ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra telekomunikační techniky

Využití vstupních a výstupních rozhraní v jazyce VHDL

Use of input and output interfaces in VHDL

DIPLOMOVÁ PRÁCE

Studijní program:	Komunikace, multimédia a elektronika
Studijní obor:	Sítě elektronických komunikací
Vedoucí práce:	Ing. Lafata Pavel Ph.D.

Bc. Tomáš Pehnelt

Praha,Květen 2016

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra telekomunikační techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Tomáš Pehnelt**

Studijní program: Komunikace, multimédia a elektronika
Obor: Sítě elektronických komunikací

Název tématu: **Využití vstupních a výstupních rozhraní v jazyce VHDL**

Pokyny pro vypracování:

Prozkoumejte možnosti vývojových kitů od společnosti Xilinx - Spartan3E a Nexys3/4, obsahující množství různých standardizovaných rozhraní a konektorů. Vytvořte v jazyce VHDL příslušné kódy pro obsluhu různých rozhraní, např. Ethernet, PS/2, D-Sub, RS232 aj. Využijte tato rozhraní pro realizaci ukázkové komunikace s vybranými periferiemi a vytvořte v jazyce VHDL kódy pro jejich praktické využití.

Seznam odborné literatury:

[1] Pinker, J.; Poupa, M.: *Číslicové systémy a jazyk VHDL* (1. vydání). BEN - Technická literatura, Praha 2006. ISBN 80-7300-198-5.
[2] Xilinx - *Manuály a materiály k vývojovým kitům Spartan3E a Nexys3/4.*
[3] Digilent - *Manuály a materiály k vývojovým kitům Spartan3E a Nexys3/4.*

Vedoucí: Ing. Pavel Lafata, Ph.D.

Platnost zadání: do konce letního semestru 2016/2017

L.S.

prof. Ing. Boris Šimák, CSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 12. 2015

Prohlášení

Prohlašuji, že jsem svou diplomovou práci na téma Využití vstupních a výstupních rozhraní v jazyce VHDL vypracoval samostatně s přispěním vedoucího práce a použil jsem pouze literaturu uvedenou v přiloženém seznamu.

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

V Praze dne 27.5.2016 .....

Poděkování

Rád bych poděkoval vedoucímu diplomové práce Ing. Pavlu Lafata Ph.D. za zapůjčení materiálů, za cenné připomínky a odborné rady při tvorbě práce.

Anotace

Diplomová práce se zabývá přípravky Digilent Nexys 4, Digilent Nexys 3 a Spartan-3E Starter Board a rozhraními, kterými tyto přípravky disponují. Dále se zabývá implementovaným řízením některých těchto rozhraní. Mezi implementovaná rozhraní patří sběrnice PS/2 určená pro obsluhu klávesnice, obvod UART, který přijímá znaky z připojeného PC a nakonec port VGA pro zobrazení přijatých dat.

Klíčová slova

VHDL,FPGA,PS/2,VGA,UART, Digilent Nexys 4, Digilent Nexys 3, Spartan-3E Starter Board

Summary

The thesis deals with Digilent products Nexys 4, Digilent Nexys 3 and Spartan-3E Starter Board and also with interfaces which these products have. It also deals with implemented management of these interfaces. The implemented interfaces includes PS/2 designed to operate the keyboard, the circuit UART that receives characters from the connected PC and port VGA display for received data.

Index Terms

VHDL,FPGA,PS/2,VGA,UART, Digilent Nexys 4, Digilent Nexys 3, Spartan-3E Starter Board

Obsah

1 Zadání práce

Cílem diplomové práce na téma Využití vstupních a výstupních rozhraní v jazyce Very high speed integrated circuits hardware design language (VHDL) bylo prozkoumat možnosti vývojových kitů s Field Programmable Gate Array (FPGA) od společnosti Xilinx. A to konkrétně na jednom z přípravků Spartan 3E Starter Board, Digilent Nexys 3 a Digilent Nexys 4. Vytvořit funkční návrhy řízení vybraných periférií a prostřednictvím demonstrační praktické ukázky toto řízení použít.

Prvním úkolem bylo prozkoumat a navrhnout řízení rozhraní PS/2 s tím, že je nutné vyřešit situaci odpojení a znovu zapojení zařízení na této sběrnici. Z toho důvodu bylo také nutné vyřešit komunikaci oběma směry (příjem i vysílání, více viz. kapitola 2.1).

Dalším úkolem bylo prozkoumat a použít komunikaci spojenou se standardem RS-232 a při implementaci zvážit, kterou část obvodu Universal Asynchronous reciever and transmitter (UART) opravdu použít a jak vyřešit předávání dat dalším komponentám v návrhu.

Dále bylo cílem diplomové práce navrhnout zobrazování pomocí portu Video Graphics Array (VGA), prozkoumat zobrazování barev a vykreslování různých tvarů a znaků. Implementovat vlastní druh fontu pomocí něhož budou zobrazována data přijatá z ostatních rozhraní.

Posledním cílem bylo všechny celky zkompletovat a společně je využít u již zmiňované demonstrační ukázky vyřešit problematiku předávání dat mezi různými komponentami.

2 Úvod do použitých sběrnic

V této části práce se nachází funkční popis vybraných periferií, které jsou použity v praktické části diplomové práce.

2.1 Sběrnice PS/2

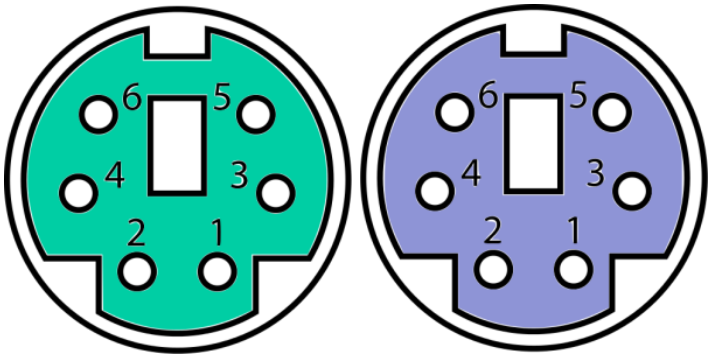
V kapitole je nastíněno fungování sběrnice PS/2 pro ovládání klávesnic a počítačových myší. Při vypracovávání bylo čerpáno z následujících zdrojů: [1], [2], [3], [4], [5] a [6].

2.1.1 Obecné informace

V případě PS/2 se jedná o komunikační sběrnici používanou pro přenos dat z periferních zařízení, jako je klávesnice či myš, do PC či jiných zařízení. Komunikační sběrnice dostala svůj název od jedné z generací počítačových systémů vyráběných společností IBM, konkrétně IBM IBM Personal System 2 (IBMPS/2). Nyní je na ústupu před technologií USB, která je schopna připojit nespočet různorodých periferií ke koncovým zařízením.

2.1.2 Mechanické a elektrické charakteristiky

Pro připojení periférii pomocí této sběrnice se po dobu její existence používalo nespočet konektorů. Za zmínku stojí 5-pinový DIN konektor, který se používal dříve. Dále pak za zmínku stojí konektor mini-DIN, který se pro tuto sběrnici používá ve většině případů.



Obrázek 2.1: Schéma konektorů mini-DIN (PS/2 Ports ATX Albedo, Albedo[CC BY-SA 3.0], via Wikimedia Commons)

Na obrázku je vidět konektor mini-DIN. Barevně odlišené bývají konektory pro klávesnici (fialový) a pro myš (zelený). V následující tabulce se nachází zapojení pinů konektoru mini-DIN pro sběrnici PS/2:

Č. pinu	Zapojení
1	Data
2	-
3	GND
4	+5V
5	CLK
6	-

Tabulka 2.1: Zapojení pinů (Zdroj [?])

Maximální proud by měl být 275 mA a napájení v rozmezí od 4,5 V do 5,5 V.

2.1.3 Princip Komunikace

Komunikace probíhá velice jednoduše, avšak pro přenos dat od klávesnice a do klávesnice se protokol pochopitelně v některých drobných detailech liší. Při komunikaci se přenáší zároveň takt (je generován periferním zařízením[1]), vzorkování se děje během vzestupné hrany hodinového taktu, při vysílání klávesnicí a při vysílání hostitele klávesnice dochází k vzorkování v okamžiku sestupné hrany hodinové signálu. Kmitočet hodinového signálu je obvykle v rozmezí od 10 do 16,7 kHz tzn. 30-50 μs[2].

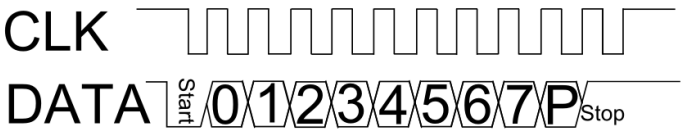
Sekvence vysílání dat probíhá způsobem ukázaným na následujícím obrázku.

Obrázek 2.2: Sekvence vysílání/přijímání dat (Zdroj: vlastní zpracování))

Na obrázku označení **P** znamená lichou paritu a 0-7 je označení bitu, který je vysílán (totéž ve všech ostatních schématech).

2.1.4 Komunikace směrem k hostiteli

Klávesnice může začít s posíláním dat, jakmile je sběrnice v nečinném stavu (tzn., když se data i hodiny nachází ve stavu vysoké napěťové úrovně, obvykle alespoň po 50 μs). Komunikovat se začíná tím způsobem, že klávesnice stáhne nejdříve datovou sběrnici na nízkou napěťovou úroveň a spustí vysílání taktovacího kmitočtu na příslušném taktovacím vodiči, vysílání na tomto vodiči tedy není permanentní, po odeslání dat je tak tento vodič v nečinném stavu[3]. Na následujícím ilustračním obrázku je průběh komunikačního procesu. Data se čtou při sestupné hraně signálu na CLK.



Obrázek 2.3: Komunikace odesílaná klávesnicí (Zdroj: vlastní zpracování)

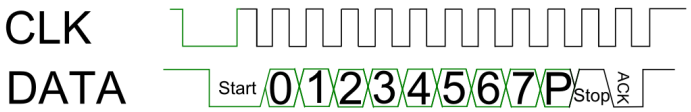
[1] Tedy může být generován klávesnicí nebo myší

[2] podle [4]

[3] tzn. nachází se ve stavu vysoké napěťové úrovně

2.1.5 Komunikace směrem od hostitele

Tato komunikace má větší prioritu, než data odesílána ve směru k hostiteli, a také se liší tím, že data jsou čteny klávesnicí při náběžné hraně signálu. Inicializace sběrnice probíhá tím způsobem, že hostitel stáhne hodinový signál na nízkou napěťovou úroveň, alespoň na 100 µs. Následně stáhne sběrnici na nízkou napěťovou úroveň a totéž provede s datovým vodičem. Po této proceduře hostitel nastaví taktovací vodič do stavu vysoké impedance a čeká na hodinový kmitočet vysílaný klávesnicí. Po příchodu prvního hodinového pulzu se data začnou vysílat podobně jako v případě 2.1.4. Na následujícím ilustračním obrázku je průběh komunikačního procesu.



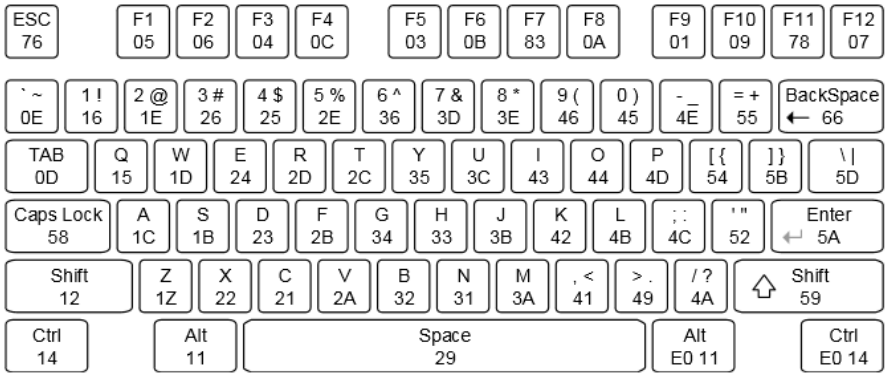
Obrázek 2.4: Komunikace odesíláná do klávesnice (zelená čára naznačuje ovládání hostitelem, Zdroj: vlastní zpracování)

2.1.6 Prezentace dat

Znaky posílané klávesnicí jsou prezentovány pomocí tzv. scan kódů. Při stisknutí klávesy se pošle kód klávesy a po puštění klávesy se pošle kód „F0“ následovaný opět kódem znaku. Existují také dvou bytové znaky, u kterých se nejprve vyšle znak „E0“ a poté, v dalším bytu, se znak specifikuje.

Po zapnutí klávesnice je nutné klávesnici inicializovat, diagnostikovat a resetovat pomocí tzv.Basic Assurance Test (BAT)(tj. hodnota 0xFF vyslána po datové sběrnici), který nastaví výchozí hodnoty do vnitřních registrů. Toto mimo jiné způsobí zapnutí a vypnutí LED na klávesnici (CAPS LOCK,SCROLL LOCK, NUM LOCK). Po vyslání kódu je nutné vyčkat, než se diagnostika dokončí, což se signalizuje tím způsobem, že klávesnice vyšle kód 0xAA[4].

Na následujícím obrázku lze shlédnout scan kódy kláves.



Obrázek 2.5: Scan kódy klávesnice (Obrázek Keyboard scan codes [2])

[4]Tato hodnota je opět vyslána pomocí datové sběrnice

2.2 Universal Asynchronous reciever and transmitter (UART)

Tato kapitola se zaobírá fungování sériové komunikace typu RS-232, jelikož obvod UART je určen právě k přijímání a vysílání takovéto komunikace v asynchronním režimu. Při zpracovávání bylo použito následujících zdrojů: [7], [8] a [9].

2.2.1 Obecné informace

Jeden z nejznámějších standardů pro sériovou komunikaci je zřejmě RS-232 potažmo standard EIA/TIA-232E (velmi podobné standardy jsou také V.24 a V.28), který na starší standard navazuje. Tato rozhraní prošla velmi dlouhým vývojem a několika standardizacemi. Pro tato rozhraní bylo také používáno nespočetné množství konektorů, mezi které patří například konektory typu D-Subminiature (D-SUB) s 9 (DE9) nebo 25 (DB25) piny. Toto rozhraní může mít v nejjednodušším případě 3 vodiče(jeden pro vysílání dat, jeden pro přijímání dat a poslední zemnící vodič) nebo může mít až 25 vodičů, kde mají vodiče různé vlastnosti určených většinou k řízení komunikace.

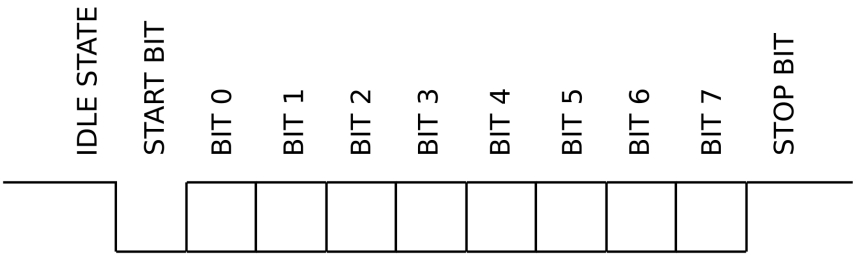
2.2.2 Princip komunikace

Pro správnou funkci rozhraní je nutné, aby si obě strany, které chtějí vysílat a přijímat data, domluvily několik parametrů.

