VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

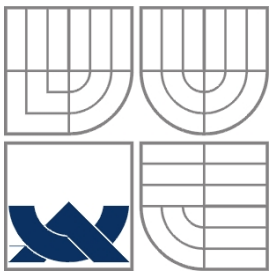
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

VÝVOJ SÉRIOVÝCH KOMUNIKAČNÍCH PERIFERIÍ POMOCÍ FPGA

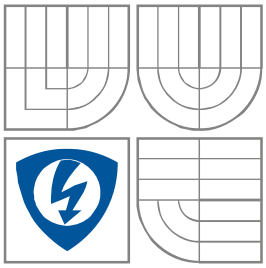
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE PAVEL ŠTRAUS
AUTHOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A
KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

VÝVOJ SÉRIOVÝCH KOMUNIKAČNÍCH PERIFERIÍ POMOCÍ FPGA

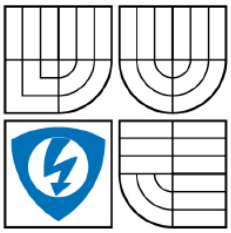
SERIAL COMMUNICATION PERIPHERIES DEVELOPMENT IN FPGA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Pavel Štraus
AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Frýza, Ph.D.
SUPERVISOR

BRNO, 2009



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor
Elektronika a sdělovací technika

Student: Pavel Štraus ***ID:*** 72796
Ročník: 3 ***Akademický rok:*** 2008/2009

NÁZEV TÉMATU:

Vývoj sériových komunikačních periferií pomocí FPGA

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s volným vývojovým prostředím ISE WebPACK pro návrh a simulaci aplikací obvodů PLD firmy Xilinx. Navrhněte periférii realizující sériový asynchronní přenos UART. Respektujte veškeré modifikace přenosového rámce tohoto systému a symbolové rychlosti. Navrženou periférii odlaďte v simulátoru a na vývojové desce.

Navrhněte periférii realizující I2C (Inter-Integrated Circuit) komunikaci běžně používanou v mikroprocesorové technice. Navrženou periférii odlaďte pomocí simulátoru a na vývojové desce.

DOPORUČENÁ LITERATURA:

[1] PINKER, J. POUPA M. Číslicové systémy a jazyk VHDL. Praha: BEN – technická literatura, 2006. 352 stran. ISBN 80-7300-198-5

[2] KOLOUCH, J. Programovatelné logické obvody. Elektronické texty přednášek a počítačových cvičení. Brno: FEKT VUT v Brně, 2007

[3] Atmel Corporation. AVR 8-Bit RISC. [online]. 2008 – [cit. 21. dubna 2008]. Dostupné na WWW: http://www.atmel.com/products/avr/

Termín zadání: 9.2.2009 ***Termín odevzdání:*** 5.5.2009

Vedoucí práce: Ing. Tomáš Frýza, Ph.D.

prof. Dr. Ing. Zbyněk Raida
předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Bakalářská práce je zaměřena na dvě periferie. První vytváří ze vstupních paralelních dat výstupní sériový signál podle standardu UART. Tento sériový signál je tvořen start bitem, datovými bity, paritou a stop bitem. Počet datových bitů je proměnný a to v závislosti na nastavení řídících signálů Dat0 a Dat1. Dále je možné zabezpečení pomocí paritního bitu, kde je možné volit mezi sudou či lichou paritou. Po tomto bitu již následuje stop bit či dva stop bity. Druhá periferie realizuje sběrnici I2C. Jedná se o dvouvodičovou sběrnici, která používá vodiče SDA a SCL. Vodič SDA slouží pro přenos dat, pokud je hodinový signál SCL v logické 1. Při použité 3,3 V logice je hodnota log. 1 rovna právě tomuto napětí. Pokud nedochází ke komunikaci je vodič SDA i SCL v úrovni log. 1. Komunikace je zahájena podmínkou startu a ukončena podmínkou stop. Periferie byly naprogramovány pomocí programovacího jazyka VHDL a po naprogramování byly funkce ověřeny pomocí simulace ve volném vývojovém prostředí Xilinx ISE WebPACK. Následně proběhly realizace pomocí FPGA Virtex-II XC2V1000. Správná funkce obou periferií je zachycena na oscilogramech.

Klíčová slova

FPGA, UART, I2C, VHDL, Periférie, ISE WebPACK

Abstrakt

This bachelor's thesis is about two peripheries. First periphery creates from input parallel signals one output serial signal. This serial signal contains a start bit, the next are data bits, parity bit and stop bit or two stop bits. Data bits are variables. It is mean their count is set with two input signals called Dat0 and Dat1. We can secure data bits with parity bit. Of course we have choice between even parity bit or odd parity bit. After parity bit there is one stop bit or there are two stop bits. Second periphery realizes I2C bus. This communication is between two devices. First device is called master and creates the communication with second device called slave. For communication there are two bidirectional lines. The first line is called SDA, which is a serial data line and second line is a serial clock line called SCL. Communication begins with a start condition. That means line SDA go from high to low while SCL is high and communication is terminate with a stop condition. That means line SDA go from low to high while SCL is high. The peripheries are programming in VHDL language and implemented in FPGA device. After successful simulation in free software ISE WebPACK the peripheries was realized in the development board V2MB1000 with device XC2V1000.

Key words

FPGA, UART, I2C-bus, VHDL, Periphery, ISE WebPACK

ŠTRAUS, P. *Vývoj sériových komunikačních periferií pomocí FPGA*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 34s. Vedoucí bakalářské práce Ing. Tomáš Frýza, Ph.D.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Vývoj sériových komunikačních periferií pomocí FPGA jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

V Brně dne 5. června 2009 .....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Tomáši Frýzovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 5. června 2009 .....
podpis autora

Obsah bakalářské práce

Seznam obrázků

Seznam tabulek

1. Úvod

Cílem bakalářské práce je softwarové i hardwarové řešení periferií, které realizují sériovou asynchronní komunikaci podle standardu UART a komunikaci po sběrnici I2C. Samotná realizace komunikace I2C spočívá, pomocí programovacího jazyka VHDL, naprogramovat obvod typu master, který řídí komunikaci po sběrnici I2C. Tento obvod je vytvořený ze tří samostatných obvodů. Prvním obvodem je předdělička hodinového signálu, která je zdrojem pro linku SCL. Druhý obvod je pojmenován VlastniIO a zde je možné volit kolik datových bytů je odesláno pomocí sběrnice I2C. Jako poslední dílčí obvod je zde vlastní realizace sběrnice I2C. U sériové asynchronní komunikace byl využit pouze jeden obvod. Po naprogramování periferií byla ověřena funkce pomocí simulace, která je součástí vývojového prostředí. Po odsimulování bylo možné nahrát vytvořený program pomocí JTAG do FPGA a vyzkoušet, zda nedošlo v návrhu či simulaci k chybě.

Bakalářská práce je rozdělena do 7 kapitol, kdy v daných kapitolách je uvedena problematika či teoretické poznatky pro správné nastavení a realizace bakalářské práce.

V kapitole 2 je uveden standard UART. Je zde uvedena struktura datového rámce, která je velice podstatná pro samotnou funkci sériové komunikace. Bez daného datového rámce by byla sériová komunikace chaotická. Standard UART zavádí do asynchronní komunikace systém přenosu dat.

V kapitole 3 je vysvětlen základ samotné komunikace po sběrnici I2C. Jak je možné realizovat zapojení obvodu master a dvou obvodů slave. Následně je zde vysvětlena funkce linek SDA a SCL. Také je zde uvedena struktura adresního rámce, která je velice podstatná pro navázání samotné komunikace po sběrnici I2C.

Kapitola 4 je zaměřena na obvody FPGA, především na sérii obvodů Virtex-II. Periferie je realizovaná pomocí obvodu Virtex-II XC2V1000. Je zde uvedena architektura tohoto obvodu. Obvody FPGA jsou tvořeny bloky, které jsou uspořádány do celků a na nich je realizována funkce obvodu, která je v našem případě popsána pomocí jazyka VHDL.

Kapitola 5 pojednává o programovacím jazyku VHDL. Je zde zmínka o vzniku tohoto jazyka a následně také náhled na software, ve kterém je bakalářská práce realizována. Je zde uvedena také zmínka jakým způsobem jsou programovány obvody FPGA pomocí tohoto prostředí.

V 6 kapitole následuje již samotná realizace periferií, včetně navržení a základního nastavení vstupních signálů. Dále je ukázáno, jakým způsobem probíhala samotná realizace v programu ISE WebPACK a jsou zde zobrazeny výsledky pomocí zachycených oscilogramů.

Poslední kapitola obsahuje zhodnocení bakalářské práce a dosažené výsledky.

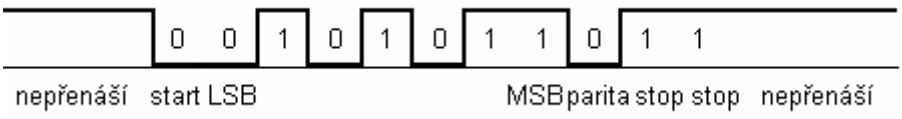
2. UART

Jedná se o universální asynchronní přenos dat mezi vysílačem a přijímačem. Základem asynchronní komunikace je, že data mohou být vysílána v libovolném okamžiku, to znamená že nejsou synchronizované pravidelným signálem, který přesně určí v jakém okamžiku se signály vysílají. Proto u asynchronní komunikace nevysíláme informace o synchronizaci. Pro bezchybnou komunikaci mezi vysílačem a přijímačem byl stanoven komunikační protokol, který obsahuje danou strukturu datového rámce. [4]

2.1. Datový rámec

Stanovení stejného datového rámce jak pro přijímač tak pro vysílač je velice důležité u asynchronního přenosu dat. Bez přesné struktury by bylo pro přijímač nemožné správně dekódovat poslaná data. Dále je nutné zajistit stejnou symbolovou rychlost jak pro přijímač, tak pro vysílač. Pokud tyto hodnoty budou nastavené na různou hodnotu, pak se stane, že vysílač data vyšle, ale přijímací strana není schopná datový rámec přijmout. Dojde tedy k chybnému přenosu dat.

Popis datového rámce je následující. Jako první bit při zahájení přenosu musí být start bit, který má délku pouze jednoho bitu, a to vždy v úrovni log. 0. Po tomto bitu následují datové bity. Počet datových bitů je proměnný a může nabývat od 5 do 8 bitů. Vždy jsou ale v takovém pořadí, že první je vysílán nejméně významný bit v datovém slově, označován jako LSB. Jako poslední datový bit je vyslán nejvýznamnější bit, který je označen jako MSB. Po datových bitech může být v rámci bit paritní, který nám slouží pro kontrolu dat. Přijímací strana ví, že se přenáší paritní bit a ví, jestli jde o sudou či lichou paritu. Potom z přijatých dat vyhodnotí paritní bit a zkontroluje, zda paritní bit, který přijal a vypočtený bit jsou shodné. Pokud ano, byla data přijata správně. Pokud se liší, je nutné poslat celé slovo znovu. Poslední část datového rámce tvoří stop bit či dva stop bity, které jsou vždy v úrovni log. 1. Příklad datového rámce je zobrazen na obrázku 1.



Obrázek 1: Struktura asynchronního rámce

2.2. Paritní bit

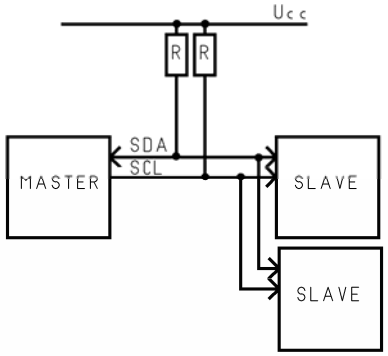
Jedná se o redundantní informaci, která je přiložena na konci datového rámce a to mezi nejvýznamnější datový bit a stop bit. Není tedy nutnou součástí datového rámce, proto můžeme paritní bit vynechat, přičemž je nutné opět nastavit na přijímací i vysílací straně, zda se bude paritní bit přenášet. Jako další informaci, kterou je nutné nastavit, pokud je paritní bit přenášen, je způsob určení paritního bitu. Toto je možné určit použitím sudé či liché parity.

