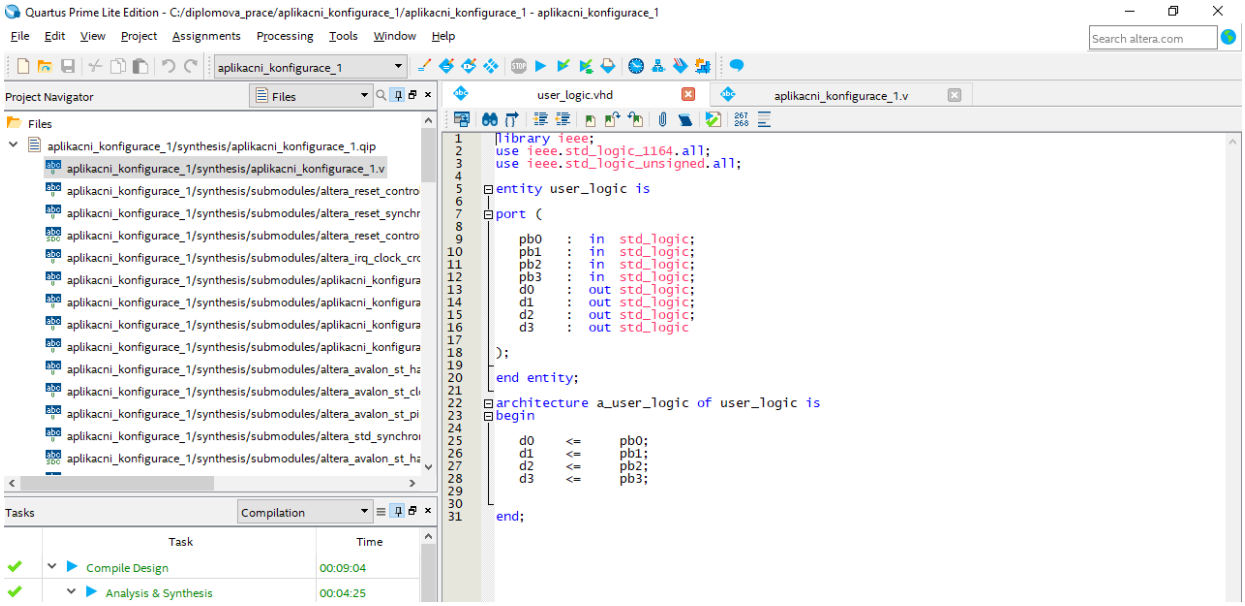
Pro realizaci projektu ve vývojovém prostředí Quartus je postupováno stejným způsobem, jako v kapitole 13.1.3. Při importování HLD souborů do projektu je pouze nutné dát pozor, zda se jedná o projekt s názvem „aplikacni_konfigurace_1“.

13.2.3.1 Návrh entity bloku User logic

Jak již bylo řečeno v kapitole 12.4, blok s názvem User logic v architektuře slouží k indikaci přepnutí na aplikační konfigurační obraz číslo 1. Jedná se o entitu s funkcí, která je dána pravdivostní tabulkou na obr. 12.20 v kapitole 12.14.1. Entitu s názvem „user_logic“ je poté nutné vložit do vrcholové entity projektu a opět spustit analýzu a syntézu nad celým projektem. K výše uvedenému slouží následující postup.

1) Stisknutím klávesové zkratky „CTRL+N“ ve vývojovém prostředí Quartus a následným vybráním možnosti „VHDL File“ je vytvořen nový soubor s názvem „Vhdl1.vhd“.

2) V tomto souboru je pomocí jazyka VHDL popsána funkce entity User logic. Funkce je dána pravdivostní tabulkou na obr. 12.20 v kapitole 12.14.1. VHDL kód funkce je znázorněn na obr. 13.13.

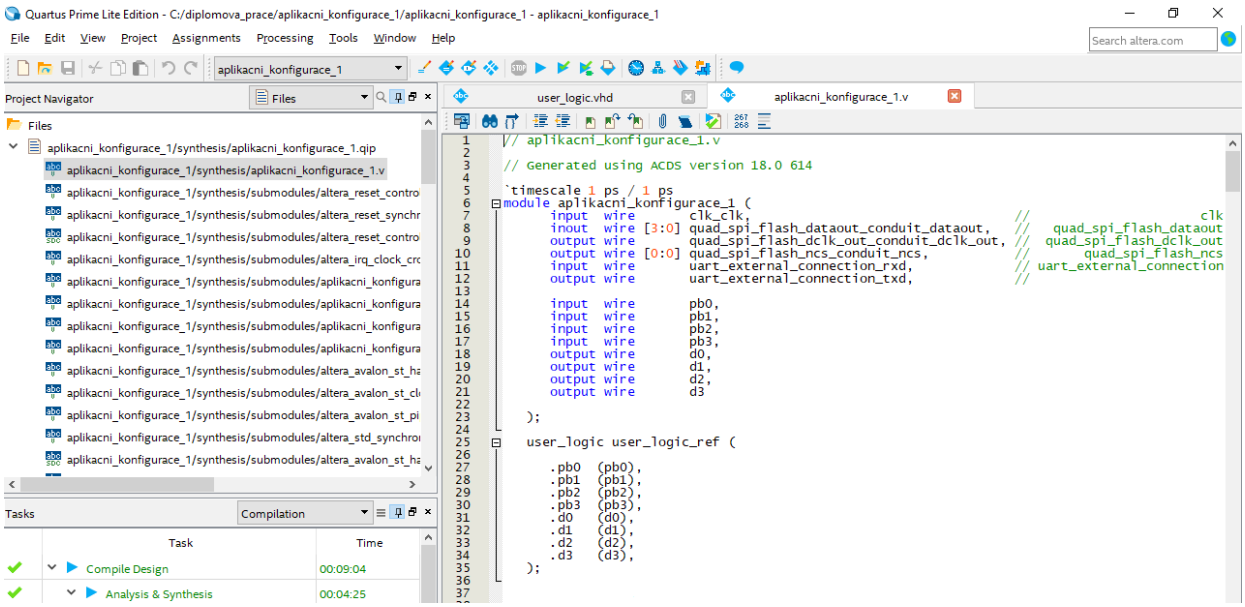


Obr. 13.13 Entita s názvem user_logic aplikační konfigurace číslo 1

3) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Save as“ je soubor uložen do pracovní složky projektu. Název souboru pro uložení byl zvolen jako „user_logic.vhd“.

4) Kliknutím na tlačítko „Save“ je soubor uložen a vložen do projektu.

5) Entitu s názvem „user_logic“ je nutné vložit do vrcholové entity projektu. To bylo provedeno jednoduchou úpravou kódu (řádek číslo 14 až číslo 35) vrcholové entity s názvem „aplikacni_konfigurace_1“ – viz obr. 13.14. Vrcholová entita s názvem „aplikacni_konfigurace_1“ je na rozdíl od entity s názvem „user_logic“ napsána v jazyce Verilog.



Obr. 13.14 Vložení entity s názvem user_logic do vrcholové entity projektu

6) Stisknutím klávesové zkratky „CTRL+K“ je opět spuštěna analýza a syntéza celého projektu.

13.2.3.2 Ošetření hodinových domén

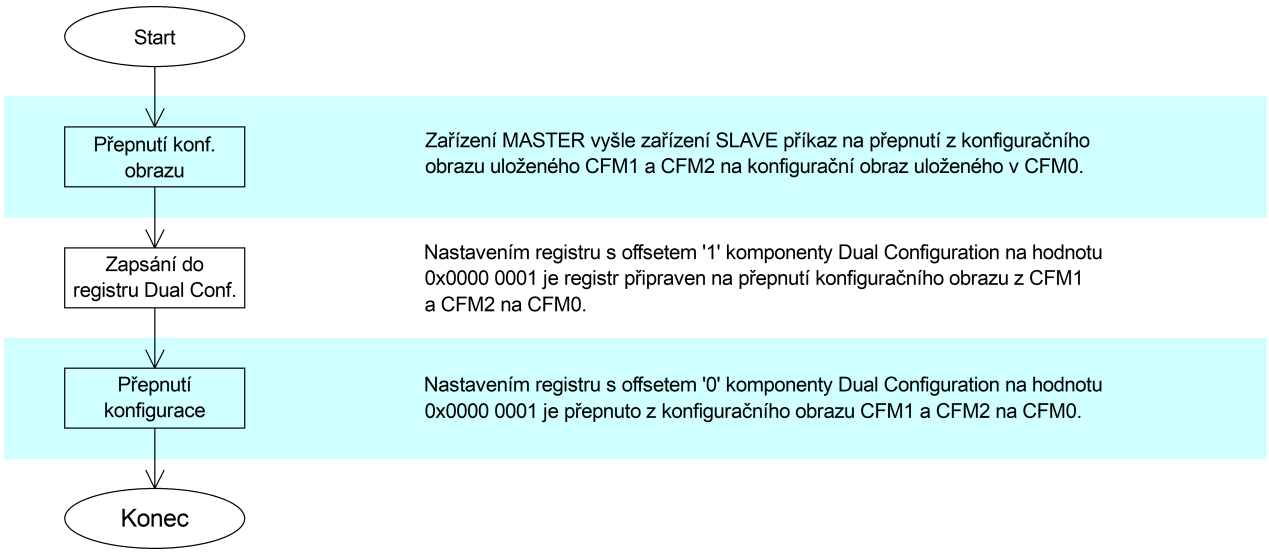
Ošetření hodinových domén v tomto projektu je provedeno stejným způsobem, jako v kapitole 13.1.3.2.

13.2.4 Návrh programu pro NIOS II

V této kapitole je popsán program pro procesor NIOS II. Nejdříve bude program popsán teoreticky pomocí vývojového diagramu a následně bude popsána jeho praktická realizace v nástroji NIOS II Software Build Tools for Eclipse.

13.2.4.1 Popis funkce programu vývojovým diagramem

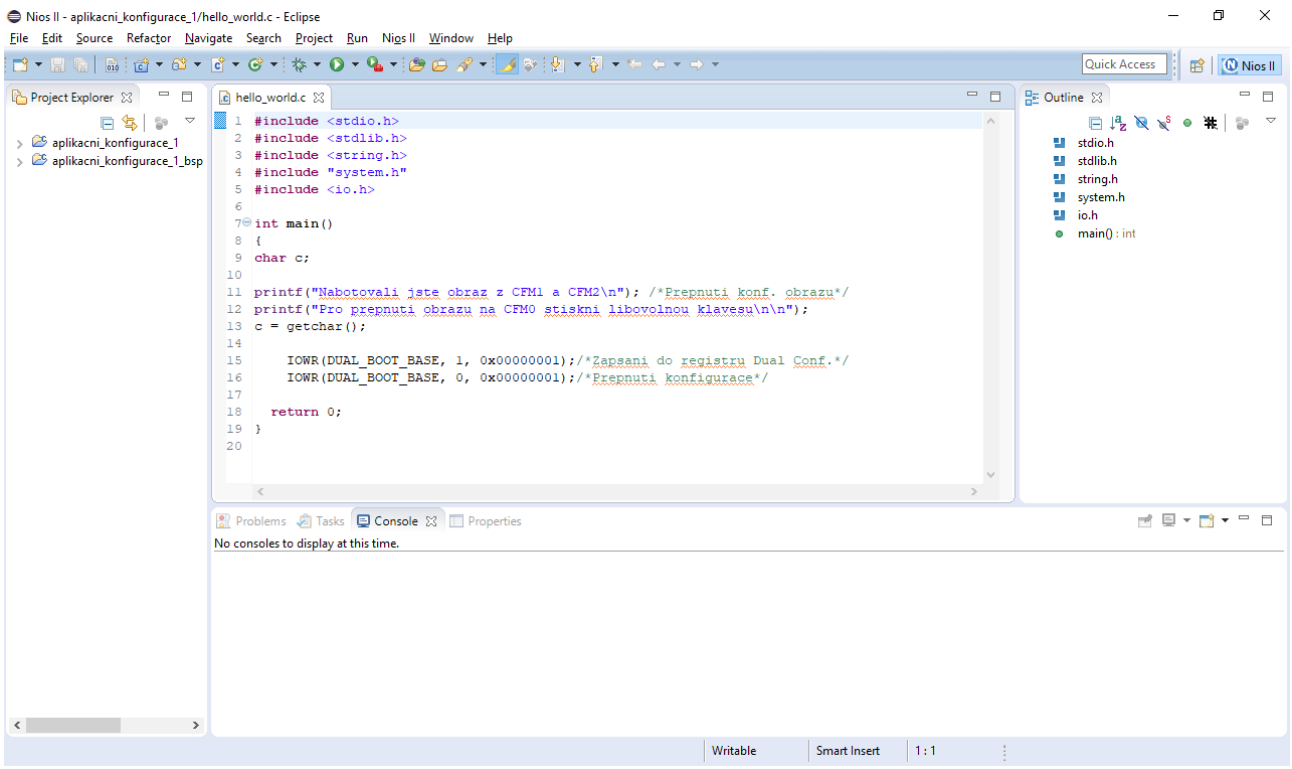
Na obr. 13.15 je vývojový diagram, který popisuje funkci programu procesoru NIOS II pro aplikační konfiguraci 1. Jednotlivé bloky vývojového diagramu jsou v tomto obrázku popsány a zároveň vysvětleny. Na základě vývojového diagramu je v následující kapitole je algoritmus naprogramován pomocí programovacího jazyka C ve vývojovém prostředí NIOS II Software Build Tools for Eclipse.



