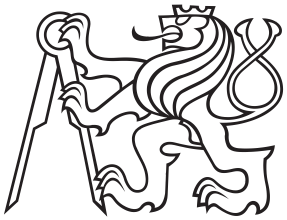
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická

Komunikace s přípravkem Spartan3E pomocí rozhraní RS232 v jazyce VHDL

Hašek Martin

Školitel: Ing. Lafata Pavel, Ph.D
Květen 2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Hašek** Jméno: **Martin** Osobní číslo: **478466**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra telekomunikační techniky**

Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Komunikace s přípravkem Spartan3E pomocí rozhraní RS232 v jazyce VHDL

Název bakalářské práce anglicky:

Using RS232 Interface and VHDL Language for Communication with Spartan 3E Kit

Pokyny pro vypracování:

Seznamte se s přípravkem Xilinx Spartan3E a jeho obsluhou pomocí jazyka VHDL. Vytvořte základní program pro obousměrnou komunikaci přípravku s PC připojeným prostřednictvím sériového rozhraní RS232 a hyperterminálu (nebo podobného prostředí). Z připojeného PC by mělo být možné ovládat základní prvky přípravku, např. stavové LED diody, naopak by měl přípravek Spartan3E po stisknuti tlačítka (či přepínače) odesílat předem stanovené znaky. Rozšiřte program o ovládání znakového LCD displeje na přípravku tak, aby textový řetězec odeslaný z PC přes sériové rozhraní byl zobrazen na LCD displeji.

Seznam doporučené literatury:

[1] Lafata, P. - Hampl, P. - Pravda, M.: Digitální technika. 1. vyd. Praha: Česká technika - nakladatelství ČVUT, 2011. 164 s. ISBN 978-80-01-04914-3.
[2] Pinker, J. - Poupa, M.: Číslicové systémy a jazyk VHDL. Praha : BEN - technická literatura, 2006. 349 s. ISBN 80-7300-198-5.
[3] Digilent: Spartan-3E Reference Manual [online]. Dostupné z: https://digilent.com/reference/programmable-logic/spartan-3e/reference-manual

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Pavel Lafata, Ph.D. katedra telekomunikační techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **24.02.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **16.02.2025**

Ing. Pavel Lafata, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Poděkování

Děkuji vedoucímu práce Ing. Pavlu Lafatovi, Ph. D za zapůjčení přípravku, příslušenství a pomoc při tvorbě této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Černošicích, 23. května 2023

.....

Abstrakt

Cílem této práce je implementace modulu v programovacím jazyce VHDL. Modul se stará o komunikaci mezi přípravkem Spartan-3E Starter Kit a terminálem po sériové lince RS232. Sériová linka je pak dále využita k odesílání textových řetězců a přijímání jednoduchých příkazů, které ovládají přípravek. Přijaté znaky se rovněž zobrazují na vestavěném LCD.

Klíčová slova: FPGA, sériová komunikace, asynchronní komunikace, VHDL

Školitel: Ing. Lafata Pavel, Ph.D

Abstract

The aim of this thesis is implementation of a module in VHDL programming language. The module provides communication between Spartan-3E Starter Kit and terminal over a RS232 serial interface. The serial interface is used to send text strings and to receive simple commands, which control the kit. Symbols, which have been received, are displayed on a built-in LCD.

Keywords: FPGA, serial communication, asynchronous communication, VHDL

Title translation: Using RS232 Interface and VHDL Language for Communication with Spartan 3E Kit

Obsah

Kapitola 1
Úvod

Programovatelná hradlová pole FPGA v dnešní době umožňují vytvořit téměř libovolné číslicové zařízení. Jejich hlavní výhoda spočívá v možnosti přeprogramovat přímo u zákazníka propojení jednotlivých hradel a získat tak logický obvod, který bude vyhovovat konkrétní aplikaci. Čip Xilinx Spartan-3E z roku 2004 obsahuje přes 10 000 [6, s. 12] logických buněk. Dnešní FPGA se mohou skládat z více než 10 milionů logických prvků, integrovat procesorová jádra ARM a další obvody sloužící k připojení na komunikační periferie a jejich výpočetní výkon může dosahovat až 8 TFLOPS [5, s. 12]. Přeprogramovatelnost může být stěžejní při vývoji zařízení pro komunikaci využívající standardů dalších generací, kdy dochází k zásadním úpravám ve standardu komunikace [3].

Cílem této práce je seznámit se s jazykem VHDL, kterým se FPGA programují, a implementovat řadič sériové linky RS232, který představuje standard již od 60. let 20. století [7] v komunikaci mezi dvěma zařízeními. Dále využít linku k odesílání příkazu z terminálu na přípravek a rozsvítit LED, přičemž přijaté znaky se zobrazí na vestavěném LCD. Z přípravku bude možné odeslat předem připravený textový řetězec na terminál.

Práce je rozdělena do dvou částí. V teoretické části popíši komunikaci na rozhraní RS232, základní vlastnosti jazyka VHDL a práci s řadičem LCD. V praktické části popíši vlastní řešení zadání.

Kapitola 2
Teoretická příprava

V této kapitole jsou stručně popsány vlastnosti programovatelných hradlových polí, jazyka VHDL, komunikace po sériové lince RS232 a LCD displeje na přípravku.

2.1 FPGA

FPGA (*Field-Programmable Gate Array*) patří do skupiny programovatelných logických obvodů (PLD). Umožňují využít masově vyráběného čipu, který se naprogramováním uzpůsobí pro konkrétní aplikaci zákazníka, což může snížit jak časové, tak finanční nároky na výrobu zařízení. Další výhodou je možnost opakovaného programování čipu v případě, že by se vyskytla chyba v návrhu, objevila bezpečnostní zranitelnost nebo by bylo potřeba přidat nové funkce do zařízení, která už byla zakoupena koncovými zákazníky. FPGA lze použít při vývoji a následně přejít k aplikačně specifickým obvodům (ASIC). Do PLD se dále řadí SPLD (*Simple Programmable Logic Device*) a CPLD (*Complex Programmable Logic Device*). Mezi přední výrobce patří mimo jiné společnosti AMD, Intel či Microchip. [1, 2, 3, 4]

K programování FPGA čipů se typicky používají jazyky pro popis hardwaru (HDL – *Hardware Description Language)* např. Verilog či VHDL. Pro lepší přístupnost nabízejí výrobci FPGA nástroje pro syntézu z populárnějších vyšších programovacích jazyků (C, C++ nebo OpenCL). [3]