Prvním důležitým parametrem je domluvení modulačních rychlostí, kterou se budou data přenášet. Je několik rychlostí,které jsou běžně používány. Zde je výpis některých používaných rychlostí:

- 9600 Bd
- 19200 Bd
- 38400 Bd
- 57600 Bd
- 115200 Bd

Existuje také několik možností jak je možné vysílat data. Vždy se začíná jedním Start bitem poté následuje blok užitečných dat, který se skládá obvykle z 5-8 bitů. Následně se vysílá paritní bit, ten může být tvořen pomocí liché nebo sudé parity. Ty se tvoří tak, že pomocí paritního bitu se přidá buď logická jednička nebo logická nula, tak aby výsledný počet jedniček ve vysílaném slově byl lichý nebo respektive sudý. Tato metoda detekce chyb je schopná objevit chybu pouze v jednom bitu. Je samozřejmě také možné aby byl paritní bit nastaven trvale v logické jedničce (Mark), logické nule (Space) nebo nemusí být nastaven vůbec a tím pádem se ani tento paritní bit nevysílá. A nakonec je přidán jeden nebo dva stop bity. Je nutné, aby obě strany posílaly stejné tvary těchto posloupností. Na následujícím obrázku je ukázáno schéma pro vysílání s jedním start bitem, 8 datovými bity a jedním stop bitem, ale bez použití paritního bitu (8N1).



Obrázek 2.6: Časování pro 8N1 (Zdroj: vlastní zpracování, založeno na [9].

2.2.3 Prezentace znaků

Ve většině případů budou bity představovat znaky kódované pomocí American Standard Code for Information Interchange (ASCII). Následuje tabulka obsahující seznam těchto znaků.

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Obrázek 2.7: ASCII tabulka (ASCII Code Chart, Anomie [Public domain], via Wikimedia Commons)

2.3 Video Graphics Array (VGA)

V této kapitole je popsáno fungování Video Graphics Array (VGA) určené pro vykreslování obrazu na monitorech. Při vypracovávání této části práce byly použity informace z následujících zdrojů:[10], [2], [11], [12], [13] a [14].

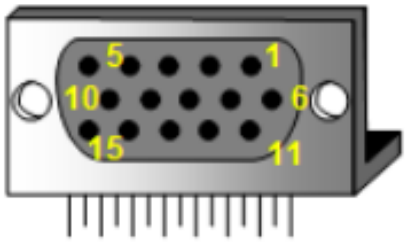
2.3.1 Obecné informace

Rozhraní VGA pro zobrazování informací na monitorech je stejně jako sběrnice PS/2 poměrně staré. Poprvé se objevila také u IBM PS/2, v roce 1987. Stejně jako sběrnice pro komunikaci s myší a klávesnicí PS/2 je tato sběrnice na ústupu před modernějšími rozhraními, jako je například Digital Visual Interface (DVI),High-Definition Multimedia Interface (HDMI), DisplayPort a jinými. Oproti ostatním uvedeným rozhraním vysílá rozhraní VGA analogový signál. Nicméně stále ještě mnoho monitorů je tímto rozhraním opatřeno. Od své publikace bylo toto rozhraní několikrát rozšířeno, původní

standard zahrnoval rozlišení 640x480 s 16 barvami nebo monochromatickými barvami či menší rozlišení 320x200 ovšem s 256 barvami. V nynější době jsou standardy k tomuto rozhraní a jeho rozšířením organizace Video Electronics Standards Association (VESA), původní standard patří společnosti IBM. Zařízení není navrženo pro vypojování konektoru za běhu zařízení, nicméně toto není z funkčního hlediska velký problém jelikož se během jednoho nebo několika málo snímku stačí obraz znovu synchronizovat a tudíž další snímky jsou již správně zobrazovány.

2.3.2 Mechanické a elektrické charakteristiky

V nejčastějším případě se pro rozhraní VGA používá konektor D-Subminiature (D-SUB), který dostal svoje jméno od tvaru mírně zkoseného čtverce, který připomíná písmeno D. Konkrétně konektor DE-15, kde E v tomto výrazu značí velikost konektoru a číslo 15 počet pinů tohoto konektoru. Následuje ilustrace upřesňující podobu zmíněného konektoru (zásuvku).



Obrázek 2.8: Rozložení pinů VGA rozhraní na přípravku Digilent Nexys 4 (Figure 11 Nexys 4 VGA Interface [2])

V následující tabulce se nachází shrnutí, jaký účel plní konkrétní piny z předchozího obrázku.

PIN 5: 1	GND	Monitor_ID(2)	BLUE	GREEN	RED
PIN 10: 6	S GND	Key	B GND	G GND	R GND
PIN 15:11	Monitor_ID(3)	VSYNC	HSYNC	Monitor_ID(1)	Monitor_ID(0)

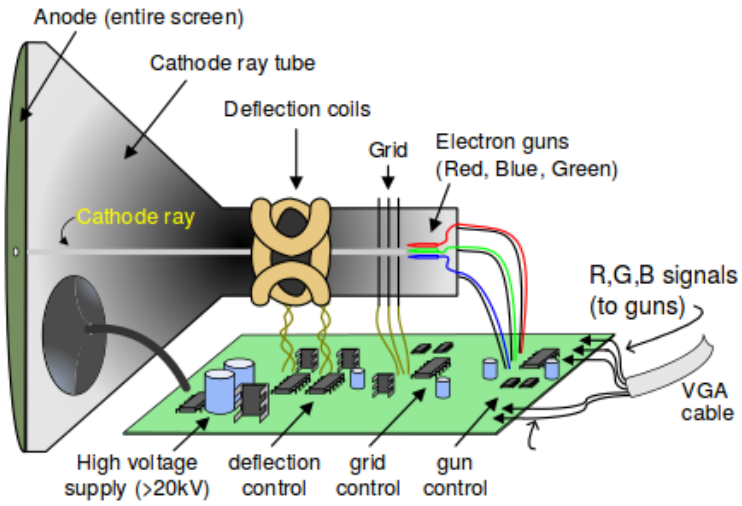
Tabulka 2.2: Zapojení pinů (Zdroj [15])

Pomocí pinů 1-3 (Red, Green, Blue) se mění a udává barva konkrétním pixelům, piny Monitor_ID jsou poměrně nepodstatné a v novějších standardech jsou tyto piny nahrazeny komunikací I^2C nebo-li DDC. Ta je určena pro domluvení podporovaných rozlišení a barev mezi monitorem a zařízením, které chce zobrazovat informace.Piny 5-10 jsou určené pro zemnící vodiče. Pin Key byl původně určen pro tlačítko, nicméně pozdější standardy VGA jej předefinovaly na + 5 V napájení. Velmi důležité jsou piny Horizontal Synchronization (HSYNC) a Vertical Synchronization (VSYNC), jelikož pomocí těchto vodičů se určuje doba trvání řádků a respektive také celého jednoho snímku, více viz. další kapitola

Co se týče elektrických charakteristik, tak vodiče RGB se chovají jako vedení s charakteristickou impedancí 75 Ω a napětí pro logickou nulu je 0 V a pro logickou jedničku je 0,7 V. Pro signály Vertical Synchronization a Vertical Synchronization je typické napětí 5 V nebo 3,3 V (oboje pro stav logické jedničky).Více viz. [15] a [11]

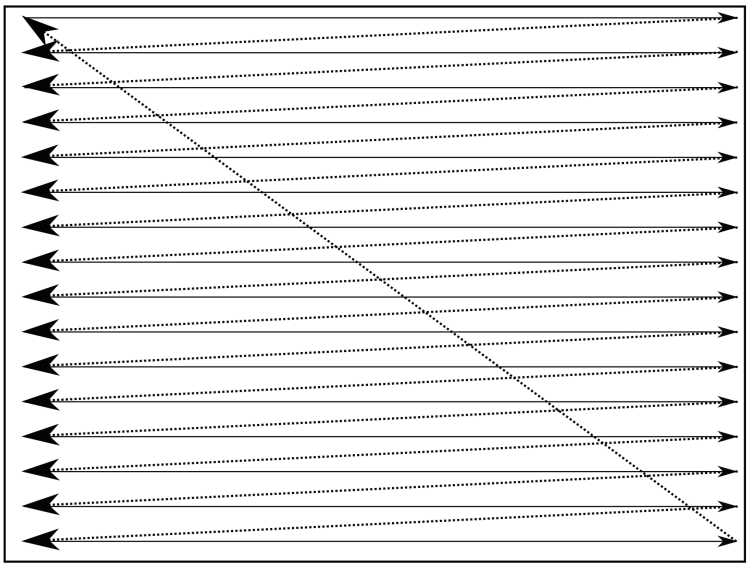
2.3.3 Zobrazování informací

Rozhraní VGA bylo původně navrženo pro monitory typu Cathode Ray Tube (CRT). Časování tohoto rozhraní tomu tedy tak trochu odpovídá. Následující obrázek ilustruje situaci u monitoru typu CRT.



Obrázek 2.9: Funkce CRT monitoru (Figure 12. Color CRT display [2])

Elektronová děla (electron guns) vyzařují paprsky elektronů trubicí přes masky na přední část obrazovky, zde jsou anodou přitahovány a pohlcovány fosforescenční vrstvou, která se poté rozzáří. Pomocí napětí na přívodech RGB lze nastavit skládáním těchto tří barev téměř jakoukoliv možnou barvu. Mřížka pak pomáhá vytvářet užší svazky proudu elektronů. Takto vytvořený paprsek je vychylován vychylovacími cívkami umístěnými za mřížkou. Proud elektronů je vždy vychylován z horního levého rohu a postupně po řádcích do pravého dolního rohu, odkud se paprsek vrací znovu do levého horního rohu (viz. následující ilustrační obrázek).



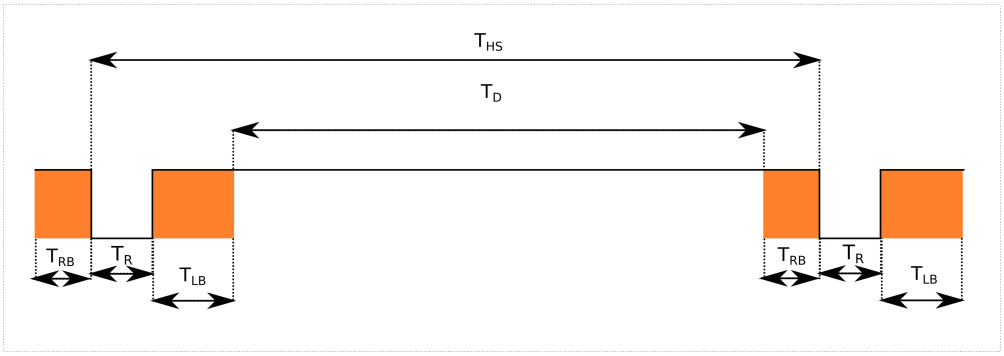
Obrázek 2.10: Trasa paprsku elektronu u CRT monitorů (Vlastní zpracování, založeno na [14])

Paprsek tedy vždy začíná v levém horním rohu, odkud putuje na konec řádku, poté se paprsek vrátí na začátek dalšího řádku, dokud paprsek nedoputuje nakonec obrazovky, odkud se vrátí opět na první řádek. Toto je řízeno pomocí signálů Horizontal Synchronization a Vertical Synchronization (více viz. další kapitola).

Co se týče novějších monitorů založených na technologii Liquid Crystal Display (LCD), tak ty používají stejný princip časování který bude nastíněn dále a který vyplývá z funkčnosti monitorů založených právě na technologii CRT.

2.3.4 Časování signálů

Pro synchronizaci obrazu u rozhraní Video Graphics Array (VGA) jsou velmi důležité synchronizační signály Horizontal Synchronization a Vertical Synchronization.Pomocí těchto signálů jsou ovládány u monitorů CRT vychylovací cívky a to tím způsobem, že signály jsou tvarovány na pilový průběhů a přechod na další řádek se děje při sestupné hraně vytvořeného pilové průběhu. Na následujícím obrázku je znázorněn průběh signálu Horizontal Synchronization.



Obrázek 2.11: Průběh signálu Horizontal Synchronization (Vlastní zpracování, založeno na [14],[2])

Průběh synchronizačního pulzu Horizontal Synchronization začíná vždy fází tzv. „Retrace"(nebo také zatemňovací pulz), kdy se paprsek vrací na začátek řádku. Zatemňovací pulz představuje logickou nulu synchronizačního signálu. V další fázi se paprsek nachází za levým „okrajem" obrazovky a nezobrazuje se na žádné viditelné ploše, tato fáze se nazývá „front porch". Poté přichází doba kdy paprsek ozařuje viditelnou plochu displeje, toto je na obrázku znázorněno dobou T_D. Mezitím paprsek postupuje dále po řádku obrazovky, než se dostane opět do místa, které opět není viditelné obrazovce, toto je za pravým „okrajem obrazovky" a nazývá se „back porch". Nakonec opět přichází fáze „Retrace" kdy se paprsek posouvá na začátek dalšího řádku. Takto se chová monitor při synchronizačním signálu Horizontal Synchronization.

Průběh vertikálního synchronizačního pulzu Vertical Synchronization je ve své podstatě stejný, liší se pouze dobou jednotlivých částí synchronizačního pulzu. Zatímco během jednoho pulzu Horizontal Synchronization dojde k vykreslení jednoho řádku, během doby jednoho pulzu Vertical Synchronization dojde k vykreslení všech řádků a tudíž celé obrazovky. Opět tedy nastává fáze „Retrace", za čímž následuje „Top porch" místo „front porch", poté přichází doba řádků, které jsou ve viditelné části obrazovky, následuje „bottom porch" místo „back porch" a opět „Retrace".

Časování těchto pulzů pro rozlišení 640x480 při době periody trvání jednoho pixelu 40 ns (25 MHz) je podle následující tabulky. Takovéto časování signálu bylo použito v praktické části práce, pro jiné rozlišení a novější standardy viz [15].

Časový okamžik	VSYNC	HSYNC
Synchronizační pulz(T_{HS})	16,7 ms	32 µs
Zobrazení(T_D)	15,36 ms	25,6 µs
Zatemňovací pulz(T_R)	64 µs	3,84 µs
Levý/Horní okraj(T_{RB})	320 µs	640 ns
Pravý/Dolní okraj(T_{LB})	928 µs	1,92 µs

Tabulka 2.3: Tabulka časování pro VGA (Zdroj [2])

3 Programové vybavení

V této kapitole se nachází popis software, který jsem při praktické části využíval.

3.1 Xilinx VIVADO

Xilinx Vivado™ je novější prostředí pro vývoj na programovatelných platformách FPGA vyráběných společností Xilinx. Oproti starší verzi prostředí Xilinx ISE by toto prostředí mělo být efektivnější a rychlejší co se týče použitých nástrojů v tomto prostředí. Dále jsou všechny nástroje integrované v prostředí, oproti tomu Xilinx ISE spoléhal v některých případech na nástroje externí. Podporuje také vysokoúrovňovou syntézu HLS, která umožňuje převod kódů C/C++ na IP-jádra která jsou vhodná pro implementaci na programovatelnou logiku. Mimo jiné také umožňuje „debugování" hardwaru pomocí vestavěných nástrojů.