Obr. 13.15 Vývojový diagram programu NIOS II pro aplikační konfiguraci číslo 1

13.2.4.2 Praktická realizace

Postup pro praktickou realizaci programu procesoru NIOS II pro aplikační konfiguraci číslo 1 je stejný, jako způsob uvedený v kapitole 13.1.4.2. Rozdíl je pouze v názvu projektu, v názvu adresáře projektu a kódem v programovacím jazyce C, který byl vytvořen na základě vývojového diagramu na obr. 13.15. Název tohoto projektu byl zvolen jako „aplikacni_konfigurace_1“ a je uložen v adresáři se shodným názvem. Na obr. 13.16 je znázorněna realizace programu ve vývojovém prostředí Nios II – Eclipse včetně komentářů, pomocí kterých lze přiřadit daný blok ve vývojovém diagramu k části příslušného kódu v programovacím jazyce C.



Obr. 13.16 Kód v programovacím jazyce C procesoru NIOS II pro aplikační konfiguraci č. 1

13.3 Návrh aplikační konfigurace číslo 2

Jelikož se aplikační konfigurace číslo 1 a aplikační konfigurace číslo 2 liší pouze v entitě bloku s názvem User logic, bude v této kapitole uveden pouze bod s návrhem entity „user_logic“, který nahrazuje kapitolu 13.2.3.1. Všechny ostatní body jsou shodné s návrhem aplikační konfigurace číslo 1 uvedené v kapitole 13.2.

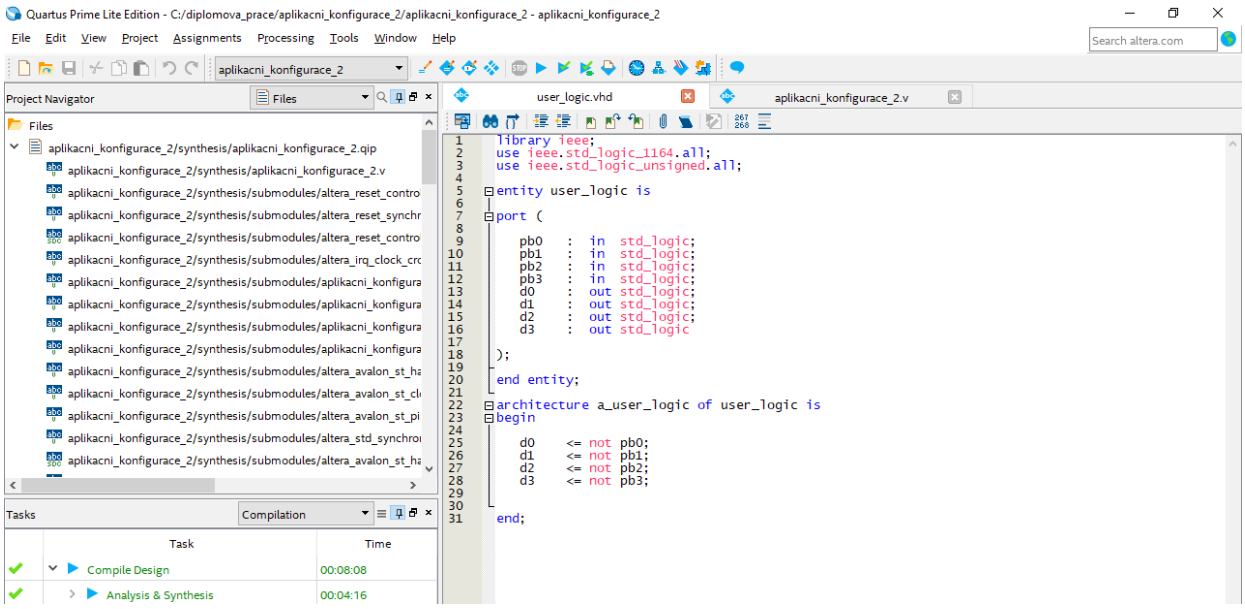
Název tohoto projektu byl zvolen jako „aplikacni_konfigurace_2“ a je uložen v adresáři se shodným názvem.

13.3.1.1 Návrh entity bloku User logic

Jak již bylo řečeno v kapitole 12.4, blok s názvem User logic v architektuře slouží k indikaci přepnutí na aplikační konfigurační obraz číslo 2. Jedná se o entitu s funkcí, která je dána pravdivostní tabulkou na obr. 12.21 v kapitole 12.14.1. Entitu s názvem „user_logic“ je poté nutné vložit do vrcholové entity projektu a opět spustit analýzu a syntézu nad celým projektem. K výše uvedenému slouží následující postup.

1) Stisknutím klávesové zkratky „CTRL+N“ ve vývojovém prostředí Quartus a následným vybráním možnosti „VHDL File“ je vytvořen nový soubor s názvem „Vhdl1.vhd“.

2) V tomto souboru je pomocí jazyka VHDL popsána funkce entity User logic. Funkce je dána pravdivostní tabulkou na obr. 12.21 v kapitole 12.14.1. VHDL kód funkce je znázorněn na obr. 13.17.

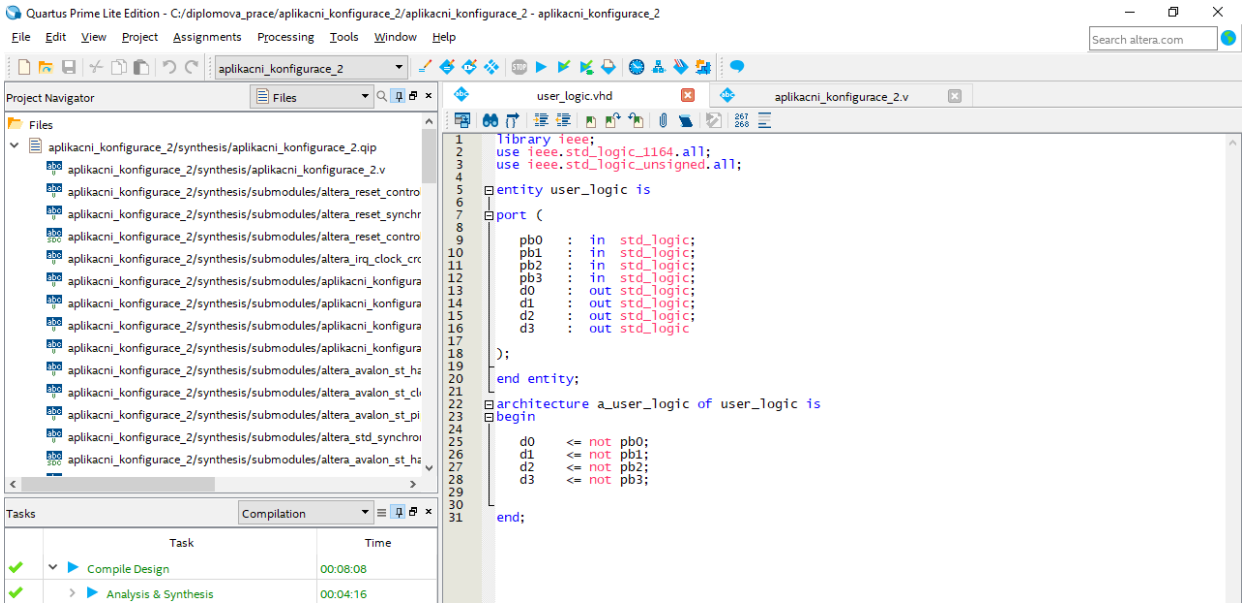


Obr. 13.17 Entita s názvem user_logic aplikační konfigurace číslo 2

3) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Save as“ je soubor uložen do pracovní složky projektu. Název souboru pro uložení byl zvolen jako „user_logic.vhd“.

4) Kliknutím na tlačítko „Save“ je soubor vložen do projektu.

5) Entitu s názvem „user_logic“ je nutné vložit do vrcholové entity projektu. To bylo provedeno jednoduchou úpravou kódu (řádek číslo 14 až číslo 35) vrcholové entity s názvem „aplikacni_konfigurace_2“ – viz obr. 13.18. Vrcholová entita s názvem „aplikacni_konfigurace_2“ je na rozdíl od entity s názvem „user_logic“ napsána v jazyce Verilog.



Obr. 13.18 Vložení entity s názvem user_logic do vrcholové entity projektu

6) Stisknutím klávesové zkratky „CTRL+K“ je opět spuštěna analýza syntéza celého projektu.

13.4 Generování výstupních souborů

V této kapitole bude popsán způsob generování finálních souborů pro prvotní nahrání a pro následnou ukázku vzdáleného přehrání aplikačního obrazu číslo 1 aplikačním obrazem číslo 2 prostřednictvím rozhraní UART.

Podle [10] je před samotným generováním finálních souborů nutné do každé složky s jednotlivými projekty umístit soubor s názvem „quartus.ini“, který obsahuje řetězec „PGMIO_SWAP_HEX_BYTE_DATA=ON“. Po umístění souboru do projektů je nutné všechny projekty znovu zkompilovat. Řetězec slouží pro správné generování .pof souboru, obsahující konfiguraci pro procesor NIOS II a následné nahrání do QSPI paměti pomocí zavaděče.

13.4.1 Soubory pro prvotní nahrání

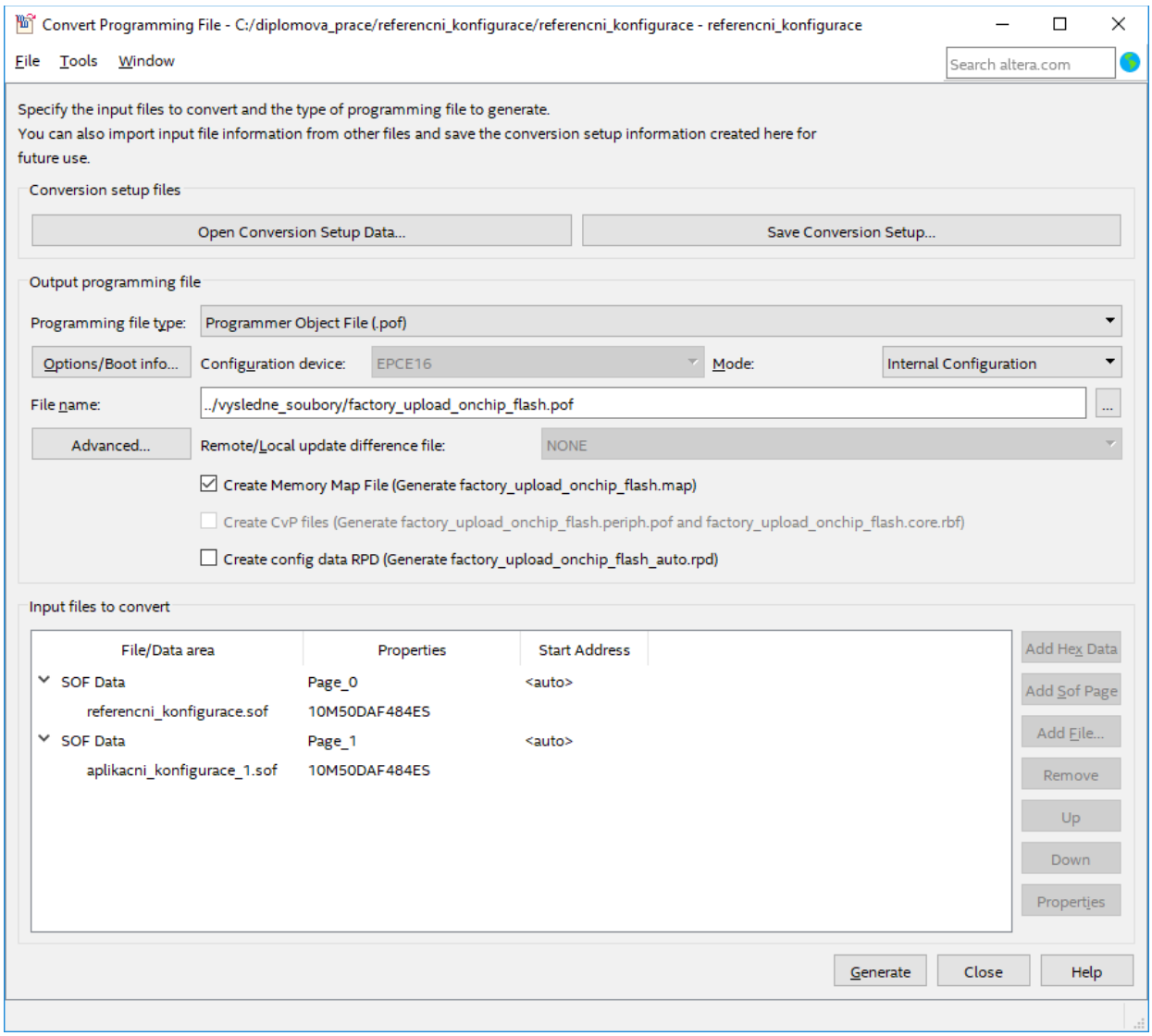
Jedná se soubory s příponou .pof, které jsou nahrány do interní flash paměti hradlového pole OCF (konkrétně se jedná o referenční konfiguraci a aplikační konfiguraci číslo 1) resp. do externí QSPI paměti, ve které jsou uloženy dva programy procesoru NIOS (pro referenční konfiguraci a aplikační konfiguraci číslo 1 společně s aplikační konfigurací číslo 2).