2.1.1 Architektura FPGA

FPGA obsahuje matici logických buněk. Každá buňka obsahuje jednu či více vyhledávacích tabulek (LUT) s typicky čtyřmi až osmi vstupy a až dvěma výstupy, multiplexor a klopný obvod typu D. Jednotlivé buňky jsou mezi sebou propojeny vodorovně a svisle, přičemž propojení jsou programovatelné. Propoje jsou pak zakončeny vstupně/výstupními bloky. Kromě těchto tří základních prvků se do FPGA integrují i specializované obvody jako např. fázové závěsy,

pameť SRAM, bloky zajišťující digitální zpracování signálu (DSP), připojení na sběrnice (I^2C, CAN, USB) nebo procesorová jádra. [1, 3]

V současné době jsou FPGA vyráběna 10nm technologií, počet logických prvků přesahuje hranici 10 milionů a výpočetní výkon dosahuje 8 TFLOPS. Přímo do čipu jsou integrována jádra ARM s připojením na různé sběrnice. [5]

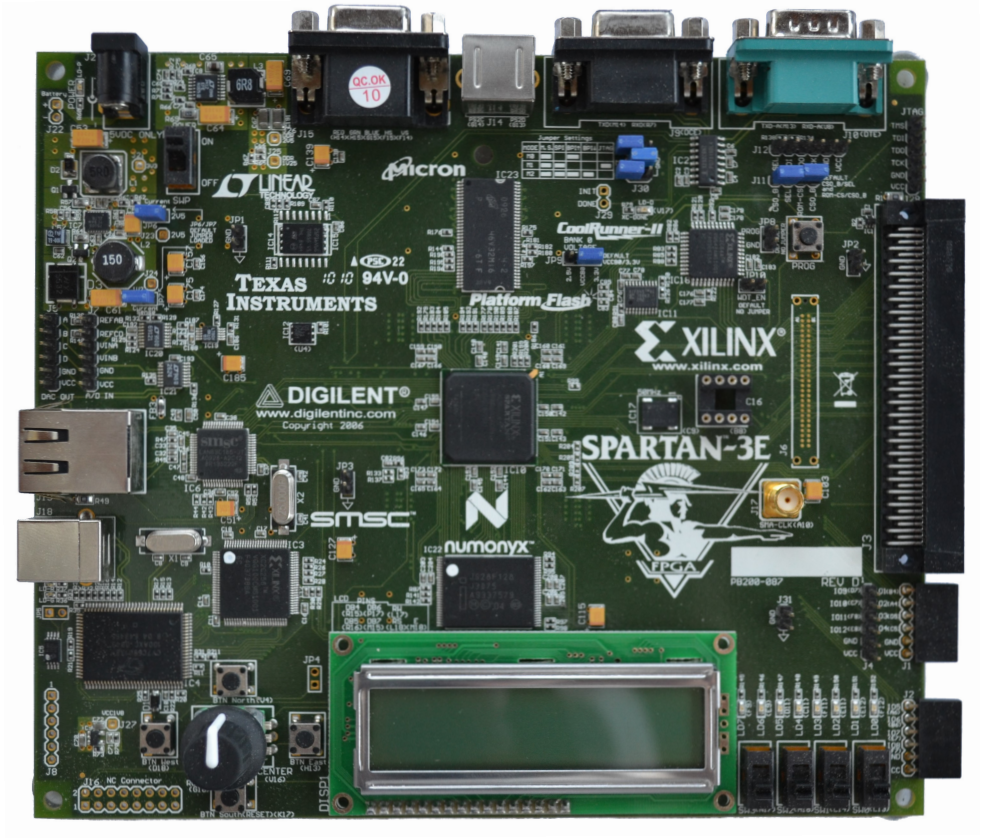
2.1.2 Spartan-3E Starter Kit

Přípravek Spartan-3E Starter Kit (obr. 2.1) je prototypovací deska vyvinutá společností Xilinx v roce 2006. Srdcem přípravku je FPGA čip Xilinx Spartan XC3S500E, který obsahuje přes 10 000 logických buněk. Přípravek je vybaven:

- 64 MB RAM pamětí,
- 16 MB paralelní flash pamětí,
- 2 MB sériovou flash pamětí,
- CPLD obvodem Xilinx XC2C64A,
- CPLD obvodem Xilinx XC2C256.

Dále se na desce nachází:

- dvouřádkový displej o 16 znacích,
- čtyři přepínače a čtyři tlačítka,
- rotační kodér s tlačítkem,
- osm LED,
- dva RS232 porty,
- PS/2 port, VGA port, RJ-45 port,
- AD a DA převodníky
- USB port pro programování,
- ladění a 50MHz oscilátor. [6]



Obrázek 2.1: Přípravek Spartan-3E Starter Kit

2.2 VHDL

Zkratka VHDL znamená *Very-High-Speed Integrated Circuits Hardware Description Language*, tedy jazyk pro popis velmi rychlých integrovaných obvodů. Vývoj byl zahájen v roce 1981 ministerstvem obrany Spojených států amerických. Jazyk umožňuje jak popisovat číslicové obvody a řetězit je do větších systémů, tak pro ně psát testovací programy (*testbench*) a simulovat jejich chování. Napsaný VHDL kód musí být (kromě testovacích programů) syntetizovatelný, to znamená, že po průchodu překladačem získáme zapojení jednotlivých logických obvodů pro konkrétní FPGA. [1, 2]

2.2.1 Struktura kódu

Základní návrhovou jednotkou v jazyce VHDL je entita (obr. 2.2). Klíčovým slovem *port* se deklarují vstupy a výstupy objektu. Entity lze také parametrizovat klíčovým slovem *generic*, čímž můžeme vytvořit např. sčítačku, která bude moct přijímat různě široké operandy v závislosti na parametru.

```
entity ADDER is
 generic ( LEN = 8 );
 port ( I1, I2 : in std_logic_vector (LEN-1 downto 0);
        O  : out std_logic_vector (LEN downto 0));
end entity ADDER;
```

Obrázek 2.2: Ukázka deklarace entity

Funkci a chování entity definuje architektura. Rozlišujeme tři typy architektur:

- strukturální – entita se modeluje propojením jednotlivých logických celků,
- dataflow – chování entity je popsáno pomocí Booleovy algebry. Příkazy se provádí souběžně,
- behaviorální – chování entity je popsáno algoritmem. Příkazy se vykonávají sekvenčně v těle procesu.