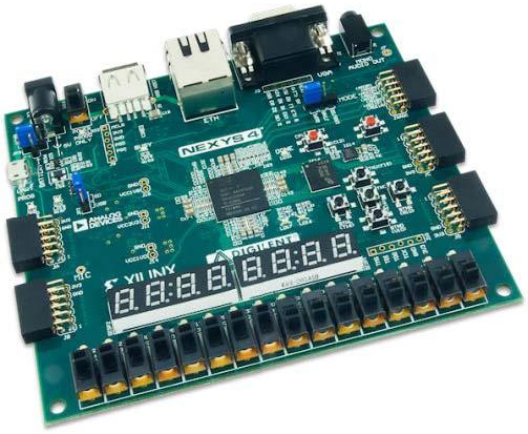
Nevýhodou tohoto softwarového nástroje je to, že společnost Xilinx zabezpečuje podporu pouze novějších typů FPGA, které umí tento nástroj použít. Pro starší typy FPGA tedy není možné vyvíjet návrhy a následně tyto návrhy do zařízení nahrát a naprogramovat. Je ale možné získat knihovny s podporou těchto starších zařízení aby bylo možné vyvíjet a syntetizovat návrhy. To nemění ovšem nic na tom že nepodporované FPGA nelze naprogramovat pomocí Xilinx Vivado. Přípravek Digilent Nexys 4 obsahuje FPGA Artix-7™, který již podporovaný je. Přípravky Spartan-3E Starter Board a Digilent Nexys 3 jsou bohužel osazeny FPGA Spartan-6 a Spartan-3E, které podporovány nejsou.

4 Technické vybavení

V této kapitole se nachází popis zařízení, která jsou využita v praktické části práce.

4.1 Digilent Nexys 4

Digilent Nexys 4 je vývojová platforma založená na integrovaném obvodu Artix-7™ Field Programmable Gate Array (FPGA) od firmy Xilinx. Obsahuje několik konektorů pro různé účely: čtyři 12-pinové Pmod™ konektory, jeden Xilinx Analog to Digital Converter (XADC) Pmod™ konektor, VGA port, 10/100 Ethernet, několik USB portů určených k připojení zařízení nebo k programování přípravku a také slot na microSD kartu. Mezi USB porty je také port USB Human Interaction Device (HID), který umožňuje připojení USB klávesnice či myši. Možností programování je několik, nejsnazší způsob programování je přes Micro A-B USB pomocí programu Xilinx Vivado. Dále přípravek obsahuje pět tlačítek, 16 přepínačů, 16 Light Emiting Diode (LED), dvakrát čtyř-číselný sedmi-segmentový displej, Pulse Width Modulation (PWM) audio výstup, dvě trojbarevné LED, Pulse Density Modulation (PDM) mikrofon, trojosý akcelerometr ADXL362 a teplotní čidlo ADT7420. Více v datasheetu k tomuto zařízení [2].



Obrázek 4.1: Náhled na přípravek Digilent Nexys 4 [2])

4.2 Digilent Nexys 3

Digilent Nexys 3 je vývojová deska, jejíž hlavní částí je Field Programmable Gate Array (FPGA) Spartan-6 Přípravek je také osazen těmito konektory: 8- bitový VGA konektor, 10/100 Ethernet (RJ-45), pro připojení rozšiřujících desek je zde konektor VHDC či víceúčelové Pmod™ konektory které mají každý 12 vývodů. Dále je přípravek osazen 3 USB porty, jeden je určen pro programování přípravku, druhý pro USB - HID použitelný například pro klávesnici nebo myš a poslední je určen pro UART, čili pro komunikaci založenou na RS-232 přenášenou přes rozhraní USB. Dále je přípravek osazen klasickými periferiemi jako jsou tlačítka (5x), přepínače(8x), LED a čtyř-číselný číselný sedmi-segmentový displej. Více viz. datasheet k tomuto přípravku [16].

4.3 Spartan 3E Starter Board

Další přípravek od společnosti Digilent, Spartan-3E Starter Board je osazen Field Programmable Gate Array (FPGA) Spartan-3E mimo jiné je zde i XC2C64A CoolRunner-II Complex Programmable Logic Device (CPLD). Tento kit je opatřen několika konektory: D-SUB DE-9 pro komunikaci založenou na RS-232, který je zde 2 krát a to pro roli DCE a DTE, dále je zde port VGA, 100-pinový port FX2, 2 6-pinové Pmod™ konektory a také PS/2 port. Přípravek je také osazen zástrčkou pro USB typu B určenou k programování celého přípravku. Za zmínku stojí též klasické periférie jako jsou přepínače (4x), LED (8x), tlačítka (4x), rotační tlačítko a znakové LCD.

5 Implementace řízení

V následujícím textu jsou popsány jednotlivé části, kterými se zabývá praktická část diplomové práce. Část týkající se ovládání klávesnice PS/2 je rozšířená o měření frekvence oscilátoru, dodávající takt této sběrnici. Měření je pouze doplňkem a není podstatnou částí diplomové práce. Více viz. část odpovídající řízení klávesnice PS/2 5.1 a podkapitola zabývající se měřením frekvence 5.1.7.

Pro implementaci jsem se nakonec rozhodl použít přípravek Digilent Nexys 4, jelikož je tento přípravek osazen novějším typem FPGA, a to konkrétně Artix-7™. Velkou výhodou je že pro návrh je možné používat vývojových nástrojů Xilinx Vivado[5]. Nevýhodou ovšem je, že oproti přípravku Spartan 3E Starter board, neobsahuje přípravek Digilent Nexys 4 konektro mini-DIN pro sběrnici PS/2. Toto je nahrazeno emulací portu PS/2 klávesnice připojené do portu USB. Více viz. odpovídající kapitola 5.1.

5.1 PS/2 klávesnice

V této podkapitole, je popsáno navržené řízení komunikace na sběrnici PS/2. Dále jsou zde popsány výsledky simulací a měření frekvence.

Velkým problémem, který bylo nutné řešit, je absence konektoru DIN pro sběrnici PS/2 na přípravku Digilent Nexys 4. Na přípravku Nexys 4 se ovšem nachází mikrořadič společnosti Microchip, a to konkrétně mikrořadič PIC24FJ128. Tento mikrořadič slouží k programování FPGA, po naprogramování se však přepne do režimu, v němž emuluje roli zařízení na sběrnici PS/2. Lze tak použít logiku řízení pro zařízení PS/2, ačkoliv ve skutečnosti je připojeno zařízení s rozhraním USB. Mikrořadič dokáže emulovat pouze jedno zařízení, tzn. na sběrnici může být v jeden čas připojena buď USB klávesnice, nebo USB myš. Funkce USB rozbočovače není podporována.

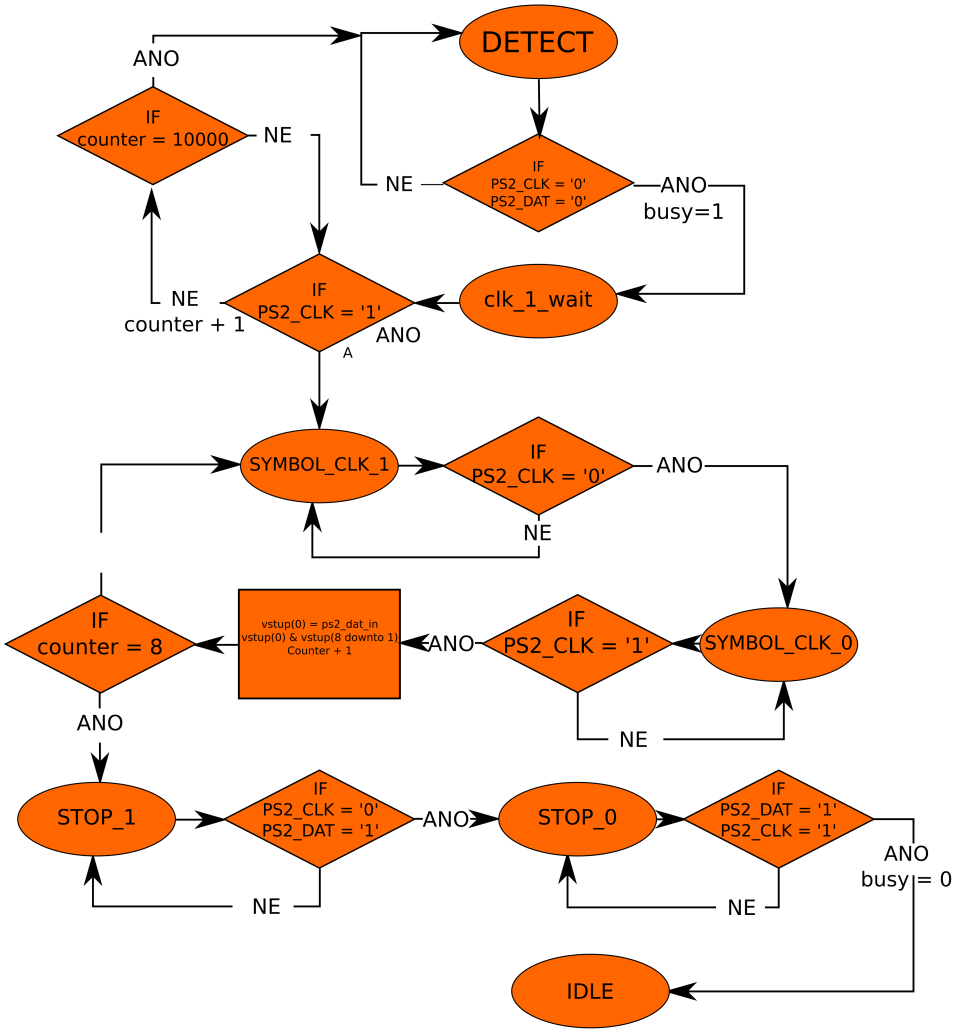
5.1.1 Návrh řízení sběrnice PS/2

V zařízení se nachází několik stavových automatů. Tři stavové automaty jsou použity pro řízení, vysílání a příjem z rozhraní PS/2. Dále se zde nachází automat řídící vysílání dat vyšším komponentám a také automat řídící překlad přijatých znaků z klávesnice do hodnot ASCII.

Automaty pro vysílání a příjem symbolů po sběrnici PS/2 jsou v podstatě přepínané automatem, který nejdříve vyčkává po 80 ms a pak vyšle signál BAT pomocí automatu pro vysílání. Po této proceduře se automat pro vysílání dostane do neaktivního režimu a automat pro řízení příjmu se zapne. Jestliže se nepřijme hodnotu 0xAA do 80 ms, nebo se přijme jiný signál, celá procedura se opakuje. V případě, že je BAT signál potvrzen pomocí 0xAA, nastává příjem. A to tak, že automat pro vysílání je v nečinném stavu a automat pro příjem je vždy po přijetí Scan Kódu ve stavu čekání na Start bit, více viz dále.

Na následujícím diagramu je zobrazen automat řídící komunikaci směrem od zařízení. Jeho slovní popis se nachází pod tímto obrázkem.

[5] viz kapitola 3.1



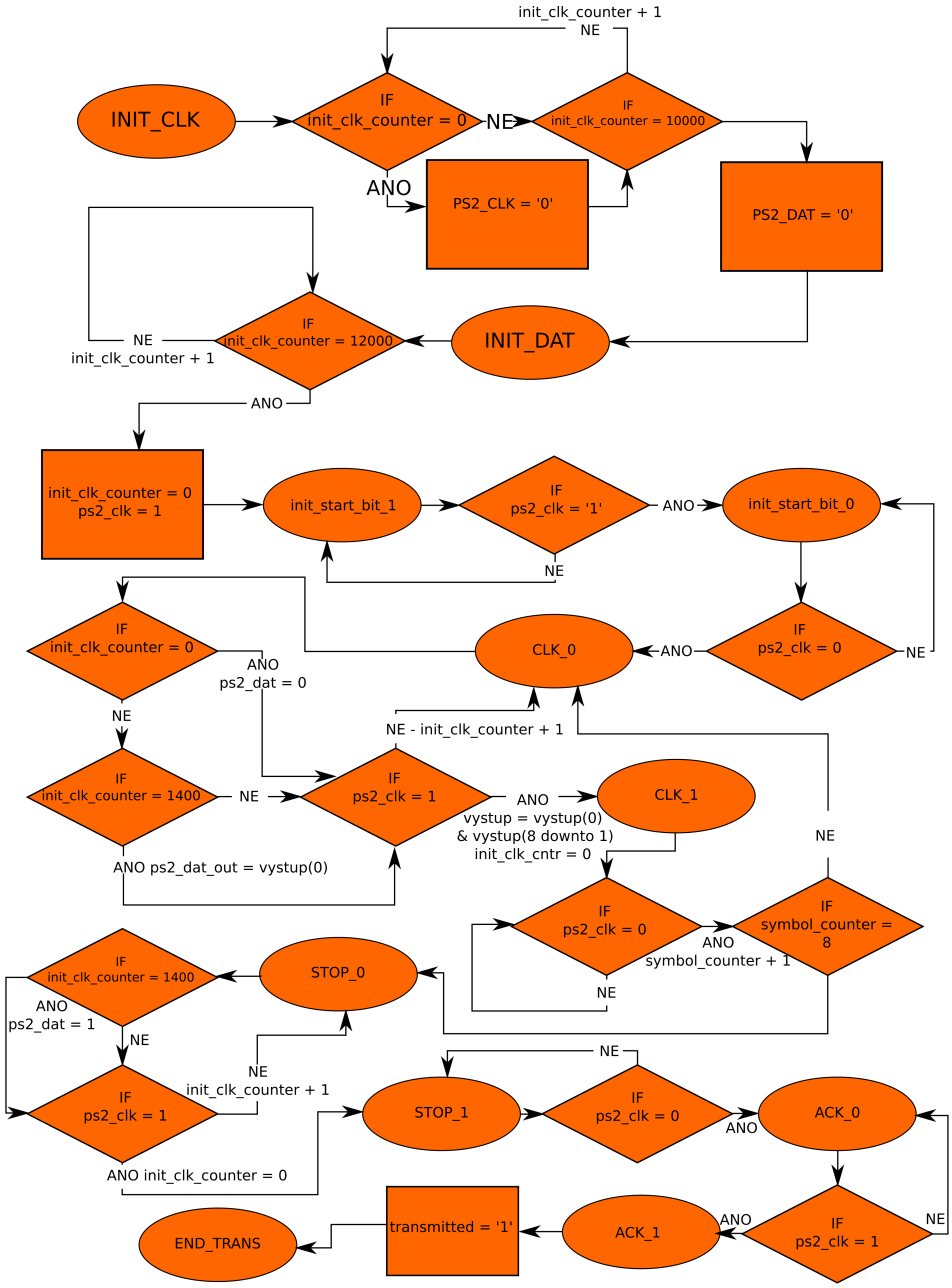
Obrázek 5.1: Stavový automat určený pro příjem dat (Zdroj vlastní)

Výchozí stav automatu je stav DETECT, tedy pokud je signál read = 0, v opačném případě se automat nijak nemění, signál write závisí na řídícím automatu, který je popsán níže. Pokud je tedy automat aktivní a stane se, že oba vodiče sběrnice jsou v logické nule, automat se přepne do stavu clk_1_wait. Oba vodiče v logické nule po stavu nečinnosti sběrnice indikují, že právě dochází k vysílání znaku protější stranou. Ve stavu clk_1_wait automat čeká na stav logické jedničky na taktovacím vodiči. Jestliže se tak nestane do 100 µs, dostane se automat opět do stavu DETECT. V obvyklém případě bude automat pokračovat do stavu symbol_clk_1, kde počká než se taktovací vodič sběrnice PS/2 dostane do stavu logické nuly. Poté se přepne do stavu symbol_clk_0, který naopak čeká na hodnotu logické jedničky. Po splnění podmínky si automat přečte hodnotu z datového vodiče sběrnice PS/2 a inkrementuje čítač. V případě, že již došlo k vyslání všech užitečných dat čítač by měl být na hodnotě 8[6], a v tom případě dojde k přepnutí do stavu STOP_1. V tomto stavu automat vyčkává do doby, než se hodnota taktovacího vodiče dostane na logickou nulu. Poté se překlopí

[6]tzn. 0-7 jsou datové bity a 8 bit je paritní

do stavu STOP_0, ve kterém opět čeká tentokrát na logickou jedničku. Následně se automat překlopí do stavu IDLE, kde vyčká než mu řídící automat dá opět možnost dostat se do stavu DETECT. Ve stavu DETECT se hodnota signálu busy nastaví na 1, a naopak ve stavu IDLE se nastaví na 0. Toto je důležité pro další automat, který řídí konverzi ze scan kódů do ASCII.

