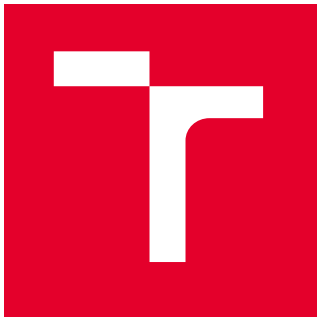


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

ROZŠÍŘENÍ ROZHRANÍ PRO ŘÍZENÍ BUŇKOVÉHO MĚNIČE POMOCÍ FPGA

INTERFACE EXTENSION FOR POWER CELL INVERTOR CONTROL WITH FPGA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

Petr Matoušek

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Petr Petyovský, Ph.D.

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Petr Matoušek ***ID:*** 217431

Ročník: 3 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Rozšíření rozhraní pro řízení buňkového měniče pomocí FPGA

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s vlastnostmi buňkového měniče a architekturou nadřazeného řídicího systému. Definujte blokové schéma výsledného technického řešení.
2. Diskutujte vlastnosti a parametry jednotlivých technických komponent blokového schématu řešení (PWM, SPI, SCI) a parametry protokolů pro komunikaci mezi komponentami.
3. Vyberte vhodný obvod FPGA, který bude připojený k řídicímu systému. Zvolte vhodné technické a programové prostředky pro vývoj i následnou realizaci technického řešení s pomocí FPGA.
4. Navrhněte a implementujte SPI modul pro zpracování přijímaných dat. Otestujte jeho funkčnost.
5. Navrhněte a implementujte PWM modul, který bude generovat požadované průběhy výstupních signálů na základě přijatých dat. Otestujte jeho funkčnost.
6. Navrhněte a otestujte modul SCI, který bude zajišťovat odesílání datového rámce obsahující informace o šířce pulsu v přesně definovaných časech.
7. Realizujte a ověřte funkci celého konceptu a otestujte celkovou funkčnost zařízení.
8. Demonstrujte správnou funkci zařízení na reálném HW.
9. Zhodnoťte dosažené výsledky a navrhněte další možná rozšíření.

DOPORUČENÁ LITERATURA:

[1] PINKER, J. POUPA, M: Číslicové systémy a jazyk VHDL. 2006, ISBN 80-7300-198-5.

[2] VIRIUS, Miroslav. Jazyky C a C++: kompletní průvodce. 2., aktualiz. vyd. Praha: Grada, 2011. ISBN 9788024739175.

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 23.5.2022

Vedoucí práce: Ing. Petr Petyovský, Ph.D.

Konzultant: Ing. Tomáš Žůrek, (ELCOM, a. s.)

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

ABSTRAKT

Bakalářská práce se zabývá návrhem a realizací 2 modulů v hradlovém poli, které budou použity jako rozšíření pro řídicí desku měniče. Práce se zabývá návrhem a realizací modulu s univerzálně definovaným počtem kanálů, generujících pulzně šířkově modulovaný signál, který je nastavitelný pomocí Serial Peripheral Interface. Dále se zabývá návrhem modulu s 9 sériovými linkami v hradlovém poli, které se ovládají pomocí Serial Peripheral Interface. Součástí práce je popis používaných periferií, popis principu fungování měniče a nadřazeného řídicího systému, popis implementace v jazyce pro popis Hardware a její testování.

KLÍČOVÁ SLOVA

komplexní PWM generátor, buňkový měnič, Robicon perfect harmony, FPGA, VHDL.

ABSTRACT

The main goal of bachelor thesis is design and implementation of 2 modules in the Field-programmable gate array, which will be used as an extension to the control board of power inverter. The thesis includes the design and implementation of a module with several channels generating a pulse width modulated signal, which is controlled using the Serial Peripheral Interface. It also includes the design of a module with 9 serial lines in the Field-programmable gate array, which are controlled using the Serial Peripheral Interface. Part of the work contains a description of the peripherals used, a description of the operation of principle of the drive and the superior control system, a description of the implementation in the language for the description of Hardware and its testing.

KEYWORDS

complex PWM generator, Power cell, Robicon perfect harmony, FPGA, VHDL.

MATOUŠEK, Petr. *Rozšíření rozhraní pro řízení buňkového měniče pomocí FPGA*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2022, 70 s. Bakalářská práce. Vedoucí práce: Ing. Petr Petyovský, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Petr Matoušek
VUT ID autora:	217431
Typ práce:	Bakalářská práce
Akademický rok:	2021/22
Téma závěrečné práce:	Rozšíření rozhraní pro řízení buňkového měniče pomocí FPGA

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Petru Petyovskému, Ph.D. za konzultace a podnětné návrhy k práci. Dále bych rád poděkoval panu Ing. Tomáši Žůrkovi za odborné vedení práce a firmě ELCOM, a. s., která mi poskytla prostředky pro realizaci bakalářské práce.

Brno . . . .

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Tématem bakalářské práce je navrhnout rozšíření řídicího rozhraní v hradlovém poli pro buňkový měnič. Měnič se typicky používá k přeměně parametrů energie. Lze si jej představit jako uzavřený systém, do kterého vstupuje stejnosměrné napětí z baterie a vystupuje napětí střídavé, které má totožné parametry jako síťové napětí. Řídicími signály v měniči jsou typicky pulzně šířkově modulované signály. Základní řídicí rozhraní však neobsahuje dostatečné množství výstupů s pulzně šířkově modulovanými signály a tedy bylo rozhodnuto, že cílem práce bude navrhnout obvod v hradlovém poli. Oproti použití dalšího mikrokontroléru má implementace, pomocí specializovaného hardware definovaného v hradlovém poli pomocí programovacího jazyku pro popis hardware, výhodu v tom, že lze takovou periferii navrhovat přesně na míru. Přesná definice rozšíření rozhraní je uvedena na následující straně.

Práce se věnuje teoretickému úvodu o využívaných periferiích a funkce měniče. Jsou zde zahrnuty i dvě bloková schémata s výslednou podobou rozšíření v hradlovém poli.

Dále se práce věnuje praktické realizaci zadání. Jsou popsány vybrané prostředky a nástroje pro realizaci zadání včetně postupu při návrhu komplexnějšího projektu v hradlovém poli.

V další části bakalářské práce jsou popsány implementace jednotlivých periferií do hradlového pole. Jmenovitě se jedná o návrh periferie, která generuje pulzně šířkově modulovaný signál s mnoha nastavitelnými parametry. Dále pak návrh periferií SPI Slave a SCI Master, které budou určené pro komunikaci s periferií generující pulzně šířkově modulované signály. Součástí textu popisu implementací jsou popisy funkce, nákresy stavových automatů, analýza zdrojů a jejich simulace.

V závěru se práce zabývá popisem testování navržených periferií na reálném hardware. Jsou uvedeny závěry z testování a případná další možná rozšíření pro řídicí rozhraní.

Úvodní motivace externího zadavatele a rozbor zadání práce

Zadání této bakalářské práce vyplynulo z mé praxe ve firmě ELCOM, a. s. Tato firma se mimo jiné zabývá zakázkovým vývojem a výrobou zdrojů. Celek zdroje tvoří několik buněk, přičemž v každé z nich se nachází pulzní měnič. Buňky jsou ovládány pulzně šířkově modulovaným signálem, který je generovaný z řídicí desky nadřazeného systému. Tento řídicí systém ale neobsahuje dostatek výstupů s těmito pulzně šířkově modulovanými signály pro řízení všech buněk, a tak se musí použít řídicích desek několik, což není optimální řešení. Z těchto skutečností vyplynul požadavek na návrh rozšiřujících karet, které budou obsahovat hradlové pole pomocí kterého bude možné navrhnout a implementovat hardware s dostatečným množstvím výstupů.

První cíl bakalářské práce je implementovat v jazyce VHDL periferii ePWM. Ta bude sloužit pro generování pulzně šířkově modulovaných signálů s mnoha nastavitelnými parametry. Tato implementace bude vycházet z ePWM, která je implementována na čipu TMS320F28335 od společnosti Texas Instruments. Dalšími cíli je implementovat komunikační rozhraní SPI Slave a SCI Master v hradlovém poli v jazyce VHDL. Výsledkem prvního rozšíření bude ePWM periferie nastavitelná pomocí komunikace SPI. Výsledkem druhého rozšíření bude devět sériových linek v programovatelném hradlovém poli, které budou odesílat zprávy z SPI do dalšího hradlového pole. Externím konzultantem pro tuto práci je Ing. Tomáš Žůrek z firmy ELCOM, a. s.

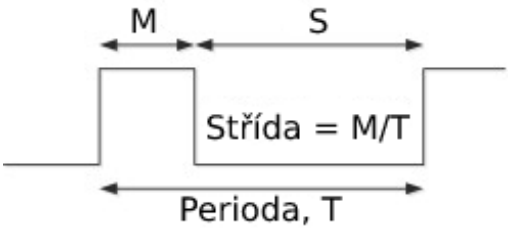
1 Teoretický rozbor použitých periferií při návrhu obvodu v FPGA

V první kapitole jsou principiálně popsány používané periferie. Tato kapitola se věnuje popisu pulzně šířkově modulovaného signálu, popisu synchronního a asynchronního způsobu komunikace a principu v práci používaných komunikací SPI a SCI.

1.1 Princip pulzně šířkové modulace - PWM

Pulzně šířková modulace (PWM) je způsob, jakým lze ovládat poměr napětí signálu s dvou stavovou logikou. Využívá se hojně ve výkonové elektronice, například pro regulaci otáček motoru, plynulé stmívání osvětlení. Možné využití je i pro přenos dat, kdy hodnota střídy odpovídá hodnotě určité měřené veličiny, např.: měření teploty.

Hodnota PWM je digitální signál, který se definovanou dobu nachází ve stavu log. 1 a definovanou dobu ve stavu log. 0. Tyto doby lze nastavovat a měnit tak šířku pulsu - střídu. Na obrázku č. 1.1 je ukázán PWM signál se základními parametry. Parametr M určuje dobu, kdy je signál v log. 1, S pak dobu, kdy je signál v log. 0, parametr T (což je součet M a S), pak určuje periodu. Střída signálu se získá jako podíl M a T [1].



Obr. 1.1: Ukázka PWM signálu s vyznačenými dobami log. hodnot [1]

Velikost amplitudy je obvykle dána napěťovou úrovní, na které mikrokontrolér pracuje. Frekvence PWM je závislá na vstupním signálu hodinového signálu a na rozlišení čítače. Používaná frekvence PWM se liší podle aplikace. Stmívače osvětlení používají frekvence okolo stovek Hz, frekvenční měniče pak od jednotek až po desítky kHz. Spínací frekvence pro audio zesilovače a spínané zdroje se pohybuje okolo desítek až stovek kHz.

S dostatečně velkou spínací frekvencí lze docílit obnovy analogového průběhu. Díky použití analogových filtrů (např. filtr dolní propust) lze průběh vyhladit. Řídicí

systémy pracující s vysokofrekvenční PWM lze realizovat pomocí polovodičových spínačů [14].

1.2 Synchronní a asynchronní způsob komunikace

V případě sériové komunikace se používají dva druhy přenosu - synchronní a asynchronní. Tato podkapitola popisuje rozdíly mezi synchronním a asynchronním přenosem dat.

1.2.1 Synchronní metoda

V tomto způsobu komunikace sdílejí všechna zařízení na sběrnici společný hodinový signál. To s sebou nese výhodu rychlejší komunikace mezi zařízeními. Nevýhodou je, že je potřeba vodič navíc. Typickými představiteli, které této metody využívají, jsou SPI nebo I2C [19].

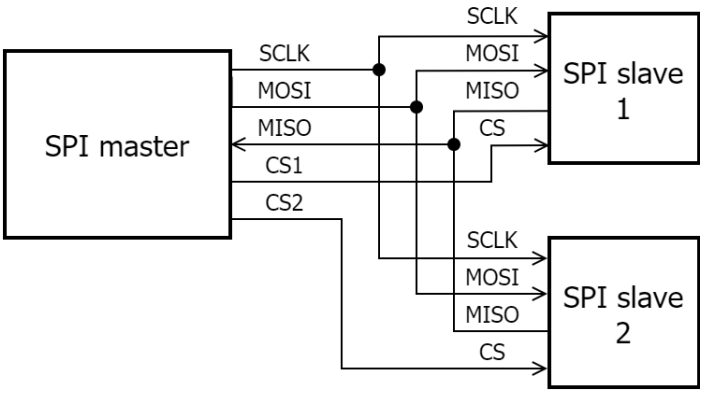
1.2.2 Asynchronní metoda

V případě asynchronní metody jsou data přenášena bez společného hodinového signálu a každé zařízení má svůj hodinový signál. Ze způsobu této metody je zřejmé, že aby přenos dat proběhl správně, tak obě zařízení musí být nakonfigurována stejně, především na stejnou přenosovou rychlost. Typickými představiteli jsou SCI nebo CAN [19].

1.3 Princip komunikace po sběrnici Serial Peripheral Interface - SPI

SPI pracuje v režimu synchronního přenosu dat. Data lze vysílat a přijímat v jeden okamžik díky tomu, že pracuje v režimu full duplex (plný duplex). SPI sběrnice obsahuje 4 vodiče - SCLK (hodinový signál, generuje Master), MISO (Master In Slave Out; data, které odesílá Slave do Master), MOSI (Master Out Slave In; data, která odesílá Master do Slave), CS (Chip Select, výběr aktivního zařízení). Pokud je potřeba ze zařízení pouze číst, nebo zapisovat, tak se lze setkat se zapojením, kdy jsou na sběrnici jen tři vodiče [10].

Využívá se pro celou řadu periferií jako např.: snímače teploty nebo tlaku, A/D převodníky, paměti typu Flash nebo EEPROM, LCD displeje, SD karty. Možné propojení SPI Master s SPI Slave je znázorněno na obr. 1.2.



Obr. 1.2: Ukázka SPI zapojení dvou Slave zařízení

1.3.1 Režimy komunikace SPI

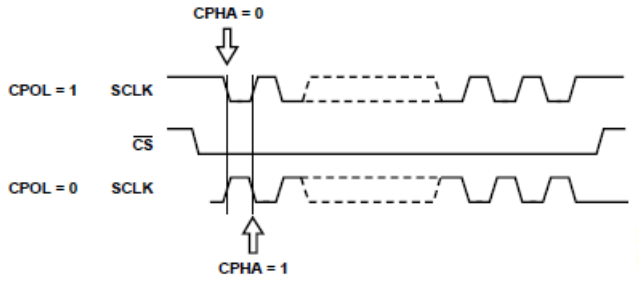
Aby mohla být správně přijata data je potřeba mít stanovený režim komunikace. SPI nemá pevně daný standard, a tak je důležité, aby Master a Slave byly nastaveni na stejný režim. SPI rozhraní definuje 4 režimy přenosu dat, viz. tab. 1.1. Tyto režimy rozlišují, při které polaritě a vzestupné nebo sestupné hraně hodinového signálu (SCLK) budou čtena data [10]. U Master zařízení lze často tyto režimy nastavit, ale u Slave zařízení je většinou k dispozici pouze jeden, ve kterém pracuje. Pro lepší názornost režimu nastavení SPI komunikace je uveden obr. 1.3.

Tab. 1.1: Režimy komunikace SPI

Režim	Polarita - CPOL	Fáze - CPHA
0	0	Náběžná hrana
1	0	Sestupná hrana
2	1	Sestupná hrana
3	1	Náběžná hrana

1.3.2 Průběh komunikace SPI

Master začíná přenos dat nastavením příslušného signálu Chip Select (CS) do log. 0 a vysíláním hodinového signálu (SCLK). Master i Slave data posílají vždy v reakci na polaritu a náběžnou/sestupnou hranu (záleží dle režimu komunikace, jak bylo zmíněno v kapitole 1.3.1). Master může kdykoliv začít komunikaci, protože ovládá hodinový signál SCLK a signál CS. Komunikace obvykle probíhá frekvencí v řádu



Obr. 1.3: Režimy výměny dat pomocí SPI [10]

.

několika MHz. Počet posílaných bytů není pevně stanoven a je možné přizpůsobit parametry podle aplikace [9].

1.4 Princip sériové komunikace - SCI

SCI je asynchronní sériová komunikace. Pracuje v režimu full duplex, kdy lze přenášet a přijímat data zároveň. Master i Slave mají svůj vlastní hodinový signál, a proto je nutné, aby byly nakonfigurovány na stejnou rychlost přenosu. Tento způsob komunikace patří mezi pomalejší a je určen spíše pro menší vzdálenosti. [16].

Sběrnice obsahuje 2 vodiče - TXD (odesílání) a RXD (příjem). Standardně se používá 8 nebo 9bitový formát dat. Rychlost komunikace se uvádí v baudech, což je počet přenášených bitů za sekundu. Používanými rychlostmi jsou 1200, 2400, 4800, 19200, 38400, 57600 a 115200 bps (baudů za sekundu).

1.4.1 Průběh komunikace SCI

Komunikace začíná nastavením START bitu do log. 0, pak následuje 8 nebo 9 bitů dat, tzv. datový rámec. Volitelně může být obsažen i paritní bit, což je bit pro kontrolu přijatých dat. Parita může být lichá nebo sudá, kdy určuje počet logických jedniček v datovém rámci. Parita se jednoduše vypočítá pomocí operace XOR mezi všemi bity datového rámce. Pak následuje STOP bit v hodnotě log. 1 a komunikace je ukončena. Pro správné proběhnutí komunikace je důležité, aby přijímač i vysílač byly ve stejném režimu.

2 Teoretický rozbor střídače a řídicí desky nadřazeného systému

V druhé kapitole jsou popsány vlastnosti střídačů a architektura nadřazeného systému, který slouží pro řízení buňkového měniče. Dále jsou uvedena bloková schéma pro obě rozšíření, které v této práci budou řešeny.

2.1 Základní definice měničů

Měniče slouží k přeměně parametrů elektrické energie. Takovými základními parametry mohou být velikost elektrického napětí, proudu nebo frekvence. Druhy měničů podle typu vstupního a výstupního signálu jsou uvedeny v tab. 2.1.

Tab. 2.1: Základní typy měničů [3]

Vstupní signál	Výstupní signál	Typ měniče
Střídavý	Střídavý	Střídavý měnič napětí
Střídavý	Stejnosměrný	Usměrňovač
Stejnosměrný	Střídavý	Střídač
Stejnosměrný	Stejnosměrný	Stejnosměrný pulzní měnič

Střídač je jeden z druhů měniče, který převádí vstupní stejnosměrné napětí na střídavé napětí. Obvykle používanými polovodičovými součástkami, ze kterých je střídač složen, jsou např.: IGBT tranzistory nebo IGCT případně GTO tyristory. Střídač může pracovat i v generátorovém režimu, kdy se energie zpět vrací do sítě. V praxi se toho využívá při přeměně z mechanické energie na elektrickou, např. jako elektrodynamická brzda [17].