Každá entita může obsahovat více architektur. V první části architektury jsou deklarovány pomocné signály, které se šiří uvnitř entity. V druhé části jsou pak samotné příkazy, které definují chování entity.

```
architecture DATAFLOW of ANDGATE is
    signal SIG: std_logic;
begin
  SIG <= A and B;
end DATAFLOW;
```

Obrázek 2.3: Ukázka deklarace architektury

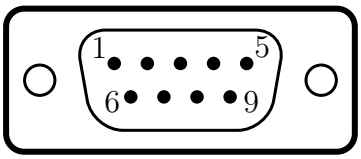
2.3 RS232

Standard RS232 užívaný k sériové komunikaci mezi dvěma body označovaných *DCE* (např. modem či jiná periferie) a *DTE* (např. počítač). Byl vyvinut v roce 1962 americkou organizací EIA (*Electronic Industries Alliance*). V průběhu let došlo k několika revizím, které zvýšily propustnost rozhraní, maximální délku vedení či představily nové konektory. Poslední revize z roku 1997 nese označení TIA/EIA- -232-F. [7]

2.3.1 Zapojení konektoru DB9

Typickým zástupcem konektorů rozhraní RS232 je Canon DB9. Na obr. 2.4 je znázorněn tvar konektoru a rozmístění jednotlivých pinů. Jejich funkce je popsána v tab. 2.1. [7]

Přípravek Spartan-3E nepodporuje řízení toku, a tedy připojeny jsou pouze piny TxD, RxD a GND. [6]



Obrázek 2.4: Konektor DB9 [8]

Tabulka 2.1: Funkce pinů konektoru DB9 [7]

Pin	Zkratka	Význam
1	DCD	Strana DCE přijímá nosný signál ze vzdáleného zařízení DCE
2	TxD	Vysílaná data
3	RxD	Přijímaná data
4	DTR	Strana DTE je připravena k příjmu či odesílání dat
5	GND	Referenční uzemnění.
6	DSR	Strana DCE je připravena k příjmu či odesílání dat
7	RTS	Strana DTE je právě připravena k odesilání dat
8	CTS	Strana DCE je připravena k příjmu dat
9	RI	Indikace vyzvánění

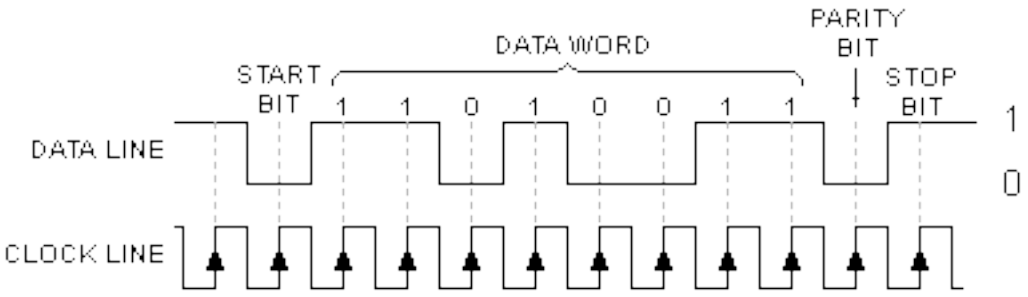
2.3.2 Popis komunikace na rozhraní RS232

Přenos dat pracuje na asynchronním principu tzn., že obě zařízení si generují vlastní hodinový signál, přičemž kmitočty se musí shodovat, aby komunikace mezi zařízeními probíhala správně.

Standardem je definována logická 0 na straně vysílače napětím v rozmezí od +5 V do +15 V a logická 1 napětím mezi -5 V a -15 V. Na straně přijímače je logická 0 stanovena napětím od +3 V do +15 V a logická 1 napětím mezi -3 V a -15 V. Hodnota napětí mezi ±3 V je nedefinována. [7]

Standardem jsou dokonce povolena napětí až ±25 V, což je pro *nízkonapěťovou tranzistorově-tranzistorovou logiku* (LVTTL), která pracuje při napětí 3,3 V, nepřípustné. Přípravek tedy obsahuje integrovaný obvod MAX3232, který zajišťuje převod logických úrovní mezi LVTTL a RS232. [6]

Na obr. 2.5 je ukázka asynchronní sériové komunikace. Při nečinnosti je linka držena v log. 1. Komunikace je zahájena start bitem, který nabývá log. 0. Dále



Obrázek 2.5: Příklad komunikace RS232 [9]

následují datové bity (typicky 8). Konec komunikace je uvozen stop bitem, který má hodnotu log. 1. Přechod mezi log. 0 a 1 není z důvodu parazitní kapacity vedení okamžitý, a proto je nutné číst bity uprostřed časového okna, kdy je signál ustálený. Pro zajištění lepší spolehlivosti přenosu může být mezi posledním datovým a stop bitem ještě bit paritní. Ve standardu jsou uvedeny 4 typy parit:

- *lichá parita* – pokud byl přenesen lichý počet jedniček, nabývá paritní bit logické 1,
- *sudá parita* – pokud byl přenesen sudý počet jedniček, nabývá paritní bit logické 1,
- *mark* – paritní bit má vždy hodnotu logické 1,
- *space* – paritní bit má vždy hodnotu logické 0.

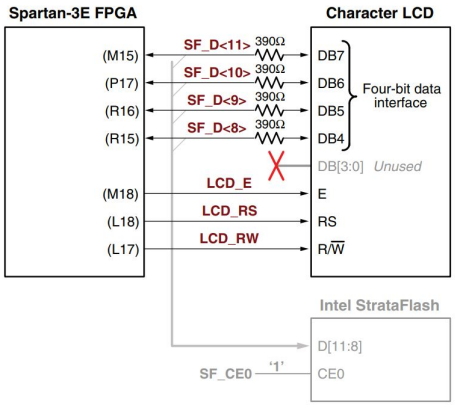
Výpočet děličky hodinového signálu

Nastavení rychlosti komunikace se provádí děličkou hodinového signálu. Vztahem 2.1 vypočteme hodnotu dělitele D, kterým dělíme kmitočet hodinového signálu zařízení f_{clk}. Rychlost sériové linky v je vhodné volit ze standardní řady, avšak teoreticky je možné zvolit libovolnou rychlost. Důležité je, aby byla shodná na obou zařízeních. Wilson [10, s. 219] uvádí rychlosti např. 9600, 19200 či 115200 Bd.

$$D = \frac{f_{\text{clk}}}{v} \tag{2.1}$$

2.4 LCD displej

Vestavěný dvouřádkový LCD displej o 16 znacích obsahuje řadič Sitronix ST7066U, který je funkčně ekvivalentní s řadiči Hitachi HD44780, SMOS SED1278, Samsung S6A0069X či KS0066U. Řadič umožňuje pracovat v osmibitovém a čtyřbitovém režimu. Na obr. 2.6 je zobrazeno zapojení LCD řadiče s FPGA. Je využito zapojení ve čtyřbitovém režimu, což je implementačně náročnější, avšak jsou ušetřeny 4 I/O linky do FPGA. Zapojení je rovněž sdíleno s paralelní flash pamětí. Při používání LCD displeje musí být přivedena log. 1 na vstup *CE0* paměti. [6]