V principu se tímto bitem určí, zda je počet log. 1 v datovém slově sudý či lichý, a to následujícím způsobem. Pokud je nastavení sudé parity a výsledný součet logických úrovní 1 je sudý, pak výsledný bit v datovém rámci je v log. 0. Pokud zůstane nastavení sudé parity, ale součet logických úrovní 1 je lichý, pak výsledný bit v datovém rámci je nastavený na logickou úroveň 1.

Při nastavení liché parity se jedná o stejný princip. Opět se bere součet logických úrovní 1 v celém datovém rámci a pokud je výsledek sudý, pak výsledný bit v datovém rámci je v logické úrovni 1, pokud je ovšem výsledný součet logických úrovní 1 lichý, pak výsledný paritní bit v datovém rámci je nastavený na logickou úroveň 0. [2]

3. Sběrnice I2C

Byla vyvinuta firmou Philips Semiconductor. Jedná se o dvouvodičovou sběrnici, která je plně duplexní. Je tedy potřeba pouze dvou vodičů, přičemž označení těchto vodičů je SDA a SCL. Každé zařízení připojené k této sběrnici musí mít označení master nebo slave. Master je označeno zařízení, které zajišťuje komunikaci, zatímco slave je adresované zařízení, které komunikuje s obvodem typu master. Je možné adresovat až 128 zařízení používající sběrnici I2C, přičemž každé zařízení má unikátní adresu. Tato adresa je vyjádřena sedmibitovým číslem, proto je zde maximální počet unikátních adres 128. Komunikaci vždy zahajuje zařízení označené jako master. Toto zařízení také vysílá synchronizační hodinový signál SCL. Zapojení obvodu master a dvou obvodů slave je zobrazeno viz. obrázek 2. V tomto zapojení jsou odpory R, které plní funkci pull-up rezistorů. [5] [6]



Obrázek 2: Zapojení master-slave

Na počátku komunikace je vyslána podmínka startu. Následuje adresa zařízení, se kterým chceme zahájit komunikaci. Adresní rámec je vysílán od nejvýznamnějšího bitu adresy po nejméně významný bit, který je bit R/W. Jedná se o určení, zda budeme data číst z obvodu slave, nebo zda budeme data zapisovat. Po potvrzení příjmu adresy obvodem slave následují datové bity, které jsou vysílány také v pořadí od MSB k LSB a po zaslání každého datového bytu je obvodem slave potvrzeno přijetí bitem ACK. Počet přenesených datových bytů není omezený počtem, ale je vždy vysílán ve formátu 8 datových bitů a potvrzení příjmu těchto bitů. Komunikaci vždy končí obvod master zasláním podmínky stop. Komunikace je také ukončena když se nezdaří rozpoznání adresy a tím nepotvrzení bitu ACK. [5] [6]

3.1. SDA a SCL linka

Linka SDA je obousměrná, umožňující tok dat z obvodu master do obvodu slave, ale i z obvodu slave do obvodu master. Linka SCL slouží pro hodinový signál, který je vysílaný obvodem master. Každé zařízení je připojeno na tyto linky pomocí pull-up rezistorů, které zaručí vysokou úroveň linky, pokud není komunikace mezi obvody. Vysoká úroveň na těchto linkách je dána použitou logikou a tedy napájecím napětím. Pro vysokou úroveň je ve standardu stanovena hodnota 0,7 napájecího napětí a pro nízkou úroveň je stanovena hodnota 0,3 napájecího napětí. Komunikační rychlost sběrnice je závislá na volbě přenosového módu viz. tabulka 1. [5]

Tabulka 1 - Rychlost přenosu I2C

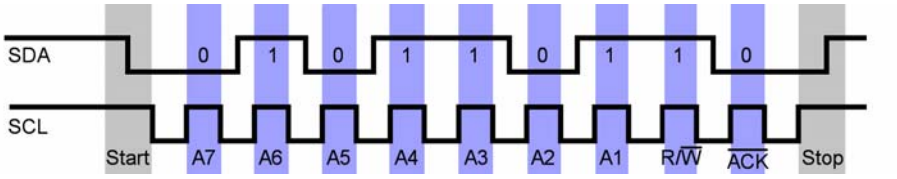
Standard mode	100 kbps
Fast mode	400 kbps
Fast mode plus	1 Mbps
High-speed mode	3.4 Mbps

3.2. Komunikační protokol

Před zahájením komunikace master ověří, zda je klidový stav na sběrnici. Pokud ano může začít komunikaci vysláním podmínky startu. Pokud se vyskytuje pouze jedno zařízení označeno jako master, pak nenastane kolize na sběrnici, protože zařízení master vždy zahajuje komunikaci.

Podmínka pro zahájení komunikace je splněna, když datová linka SDA přejde z hodnoty logická 1 do hodnoty logická 0, zatímco hodinový signál SCL je stále v hodnotě logická 1. Po startu následuje adresa zařízení, se kterým chce master komunikovat. Adresní rámec je složen z 9 bitů, kde prvních sedm tvoří adresu a jako první je vysílán nejvýznamnější bit této unikátní adresy. Po odeslání těchto sedmi bitů následuje bit s označením R/W, který určí, zda obvod master bude data do obvodu slave zapisovat nebo z obvodu slave číst. Pokud chceme data zapisovat, pak volíme možnost write a tento bit bude mít hodnotu log. 0. Jestliže budeme data z obvodu číst, pak bit R/W bude mít hodnotu logická 1. Jsou také vyhrazené adresy, které se používají pro zvláštní účely. Proto standard uvádí maximální teoretický počet adres 128, ale ve skutečnosti je tento počet menší. Poslední bit adresního rámce již nevysílá obvod master. Jedná se o potvrzovací bit, který je zaslán obvodem slave po rozpoznání své unikátní sedmibitové adresy. Obvod master nastaví linku do vysoké úrovně či stavu vysoké impedance a ověří hodnotu na lince SDA, když je SCL rovno log. 1. Pokud linka SDA bude mít hodnotu log. 0, pak obvod slave rozpoznal svou adresu a komunikace pokračuje. V opačném případě je zaslána obvodem master podmínka stop a komunikace je ukončena. [5]

Pokud budeme mít obvod slave, který bude mít unikátní sedmibitovou adresu 0b0101101, budeme chtít navázat komunikaci a číst z tohoto obvodu data, pak výsledný adresní rámec musí mít tvar 0b01011011. Obvod slave adresu zachytí, rozpozná a potvrdí bit ACK, tedy dá linku SDA do hodnoty log. 0 viz. obrázek 3.



Obrázek 3: Komunikace po I2C

Po vyslání adresy a příjmu potvrzení existující adresy mohou být zaslané datové rámce, které obsahují stejně jako adresní rámec devět bitů, přičemž se skládají z jednoho byte dat a jednoho bitu, který slouží pro potvrzení příjmu. Data jsou vysílána stejným způsobem jako adresa a to od nejvýznamnějšího bitu po nejméně významný bit.

3.3. Komunikační módy

Obvod master může působit jako vysílač, tak i jako přijímač. Také obvody slave mohou působit jako přijímač i jako vysílač. Toto pravidlo ale neplatí u všech zařízení. Existují totiž obvody, které umožňují pouze danou komunikaci podle druhu svého účelu. Mezi komunikaci master vysílač a slave přijímač můžeme zařadit např. obvody LCD, které data pouze přijímají a není možné z LCD data vysílat do obvodu master. Komunikace s obvodem LCD je zobrazena na obrázku 4. Použité symboly M a S určují obvod, který vysílá. Symbol M znamená obvod master a symbol S určuje, že vysílá obvod slave. [5] [6]

Start	Adresa	ACK	Data	ACK	Data	ACK	Stop
M	M	S	M	S	M	S	M

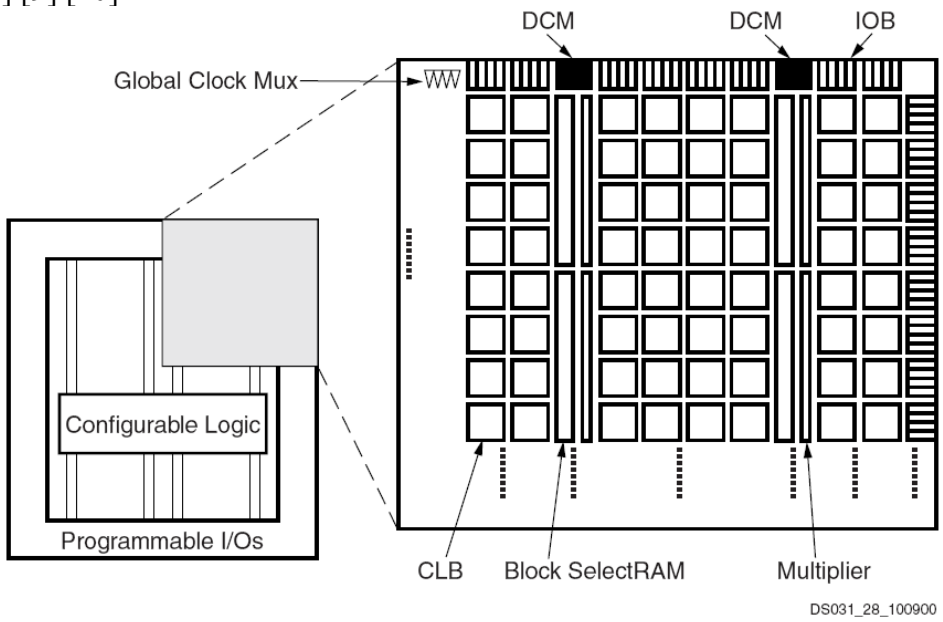
Obrázek 4: Komunikace LCD po I2C

4. FPGA

Pod označením FPGA je možné si představit programovatelné hradlové pole. Tyto obvody jsou schopné realizovat číslicové systémy či podsystémy integrované na jednom čipu. Tyto systémy lze jednoduše aktualizovat pomocí jiných programů, které je možné nahrát do obvodu FPGA. [2]

4.1. Architektura

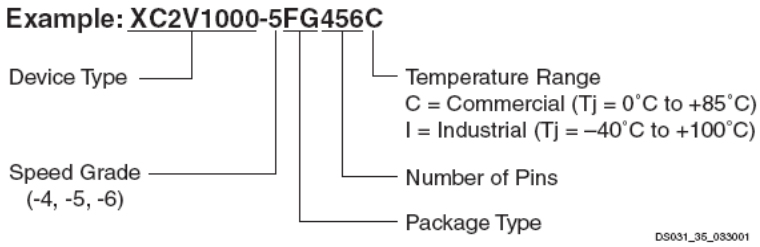
Hradlová pole určují samotnou architekturu FPGA, která obsahuje různé programovatelné bloky. Nejobecnější struktura je dána použitím čtyř typů bloků. Tyto bloky se pravidelně opakují. Jedná se o vstupně-výstupní bloky, označované jako IOB, které zprostředkovávají tok dat mezi výstupem FPGA a interní logikou. Vstupně-výstupní bloky jsou umístěné po celém pouzdru FPGA. Každý z těchto pinů podporuje obousměrný tok dat, přičemž obsahují DDR registry, které slouží pro určení aktivního vstupu či výstupu daného pinu. Druhým blokem je CLB, ve kterém jsou vytvořeny logické funkce. Tento blok je složen z paměťových prvků, většinou RAM paměti, které jsou založené na LUT tabulkách. Do těchto tabulek je možné implementovat všechny logické funkce daných vstupních signálů. Jsou zde také další speciální komponenty, které umožňují efektivnější a snadnější vytvoření častých zapojení. Dalším důležitým blokem je paměť RAM, která je velikosti 18 kbit. Čtvrtým blokem jsou násobičky a obvody pro správu hodinového signálu, označované jako DCM. Správce digitálních hodin poskytuje kalibraci, distribuci, zpoždění, násobení, dělení a fázově posunuté signály. Dále je nutná komunikace mezi těmito bloky, proto musí být navrženy tak, aby výsledné propojení bylo možné realizovat v souvislosti se zadanou funkcí viz. obrázek 5. Je zde zavedeno hierarchické uspořádání, které umožňuje vysokorychlostní a efektivní návrh. Architektura FPGA nám tedy vyvodí nevýhodu a to především v propojovací struktuře, kde je velké časové zpoždění. Toto zpoždění se nám projeví v analýze a především v její realizaci. Proto je v dnešní době takřka nemožné analyzovat zapojení bez počítače s příslušným softwarem. [2] [9] [10]