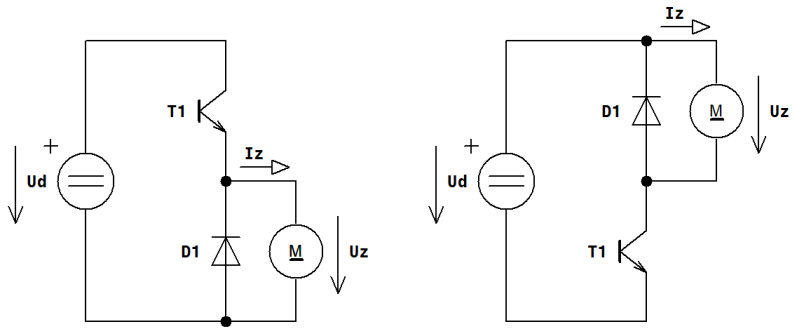
2.2 Konstrukce pulzních měničů

V této podkapitole jsou popsány konstrukce měničů pracující v jednom, dvou a čtyřech kvadrantech. Dělení na kvadranty vychází ze směru napětí na zátěži a směru proudu zátěží.

Konstrukce měniče pracujícího v jednom kvadrantu

Základním stavebním prvkem pulzního měniče je spínač. Ten je tvořen spínacím prvkem (např. tranzistorem nebo tyristorem) a nulovou diodou. Spínač lze realizovat dvěma způsoby, jako horní spínač nebo dolní spínač, viz obr. 2.1. Kombinací

těchto spínačů lze sestavit další typy měničů. Často vznikne antiparalelní zapojení tranzistoru a diody. Tranzistor se podílí na vedení proudu vždy s diodou, která je zapojena v sérii, a nikoliv s diodou antiparalelní [4].



Obr. 2.1: Jednokvadrantový pulzní měnič pracující v I. kvadrantu [4]

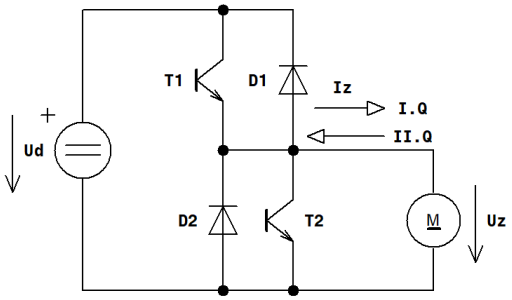
Měniče lze také dělit podle pracovních kvadrantů. Dělení vychází ze směru napětí, které měnič umí vygenerovat a směru proudu, který může téct zátěží. Pokud definujeme kladné napětí na motoru ve směru šipky U_z, tak na obrázku 2.1 můžeme vidět, že měnič umí vygenerovat sepnutím spínacího prvku kladné napětí a zátěží může téct pouze kladný proud ve směru šipky napětí I_z. Ten může téct buď tranzistorem, nebo nulovou diodou. Tento typ měniče pracuje pouze v prvním kvadrantu [4].

Konstrukce měniče pracujícího ve dvou kvadrantech

Na obrázku 2.2 je vidět paralelní kombinace výše uvedených spínačů. Tím lze vytvořit měnič, který umí vygenerovat pouze kladný směr napětí, ale proud dokáže vést oběma směry. V kladném směru (při sepnutí T1) teče proud přes tranzistor T1 do zátěže. V opačném směru proud teče přes diodu D1 a motor pracuje v generátorovém režimu, případně zátěž pracuje v generátorovém režimu při sepnutí tranzistoru T2 [4].

Konstrukce měniče pracujícího ve čtyřech kvadrantech

Kombinací 2 dvoukvadrantových měničů uvedených na obrázku 2.2 vznikne plný můstek, viz. obr. 2.3. Běžně se označuje jako H-můstek (H-Bridge). Takový měnič umí pracovat ve čtyřech kvadrantech, což znamená, že umí vygenerovat oba směry napětí a zátěží může téct proud oběma směry [4].



Obr. 2.2: Dvoukvadrantový pulzní měnič pracující v I. a II. kvadrantu [4]

2.3 Zátěže pulzních měničů

Na výstupu pulzního měniče může být zapojen LC filtr nebo motor. V obou případech se jedná o zátěž s výraznou induktivní složkou, což lze chápat jako proudově setrvačný prvek. Indukční zákon popisuje vztah mezi napětím a proudem na indukčnosti. V diferenciálním tvaru je vyjádřen následujícím vztahem:

$$u(t) = L\frac{di(t)}{dt} \tag{2.1}$$

Vztah pro výpočet proudu by v integrálním tvaru vypadal následovně:

$$i(t) = I_0 + \frac{1}{L}\int u(t) \cdot dt \tag{2.2}$$

Ze vztahu je dáno, že proud induktivní zátěží je dán integrálem napětí. To lze chápat jako setrvačnost. Hodnota proudu je úměrná časovému integrálu napětí a roste s tím, jak se integruje plocha pod signálem napětí s časem.

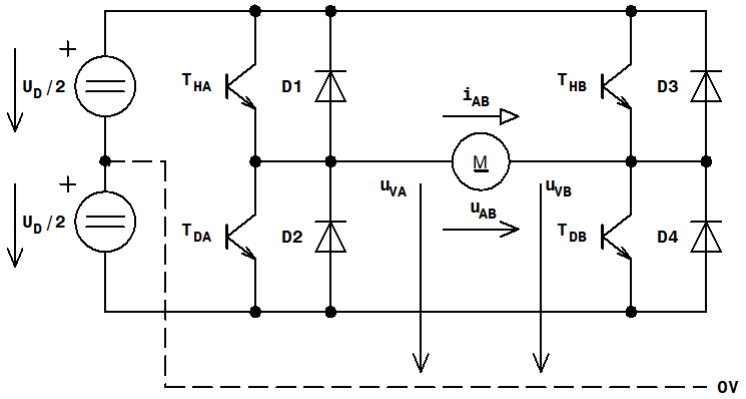
Z diferenciálního tvaru je patrné, že pokud se skokově změní proud tekoucí induktivní zátěží, tak k tomu bude potřeba napětí o nekonečné velikosti. Pro analýzu měničů je důležitý průběh proudu induktivní zátěží, kdy se střídavě připojuje ke konstantnímu napětí, které ale mění polaritu. Vztah může být vyjádřen následovně:

$$i(t) = I_0 \pm \frac{1}{L}\int U_{konst} \cdot dt = I_0 \pm \frac{U_{konst}}{L}\int 1 \cdot dt = I_0 \pm \frac{U_{konst}}{L}t \tag{2.3}$$

Ze vztahu (2.3) vyplývá, že se jedná o rovnici přímky. Lze tedy říct, že proud lineárně narůstá a klesá, což způsobuje pilovité zvlnění proudu na zátěži [5].

2.4 Řízení pulzních měničů

V tomto textu jsou popsány bipolární a unipolární způsoby řízení pulzních měničů. Na obr. 2.3 je znázorněn H-můstek s popisky prvků obvodu, na kterých budou popsány řídicí signály a napěťové výstupy.

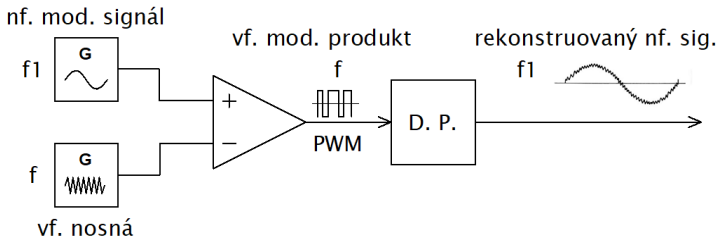


Obr. 2.3: Čtyřkvadrantový pulzní měnič [4]

2.4.1 PWM pro řízení pulzních měničů

V měničích se PWM signálem ovládají výkonové spínací prvky. Nejjednodušší PWM modulátor, znázorněn na obr. 2.4, je v analogové implementaci tvořen komparátorem, na jehož vstup je přiveden trojúhelníkový nosný signál o kmitočtu f, na druhý vstup je přiveden nízkofrekvenční (nf) modulační signál o kmitočtu f_1 Na výstupu komparátoru je potom vysokofrekvenční (vf) modulační produkt. K demodulaci vf modulačního produktu je potom použit filtr typu dolní propust. Nízkofrekvenční modulační signál může mít makroskopicky podobu stejnosměrného napětí v případě stejnosměrného pulzního měniče, nebo sinusového napětí o frekvenci f_1 v případě střídače. Pro kvalitní demodulaci je nutné, aby byla dobře splněna následující nerovnost (2.4), kde f_z je zlomový kmitočet filtru dolní propusti [4].

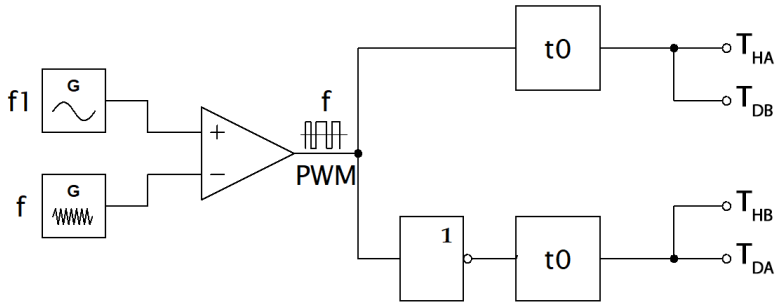
$$f_1 < f_z << f \tag{2.4}$$



Obr. 2.4: Analogová implementace PWM modulátoru [4]

2.4.2 Bipolární řízení čtyřkvadrantového pulzního měniče

V případě bipolárního řízení je PWM modulátor pro tento měnič tvořen komparátorem. Za komparátorem je ve spodní větvi PWM signálu zapojen invertor. Následuje blok, který zajistí vložení Dead Time, což je blok, který zajišťuje časovou prodlevu mezi vypnutím tranzistoru T_{HA} a zapnutím tranzistoru T_{HB} v rámci jedné větve. Časová prodleva, která je odlišná při zapnutí a vypnutí, je nutná, protože tranzistory mají určitou časovou prodlevu při zapnutí nebo vypnutí. V případě, že by tato prodleva nebyla zařazena, docházelo by při přepínání tranzistorů krátkodobě ke zkratu mezilehlého napětí U_d znázorněném na obr. 2.3. Pro moderní polovodiče na bázi karbidu křemíku, viz kapitola 2.5, bývá Dead Time v řádu stovek ns, v případě křemíkových tranzistorů IGBT bývá Dead time v řádu jednotek µs. Implementace řídicího obvodu je znázorněna na obr. 2.5.



Obr. 2.5: Řídicí obvody H-můstku - bipolární řízení [4]

Signál pro sepnutí tranzistorů je aktivní v log. 1. To, že je připojen aktivní signál na řídicí elektrodu ještě neznamená, že tranzistorem poteče proud. Tranzistor (bipolární, IGBT) vede proud pouze ve směru kolektor-emitor. Pokud by měl proud téct jiným směrem, povede ho protilehlá dioda vedlejšího spínače.

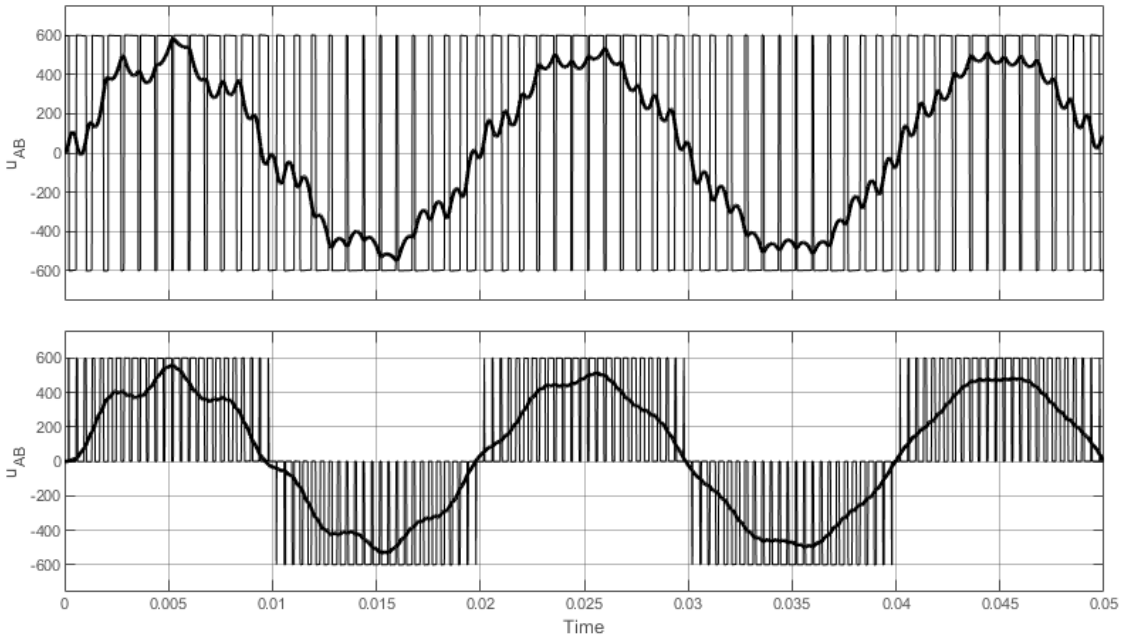
Podle obr. 2.3 bude výstupní napětí u_{AB} nabývat střídavě hodnoty $\pm U_D$. Průběh napětí u_{AB} je znázorněn v horní části obr. 2.6. Pro vygenerování napětí se střední hodnotou rovnou 0, by se musely střídat impulsy o amplitudě $\pm U_D$ se střídou 0,5 [4].

2.4.3 Unipolární řízení čtyřkvadrantového pulzního měniče

Hlavní rozdíl oproti bipolárnímu řízení spočívá v tom, že pro generování řídicích signálů jsou použity dva komparátory. Vstupy jsou připojeny na společný trojúhelníkový nosný vf. signál. Nízkofrekvenční modulační signály jsou dva a jsou mezi sebou vzájemně posunuty o 180°. Výstupy komparátoru jsou rozvětveny a stejným

způsobem, jako v případě bipolárního řízení, přivedeny na řídicí elektrody tranzistorů. To znamená, že všem signálům z komparátoru je přidán Dead Time a vždy jeden ze signálů komparátoru je invertován. Narozdíl od řízení bipolárního nejsou přepínany všechny spínače najednou, ale vždy jen dva v rámci větve. Podle obr. 2.3 bude výstupní napětí u_{AB} nabývat hodnot $+U_D$ a 0, případně $-U_D$ a 0 [4].

Výstupní napětí u_{AB} v případě unipolárního řízení je patrné ze spodní části obr. 2.6. Tučnou čarou je na obr. 2.6 znázorněn průběh napětí na zátěži typu LC filtr.



Obr. 2.6: Bipolární a unipolární řízení - průběhy výstupního napětí

2.5 Popis bipolárního tranzistoru s izolovaným hradlem

Insulated Gate Bipolar Transistor (IGBT) je součástka, kterou si lze představit jako bipolární tranzistor, která má izolované hradlo. Je kombinací tranzistorů typu Metal Oxide Semiconductor Field Effect Transistor (MOSFET) a bipolárního tranzistoru (BJT). Obsahuje čtyřvrstvou strukturu P-N-P-N, což je topologicky stejná struktura jako u MOS tyristorů. Je konstruován pro velký rozsah spínacích výkonů. Výhodami jsou malé ztráty v sepnutém stavu (vysoká účinnost), nízký budící výkon a větší rozsah pracovního napětí nebo proudu, než u tranzistorů typu MOSFET. Byl vyvinut z důvodu potřeby rychlejšího spínání, které je u bipolárních tranzistorů nebo tyristorů nedostatečné. Používají se ve výkonové elektronice, např. v měničích pro napájení

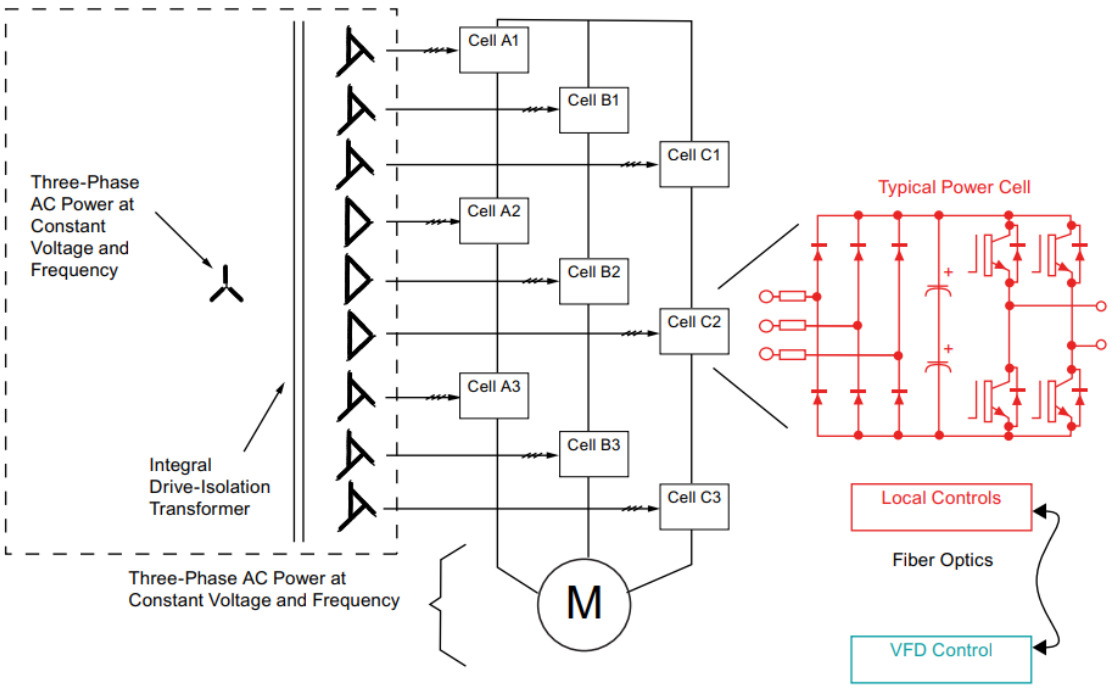
střídavých nebo stejnosměrných elektrických pohonů (elektrické lokomotivy nebo tramvaje) [13].