Obrázek 2.6: Zapojení řadiče LCD displeje [6]

Řadič obsahuje celkem 3 paměti:

- CG ROM – paměť, ve které jsou uloženy ASCII a japonské kana znaky, znaky řecké abecedy a matematické symboly,
- DD RAM – paměť pro uložení znakových kódů, které se mají zobrazit na displeji. Znakové kódy odpovídají adrese v CG ROM. Do paměti lze uložit až 2×40 znakových kódu, z nichž 2×16 lze zobrazit,
- CG RAM – paměť, do které lze uložit až 8 vlastních znaků. [6]

Příkazy řadiče

Řadič podporuje následující příkazy:

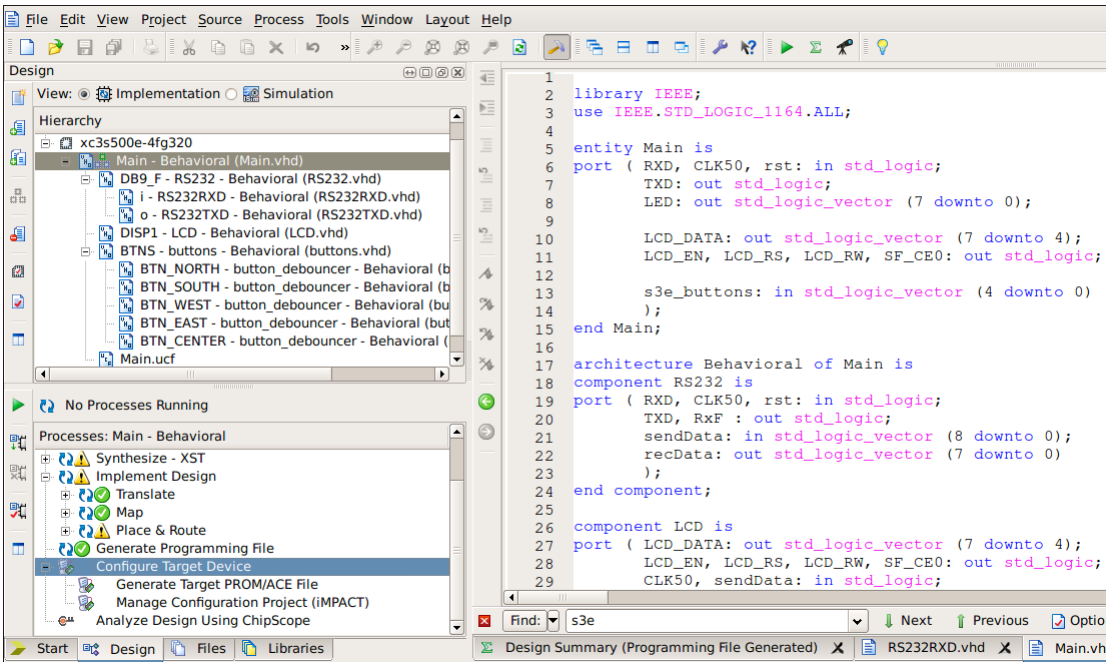
- vymazání displeje,
- vrácení kurzoru do výchozí polohy,
- nastavení směru pohybu kurzoru,
- zapnutí nebo vypnutí displeje, blikání a zobrazení kurzoru,

- posun kurzoru nebo displeje bez změny obsahu v DD RAM. Tato funkce umožňuje rolovat displej a zobrazit tak znakové kódy, které jsou uloženy za 16. znakem řádku,
- nastavení čtyř nebo osmibitového režimu, počet řádků a font znaků,
- nastavení CG RAM adresy,
- nastavení DD RAM adresy,
- přečtení příznaku zaneprázdnění displeje a aktuální CG/DD RAM adresy,
- zápis dat do CG/DD RAM,
- čtení dat z CG/DD RAM. [6]

2.5 Vývojové prostředí ISE WebPACK

Na obr. 2.7 je vidět uživatelské rozhraní vývojové prostředí ISE WebPACK od společnosti Xilinx. Balíček umožňuje založení projektu přímo pro konkrétní FPGA nebo CPLD obvod od Xilinx. Podporuje jak jazyk VHDL, tak Verilog. Umí provádět syntézu i simulaci naprogramovaných obvodů. Lze nastavit podmínky časování a program při syntéze dohlíží na jejich splnění. Je také možné se podívat, jak moc využíváme možnosti FPGA čipu, a rozhodnout se, zda nebude potřeba použít čip z vyšší řady, nebo naopak z řady nižší. Po syntéze a úspěšném namapování obvodů lze vygenerovat soubor, kterým se FPGA naprogramuje. K naprogramování slouží komponenta iMPACT. [11]

Poslední verze 14.7 je z roku 2013. Podporuje Windows XP, Windows 7 a Linux. V operačním systému Ubuntu 20.04 LTS občas došlo k pádu programu, avšak ke ztrátě napsaného kódu nedošlo. Náhradou je balíček Xilinx Vivado, který však FPGA Spartan-3E nepodporuje.



Obrázek 2.7: Uživatelské rozhraní ISE WebPACK

Kapitola 3
Implementace

Přípravek Xilinx Spartan-3E Starter Kit byl programován ve vývojovém prostředí ISE Design Suite 14.7 v programovacím jazyce VHDL. K propojení počítače s přípravkem byl použit převodník USB-RS232 a terminál PuTTY. V následující kapitole popisuji jednotlivé entity, které jsem naprogramoval.

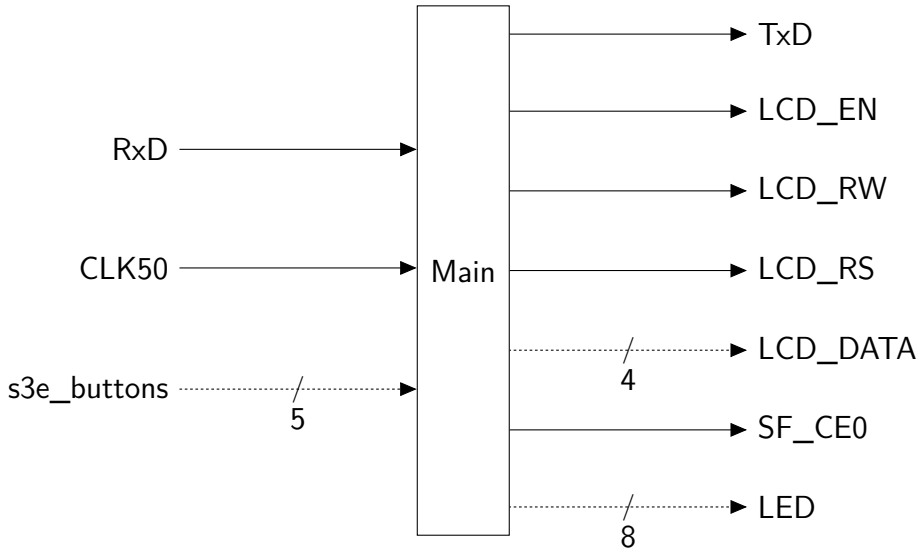
3.1 Modul Main

Nejvýše postaveným modulem v implementaci je modul *Main*, který propojuje periferie a další moduly (*RS232*, *LCD*, *buttons*), jejichž funkci popisuji dále. V tabulce 3.1 je popsáno fyzické zapojení mezi modulem a I/O bloky FPGA čipu, které jsou fyzicky propojeny s periferiemi na desce. Tyto definice jsou umístěny v souboru *Main.ucf* a jsou potřeba pro generování binárního souboru, kterým se FPGA programuje.