Následuje popis automatu, který se zabývá vysíláním dat. Tento automat je nutný z toho důvodu, že před začátkem přijímání znaků se musí vyslat tzv. signál BAT.

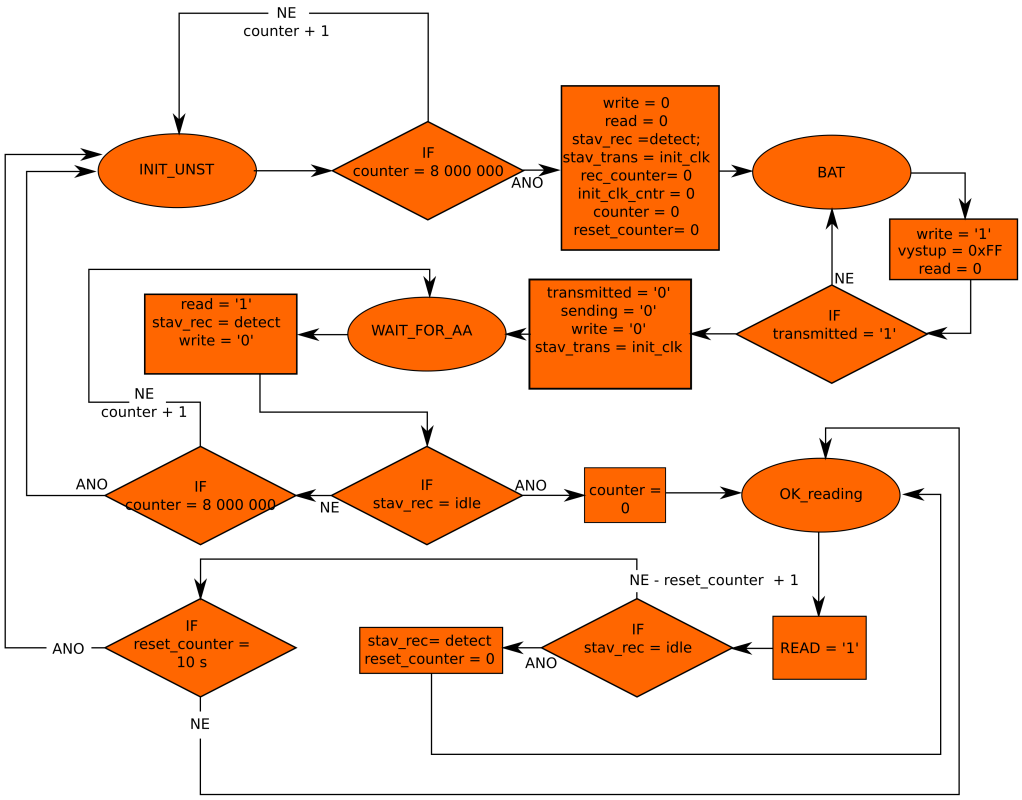


Obrázek 5.2: Stavový automat určený pro vysílání dat (Zdroj vlastní)

Tento automat se nachází na začátku vždy ve stavu INIT_CLK, podobně jako v případě automatu pro příjem je tento automat aktivní, když je signál write roven 1.

Pokud automat právě vstoupil do tohoto stavu, nastaví se na taktovacím vodiči sběrnice stav logické nuly. Poté se spustí čítač, který po 100 µs stáhne do stavu logické jedničky také datový vodič, a také nastaví stav INIT_DAT. V tomto stavu čítač pokračuje v čítaní na 20 µs. Automat poté vynuluje čítač, vrátí ovládání taktovacího vodiče zařízení, tím že nastaví logickou jedničku (mimo automat dojde na nastavení stavu vysoké impedance). Následně se automat přepne do stavu init_start_bit_1, ve kterém automat čeká na logickou jedničku na taktovacím vodiči sběrnice PS/2. Za čímž následuje přepnutí do stavu init_start_bit_1, ve kterém naopak čeká na logickou nulu na taktovacím vodiči. Následuje stav CLK_0, vždy po příchodu do tohoto stavu se vynuluje datový vodič sběrnice PS/2. Po uplynutí 14 µs se nastaví hodnota příslušející danému bitovému místu, které má být vysláno. Poté se čeká, než taktovací vodič přejde do stavu logické jedničky. Následuje přepnutí do stavu CLK_1, čímž se načte nová hodnota, která má být v následující iteraci vyslána a vynuluje se čítač. Automat opět čeká na stav logické nuly na taktovacím vodiči poté inkrementuje čítač bitového místa, pokud je tento čítač roven 8, ukončí se vysílání užitečných bitů a přejde se k vyslání STOP bitů. V opačném případě se automat přepne opět do stavu CLK_0 a vysílání pokračuje. Po přechodu do stavu STOP_0 automat počká 14 µs, poté nastaví logickou jedničku na datový vodič, což symbolizuje stop bit. Po přechodu logické úrovně na taktovacím vodiči dojde k nastavení stavu STOP_1, ve kterém se pouze čeká na logickou nulu na taktovacím vodiči. Následně ve stavu ACK_0 automat opět čeká na změnu logické úrovně na taktovacím vodiči na logickou jedničku, po čemž následuje stav ACK_1 a ihned poté stav END_TRANS a dojde k nastavení signálu transmitted na logickou jedničku, čímž se signalizuje řídícímu automatu, že vysílání bylo dokončeno. Automat se do stavu INIT_CLK nastaví pomocí řídícího automatu.

Na následujícím diagramu je poslední část nutná ke komunikaci mezi klavesnicí a přípravkem. Jedná se o třetí stavový automat, který řídí chod obou předešlých. Následuje popis tohoto automatu.



Obrázek 5.3: Stavový automat pro řízení komunikace na sběrnici PS/2 (Zdroj vlastní)

Výchozí stav je pro tento automat stav INIT_UNST, ve kterém vyčkává na počátku naprogramování přípravku po 80 ms. Po této době se automat přepne do stavu BAT, ve kterém dojde k vyslání BAT signálu klávesnici. Před přechodem do stavu BAT se z důvodu, že je stav INIT_UNST vychozí (při resetování automatu), vynulují některé důležité indikátory a čítače. Poté se ve stavu BAT nastaví hodnoty write = 1 a read = 0 čímž se aktivuje automat určený odesílání dat klávesnici. Dále se nastaví hodnota 0xFF, která se vyšle po sběrnici PS/2 za pomocí vysílacího automatu. Hodnota 0xFF symbolizuje signál BAT. Když vysílací automat ukončí vysílání, řídící automat resetuje vysílací automat a přepne se do stavu WAIT_FOR_AA, ve kterém povolí příjem pomocí nastavení read = 1. Následně se čeká na to, zda-li se přijímací automat dostane do nečinného stavu[7], zdali toto nenastane do 80 ms, tak se celý automat resetuje a nastaví se do výchozího stavu, tedy do stavu INIT_UNST. V případě, že zařízení zvládne včas odpovědět, automat nastaví stav OK_reading. V tomto stavu automat nastavuje neustálé čtení, pokud ale žádný znak nepřijde během 10 sekund, celý automat se resetuje a nastaví do výchozího stavu.

5.1.2 Návrh převodníku Scan kód - ASCII

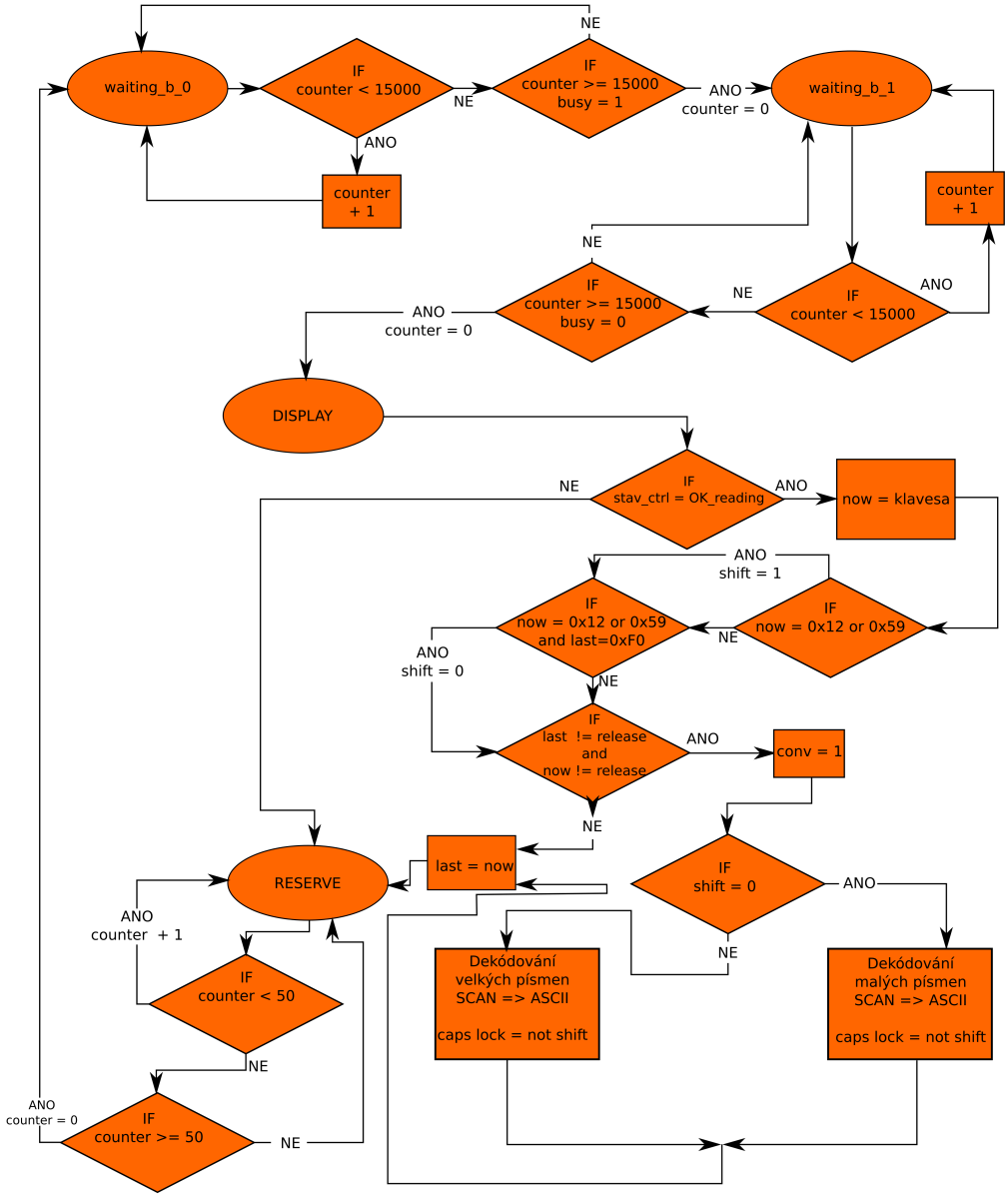
Převod Scan Kódu se děje vždy po přijetí symbolu. Převádí se vždy stisk klávesy, automat ignoruje puštění klávesy. Výjimkou je stisk klávesy shift, kdy puštění klávesy

[7] tzn. že dojde k přijetí dat pomocí tohoto automatu a sběrnice PS/2

rozhoduje o tom, zda bude velké či malé písmeno. Překládají se pouze Scan kódy alfanumerických znaků[8]. Také se identifikuje stisk kláves Caps Lock, Shift, Enter a Backspace, ostatní nejsou nikterak ošetřeny a při jejich stisku nedojde k vyhodnocení. Toto by šlo jednoduše implementovat přidáním řádků v dekodéru, ale jelikož je cílem diplomové práce demonstrační úloha a ne plnohodnotný produkt, není tato funkčnost nutná. Dalším důvodem pro neimplementování je fakt, že v dalších komponentách nejsou ostatní klávesy nikterak potřebné nebo implementované.

Následuje popis automatu pomocí diagramu (viz 5.4 obrázek a slovní vysvětlení. Je popsán pouze automat pro přeposílání ASCII znaků pro vyhodnocení komponentami souvisejícími s VGA.

[8] tzn. nejsou implementovány znaky jako například !:?_=+ a jiné



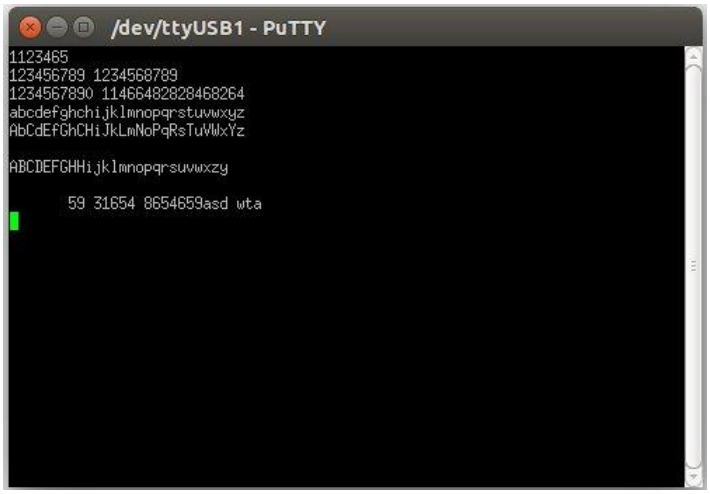
Obrázek 5.4: Stavový automat pro převod reprezentace znaků ze Scan kódu na ASCII (Zdroj vlastní)

Při zapnutí se automat nachází ve stavu waiting_b_0, kdy automat čeká na signál busy od přijímacího automatu. Signál busy indikuje, že automat započal přijímat znak a čítač je v tomto stavu z důvodu, aby nedošlo k chybné indikaci tohoto stavu. Zda dochází k přijímání znaků dovolí čítač v automatu až po 15 µs. Poté dojde k překlopení do stavu waiting_b_0, kde probíhá podobný postup, s tím, že automat čeká, než přijímací automat dokončí příjem. Dochází k překlopení do stavu DISPLAY, kde se nejprve kontroluje, zda dochází k přijmu znaků, nebo dochází k výměně Scan kódů nutných k započetí komunikace mezi zařízeními. Pokud přišel stisk klávesy, dojde k zapsání této klávesy do paměti automatu a k analýze tohoto znaku. Pokud se jednalo o znaky pravý či levý shift dojde k nastavení proměnné shift na jedničku, naopak dojde-li k puštění těchto kláves nastaví se proměnná shift na nulu. Tato proměnná později

slouží k určení, zda se mají použít velká či malá písmena. Pokud jako poslední příchozí Scan kód přijde 0xF0 (indikátor puštění klávesy), dojde k přeskočení vyslání, uloží se hodnota nynějšího aktuálního Scan kódu a dojde se ke stavu RESERVE (viz. dále). Pokud tedy poslední znak není 0xF0, dojde k rozhodnutí, zda se má vyslat malé či velké písmeno. Na číslice tato událost nemá vliv.V případě, že došlo ke stisku klávesy Caps Lock, dojde k znegování hodnoty shift. Po dekódování dojde také k uložení aktuální hodnoty Scan kódu a přejde se do stavu RESERVE, který slouží pouze k prodloužení doby, než se automat dostane opět do stavu waiting_b_0.