2.6 Zapojení měničů dle topologie

V podkapitole je popsáno zapojení několika dílčích střídačů dle topologie. Dále je uvedeno jaké výhody z takového zapojení vyplývají.

2.6.1 Topologie Robicon Perfect Harmony

Topologie Robicon Perfect Harmony je zapojení několika dílčích měničů (buněk) sériově. Buňka je jednofázový čtyřkvadrantový střídač tvořený 2 páry IGBT tranzistorů. Hodnota výsledného napětí se získá sečtením napětí na všech buňkách. Na obr. 2.7 je uvedeno typické zapojení tří buněk pro třífázovou soustavu. Každá buňka je napájena svým izolovaným sekundárním vinutím z transformátoru. V tomto případě je buněk 9, tudíž každá je zatížena devítinou výstupního výkonu. Pro vyšší výstupní napětí je potřeba zvýšit počet buněk a sekundárních vinutí u transformátoru. Vývod gate IGBT tranzistorů buněk je propojen pomocí optických kabelů a ovládán z řídicího systému [7].



Obr. 2.7: Topologie Robicon Perfect Harmony [7]

2.7 Popis nadřazeného řídicího systému

V této kapitole je popsána základní deska ELDE20 a definováno blokové schéma výsledného řešení. Deska je vyvíjena firmou ELCOM, a. s. Je určena a přizpůsobena pro řízení měničů.

2.7.1 Základní deska ELDE20

Základní deska ELDE20 je založena na signálovém procesoru TMS320F28335, který byl vyvinut společností Texas Instruments. Deska obsahuje dva sloty pro rozšiřující moduly, kterými mohou být EtherCAT Slave, CAN-A nebo SCI-B.

Analogové vstupy

Deska obsahuje vstupy: 2x (0 až 3 V), jeden z nich se používá pro měření teplotním čidlem KTY83; 5x (-5 až +5 V) (R_i = 100 kΩ); 5x -10 až 10 V (R_i = 100 kΩ). Všechny analogové vstupy jsou bez galvanického oddělení. Na každém vstupu je zapojen 12bitový převodník.

Digitální vstupy

Obsahuje digitální vstupy na 24 V s galvanickým oddělením pomocí Schmittova hradla. Deska používá: 4x standardní vstupy (standardní optočlen pro galvanické oddělení); 2x rychlé vstupy (rychlý optočlen pro galvanické oddělení)

Digitální výstupy

Obsahuje digitální výstupy na 24 V s galvanickým oddělením. Deska obsahuje: 4x standardní výstupy (standardní optočlen pro galvanické oddělení); 2x rychlé výstupy (rychlý optočlen pro galvanické oddělení). Maximální proud výstupu je 100 mA.

Reléové výstupy

Deska obsahuje 6 reléových spínacích kontaktů 6A/230V AC.

PWM výstupy

Deska obsahuje dvě kompletní jednotky pro řízení třífázového můstku. Každá jednotka obsahuje: 6 PWM výstupů (TTL, oddělené výkonovým hradlem), 6 FAULT vstupů (signály pro Trip Zone TZ1-TZ6). Obsahují hardwarové blokování signálem FAULT. Výstupní napětí jsou +15 V a +24 V.

RS422/485

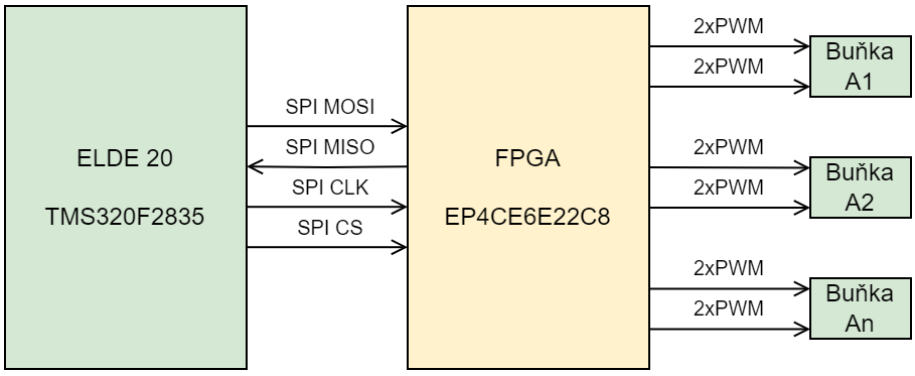
Deska obsahuje dvě nezávislé komunikační linky SCI-A a SCI-C. Přepínání směru linky SCI-A mezi signálovým procesorem a modulem Ethernetu je realizováno pomocí signálu z modulu Ethernetu.

2.8 Blokové schéma rozšíření řídicího systému

Cílem práce je navrhnout rozšíření řídicího systému se signálovým procesorem pro řízení buňkového měniče. Rozšíření se skládá z desky s hradlovým polem připojené k řídicímu systému a modulů připojených ke střídačům buňkového měniče. V této kapitole jsou stručně popsány obě rozšíření desky popisované v předchozí podkapitole a definována jejich bloková schémata.

2.8.1 Rozšíření řídicí desky s nastavitelnými PWM modulátory pomocí SPI

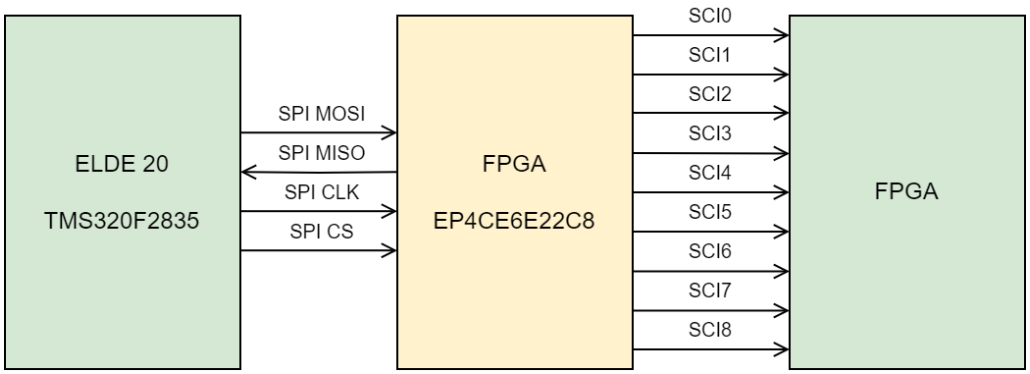
Cílem této práce je navrhnout rozšíření řídicí desky spočívající v návrhu PWM modulu. Původní řídicí deska totiž neobsahuje dostatečný počet PWM kanálů. Rozšiřující modul bude obsahovat několik PWM kanálů, které budou nastavitelné pomocí rozhraní SPI. Počet PWM kanálů bude genericky definovatelný. Rozšíření se bude chovat jako zařízení typu SPI Slave. Blokové schéma propojení hradlového pole s nadřazenou deskou je na obr. 2.8.



Obr. 2.8: Blokové schéma rozšíření řídicí desky s PWM modulátory

2.8.2 Rozšíření řídicí desky s devíti sériovými linkami ovládaných po SPI

Další rozšíření, které je realizováno v této práci spočívá v implementaci sériových linek do hradlového pole. V tomto případě jsou výstupem regulační smyčky realizované v systému s digitálním signálovým kontrolérem (DSC) žádané šířky pulsů, které jsou přes SPI předávány do hradlového pole. V hradlovém poli je realizováno 9 sériových linek prostřednictvím kterých jsou distribuovány požadované šířky pulsů do jednotlivých buněk s fázovým přesazením. Blokové schéma propojení hradlového pole s nadřazenou deskou je na obr. 2.9.



Obr. 2.9: Blokové schéma rozšíření řídicí desky se sériovými linkami

3 Prostředky a nástroje pro realizaci práce

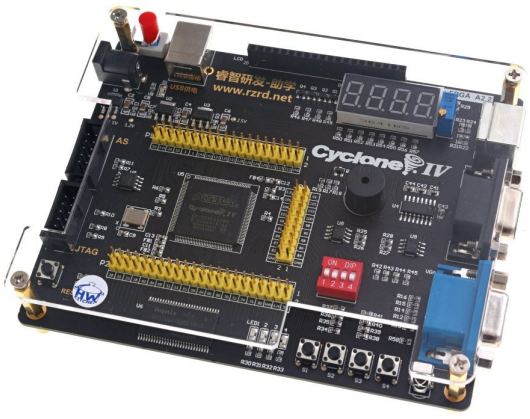
V této kapitole je popsán zvolený vývojový kit, podpůrné prostředky pro vývoj a proces návrhu obvodu v hradlovém poli.

3.1 Vývojový kit FPGA s obvodem EP4CE6E22C8

Pro tuto práci byl zvolen vývojový kit s FPGA obvodem EP4CE6E22C8. Deska pracuje na kmitočtu 50 MHz a napájí se buď přes Universal Serial Bus (USB), nebo přes napájecí konektor. Nahrávání konfigurace do FPGA probíhá přes rozhraní JTAG (Joint Test Action Group). Kit obsahuje mnoho periferií, jako například paměť typu SDRAM a EEPROM, rozhraní RS232 či sedmisegmentový displej [12]. Vývojový kit je znázorněn na obr. 3.1.

ALTERA Cyclone IV - EP4CE6E22C8

Tento obvod je programovatelné hradlové pole od společnosti Intel (dříve Altera) a patří do rodiny Cyclone IV E. Vstupně-výstupní piny lze nakonfigurovat do několika napěťových úrovní, v práci je používána standardní 3,3 V logika. Parametry tohoto FPGA jsou uvedeny v tab. 3.1.



Obr. 3.1: FPGA kit s Altera Cyclone IV [12]

Tab. 3.1: Parametry hradlového pole EP4CE6E22C8 [6]

Logické bloky	6272
M9K paměťové bloky	30
Vestavěná paměť (Kbits)	270
18 bitová násobička	15
PLLs	2
Definovatelné I/O piny	179
Diferenciální vstupy	66

3.2 Výukový modul ADALM2000

Tento modul je výukové zařízení, které obsahuje mnoho užitečných periferií a funkcionalit, které mohou být užitečné při vývoji. Součástí je i software Scopy, který komunikuje pomocí USB s tímto modulem a uživateli poskytuje komplexní software pro testování a vývoj. Hlavní součástí, které bude využíváno, je 16 kanálový logický analyzátor, který dokáže vzorkovat až s frekvencí 100 MSPS (mega samples per second). Součástí je 2 kanálový osciloskop s diferenčními vstupy, 2 funkční generátory atd. Modul lze nakonfigurovat jako zařízení pro komunikace: Serial Peripheral Interface, Two Wire Interface, Universal Asynchronous Receiver-Transmitter, Paralelní. Těchto komunikací bude využíváno při řešení úkolů této práce [11].

3.3 Vybraný software pro vývoj

Vývojovým prostředím (IDE) pro FPGA od firmy Altera je Intel Quartus Prime ve verzi 20.1. IDE je výchozí a doporučené pro popsaný FPGA obvod od téhož výrobce. Syntéza a časové analýzy entit popisovaných dále v této práci byly prováděny tímto vývojovým prostředím. Součástí IDE je i nadstavba Questa pro Register-Transfer Level (RTL) nebo Gate-Level simulace, které jsou nezbytné pro testování a ověření správnosti napsaného kódu. Jazykem pro popis hardware může být VHSIC (Very High Speed Integrated Circuit Program) Hardware Description Language (VHDL) nebo jazyk Verilog. Verilog se více podobá klasickému programování (jazyku C). Pro tuto práci byl zvolen jazyk VHDL, protože je více používaný v Evropě.

3.4 Průběh návrhu obvodu v FPGA

Pro správnou a robustní realizaci komponenty v hradlovém poli je dobré si promyslet jednotlivé kroky realizace. Prvním krokem je správná specifikace zadání, kde je dána funkce navrhovaného obvodu. Spolu s tím musí být známé i ostatní provozní požadavky jako jsou: teplotní rozsah nebo pracovní kmitočet.

V dalším kroku se navrhovaný systém rozdělí do funkčních bloků, kde typicky každý blok plní funkci, která je odlišná od ostatních. Tímto způsobem lze docílit dobře srozumitelného, jednoduše modifikovatelného a spravovatelného kódu. Pro popis takového kódu se použije jeden z jazyků pro popis hardware, tedy VHDL, Verilog, SystemC, SystemVerilog atd. Dále je vhodné mít pro každý takto napsaný blok svůj testovací kód - Testbench. Testovat celý koncept (několik propojených funkčních bloků mezi sebou) je vhodné až po otestování každého bloku samostatně. Často jsou tyto dvě činnosti prováděny oddělenými vývojovými týmy, čímž se docílí určité nezávislosti při testování návrhu. Každý tým může na problém pohlížet trochu jinak a lze tak odhalit např. špatné pochopení zadání. Pokud jsou funkční simulace otestovány a navržený koncept se chová podle očekávání, tak lze přejít k syntéze kódu, jinak je třeba upravit původní kód.

Při syntéze se transformuje RTL kód do vzájemného propojení logických bloků hradlového pole. Z toho vyplývá, že je potřeba nastavit požadavky na syntézu, čímž se rozumí časové parametry neboli Constraints, což je přiřazení signálů na vývody pouzdra FPGA. Výsledkem úspěšné syntézy je Netlist, což je soubor obsahující propojení logických bloků.

Dalším krokem je rozmístění a propojení. V rámci této operace se provádí optimalizace celé logiky a rozmístění prvků v FPGA. Dále se bloky v FPGA spolu propojí.

Nyní je na řadě časová simulace. Ta díky zadaným frekvencím vnitřních signálů (většinou hodinového signálu) určí, zda-li jsou splněny požadavky na řešení. Výstupy z časové simulace jsou přesné, protože je již dáno rozmístění logických bloků v FPGA a na základě takových informací lze přesně určit hodnoty časových zpoždění. Pokud taková časová simulace dopadne správně, tak lze s velkou pravděpodobností předpokládat, že výsledný koncept bude fungovat i v reálném FPGA [2].

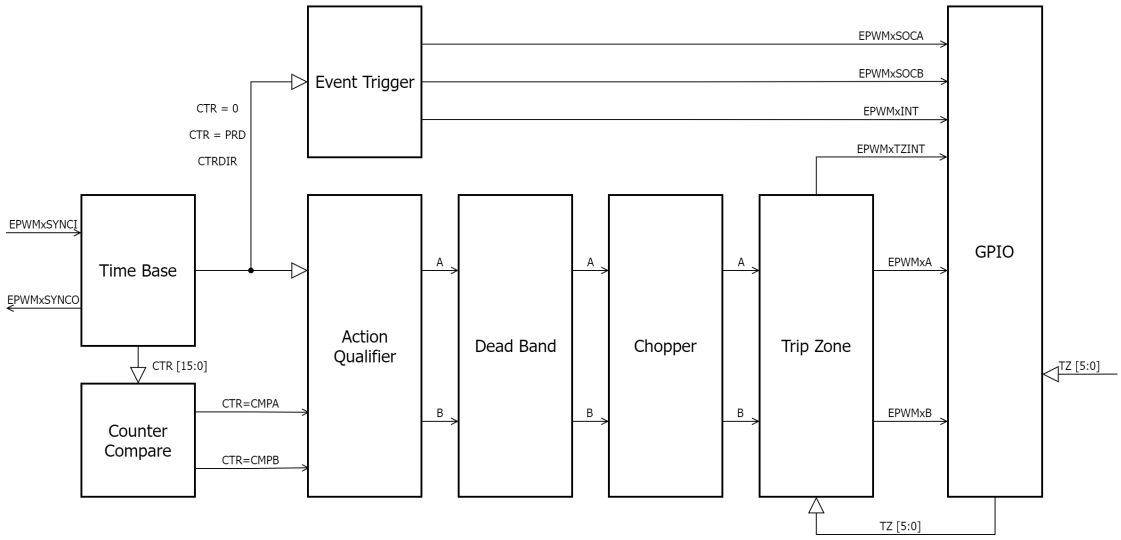
Výsledky z navrženého modulu mohou být počty logických bloků, registrů, potřebných vstupně-výstupních pinů nebo maximální frekvence hodinového signálu. Tyto výsledky jsou uváděny v následujících kapitolách o implementaci modulů ve VHDL.

4 Implementace ePWM ve VHDL

Ve čtvrté kapitole je popsán modul ePWM a definováno vnitřní blokové schéma. Jsou vysvětleny funkce jednotlivých bloků a činnosti registrů. Dále je uvedena analýza a test implementace.

4.1 Blokové schéma ePWM

Celý PWM modul se skládá z několika dílčích bloků, které jsou mezi sebou propojeny a každý blok plní svojí funkci. Implementace v jazyce VHDL je provedena obdobně, každý blok funguje jako samostatný celek a nezávisle na ostatních blocích. V jazyce VHDL se takové bloky nazývají Entity. Díky tomu lze Entity jednoduše později modifikovat a celý kód se stává přehlednějším. Tento modul vychází z koncepce ePWM periferie, která je implementovaná na čipu TMS320F28335. Blokové schéma znázorňující propojení bloků ePWM modulu je uvedeno na obr. 4.1.



Obr. 4.1: Blokové schéma ePWM [8]

4.2 Implementace ePWM do hradlového pole

V podkapitole jsou popsány jednotlivé bloky ePWM modulu, popis jejich funkce a registrů.

4.2.1 Blok Time Base

Blok Time Base slouží pro časování ePWM a obsahuje logiku, díky které lze několik ePWM modulů synchronizovat.

Popis funkce

V tomto bloku se nachází 16bitový čítač, dělička kmitočtu a komparátor. Čítač může pracovat ve třech režimech: `UP, DOWN, UP-DOWN`. Čítá do hodnoty definované v registru `TBPRD`. Komparátor porovnává hodnotu v aktivním registru `TBPRD` s aktuální hodnotou čítače v registru `TBCTR`, v případě shody výstup `CTR=PRD` přejde do 1 a dojde k resetování čítače. Čítač obsahuje výstupní signály `CTR=ZERO`, `CTRDIR`, `CTRMAX`.

V bloku se dále nachází dělička kmitočtu. Vstupním signálem je `SYSCLKOUT` a výstupním signálem `TBCLK`. Dělící poměr se nastavuje pomocí registrů `HSPCLKDIV` a `CLKDIV`. Výstupní kmitočet se dále používá jako hodinový signál celého obvodu.