Kromě propojení podmodulů zajišťuje procesy, které přeposílají znaky ze sériové linky do LCD a vykonávají jednoduché příkazy. Činnost procesů je popsána níže.

Tabulka 3.1: Zapojení modulu Main

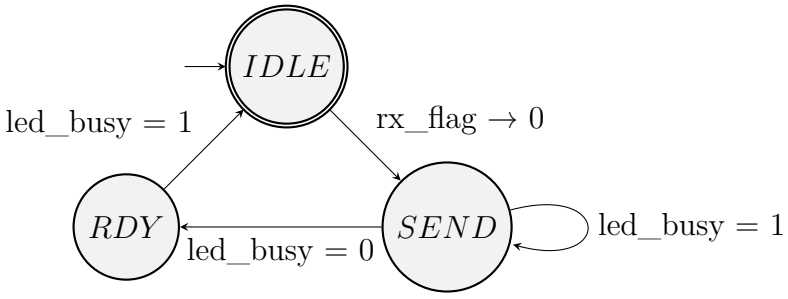
Název signálu	Orientace	Pin FPGA
CLK50	Vstup	C9
RXD	Vstup	R7
TXD	Výstup	M14
LED[7..0]	Výstup	F12, E12, E11, F11, C11, D11, E9, F9
LCD_EN	Výstup	M18
LCD_RS	Výstup	L18
LCD_RW	Výstup	L17
LCD_DATA[7..4]	Výstup	M15, P17, R16, R15
SF_CE0	Výstup	D16
s3e_buttons[4..0]	Vstup	V4, D18, V16, H13, K17



Obrázek 3.1: Modul Main

3.1.1 Proces pro přeposlání znaků na LCD

Proces lze popsat stavovým automatem na obr. 3.2. Proces čeká na přechod příznaku *rx_flag* sériové linky do log. 0, čímž je signalizován příjem rámce. Po příjmu dojde k přenesení rámce do vnitřní vyrovnávací paměti *dataBuffINT*. Pokud není LCD řadič zaneprázdněný (signál *lcd_busy*), přenese se rámec do vyrovnávací paměti LCD řadiče *dataBuffLCD* a příznakem *lcd_send* v log. 1 se odešle. Jakmile se příznak zaneprázdněnosti změní na log. 1, *lcd_send* přejde do log. 0 a proces se vrátí do výchozího stavu.

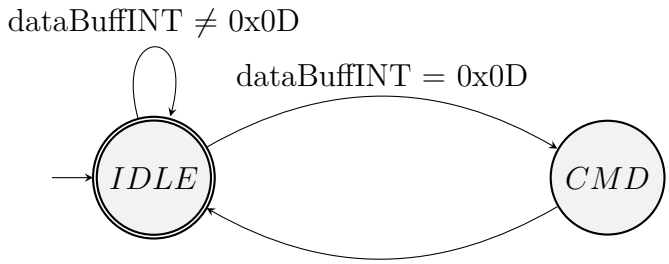


Obrázek 3.2: Stavový automat přeposílání znaku na LCD

3.1.2 Proces dekódování příkazu

Stavový automat na obr. 3.3 popisuje činnost tohoto procesu. Ve výchozím stavu *IDLE* proces čeká na náběžnou hranu signálu *lcd_send*, čímž je signalizován příjem rámce ze sériové linky. Ten se uloží do vyrovnávací paměti *dataBuffINT*. Pokud se jedná o znak, uloží jej do paměti. Jedná-li se o mazání znaku

(*backspace* – 0x7F), smaže poslední přijatý znak a vrátí ukazatel o jednu pozici zpět. V případě, že dojde k přijetí znaku signalizující stisknutí klávesy Enter (*carriage return* – 0x0D) dojde k přechodu do stavu *CMD*, kde se dekóduje příkaz. Přípravek umí dekódovat příkaz, který rozsvítí, nebo zhasne konkrétní LED na desce. Syntaxe příkazu je: led *X* { on | off }, kde X je číslo diody umístěné na desce přípravku. K převodu čísla X z ASCII kódu do desítkové soustavy slouží funkce *ascii2int*, která sečte příspěvky spodních 4 bitů ASCII znaku. Výsledek po odeslání příkazů „led 2 on“ a „led 4 on“ je vidět na obr. 3.4.



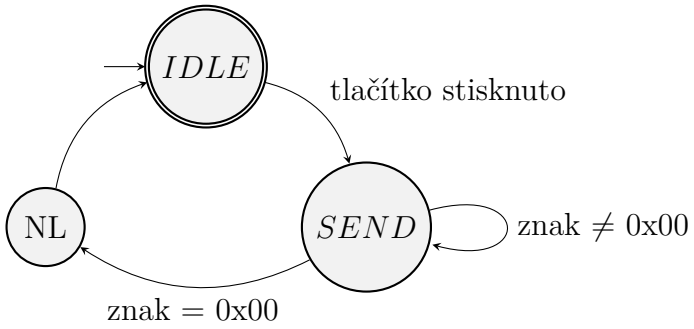
Obrázek 3.3: Stavový automat dekódování příkazu

Obrázek 3.4: Řetězce odeslané po stisku tlačítek

3.1.3 Proces odesílání řetězce na RS232

Po stisknutí některého z pěti tlačítek na desce přípravku dojde k odeslání předem stanoveného řetězce na sériovou linku. Proces lze charakterizovat stavovým automatem na obr. 3.6. Ve výchozím stavu *IDLE* čeká na stisk některého z tlačítek. Po stisku nahraje předem připravený textový řetězec do paměti a ve stavu *SEND* jej znak po znaku odešle do modulu *RS232*. Při načtení hodnoty 0x00, kterou je řetězec ukončen, přejde automat do stavu *NL* a na sériovou linku odešle znaky nová řádka (*line feed* – 0x0A) a vrácení kurzoru na začátek řádky (*carriage return* – 0x0D). Poté se vrátí zpět do výchozího stavu. Na obr. 3.5 je vidět výstup sériové linky po stisku tlačítek.