Obrázek 5: Bloky FPGA obvodu Virtex-II (Obrázek převzat z [9])

4.2. Značení FPGA

V dnešní době je na trhu velké množství obvodů FPGA, proto je důležité zavedené označení od výrobce, abychom již z názvu byly schopni určit použité zapouzdření, počet pinů a další základní vlastnosti vybraného obvodu. Výrobce dodržuje zavedené značení obvodů viz. obrázek 6. Periferie jsou realizovány pomocí FPGA XC2V1000-4FG456C. [10]



Obrázek 6: Značení obvodu Virtex-II (Obrázek převzat z [9])

Typ obvodu má označení XC2V1000, což znamená, že je použit obvod Virtex série II. Další důležité údaje o samotném obvodu následují za tímto označením. Jako první je zde uvedena samotná rychlost obvodu. Xilinx poskytuje obvody s rychlostí -4, -5 a -6, kde hodnota -6 označuje maximální možnou rychlost a tím tedy nejvyšší výkon. [9]

Další údaj, který je uvedený za rychlostí obvodu je typ zapouzdření, v našem případě je použito FG zapouzdření. Za tímto údajem následuje počet pinů. Obvod tedy obsahuje 456 pinů a jeho velikost je podle dokumentace 23 mm^2. [9]

Posledním údajem, který je součástí názvu je teplotní rozsah. Pokud nastane situace, kdy tento rozsah nebude dodržen, obvod bude vykazovat chyby, které nelze ovlivnit ani předpokládat. Xilinx vyrábí FPGA s označením I a C, kde C jsou obvody pro standardní využití a I jsou obvody pro průmyslové využití. [9]

5. Jazyk VHDL

5.1. Vznik jazyka

Jazyk VHDL vznikl v USA díky projektu ministerstva obrany Spojených států amerických. Projekt byl nazván VHSIC a na jeho realizaci se podílely také firmy IBM a Texas Instrument, které v roce 1985 své výsledky publikovaly a tím položily základy pro jazyk VHDL. Následující rok organizace IEEE převzala vytvořené standardy jazyka a začala jej upravovat. Tímto se jazyk stal otevřeným standardem a ne jazykem, který by byl vyvíjen pouze jednou firmou a následné využití a rozvíjení pro jiné firmy by bylo velice obtížné a značně nákladné. Označení se také přizpůsobilo organizaci IEEE a začalo se mluvit o IEEE Standard VHDL Language. Do současné doby již bylo vydáno 5 standardů IEEE a šestá verze je očekávána. V roce 1999 byla zveřejněna organizací IEEE nadstavba označená jako VHDL-AMS. Výhoda této nadstavby je, že umožňuje simulaci analogových a smíšených systémů. Poslední verze IEEE by měla mít především přednosti s prací s knihovními balíky a také možnost šifrování zapsaného kódu podle již známých algoritmů. [1] [6] [10]

Jazyk VHDL tedy vznikl pro návrh a simulaci vysokorychlostních integrovaných obvodů. Tyto obvody jsou většinou velmi rozsáhlé a jejich návrh a odsimulování pomocí jednotlivých integrovaných obvodů v laboratořích by bylo velice složité, či takřka nemožné. [1] [10]

5.2. Vývojové prostředí Xilinx ISE WebPACK

Vývojové prostředí Xilinx ISE WebPACK slouží pro popis a simulaci programovatelných logických obvodů firmy Xilinx, které se neustále vyvíjí podle požadavků na práci s obvody. Jedná se tedy o proprietární software firmy Xilinx, který je ale volně šiřitelný a je možné jej spustit na platformách operačního systému Windows i Linux. ISE WebPACK pracuje s obvody typu FPGA a CPLD, které lze popsat pomocí programovacího jazyka VHDL či programovat v jazyce Verilog. Obvody typu FPGA je možné rozdělit na série podle typu daných obvodů. Vyskytuje se série Spartan a série Virtex. Základní parametry série Virtex jsou uvedeny v tabulce 2. Dále je možné pomocí tohoto vývojového prostředí sestrojit schéma, které lze simulovat a ověřit. Po odsimulování a ověření následuje realizace, která je možná pomocí rozhraní JTAG a tím daný obvod naprogramovat. [7] [8] [10]

Tabulka 2 - Série Virtex

Série	Systém hradel	Počet logických buněk	Zpoždění T_{IOPI} [ns]
Virtex-II	40k - 8 M	576 - 104832	0,88
Virtex-4	100k - 1,6 M	12312-142128	0,82
Virtex-5	1M - 3M	30720 -239616	0,55

Rozhraní JTAG využívá metodu známou jako boundary scan neboli metodu hraničního testu. Jeho výhoda spočívá v odstranění mechanických operací a umožňuje měnit obsah programu, ať již se jedná jenom o aktualizaci či o nový program. To znamená, že jsou programovatelné v daném systému. Toto možné programování v daném systému se označuje jako ISP, které využívá rozhraní JTAG. [2] [10]

5.3. Ukázka programu v jazyce VHDL

Ve vývojovém prostředí je vygenerován automatický text jazyka VHDL, který obsahuje samotný základ nového souboru. Jedná se o knihovny, které jsou vloženy na začátek souboru. Následuje entita, která popisuje rozhraní daného systému. Funkce a chování entity je naprogramováno v architektuře. [1] [10]

Uvedeme si jednoduchý program, který ze vstupních signálů A a B vypočítá jejich logický součin. Výsledek je uložen do signálu C. Je nutné tyto signály definovat jako port entity. Funkce systému AandB je popsána v architektuře.

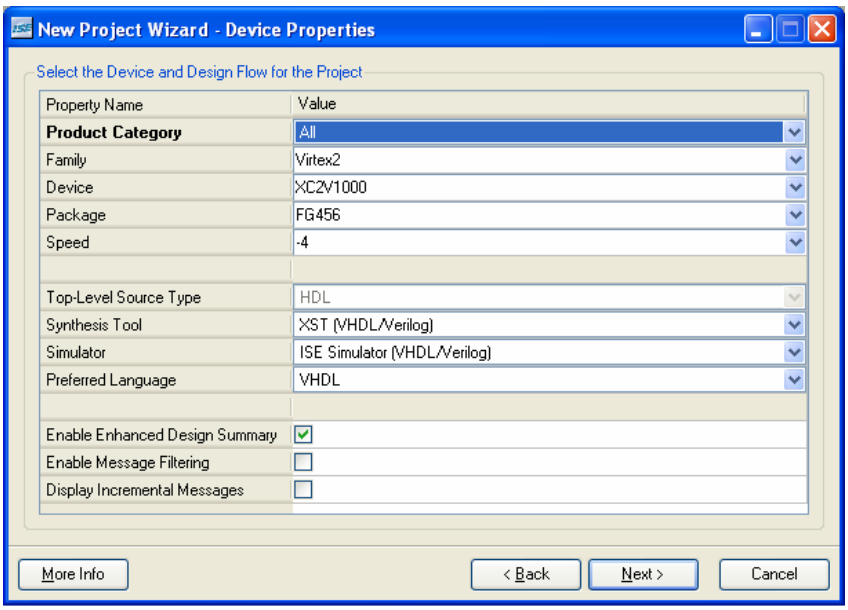
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity AandB is
port(
A : in std_logic;
B : in std_logic;
C : out std_logic
      );
end AandB;

architecture Behavioral of AandB is
begin
      C <= A AND B;