13.4.1.1 Výsledný soubor pro interní konfigurační paměť

Jedná se o soubor, který v sobě obsahuje dva konfigurační obrazy. Jedná se o referenční

konfigurační obraz, uložený v části paměti CFM0 a aplikační konfigurační obraz číslo 1, uložený v části paměti CFM1 a CFM2 paměti OCF. K vygenerování výsledného souboru slouží následující postup.

1) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Convert Programming Files“ je spuštěn nástroj pro generování finálních souborů.
2) V poli s názvem „Mode“ je vybrána možnost „Internal Configuration“.
3) V okně s názvem „Input files to convert“, vybráním řádky s popisem „Page_0“ a následným kliknutím na tlačítko „Add File“ je spuštěn průvodce, ve kterém je nutné vybrat umístění vygenerovaného souboru s názvem „referencni_konfigurace.sof“. Tento soubor prezentuje referenční konfigurační obraz, který bude uložen do části CFM0 paměti OCF. Soubor se nachází v hlavním adresáři projektu s názvem „referencni_konfigurace“ v podadresáři „output_files“. Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Convert Programing file.
4) Kliknutím na tlačítko s názvem „Add Sof Page“ se v okně s názvem „Input files to convert“ objeví nový řádek s popisem „Page_1“.
5) Vybráním řádky s popisem „Page_1“ a následným kliknutím na tlačítko „Add File“ je spuštěn průvodce, ve kterém je nutné vybrat umístění vygenerovaného souboru s názvem „aplikacni_konfigurace_1.sof“. Tento soubor prezentuje aplikační konfigurační obraz číslo 1, který bude uložen do části CFM1 a CFM2 paměti OCF. Soubor se nachází v hlavním adresáři projektu s názvem „aplikacni_konfigurace_1“ v podsložce „output_files“.Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Convert Programing file.
6) Do pole s názvem „File name“ je nutné zadat cestu a název výsledného soboru s příponou .pof. Název výstupního souboru byl zvolen jako „factory_upload_onchip_flash.pof“ a je uložen v adresáři se všemi projekty v podadresáři s názvem „vysledne_soubory“.
7) Na obr. 13.19 je znázorněno nastavení nástroje Convert Programing File pro vygenerování výsledného souboru. Kliknutím na tlačítko „Generate“ je výsledný soubor vygenerován.



Obr. 13.19 Generování výsledného souboru pro paměť OCF

13.4.1.2 Výsledný soubor pro externí konfigurační paměť

Jedná se o soubor, který v sobě obrahuje program procesoru NIOS II pro referenční konfiguraci a program procesoru NIOS II pro aplikační konfiguraci číslo 1 společně s aplikační konfigurací číslo 2. Výsledný soubor, obsahující oba programy je uložen v externí QSPI paměti. K vygenerování výsledného souboru slouží následující postup.

1) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Convert Programming Files“ je spuštěn nástroj pro generování finálních souborů.
2) V poli s názvem „Configuration device“ byla vybrána možnost „CFI_512Mb“.
3) V okně s názvem „Input files to convert“ je nutné odstranit všechny přednastavené možnosti.

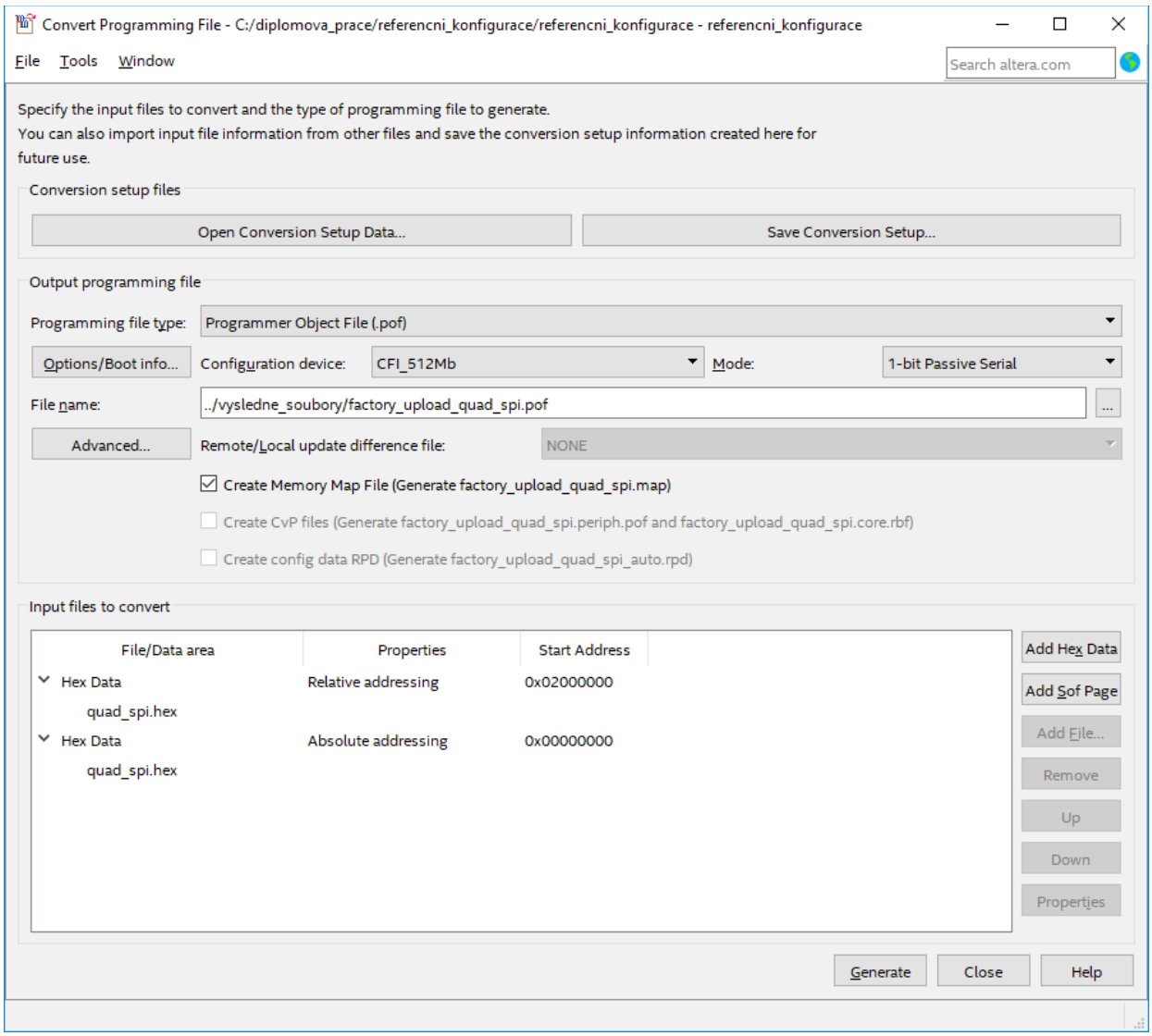
4) Kliknutím na tlačítko s názvem „Add Hex Data“ je otevřen formulář z názvem „Add Hex data“. Ve formuláři není nutné editovat další parametry a stačí zadat pouze umístění souboru s programem procesoru NIOS II pro referenční konfiguraci. Umístění souboru je nutné zadat do pole s názvem „Hex file“. Tento soubor s názvem „quad_spi.hex“ je uložen v hlavním adresáři projektu *Referencni_konfigurace => software => Refererencni_konfigurace => mem_init.*

5) Kliknutím na tlačítko „Open“ a následným kliknutím na tlačítko „OK“ ve formuláři „Add Hex Data“ je soubor vložen do nástroje Convert Programing file.

6) Kliknutím na tlačítko s názvem „Add Hex Data“ je otevřen formulář z názvem „Add Hex data“. Ve formuláři je nutné zadat adresu, kde bude uložen program procesoru NIOS II pro aplikační konfiguraci číslo 1 a aplikační konfiguraci číslo 2. Tato adresa byla zvolena v kapitole 12.5.1 na hodnotu 0x0200 0000 (jedná se o polovinu adresního rozsahu QSPI paměti, ve které je program uložen). Adresu ve formuláři je možné zadat vybráním možnosti „Relative addressing“ a následně vybráním možnosti „Set start address“. Dále je nutné zadat umístění souboru s programem procesoru NIOS II pro aplikační konfiguraci číslo 1 a aplikační konfiguraci číslo 2. Tento soubor s názvem „quad_spi.hex“ je uložen v hlavním adresáři projektu *aplikacni_konfigurace_1 => software => aplikacni_konfigurace_1 => mem_init*.

7) Do pole s názvem „File name“ je nutné zadat cestu a název výsledného souboru s příponou .pof. Název výstupního souboru byl zvolen jako „factory_upload_quad_spi.pof“ a je uložen v adresáři se všemi projekty v podadresáři s názvem „vysledne_soubory“.

8) Na obr. 13.20 je znázorněno nastavení nástroje Convert Programing File pro vygenerování souboru s názvem „factory_upload_quad_spi.pof“. Kliknutím na tlačítko „Generate“ je výstupní soubor vygenerován.



Obr. 13.20 Generování výsledného souboru pro paměť QSPI

13.4.2 Soubor pro vzdálené přehrání

Jedná se soubor s příponou .rpd, který je nahrán do interní flash paměti hradlového pole OCF. Soubor obsahuje aplikační konfiguraci číslo 2, která je odeslána ze zařízení MASTER přes rozhraní UART do zařízení SLAVE. Konfigurace je následně uložena do interní flash paměti (do části CFM1 a CFM2) hradlového pole. K vygenerování výsledného souboru slouží následující postup.

1) Kliknutím na tlačítko „File“ a následným vybráním možnosti „Convert Programming Files“ je spuštěn nástroj pro generování finálního souboru.
2) V poli s názvem „Mode“ je vybrána možnost „Internal Configuration“.
3) V okně s názvem „Input files to convert“, vybráním řádky s popisem „Page_0“ a následným

kliknutím na tlačítko „Add File“ je spuštěn průvodce, ve kterém je nutné vybrat umístění vygenerovaného souboru s názvem „referencni_konfigurace.sof“. Tento soubor prezentuje referenční konfigurační obraz, který bude uložen do části CFM0 paměti OCF. Soubor se nachází v hlavním adresáři projektu s názvem „referencni_konfigurace“ v podadresáři „output_files“. Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Convert Programing file.