Obrázek 3.5: Řetězce odeslané po stisku tlačítek



Obrázek 3.6: Stavový automat odesílání řetězce přes sériovou linku

3.2 Modul RS232

Modul RS232 se skládá ze dvou částí – přijímacího *Rx* a vysílacího *Tx* modulu. V manuálu k přípravku Spartan-3E [6, s. 22] se dočteme, že integrovaný oscilátor má kmitočet $f_{\text{clk}} = 50$ MHz. Pro přenosovou rychlost $v = 9600$ Bd vypočteme dle vztahu 2.1 hodnotu dělitele $D = 5208$.

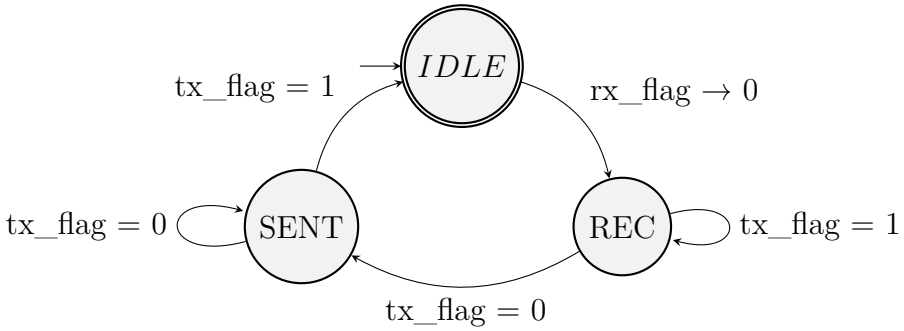
Na vstup *CLK50* je přiveden signál z integrovaného oscilátoru, vstup *RXD* je zapojený na pin RxD sériové linky a výstup *TXD* je přiveden na pin TxD sériové linky. Osmibitový rámec k odeslání se přivede na sběrnici *sendData[7..0]*. Devátý bit sběrnice *sendData(8)* je využívaný jako příznak k odeslání rámce. Výstup *RxF* označuje příznak právě probíhajícího příjmu rámce. Na výstupní sběrnici *recData* o šířce 8 bitů se objeví rámec po bezchybném přijetí.

3.2.1 Zpětná vazba uživateli terminálu

Modul se dále stará o okamžitou zpětnou vazbu uživateli. Rámec, který je přijat, se obratem odešle do vysílacího modulu a zobrazí se v terminálu. Proces, který zajišťuje zpětnou vazbu lze charakterizovat stavovým automatem na obr. 3.7.

Ve výchozím stavu *IDLE* čeká proces na sestupnou hranu příznaku příjmu sériové linky *rx_flag*. Při detekci sestupné hrany uloží přijatý rámec do vnitřní

vyrovnávací paměti *dataBuffINT* a přejde do stavu ***REC**ieved*, kde ověří nečinnost vysílání sériové linky, který signalizuje příznak *tx_flag* v log. 0. Při detekci nečinnosti nahraje rámec do vyrovnávací paměti vysílacího členu sériové linky *datBufferTX* a dá mu pokyn k zahájení sekvence odesílání rámce nastavením signálu *send* do log. 1. Poté přejde do stavu *SENT*, kde detekuje příznak vysílání sériové linky, nastaví signál *send* do log. 0 a přejde zpět do výchozího stavu.



Obrázek 3.7: Stavový automat zpětné vazby uživateli

3.2.2 Modul Rx

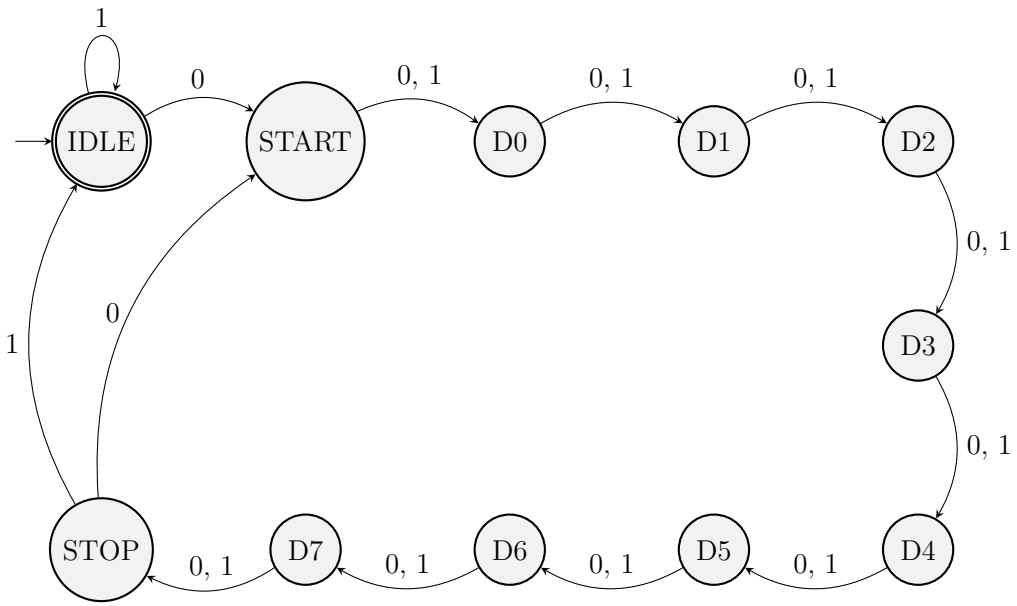
Na vstup *CLK* je zapojen hodinový signál a vstup *RXD* je připojen na pin RxD sériové linky. Výstupem modulu je osmibitová sběrnice *recievedData* a signál *rx_flag*, který signalizuje probíhající příjem dat na sériové lince.

Činnost modulu by se dala popsat stavovým diagramem (obr. 3.8). Ve stavu *IDLE* modul čeká na okamžik, kdy sériová linka přejde z log. 1 do 0. Po přijetí log. 0 přejde do stavu *START*, poté modul přijme 8 datových bitů (stavy *D0* až *D7*) a nakonec přijme stop bit ve stavu *STOP*. Jednotlivé bity jsou vzorkovány uprostřed časového okna, aby měla linka dostatek času k ustálení úrovně. Modul pak zkontroluje zda přijatý start bit byl opravdu log. 0 a stop bit log. 1 a data odešle na sběrnici *recievedData*. Pokud start bit či stop bit neodpovídají požadovaným hodnotám, data se zahodí. Ze stavu *STOP* pak přejde zpět do výchozího stavu *IDLE*, nebo rovnou do stavu *START* dle hodnoty sériové linky.