end Behavioral;
```

6. Realizace

Tato kapitola se věnuje realizaci periferií. Jsou zde uvedeny schéma zapojení obou obvodů a popis nastavení vstupních signálů. U komunikace podle standardu UART je uvedeno jakým způsobem jsou nastaveny jednotlivé vstupy a jaká data jsou očekávána na výstupu. U komunikace I2C je realizován obvod master sběrnice I2C, který celou komunikaci řídí. Při zahájení komunikace zasílá podmínku startu. Po odeslání této podmínky následuje adresní rámec, který určí obvod slave, se kterým bude master komunikovat. Po potvrzení rozpoznání adresy obvodu slave master zasílá datové rámce. Periferie byly vytvořeny ve volném vývojovém prostředí ISE WebPACK. Jsou zde uvedeny postupy návrhu v tomto prostředí. Programování probíhalo pomocí programovacího jazyka VHDL. Po samotném naprogramování byla provedena simulace a následně byly programy implementovány do odvodu Xirtex-II XC2V1000 a výsledky byly ověřeny pomocí osciloskopu.

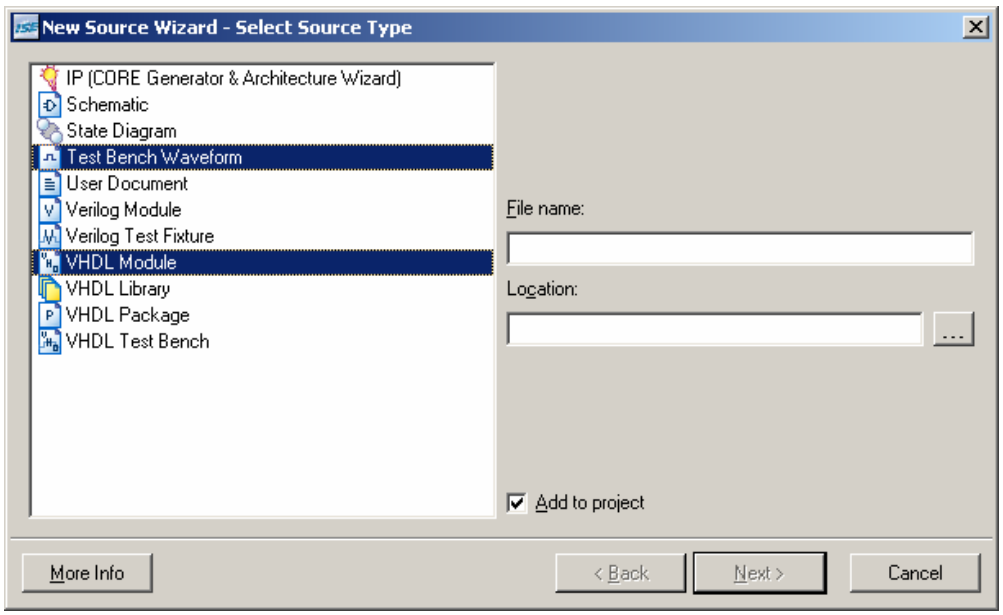
6.1. ISE WebPACK

Programové prostředí ISE WebPACK obsahuje řadu nástrojů a možností. Pro realizaci obvodů nebyly veškeré nástroje využity. Při vytvoření nového projektu je nutné zvolit typ HDL. Dále následuje zadání obvodu, který bude programován. Tento krok je důležitý z hlediska správného vytvoření formátu u programovacího souboru. Jedná se zde o zadání správné série, obvodu FPGA, zapouzdření a rychlosti viz. obrázek 7. Pokud jsou tyto informace chybně nastaveny, je vytvořen programovací soubor pro obvod s jiným uspořádáním architektury a vstupně-výstupních pinů. Proto není možné nahrát soubor vytvořený pro specifický obvod do jiného obvodu.



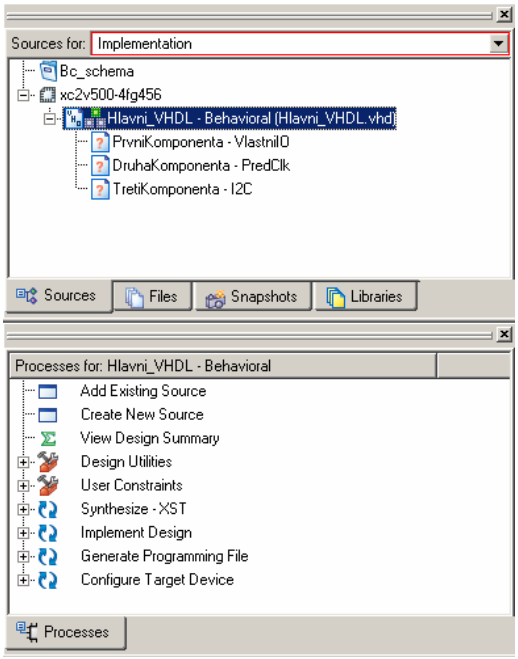
Obrázek 7: Výběr FPGA

Po zadání potřebných informací se nám vytvoří nový projekt. V tomto projektu budeme dále používat pouze dva zdrojové typy viz. obrázek 8. Jedná se o zdroje VHDL Modul a Test Bench Waveform. Je nutné podotknout, že tyto zdroje nelze vytvořit současně. Je tedy nutné vytvoření nejprve prvního zdroje, který bude VHDL Module. Po vytvoření tohoto zdroje si vytvoříme zdroj Test Bench Waveform.



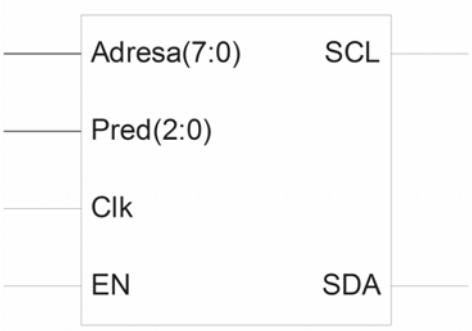
Obrázek 8: Použité zdrojové typy

Jako první byl vytvořen VHDL Module. Tento zdroj se objeví v části implementace viz. obrázek 9.



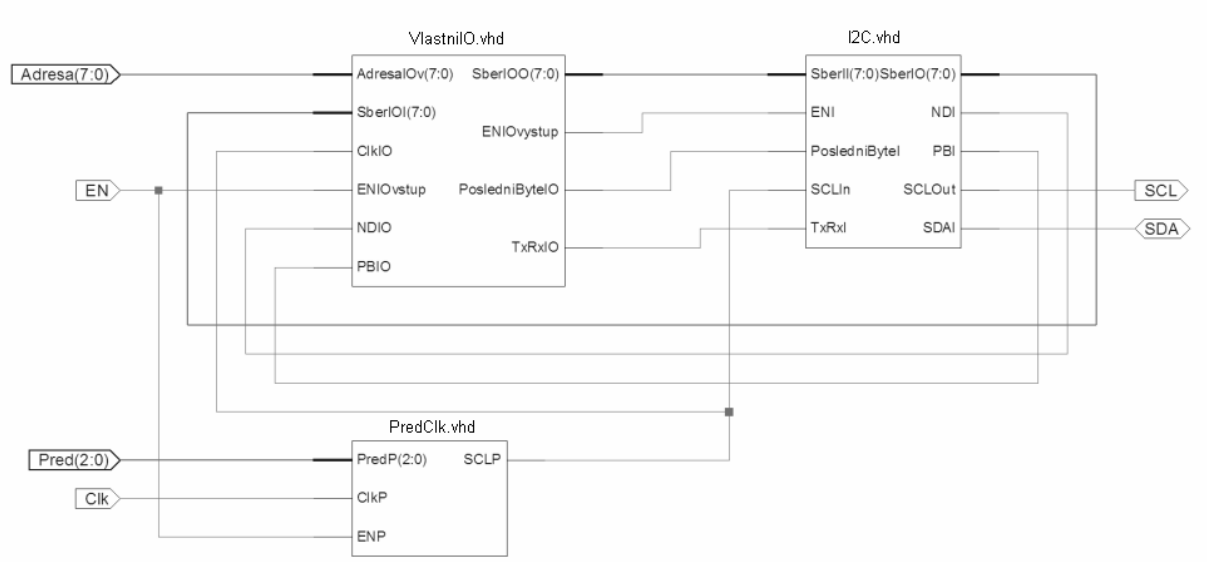
Obrázek 9: Implementace do modulu

Po otevření souboru se zobrazí textový editor. Pro vytvoření obvodu realizující asynchronní komunikaci bude VHDL modul pouze jeden. Obvod master je ale vytvořený ze tří dílčích, samostatně funkčních, bloků. Ve vytvořeném modulu tedy naprogramujeme vstupní a výstupní piny obvodu master a vstupní a výstupní piny daných bloků. Vytvoření vstupních a výstupních pinů se provede pomocí portů. Po vytvoření vstupních a výstupních pinů jsou přidány další moduly, které slouží již pro samotné programování dílčích bloků. Tyto moduly jsou ale bez zdroje, proto je nutné opět vytvořit nový VHDL modul, který bude pojmenován stejně jako dílčí blok. Po vytvoření všech modulů je možné vygenerovat schéma periferie viz. obrázek 10.



Obrázek 10: Schéma periferie realizující I2C

Vstupní signály jsou Adresa(7:0), tato 8 bitová sběrnice slouží pro nastavení sedmibitové adresy. Hodnota Adresa(0) je bit R/W, který určí, zda se budou data z obvodu slave číst nebo do něj budou data zapisována. Hodnota Adresa(7) je nejvýznamnější bit, tento bit je zasílán jako první po zahájení komunikace. Vstupní sběrnice označená Pred(2:0) slouží pro nastavení předděličky hodinového signálu. Jedná se o tříbitovou sběrnici. V závislosti na nastavení bitů je hodinový signál Clk podělen přesně danou hodnotou. Poslední vstupní signál je EN, který aktivuje periferii přivedením log. 0. Schéma periferie neukazuje podrobné zapojení dílčích bloků, které je zobrazeno viz. obrázek 11.

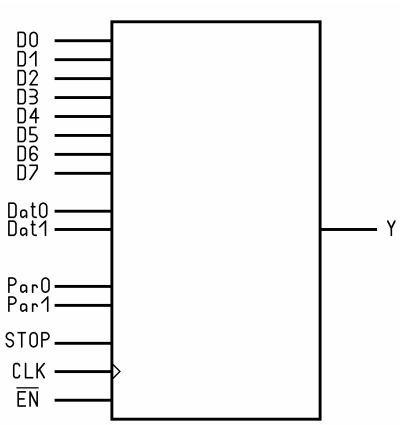


Obrázek 11: Detailní schéma periferie

Samotná periferie je složena ze tří dílčích obvodů nazvaných VlastniIO, I2C a PredClk. Tyto obvody jsou naprogramovány pomocí programovacího jazyka VHDL.

Obvod VlastniIO slouží pro řízení komunikace. Zde se nastavuje kolik datových bytů je posláno za adresou obvodu slave. Naprogramovaný počet je až 5 datových bytů a tento počet se může jednoduchým zásahem přeprogramovat. Je možné datové byty vysílat a to pomocí sběrnice SberIOO(7:0), nebo data přijímat pomocí sběrnice SberIOI(7:0). Dále jsou zde signály pro komunikaci s obvodem I2C, které doplňují informace o přenosu. Výstupní signál TxRxIO určuje, zda budou data vysílána či přijímána, dále PosledníByteIO určí, zda je odesílán již poslední datový byte. Pokud ano, samotný obvod I2C zašle podmínku pro ukončení komunikace a nastaví linky SDA a SCL do hodnoty log. 1. Poslední obvod je PredClk, který podle definovaných stavů dělí kmitočet hodinového signálu Clk.

Obvod realizující komunikaci podle standardu UART již tak složitý nebude. Jeho schéma zapojení je zobrazeno viz. obrázek 12.



Obrázek 12: Schéma periferie realizující UART

Samotná periferie je navrhnuta podle standardu UART, tedy datový rámec je tvořen jedním start bitem, který je realizován logickou 0. Po tomto bitu následují datové bity v pořadí od LSB k MSB. Je možnost si zvolit počet datových bitů na výstupu. Minimální počet těchto datových bitů je 5, což znamená, že výstupní datový rámec je tvořen pouze datovými bity D0-D4. Logické úrovně na D5-D7 v tomto případě nejsou přenášeny. Maximální počet přenášených dat je 8, tedy vstupy D0-D7. Určením počtu výstupních dat je dáno pomocí nastavení řídících vstupů, které jsou označeny Dat0 a Dat1. Podrobnější popis jak přesně nastavit počet datových typů v periferii je určen tabulkou 3.

Tabulka 3 – Hodnoty Dat0 a Dat1

Dat0	Dat1	Výstupní data
0	0	D0-D4 (5 bitů)
0	1	D0-D5 (6 bitů)
1	0	D0-D6 (7 bitů)
1	1	D0-D7 (8 bitů)

Po datových bitech je zvolena parita. Jedná se o redundantní informaci, proto je možné si zvolit jaký typ parity použijeme a zda vůbec kontrolu použijeme. Paritu je opět možné nastavit pomocí vstupních pinů označené jako Par0 a Par1. Je nutné použít 2 bity, protože nastavení parity je možné na sudou paritu, lichou paritu a také je zde možnost zvolit datový přenos bez parity. Nastavení těchto bitů je znázorněné v tabulce 4.

Tabulka 4 – Hodnoty Par0 a Par1 pro zvolení parity

Par 0	Par 1	Výsledná parita
0	0	bez parity
0	1	sudá parita
1	0	lichá parita
1	1	bez parity

Poslední bity v datovém rámci jsou stop bity. Jedná se o bity, které mají logickou úroveň 1 a jejich nastavení je možné určit pomocí pinu, který je pojmenován Stop. Je-li tento bit v logické úrovni 0, pak je v datovém rámci pouze jeden stop bit. Je-li na pin Stop přivedena logická úroveň 1, pak jsou v datovém rámci stop bity dva.