5.1.3 Reprezentace přijatých znaků

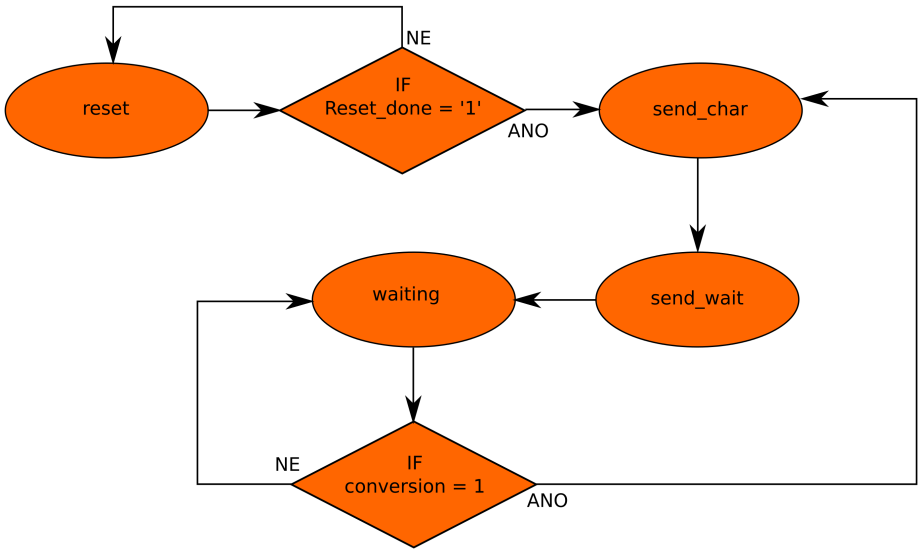
Při prvotním návrhu a testování kdy byl testován pouze návrh řízení sběrnice s následnou kontrolou korektního chování. Vyslání probíhá v tomto případě tak, že automat vždy vyčká na konverzi do ASCII, a poté vyšle přes pin RXD do obvodu UART, jenž zajišťuje přenos této komunikace přes rozhraní USB. Tento automat nebere ohledy na většinu symbolů a prostě je vyšle tak jak přijdou. Výjimka nastává po přijetí klávesy Enter a Backspace, kdy je nutné vyslat dva ASCII znaky.



Obrázek 5.5: Výstup Sériového rozhraní (Zdroj vlastní)

Toto přeposílání bylo použito pouze pro kontrolu, ve výsledném zapojení je tato funkčnost odebrána a ASCII znaky jsou posílány do komponenty na vyhodnocování znaků pro VGA (5.3), konkrétně do bloku pixel_logic.

Dále jsou popsány pouze automaty pro přeposílání ASCII znaků pro vyhodnocení komponentami souvisejícími s VGA. Automat je popsán pod svým diagramem.



Obrázek 5.6: Stavový automat řídící odesílání ASCII znaků nadřazeným komponentám (Zdroj vlastní)

Po spuštění je automat ve stavu reset, ve kterém přečkává prvních 80 milisekund, kdy nedochází k přijmu znaků. Po uplynutí teto doby, tzn. až externí čítač dopočítá 8 000 000 pulzů 100 MHz kmitočtu, dojde k přepnutí do stavu send_char, kdy se vyšle znak. Při zapnutí se vyšle prázdný znak. Po překlopení stavu waiting se čeká, než dojde k přeložení znaku reprezentovaného scan kódem na znak ASCII automatem pro převod, který je popsán výše. Po přeložení dojde k vyslání tohoto znaku během doby, kdy se nachází automat ve stavu send_char. Mezilehlý stav send_wait je zde pouze proto, aby se docílilo zpoždění mezi čekáním a vysíláním, a to z důvodu, aby se hodnotu signálu conversion podařilo včas změnit a znak nebyl vyslán vícekrát.

5.1.4 Doplňková funkcionalita - Čítač pro měření frekvence hodinového taktu na sběrnici PS/2

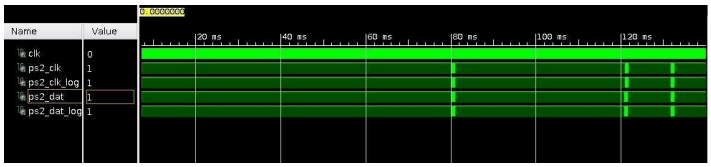
Součástí z prvotního návrhu je také čítač, který čítá pulzy s frekvencí 100MHz vestavěného oscilátoru přípravku během dob, kdy je signál PS/2 clk ve stavu nízké napěťové úrovně. Toto je využito pro měření frekvence signálu PS/2 clk. Hodnota tohoto čítače je vypisována na integrovaný 7-segmentový displej.

Tato funkcionalita je z finálního zapojení odebrána, jelikož se jedná pouze o měření v rámci sběrnice PS/2, a pro funkčnost celkového zapojení není nutná a navíc se jedná o indikátor zda navržené řízení funguje správně.

5.1.5 Simulace navrženého řešení

Simulace byla prováděna pomocí simulačního software v programu Xilinx Vivado viz. 3.1. Simulace je provedena pro případ vyslání signálu BAT, následovného příjmu potvrzujícího Scan kódu 0xAA a následného přijetí znaku po stisku klávesy C (0x48).

Všechny obrázky zobrazující simulaci ukazují čtveřici signálů. Ps2_clk, ps2_dat zobrazují poměry na pinech připojených k rozhraní PS/2.Ps2_clk_log, ps2_dat_log zobrazují tytéž poměry, ovšem v některých případech nezahrnují reálné stavy, ale pouze vysoké napěťové úrovně a nízké napěťové úrovně.

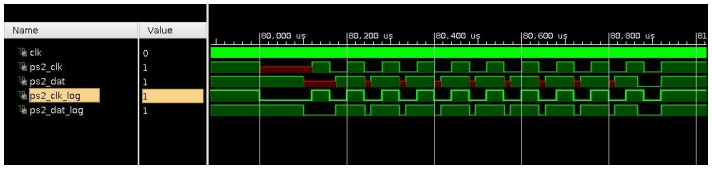


Obrázek 5.7: Celý průběh simulace (Zdroj: simulace v programu Xilinx Vivado)

5.1.6 Simulace vyslání signálu BAT

Tato část simulace proběhla v naprostém souladu s předpoklady. Při vysílání se tedy navržené zařízení chová tak, jak již bylo nastíněno v části popisující funkci tohoto návrhu (5.1.1). Na obrázku je vidět část, kdy si hostitel stáhne na nízkou napěťovou úroveň taktovací vodič, a následně jej pustí s tím, že zařízení následně vyšle 10 pulzů. Dále je vidět Start bit (nízká napěťová úroveň), 8 bitů Scan kódu (8x vysoká napěťová úroveň), Stop bit (vysoká napěťová úroveň), bit liché parity (v tomto případě vysoká napěťová úroveň) a ACK generovaný zařízením.

Na obrázcích je patrná červená úroveň u signálů Ps2_clk, ps2_dat. V simulačním souboru je totiž nastavená hodnota těchto signálů na logickou jedničku. Ovšem během simulace navržený obvod stahuje sběrnici na nízkou napěťovou úroveň, čímž dochází ke konfliktu s hodnotou nastavenou na vysokou napěťovou úroveň v simulaci. Dále se jedná o Behaviorální simulaci, kdy simulační prostředí nebere v úvahu nastavené pull-up či pull-down rezistory.

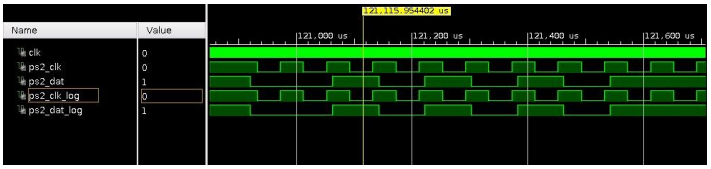


Obrázek 5.8: Průběh simulace v části vysílání signálu BAT (Scan kód 0xFF)(Zdroj: simulace v programu Xilinx Vivado)

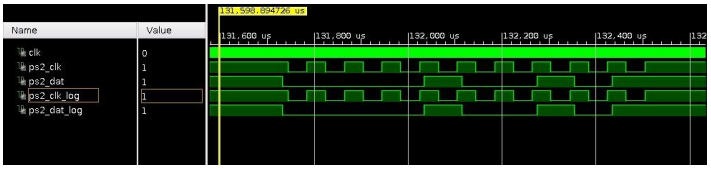
5.1.7 Simulace příjmu potvrzení signálu BAT a přijetí Scan kódu

V tomto případě navržené řešení pouze naslouchá, takže následující obrázky jsou pouze ilustrativní, jelikož jsou generovány simulačním souborem. V tomto případě jsou taktéž uvedeny vodiče Ps2_clk_log, ps2_dat_log, které jsou zde spíše pro úplnost.

Je zde vidět, že hodnota se čte v průběhu nízké napěťové úrovně u hostitele. První průběh je sekvence 01010101 (tedy 10101010 0xAA, data se posílají od nejméně významného bitu po nejvýznamnější) a dále sekvence 00010010 (tedy 01001000 0x48).



Obrázek 5.9: Simulace příjmu Scan kódu 0xAA (Zdroj: simulace v programu Xilinx Vivado)



Obrázek 5.10: Simulace příjmu Scan kódu 0x48 (Zdroj: simulace v programu Xilinx Vivado)

5.1.8 Odhad frekvence

Měření je provedeno pomocí již zmiňovaného čítače. Obvod čítá pulzy během nízké napěťové úrovně vždy po deseti pulzech hodnoty sečte a spočítá průměrnou hodnotu. Při výpočtech celkové periody, potažmo frekvence, bylo postupováno následovně:

- Pro každou z kláves byla změřena alespoň dvojice hodnot pomocí čítače.
- Z takto naměřených hodnot je spočítána průměrná hodnota s vyloučením krajních bodů.
- Takto naměřená hodnota je vynásobená dvěma, čítač počítá pouze dobu, kdy je sběrnice ve stavu logické jedničky. Tímto se dostane počet pulzů 100MHz oscilátoru z dobu logické jedničky na taktovacím vodiči sběrnice PS/2.
- Následně je hodnota převedena na µs, a to díky znalosti periody 100MHz oscilátoru, ke kterému je měření vztaženo.
- Následně je z této hodnoty spočítána frekvence ze známého vzorce:

$$f = \frac{1}{T}$$

Následuje tabulka s takto spočítanými hodnotami. Tabulku obsahující všechny naměřené hodnoty lze shlédnout v Příloze A na konci dokumentu.

Průměrné trvání nízké napěťové úrovně (× 10 ns)	3157,63
Průměrná perioda pulzů PS2_CLK (× 10 ns)	6315,26
Průměrná perioda pulzů PS2_CLK [µs]	63,1526
Průměrná perioda pulzů PS2_CLK [s]	6,3153E-05
Frekvence PS2_CLK [kHz]	15,84

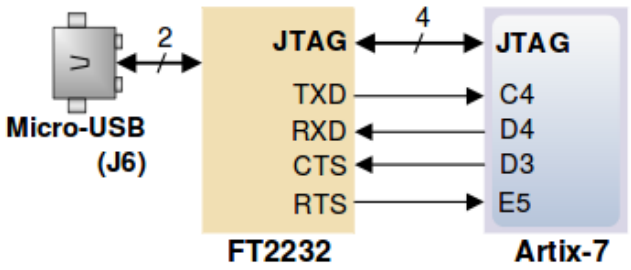
Frekvence signálu PS/2 clk vychází přibližně 15,84 kHz, což odpovídá předpokladu, že tato frekvence bude v rozmezí mezi 10 a 16,7 kHz.

5.2 Implementace UART

Tato kapitola bude věnovaná vysvětlení implementace obvodu UART v rámci diplomové práce.

Jelikož není součástí vybraného přípravku (Digilent Nexys 4) D-SUB konektor, pomocí kterého by bylo možné připojit klasickou sériovou komunikaci založenou na RS-232, je nutné využít převodník UART-USB FTDI FT2232HQ, aby bylo možné pomocí virtuálního COM portu na PC komunikovat s přípravkem. Komunikace pak probíhá po vývodech TXD/RXD.

V případě této práce byla použita rychlost 9600 Baudů a použité schéma je 8N1 (8 datových bitů jeden stop bit a bez paritního bitu), více viz kapitola 2.2 a obrázek 2.6. Signály pro hardwarové řízení toku informací nejsou používané, na následujícím obrázku je připojení obvodu FT2232HQ k FPGA Artix-7 na desce Nexys 4.

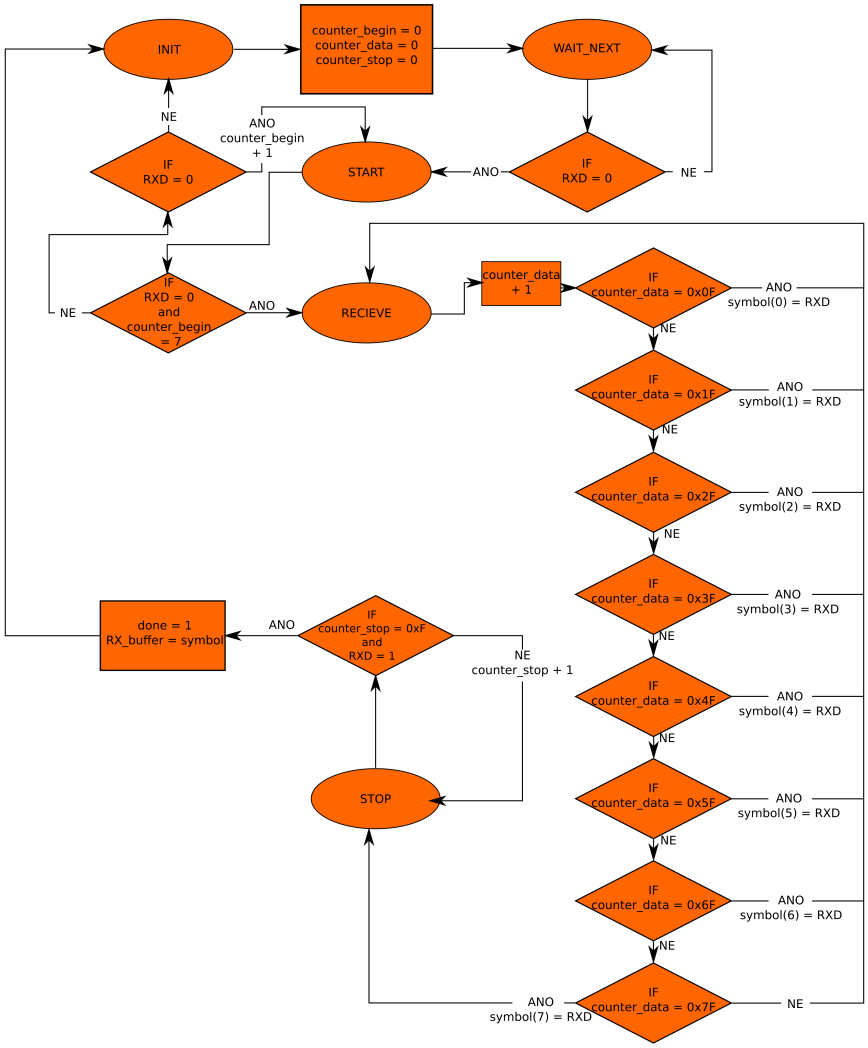


Obrázek 5.11: Připojení FT2232HQ k FPGA Artix-7 (Zdroj:Figure 6.Nexys 4 FT2232HQ connections [2])

5.2.1 Nárh řízení

V rámci práce je implementován pouze příjem znaků, ne vysílání. Komunikace směrem od přípravku nebyla tímto způsobem nutná nicméně pro vytvoření této části by stačilo vytvořit komponentu pro příjem s velmi podobným stavovým automatem, jako je automat přijímací, který bude popsán dále v této kapitole.