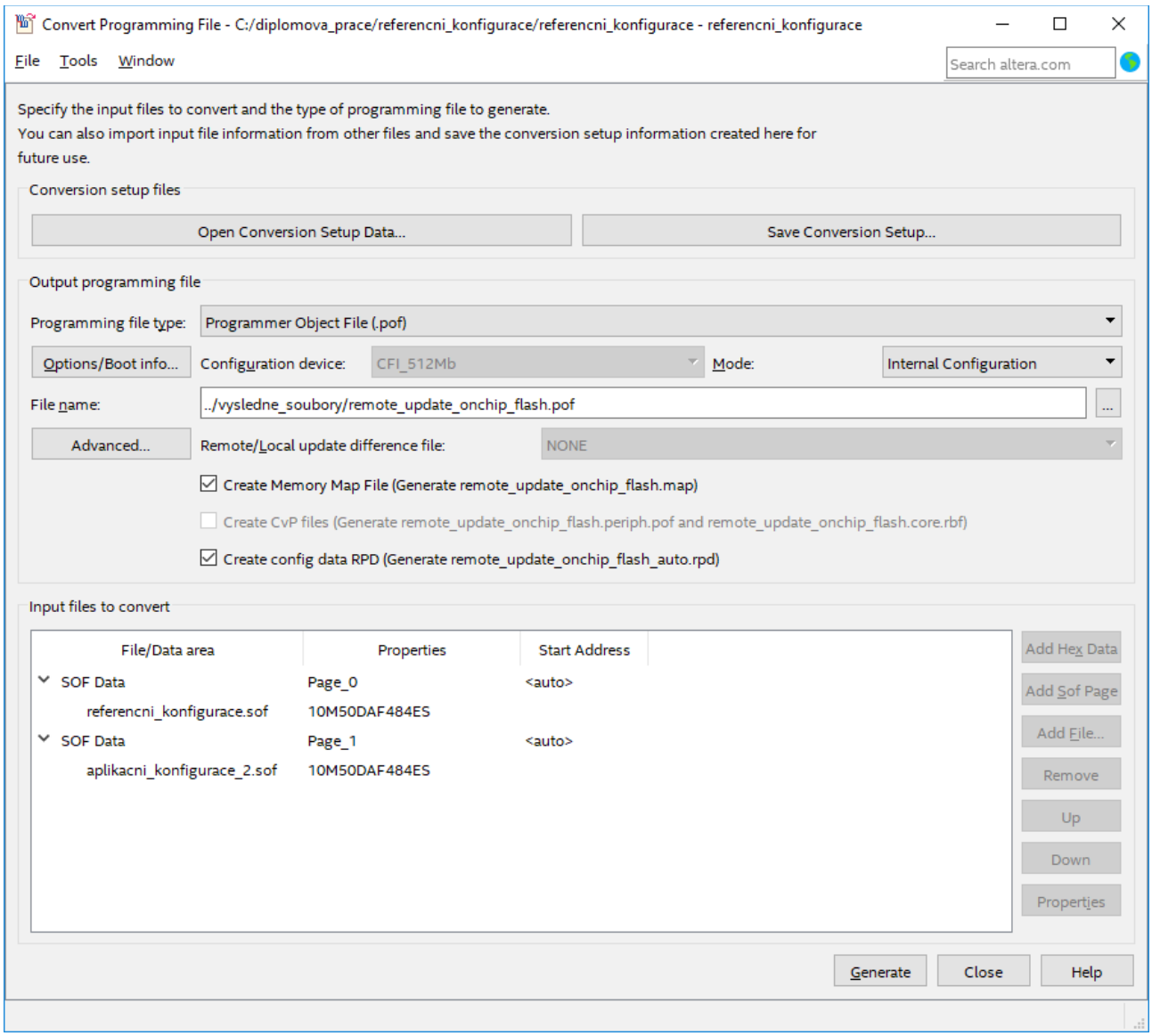
4) Kliknutím na tlačítko s názvem „Add Sof Page“ se v okně s názvem „Input files to convert“ objeví nový řádek s popisem „Page_1“.

5) Vybráním řádky s popisem „Page_1“ a následným kliknutím na tlačítko „Add File“ je spuštěn průvodce, ve kterém je nutné vybrat umístění vygenerovaného souboru s názvem „aplikacni_konfigurace_2.sof“. Tento soubor prezentuje aplikační konfigurační obraz číslo 2, který bude uložen do části CFM1 a CFM2 paměti OCF. Soubor se nachází v hlavním adresáři projektu s názvem „aplikacni_konfigurace_2“ v podsložce „output_files“. Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Convert Programing file.

6) Pro vygenerování výsledného souboru ve formátu .rpd je nutné vybrat možnost „Create config data RPD“.

7) Do pole s názvem „File name“ je nutné zadat cestu a název výsledného soboru s příponou .pof. Název výstupního souboru byl zvolen jako „remote_update_onchip_flash.pof“ a je uložen v adresáři se všemi projekty v podadresáři s názvem „vysledne_soubory“.

8) Na obr. 13.21 je znázorněno nastavení nástroje Convert Programing File pro vygenerování souboru s názvem „remote_update_onchip_flash.pof“. Kliknutím na tlačítko „Generate“ je vygenerován výstupní soubor s příponou .pof a zároveň .rpd.



Obr. 13.21 Generování výsledného souboru pro vzdálené přehrání paměti OCF

14 Zavaděč pro nahrání QSPI paměti

Jak již bylo řečeno výše, externí QSPI paměť slouží k uložení programu pro procesor NIOS referenční aplikační konfigurace, aplikační konfigurace číslo 1 a aplikační konfigurace číslo 2. Paměť lze nahrát externím programátorem nebo je nutné vytvořit vhodný zavaděč. Jelikož je QSPI paměť obtížné z vývojového kitu vyjmout, bylo nutné vytvořit vhodný zavaděč, díky kterému není nutné paměť z vývojového kitu vyjímat. V následujících podkapitolách je popsán postup, jakým bylo postupováno pro vytvoření zavaděče QSPI paměti.

14.1 Založení nového projektu v prostředí Quartus

Postup pro založení nového projektu ve vývojovém prostředí Quartus je shodný s postupem v kapitole 13.1.1. Rozdíl je pouze v názvu projektu a v názvu adresáře projektu. Název tohoto projektu byl zvolen jako „zavadec“ a je uložen v adresáři se shodným názvem.

14.2 Realizace v prostředí Platform Designer

Zavaděč pro QSPI paměť byl realizován v nástroji Platform Designer, který je součástí vývojového prostředí Quartus. Pro realizaci zavaděče nástrojem Platform Designer bylo postupováno následujícím způsobem.

1) Kliknutím na tlačítko ve vývojovém prostředí Quartus a názvem „Tools“ a následným vybráním možnosti „Platform Designer“ je spuštěn nástroj Platform Designer.
2) Pomocí IP katalogu byla vyhledána a následně vložena komponenta Parallel Flash Loader Intel FPGA IP a byly nastaveny parametry komponenty tak, jak je uvedeno v kapitole 12.5.1.
3) Kliknutím na tlačítko „File“ a následně na „Save As“ je projekt v nástroji Platform Designer uložen pod názvem „zavadec“ v adresáři projektu se shodným názvem.
4) Kliknutím na tlačítko „Generate“ a následně na „Generate HDL“ je zobrazen formulář s nastavením generování výstupních souborů. Není potřeba žádné dodatečné nastavení parametrů a kliknutím na tlačítko „Generate“, proběhne generování výstupních HDL souborů projektu.

Pokud generování výstupních souborů proběhlo bezchybně, je možné nástroj Platform Designer opustit.

14.3 Návrh projektu ve vývojovém prostředí Quartus

Výstupní soubory, které byly vygenerovány nástrojem Platform Designer je nyní nutné vložit do projektu. Pro účely použití zavaděče, je nutné výstupní soubory vygenerované nástrojem Platform Designer vhodným způsobem modifikovat. Pro realizaci zavaděče bylo postupováno následujícím způsobem.

1) Kliknutím na rozbalovací pole v okně „Project Navigator“ ve vývojovém prostředí Quartus a následným vybráním možnosti „Files“, jsou zobrazeny zdrojové soubory projektu. Jelikož zatím nebyly vloženy žádné soubory, bude okno prázdné.

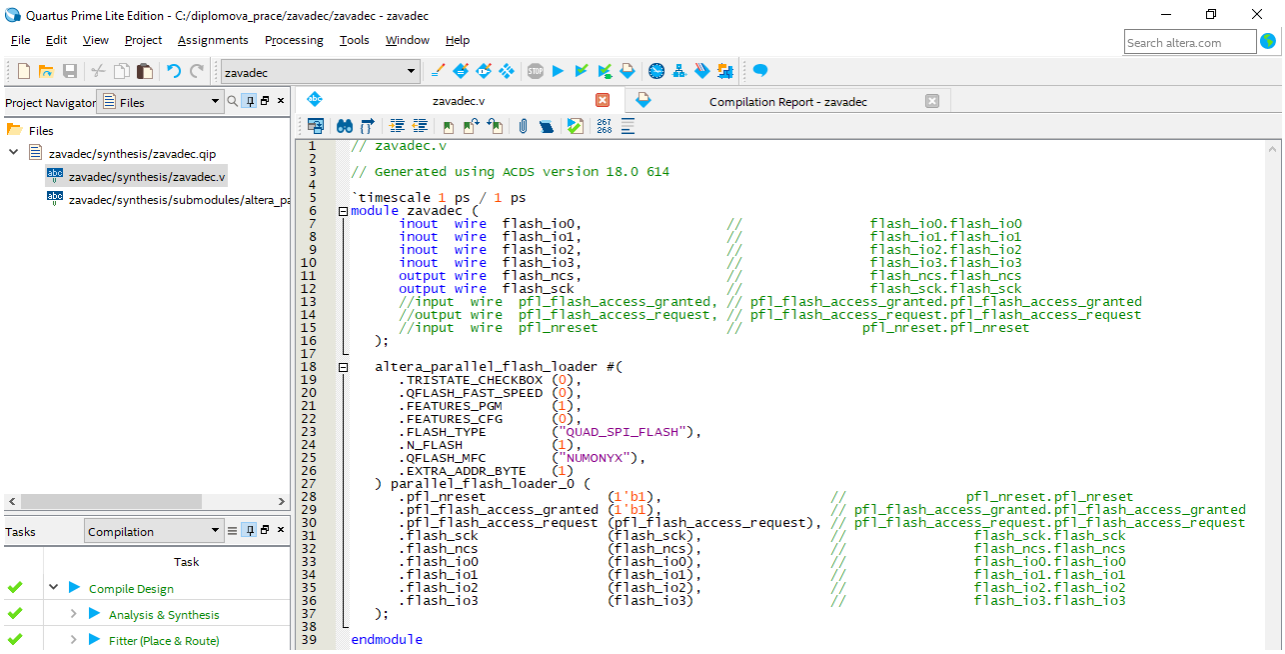
2) Kliknutím na tlačítko „Assignments“ a následným vybráním možnosti „Settings“ je otevřen průvodce nastavením projektu.

3) V kategorii „Files“ otevřeného formuláře je potřeba vyplnit cestu k souboru s názvem „zavadec.qip“. Tento soubor obsahuje odkazy na všechny HDL soubory, které byly vytvořeny nástrojem Platform Designer. Tento soubor se nachází ve složce s názvem „synthesis“, uložené ve složce s projektem, který byl vytvořen pomocí nástroje Platform Designer.

4) Vybráním souboru a následným kliknutím na tlačítko „Open“ je soubor vložen do otevřeného formuláře. Následným kliknutím na tlačítko „OK“ jsou všechny HDL soubory vloženy do projektu v prostředí Quartus.

5) V okně s názvem „Project Navigator“ je nyní potřeba nastavit HDL soubor, který reprezentuje vrcholovou entitu. V tomto případě se jedná o soubor s názvem „zavadec.v“. Kliknutím pravým tlačítkem myši nad tímto souborem a následným vybráním možnosti „Set as Top-Level Entity“ je daná entita určena jako vrcholová.

6) Pro správnou funkci zavaděče, je nutné modifikovat soubor s názvem „zavadec.v“. Modifikace spočívá v nastavení signálu pfl_nreset a pfl_flash_access_granted do logické úrovně „1“ tak, jak je znázorněno na obr. 14.1. Toto nastavení vychází z dokumentu [9]. Signály pfl_nreset, pfl_flash_access_granted a pfl_flash_access_requested poté není nutné připojovat k pinům hradlového pole.



Obr. 14.1 Modifikace vrcholové entity souboru s názvem zavadec.v

7) Stisknutím klávesové zkratky „Ctrl+K“ je spuštěna analýza a syntéza celého projektu.

8) Kliknutím na tlačítko ve vývojovém prostředí Quartus s názvem „Assignments“ a následným vybráním možnosti „Pin Planner“ je spuštěn nástroj, pomocí kterého jsou jednotlivým signálům vrcholové entity přiřazeny fyzické piny hradlového pole. Čísla pinů a napěťové hladiny jednotlivých signálů jsou uvedeny v tabulce na obr. 14.2.