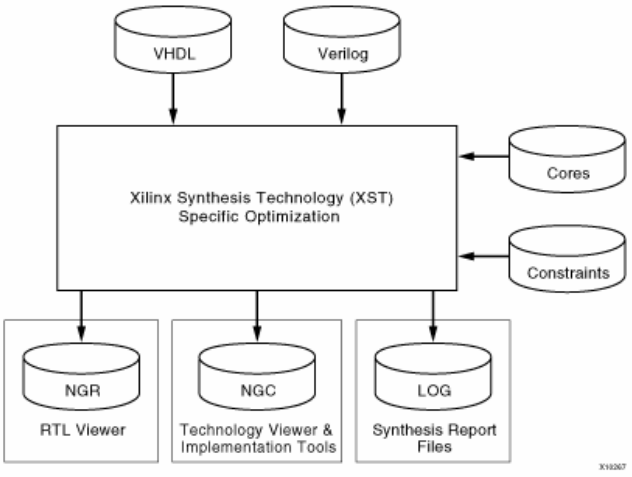
Dalším vstupním pinem je pin označen jako EN (Enable), který je aktivní v log. 0. Pokud tedy na tento vstup dáme tuto úroveň, pak se nám v tomto okamžiku nasunou datové bity do paměti. Jakmile je ale EN spouštěn periodicky ve velmi krátkých časových intervalech, tak by se data v paměti přepisovala a tím by se porušil datový rámec. Přijímač takového sériového signálu by

nebyl schopen signál přijmout. Proto periferie nasouvá nová data až po odeslání kompletního datového rámce.

Hodinovým signálem jsou data na výstupu synchronizována. To znamená, že data v datovém rámci jsou vysílána pomocí tohoto signálu a tím je zaručena stejná perioda výstupního signálu. Tento hodinový signál je aktivní na náběžnou hranu, takže s každou náběžnou hranou je změna výstupního signálu v závislosti na obsahu datového rámce. Perioda hodinového signálu nám také určuje rychlost sériové komunikace.

6.2. Implementace

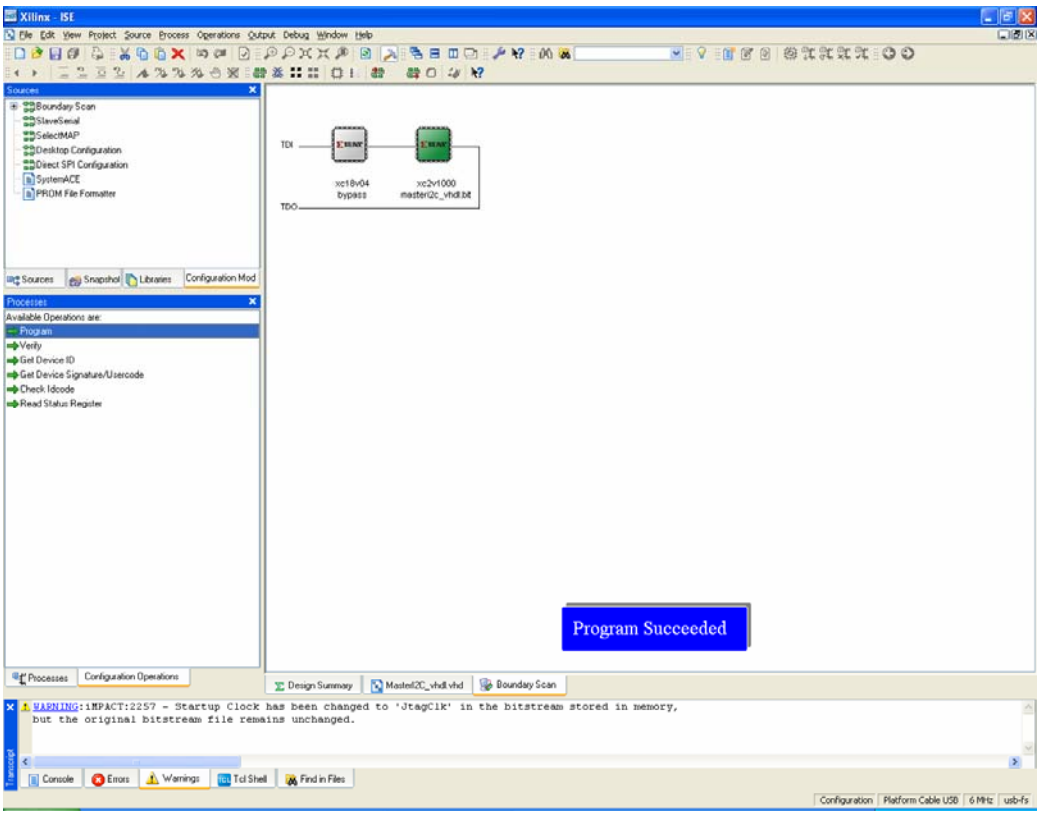
Po naprogramování je možné provést implementaci. V prvním kroku implementace se spustí syntéza, při které jsou vytvořeny soubory netlistů. Tyto soubory jsou uloženy jako soubory s příponou *.ngc viz. obrázek 13. Protože bylo možné psát program i v programovacím jazyce Verilog, či vytvoření programu pomocí schématu, došlo tímto krokem ke sjednocení rozdílů. [12]



Obrázek 13: Syntéza (Obrázek převzat z [12])

Po syntéze se vygenerují soubory s příponou *.ngc, které obsahují návrh pro uspořádání logiky v FPGA, ale i údaje o omezení použití. Tyto soubory byly dále použity jako vstupní data pro druhý krok implementace, což je návrh. Syntéze je označována jako Synthesis – XST. Toto označení je uvedeno proto, že je využito optimalizací, které jsou provedeny algoritmem Xilinx Synthesis Technology (XST). Po dokončení tohoto procesu se dále spustí překlad, ve kterém dochází ke sloučení všech souborů vytvořených při syntéze do jednoho souboru s příponou *.ngd. Tento soubor se pak může po malé úpravě nahrát do obvodu FPGA. Je možné se podívat také na zprávu, která je vytvořená při tomto procesu. Tato zpráva obsahuje upozornění a chyby, které jsou objeveny. Dále se v návrhu kontroluje celkové navržení a uspořádání logiky. Tento krok musí být proveden až po překladu, protože tato kontrola vyžaduje soubory s příponou *.ngd, které obsahují celkové navržení logiky. Výsledky z této kontroly se uloží do souboru s příponou *.ncd. Je možné opět vygenerovat zprávu z tohoto procesu, kde je možné zjistit počet chyb a varování, které jsou zde vypsány. Jako poslední krok při návrhu je umístění a směrování. Zde je nalezen nejlepší výsledek pro umístění navržené logiky tak, aby bylo dané FPGA nejlépe využito. Tento krok je mnohokrát opakován, než se najde nejlepší umístění logiky pro dané FPGA. [12]

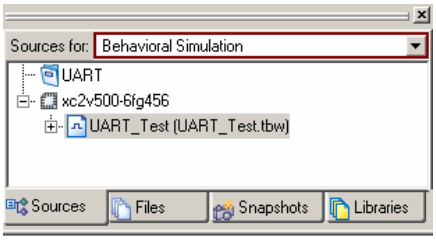
Programování FPGA probíhá pomocí souboru, který je vytvořen v procesu pro generování programovacího souboru. Přípona pro obvody FPGA je většinou *.bit nebo *.isc. Jakmile je tento soubor vytvořen, obsahuje veškeré informace, které definují uspořádání vnitřní logiky a vzájemná propojení. Samotné naprogramování je potom tvořeno pomocí software iMPACT. Před samotným naprogramováním je potřeba tento software nastavit a to především soubory pro samotný zápis. V našem případě je realizace na FPGA série Vitrex-II, označení obvodu je XC2V1000. Tento obvod je naprogramován pomocí vývojového nástroje V2MB1000. Prostředí iMPACT pro programování FPGA je zobrazeno viz. obrázek 14.



Obrázek 14: Programování FPGA

6.3. Simulace UART

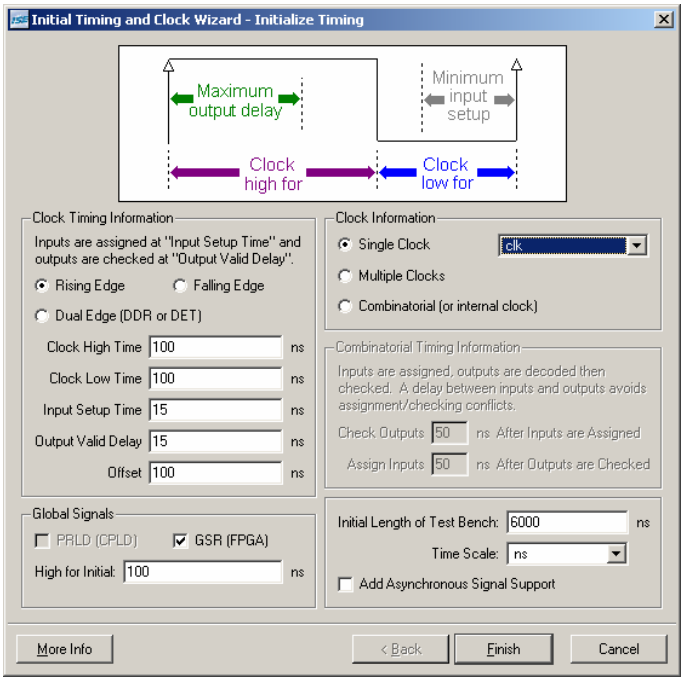
Simulace byla prováděna v druhém zdrojovém typu, který byl vytvořen. Jedná se o Test Bench Waveform viz. obrázek 8. Je tedy nutné změnit zdroj na režim simulace viz. obrázek 15.



Obrázek 15: Režim simulace

6.3.1. Nastavení simulace

Při otevření souboru je nutné počáteční nastavení simulace viz. obrázek 16, kde je možné nastavení náběžné a sestupné hrany. Nastavením periody hodinového signálu určujeme frekvenci výstupního signálu. Ponecháme nastavení náběžné hrany (Rising Edge) a doby 100 ns pro délku trvání logické úrovně 0 a také 100 ns pro délku trvání logické úrovně 1. Hodinový signál je Clk, což je vybráno pomocí comboboxu. Důležité pro nás je nastavení pro délku simulace, která je přednastavena na 1000 ns. Tato doba je ale krátká, a proto je nutné nastavení delšího úseku. Volíme 6000 ns. Lze volit pouze vstupní signály. Výstup Y bude určen pomocí simulace, proto jej není možné nastavit. Při pokusu nastavit výstupní signál program zahlásí chybu.

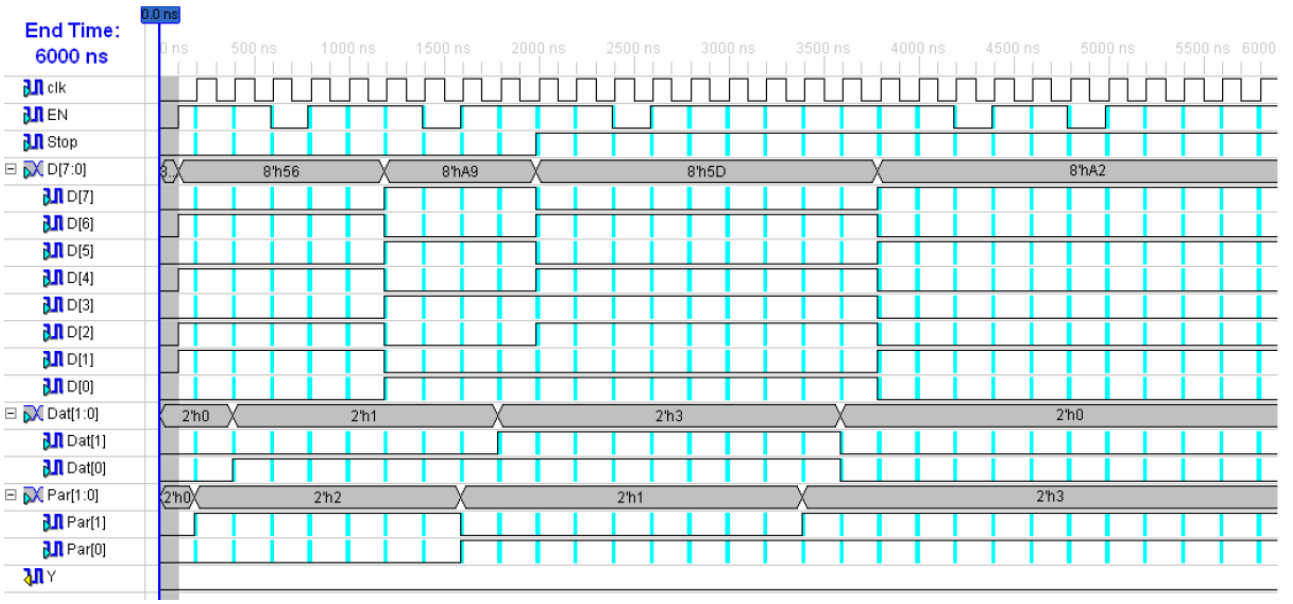


Obrázek 16: Nastavení simulace

Následně již můžeme navolit libovolné průběhy signálů, přičemž musíme mít na paměti, že nejmenší počet bitů na výstupu je 7. Tedy Start bit, dále 5 datových (D0-D4), paritu nevolíme, tedy vstupy Par0 a Par1 jsou v úrovních 0 či 1 viz. tabulka 4. Volíme pouze jeden Stop bit. Perioda jednoho hodinového signálu je 200 ns, tedy přenos tohoto nejmenšího počtu výstupních dat se bude přenášet 1400 ns. Proto je důležité počáteční nastavení simulace dát alespoň 2000 ns pro zobrazení jednoho celého výstupního datového rámce o celkové hodnotě 7 bitů.

Kdybychom chtěli mít datový rámec o maximální délce 12 bitů, pak volíme všechny datové vstupy viz. tabulka 3. Paritu a dva Stop bity. Výstup takového datového rámce by potom trval při jedné periodě hodinového signálu 200 ns celkem 2400 ns.

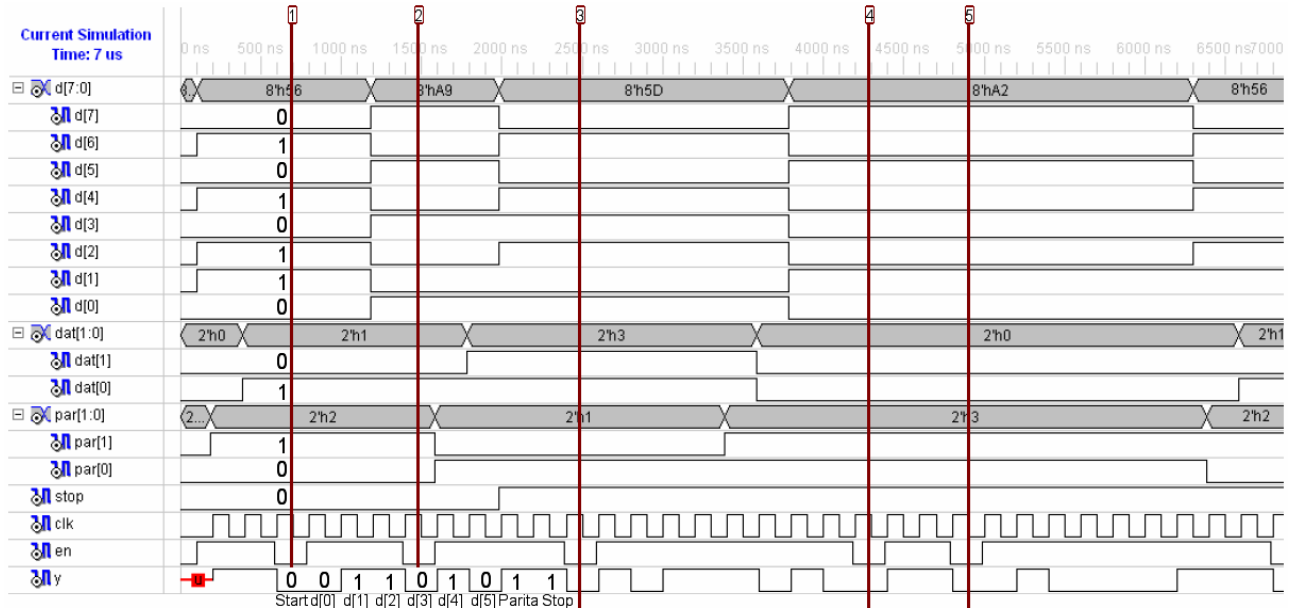
Pro volby jednotlivých průběhů v simulaci dále musíme zohlednit vstup EN, který je aktivní na log. 0. To znamená, je-li vstup EN aktivní, pak v tomto okamžiku jsou všechny vstupní signály zachyceny do paměti v závislosti na nastavení jejich logických úrovní. To ale neplatí v případě, že vstup EN je aktivní v době, kdy nebyl přesunut celý datový rámec na výstup Y z předchozího aktivního vstupu EN. Pokud tedy nastane tento případ, pak tento opakující se aktivní EN nemá vliv na výstupní datový rámec a je zde zcela nadbytečný. Pokud by však nebyl tento případ ošetřen, tak by se nám přepsaly data uložená v paměti a tím i data v datovém rámci připravené pro výstup. Tímto krokem by byl vyslaný datový rámec složen z několika různých vstupních dat a tím by nebyl shodný se standardem UART. Nastavení jednotlivých průběhů je zvoleno viz. obrázek 17.



Obrázek 17: Nastavení průběhů

6.3.2. Výsledky simulace