Přijímací automat pracuje na principu převzorkování, takže vzorkovací frekvence je 16 krát větší než je rychlost (v tomto případě tedy $16 \times 9600 = 153600Hz$). Toto je využito, protože se jedná o asynchronní příjem, a tudíž hodinové generátory protějších zařízení nejsou nikterak synchronizovány. Pro referenční kmitočtem 100 MHz je tedy nutné vytvořit čítač modulo 651 (jelikož $\frac{100\times10^6}{153600} = 651,041\overline{6}$), aby byla dosažena požadovaná frekvence, o toto se stará čítač, který posílá komponentě pro příjem pulz vždy když dojde k nulování čítače. Na obrázku 5.12 se nachází vyobrazení tohoto stavového automatu.



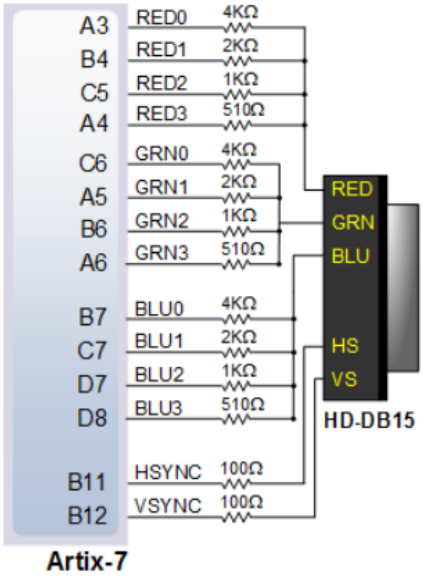
Obrázek 5.12: Stavový automat pro řízení přijímání (Zdroj: Vlastní zpracování)

Výchozím stavem tohoto automatu je stav INIT, ve kterém dojde vždy k vynulování všech použitých čítačů. Poté následuje stav WAIT_NEXT, ve kterém se čeká na logickou nulu na přijímacím vodiči.Pokud automat zaznamená logickou nulu, přejde do stavu START, ve kterém se pomocí čítače počítá do 7, to indikuje, že se automat dostal přibližně do časové polohy uprostřed start bitu (jelikož je použita 16 krát větší vzorkovací frekvence), to se stane pouze pokud je přijímací vodič ve stavu logické nuly v opačném případě se automat přepne do stavu INIT. V případě, že vše probíhá správně, přejde se na stav RECIEVE, ve kterém dochází k vlastnímu přijmu dat. V tomto stavu čítač čítá vždy po 16 krocích pro každý bit (15,31,47,63,79,95,111,127) a ukládá navzorkovaný stav vodiče. Po posledním kroku dojde k přechodu do stavu STOP.V tomto stavu se automat ujistí, že se nachází na pozici stop bitu a stav vodiče je logická jednička. Poté dojde k poslání příchozího bytu směrem k vyšším komponentám a signalizaci, že přišel nový bajt a následně se automat překlopí do stavu INIT, ve kterém dojde k vynulování čítačů a automat opět čeká na nový bajt.

5.3 Zobrazení pomocí rozhraní VGA

V této kapitole je popsána implementace řízení rozhraní VGA, tedy popsáno generování synchronizačních signálů, generování barev pixelů a také výsledky simulací s tím spojené. Dále jsou níže popsány upřesňující informace fungování VGA na přípravku Digilent Nexys 4.

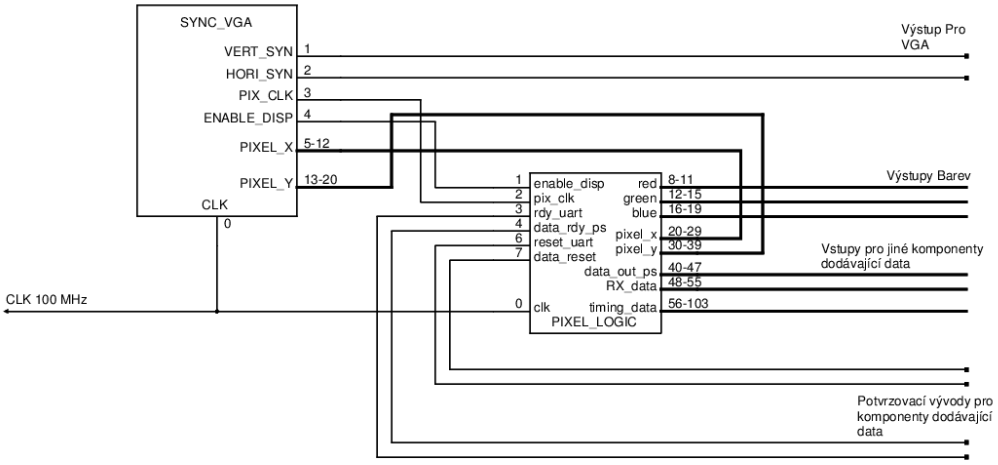
Na vývodech pinů symbolizujících barvy RGB jsou napojeny odporové děliče, které pracují společně s 75 Ω zakončením. Pomocí těchto děličů je možno nastavovat na pin konkrétní barvy napětí v rozmezích 0 - 0,7 V o 16(4-bitová slova) hladinách, což nám umožňuje zobrazit $2^{4*3} = 2^{12} = 4096$ barev. Konkrétnější zapojení lze shlédnout na obrázku 5.13.



Obrázek 5.13: Zapojení Rozhraní VGA na přípravku Nexys 4 (Figure 11 Nexys 4 VGA Interface [2])

5.3.1 Navržené ovládání zobrazení obrazu pomocí rozhraní VGA

Oproti ostatním navrženým implementacím, ovládání řízení rozhraní VGA neobsahuje žádný stavový automat. Na následujícím obrázku je zapojení komponent použitých pro řízení VGA.



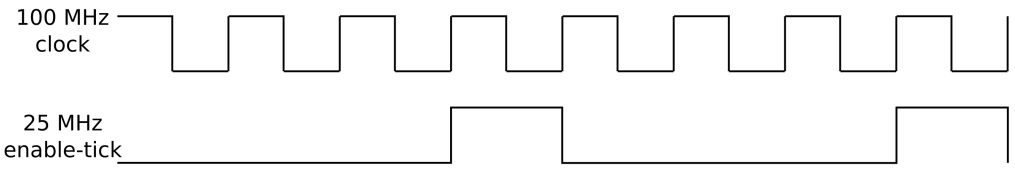
Obrázek 5.14: Zapojení komponent při řízení

Řízení se v podstatě sestává ze dvou bloků, první je blok SYNC_VGA. Tento blok generuje na základě vstupního hodinového taktu 100 MHz výstupy se synchronizačními signály VSYNC a HSYNC, viz kapitola 5.3.2. Dále se zde pomocí čítače generuje ukazatel na řádek a také ukazatel na sloupec, pomocí kterých lze dostat polohu aktuálního zobrazovaného pixelu. Rovněž tento blok dává vědět pomocí signálu ENABLE_DISP, kdy je možné zobrazovat data na obrazovku a kdy se žádná data nezobrazují.

Dalším blokem, který se nachází na obrázku je blok PIXEL_LOGIC, který udává, jakou barvu bude mít každý pixel a ovládá výstupy RGB. Do tohoto bloku putují data z jiných komponent, která mají data k zobrazení. O tomto bloku viz. 5.3.3.

5.3.2 Synchronizace a navržené časování - VGA_SYNC

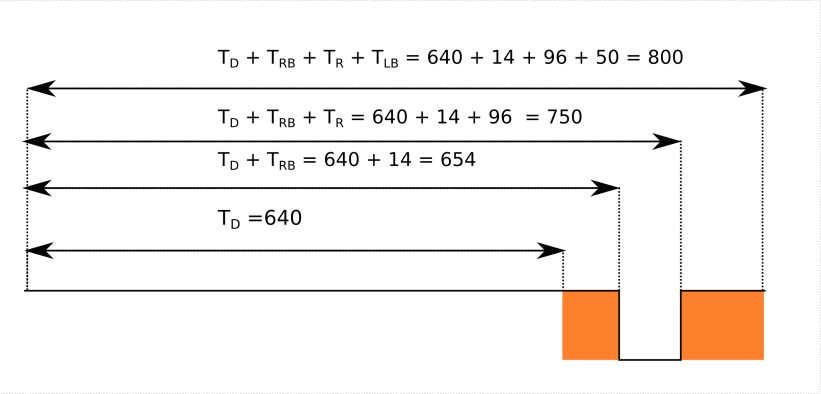
Aby bylo dodrženo správné časování při zobrazování pixelů, tak je nutné z referenčního kmitočtu odvodit 25 MHz kmitočet, tímto krokem se ovšem nevytvoří plnohodnotný hodinový kmitočet o frekvenci 25 MHz, ale spíše pulz, který bude aktivní jednou za každých 40 ns, viz následující obrázek.



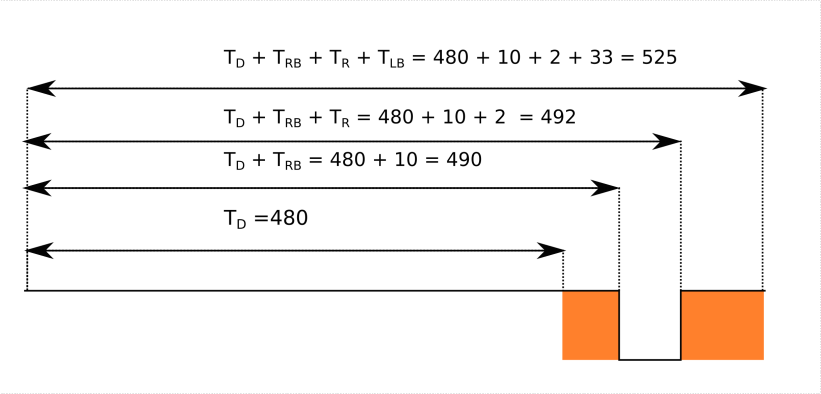
Obrázek 5.15: 100MHz Referenční kmitočet a generovaný pulz (Zdroj: Vlastní zpracování)

K odvození polohy aktuálního pixelu je nutné vycházet ze synchronizačních signálů Vertical Synchronization a Horizontal Synchronization. Pomocí čítače vytvořených 25

MHz pulzů lze vygenerovat tyto synchronizační signály. Pro jednoduchost je lepší nechat čítač čítat od okamžiku, kdy v diagramu dochází k zobrazování informací, více viz. následující obrázky a tabulka.



Obrázek 5.16: Časování Horizontal Synchronization (HSYNC) (Zdroj: Vlastní zpracování)



Obrázek 5.17: Časování Vertical Synchronization (VSYNC)

Časový okamžik	VSYNC	HSYNC
Synchronizační pulz(T_{HS})	525	800
Zobrazení(T_D)	480	640
Zatemňovací pulz(T_R)	2	96
Levý/Horní okraj(T_{RB})	33	50
Pravý/Dolní okraj(T_{LB})	10	14

Tabulka 5.1: Tabulka čítání 25 MHz pro rozhraní VGA (Zdroj [2])

Čítač bude tedy pro rozlišení 640x480 počítat v rámci Horizontal Synchronization do 800 a v rámci Vertical Synchronization do 525. Je ale nutné přidat podmínky pro ostatní signály. Pro synchronizační signál HSYNC bude platit, že pokud čítač není v mezích od 654 do 749, bude výstup HSYNC roven logické nule, jindy bude ve stavu logické jedničky. Taktéž pro signál VSYNC, kde toto platí pro hodnoty od 490 do 491.

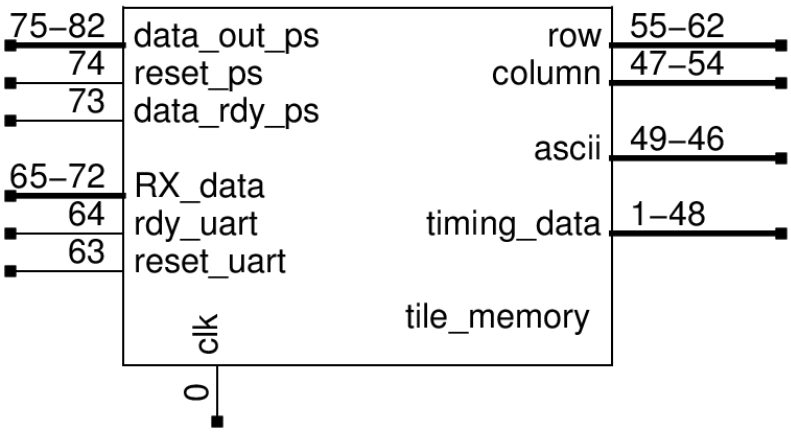
Co se týče hodnoty signálu enable_disp tak ta se bude rovnat stavu logické jedničky v případě, že oba čítače (HSYNC a VSYNC) budou odpovídat na viditelnou oblast (tedy 0-639 respektive 0-479). Čítače jsou v rámci bloku sync_vga vyvedené, aby ostatní komponenty měly informaci o aktuálním pixelu a to v rámci vývodů pixel_x a pixel_y.

5.3.3 Generování pixelů - pixel_logic

Pro jednoduchost je obrazovka zmenšena o rámeček s šířkou 20 pixelů, dále je obrazovka rozdělena do dlaždic 20x10 pixelů, což umožňuje vypisovat různé znaky a není nutnost velké paměti, která by uchovávala obsah všech pixelů obrazovky.

Blok pixel_logic se sestává ze tří částí, první část souvisí se zapouzdřenou komponentou tile_memory, která uchovává informace o "dlaždicích"viz dále. Další část je dekodér, který pomocí ASCII hodnoty určí, o který symbol se jedná a tuto hodnotu převede na font o velikosti 16 x 8 bitů. Jako poslední jsou v tomto bloku výpočty různých důležitých hodnot.

Komponenta tile_memory, která je zapouzdřena do bloku pixel_logic uchovává pro každý řádek o šířce 20 pixelů (16 viditelných, ostatní jsou mezera) znaky ve formě ASCII hodnot. Každý řádek může obsahovat až 60 znaků. Pro ukládání znaků je využita proměnná velká 480 bitů, pro první sloupec je použit bit 479 až bit 472, ostatní sloupce jsou v této proměnné sestupně(tzn. 471-464,463-456 atd.). Výjimkou jsou první řádky, pro které je použita jedna proměnná obsahující všechny implementované znaky. Dále tento blok obsahuje vyhodnocování pro příjem ASCII znaků od klávesnice a příchozí znaky z obvodu UART více viz. 5.4.3. Příchozí signály column a row představují aktuální zobrazovanou dlaždici, toto je generováno z hodnot pixel_x respektive pixel_y. Hodnoty těchto dvou signálu jsou v podstatě zbytky po dělení 10 respektive 20, tak aby se zjistila poloha dlaždice 10x20 (šířka x výška dlaždice).