Název signálu vrcholové entity	Napěťová hladina	Číslo pinu na FPGA	Popis
flash_io0	3,3V LVCMOS	C6	Adress bus
flash_io1	3,3V LVCMOS	C3	Adress bus
flash_io2	3,3V LVCMOS	C5	Adress bus
flash_io3	3,3V LVCMOS	B1	Adress bus
flash_ncs	3,3V LVCMOS	C2	Chip select
flash_sck	3,3V LVCMOS	B2	Clock

Obr. 14.2 Parametry a čísla signálu pro připojení k hradlovému poli

9) Po přiřazení jednotlivých signálů fyzickým pinům hradlového pole je možné nástroj Pin Planner zavřít.

10) Stisknutím klávesové zkratky „CTRL+L“ je spuštěna kompilace celého projektu

11) Tímto je celý projekt s názvem „zavadec“ dokončen a je možné vývojové prostředí Quartus opustit.

15 Názorná ukázka

V této kapitole je uveden postup pro nahrání prvotní konfigurace (funkce factory upload). Následuje postup pro vzdálené přehrání konfigurace prostřednictvím rozhraním UART (funkce remote update). Nakonec je ověřeno, zda je zařízení schopno rozeznat chybu ve vzdáleně nahrávaném obrazu (kontrola funkčnosti) a na jednoduchém příkladu je vyzkoušena diagnostika zařízení SLAVE.

15.1 Nahrání prvotní konfigurace zařízení SLAVE

Nahrání prvotní konfigurace spočívá v nahrání výše vytvořených výsledných souborů pro interní flash paměť hradlového pole a v nahrání programu pro procesor NIOS II do externí QSPI paměti prostřednictvím rozhraní JTAG.

15.1.1 Nahrání QSPI paměti

Pro nahrání QSPI paměti zařízení SLAVE je uveden následující postup.

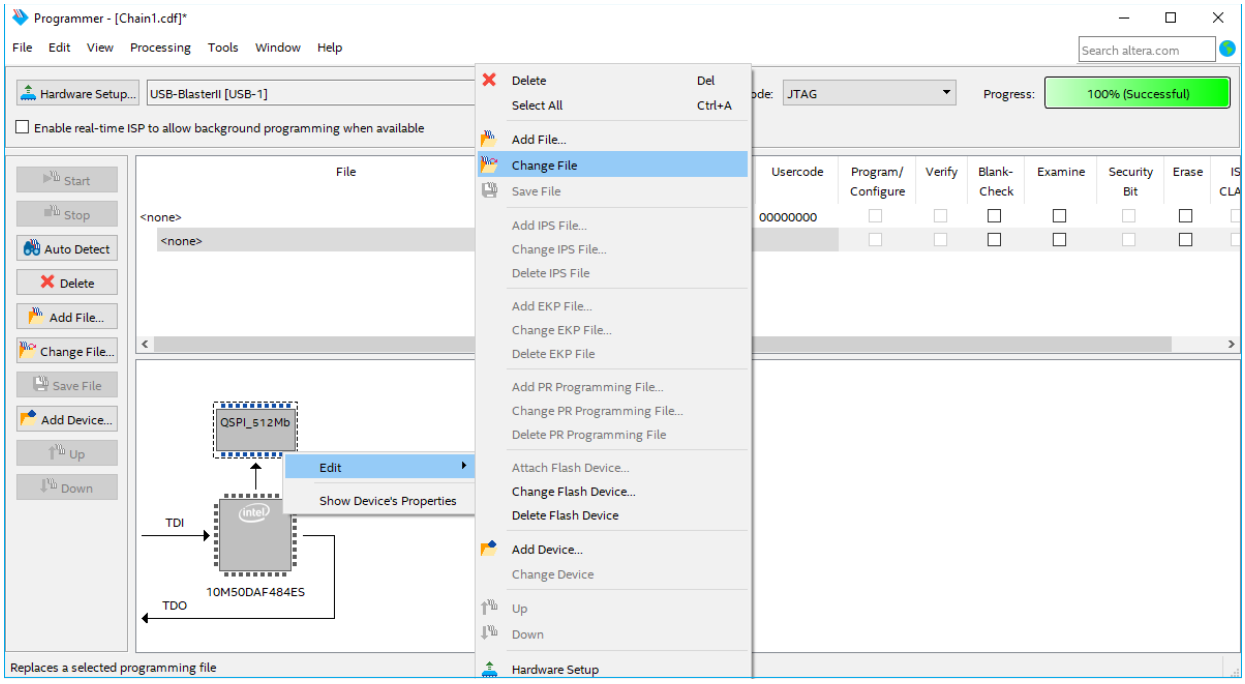
1) Vývojový kit MAX10 FPGA Development kit je připojen k PC prostřednictvím rozhraní JTAG.

2) Kliknutím na tlačítko ve vývojovém prostředí Quartus a názvem „Tools“ a následným vybráním možnosti „Programmer“ je spuštěn nástroj pro nahrání prvotní konfigurace.

3) Nejdříve je nutné nahrát externí QSPI paměť. K tomu byl vytvořen v kapitole 14 zavaděč, který je nyní nutné nahrát do zařízení SLAVE. Kliknutím na tlačítko „Add File“ a vybráním souboru s názvem „zavadec.sof“, který se nachází ve složce „output_files“ ve složce s projektem „zavadec“ a následným kliknutím na tlačítko „Open“ je tento soubor připraven na nahrání do zařízení SLAVE.

4) Kliknutím na tlačítko „Start“ je soubor nahrán do hradlového pole.

5) Kliknutím na tlačítko „Auto Detect“ je zobrazena externí QSPI paměť která bude následně nahrána – viz obr. 15.1.



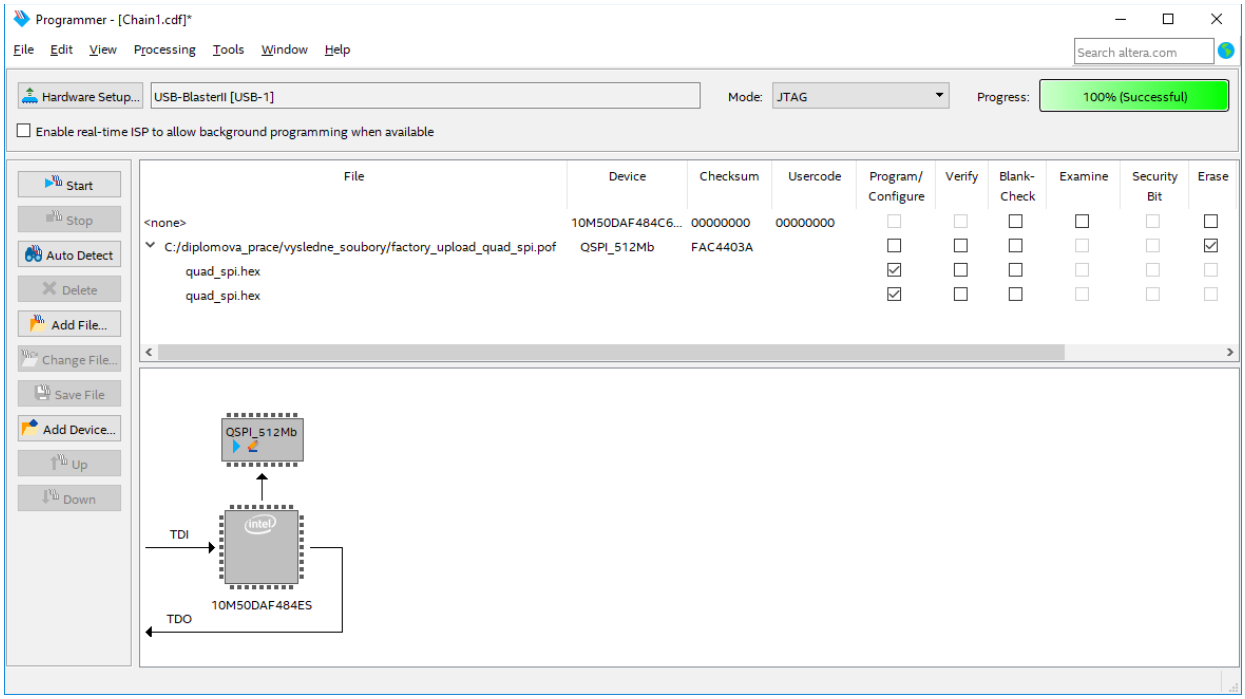
Obr. 15.1 Vybrání souboru pro nahrání QSPI paměti

6) Kliknutím pravým tlačítkem myši na zařízení QSPI_512Mb, vybráním možnosti „Edit“ a následně „Change File“ je nutné zadat umístění souboru, který obsahuje program procesoru NIOS II jak pro referenční konfiguraci, tak pro aplikační konfiguraci číslo 1 a aplikační konfiguraci číslo 2. Jedná se o soubor s názvem „factory_upload_quad_spi.pof“ a je umístěn

ve složce s názvem „vysledne_soubory“.

7) Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Programmer.

8) Nastavením dle obr. 15.2 je nejdříve QSPI paměť kompletně smazána a následně je nahrán program procesoru NIOS II jak pro referenční konfiguraci, tak pro aplikační konfiguraci číslo 1 a aplikační konfiguraci číslo 2.



Obr. 15.2 Nastavení pro nahrávání QSPI paměti

9) Kliknutím na tlačítko „Start“ je provedeno nahrání externí QSPI paměti.

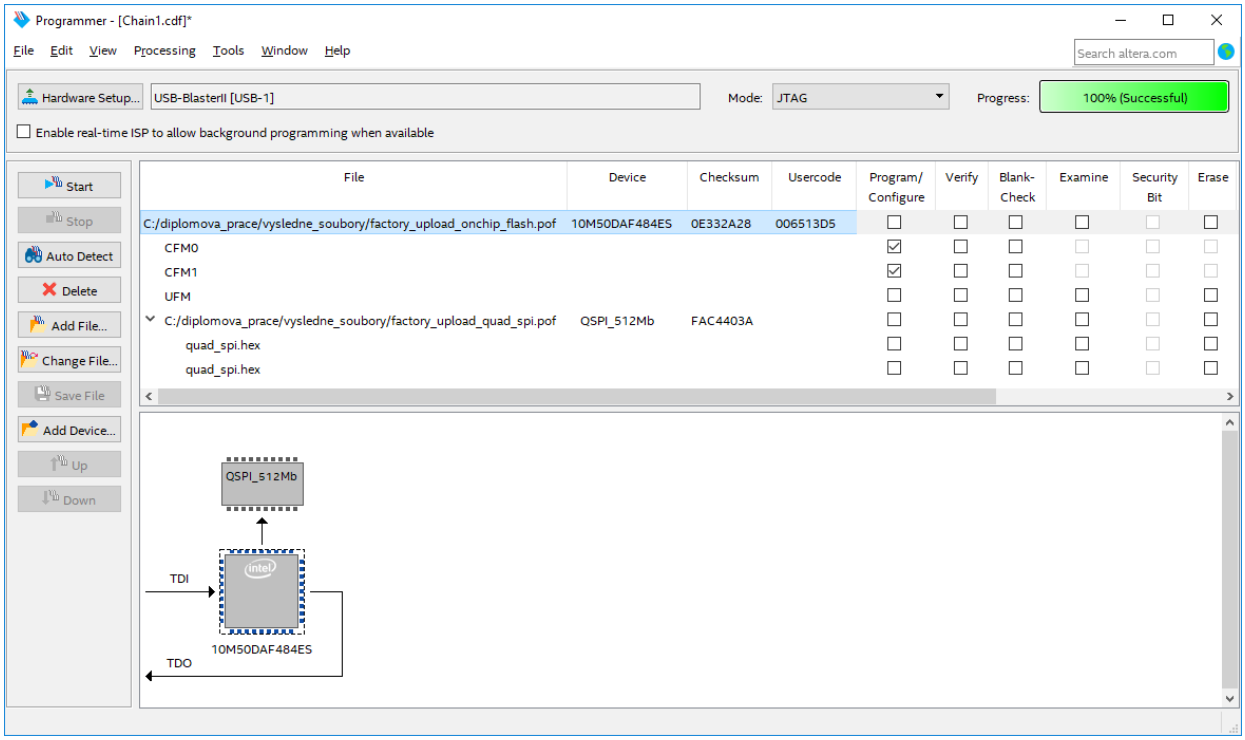
15.1.2 Nahrání interní flash paměti

Tento postup navazuje na postup z předešlé kapitoly 15.1.1. Pro nahrání interní flash paměti zařízení SLAVE je uveden následující postup.

1) Kliknutím pravým tlačítkem myši na zařízení 10M50DAF484ES (viz obr. 15.2), vybráním možnosti „Edit“ a následně „Change File“ je nutné zadat umístění souboru, který obsahuje referenční konfigurační obraz a aplikační konfigurační obraz číslo 1. Jedná se o soubor s názvem „factory_upload_onchip_flash.pof“ a je umístěn ve složce s názvem „vysledne_soubory“.