Po nastavení požadovaných průběhů je možné spustit simulaci. Po dokončení tohoto kroku se automaticky zobrazí výsledky simulace viz. obrázek 18. V prvním okamžiku není možné nastavit úroveň log. 1 na výstup, tedy kdy nedochází k datovému přenosu na výstupu. Do tohoto stavu se dostaneme až s první náběžnou hranou hodinového signálu. Proto je na počátku úroveň nedefinována, tedy hodnota U.



Obrázek 18: Výsledky simulace

Kurzor č. 1 nám určuje první spuštění. V tomto okamžiku se tedy data na vstupech nahrají do paměti a zpracují se. Datové vstupy jsou tedy v posloupnosti LSB-MSB následující 01101010, ale počet zobrazených dat na výstup Y je určen pomocí vstupů dat1 a dat0, které mají hodnotu dat1=0 a dat0 = 1. Tedy výstupní datový rámec obsahuje pouze 6 datových bitů a to d[0] až d[5]. Tedy předpokládaný výstup je Y = 0011010. Následuje ověření přítomnosti paritního bitu. Vstup par1=1 a par0 = 0, tedy je nastavena sudá parita. Celkový počet logických úrovní 1 v datovém rámci jsou 3. Tedy paritní bit při nastavení sudé parity musí být v logické úrovni 1. Jako poslední následují stop bity. Vstup Stop je nastavený na log. 0, tedy je zde pouze jeden stop bit. Celkový výstupní rámec bude obsahovat posloupnost bitů následující 001101011. Po odeslání tohoto

datového rámce je možné odeslat další. V průběhu odesílání tohoto prvního datového rámce není možné přijmout jiná data do paměti, proto opakovaný EN v tomto případě není aktivní. Tento stav nám zobrazuje kurzor č.2 viz. obrázek 18. Kurzor č.3 nám ale opět zajistí nakopírování dat do paměti, protože v tomto kroku již byl datový rámec kompletně odvysílán. Tedy se provede stejný postup jako při první detekci aktivní periferie, a to takový, že se opět data ze vstupních signálů přesunou do paměti. Datové vstupy jsou ale nastaveny na jiné hodnoty, a to na 10111010. V tomto případě bude výstupní datový rámec tvořeny všemi datovými bity, protože vstupy Dat0 a Dat1 jsou oba v úrovni log. 1. Paritní bit bude v tomto případě určen pro lichou paritu, protože vstupy par0 a par1 jsou nastavené na hodnoty 1 pro par0 a 0 pro par1. Celkový počet log. 1 v datovém rámci je 5, tedy paritní bit musí být nastaven v tomto případě na hodnotu logická 0. Výsledný výstupní datový rámec je tedy tvořen bity 010111010011. Stop bity budou v tomto případě 2, protože vstup Stop je nastaven na log. 1. Je zde opět opakované spuštění periferie, které na výstupní rámec nemá vliv. Toto spuštění je znázorněno kurzorem č. 4 viz. obrázek 18.

Poslední je zobrazený kurzor 5. V tomto případě bude výstupní rámec složen pouze z 5 datových bitů, což odpovídá bitům 01000. Vstupní dat jsou tedy nastaveny na hodnotu 0. Paritní bit v tomto případě není v datovém rámci, protože hodnoty Par jsou nastaveny na 1 viz. tabulka 4. Vstup Stop je nastavený na log. 1, tedy výstupní datový rámec obsahuje 2 stop bity. Očekávaný výstup Y je tedy 00100011. Po odeslání těchto bitů je opět periferie připravena na spuštění. Což se ale neděje hned v následujícím hodinovém cyklu, ale o jeden cyklus dále. Zde je ověřeno, že nejsou vysílána žádná data na výstupu a periferie tedy nepřenáší žádná data až do doby, než je opět aktivní EN. Je zde nutné opomenout, že na počátku datového rámce je vždy start bit, který odpovídá log. úrovni 0. Tento start bit je na výstupu již při aktivaci periferie, která vysílá data vždy s náběžnou hranou hodinového signálu Clk.

6.4. Simulace I2C

Nastavení simulace probíhalo stejným způsobem jako u periferie realizující komunikaci podle standardu UART.

Z hlediska uživatele je důležité nastavení vstupních signálů. Obvod master je tvořen vstupními signály Adresa(7)-Adresa(1), kde Adresa(7) je nejvýznamnější bit a Adresa(1) je nejméně významný bit zasílané adresy. Posloupnost bitů přivedená na tyto vstupy tvoří odesílanou adresu obvodu slave. Součástí této adresy je také bit R/W, který je daný hodnotou Adresa(0). Pokud je tento bit log. 1, pak obvod master bude data číst z obvodu slave. Pokud bude tento bit roven log. 0, pak se budou data zapisovat.

Periferie je aktivní v případě, že na řídícím vstupu EN je hodnota log. 0. Tato hodnota zde musí být po celou dobu komunikace. Pokud je během komunikace tento vstup neaktivní, je komunikace ukončena tak, že v okamžiku příchodu log. 1 se linky SDA a SCL nastaví do log. 1.

Počet odesílaných datových bytů je nastaven v programu, stejně jako posloupnosti jednotlivých bitů v daných datových bytech. Tyto posloupnosti jsou tvořeny pomocí konstant a jsou umístěny v architektuře dílčího obvodu VlastniIO. Program umožňuje odeslat až 5 datových bytů do obvodu slave. Je zde také možnost data přijmout z tohoto obvodu a následně dále zpracovat.

Periferie obsahuje dva výstupní signály SDA a SCL, kde signál SDA je realizován jako vstupně-výstupní, to znamená, že data může vysílat ale po zadání požadavku obvodu slave i přijímat. Signál SCL je výstupní signál. Pokud je periferie aktivní, je tento výstup určen vstupním hodinovým signálem, který je upraven pomocí předděličky. Pokud je periferie neaktivní, je tento signál v hodnotě log. 1.

6.4.1. Předdělička Clk

Součástí obvodu master je i předdělička hodinového signálu Clk, která dle dané rovnice upravuje výstupní hodinový signál SCL. Předdělička hodinového signálu je řízena vstupními datovými signály Pred2:0. Jedná se o 3-bitové číslo, je tedy možné volit až 8 různých hodnot

výstupního hodinového signálu v závislosti na vstupním hodinovém signálu viz. tabulka 5. Hodnoty f_{SCL} byly vypočteny podle rovnice 1 a zároveň ověřeny měřením. Zdrojem hodinového signálu byl krystal o frekvenci 24 MHz a 100 MHz.

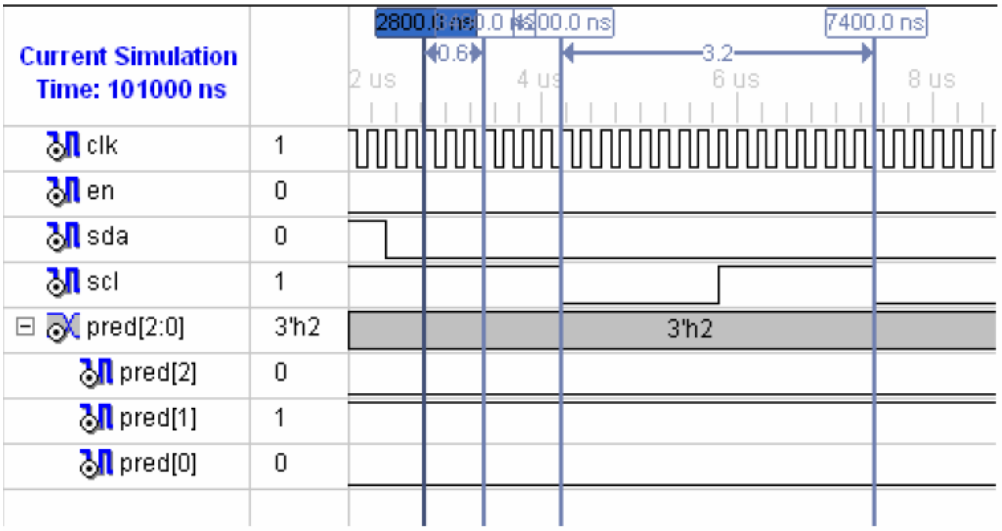
Tabulka 5 - Hodnoty předděličky

Pred[2]	Pred[1]	Pred[0]	n	f_{SCL}(f_{Clk} = 24 MHz)	f_{SCL} (f_{Clk} = 100 MHz)
0	0	0	1	12 MHz	50 MHz
0	0	1	4	3 MHz	12,5 MHz
0	1	0	8	1500 kHz	6,25 kHz
0	1	1	32	375 kHz	1562,5 kHz
1	0	0	64	187,5 kHz	781250 Hz
1	0	1	256	46,875 kHz	195312,5 Hz
1	1	0	1024	11718,75 Hz	48828,13 Hz
1	1	1	2048	5859,375 Hz	24414,06 Hz

Výpočet pro hodinový signál SCL je tedy dán rovnicí

$$f_{SCL} = \frac{f_{CLK}}{2 \cdot n}, \qquad (1)$$

kde f_{Clk} je frekvence hodinového signálu Clk a n je hodnota předděličky podle nastavení signálu Pred(2:0). Pro simulaci bylo nastaveno Pred(2:0) na hodnotu 010. Hodnota n tedy bude rovna 8. Perioda hodinového signálu nastavená pro simulaci odpovídá hodnotě 200 ns, frekvence hodinového signálu je tedy rovna 5 MHz. Výsledná frekvence výstupního signálu z předděličky je rovna 312,5 kHz, což odpovídá periodě 3,2 µs viz. obrázek 19.



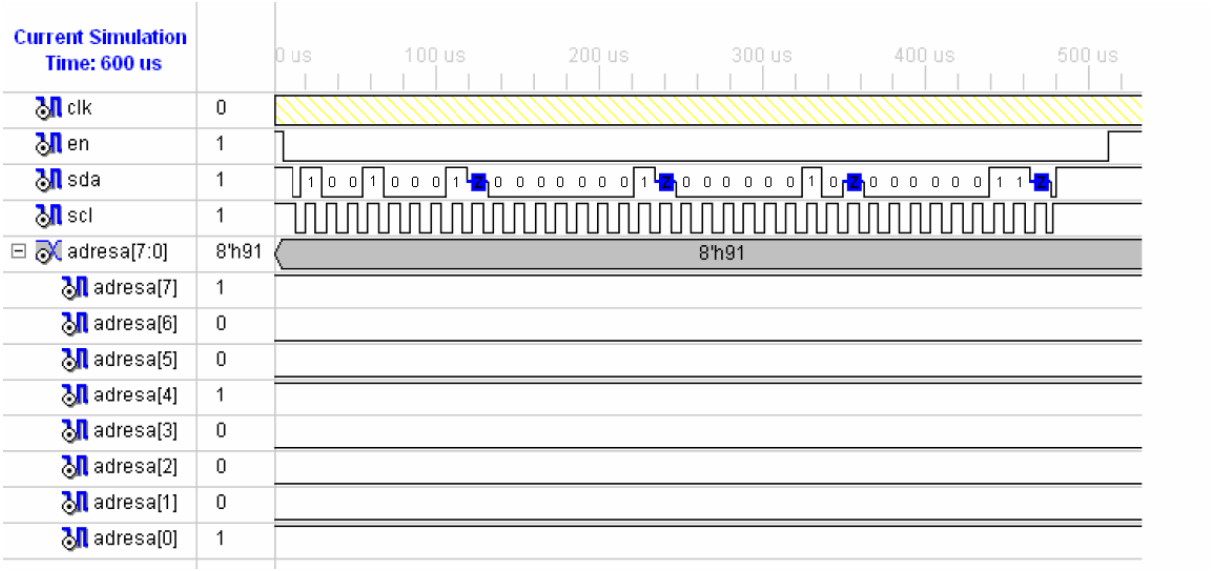