Synchronizační výstup `EPWMxSYNCO` může pracovat v několika režimech, které se nastavují podle registru `SYNCOSEL`. Podle nastavení multiplexoru se na výstupu mohou objevit signály: `EPWMxSYNCI, CTR=ZERO, CTR=CMPB` nebo výstup může být zcela zakázaný.

Synchronizační vstup `EPWMxSYNCI` slouží pro nastavení hodnoty čítače z registru `TBPHS` (slouží pro posun fáze výstupních PWM signálů), jestliže je `EPWMxSYNCI=1` a `PHSEN=1` dojde k přepsání aktuální hodnoty čítače, tedy registru `TBCTR`.

Synchronization Select (`SYNCOSEL`) [1:0]

Bity nastavují multiplexor na jehož výstupu je připojen signál `EPWMxSYNCO`.

Counter Mode (`CTRMODE`) [1:0]

Bity nastavují čítací režim čítače.

Clock Divider, High Speed Clock Divider (`CLKDIV, HSPCLKDIV`) [2:0], [1:0]

Bity nastavují hodnotu dělení kmitočtu.

Period Load (`PRDLD`)

Bit aktivuje načítání periody čítače ze stínového (Shadow) registru.

Phase Enable (`PHSEN`)

Bit povoluje nastavení fáze.

Counter Direction (CTRDIR)

Bit je pro čtení a obsahuje informaci o směru čítání.

Time Base Phase (TBPHS) [15:0]

Bity nastavují fázové posunutí výstupního signálu PWM.

Time Base Counter (TBCTR) [15:0]

V registru je aktuální hodnota čítače. V případě zápisu do toho registru se aktuální hodnota přepíše a čítač přičítá/odečítá od této hodnoty.

Time Base Period (TBPRD) [15:0]

Bity nastavují periodu výstupního signálu PWM, mohou pracovat ve stínovém a aktivním (Active) režimu. Komparátor používá vždy aktivní režim registru. K přepsání dochází, když se hodnota čítače rovná 0. V případě, že bity PRDLD jsou nastaveny na 1, tak je tento režim vypnut a zapsaná hodnota se hned projeví v aktivním režimu.

Counter Max (CTRMAX)

Bit je pro čtení a obsahuje informaci o stavu TBCTR=0xFFFF

4.2.2 Blok Counter Compare

Blok Counter Compare slouží pro porovnávání aktuální hodnoty čítače s nastavenými hodnotami v příslušných registrech.

Popis funkce

V tomto bloku jsou implementovány dva číslicové komparátory. Společným vstupem do komparátorů je signál TBCTR (aktuální hodnota čítače v bloku Time base), který se porovnává s registry CMPA a CMPB. V případě shody se na výstupech CTR=CMPA nebo CTR=CMPB objeví log. 1.

LOADAMODE a LOADBMODE [1:0]

Těmito daty se ovládá multiplexor, který určuje, kdy přejde hodnota CMPA (LOADAMODE) nebo CMPB (LOADBMODE) z Shadow registru do Active registru.

Compare A a Compare B (CMPA a CMPB) [15:0]

Jsou dvě 16bitové hodnoty, kterými se nastavují hodnoty obou komparátorů - CMPA a CMPB.

4.2.3 Blok Action Qualifier

Blok Action Qualifier generuje 2 PWM signály na základě nastavení svých registrů a na základě vstupních událostí.

Popis funkce

Vstupy jsou signály z bloku Time base (CTR=PRD, CTR=0, CTRDIR) a signály z komparátoru (CTR=CMPA, CTR=CMPB). K dispozici je tedy celkem 6 událostí na které lze reagovat, protože se rozlišuje směr čítání. Těchto šest nastavení se provádí pro výstup A a B zvlášť.

CBD [1:0]

Akce, která se má provést, když TBCTR=CMPB a čítač čítá dolů.

CBU [1:0]

Akce, která se má provést, když TBCTR=CMPB a čítač čítá nahoru.

CAD [1:0]

Akce, která se má provést, když TBCTR=CMPA a čítač čítá dolů.

CAU [1:0]

Akce, která se má provést, když TBCTR=CMPA a čítač čítá nahoru.

PRD [1:0]

Akce, která se má provést, když TBCTR=TBPRD.

ZRO [1:0]

Akce, která se má provést, když TBCTR=0.

4.2.4 Blok Dead Band

Blok Dead Band může generovat Dead Time výstupního signálu.

Popis funkce

Vstupnímu signálu lze přiřadit Dead Time s náběžnou, nebo sestupnou hranou signálu o definovaný čas. Signály lze invertovat, případně ponechat beze změny.

K dispozici jsou 10bitové hodnoty `DBRED` a `DBFED`. Tato data definují maximální hodnotu 10bitového čítače, kterým se generuje Dead Time. Dead Time je generován tak, že pokud vstupní signál změní svoji log. úroveň, tak se spustí čítač. V případě, že je aktivní (hodnota čítače nedosáhla maximální hodnoty uložené v registru), tak signál zůstává ve své minulé log. úrovni, do nové log. úrovně přejde v okamžiku, když se bude hodnota čítače rovnat maximální hodnotě uložené v registru `DBRED` případně `DBFED`.

`INMODE` [1:0]

Tyto 2 bity nastavují vstupní signály, které mají být zpožděny.

`POLSEL` [1:0]

Bity se nastavuje polarita výstupních signálů. Výstupní signály jsou uvedeny jako `EPWMxRED` a `EPWMxFED`.

`OUTMODE` [1:0]

Bity se nastavují vstupní signály pro výstupní signály z bloku. Těmito bity lze ignorovat funkci tohoto bloku (tzv. bypass).

`DBRED` [9:0]

Bity nastavují hodnotu Dead Time při náběžné hraně signálu.

`DBFED` [9:0]

Bity nastavují hodnotu Dead Time při sestupné hraně signálu.

4.2.5 Blok Chopper

Blok Chopper může vkládat do signálu signál o určité nosné frekvenci. Takto modulovaný signál je důležitý, pokud se používají hradlové budiče založené na pulzním transformátoru, kterými se ovládají výkonové spínací prvky [8].

Vstupní a výstupní signály z bloku jsou patrné na obr. 4.2.

Popis funkce

Pokud je blok aktivní, tak se na výstupu PWM signálu z bloku Chopper objeví PWM signál. První puls má nastavitelnou šířku, ostatním pulsům lze nastavit střídu a frekvenci. Pro lepší názornost je uveden obr. 4.2, kde jsou znázorněny vstupující a vystupující signály z bloku.

Blok se řídí systémovým hodinovým signálem, který musí mít frekvenci alespoň 8x větší, než je frekvence vstupního pulzně modulovaného signálu. Tento vstupní signál se po prvním pulsu synchronizuje s nastavenou frekvencí a střídou.

`OSHTWTH` [3:0]

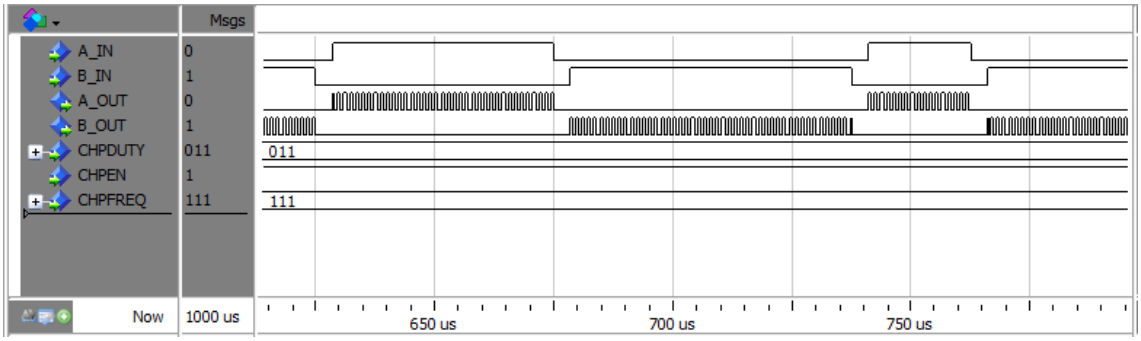
Pomocí těchto 4 bitů se nastavuje šířka prvního pulsu.

`CHPFREQ` [2:0]

Pomocí těchto 3 bitů se nastavuje frekvence modulovaného signálu. Hodnota reprezentuje dělicí poměr.

`CHPDUTY` [2:0]

Pomocí těchto 3 bitů se nastavuje střída modulovaného signálu. Hodnota v registru reprezentuje procentuální zastoupení log. 1.



Obr. 4.2: Ukázka vstupního a výstupního signálu z bloku Chopper

4.2.6 Blok Trip Zone

Blok Trip Zone slouží pro vykonání definované akce, pokud se na jednom ze 6 vstupů do bloku objeví log. 1.

Popis funkce

Vstupem do tohoto bloku je 6 signálů - `TZ`, které jsou podle potřeby zapojeny tak, aby detekovaly poruchu. V případě, že některý ze signálů signalizuje poruchu, dojde k definovanému stavu na PWM výstupech. Blok obsahuje dva režimy chování, kterými jsou Cycle-by-Cycle(CBC) a One-Shot(OSHT). CBC režim se resetuje po každé, když `TBCTR=0`. OSHT režim se musí resetovat pomocí registru `TZCLR[OST]`, jinak jsou výstupy stále ve stavu poruchy. V případě, že nastane porucha, tak blok podle konfigurace generuje přerušení `EPWMxTZINT`.

`CBC` [5:0] a `OSHT` [5:0]

Tyto bity definují, které vstupní signály `TZ` budou zdrojem pro CBC a OSHT režim. Jestliže je bit v log. 1, tak se vstup použije, jinak je ignorován.

`TZA` a `TZB` [1:0]

Tyto 2 bity nastavují výstup v režimu poruchy.

`TZEINT(OST, CBC)`

Tyto bity povolují přerušení. Při nastavení log. 1 se generuje přerušení.

`TZFLG(CBC, INT)`

Tyto bity jsou pouze pro čtení. CBC bit se nastaví na log. 1, když nastane událost v CBC režimu. `INT` bit zůstává v log. 1 po vygenerování přerušení, dokud není resetován.

`TZCLR(OST, CBC, INT)`

Tyto bity jsou pro resetování OST režimu.

4.2.7 Blok Event Trigger

Blok Event Trigger slouží jako generátor synchronizačních pulsů pro A/D převodník a jako generátor přerušení.

Popis funkce

Obsahuje 5 vstupních signálů (událostí) a 3 výstupní signály - dva synchronizační (`EPWMxSOCA` a `EPWMxSOCB`) a jedno přerušení (`EPWMxINT`). K vygenerování pulsu dojde na základě vstupní události. Vstupními událostmi jsou signály čítače z bloku Time Base a signály komparátoru z bloku Counter Compare.

`SOCEN(A,B)` a `INTEN`

Tento bit při nastavení log. 1 povoluje výstupní synchronizační puls `EPWMxSOCA`, `EPWMxSOCB` (v případě `SOCEN`) a `EPWMxINT` (v případě `INTEN`).

`SOCSEL(A,B)` a `INTSEL [2:0]`

Tyto 3 bity ovládají multiplexor, který určuje událost (signál), která zapříčiní vygenerování synchronizačního pulsu (`SOCSEL`) nebo pulsu přerušení (`INTSEL`).

`SOCCNT(A,B` a `INTCNT [1:0]`

Tyto dva bity indikují počet událostí, které se zaznamenaly. Hodnota přímo odpovídá počtu událostí.

`SOCPRD(A,B)` a `INTPRD` [1:0]

Nastavením těchto dvou bitů se určí, při kolikáté zachycené události (určuje `SOCCNT`) dojde ke generování synchronizačního pulsu. Hodnota přímo odpovídá číslu události.

`SOC(A,B)` a `INT`

Tento bit je pouze pro čtení a indikuje generování synchronizačního nebo přerušovacího pulsu.

4.3 Shrnutí důležitých funkcí periferie ePWM

Ve čtvrté kapitole byly popsány registry ePWM. V této podkapitole jsou shrnuty důležité registry, které jsou nejčastěji využívány při nastavování periferie. Blok Time Base slouží pro nastavení časové základny, tzn. periody signálu (registr `TBPRD`), směru čítání čítače (registr `CTRMODE`) a nastavení fázového posunu (registr `TBPHS`). Blok Counter Compare slouží jako komparátor aktuální hodnoty čítače s hodnotami v registrech `CMPA` a `CMPB`, čímž se nastavuje střída signálu. Blok Action Qualifier slouží pro generování určité hodnoty signálu na základě signálů z Counter Compare a Time Base. Blok Dead Band slouží pro generování Dead Time výstupního signálu. Registry, které definují délku Dead Time jsou `DBRED` a `DBFED`. Blok Chopper slouží pro vložení signálu o definované nosné frekvenci a střídě. Frekvence se definuje registrem `CHPFREQ` a střída pak registrem `CHPDUTY`. Blok Trip Zone slouží pro odstavení výstupu na základě 6 vstupních signálů. Event Trigger slouží jako generátor synchronizačních impulsů a jako generátor přerušení. Přehledné zobrazení používaných registrů je v tab. 4.1.

Tab. 4.1: Shrnutí používaných registrů ePWM

Název registru	Funkce	Šířka v bitech
`TBCTL`	Nastavení časové základny	[15:0]
`TBPHS`	Nastavení fáze	[15:0]
`TBCTR`	Aktuální hodnota čítače	[15:0]
`TBPRD`	Nastavení periody	[15:0]
`CMPCTL`	Nastavení komparátoru	[5:0]
`CMPA`	Hodnota v komparátoru A	[15:0]
`CMPB`	Hodnota v komparátoru B	[15:0]
`AQCTLA`	Nastavení signálu A (generování PWM)	[11:0]
`AQCTLB`	Nastavení signálu B (generování PWM)	[11:0]
`DBCTL`	Nastavení bloku pro Dead Time	[5:0]
`DBRED`	Nastavení Dead Time náběžné hrany	[9:0]
`DBFED`	Nastavení Dead Time sestupné hrany	[9:0]

4.4 Analýza implementace periferie ePWM

Pro plnohodnotnou implementaci (lze konfigurovat paralelně všechny registry) je potřeba 668 logických bloků, 181 registrů a 415 pinů.

4.5 Test implementace ePWM

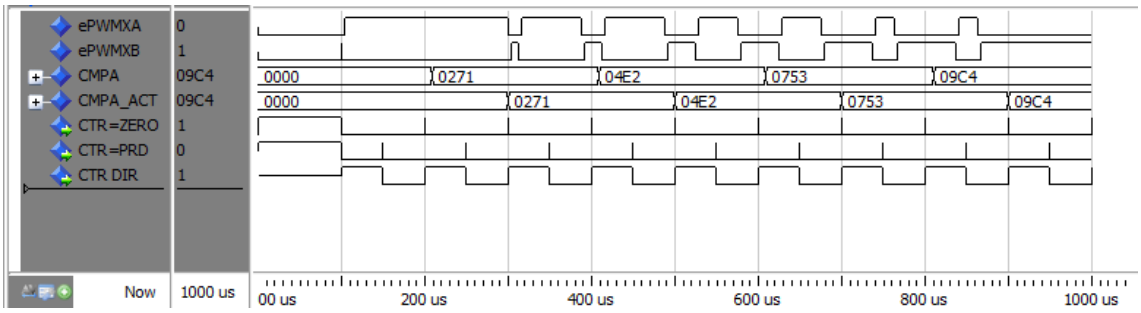
V této kapitole jsou uvedeny nastavení a výsledky simulací pro konkrétní aplikace, pro které může být v praxi tato periferie využita.

4.5.1 Nastavení symetrické PWM

V této aplikaci je uvedeno generování symetrické PWM. Nejdříve se 100 µs čeká a testuje se, jestli jsou signály v log. 0. V případě neúspěchu se generuje chybové hlášení. Poté se nastaví konfigurace na symetrickou PWM a jsou vždy testovány dvě periody se střídami 0 %, 25 %, 50 %, 75 % a 100 %. Hodnota `CMPA` se zapisuje vždy 10 µs po první periodě signálu, díky čemuž lze otestovat správnost Shadow registrů, které se načítají když registr `TBCTR` je nastaven na 0. Délka simulace je 1000 µs. Hodnota registru `CMPA` není uvedena, protože se mění v čase a je patrná z obr. 4.3. Nastavení hodnot registrů je uvedeno v tab. 4.2.

Tab. 4.2: Nastavení registrů pro symetrickou PWM

Registr	Hodnota registru ePWM
`TBCTL`	0xc032
`TBPRD`	0x09C4
`TBPHS`	0x0000
`CMPCTL`	0x0050
`CMPB`	0x0001
`AQCTLA`	0x0061
`DBCTL`	0x000b
`DBFED`	0x00AF
`DBRED`	0x00AF



Obr. 4.3: Simulace symetrického PWM signálu

4.5.2 Nastavení PWM pro řízení H-můstku

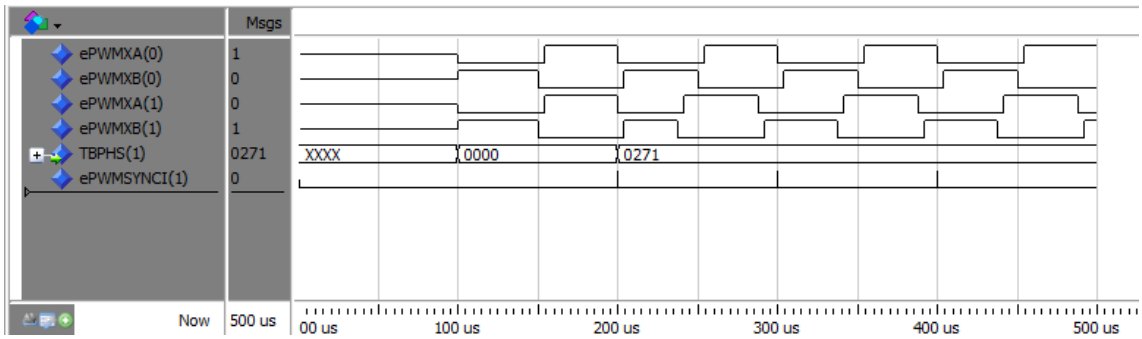
V této testovací aplikaci je řízení výstupního napětí prováděno buď pomocí modulování střídy řídicího signálu, nebo pomocí změny fáze mezi dvěmi ePWM moduly, tím ovládáme H-můstek pomocí fázového posunu (Phase-Shift Full-Bridge). V tabulce 4.3 jsou uvedeny nastavení registrů, celý výčet je uveden v datovém listu [8].

