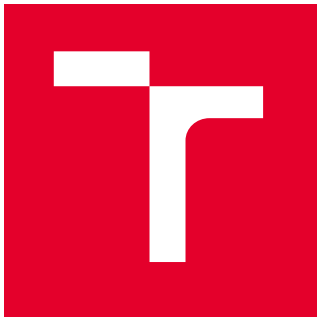


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

NÁVRH BEZDRÁTOVÉHO KOMUNIKAČNÍHO ROZHRANÍ S PC PRO EMULAČNÍ PLATFORMU URČENOU K MODELOVÁNÍ INTEGROVANÝCH OBVODŮ

DESIGN OF A WIRELESS COMMUNICATION INTERFACE WITH A PC FOR AN EMULATION PLATFORM FOR MODELING INTEGRATED CIRCUITS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Marek Benc**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Pavel Šteffan, Ph.D.**
SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Mikroelektronika**

Ústav mikroelektroniky

Student: Bc. Marek Benc ***ID:*** 203192

Ročník: 2 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Návrh bezdrátového komunikačního rozhraní s PC pro emulační platformu určenou k modelování integrovaných obvodů

POKYNY PRO VYPRACOVÁNÍ:

V první části se má diplomová práce zabývat vyhodnocením možností bezdrátového přenosu dat z emulační platformy, konkrétně z kitu s Xilinx zynq do PC a zpět. Tento přenos dat bude naprogramován v systému Xilinx Zynq. Součástí práce bude návrh a výroba elektronického modulu pro přenos dat. Data budou posílány z aplikace v PC, jejíž vytvoření bude také součástí práce. Data budou obsahovat hlavně nastavení konfigurace emulační platfomy pro konkrétní vývojový projekt nového integrovaného obvodu.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 24.5.2022

Vedoucí práce: doc. Ing. Pavel Šteffan, Ph.D.

doc. Ing. Lukáš Fujcik, Ph.D.
předseda rady studijního programu

ABSTRAKT

V design centru Onsemi v Rožnově pod Radhoštěm využívají pro ověření funkce a ladění některých integrovaných obvodů emulační platformu s FPGA, jejíž součástí je rozhraní modelující konfigurační pojistky pomocí fyzických spínačů. Pro čipy s velkým počtem pojistek je využití tohoto rozhraní pracné a náchylné na chyby, proto se tato práce zaobírá možnostmi vytvoření bezdrátového konfiguračního rozhraní, které by bylo ovládáno z PC.

KLÍČOVÁ SLOVA

integrovaný obvod, emulace, konfigurace, pojistky, komunikační linka, bezdrátová technologie, FPGA

ABSTRACT

In the design center of Onsemi in Rožnov pod Radhoštěm, they use an emulation platform based around an FPGA for verification and debugging tasks for certain integrated circuits, and it's equipped with an interface that models an IC's configuration fuses with physical switches. For ICs with large fuse counts, working with this interface is tedious and error-prone, which has lead to this thesis investigating the possibilities for creating a wireless configuration interface that would be controlled by a PC.

KEYWORDS

integrated circuit, emulation, configuration, fuses, communication line, wireless technology, FPGA

BENC, Marek. *Návrh bezdrátového komunikačního rozhraní s PC pro emulační platformu určenou k modelování integrovaných obvodů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky, 2022, 67 s. Diplomová práce. Vedoucí práce: doc. Ing. Pavel Šteffan, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Marek Benc
VUT ID autora:	203192
Typ práce:	Diplomová práce
Akademický rok:	20221/22
Téma závěrečné práce:	Návrh bezdrátového komunikačního rozhraní s PC pro emulační platformu určenou k modelování integrovaných obvodů

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 konzultantovi diplomové práce panu Ing. Andreji Čaplickému za trpělivost a praktické rady při vypracovávání projektu, a také vedoucímu diplomové práce panu doc. Ing. Pavlu Šteffanovi, Ph.D. za odborné vedení, konzultace a podnětné návrhy k práci.

Obsah

Seznam obrázků

Úvod

V design centru Onsemi v Rožnově pod Radhoštěm využívají pro ověření funkce a odladění digitální části smíšených analogovo-digitálních integrovaných obvodů emulační systémy. Jedná se o DPS obvody s programovatelným logickým členem, ke kterému jsou připojeny analogové prvky jako ADC a DAC převodníky, zesilovače, filtry, oscilátory a jiné, které odpovídají analogové části emulovaného integrovaného obvodu. Tato práce se konkrétně věnuje emulační platformě čipu pro korekci účiníku spínaného napájecího zdroje a jako programovatelný logický člen je v ní použit systém na čipu Xilinx Zynq 7000, který kombinuje ARMv7 procesor a FPGA jádro.

Pro integrované obvody je v rámci výroby typické provádět konfiguraci pomocí tavných pojistek, u kterých přepálením dochází ke změně vlastností obvodu, např. doladění odchylek elektrických parametrů vyrobené součástky (angl. *trimming*). Lze měnit i samotnou funkci obvodu, kde přepálením pojistek dochází ke změnám ve vyhodnocovacích obvodech, na základě kterých vykonává obvod svou funkci.