Obrázek 19: Simulace předděličky

Samotná simulace dopadla podle očekávání, proto byla provedena i realizace. V tomto případě byl využit 24 MHz krystal jako zdroj pro hodinový signál Clk. Signál Pred(2:0) byl nastaven na stejnou hodnotu jako při simulaci, tedy na hodnotu 010. Hodnota n se tedy nezměnila a je rovna 8. Vypočítaná hodnota podle rovnice(1) výstupního hodinového signálu je 1500 kHz. Skutečná hodnota výstupního hodinového signálu byla ověřena pomocí osciloskopu a byla rovna 1499,962 kHz. Vzniklá nepřesnost je dána použitím reálných součástek, a především krystalem, kde jeho frekvence je v určité toleranci. Nepřesnost vypočítané hodnoty a změřené hodnoty je 38 Hz.

6.4.2. Výsledky simulace

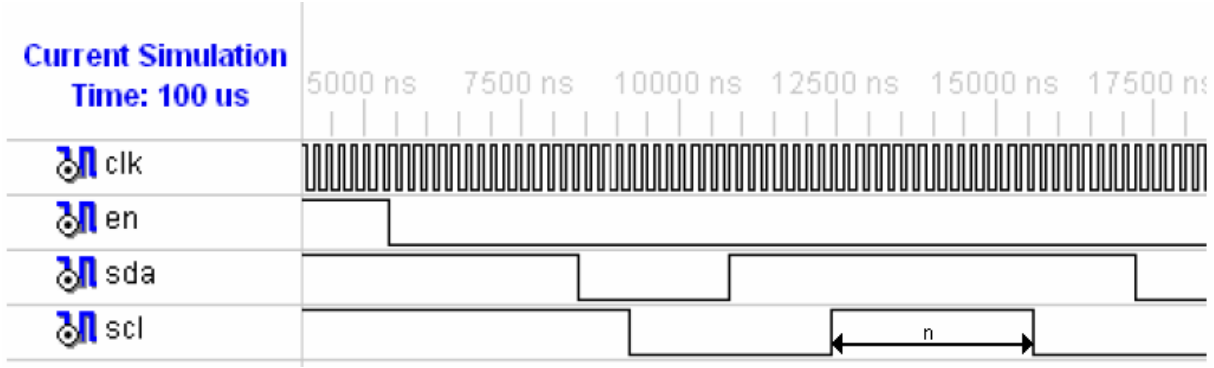
Po navolení požadovaných průběhů bylo možné spustit simulaci, ve které byla zvolena adresa 0b1001000 a R/W byl zvolen na hodnotu log. 1. Celý adresní rámec je tedy 0b10010001.

Následují datové rámce o hodnotě 0b00000001, 0b00000010 a 0b00000011. Tyto hodnoty jsou nastavovány v programu. Po odeslání je komunikace ukončena podmínkou stop. Po odeslání každého bytu slave zasílá potvrzení o příchodu dat a tím i správného zpracování hodnotou log. 0. V tomto případě je po dobu jedné periody master odpojen a na výstupu je stav vysoké impedance označen Z. Tím je zaručeno, že obvod slave může potvrdit příchod dat. Výsledky simulace jsou zobrazeny viz. obrázek 20.



Obrázek 20: Výsledky simulace komunikace I2C

Při příchodu log. 0 na řídící signál EN se před vlastním zahájením komunikace nastaví řídící signály. Adresní bity se přesunou ze vstupní sběrnice obvodu VlastniIO na výstupní sběrnici. Tento krok je proto, aby se předešlo časovým problémům, které by mohli nastat při zahájení komunikace se současným nasunutím adresy a nastavením řídících signálů. Po nastavení všech řídících signálů je z obvodu VlastniIO spuštěn i obvod I2C. Doba nastavení zabere právě hodnotu n, která je určena signály Pred(2:0). S příchodem n+1 náběžných hran hodinového signálu Clk je zaslána podmínka startu. Linka SDA jde do log. 0, zatímco linka SCL je v hodnotě log. 1 a to po dobu jedné čtvrtiny hodnoty n. Po této době se změní stav i na této lince a tím je vygenerována podmínka startu. První bit je nasunut v polovině n a trvá po dobu 2 n, kde dojde k nasunutí nové hodnoty viz. obrázek 21.



Obrázek 21: Simulace podmínky start

6.5. Realizace pomocí vývojové desky

K realizaci byla využita vývojová deska V2MB1000. Napájení této desky je zajištěno pomocí externího 5 V zdroje. Toto napětí se dále upraví podle potřeby a to na 1,5 V, 2,5V a nebo 3,3 V pomocí stabilizátorů. Při realizaci byla použita 3,3 V logika, tedy hodnota log. 1 má právě tuto hodnotu napětí a hodnota log. 0 odpovídá napětí 0 V. Dále tato deska obsahuje dva krystaly o hodnotách 24 MHz a 100 MHz. Při realizaci byl využit 24 MHz krystal jako zdroj hodinového signálu Clk. Na této desce byly dále využity DIP přepínače, které pro sériovou asynchronní

komunikaci tvořili datové bity D0-D7. U komunikace I2C sloužili pro nastavení adresy obvodu slave a bitu R/W. Dále bylo využito tlačítko pro řídící signál EN. Funkce tohoto vstupu je u obou periferií totožná. Obvod FPGA byl programován pomocí rozhranní JTAG. [11]

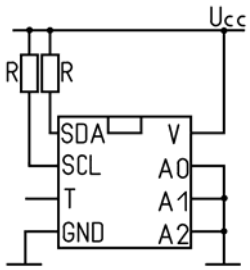
6.5.1. Realizace UART

Frekvence oscilací hodinového signálu je 24 MHz, tedy jedna perioda hodinového signálu bude mít hodnotu okolo 41,6 ns. Za tuto dobu se tedy přenese jeden bit z datového rámce na výstup. Pokud budeme přenášet maximální délku datového rámce, a to 12 bitů, pak tento datový rámec bude přenesen za dobu okolo 500 ns. Pokud máme nastavenou časovou základnu osciloskopu na 100 ns/div, pak tato hodnota odpovídá 5 dílkům viz. obrázek 22. Na tomto oscilogramu můžeme vidět průběh, který zobrazuje všechny datové bity. Jako první bit je zobrazený Start bit, následují datové bity, které jsou nastaveny na hodnoty 10110010 v pořadí LSB - MSB. Další bit nám zobrazuje paritní bit. V tomto případě byla nastavena sudá parita. Počet log. úrovní 1 v datovém rámci je sudý, tedy výsledný paritní bit bude mít hodnotu log. 0. A jsou nastaveny dva stop bity. Výsledný datový rámec je tedy 010110010011 a jsou přenesena data 0b01001101.

Obrázek 22: Asynchronní komunikace

6.5.2. Realizace I2C

Komunikace obvodu FPGA byla testována pomocí teplotního čidla DS1631, které bylo propojeno s obvodem FPGA pomocí nepájivého pole. Schéma teplotního čidla zapojeného na nepájivém poli je zobrazeno viz. obrázek 23. Jsou zde použity odpory o hodnotě 4k7 které plní funkci pull-up rezistorů. Vodiče SDA a SCL jsou propojeny s deskou V2MB1000 a zároveň jsou zde připojeny sondy osciloskopu, aby bylo možné sledovat průběh komunikace. [13]



Obrázek 23: Zapojení teplotního čidla DS1631

Napájecí napětí teplotního čidla je v rozmezí od 2,7 V do 5,5 V. Nastavené napájecí napětí bylo 4,0 V z důvodu částečného přizpůsobení použité logiky u obvodu FPGA. Hodnota log. 1 je podle výrobce daná hodnotou 0,9 napájecího napětí. To odpovídá hodnotě 3,6 V. Hodnota log. 0 je daná desetinou napájecího napětí, tedy 0,4 V. Frekvence hodinového signálu SCL je pro správnou funkci do 400 kHz. Pomocí vstupních pinů A0, A1 a A2 je možné částečně volit adresu teplotního čidla, která je složena z pevné části a volitelné. Pevná část má vždy hodnotu 1001 a za ní následují bity, které jsou dané vstupními piny A0, A1 a A2. Podle zapojení z obrázku 22 bude výsledná adresa obvodu slave 0b1001000. Poslední bit na pozici nejméně významného bitu tvoří bit R/W. Tato hodnota je volitelná a záleží, zda budeme data zapisovat nebo číst. [13]

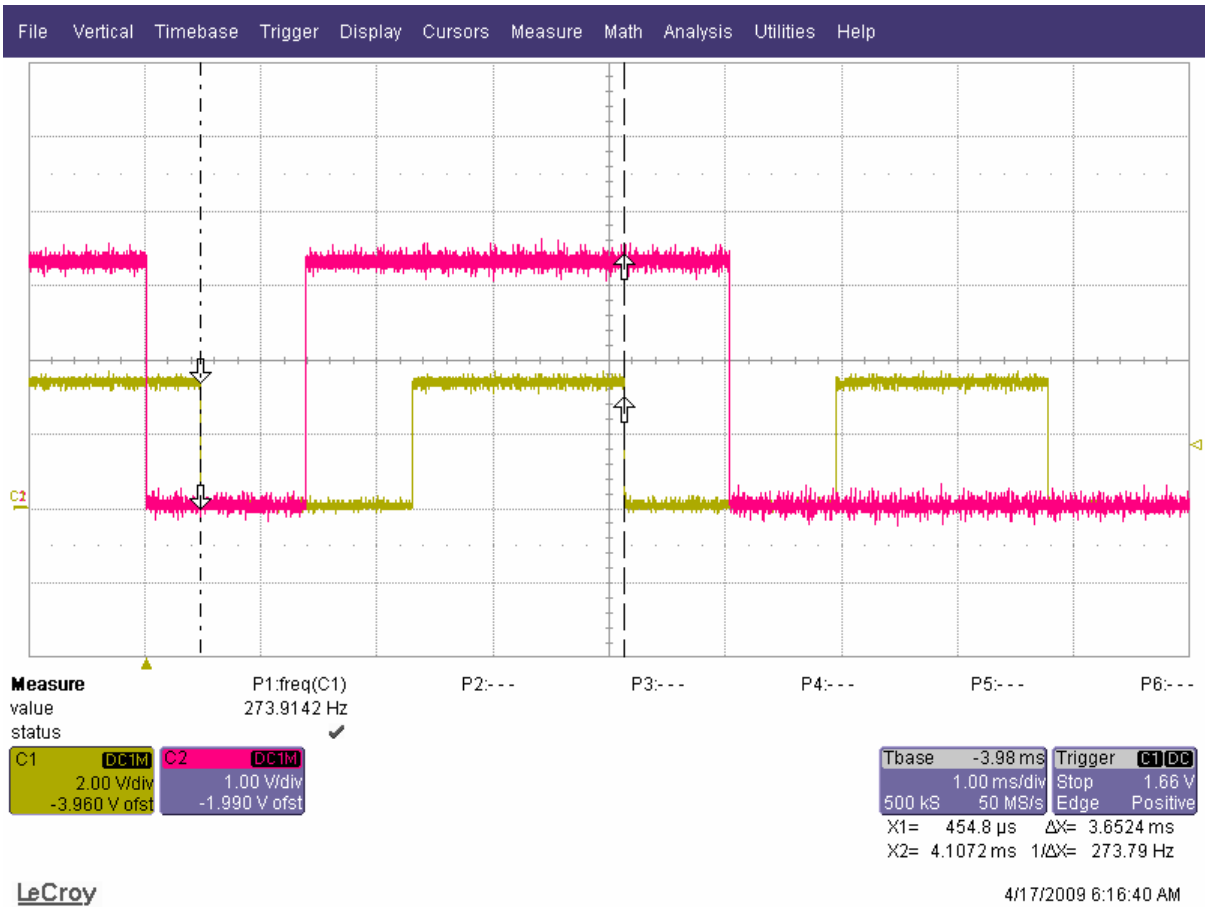
Pomocí DIP přepínačů, které jsou součástí vývojové desky, byla zvolena adresa obvodu slave 0b10010001 a v programu bylo nastaveno, že za adresou budou odeslány tři datové byty o hodnotách stejných, jako byly zvoleny v simulaci. Tedy první datový byte má hodnotu 0b00000001, druhý datový byte je roven 0b00000010 a bity v posledním datovém bytu jsou 0b00000011. Po odeslání této kombinace se ukončí komunikace zasláním podmínky stop. Výsledný oscilogram je zobrazen viz. obrázek 24.

Obrázek 24: Výsledek komunikace I2C