2) Kliknutím na tlačítko „Open“ je soubor vložen do nástroje Programmer.

3) Nastavením dle obr. 15.3 do interní flash paměti hradlového pole nahrán referenční konfigurační obraz a aplikační konfigurační obraz číslo 1.



Obr. 15.3 Nastavení pro nahrávání interní flash paměti hradlového pole

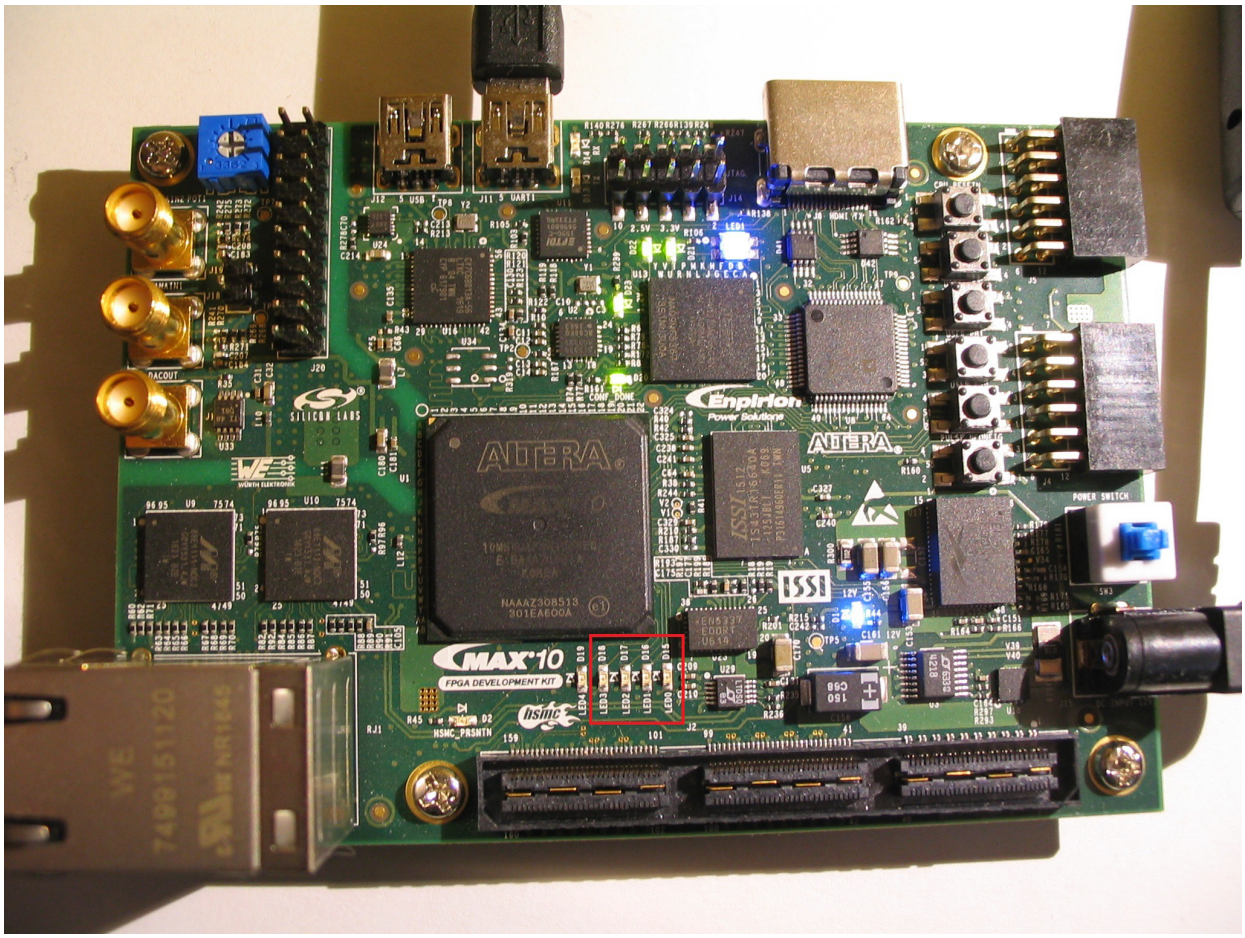
4) Kliknutím na tlačítko „Start“ je provedeno nahrání interní flash paměti hradlového pole.

15.1.3 Ověření funkčnosti

K ověření funkčnosti zařízení SLAVE bylo toto zařízení připojeno k zařízení MASTER prostřednictvím rozhraní UART. V případě zařízení MASTER se jedná o PC, které je vybaveno komunikačním terminálem, který je schopen odesílat řídící znaky a také je schopen odeslat datový soubor. Pro tyto účely byl zvolen terminál Termie, který je volně dostupný. K ověření funkčnosti je uveden následující postup.

1) Vývojový kit MAX10 FPGA Development kit (zařízení SLAVE) je připojeno k PC (zařízení MASTER) prostřednictvím rozhraní UART.

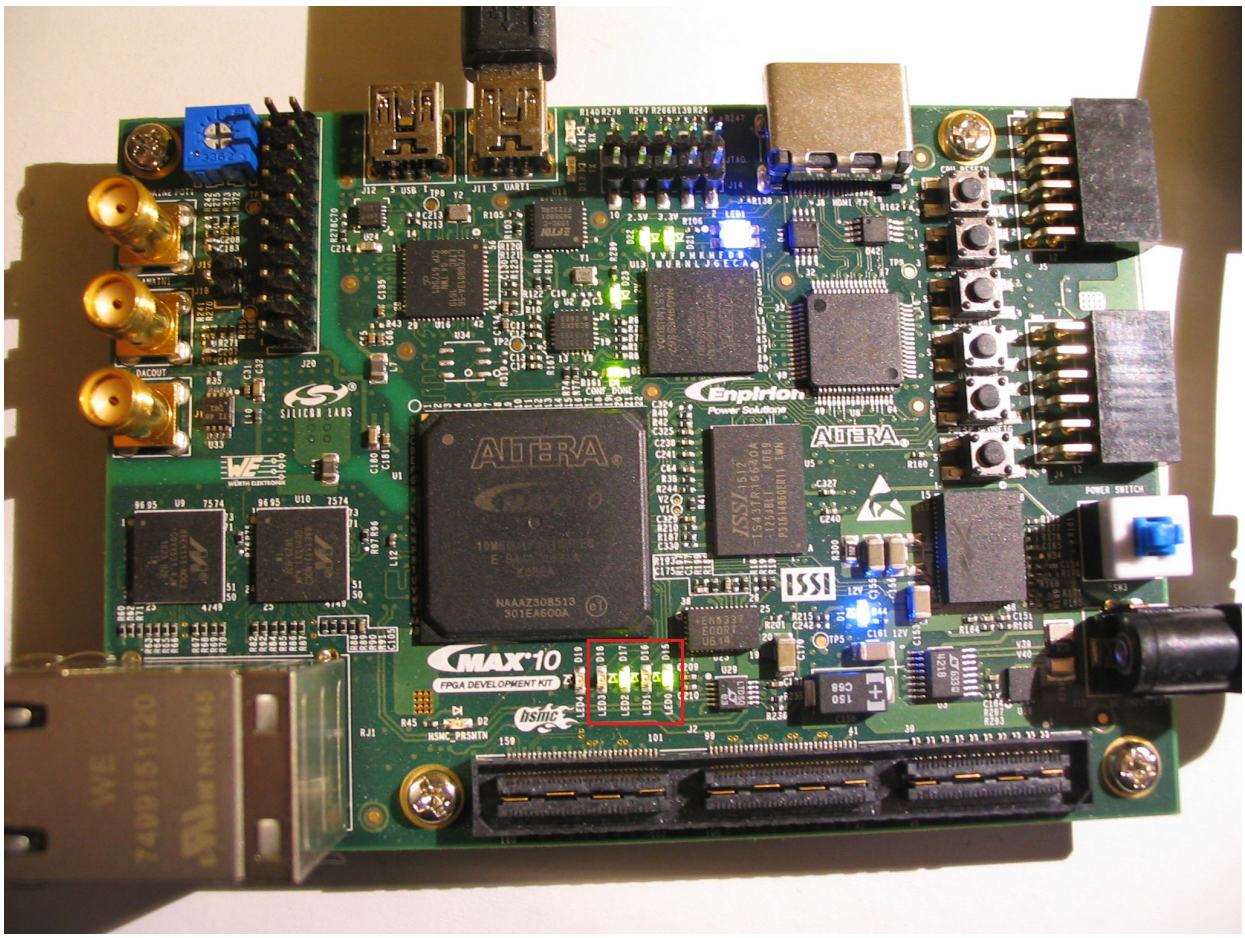
2) V zařízení SLAVE je načtena aplikační konfigurace číslo 1 – viz obr. 15.4.



Obr. 15.4 Funkce user_logic aplikační konfigurace číslo 1

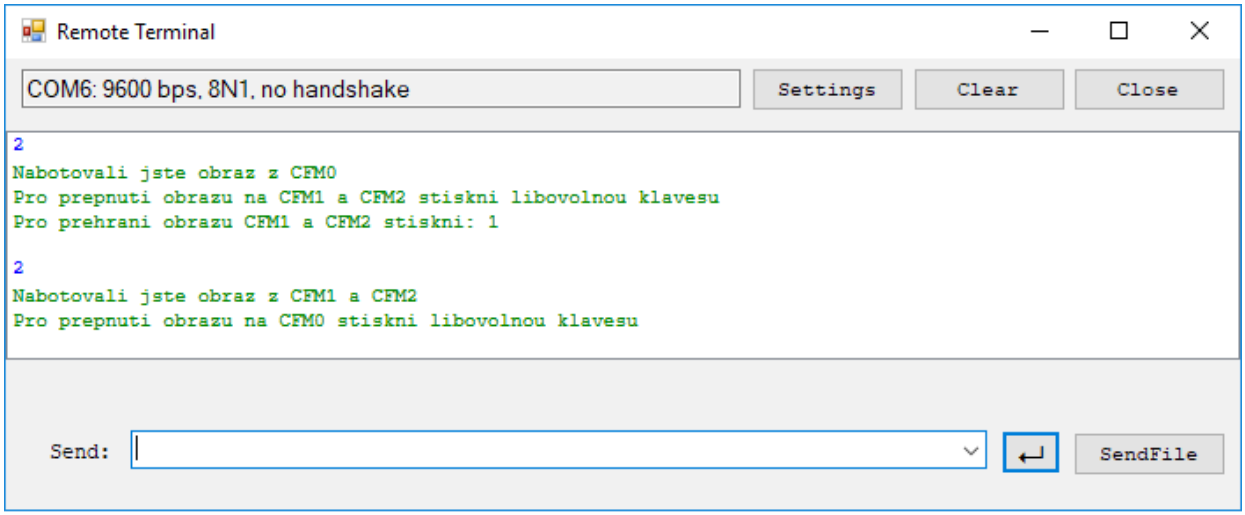
3) V PC je spuštěn komunikační terminál, kde je nastavena komunikační rychlost na 9600 bits/s a je zvolen komunikační kanál s označením COM6 – viz obr. 15.6.

4) Stisknutím libovolné klávesy, je možné přepnout na referenční konfiguraci – viz obr. 15.5



Obr. 15.5 Funkce user_logic referenční konfigurace

5) Opětovným stiskem libovolné klávesy (kromě klávesy s číslem 1) je opět možné přepnout na aplikační konfiguraci číslo 1. O tom, jaký obraz je právě načten, odesílá zařízení SLAVE informaci zařízení MASTER prostřednictvím rozhraní UART – viz obr. 15.6.