3.2.3 Modul Tx

Na vstup *CLK* je zapojen hodinový signál. Vstup *send* slouží jako povel k zahájení přenosu bitů, které jsou zapsané na osmibitovém vstupu *data2send*. Na výstupu je připojen TxD pin sériové linky a signál *tx_flag*, který signalizuje probíhající odesílání dat na sériovou linku.

Činnost modulu Tx je velmi podobná modulu Rx. Modul Tx je ve výchozím stavu *IDLE*, ve kterém drží sériovou linku v log. 1. Při přijetí signálu *send* = 1



Obrázek 3.8: Stavový diagram modulu *Rx*

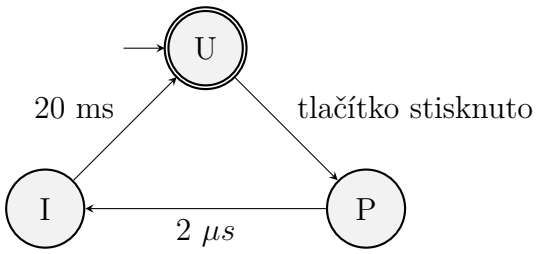
začne přenos rámce (start bit, 8 bitů ze sběrnice *data2send* a stop bit), příznak *tx_flag* nastaví do log. 1 a postupně projde ostatní stavy. Po odeslání stop bitu se modul vrátí do výchozího stavu a příznak *tx_flag* nastaví zpět do log. 0.

3.3 Modul buttons

Modul *buttons* slouží k propojení všech pěti tlačítek, která jsou umístěna na desce přípravku. Tlačítka nejsou připojena přímo, ale přes moduly *debouncer*, aby nedocházelo k zákmitům při stisku. V manuálu k přípravku [6, s. 17] se uvádí, že za definici pinu FPGA v souboru *Main.ucf* se musí specifikovat příznak *PULLDOWN*, čímž se k tlačítku přípojí *pulldown* rezistor, který je spojený s nulovým potenciálem, aby po uvolnění tlačítka došlo k přechodu do log. 0.

3.3.1 Modul debouncer

Na obr. 3.9 je stavový automat, který popisuje funkce modulu debouncer. Ve výchozím stavu ***U**NPRESSED* čeká na stisk tlačítka (signál *i_btn*). Po stisku tlačítka přejde do stavu ***P**RESSED* a odešle krátký puls (2 μs) na výstup *o_btn*. Po odeslání pulsu přejde do stavu ***I**DLE*, kdy je další stisk tlačítka na chvíli (20 ms) softwarově blokován. Poté přejde zpět do výchozího stavu.



Obrázek 3.9: Stavový automat přeposílání znaku na LCD

3.4 LCD řadič

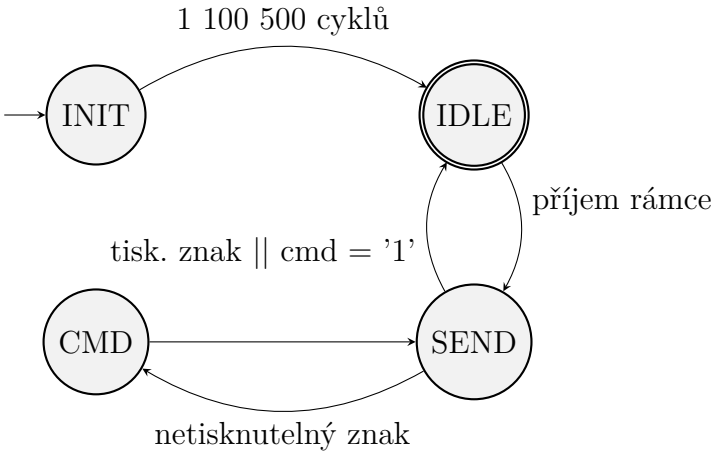
Činnost modulu *LCD řadič* popisuje stavový automat na obr. 3.10. Ve výchozím stavu *INIT* dochází k inicializaci displeje – nastavení čtyřbitového režimu, která je popsána v [6, s. 53] a skládá se z:

- ustálení napěťových úrovní – 15 ms (750 000 hodinových cyklů při 50 MHz),
- LCD_DATA = 0x3 a LCD_E = '1' – puls po dobu 12 cyklů,
- počkat alespon 4,1 ms (205 000 cyklů),
- LCD_DATA = 0x3 a LCD_E = '1' – puls po dobu 12 cyklů,
- počkat alespon 100 μs (5 000 cyklů),
- LCD_DATA = 0x3 a LCD_E = '1' – puls po dobu 12 cyklů,
- počkat alespon 40 μs (2 000 cyklů),
- LCD_DATA = 0x2 a LCD_E = '1' – puls po dobu 12 cyklů,
- počkat alespon 40 μs (2 000 cyklů),

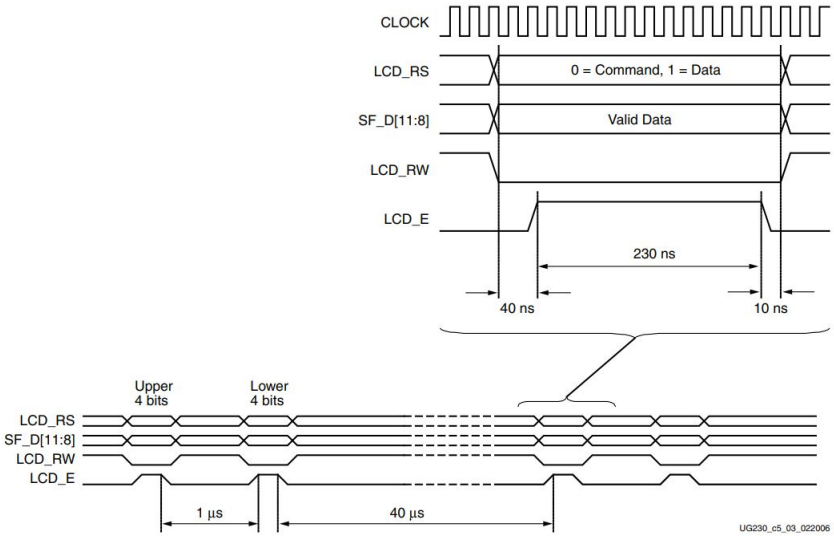
poté je nutné nastavit chování displeje. Nejprve se zvolí písmo a počet řádků: LCD_DATA = 0x28, poté se nastaví posunu kurzoru doprava bez posunu displeje: LCD_DATA = 0x06, dále zapnutí displeje a blikání kurzoru: LCD_DATA = 0x0F a nakonec se vymaže displej: LCD_DATA = 0x01. Osmibitové příkazy se musí odesílat ve dvou čtyřbitových *nibblech*. Nejdříve horní a poté spodní čtyři bity. Na obr. 3.11 je znázorněno časování jednotlivých signálů při čtyřbitovém režimu komunikace.