Zachycené výsledky pomocí osciloskopu jsou stejné jako v simulaci. Komunikace začíná podmínkou startu. Po jejím odvysílání následuje adresa obvodu slave. Jakmile je celá adresa odvysílána, je linka SDA daná do stavu vysoké impedance, proto je po odvysílání bitu R/W hodnota okolo 2,5 V do doby, kdy slave potvrdí adresu bitem ACK, tedy linka přejde do stavu log. 0. Pokud by neexistoval obvod s vysílanou adresou, byla by tato hodnota daná do hodnoty log. 1 pomocí připojených pull-up rezistorů. Po potvrzení adresy se následně zasílají určené datové byty. Tyto byty byly zvoleny zcela náhodně, proto nejsou potvrzené obvodem slave. Frekvence hodinového signálu SCL je rovna 313,0474 Hz. Tato frekvence je ale určena z celého oscilogramu, kde před zahájením komunikace a po ukončení komunikace je rovna log. 1. Proto je zobrazeno varování ve formě vykřičníku u této hodnoty. Oscilogram také dokazuje použití dvou různých napěťových

logik. Pokud vysílá data obvod master, je použitá logika 3,3 V, tedy log. 1 má právě tuto velikost napětí, zatímco při vysílání obvodu slave je tato hodnota 3,6 V. Z oscilogramu je patrné, že při potvrzování datových bytů je úroveň napětí viditelně vyšší právě z tohoto důvodu. Při zasílání hodnot log. 0 obvodem master je úroveň napětí 0 V, zatímco při potvrzení adresy obvodem slave tato hodnota není nulová, ale dosahuje hodnoty 0,6 V. Tato nepřesnost je také výsledkem použití dvou napěťových logik, kdy u obvodu slave je hodnota log. 0 daná desetinou vstupního napájecího napětí teplotního čidla.

Podmínka start je realizována po zahájení komunikace. Linka SDA přejde do hodnoty log. 0, zatímco linka SCL je v hodnotě log. 1. Po době osminy periody SCL také přejde do hodnoty log. 0 a tím je dokončena podmínka startu. První bit je nasunut ve čtvrtině periody signálu SCL a trvá jednu periodu. Frekvence hodinového signálu při komunikaci je rovna 273,9142 Hz. Simulace podmínky start a její realizace jsou naprosto shodné viz. obrázek 25.



Obrázek 25: Detail podmínky start

Podmínka stop je vytvořena stejným způsobem jako podmínka start. Linka SCL přejde z hodnoty log. 0 do hodnoty log. 1, zatímco linka SDA je v hodnotě log. 0, po osmině periody signálu SCL také přejde do hodnoty log. 1 a tím je ukončena komunikace s obvodem slave viz. obrázek 26.

Obrázek 26: Detail podmínky stop

7. Závěr

Ve volném vývojovém prostředí ISE WebPACK byly navrženy periférie realizující sériovou asynchronní komunikaci podle standardu UART a obvod master komunikace I2C. Periferie realizující komunikaci I2C je vytvořena jako číslicový systém, který obsahuje tři funkční obvody. Jedná se o řídící obvod, ve kterém určujeme počet odeslaných datových bytů. Druhý funkční obvod realizuje samotnou komunikaci a poslední funkční obvod je předdělička hodinového signálu, která je nezávislá na předchozích dvou obvodech a umožňuje dělení hodinového signálu podle přednastavených konstant. Celý tento systém vytváří obvod master, který je kompatibilní s ostatními obvody využívající komunikaci I2C. Periferie byly naprogramovány pomocí programovacího jazyka VHDL a po naprogramování byly provedeny simulace v software ISE WebPACK. Po úspěšných simulacích byly provedeny realizace periferií pomocí vývojové desky V2MB1000 obsahující FPGA série Virtex-II XC2V1000. Funkčnost periferie realizující sériovou asynchronní komunikaci byla ověřena pomocí osciloskopu, kde zachycené oscilogramy potvrzují správnost návrhu a realizace. Komunikace I2C byla ověřena pomocí teplotního čidla DS1631. Komunikace pomocí standardu I2C s tímto obvodem proběhla bez problémů a odpovídala předpokladům. Výsledky komunikace s tímto obvodem jsou zobrazeny na oscilogramech.

8. Seznam literatury

[1] PINKER, J. POUPA M. Číslicové systémy a jazyk VHDL. Praha: BEN – technická literatura, 2006. 352 stran. ISBN 80-7300-198-5

[2] KOLOUCH, J. Programovatelné logické obvody. Elektronické texty a přednášek a počítačových cvičení. Brno:FEKT VUT v Brně, 2007

[3] Atmel Corporation. AVR 8-Bit RISC. [online]. 2008 – [cit. 20. února 2009]. Dostupné na WWW: http://www.atmel.com/products/avr/

[4] FRÝZA, T. FERDA, Z. ŠEBESTA, J. Mikroprocesorová technika. Elektronické texty laboratorních cvičení. Brno: FEKT VUT v Brně

[5] NXP Semiconductors – specifikace I2C [online]. květen 2007 – [cit. 5. února 2009] Dostupné na WWW: http://www.standardics.nxp.com/support/documents/i2c/pdf/i2c.bus. specification.pdf

[6] MATOUŠEK, D. Udělejte si z PC generátor, čítač, převodník, programátor...1.díl. Praha: BEN – technická literatura, 2004. 176 stran + CD ROM. ISBN 80-7300-036-9

[7] Xilinx. ISE WebPACK. [online]. 2008 – [cit. 20. února 2009]. Dostupné na WWW: http://www.xilinx.com/ise/logic_design_prod/webpack.htm

[8] Xilinx. ISE WebPACK. [online]. 2008 – [cit. 20. února 2009]. Dostupné na WWW: http://www.xilinx.com/publications/prod_mktg/ISE_sellsheet.pdf

[9] Xilinx. Virtex-II datasheet [online]. 2008 – [cit. 20. února 2009]. Dostupné na WWW: http://www.xilinx.com/support/documentation/data_sheets/ds031.pdf

[10] ŠTRAUS, P. *Vývoj sériových komunikačních periferií pomocí FPGA*.Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 24s. Vedoucí semestrální práce Ing. Tomáš Frýza, Ph.D.

[11] Virtex-II development board V2MB1000 [online]. 2008 – [cit. 20. února 2009]. Dostupné na WWW:http://www.ee.ucla.edu/~herwin/ocdma/Insight/V2MB_User_Guide_1_4.PDF

[12] Nápověda programu ISE WebPACK, spustitelná z programu. Xilinx

[13] DS1631 datasheet [online]. 2009 – [cit. 20. února 2009]. Dostupné na WWW: http://datasheets.maxim-ic.com/en/ds/DS1631-DS1731.pdf

[14] AVR274: Single-wire Software UART [online]. 2008 – [cit. 10. října 2008]. Dostupné na WWW: http://www.atmel.com/dyn/resources/prod_documents/AVR274.pdf

9. Seznam zkratek

AMS	Analogový a smíšený signál (**A**nalog and **M**ixed **S**ignals)
CLB	Konfigurovatelný logický blok (**C**onfigurable **L**ogic **B**lock)
FPGA	Programovatelné pole (**F**ield **P**rogrammable **G**ate **A**rray)
IEEE	(**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers)
IOB	Vstupně-výstupní blok (**I**nput-**O**utput **B**lock)
JTAG	Skupina výrobců IO usilující o testování IO od různých výrobců (**J**oint **T**est **A**ction **G**roup)
LSB	Nejméně významný bit (**L**east **S**ignificant **B**ite)
LUT	Vyhledávací tabulky (**L**ook-**U**p **T**able)
MSB	Nejvýznamnější bit (**M**ost **S**ignificant **B**ite)
NGC	Přípona souborů netlistu při syntéze
NGD	Přípona souboru vzniklého při překladu (**N**ative **G**eneric **D**atabase)
SCL	Hodinový signál (**S**erial **C**lock **L**ine)
SDA	Datový vodič (**S**erial **Da**ta Line)
T_{IOPI}	Zpoždění, které vznikne průchodem signálu IOB blokem
UART	Universální asynchronní přenos (**U**niversal **A**synchronous **R**eceiver and **T**ransmitter)
VHDL	Jazyk popisující hardware (**V**hsic **H**ardware **D**escription **L**anguage)
VHSIC	Vysokorychlostní integrované obvody (**V**ery **H**igh **S**peed **I**ntegrated **C**ircuits)