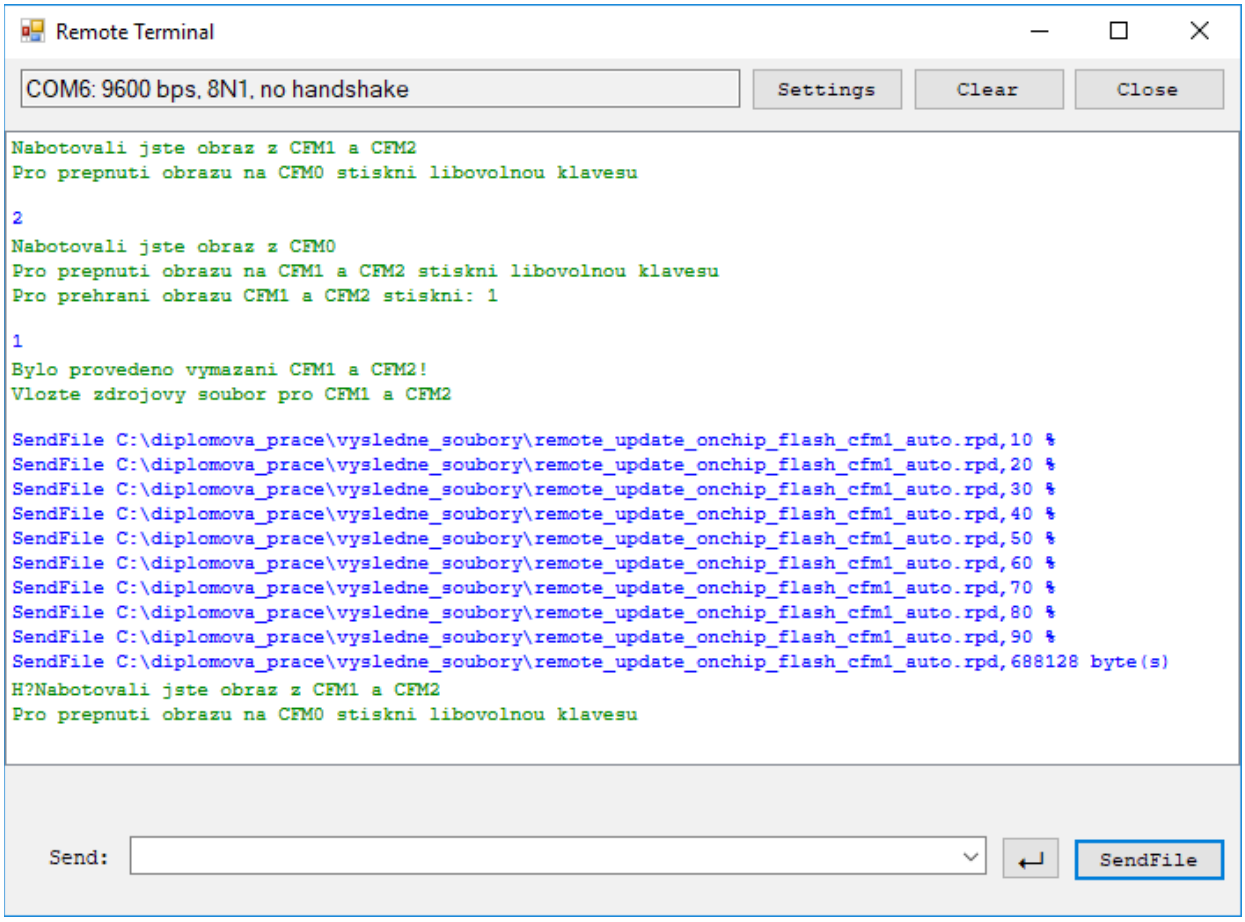


Obr. 15.6. Komunikační terminál zařízení MASTER

15.2 Vzdálené přehrání konfigurace zařízení SLAVE

Vzdálené přehrání konfigurace spočívá v přehrání stávající aplikační konfigurace číslo 1 aplikační konfigurací číslo 2 ze strany zařízení MASTER prostřednictvím rozhraní UART. Pro přehrání aplikační konfigurace číslo 1 aplikační konfigurací číslo 2 je uveden následující postup.

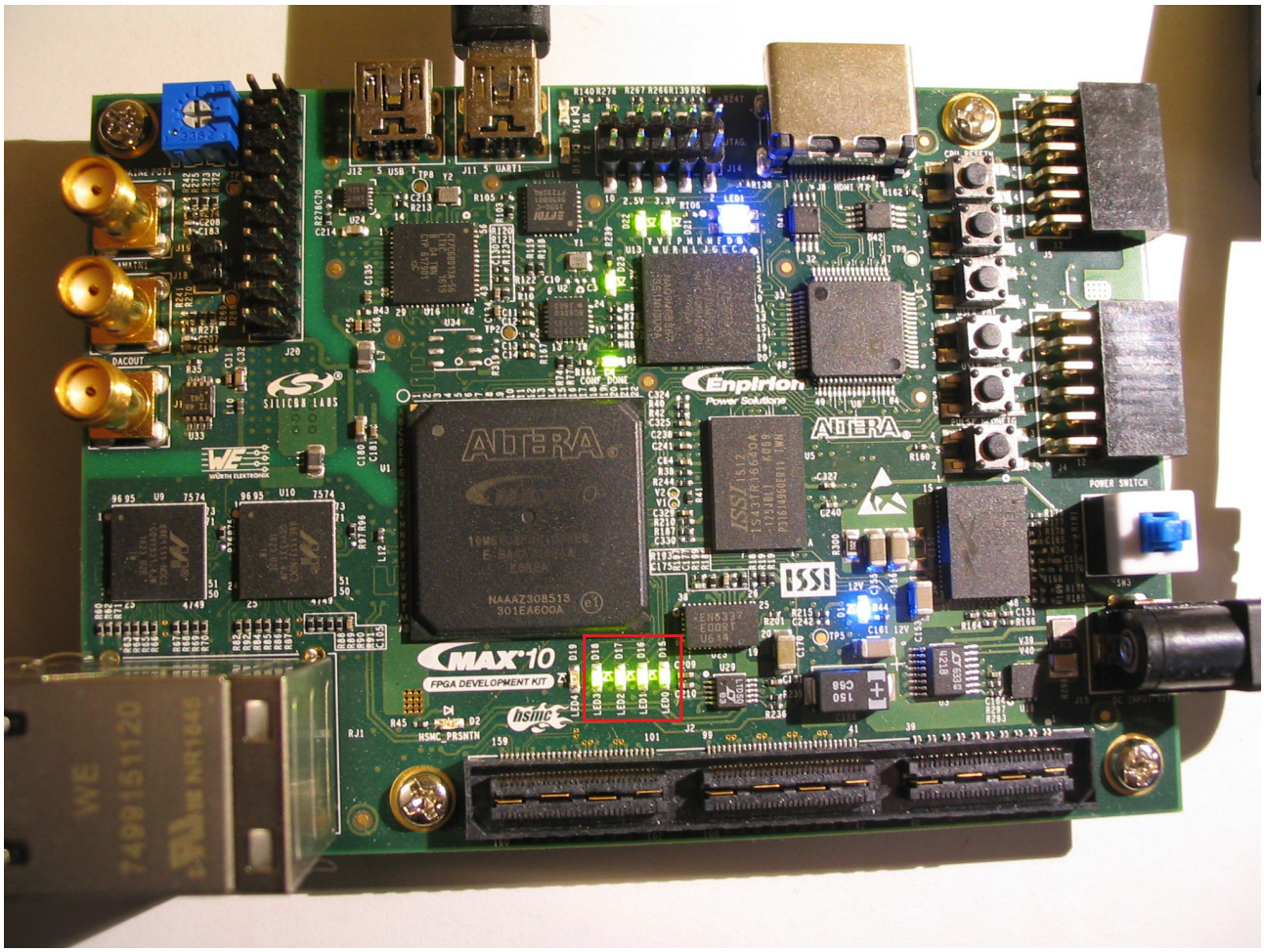
1) Vývojový kit MAX10 FPGA Development kit (zařízení SLAVE) je připojeno k PC (zařízení MASTER) prostřednictvím rozhraní UART.
2) V zařízení SLAVE je načtena aplikační konfigurace číslo 1.
3) V PC je spuštěn komunikační terminál, kde je nastavena komunikační rychlost na 9600 bits/s a komunikační je zvolen kanál s označením COM6 – viz obr. 15.7.
4) Stisknutím libovolné klávesy, je nutné přepnout na referenční konfiguraci.
5) Stisknutím klávesy s číslem „1“, je vymazána část interní flash paměti hradlového pole CFM1 a CFM2. Následně je zařízení SLAVE připraveno na příjem aplikačního obrazu číslo 2, který je odeslán zařízením MASTER prostřednictvím komunikačního terminálu – viz obr. 15.7.
6) Kliknutím na tlačítko v komunikačním terminálu s názvem „SendFile“ je následně nutné zadat umístění souboru, který obsahuje aplikační konfigurační obraz číslo 2. Jedná se o soubor s názvem „remote_update_onchip_flash_cfm1_auto.rpd“ a je umístěn ve složce s názvem „vysledne_soubory“.
7) Kliknutím na tlačítko „Open“ je zahájeno vysílání souboru s aplikační konfigurací číslo 2.
8) Po dokončení vysílání aplikační konfigurace číslo 2, zařízení SLAVE přepne na nově nahranou aplikační konfiguraci číslo 2 – viz obr. 15.7.



Obr. 15.7 Vysílání souboru aplikační konfigurace číslo 2

15.2.1 Ověření funkčnosti

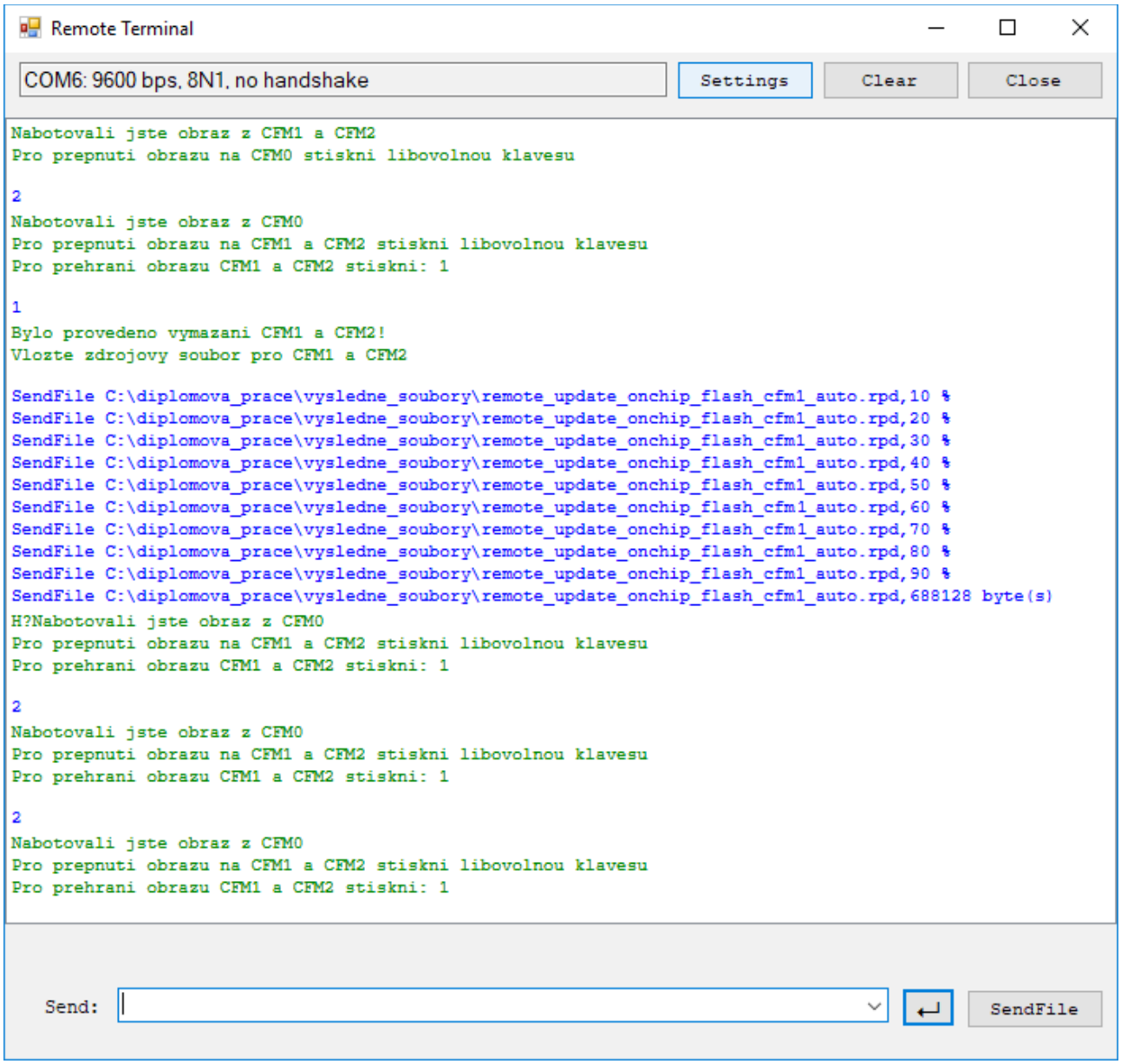
Ověření funkčnosti je možné provést stejným způsobem, jako tomu bylo v kapitole 15.1.3 s tím rozdílem, že nyní bude přepínáno mezi referenčním konfiguračním obrazem a aplikačním obrazem číslo 2 – viz obr. 15.8. Tím je ověřeno, že zařízení MASTER úspěšně odesláno novou konfiguraci zařízení SLAVE prostřednictvím rozhraní UART.