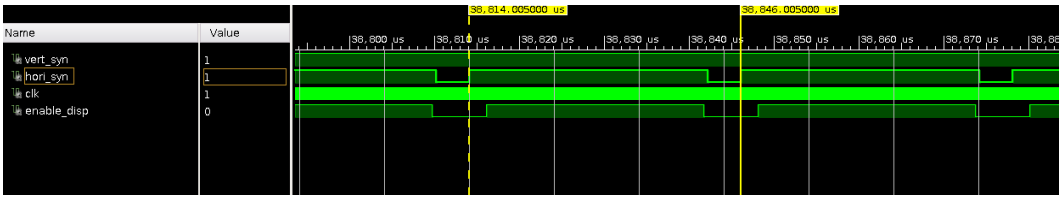


Obrázek 5.18: Komponenta tile_memory Vertical Synchronization (VSYNC)

5.3.4 Simulace synchronizačních pulzů

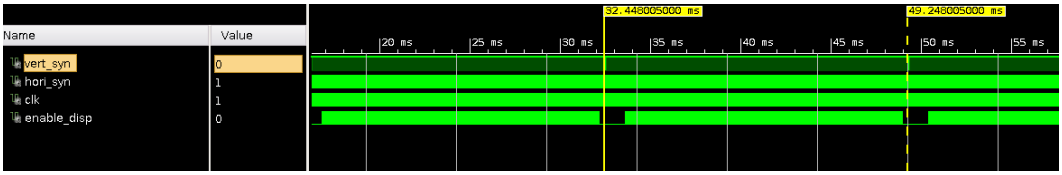
Simulace je jako v případě simulací na sběrnici PS/2 prováděna v simulačním software, jenž je součástí Xilinx Vivado.

Oproti simulacím na sběrnici PS/2 je tato simulace prováděna pouze s hodinovým vstupem a vše ostatní je od tohoto odvozeno. Následující obrázky dokazují, že časování synchronizačních signálů HSYNC a VSYNC odpovídá předpokladům definovaných v kapitole 2.3.4.



Obrázek 5.19: Simulace a kontrola pulzů HSYNC (Zdroj: Vlastní zpracování)

Podle kurzorů na obrázku 5.19 doba trvání jednoho pulzu je 32 µs, což odpovídá, jelikož $(32 * 10^{-}6)/800 = 40 * 10^{-}9 = 40$ ns. To odpovídá jednomu pulzu 25 MHz.



Obrázek 5.20: Simulace a kontrola pulzů VSYNC (Zdroj: Vlastní zpracování)

Na obrázku 5.20 kurzory zaznamenali 16,8 ms, což opět odpovídá předpokladu $(16,8 * 10^{-}3)/525 = 32$ µs. Toto opět souhlasí, protože vypočtená hodnota je stejná jako doba trvání jednoho řádku (HSYNC).

Na obrázcích lze také vidět fungování signálu enable_disp, který značí, kdy je možné vysílat na viditelnou část obrazovky.

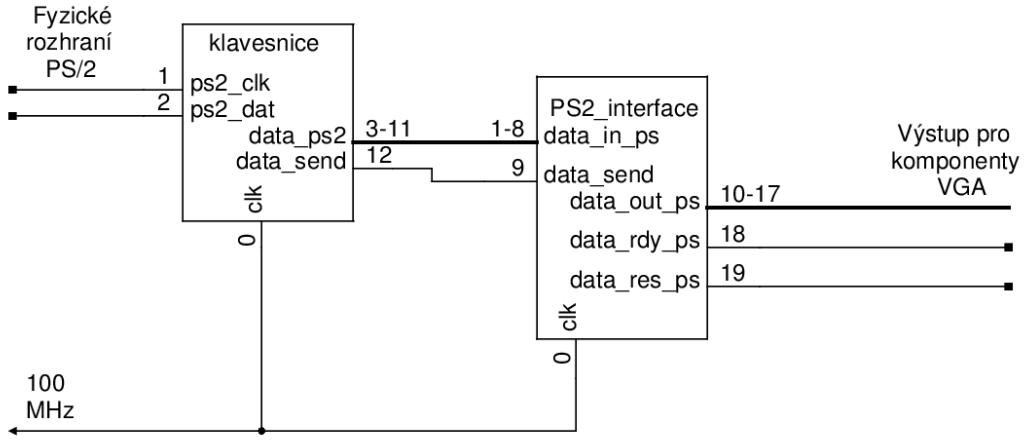
5.4 Finální zapojení

V této podkapitole je popsáno, jakým způsobem jsou všechny popsáné části propojené do jednoho celku.

5.4.1 Připojení klávesnice PS/2 a přijímací strany obvodu UART

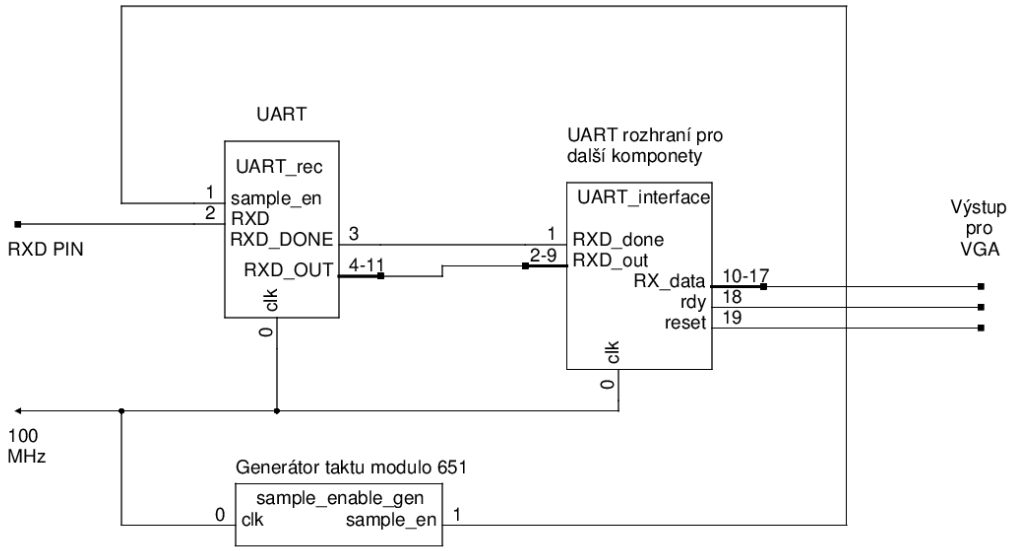
Znaky z PS/2 klávesnice a obvodu UART příchazejí ve tvaru ASCII znaků. Díky tomu je připojení zmiňovaných komponent v podstatě stejné a proto je je jejich popis shrnut do jedné kapitoly.

Princip připojení je poměrně jednoduchý, nejdříve je zde blok, který se stará o přijímaní znaků a po přijetí vyšle krátký impulz, že došlo k přijetí, vyšší komponenta, která slouží jako rozhraní pro další komponenty toto přijme a na výstup převede přijatá data s tím, že nastaví logickou jedničku na výstup označený ready signalizující že jsou data připravena. Pokud vyšší komponenta přijme data, dojde k resetování tohoto ready signálu pomocí resetovacího vstupu. Na následujícím propojení je toto zobrazeno u rozhraní PS/2.



Obrázek 5.21: Propojení komponent PS/2 pro další komponenty (Zdroj:Vlastní zpracování)

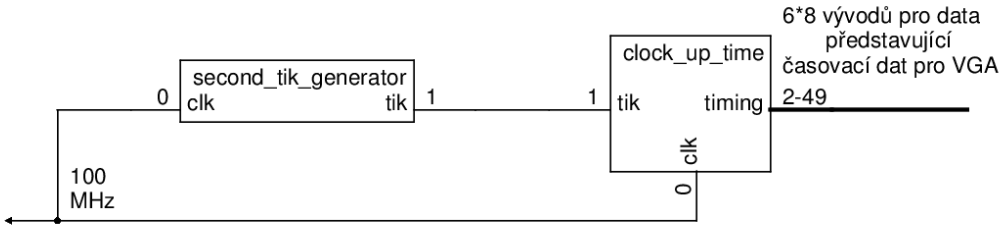
Z následujícího i předešlého obrázku lze vidět, že připojení obou rozhraní k vyšší komponentě je v podstatě stejné, s tím rozdílem, že u obvodu UART je čítač modulo 651, který dodává takt pro vzorkovací kmitočet 16 krát vyšší než přenosová rychlost.



Obrázek 5.22: Propojení komponent UART pro další komponenty (Zdroj:Vlastní zpracování)

5.4.2 Připojení počítání času

Poslední blok který je připojený na komponenty ovládájící VGA jsou komponenty pro počítání času. Toto propojení je vyobrazeno na následujícím obrázku.



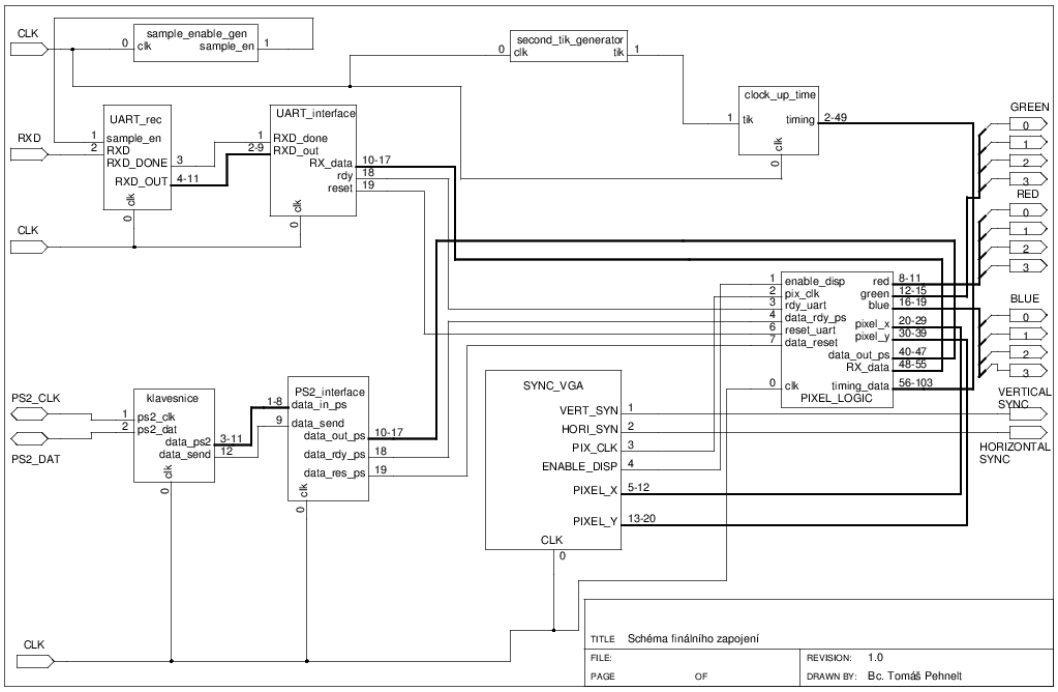
Obrázek 5.23: Propojení komponent pro počítání doby spuštění přípravku

V první komponentě second_tik_generator jsou generovány pulzy s periodou 1 s což představuje buzení pro čítač v následující komponentě. Komponenta clock_up_time je v podstatě několik čítačů modulo 6 a modulo 10, díky kterým se převádí pulzy na následující schéma: HH : MM : SS (HH = hodiny, MM = minuty, SS = sekundy). Každá hodnota pozice je nejdříve dekódovaná na ASCII znak, a poté je vlastním vývodem přenesena do komponenty řídící VGA.

5.4.3 Připojení VGA - Finální propojení

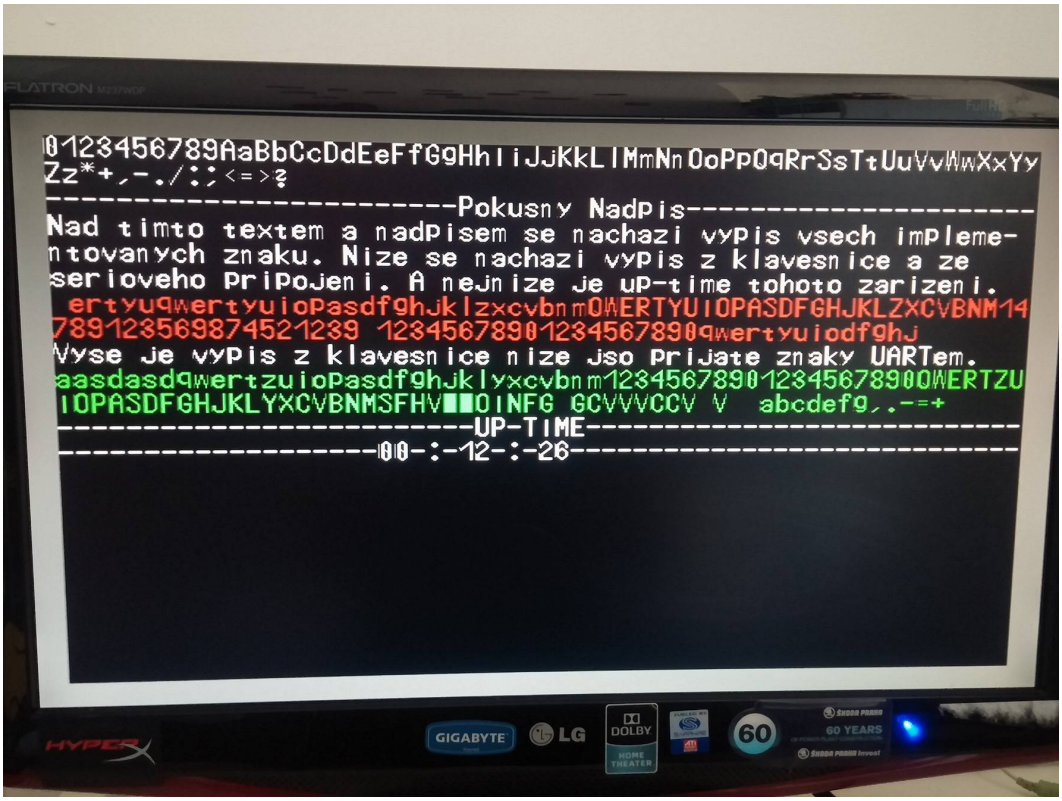
V této poslední části je popsáno, jak jsou všechny komponenty pospojované spolu, nebo spíš spojené do komponent pro řízení VGA.

Všechny ostatní rozhraní jsou přivedena do komponenty pixel_logic, ve kterém dochází k vyhodnocení. Ve své podstatě jsou data v komponentě pixel_logic směrována dále do tile_memory, kde dochází k vlastnímu vyhodnocení. V tomto bloku podle příchozích ASCII znaků komponenta vyhodnotí, zda se jedná o znak nebo o řídící symbol typu nový řádek nebo mazání předchozího znaku. Na následujícím obrázku je vyobrazeno celkové propojení všech vytvořených komponent.