Tab. 4.3: Nastavení registrů pro H-můstek [8]

Registr	Hodnota registru ePWM1	Hodnota registru ePWM2
TBCTL	0x0014	0x0014
TBPRD	0x1388	0x1388
CMPCTL	0x0050	0x0050
CMPA	0x09C4	0x09C4
CMPB	0x0001	0x0001
AQCTLA	0x0061	0x0061
DBCTL	0x000B	0x000B
DBFED	0x00AF	0x00AF
DBRED	0x00AF	0x00AF

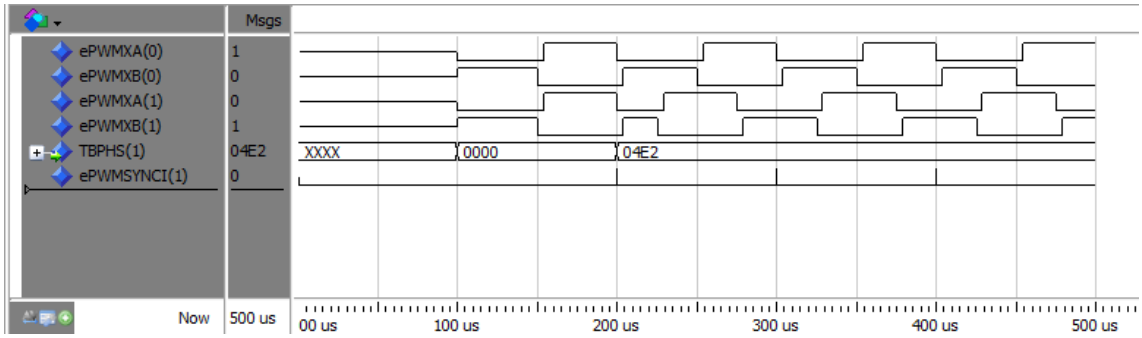
Frekvence signálu na výstupu má být 10 kHz, to odpovídá periodě 100 µs. Aby bylo možné nastavit správnou hodnotu do registru `TBPRD`, je potřeba spočítat počet hodinových impulsů za 100 µs. Hodinový impuls má 20 ns (50 MHz), takže hodnota v registru `TBPRD` bude odpovídat podílu požadované periody a periodě hodinového impulsu. Signál se přepne do stavu log. 0 v případě, že hodnota čítače je 0. Při hodnotě v registru `CMPA` se přepne do log. 1. Dead Time se povolují registrem `DBCTL` a jejich délka se nastavuje registry `DBFED` (zpoždění sestupné hrany) a `DBRED` (zpoždění vzestupné hrany). Výsledné průběhy s fázovými posuny jsou uvedeny níže. Střída nemá přesně 50 % z důvodu používání Dead Time.

Na následující straně jsou uvedeny výsledky z testovacích programů. Na obr. 4.4 jsou uvedeny PWM signály, které jsou od sebe vzájemně posunuty o 45°, přičemž odpovídající hodnota v registru `TBPHS` je 0x0271.



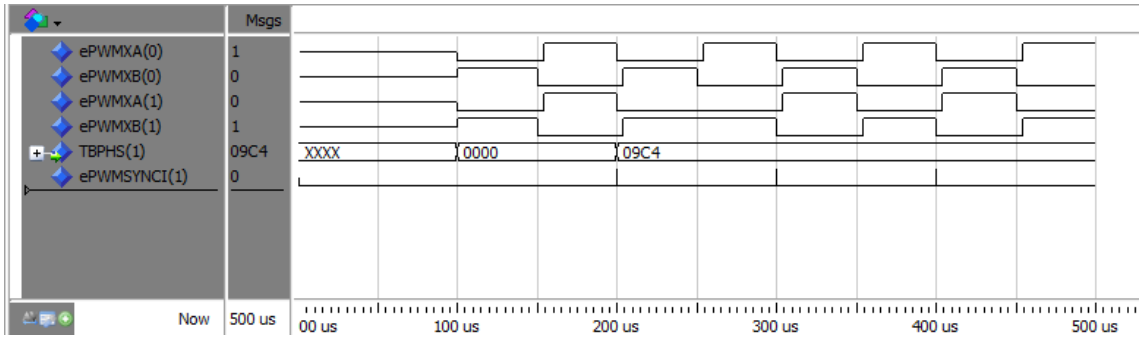
Obr. 4.4: Průběhy signálů z ePWM s fázovým posunem 45°

Na obr. 4.5 jsou uvedeny PWM signály, které jsou od sebe vzájemně posunuty o 90°, přičemž odpovídající hodnota v registru `TBPHS` je 0x04E2.



Obr. 4.5: Průběhy signálů z ePWM s fázovým posunem 90°

Na obr. 4.6 jsou uvedeny PWM signály, které jsou od sebe vzájemně posunuty o 180°, přičemž odpovídající hodnota v registru `TBPHS` je 0x09C4.



Obr. 4.6: Průběhy signálů z ePWM s fázovým posunem 180°

5 Implementace SPI ve VHDL

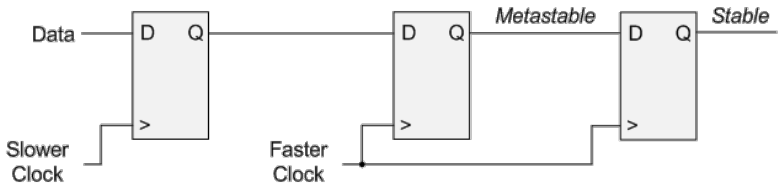
V páté kapitole je stručně popsána problematika časových domén v hradlovém poli, která souvisí s řešením implementace popisovaných modulů v této kapitole. Dále jsou zde popsány navržené moduly ve VHDL, do kterých je rozdělena činnost při komunikaci s ePWM po SPI.

5.1 Přechody mezi časovými doménami v FPGA

V dnešní době často pracují (v případě sekvenční logiky) zařízení se svým interním hodinovým signálem, tedy ve své časové doméně. Každý hodinový signál je jiný a často se liší v rychlosti oproti ostatním zařízením. Je potřeba si uvědomit, že pokud se propojuje zařízení, jehož interní hodinový signál pracuje v jiné časové doméně, než zařízení, do kterého je připojováno, tak se musí řešit synchronizace jeho signálů do hlavní časové domény. Taková synchronizace se typicky může provádět několika způsoby, přičemž je nutné rozlišovat, která z frekvencí hodinového signálu je vyšší. Při počáteční implementaci SPI Slave v této práci, byl přechod mezi časovými doménami špatně vyřešen, což vedlo k nestabilní funkci této periferie. Novější implementace, popisovaná v kapitole dále, již tyto nedostatky nemá.

Přechod z pomalejší do rychlejší časové domény

Poměrně jednoduchým řešením, jak takový přechod mezi časovými doménami vyřešit, je využít dvou bistabilních klopných obvodů (Flip-flop), které jsou zapojeny podle obr. 5.1. Data z pomalejší hodinové domény (klopný obvod vlevo) se vzorkují klopným obvodem (uprostřed), ale výstupem je signál, který může obsahovat metastabilitu. To znamená, že nelze určit, ve kterém logickém stavu se signál nachází. Metastabilita nastává při nesplnění časových podmínek (běžně označovaných jako t_{setup} a t_{hold}), které definují, jak dlouho musí být signál stabilní (neměnný) před a po překlopení klopného obvodu. Na eliminaci metastability se využívá připojení dalšího klopného obvodu. Tím pádem je signál již stabilní v hlavní hodinové doméně [18].



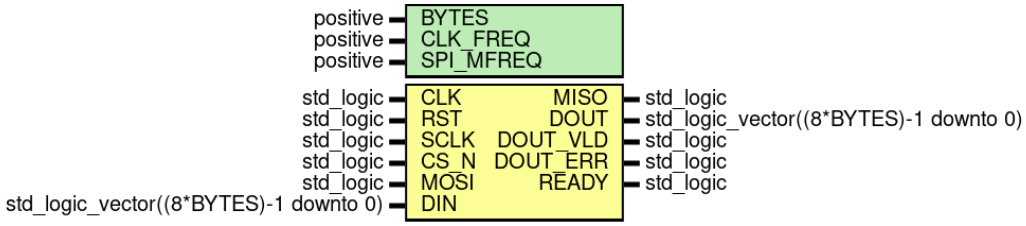
Obr. 5.1: Zapojení pro synchronizaci časových domén [18]

5.2 Návrh SPI Slave

V této podkapitole je popsána funkce, implementace a ukázka stavového automatu periferie SPI Slave.

5.2.1 Popis funkce SPI Slave

Periferie obsahuje vstupy (vlevo) a výstupy (vpravo), které jsou uvedeny na obr. 5.2. Jejich datový typ je taktéž patrný z obr. 5.2.



Obr. 5.2: Nákres vstupů a výstupů SPI Slave

Komunikace je navržena standardně tak, aby zařízení pracovalo v režimu SPI 0. Lze genericky definovat délku datového rámce v bytech.

K dispozici je několik výstupních signálů, které indikují, ve kterém stavu se SPI Slave nachází. V případě, že je modul připravený přijímat datové rámce od SPI Master, tak je výstup `READY` v log. 1. Pokud právě proběhlá komunikace byla úspěšná, tak výstup `DOUT_VLD` bude v log. 1. Pokud právě proběhlá komunikace úspěšně nedopadne, tak výstup `DOUT_ERR` bude v log. 1 až do té doby, dokud se zařízení vyresetuje, nebo Master ukončí komunikaci. Na výstupu `DOUT` se pak v případě validní komunikace budou k dispozici data od SPI Master. Na vstupu `DIN` se nacházejí data určená pro odeslání do SPI Master.

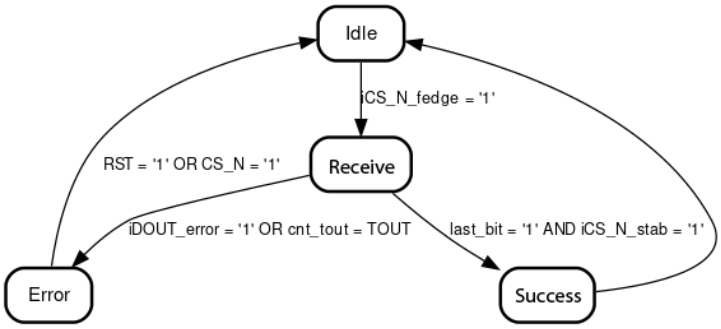
5.2.2 Popis implementace SPI Slave

Implementace je rozdělena do několika částí. V první části kódu se vzorkují vstupní signály z SPI Master do hlavní časové domény způsobem popsaným v předchozí podkapitole. Díky této synchronizaci lze již bezpečně pracovat se signály. U signálu `SCLK` je potřeba detekovat vzestupnou hranu a u signálu `CS_N` zase sestupnou hranu. Takové detekce jsou realizovány pomocí negovaného logického součinu signálů.

Další část kódu se stará o počítání náběžných hran signálu `SCLK`, což je nutnost pro správnou detekci komunikace. Správnost komunikace je zajištěna tím, že SPI Slave zná počet bitů, které očekává od Master, a tento počet přijatých bitů musí přesně odpovídat. Další kontrolní součástí je čítač, který měří čas uplynulý

od začátku komunikace. Mezní hodnota čítače je vypočtena z generických vstupů `SPI_MFREQ` a `CLK_FREQ`. Tím je ošetřen stav, kdy by například došlo k odpojení SPI Master z komunikace.

O samotný příjem se stará stavový automat, který je znázorněn na obr. 5.3. Výchozím stavem je stav `Idle`, ze kterého se pomocí sestupné hrany signálu `CS_N` přejde do stavu `Receive`. Ve stavu `Receive` probíhá příjem dat z Master. Pokud komunikace proběhne korektně, tak se přejde do stavu `Success`, kdy se hodnoty z interních registrů překlopí do výstupních registrů. Pokud však nastane během příjmu chyba, tak se přejde do stavu `Error`, kde je nutné modul resetovat.



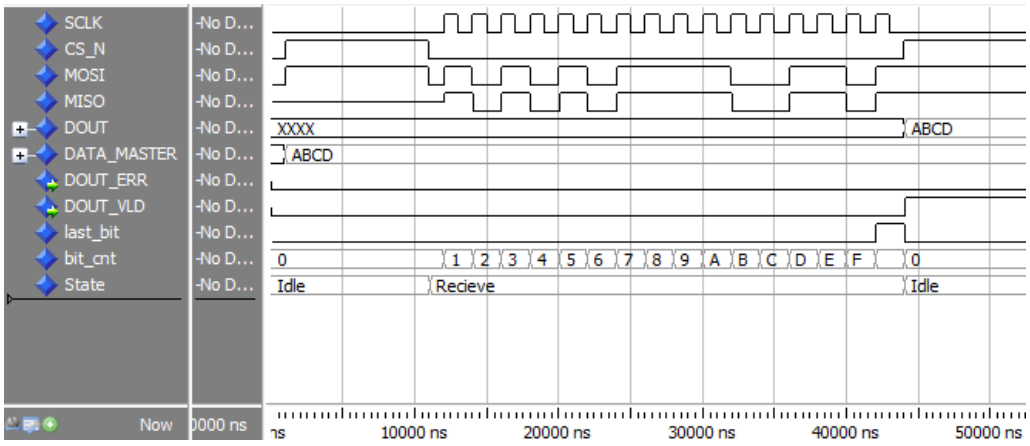
Obr. 5.3: Stavový automat pro SPI Slave

5.2.3 Analýza implementace SPI Slave

Pro implementaci je potřeba 192 logických bloků, 126 registrů a 40 pinů. Nejvyšší možná frekvence hodinového signálu f_{max} je stanovena na 109 MHz.

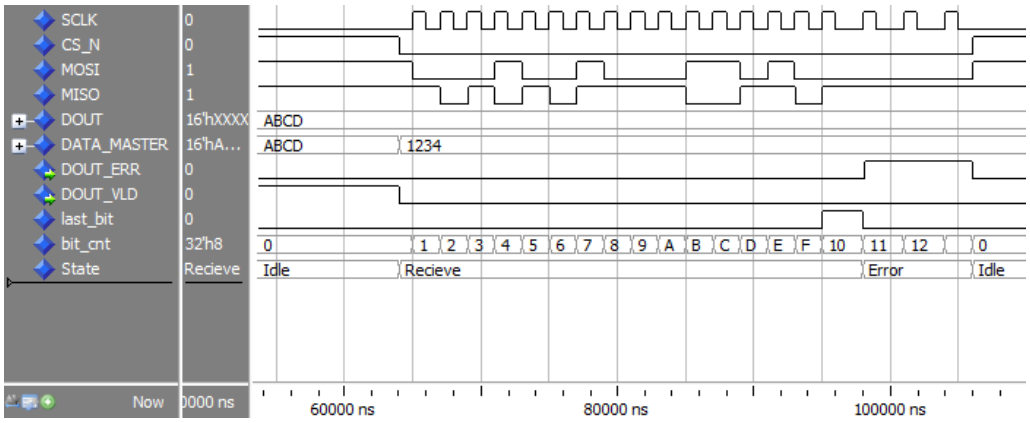
5.2.4 Test implementace SPI Slave

Na obr. 5.4 je znázorněna správná komunikace s SPI Slave. Je patrné, že správně přijatá data se překlopí po ukončení komunikace do výstupního registru tak, jak je očekáváno.



Obr. 5.4: Simulace SPI Slave při správné komunikaci

Na obr. 5.5 je znázorněna chybová komunikace s SPI Slave, kdy je odesláno víc pulsů `SCLK` než je očekáváno. Z obr. 5.5 je pak patrné, že se přijaté bity zahodí a zařízení přejde do chybového stavu, kde je nutný signál `RST` nebo `CS_N` v log. 1.



Obr. 5.5: Simulace SPI Slave při špatné komunikaci

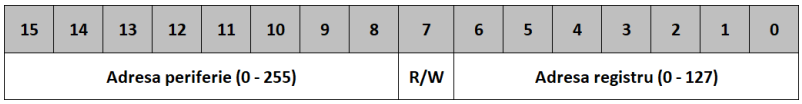
5.3 Návrh SPI Parser

Tento modul slouží pro správné rozdělení dat přijatých SPI Slave. Výstupem jsou data o adrese periferie a registru, o hodnotě registru a čtení nebo zápisu do registru.

5.3.1 Skladba datového rámce pro komunikaci s ePWM

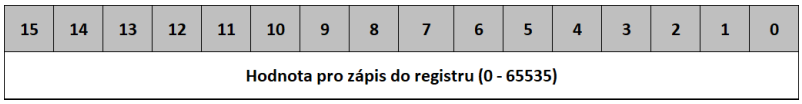
Nyní je implementována periferie, která zabezpečuje bezchybný přenos dat po SPI v definovaném datovém rámci. Pokud se má pomocí přijatých dat zapisovat, nebo

číst z registrů, tak je nutné stanovit, skladbu dat v datovém rámci. Potom je již přesně dané, kde se nachází adresa periferie, adresa registru a hodnota, která se má zapsat do registru. Pro tuto komunikaci jsou potřeba dva 16bitové datové rámce. První obsahuje informace o adrese periferie, informaci o čtení nebo zapisování do registru a samotnou adresu registru. Takový datový rámec je znázorněn na obr. 5.6.



Obr. 5.6: Ukázka prvního datového rámce SPI pro ePWM

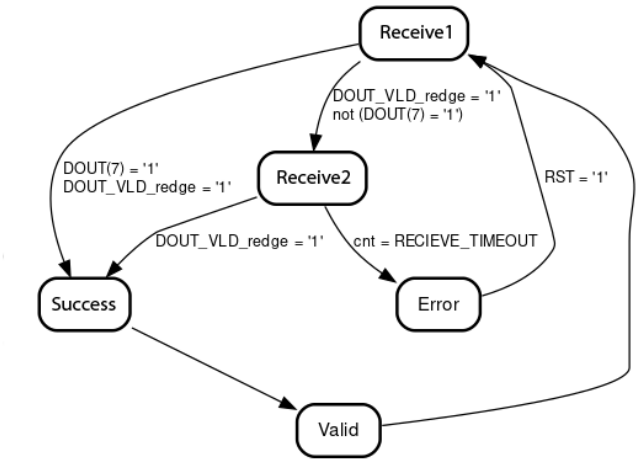
Druhý datový rámec obsahuje informaci o zapisujících datech do registru. Je znázorněn na obr. 5.7. Při režimu čtení z registru na těchto datech nezáleží a jsou ignorována. Vyslání druhého datového rámce při čtení je ale povinné, protože čtená data z registru jsou posílána ze Slave do Master.