Obr. 15.8 Funkce user_logic aplikační konfigurace číslo 2

15.3 Kontrola funkčnosti vzdáleně přehrávané konfigurace

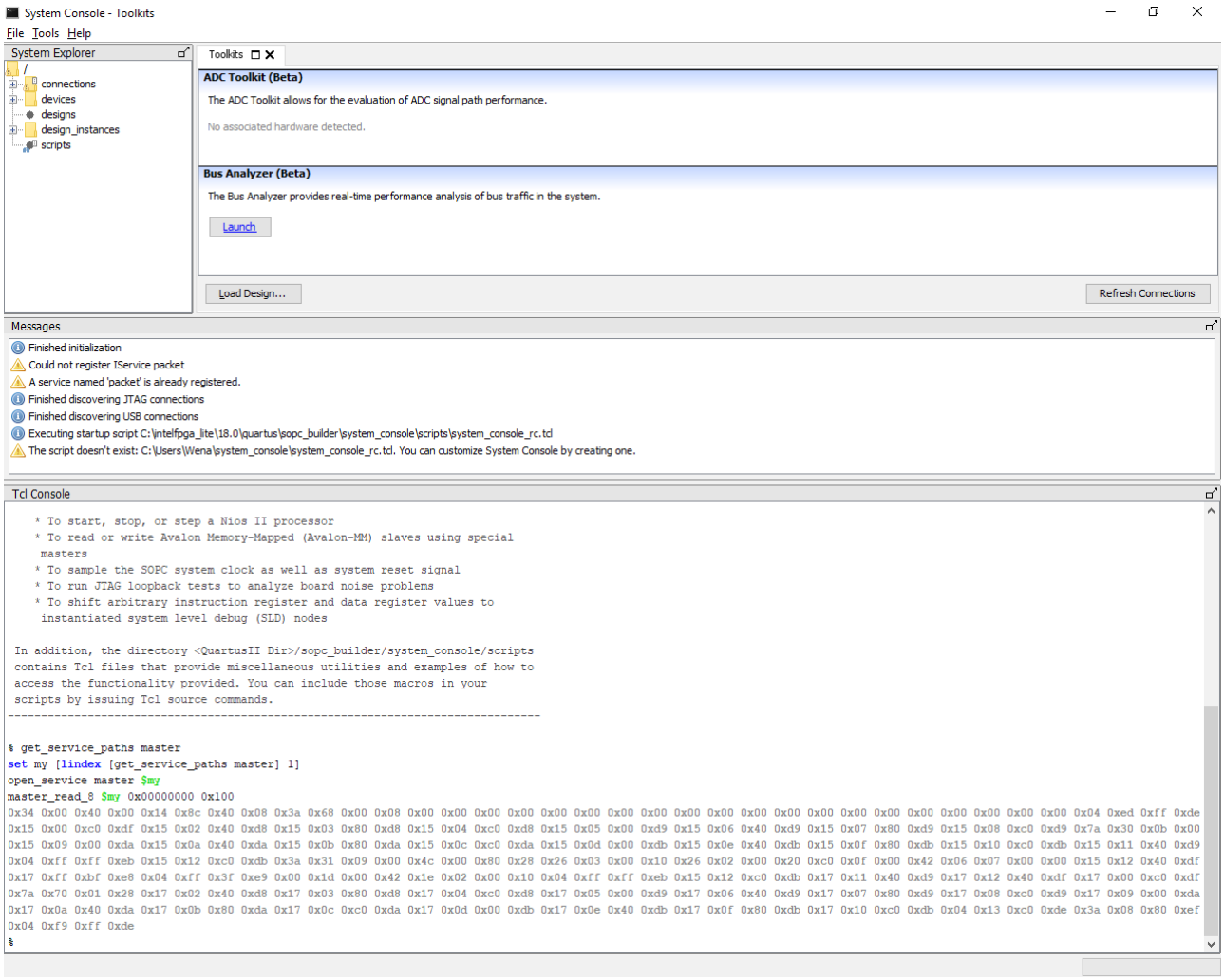
V případě bezchybného vzdáleného přehrání, je mezi konfiguračními obrazy možné libovolně přepínat. V případě, že je soubor pro vzdálené přehrání poškozen, případně vznikne nějaká nahodilá chyba při vzdáleném přehrávání nové aplikační konfigurace, zařízení SLAVE je schopno tuto skutečnost rozpoznat a na poškozenou aplikační konfiguraci nedovolí přepnout. Tento stav lze navodit záměrným poškozením souboru s názvem remote_update_onchip_flash_cfm0_auto.rpd (přepsáním či umazáním části tohoto souboru). Poškozený soubor s aplikační konfigurací je pomocí komunikačního terminálu úspěšně odeslán, ale nelze na něj přepnout – viz obr. 15.9. Tato chyba lze následně opravit vygenerováním nového souboru s aplikační konfigurací číslo 2 (kapitola 13.4.2), který je pomocí funkce remote update znovu odeslán ze strany zařízení MASTER do zařízení SLAVE prostřednictvím rozhraní UART (kapitola 15.2).



Obr. 15.9 Vysílání poškozeného souboru aplikační konfigurace číslo 2

15.4 Diagnostika zařízení SLAVE

Diagnostikovat zařízení SLAVE lze pomocí nástroje System Console, který je obsahem vývojového prostředí Quartus. Tento nástroj umožňuje přistupovat přes rozhraní JTAG k registrům procesoru NIOS a k registrům všech komponent, které jsou k rozhraní JTAG připojeny. Tyto registry je možné vyčítat nebo do nich přímo zapisovat, což je velmi výhodné při vývoji nebo diagnostice zařízení při chybném vykonávání funkce. Na obr. 15.10 je znázorněno, jakým způsobem lze např. vyčíst část obsahu z externí QSPI paměti. Možnosti zacházení s nástrojem System Console jsou podrobně popsány v dokumentu [13].



Obr. 15.10 Vyčtení obsahu QSPI paměti pomocí systémové konzole

Závěr

V první části práce byly popsány možnosti komunikace a komunikačních rozhraních mezi elektrickými zařízeními trakčních vozidel. Byly zde popsány standardy, rozhraní a sběrnice, které jsou v dnešní době běžně používána pro komunikaci mezi elektronickými zařízeními trakčních vozidel. Byla zde popsána také rozhraní, která slouží ke komunikaci mezi mikrokontrolérem a jeho perifériemi v rámci jednoho elektronického zařízení.

V druhé části práce byl popsán návrh systému obsahující hradlové pole, jehož konfiguraci je možné vzdáleně přehrát. V kapitole 10 byl popsán koncept celého systému, který definuje vlastnosti zařízení MASTER, SLAVE a rozhraní, pomocí kterého jsou mezi sebou schopna zařízení komunikovat. V kapitole 12 byla popsána architektura systému, která popisuje jednotlivé funkce a činnosti bloků. Ke každému bloku architektury je popsána jeho praktická realizace. Následuje kapitola 13, ve které byla popsána praktická realizace systému na základě architektury ve vývojovém prostředí Quartus. Byl zde popsán návrh referenční konfigurace, aplikační konfigurace číslo 1 a aplikační konfigurace číslo 2 pro hradlové pole. Ke konci práce bylo ověřeno naplnění všech bodů zadání. Byla ověřena funkce referenční konfigurace a aplikační konfigurace číslo 1, která byla následně vzdáleně přehrána prostřednictvím rozhraní UART aplikační konfigurací číslo 2. Dále byla otestována schopnost zařízení SLAVE rozeznat poškozený konfigurační obraz a také byla otestována možnost diagnostiky zařízení. Všechny zdrojové soubory, které jsou obsahem této práce jsou v elektronické podobě v její příloze.

Systém podporující vzdálené přehrání přes komunikační rozhraní, má v praxi řadu výhod. Jedna z hlavních výhod je snížení nákladů na servis v případě, že byla při vývoji zařízení objevena chyba, případně je nutné zařízení doplnit o novou funkcionalitu a zařízení je již v provozu. Pokud by totiž zařízení nepodporovalo možnost vzdáleného přehrání a bylo umístěno na velmi obtížně přístupném místě, bylo by z důvodu přehrání nové konfigurace zařízení potřeba např. demontovat větší část celku, ve kterém je zařízení umístěno. To může být velmi náročné jak časově, tak finančně. Tuto nevýhodu lze odstranit tím, že zařízení podporuje vzdálené přehrání konfigurace. Pro přehrání konfigurace v tomto případě postačí, pokud je k danému zařízení přivedena komunikační linka, prostřednictvím které je nová konfigurace do zařízení SLAVE odeslána ze strany zařízení MASTER. Díky tomu není nutná demontáž zařízení a přehrání konfigurace je provedeno během několika minut. Nevýhodou systému pro vzdálenou konfiguraci navrženého v této práci je, že zabírá velkou část logických elementů hradlového pole, které by mohly být použity pro jiné, např. aplikační účely.

Použitá literatura

[1] Polarities for Differential Pair Signals (RS-422 and RS-485) - B&B Electronics. *Industrial Networking Solutions - Serial, Ethernet & USB | B&B Electronics - B&B Electronics* [online]. Copyright ©2018 [cit. 08.05.2019]. Dostupné z: http://www.bb-elec.com/Learning-Center/All-White-Papers/Serial/%E2%80%A2-Polarities-for-Differential-Pair-Signals-%28RS-422.aspx

[2] HW server představuje - Sériová linka RS-232 | Vývoj.HW.cz. *Vývoj.HW.cz | Vše o elektronice a programování* [online]. Copyright © 1997 [cit. 08.05.2019]. Dostupné z: https://vyvoj.hw.cz/rozhrani/hw-server-predstavuje-seriova-linka-rs-232.html

[3] Přenos dat po linkách RS485 a RS422 | Vývoj.HW.cz. *Vývoj.HW.cz | Vše o elektronice a programování* [online]. Copyright © 1997 [cit. 08.05.2019]. Dostupné z: https://vyvoj.hw.cz/teorie-a-praxe/dokumentace/prenos-dat-po-linkach-rs485-a-rs422.html

[4] RS 485 & 422 | Vývoj.HW.cz. *Vývoj.HW.cz | Vše o elektronice a programování* [online]. Copyright © 1997 [cit. 08.05.2019]. Dostupné z: https://vyvoj.hw.cz/teorie-a-praxe/dokumentace/rs-485-422.html

[5] *Archív kurzů FPF SU* [online]. Copyright © [cit. 08.05.2019]. Dostupné z: https://archiv.elearning.fpf.slu.cz/pluginfile.php/16403/mod_resource/content/0/prezentace/site03_ethernet.pdf

[6] *Domo*vske stranky uzivatelu [online]. Copyright © [cit. 08.05.2019]. Dostupné z: http://home.zcu.cz/~dudacek/NMS/Seriova_rozhrani.pdf

[7] *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright © [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug-max10m50-fpga-dev-kit.pdf

[8] N25Q512A83GSF40F. Micron Technology, Inc. [online]. Copyright ©2019 Micron Technology, Inc. All rights reserved. Information, products, and [cit. 08.05.2019]. Dostupné z: https://www.micron.com/products/nor-flash/serial-nor-flash/part-catalog/

n25q512a83gsf40f

[9] Parallel Flash Loader Intel FPGA IP Core User Guide. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

sss1411439280066.html

[10] Embedded Design Handbook. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

iga1446487888057.html#fwl1479912385825

[11] Intel MAX 10 Clocking and PLL User Guide. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

mcn1395213337540.html#mcn1396269818879

[12] Intel MAX 10 FPGA Configuration User Guide. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

sss1393988509894.html#sss1397549479122

[13] System Console - Altera Wiki. [online]. Copyright © 2018 Intel Corporation. The material in this wiki page or document is provided AS [cit. 08.05.2019]. Dostupné z: https://fpgawiki.intel.com/wiki/System_Console

[14] AN 741: Remote System Upgrade for MAX 10 FPGA Devices over UART with the Nios II Processor. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/

programmable/documentation/sss1430185325021.html#sss1430812492914

[15] Embedded Peripherals IP User Guide. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

sfo1400787952932.html#nnb1488475996333

[16] Intel MAX 10 User Flash Memory User Guide. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/documentation/

vgo1395753117436.html#vgo1397656193368

[17] Nios® II Processor - Nios® II Processors Support. *Intel | Data Center Solutions, IoT, and PC Innovation* [online]. Copyright ©Intel Corporation [cit. 08.05.2019]. Dostupné z: https://www.intel.com/content/www/us/en/programmable/products/processors/support.html