Obrázek 5.24: Propojení všech komponent

Jak již bylo předesláno v kapitole 5.3.3, na obrazovce jsou nejdříve vypsány všechny implementované znaky pro zobrazení s pomocí rozhraní VGA. Dále je zde vypsán následující popis: „Nad timto textem se nachazi vypis vsech implementovanych znaku. Nize se nachazi vypis z klavesnice a ze serioveho pripojeni. A nejnize je up-time zarizeni." Za tímto textem se nachází červenou barvou výpis znaků zadávaných do klávesnice připojené k přípravku. Následně je vypsán další popis: „Vyse je vypis z klavesnice nize jsou prijate znaky UARTem." Za čímž následuje výpis příchozích znaků ze sériové konzole přijaté obvodem UART, které jsou vypisovány zelenou barvou. A nakonec je zde řádek zobrazující dobu běhu demonstrace. Takto popsaná grafika je ke shlédnutí na následující fotografii výpisu (5.25).



Obrázek 5.25: Výstup na obrazovce

6 Zhodnocení

Výsledkem diplomové práce je návrh obvodu pomocí jazyka VHDL určený pro přípravek Digilent Nexys 4, který je schopen zobrazovat znaky přicházející z různých rozhraní. V demonstrační ukázce jsou implementovány PS/2, ve své podstatě i RS-232 a VGA.

Při vypracovávání práce byl použit přípravek Digilent Nexys 4 a to z několika důvodů. Prvním důvodem bylo, že pro vývoj návrhu bylo použito nové moderní vývojové prostředí od společnosti Xilinx, a to prostředí Xilinx Vivado. Dalším důvodem bylo novější FPGA typu Artix-7™, které obsahuje více programovatelných logických buněk a také je z FPGA dostupných pro tuto práci nejnovější. Nicméně tento výběr má několik úskalí. Kit Digilent Nexys 4 neobsahuje rozhraní pro RS-232, tedy konektor D-SUB s devíti nebo dvaceti-pěti piny, který je nutný ke klasické RS-232 komunikaci. Toto je řešeno pomocí můstku FTDI FT2232H, který přenáší tuto sériovou komunikaci po USB. Dále tento přípravek neobsahuje konektor typu mini-DIN, který je nutný pro fungování sběrnice PS/2. Tento problém je opět překonán pomocí USB, jelikož přípravek Digilent Nexys 4 obsahuje port USB-HID, na který je připojen mikrořadič fungující jako emulátor sběrnice PS/2 a vysílá data směrem k FPGA, jako by se jednalo o plnohodnotné PS/2 zařízení. Z pohledu portu VGA zde není žádné vážné omezení.

Pokud se jedná o ostatní přípravky, tak návrh by měl být převeditelný i na ně, jelikož VHDL návrh je přenositelný. Byla by nutná změna několika věcí. Za prvé by bylo nutné zadat správné vývody místo nynějších, jelikož se jejich značení jistě bude lišit. Dále by bylo nutné vzít v úvahu jiné frekvence vestavěných oscilátorů, v případě přípravku Nexys 3 by toto problém nebyl, jelikož obsahuje oscilátor o stejné frekvenci jako přípravek Nexys 4. Toto neplatí pro přípravek Spartan-3E Starter Board, který obsahuje oscilátor s frekvencí 50 MHz, v případě tohoto přípravku by se tedy musely změnit hraniční hodnoty čítačů, aby docházelo ke správnému chodu navrženého zařízení.

V teoretické části práce jsou probrány základy nutné k pochopení a implementaci jednotlivých rozhraní. Zaměřil jsem se na rozhraní VGA,PS/2 a RS-232. V praktické části práce je popsána navržená implementace řízení těchto rozhraní.

Komunikace přes rozhraní PS/2 je řešena pomocí 3 stavových automatů a to automatem řídícím, vysílacím a přijímacím. Řídící automat přepíná automat vysílací a přijímací. Data jsou poté dále dekódována ze Scan kódů do ASCII, čehož je dále využito v dalším zpracovávání.

Komunikaci typu RS-232 byla realizována ve finální podobě jen v přijímacím směru, tudíž nebyla implementována vysílací část, jelikož nebylo vyžadováno aby byla jakákoliv data vysílána do PC přes toto rozhraní.

Ke komponentám ovládající rozhraní PS/2 a RS-232 bylo nutné přidat další komponentu, která se chová jako rozhraní, jež umožňuje jednodušší napojení na další komponenty, jako je například ovládání portu VGA.

Ovládání portu VGA je provedeno dvěma komponentami, první zajišťující správné synchronizační signály a druhá, která generuje korektní hodnoty signálů náležejícím barevným složkám obrazu, který má být zobrazen na displeji, jenž je připojen tímto rozhraním k přípravku.

Navržené řešení je možné jednoduše rozšířit o další rozhraní, nicméně tohoto z důvodu časové náročnosti nebylo dosaženo. Navržené řešení pro ovládání VGA je orientované znakově, tento efekt lze ovšem potlačit a vykreslit libovolný obrazec. Další rozšíření by bylo možné například změnou rozlišení, které by se dalo změnit pouhou změnou parametrů v synchronizační komponentě pro řízení VGA.

Seznam zkratek

ACK Acknowledgement. 30
ASCII American Standard Code for Information Interchange. 6, 13, 21, 23, 25–29, 37, 39–41, 43, 46

BAT Basic Assurance Test. 6, 11, 21, 23, 25, 29, 30, 46

COM Communication Port. 32
CPLD Complex Programmable Logic Device. 20
CRT Cathode Ray Tube. 15, 16, 46

D-SUB D-Subminiature. 12, 14, 20, 32, 43
DDC Display Data Channel. 14
DIN Deutsches Institut für Normung. 9, 21, 43
DVI Digital Visual Interface. 13

EIA Electronic Industries Organisation. 12

FPGA Field Programmable Gate Array. 5, 8, 18–21, 43
FTDI Future Technology Devices International. 32, 43

GND Ground. 14

HDMI High-Definition Multimedia Interface. 13
HID Human Interaction Device. 19, 43
HLS High-Level Synthesis. 18
HSYNC Horizontal Synchronization. 14, 16, 17, 35–38, 46, 47

IBM International Business Machines Corporation. 9, 14
IP Intelectual Property. 18

LCD Liquid Crystal Display. 16, 20
LED Light Emiting Diode. 19, 20

PC Personal Computer. 9, 32
PDM Pulse Density Modulation. 19
PS/2 IBM Personal System 2. 5, 6, 8, 9, 13, 20–22, 24, 25, 29, 30, 38, 39, 43, 46
PWM Pulse Width Modulation. 19

RGB Red, Green and Blue. 15, 34, 35
RJ Registered Jack. 19
RS Registered Jack. 8, 12, 19, 32, 43

TIA Telecommunications Industries Organisation. 12
TM Trade Mark. 18–21, 43

UART Universal Asynchronous reciever and transmitter. 5, 6, 8, 12, 19, 28, 32, 39, 41
USB Universal Serial Bus. 9, 19–21, 28, 32, 43

VESA Video Electronics Standards Association. 14
VGA Video Graphics Array. 5, 6, 8, 13–16, 19, 20, 34, 36, 40, 41, 43, 44, 47
VHDC Very High Density Connector. 19
VHDL Very high speed integrated circuits hardware design language. 5, 8, 43
VSYNC Vertical Synchronization. 14–17, 35–38, 47

XADC Xilinx Analog to Digital Converter. 19

Seznam obrázků

Seznam tabulek

Reference

[1] Lazaridis, G.: The PS2 protocol. 2009.
Dostupné z: <http://pcbheaven.com/wikipages/The_PS2_protocol/>

[2] Nexys4™ FPGA Board Reference Manual. 2013.
Dostupné z: <https://www.digilentinc.com/Data/Products/NEXYS4/Nexys4_RM_VB2_Final_5.pdf>

[3] Chapweske, A.: The PS/2 Keyboard Interface.
Dostupné z: <http://www.computer-engineering.org/ps2keyboard/>

[4] Chapweske, A.: The PS/2 Mouse/Keyboard Protocol.
Dostupné z: <http://www.computer-engineering.org/ps2protocol/>

[5] Spartan-3E Starter Kit Board User Guide. 2006.
Dostupné z: <http://www.digilentinc.com/Data/Products/S3EBOARD/S3EStarter_ug230.pdf\#G4.261995>

[6] Spartan 3E Starter Board.
Dostupné z: <http://www.digilentinc.com/Products/Detail.cfm?NavTop=2\&NavSub=423\&Prod=S3EBOARD\&CFID=19687683\&CFTOKEN=616379f1b7b0d4-DEBCF4D3-5056-0201-0208DEBBC79B4FD8>

[7] Sinclair, I. R.: *Practical electronics handbook*. Boston: Newnes, 6 vydání, 2007, ISBN 9780750680714.

[8] Wilson, P. R.: *Design recipes fot FPGAs*. Amsterdam: Elsevier, c2007, ISBN 978-0-7506-6845-3.

[9] Axelson, J.: *Serial port complete*. Madison, WI: Lakeview Research, druhé vydání, c2007, ISBN 193144806X.

[10] VGA.
Dostupné z: <http://www.computerhope.com/jargon/v/vga.htm>

[11] Jack, K.: *Video demystified*. Boston: Newnes/Elsevier, páté vydání, c2007, ISBN 9780750683951.

[12] Dhir, A.: *The digital consumer technology handbook*. Boston, Mass.: Newnes, duben 2004 vydání, 2004, ISBN 9780750678155.

[13] Catura-Houser, T.; O'Boyle, H.: ExamInsight for A+ core hardware technology exam (220-221). c2001.

[14] Chu, P. P.: *FPGA prototyping by VHDL examples*. Hoboken: Wiley-Interscience, c2008, ISBN 9780470185315.

[15] VGA Video Signal Format and Timing Specifications. 2007.
Dostupné z: <http://javiervalcarce.eu/html/vga-signal-format-timming-specs-en.html>

[16] Nexys 3 ™ FPGA Board Reference Manual. April 11 , 2016.
Dostupné z: <https://reference.digilentinc.com/_media/nexys:nexys3:nexys3_rm.pdf>

Seznam příloh

Příloha A

Tabulky hodnot – trvání nízké napěťové úrovně na sběrnici PS/2

HEX	DEC
0C59	3161
0C04	3076
0C65	3173
0CCE	3278
0CA6	3238
0C59	3161
0C00	3072
0C58	3160
0C64	3172
0CAE	3246
0C57	3159
0CB2	3250
0C58	3160
0C58	3160
0CAC	3244
0C57	3159
0C3F	3135
0C3E	3134
0C5A	3162
0C67	3175
0C66	3174
0C59	3161
0C64	3172
0C81	3201
0C58	3160
0C59	3161
0C6F	3183
0C3E	3134
0C59	3161
0CCA	3274
0BAF	2991
0C59	3161
0C3F	3135
0C3F	3135
0C59	3161
0C3F	3135
0C78	3192
0C64	3172
0BC7	3015
0C92	3218
0C59	3161
0CBE	3262
0C3F	3135
0C57	3159
0BF8	3064

HEX	DEC
0C40	3136
0C7E	3198
0C59	3161
0C3E	3134
0BEE	3054
0C59	3161
0C23	3107
0BE4	3044
0C59	3161
0C40	3136
0C3F	3135
0C3E	3134
0C59	3161
0C63	3171
0CAA	3242
0C58	3160
0CA9	3241
0CE8	3304
0C59	3161
0C56	3158
0C66	3174
0C50	3152
0C58	3160
0CC6	3270
0C64	3172
0C57	3159
0C70	3184
0C3F	3135
0C59	3161
0C40	3136
0CC9	3273
0C3F	3135
0C5A	3162
0C40	3136
0BAF	2991
0C5A	3162
0C40	3136
0B04	2820
0C59	3161
0C3E	3134
0CF3	3315
0C75	3189
0CF3	3315
0C3F	3135
0C59	3161

HEX	DEC
0C0A	3082
0C30	3120
0C58	3160
0C58	3160
0C71	3185
0C47	3143
0C55	3157
0C63	3171
0C3F	3135
0C58	3160
0C3F	3135
0C3F	3135
0C58	3160
0C50	3152
0CCA	3274
0C58	3160
0C3F	3135
0C40	3136
0C5A	3162
0BE2	3042
0C2B	3115
0C5A	3162
0C40	3136
0CC0	3264
0C75	3189
0C59	3161
0C40	3136
0BE7	3047
0C59	3161
0C40	3136
0C66	3174
0C59	3161
0C59	3161
0CF3	3315
0C40	3136
0C58	3160
0C3F	3135
0C40	3136
0C58	3160
0C66	3174
0C66	3174
0C59	3161
0C2C	3116
0C65	3173
0C1A	3098

HEX	DEC
0C58	3160
0C7F	3199
0C74	3188
0C58	3160
0C40	3136
0C3E	3134
0C59	3161
0C47	3143
0C3B	3131
0B75	2933
0C59	3161
0C3F	3135
0C3A	3130
0C57	3159
0C3F	3135
0CEA	3306
0C5A	3162
0BB9	3001
0C59	3161
0C59	3161
0C66	3174
0C58	3160
0C1B	3099
0C40	3136
0C58	3160
0C5A	3162
0C3E	3134
0BEC	3052
0C59	3161
0C77	3191
0C9F	3231
0C5A	3162
0C65	3173
0C73	3187
0C59	3161
0C59	3161
0C3C	3132
0C67	3175
0C58	3160
0BC3	3011
0C5B	3163
0C58	3160
0C63	3171
0C79	3193
0C5A	3162

HEX	DEC
0C59	3161
0C00	3072
0CF0	3312
0C58	3160
0C6F	3183
0CAF	3247
0C58	3160
0C40	3136
0C40	3136
0C58	3160
0C40	3136
0C5B	3163
0C40	3136
0C58	3160
0C58	3160
0C1F	3103
0C59	3161
0CF3	3315
0C5E	3166
0C59	3161
0C64	3172
0CE9	3305
0C64	3172
0CA0	3232
0C59	3161
0C56	3158
0C38	3128
0CE5	3301
0C65	3173
0BB1	2993
0C3F	3135
0C58	3160
0BC9	3017
0C40	3136
0C59	3161
0C3F	3135
0C66	3174
0C59	3161
0C57	3159
0C3F	3135
0C40	3136
0C58	3160
0C3F	3135
0C3F	3135
0C59	3161

HEX	DEC
0C64	3172
0CF3	3315
0C58	3160
0C57	3159
0C0F	3087
0CBF	3263
0C41	3137
0C58	3160
0BE8	3048
0C59	3161
0BBA	3002
0CA5	3237
0C58	3160
0C40	3136
0BF4	3060
0BF0	3056
0C59	3161
0C4B	3147
0CE0	3296
0C3C	3132
0C7B	3195
0C59	3161
0C59	3161
0C58	3160
0CCC	3276

Priloha B

Z důvodu velikosti všech napsaných kódů jsou všechny napsané kódy pouze na přiloženém CD a nejsou fyzickou součástí příloh.

Priloha C

V této příloze se nachází seznam položek, které jsou přítomny na přiloženém CD. Všechny kódy jsou součástí těchto projektů a nacházejí se ve složce vždy src/new/ nebo src/imported.

- Tato Diplomová Práce včetně příloh ve formátu PDF
- Soubor .zip obsahující projekt finálního podoby praktické části diplomové práce
- Soubor .zip obsahující projekt obsahující pouze PS/2 implementaci
- Soubor .zip obsahující projekt obsahující pouze UART implementaci