Obr. 5.7: Ukázka druhého datového rámce SPI pro ePWM

5.3.2 Popis funkce SPI Parser

Funkce je řízena stavovým automatem, který je znázorněn na obr. 5.8. Výchozím stavem je stav `Receive1`. Při správném přijetí dat může být následující stav `Success` nebo `Receive2`. To se rozhoduje podle logické úrovně R/W bitu. V případě, že se jedná o čtení, tak se data překlopí do výstupního registru a generuje se validní impuls o délce jedné periody hodinového signálu. V případě, že se jedná o zápis, tak se čeká na přijetí druhého datového rámce s daty pro zápis. Pokud se druhý datový rámec neobjeví do časového limitu stanoveném v kódu (60 µs), tak se přejde do stavu `Error`, kde je nutný reset. V opačném případě byly přijaty datové rámce korektně a modul přejde do stavu `Success`, kde dojde k zápisu dat na výstup a dále do stavu `Valid`, kde dojde k vyslání validního pulsu.

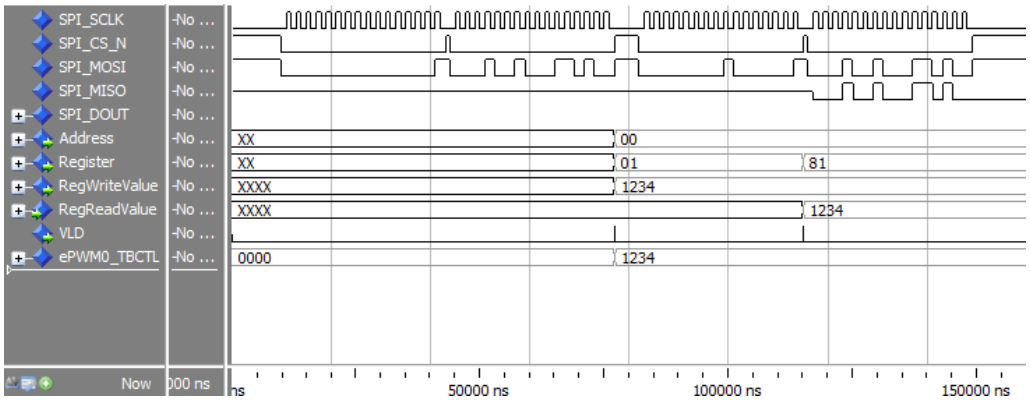


Obr. 5.8: Stavový automat SPI Parser

5.4 Návrh SPI ePWM a test celého konceptu rozšíření

Poslední modul zabezpečuje správné propojení mezi navrženými entitami (ePWM, SPI Parser a SPI Slave). V modulu lze genericky definovat počet implementovaných periferií. V případě, že přijde validní puls z SPI Parser, tak dojde k zápisu hodnot do příslušné periferie a do příslušného registru. Je zabezpečeno, aby šlo adresovat pouze daný definovaný počet periferií.

Na obr. 5.9 je vidět průběh zápisu a čtení z registru pomocí komunikace SPI. První čtyři signály zachycují komunikaci od SPI Master, přičemž signál SPI_DOUT obsahuje vektor přijatých dat. Tato data následně SPI Parser rozdělí na `Address`, `Register` a `RegWriteValue`. Po úspěšném zpracování je vidět puls signálu `VLD` a okamžitý zápis do registru `TBCTL` periferie ePWM, což je registr, který odpovídá adrese 0x01 a hodnota pro zápis je 0x1234. V další části Testbench testuje čtení těchto zapsaných dat z registru. Z obr. 5.9 je patrné, že při příjmu R/W bitu v log. 1 se data z registru překlopí na výstup a jsou posílány ze Slave do Master. Tímto způsobem je otestován příjem dat z Master i vysílání dat do Master po SPI a taktéž všechny dosud navržené moduly v této kapitole.



Obr. 5.9: Simulace zápisu a čtení z registru pomocí SPI

5.4.1 Analýza implementace SPI ePWM

Je nutno podotknout, že tato implementace ePWM obsahuje některé registry, které jsou na pevno nastavené a nelze do nich po SPI zapisovat. To je možnost, jak lze ušetřit zdroje v hradlovém poli a zjednodušit implementaci. Důvodem je, že některé registry se nastavují pouze na začátku a pokaždé na stejné hodnoty, tudíž není důvod je během chodu měnit. Registry, které jsou aktivní, jsou zobrazeny v tab. 5.1.

Tab. 5.1: Adresy registrů ePWM přístupných po SPI

Adresa	Registr	Pouze čtení
0x01	`TBCTL`	Ne
0x02	`TBSTS`	Ano
0x03	`TBPHS`	Ne
0x04	`TBCTR`	Ano
0x05	`TBPRD`	Ne
0x06	`CMPCTL`	Ne
0x07	`CMPA`	Ne
0x08	`CMPB`	Ne
0x09	`AQCTLA`	Ne
0x0A	`AQCTLB`	Ne
0x0B	`DBCTL`	Ne
0x0C	`DBRED`	Ne
0x0D	`DBFED`	Ne
0x0E	`PCCTL`	Ne

Pro implementaci s počtem 1 ePWM periferie v FPGA je potřeba 1350 logických bloků, 663 registrů a 36 pinů. Počet periferií lze genericky nastavit. Původním plánem bylo mít 36 fázově posunutých PWM signálů, což odpovídá 18 ePWM periferií. Pokud by se takové množství reálně implementovalo do FPGA, tak by bylo potřeba 17 040 logických bloků, 7089 registrů a 121 pinů. Z tohoto vyplývá, že FPGA určené pro vývoj nemá dostatečnou kapacitu zdrojů pro implementaci 18 ePWM periferií. Tato situace by se dala řešit tím, že by se použilo jiné FPGA, nebo by se nastavilo víc registrů s pevnými hodnotami. Jako maximální hodnota počtu ePWM periferií v používaném FPGA pro tuto práci je číslo 7, kdy je hradlové pole téměř celé využito (93%). Maximální počet periferií je omezen velikostí datového rámce pro adresu periferie u SPI. To odpovídá maximální hodnotě 255 zařízení. Lze realizovat i hromadný zápis do všech periferií a to tak, že adresa periferie v 1. datovém rámci bude odpovídat 0.

Aby bylo možné implementovat všech plánovaných 18 ePWM periferií, tak by bylo nutné využít FPGA s větším počtem logických bloků. FPGA by mohlo být zvoleno ze stejné rodiny jako používané FPGA pro tuto práci od společnosti Intel (Cyclone IV E), ale s čipem EP4CE22.

6 Implementace SCI ve VHDL

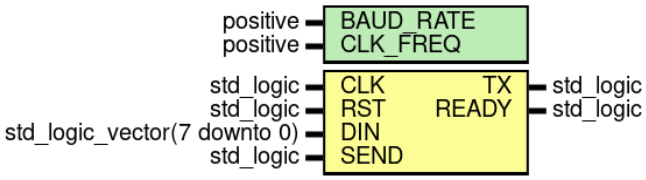
V šesté kapitole je popsána implementace SCI Master a nadřazené periferie s devíti sériovými linkami v jazyce VHDL.

6.1 Návrh SCI Master

V této podkapitole je popsána funkce, implementace, ukázka stavového automatu a použité zdroje periferie SCI Master.

6.1.1 Popis funkce SCI Master

Periferie obsahuje vstupy (vlevo) a výstupy (vpravo), které jsou uvedeny na obr. 6.1. Jejich datový typ je taktéž patrný z obr. 6.1.



Obr. 6.1: Nákres vstupů a výstupů SCI Master

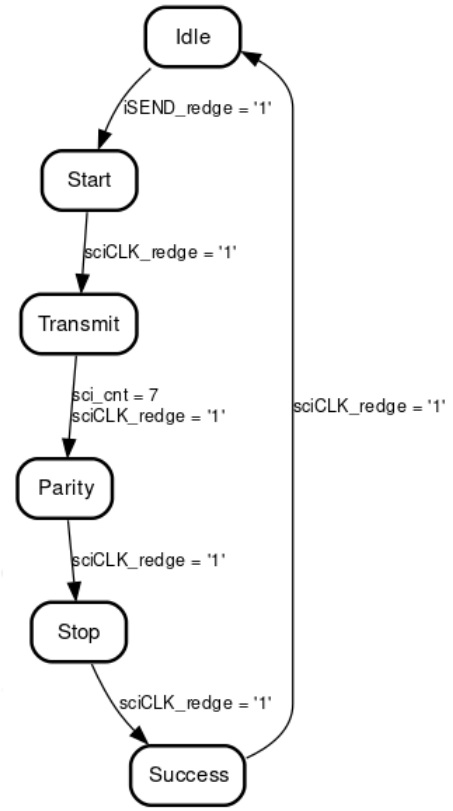
V případě, že aktuálně neprobíhá žádné vysílání, nebo není aktivní vstup `RST`, tak se na výstupu `READY` nachází log. 1. Tento stav indikuje, že je zařízení připraveno vysílat data. Při detekci náběžné hrany signálu `SEND` začne vysílání. Odesílaný rámec se skládá z 1 start bitu, 8 datových bitů, 1 bitu se sudou paritou a 1 stop bitu.

6.1.2 Popis implementace SCI Master

Implementace této periferie je rozdělena do několika částí. První část kódu zahrnuje děličku kmitočtu na požadovaný Baud Rate, který se nastavuje pomocí generické proměnné `BAUD_RATE`. Pro správný přepočet hodinových pulzů pro děličku kmitočtu je nutno zadat i frekvenci interního hodinového signálu označeného jako `CLK_FREQ`.

V druhé části kódu se nachází synchronizace vstupu `SEND` do hlavní časové domény pomocí dvou bistabilních klopných obvodů (Flip-flop). To znamená, že pro spolehlivou detekci, musí být puls signálu `SEND` delší než 1 perioda interního hodinového signálu. Detekce náběžné hrany je realizována negovaným logickým součinem signálů z obou bistabilních klopných obvodů. To zajistí generování pulsu při náběžné hraně o délce periody jednoho hodinového impulsu.

Ve třetí části kódu se nachází stavový automat, jehož vizualizace je znázorněna na obr. 6.2. Koncepce stavového automatu se skládá z několika stavů tak, aby jednoduchou změnou bylo možné upravit parametry vysílání. Stav `Idle` je vychozí stav, ve kterém je detekovaná náběžná hrana signálu `SEND` a spouští vysílání. Při detekci spuštění vysílání dojde k překlopení dat ze vstupu `DIN` do interního registru. Ve stavu `Start` se odesílá 1 Start bit a do dalšího stavu se přechází v případě náběžné hrany děleného hodinového signálu SCI. Ve stavu `Transmit` probíhá vysílání dat uložených v interním registru. V případě, že čítač bitů bude mít hodnotu 7, tak dojde k přechodu do dalšího stavu. Počet odesílaných bitů lze lehce upravit změnou čísla 7. Ve stavu `Parity` se počítá a odesílá sudá parita odeslaných bitů pro kontrolu přijímaným zařízením. Dalším stavem je `Stop`, který vysílá 1 Stop bit. Poslední stav `Success` indikuje úspěšné dokončení a zajišťuje přechod do stavu `Idle`.



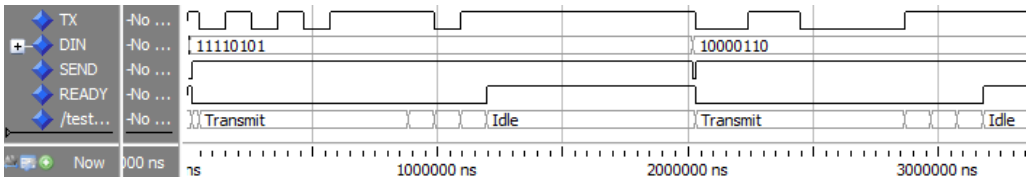
Obr. 6.2: Stavový automat pro SCI Master

6.1.3 Analýza implementace SCI Master

Pro implementaci je potřeba 128 logických bloků, 83 registrů a 13 pinů. Nejvyšší možná frekvence hodinového signálu f_{max} je stanovena na 153 MHz.

6.1.4 Test implementace SCI Master

Na obr. 6.3 je vidět simulace navrženého modulu SCI Master. Je patrné, že sled komunikace je takový, jaký byl uváděn v kapitole 6.1.2. Signál `TX` je hlavním výstupním signálem modulu. Signál `DIN` obsahuje data určená pro vysílání, náběžnou hranou signálu `SEND` pak toto vysílání odstartuje. Pokud je zařízení připraveno vysílat, tak signál `READY` bude v log. 1. Dělení kmitočtu na požadovaný Baud Rate funguje správně.



Obr. 6.3: Simulace SCI Master

6.2 Návrh SCI Parser

Tato část textu popisuje návrh modulu, který zabezpečuje správné rozdělení dat přijatých SPI Slave. Název SCI Parser může být zavádějící, ale jedná se o přijetí dat z SPI komunikace a byl takto zvolen z důvodu rozlišení Parser modulu pro obě navrhovaná rozšíření v této práci. Výstupy jsou číslo sériové linky a data, která se po sériové lince budou odesílat.

6.2.1 Skladba datového rámce pro komunikaci po SCI

Stejně jako v předchozí kapitole, tak i zde je nutné stanovit, jak bude vypadat datový rámec posílaný po SPI. Stanovený datový rámec je patrný z obr. 6.4. První čtyři bity jsou rezervovány, další čtyři určují číslo sériové linky, po kterých se bude posílat zbylých 8 bitů dat. Jelikož není k dispozici nadřazené zařízení, které tyto SCI zprávy bude dekódovat, tak se v budoucnu skladba datového rámce možná změní. To nebude složité upravit, protože modul pracuje ve své entitě, které je poměrně jednoduše modifikovatelná.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rezervováno				Číslo SCI linky				Data odesílaná po SCI							

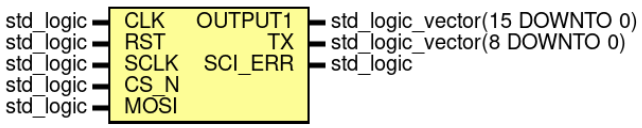
Obr. 6.4: Ukázka datového rámce SPI pro 9 sériových linek

6.2.2 Popis funkce SCI Parser

Funkce entity je velmi podobná předchozí funkci SPI Parser. Obsahuje v sobě modul SPI Slave. Při přijetí validních dat po SPI, dojde k vyslání impulsu o délce jednoho hodinového kmitočtu. Data z SPI se zapíší do patřičných výstupů, kde jsou číslo SCI linky a data pro SCI linku. Komunikace se řídí stavovým automatem se dvěma stavy - `Receieve` a `Valid`. Realizace by byla samozřejmě možná i bez stavového automatu, nicméně v rámci dodržení koncepce s již hotovým modulem SPI Parser byl stavový automat ponechán.

6.3 Návrh nadřazeného komunikačního zařízení se sériovými linkami a test celého konceptu

Vrcholový modul (Top Level Entity) obsahuje SCI Parser, popsaný v předchozí podkapitole, a devět sériových linek. Počet sériových linek je nastavitelný konstantou `SCI_CNT` a jejich generování probíhá pomocí FOR cyklu. Pokud SPI Parser bude mít na svém výstupu validní data, tak se pomocí přijatého impulsu zapíší data pro příslušnou SCI linku a tato linka začne vysílat data. Je ošetřen případ, kdy adresa SCI linky bude větší než 9. V takovém případě bude výstup `SCI_ERR` v log. 1 až do resetu, nebo do přijetí nových dat. Pro lepší názornost je uveden obr. 6.5 s patrnými vstupy a výstupy tohoto modulu. `OUTPUT1` slouží pro kontrolu dat přijatých po SPI Slave během testování.



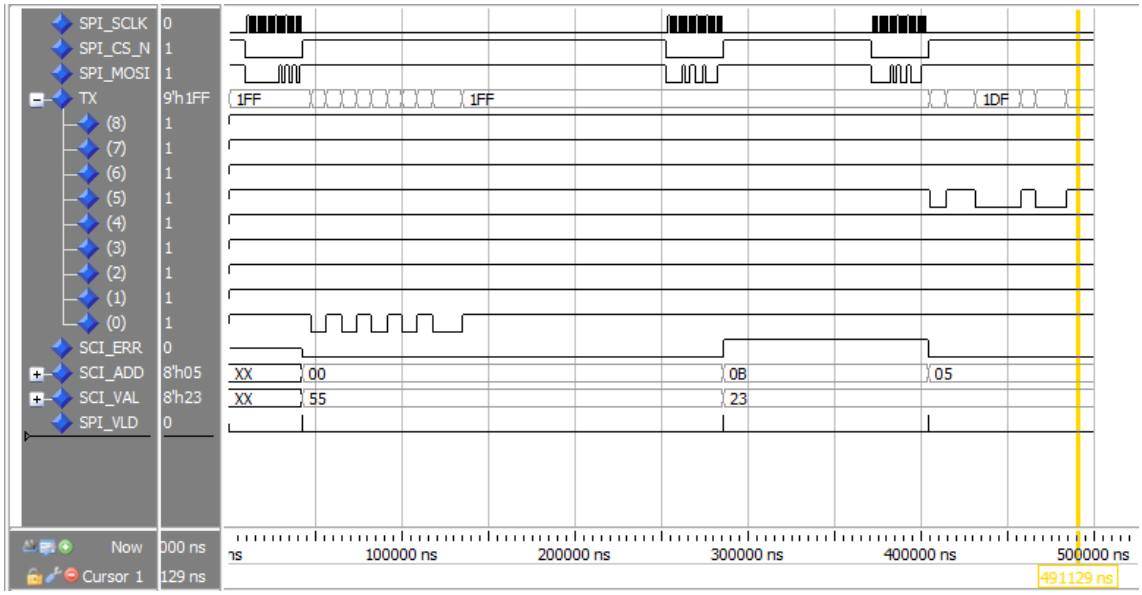
Obr. 6.5: Nákres vstupů a výstupů modulu s 9 sériovými linkami

6.3.1 Test implementace navrženého komunikačního zařízení s devíti sériovými linkami

Na obr. 6.6 je vidět simulace navržené periferie. Nejdříve je vidět v horní části probíhající SPI komunikace, která proběhla validně. Následně jsou přijatá data rozdělena a předána do registrů `SCI_ADD` a `SCI_VAL`. Náběžná hrana `SPI_VLD` pak způsobí start vysílání příslušné sériové linky. Nejdříve je testována první sériová linka s vysílanou hodnotou 0x55.