Po prvotním nastavení automat (obr. 3.10) přejde do stavu *IDLE*, kde čeká na rámec ze sériové linky. Po přijetí rámce přechází do stavu *SEND*. Pokud je znak tisknutelný, zobrazí se ihned na displeji. Pokud se jedná o znak *carriage return*, kurzor se posune na začátek dalšího řádku. Přijme-li řadič znak *delete* nebo *backspace* smaže poslední vytisknutý znak a posune kurzor o jednu pozici zpět. Tyto příkazy netisknutelných znaků se připraví ve stavu *COMMAND*,

kde dojde k vypočítání adresy DD RAM a nastavení signálu *cmd* do log. 1. Samotné odeslání příkazu do řadiče probíhá ve stavu *SEND*.



Obrázek 3.10: Stavový diagram LCD řadiče



Obrázek 3.11: Časování čtyřbitové komunikace s řadičem LCD

3.5 Shrnutí

Přípravek tedy umí při stisku některého z tlačítek na desce odeslat po sériové lince předem stanovené textové řetězce. Dále umí přijmout znaky a další speciální ASCII kódy (*CR, DEL, BS*), zobrazit je na vestavěném LCD a případně dekódovat přijatý příkaz, kterým lze rozsvítit konkrétní LED na desce. Při psaní do terminálu dává uživateli zpětnou vazbu okamžitým odesíláním znaků, které byly po sériové lince přijaty.

3.6 Možné rozšíření

LCD řadič by se dal rozšířit o pohyb kurzoru pomocí šipek na klávesnici na straně terminálu a využít funkci rolování celého řádku. Velmi zajímavá by byla celková parametrizace jednotlivých modulů, čímž by byl kód využitelný pro široké spektrum aplikací pouhým nastavením konstant přímo v deklaraci komponent. V neposlední řadě by se dal impementovat příkaz pro řadič RS232, který by měnil parametry přenosu jako rychlost, počet bitů či parita za běhu systému. Pro lepší kontrolu neporušenosti dat by se dalo zavést trojnásobné vzorkování bitů s většinovým hlasováním popsané v [12, s. 419].

Kapitola 4

Závěr

Cílem práce bylo implementovat komunikaci s přípravkem Xilinx Spartan-3E Starter Kit pomocí rozhraní RS232 v jazyce VHDL a následně využít linku pro ovládání přípravku pomocí jednoduchých příkazů a obrazit přijaté znaky na vestavěném LCD. Dále po stisknutí tlačítka na přípravku odeslat předem stanovený textový řetězec do terminálu.

Všechny body zadání se mi podařilo splnit. Pro splnění zadání bylo potřeba osvojit si specifika programovacího jazyka VHDL, práci s LCD řadičem a průběh komunikace na sériové lince RS232. Přípravek rovněž ihned odesílá přijatá data zpět do terminálu a poskytuje tak zpětnou vazbu uživeteli terminálu. Komunikuje probíhá rychlostí 9600 Bd. Jednotlivé moduly lze využít i v dalších aplikacích.

Literatura

[1] PINKER, Jiří a Martin POUPA. Číslicové systémy a jazyk VHDL. Praha: BEN - technická literatura, 2006. ISBN 80-7300-198-5.

[2] LAFATA, Pavel, Petr HAMPL a Michal PRAVDA. Digitální technika. V Praze: České vysoké učení technické, 2011. ISBN 978-80-01-04914-3.

[3] MAXFIELD, Clive. Fundamentals of FPGAs: What Are FPGAs and Why Are They Needed? *In: DigiKey* [online]. 14. 11. 2019 [cit. 14. 5. 2022]. Dostupné z: https://www.digikey.cz/en/articles/fundamentals-of-fpgas-what-are-fpgas-and-why-are-they-needed

[4] ASICNORTH. *ASIC vs FPGA difference*. IN: Asic North [online]. 2020 [cit. 14. 5. 2022]. Dostupné z: https://www.asicnorth.com/blog/asic-vs-fpga-difference/

[5] INTEL. *Intel FPGA Product Catalog* In: Intel [online]. 2021. [cit. 14. 5. 2022]. Dostupné z: https://www.intel.com/content/dam/www/central-libraries/us/en/documents/product-catalog.pdf

[6] XILINX. *Spartan-3E FPGA Starter Kit Board User Guide* [online]. 2011. [cit. 8. 12. 2021]. Dostupné z: https://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf

[7] TEXAS INSTRUMENTS. *Interface Circuits for TIA/EIA-232-F* [online]. 2002. [cit. 8. 12. 2021]. Dostupné z: https://www.ti.com/lit/an/slla037a/slla037a.pdf

[8] Aeroid. CAN Connecteur. In: *Wikipedia* [online]. 23. 5. 2021 [cit. 8. 12. 2021]. Dostupné z: https://commons.wikimedia.org/wiki/File:CAN_Connecteur.svg

[9] In: Vývoj.HW.cz [online]. 12. 12. 2005 [cit. 8. 12. 2021]. Dostupné z: https://vyvoj.hw.cz/files/rozhrani/images/rs232-f1.gif

[10] WILSON Peter. *Design Recipes for FPGAs : Using Verilog and VHDL*. 2. vyd. Elsevier, 2015. ISBN 978-0-0809-7129-2

[11] Xilinx. *ISE WebPACK Design Software*. IN: Xilinx [online]. [cit. 20. 5.+2022]. Dostupné z: https://www.xilinx.com/products/design-tools/ise-design-suite/ise-webpack.html

[12] TEXAS INSTRUMENTS. *MSP430x2xx Family User's Guide*. In: Texas Instuments [online]. 2013. [cit. 30. 1. 2022]. Dostupné z: https://www.ti.com/lit/ug/slau144j/slau144j.pdf

[13] MAXIM INTEGRATED. *Clock accuracy requirements for UART communications protocol*. In: Maxim Integrated [online]. 2003. [cit. 8. 12. 2021]. Dostupné z: https://www.maximintegrated.com/en/design/technical-documents/tutorials/2/2141.html

Příloha A

Obsah přiloženého CD

- code – složka s VHDL kódem
  - buttons.vhd
  - debouncer.vhd
  - LCD.vhd
  - Main.ucf
  - Main.vhd
  - RS232.vhd
  - RS232RXD.vhd
  - RS232TXD.vhd
- hasekma2_bp.pdf – elektronická verze této práce