Dále je testována adresa mimo rozsah, což způsobí překlopení výstupu `SCI_ERR` do log. 1. Tak lze poznat, že přijatá data po SPI byla sice správně přijatá, nicméně adresa linky je mimo rozsah. Při dalším přijetí správných dat je chybový výstup resetován a celý modul pracuje dále správně.

Rychlosti SCI linek jsou genericky definovatelné, v tomto případě je nastavena na rychlost 115 200 baudů.



Obr. 6.6: Simulace navrženého komunikačního zařízení s devíti sériovými linkami

Z obr. 6.6 je patrné, že funkce je přesně taková, jaká byla předpokládána. Navržený koncept s devíti sériovými linkami lze považovat za funkční. Předmětem dalšího testování bude jejich test na reálném hardware a zpětné dekódování signálů z 9 linek zpět.

6.3.2 Analýza implementace modulu s devíti sériovými linkami

Pro implementaci je potřeba 1073 logických bloků, 697 registrů a 32 pinů. Nejvyšší možná frekvence hodinového signálu f_{max} je stanovena na 218,29 MHz.

7 Testování navržených komponent na reálném hardware

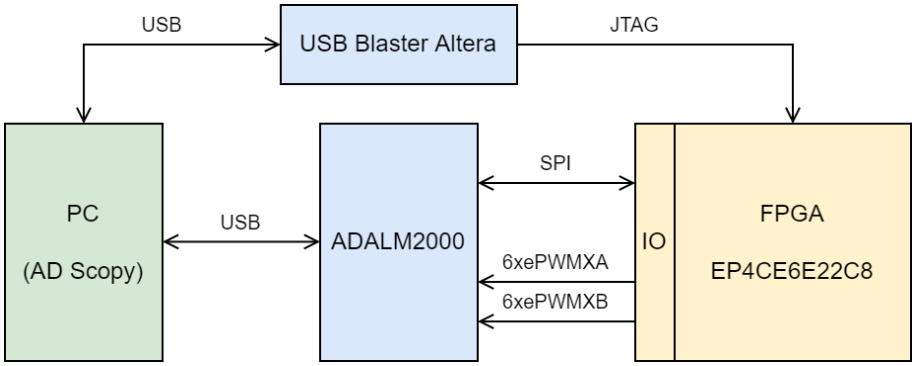
V této kapitole jsou prezentovány dosažené výsledky a provedeno testování na reálném hardware (hradlovém poli) navržených komponent v jazyce VHDL.

7.1 Testování ePWM s rozhraním SPI v FPGA

V tomto textu je uvedeno testování ePWM řízené pomocí SPI, tedy jedné z variant rozšíření pro řídicí desku. V tomto konkrétním příkladě se nachází 6 ePWM periferií v jednom zařízení a jsou ovládány pomocí SPI. Jako logický analyzátor výstupů hradlového pole a jako SPI Master, po kterém se budou posílat datové rámce, je použit již dříve zmíněný výukový modul ADALM2000.

7.1.1 Blokové schéma pro analyzování výstupů s PWM FPGA

Na obr. 7.1 je blokově uvedeno propojení jednotlivých komponent mezi sebou. Konfigurace je do hradlového pole je nahrávána pomocí rozhraní JTAG, které s PC komunikuje pomocí sběrnice USB. Výukový modul komunikuje s PC také pomocí USB a jsou do něj přivedeny vstupy s PWM signálem z FPGA. Výstupními signály z ADALM2000 jsou ty na kterých se provádí komunikace SPI. Tyto popsané signály jsou přivedeny na vstupně-výstupní piny hradlového pole.



Obr. 7.1: Blokové schéma pro testování ePWM s SPI

7.1.2 Inicializace ePWM

Po spuštění je na výstupu všech PWM signálů log. 0. Takový stav je vhodný a očekávaný, protože hodnota registrů po spuštění tomuto chování odpovídá. Je tedy

nutné provést inicializaci registrů, kde můžeme s výhodou využít konfigurace, kdy se nastavují všechny ePWM periferie najednou. Z tohoto vyplývá, že Adresa periferie bude 0x00 (prvních 8 bitů). R/W bit bude mít hodnotu 0 a zbývajících 7 bitů bude obsahovat adresu registru. Dalších 16 bitů dat v datovém rámci obsahují hodnotu zapisovanou do registru. Zapisované hodnoty jsou uvedeny v tab. 7.1. Jedná se o nastavení, kdy je čítač v režimu `UP`. Maximální hodnota čítače `TBPRD` je 0x1388, což odpovídá hodnotě periody 100 µs. Hodnota pro komparátor `CMPA` je 0x04E2, což odpovídá střídě o velikosti 25 %.

Tab. 7.1: Hodnoty zapisované do registrů ePWM při inicializaci

Adresa	Registr	Hodnota
0x01	`TBCTL`	0x0050
0x02	`TBSTS`	Pouze čtení
0x03	`TBPHS`	0x0000
0x04	`TBCTR`	Pouze čtení
0x05	`TBPRD`	0x1388
0x06	`CMPCTL`	0x0050
0x07	`CMPA`	0x04E2
0x08	`CMPB`	0x02E2
0x09	`AQCTLA`	0x0012
0x0A	`AQCTLB`	0x0012
0x0B	`DBCTL`	0x000B
0x0C	`DBRED`	0x00FF
0x0D	`DBFED`	0x00FF
0x0E	`PCCTL`	0x0000

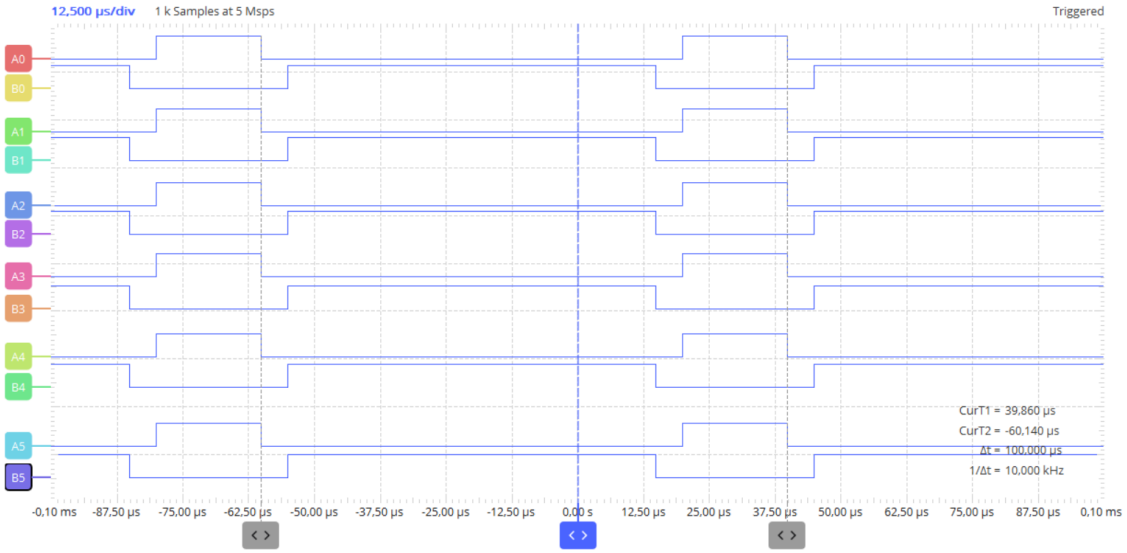
Pro lepší názornost je uveden příklad prvních dvou datových rámců na obr. 7.2. Jedná se o zapisování do všech periferií, do registru `TBCTL` a hodnoty 0x0050. V horní části obr. 7.2 je uveden první datový rámec o velikosti 2 byty a ve spodní části zase druhý s totožnou velikostí. Mezi posláním datových rámců je nutné dodržet časový limit, který je definován ve VHDL kódu konstantou `RECIEVE_TIMEOUT`. Jeho výchozí hodnota je 3000 hodinových impulsů, což při frekvenci 50 MHz odpovídá 60 µs.

Na obr. 7.3 je znázorněno, jak se v čase mění výstupy na pinech hradlového pole. Jednotlivé dvojice signálů z periferie jsou znázorněny písmeny A nebo B s příslušným indexem, který odpovídá pořadí periferie. Z obr. 7.3 je patrné, že výstup je přesně takový, jaký byl očekáván. To znamená, že délka jedné periody je 100 µs, střída je 25% a jsou vložené tzv. Dead Time o délce odpovídající 5 µs.

Adresa periferie (0 - 255)								R/W	Adresa registru (0 - 127)						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Hodnota pro zápis do registru (0 - 65535)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0

Obr. 7.2: Ukázka prvních odesílaných datových rámců při testování ePWM s SPI



Obr. 7.3: Ukázka výstupů hradlového pole po inicializování ePWM s SPI

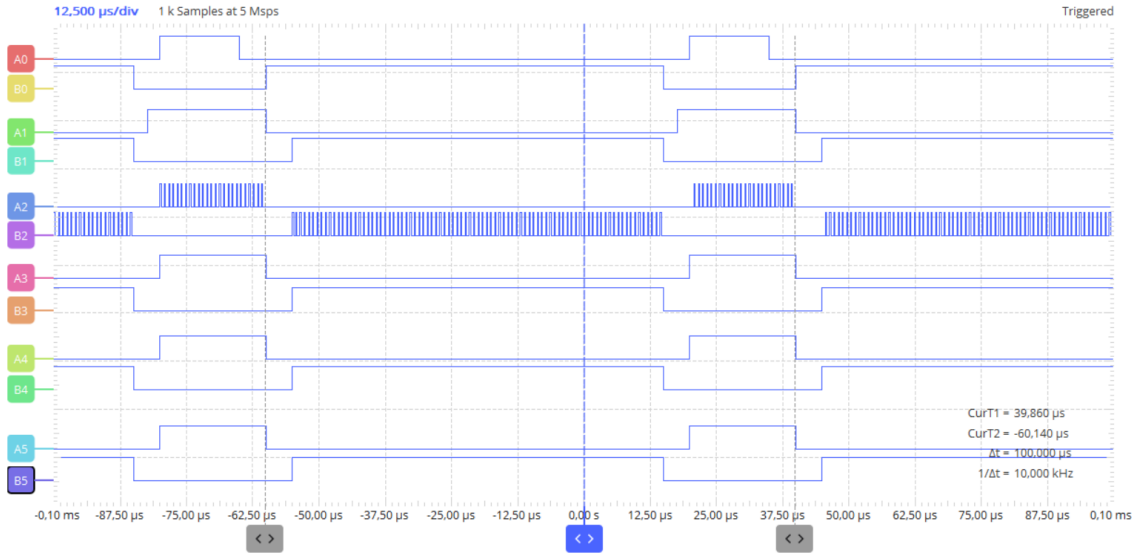
7.1.3 Změna parametrů ePWM za provozu

Pro popis testování změny parametrů ePWM za provozu bylo rozhodnuto na ukázku vybrat tři. Jedná se o změnu periody první periferie (tedy A0 a B0), zmenšení délky Dead Time náběžné hrany druhé periferie a aktivaci PWM Chopper u periferie třetí. Postupné hodnoty, které jsou po SPI Master posílány jsou znázorněny v tab. 7.2.

Výstup po odeslání těchto datových rámců je vidět na obr. 7.4. Je patrné, že vlastnosti signálů se změnily, tak jak bylo očekáváno.

Tab. 7.2: Postupně posílané datové rámce po SPI Master

Pořadí	Hodnota	Poznámka
1	0x0107	Zápis do periferie 1, registru 0x07
2	0x03e8	Zápis hodnoty 0x03E8
3	0x020c	Zápis do periferie 2, registru 0x0C
4	0x007f	Zápis hodnoty 0x007F
5	0x030e	Zápis do periferie 3, registru 0x0E
6	0x0001	Zápis hodnoty 0x0001



Obr. 7.4: Ukázka výstupů hradlového pole po změně parametrů ePWM s SPI

7.1.4 Analýza implementace 6 ePWM s SPI

Pro konečnou implementaci v FPGA bylo potřeba 5157 logických bloků, 2553 registrů a 61 pinů.

7.1.5 Zhodnocení testování ePWM s SPI na hradlovém poli

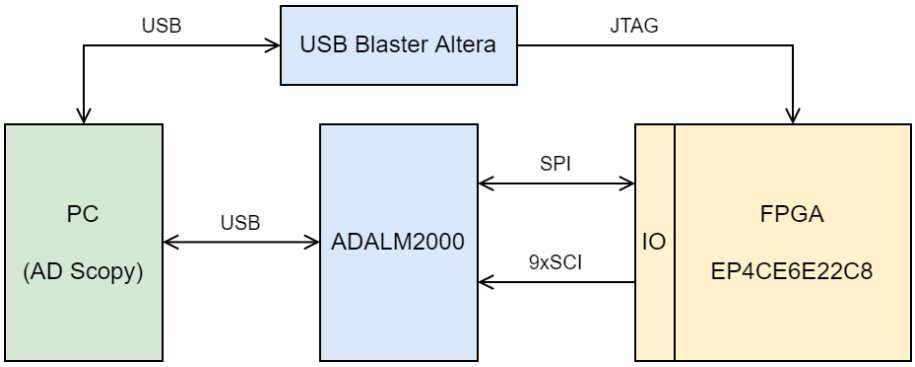
Navržený koncept periferie byl otestován. Z přiložených obrázků v podkapitolách výše je patrné, jak byla periferie zkoušena. Při popisu testování byly uvedeny pouze některé zkoušky nastavení, ale při důkladném testování byl vyzkoušen zápis i čtení ze všech dostupných registrů. Testování proběhlo úspěšně a lze tvrdit, že takto navržený koncept funguje jak v simulaci, tak i na reálném hardware. Dalším předmětem testování může být test na reálném měniči.

7.2 Testování rozšíření s devíti sériovými linkami v FPGA

V této podkapitole je uvedeno testování devíti SCI linek, které jsou ovládány pomocí SPI. Pro analyzování sériových linek a jako SPI Master je použit logický analyzátor ADALM2000.

7.2.1 Blokové schéma pro analyzování výstupů sériových linek FPGA

Na obr. 7.5 je uvedeno blokové schéma zapojení pro testování rozéíření s 9 sériovými linkami. V podstatě se jedná o totéž zapojení, které bylo uvedeno v předchozí kapitole, jen s tím rozdílem, že na výstupu nejsou PWM signály, ale 9 sériových linek. Konfigurace je do FPGA opět nahrávána pomocí JTAG a výukový modul pracuje s PC pomocí sběrnice USB.

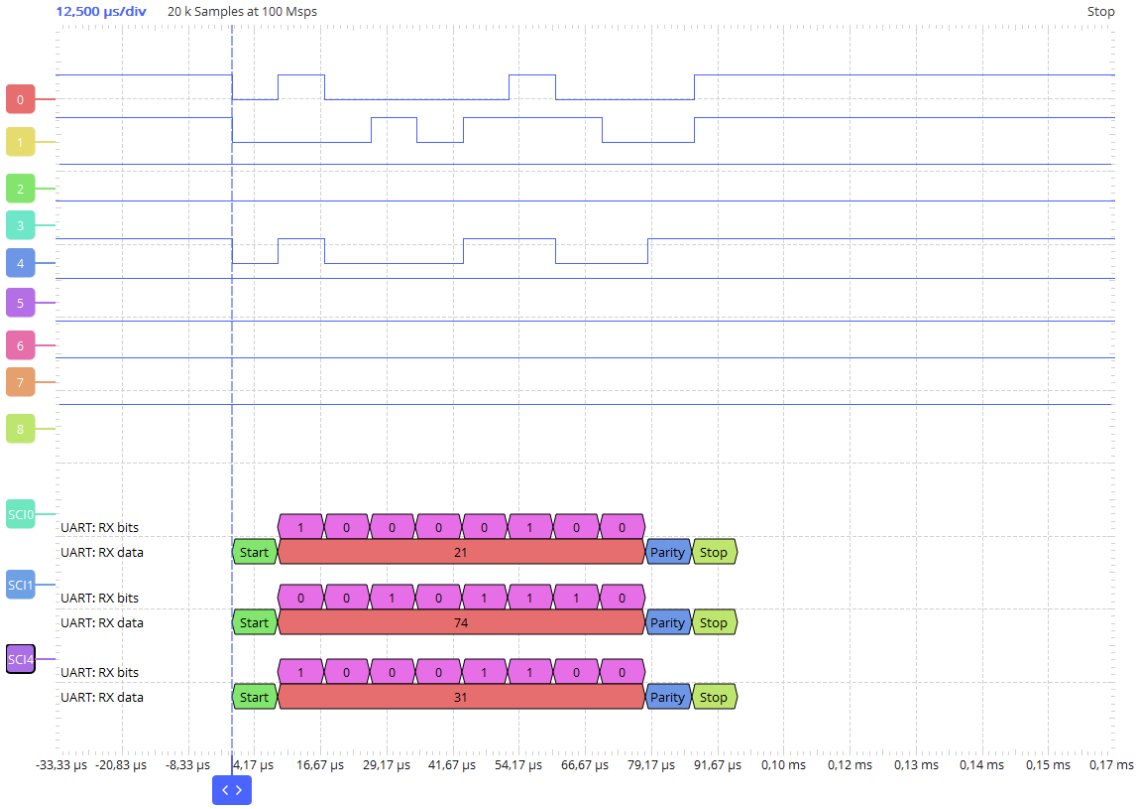


Obr. 7.5: Blokové schéma pro testování rozšíření se sériovými linkami

7.2.2 Předvedení funkce

Testování bude v tomto případě jednoduché, protože se v podstatě jedná pouze o převod z SPI na SCI. Po SPI jsou odeslány celkem tři datové rámce, kdy jejich velikost jsou 2 byty. Tak jak bylo popsáno již dříve, tak datový rámec obsahuje informaci o číslu SCI linky a o datech určených na odeslání SCI linkou. V tomto konkrétním případě jsem se rozhodl odeslat po SPI Master následující data: 0x0021, 0x0174, 0x0431. Data odpovídají lince 0, hodnotě 0x21; lince 1, hodnotě 0x74; lince 4, hodnotě 0x31. Z obr. 7.6 je patrný výstup FPGA, kde je zobrazeno všech 8 linek. Po odeslání popsaného datového rámce výstupy sériových linek vypadají podle obr. 7.6 následovně. Pro tři testované linky jsem přidal SCI dekodér, pro lepší znázornění.

Jsou tedy patrná data posílaná po SCI, což odpovídá očekávaným datům, včetně paritních bitů.



Obr. 7.6: Ukázka výstupů hradlového pole po odeslání datových rámců pomocí SCI

7.2.3 Zhodnocení testování rozšíření s devíti sériovými linkami

Navržený koncept periferie byl otestován. Popsaný způsob testování dokumentoval test tří linek, ale reálně byly otestovány všechny linky. Zatím není k dispozici rozšiřující FPGA, které by tyto signály dekódovalo a generovalo na základě nich PWM impulsy, tudíž to bude dalším předmětem testování, jakmile bude navrženo.

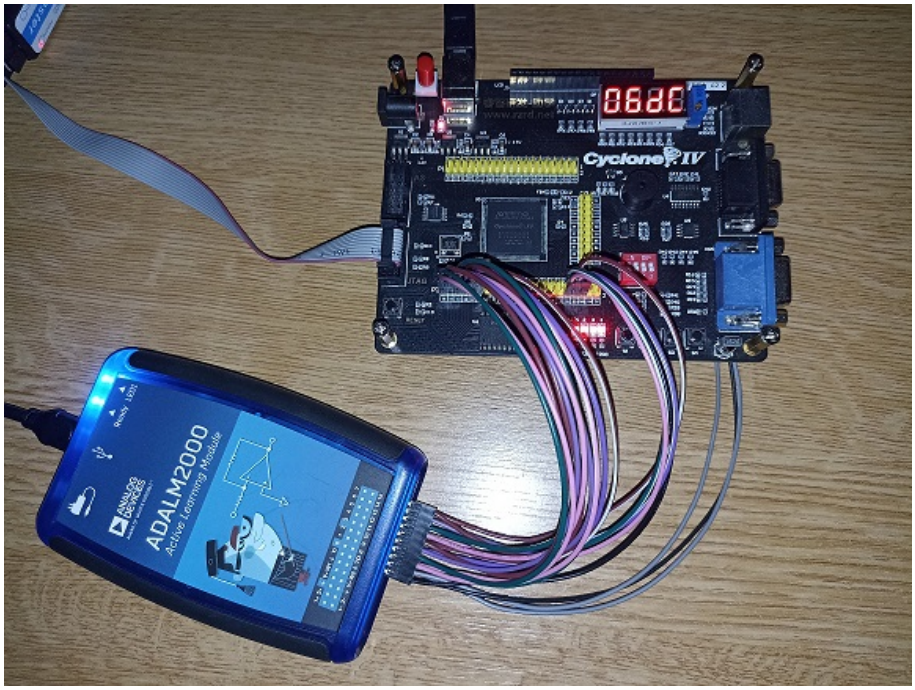
7.2.4 Analýza implementace rozšíření s devíti sériovými linkami

Pro konečnou implementaci v FPGA bylo potřeba 1056 logických bloků, 718 registrů a 34 pinů.

7.3 Zhodnocení testování na reálném HW

Jak bylo již zmíněno, tak oba koncepty se ukázaly funkční jak v simulaci, tak i na reálném HW. Při počáteční implementaci SPI Slave nebyl správně řešen přechod mezi časovými doménami, a tak jsem se rozhodl, že celou implementaci udělám znovu. Ta se již ukázala jako podstatně robustnější a dokáže zaručit správnost operací.

Reálné zapojení, které bylo využíváno při testování, je podle blokového schéma na obr. 7.1 a 7.5 znázorněno na obr. 7.7. Na spodní části obr. 7.7 je vidět výukový modul ADALM2000, nad ním se nachází USB Blaster, který slouží pro nahrávání konfigurace do FPGA, dále pak napájecí kabel FPGA. Z periferií, které kit nabízí byl využíván 7 segmentový displej, a to pro zobrazení hodnoty, která se zapisuje do registru.



Obr. 7.7: Ukázka zapojení při testování na reálném HW

V příloze A jsou k dispozici celkem tři videonahrávky demonstrující testování popisované v této kapitole. První video `FPGA_ePWM_test.mp4` demonstruje testování popsané v kapitole 7.1. Druhé video `FPGA_ePWM_duty_cycle_test.mp4` demonstruje testování, kdy byla modifikována střída signálu. Signály byly připojeny jak na logický analyzátor, tak i na LED diody, které FPGA obsahuje. Z průběhu signálu je pak patrná změna střídy. Třetí video `FPGA_SCI_test.mp4` demonstruje testování popsané v kapitole 7.2.

Je nutno podotknout, že obě navržená rozšíření byla testována za ideálních podmínek, tedy bez rušení a ostatních parazitních vlivů, které se běžně v průmyslu vyskytují. To znamená, že pro průmyslové použití budou potřeba další testy robustnosti zařízení.

7.4 Další možná rozšíření řídicího rozhraní pro buňkový měnič

V této podkapitole jsou uvedena další dvě rozšíření s FPGA, která by byla možná v budoucnu realizovat.

Pro rozšíření s 9 sériovými linkami, navrhované v této práci, není v současné době k dispozici navazující zařízení, a tak jako další možné rozšíření pro rozhraní buňkového měniče může být rozšíření s programovatelným hradlovým polem, které bude zpracovávat data z 9 sériových linek. Data by byla hradlovým polem zpracována a posílána do příslušných buněk měniče.

Dalším rozšířením může být řídicí karta pro jednu buňku. Vstupem by byly dva PWM signály, kterým by byl přidán Dead Time a které by byly na výstupu invertovány tak, aby bylo možné buňku řídit unipolárně. Výstupem by byly 4 signály pro řízení pulsního měniče. Uvažovalo by se sériové zapojení řídicích karet tak, aby bylo možné řídit několik buněk, kde by byl komunikační protokol zvolen RS485. Rozšíření by navíc mohlo obsahovat i PWM řízení ventilátorů měniče, měření teploty v měniči nebo přepěťovou ochranu případně podpěťovou ochranu.

Závěr

Cílem bakalářské práce bylo navrhnout rozšíření řídicího rozhraní pro buňkový měnič v FPGA. Konkrétně se jedná o periferii generující PWM signál na základě dat z SPI a o periferii, která vysílá data po SCI taktéž na základě dat z SPI.

V první kapitole jsou popsány použité periferie a principy, které jsou v práci používány. To zahrnuje popis modulů a principů PWM, SPI a SCI.

Ve druhé kapitole je popsána funkce střídače a popsána nadřazená řídicí deska, která bude určena pro ovládání navržených rozšíření. Jsou zde definovány základní principy převodu stejnosměrného napětí na střídavé. Bloková schémata pro obě rozšíření jsou taktéž uvedeny v této kapitole.

Následující kapitoly jsou věnovány popisu praktické realizace práce. Třetí kapitola popisuje prostředky a nástroje zvolené pro realizaci práce. Pro realizaci této práce byl zvolen vývojový kit s čipem Altera Cyclone IV EP4CE6E22C8. Další pomůckou při realizaci práce byl výukový modul ADALM2000. Ten sloužil jako logický analyzátor a jako Master a Slave pro testování komunikací SPI a SCI. Tím bylo možné ověřit správnost implementace programu pro SPI a SCI ve VHDL. Jako vývojové prostředí bylo použito Intel Quartus Prime 20.1. V kapitole je popsán i průběh návrhu hardware. To je důležité, protože pro návrh hardwaru s takovou komplexností je již třeba postupovat dle metodiky. Jako vhodnou metodiku jsem zvolil návrh jednotlivých modulů formou Bottom-up designu.

Ve čtvrté kapitole je popsána implementace modulu ePWM v jazyce VHDL. Architektura pochází z konceptu ePWM, která je implementována na řídicím rozhraní se signálovým procesorem TMS320F28835 od Texas Instruments. Koncept periferie ePWM je rozdělen na několik bloků a u každého z nich jsou popsány registry, včetně jejich funkce. V kapitole je uveden i počet zdrojů potřebných pro tuto periferii. U ePWM periferie nebyly implementovány všechny registry, jaké jsou dostupné na signálovém procesoru od Texas Instruments, ale jen ty důležité, které jsou potřeba. V závěru kapitoly je uvedeno testování některých praktických příkladů v simulaci, které jsou pro danou aplikaci potřeba.

V páté kapitole je popsána implementace SPI Slave v jazyce VHDL. Při návrhu této periferie bylo nutné brát v potaz několik věcí. Jedná se o přechod mezi časovými doménami nebo o zajištění atomičnosti operací. Při původní implementaci této periferie to nebylo správně zajišťováno a mohlo docházet k nesprávné funkci celé periferie. Při nové implementaci, popsané v práci, je již periferie robustnější a umí detekovat i chybu komunikace. Implementace SPI Slave rozhraní se skládá z několika částí - SPI Slave, SPI Parser. SPI Slave zajišťuje správné přijetí komunikace. Jsou ošetřeny stavy, kdy přijde více, nebo méně bitů, než je očekáváno nebo když se komunikace v průběhu přeruší. V takových případech zařízení signalizuje chybu. SPI

Parser zajišťuje správné rozdělení přijatých dat podle stanovených datových rámců. SPI s ePWM je rozhraní, které má jednoduchou logiku a jen zajišťuje rozhraní mezi SPI a ePWM. V rozhraní je možné genericky nastavovat počet periferií. V kapitole jsou uvedeny testy navržených modulů a test celého konceptu v simulaci.

V šesté kapitole je popsána implementace SCI Master v jazyce VHDL. Návrh Master zařízení je o něco jednodušší oproti Slave zařízení, protože stačí zajistit správné vysílání dat a nevyskytují se problémy, jaké byly popsané v předchozím odstavci. Implementace je opět rozdělena do tří modulů. SCI Master zajišťuje správné odesílání dat po SCI. Genericky lze nastavit jen rychlost komunikace. Vysílá se vždy 8 bitů + 1 paritní bit. SCI Parser zajišťuje správné rozdělení dat přijatých po SPI podle stanovených datových rámců. Implementace je řešena analogicky jako u modulu SPI Parser, protože plní tu samou funkci, jen podle jiného nastavení. Nadřazené komunikační zařízení se sériovými linkami obsahuje 9 modulů SCI Master a 1 SCI Parser. Na základě přijatých dat se po sériových linkách odesílají data. V kapitole jsou uvedeny testy navržených modulů a test celého konceptu v simulaci.

V poslední kapitole je uvedeno testování výše popsaných navržených komponent, na reálném hardware - FPGA. Pro testování byl použit modul ADALM2000 spolu s programem Scopy, jehož výstupy jsou uvedeny v poslední kapitole práce. Byly popsány jen některé konfigurace, protože nelze obsáhnout v práci všechny. Dalším předmětem testováním by měl být test na reálném měniči. Je zřejmé, že zařízení se může v průmyslovém prostředí chovat méně stabilněji, než v prostředí laboratorním. Takové testování je v budoucnu plánováno a pro průmyslové využití je nezbytné.

I přes počáteční nesprávnosti při implementacích byly stanovené cíle práce splněny, otestovány a popsány. Při realizaci práce jsem mohl využít znalostí z předmětu BPC-LOS a taktéž mi realizace práce přinesla větší náhled do problematiky při návrhu hardware. Časová náročnost této práce byla větší, než jsem zpočátku čekal. Přesný čas realizace nelze vyčíslit, ale byl v řádu stovek hodin práce. Více času zabíralo testování navržených modulů, než samotné psaní zdrojového textu. Všechny zdrojové texty kódů ve VHDL byly psány mnou v celém rozsahu. Díky konzultacím ve firmě ELCOM, a. s. jsem mohl lépe pochopit problematiku ohledně pulzních měničů a jejich řízení.

Využití této práce externí firmou ELCOM, a. s. se předpokládá v budoucích projektech s buňkovými měniči. Nejprve bude nutné navrhnout desku plošných spojů, kde bude hradlové pole obsahující konfiguraci navrženou v této práci, poté bude následovat test s měničem. V případě, že se takový návrh osvědčí je možné předpokládat budoucí použití v práce v komerčních projektech.

Literatura

[1] IBRAHIM, D. *Designing Embedded Systems with 32-Bit PIC Microcontrollers and MikroC*. Oxford: Newnes, 2014. 359-442 s. ISBN 978-0-08-097786-7.

[2] PINKER, J., POUPA, M. *Číslicové systémy a jazyk VHDL*. Praha: 2009.

[3] PATOČKA, M. *Vybrané statě z výkonové elektroniky, Svazek I.* Brno: 2005, skriptum.

[4] PATOČKA, M. *Vybrané statě z výkonové elektroniky, Svazek II.* Brno: 2005, skriptum.

[5] PATOČKA, M. *Výkonová elektronika BVEL, 1. část - usměrňovače, střídavé měniče napětí*. Brno: 2010, skriptum.

[6] ALTERA CORPORATION. *Cyclone IV Device Handbook, Volume 1*. 2016.

[7] SIEMENS. *Medium-Voltage AC-Converter ROBICON Perfect Harmony*. 2009.

[8] TEXAS INSTRUMENTS INC. *TMS320x2833x, 2823x Enhanced Pulse Width Modulator (ePWM) Module Reference Guide*. 2009. Datasheet.

[9] TEXAS INSTRUMENTS INC. *TMS320x2833x, 2823x Serial Peripheral Interface (SPI) Reference Guide*. 2009. Datasheet.

[10] USACH M. *SPI Interface*. 2015. Application note.

[11] *ADALM2000 Overview* [online]. [cit 2021-12-19]. Dostupné z: `https://wiki.analog.com/university/tools/m2k`

[12] *FPGA KIT S CYCLONE IV EP4CE6 + USB BLASTER + IR OVLADAČ* [online]. [cit 2021-9-28]. Dostupné z: `https://www.hwkitchen.cz/fpga-kit-cyclone-iv-ep4ce6-usb-blaster-ir-ovladac/`

[13] *A Brief Overview of IGBT - Insulated Gate Bipolar Transistor* [online]. [cit 2021-11-13]. Dostupné z: `https://components101.com/articles/what-is-igbt-working-operation-symbol-and-types`

[14] *Pulse Width Modulation (PWM)* [online]. [cit 2021-11-12]. Dostupné z: `https://www.elprocus.com/pulse-width-modulation-pwm/`

[15] *SCI Overview* [online]. [cit 2021-12-13]. Dostupné z: `http://www.ece.utep.edu/courses/web3376/SCI%20Overview.html`

[16] *Serial Communication Protocols* [online]. [cit 2021-10-4]. Dostupné z: https://circuitdigest.com/tutorial/serial-communication-protocols

[17] *Střídač* [online]. [cit 2021-11-13]. Dostupné z: https://cs.wikipedia.org/wiki/St%C5%99%C3%ADda%C4%8D

[18] *Crossing Clock Domains in an FPGA* [online]. [cit 2022-03-28]. Dostupné z: https://www.nandland.com/articles/crossing-clock-domains-in-an-fpga.html

[19] *Serial Communication Methods - Synchronous Asynchronous* [online]. [cit 2022-04-03]. Dostupné z: https://pijaeducation.com/communication/serial-communication-methods-synchronous-asynchronous/

Seznam symbolů a zkratek

ePWM	Rozšířený generátor pulzně šířkové modulace
FPGA	Programovatelné hradlové pole
HDL	Jazyk pro popis Hardware
HW	Hardware
IDE	Vývojové prostředí
IGBT	Bipolární tranzistor s izolovaným hradlem
PWM	Pulzně šířková modulace
RTL	Resistor–transistor logic
SCI	Serial Communications Interface
SPI	Serial Peripheral Interface
SW	Software
USB	Universal Serial Bus
VHDL	VHSIC Hardware Description Language

Seznam příloh

A Obsah elektronické přílohy

- elektronicka_priloha
  - bakalarska_prace.pdf ..... Elektronická verze práce
  - video ..... Adresář se záznamy demonstrující funkčnost na reálném HW
    - FPGA_ePWM_test.mp4 ..... Video z testování popsaného v kapitole 7.1
    - FPGA_ePWM_duty_cycle_test.mp4 Video z testování modifikace střídy ePWM signálů
    - FPGA_SCI_test.mp4 ..... Video z testování popsaného v kapitole 7.2
  - vhdl ..... Adresář se zdrojovými soubory v jazyce VHDL
    - 1_rozsireni ..... Adresář obsahující soubory 1. rozšíření
      - design_epwm_spi.vhd ..... Reálný test 1. rozšíření v FPGA
      - epwm_spi.vhd ..... Vrcholová entita 1. rozšíření
      - spi_parser.vhd ..... Implementace SPI Parser
      - testbench_epwm_spi.vhd ..... Testovací soubor pro 1. rozšíření
      - testbench_spi_parser.vhd ..... Testovací soubor pro SPI Parser
    - 2_rozsireni ..... Adresář obsahující soubory 2. rozšíření
      - design_sci_spi.vhd ..... Reálný test 2. rozšíření v FPGA
      - sci.vhd ..... Vrcholová entita 2. rozšíření
      - sci_parser.vhd ..... Implementace SCI Parser
      - testbench_sci_spi.vhd ..... Testovací soubor pro SCI Parser
    - epwm ..... Adresář obsahující soubory modulu ePWM
      - design_epwm.vhd ..... Reálný test ePWM v FPGA
      - ePWM.vhd ..... Propojení všech modulů ePWM
      - ePWM_registers.vhd ..... Propojení ePWM s registry (jako paměť)
      - testbench_hbridge.vhd ..... Testovací soubor pro ePWM
      - testbench_symetricPWM.vhd ..... Testovací soubor pro ePW
      - submodules ..... Adresář obsahující submoduly ePWM
        - action_qualifier.vhd
        - counter_compare.vhd
        - dead_band.vhd
        - event_trigger.vhd
        - event_trigger_generator.vhd
        - pwm_chopper.vhd
        - time_base.vhd
        - trip_zone.vhd
    - sci ..... Adresář se soubory modulu SCI Master
      - sci_master.vhd ..... Implementace SCI Master
      - testbench_sci.vhd ..... Testovací soubor pro SCI Master
    - spi ..... Adresář se soubory modulu pro SPI Slave
      - spi_slave.vhd ..... Implementace SPI Slave
      - testbench_spi.vhd ..... Testovací soubor pro SPI Slave
    - test_spi+sci ..... Adresář se soubory pro testování SPI a SCI
      - design_spi+sci.vhd ..... Test SPI a SCI v hradlovém poli
      - segment_display.vhd ..... Dekodér pro sedmisegmentový displej