Pro přepálení pojistek na daném čipu mají v Onsemi přípravek, do kterého je čip vložen, a pomocí konfigurační GUI aplikace na počítači je provedena volba konfigurace, a dle ní jsou pojistky přepáleny. Z uživatelského hlediska probíhá konfigurace formou výběru možností ze sady rozbalovacích seznamů a zaškrtávacích polí, kterých hodnoty jsou programem převedeny do bitového formátu odpovídajícího tavným pojistkám čipu.

Tuto konfiguraci lze samozřejmě provádět i na emulační platformě. Problémem u konkrétní platformy je, že analogie konfiguračních pojistek je realizována pomocí fyzických spínačů odpovídajících jednotlivým pojistkám, a těchto spínačů jsou řádově stovky (jeden konfigurační modul má 128 spínačů a tyto moduly lze řadit do série). Znamená to, že nastavování je pracné, a může lehce dojít k chybě, což pro obvod, který je součástí napájecího zdroje, může znamenat i riziko požáru.

Z tohoto důvodu byl stanoven požadavek na vytvoření konfiguračního modulu, který by fungoval podobně jako přípravek pro vyrobený čip s připojením na počítač a komfortní konfigurací pomocí GUI aplikace. Jelikož se jedná o obvod, který pracuje se síťovým napájecím napětím, a mohlo by teoreticky dojít k chybě, kde by se síťové napětí objevilo na modulu, byl stanoven požadavek na bezdrátové řešení, aby v případě výskytu takové chyby nedošlo k poškození drahé výpočetní techniky.

Tato práce se věnuje teoretickému rozboru problematiky návrhu bezdrátového konfiguračního modulu, který by moduly se spínači nahradil, a jeho následnou výrobou a programováním.

1 Rozbor problematiky bezdrátové komunikace

V této kapitole je proveden rozbor bezdrátových technologií, které by se pro realizaci konfiguračního modulu daly využít.

1.1 Základy bezdrátové komunikace

Existuje několik možných principů, pomocí kterých lze mezi elektrickými obvody bezdrátově přenášet informace. Jedním z nejstarších je pomocí elektromagnetické indukce, kde přivedením střídavého napětí na cívku vzniká střídavé magnetické pole, které má za následek vznik střídavého elektrického pole v okolních vodičích. Modulací střídavého napětí na cívce, případně proudu odebíraného přes cívku, lze pomocí vzniklého magnetického pole bezdrátově přenášet informaci.

Problémem této technologie je krátká vzdálenost dosahu, proto se typicky využívá v rámci jednoho obvodu na galvanické oddělení signálů. Elektromagnetická indukce je využívaná častěji pro přenos energie než pro přenos informací. Typickou aplikací, která kombinuje přenos energie a informací, je bezdrátové nabíjení mobilních telefonů. Pro tento účel byla zavedena norma Qi, která dovoluje nabíjet zařízení ve vzdálenosti 4 cm od nabíjecí podložky. Pro přenos informací směrem z podložky do telefonu je využito klíčování frekvenčním posuvem nabíjecího magnetického pole, pro směr z telefonu do podložky zase amplitudové modulace odebíraného proudu indukovaného napětí. [1]

Pro přenos na větší vzdálenosti se typicky využívají elektromagnetické vlny, existenci kterých roku 1863 popsal James Clerk Maxwell, a experimentálně ji v roce 1887 potvrdil Heinrich Hertz. [2]

Vlnová délka určuje možnosti šíření elektromagnetické vlny. Pro kmitočty do 1 GHz se zemská atmosféra chová téměř jako vakuum, pro vyšší kmitočty dochází k útlumu, který je daný atmosférickými srážkami a vlastní rezonancí molekul atmosférických plynů. Pro kmitočty do 3 GHz dochází také k odrazu vlnění od zemského povrchu a vzniku prostorové vlny. Vlna se také může šířit po zemském povrchu mechanizmem difrakce, tento mechanizmus lze v praxi využít pro kmitočty do zhruba několika jednotek MHz a pro kmitočty do zhruba několika desítek MHz dochází k odrazu vlny od ionizované oblasti atmosféry a vzniku ionosférické vlny. [3]

Na vlnové délce signálu také závisí míra průhlednosti různých materiálů, tato problematika je obzvlášť důležitá u optického přenosu, u kterého se vlnová délka blíží rozměrům jednotlivých molekul a atomů materiálů, a dochází k molekulární

absorpci. Tento jev se dá využít pro zkoumání chemické kompozice materiálů (optická spektroskopie), ale z pohledu komunikace to hlavně znamená, že musí být vhodně voleny materiály, aby byl přenos možný, případně vhodně volena vlnová délka tak, aby byla komunikace možná ve stanovených podmínkách. [4]

Dalším důležitým faktorem, který silně závisí na vlnové délce, je způsob generace a zachycení záření. Pro nižší kmitočty až do několika GHz se typicky využívají lineární antény, které lze chápat jako soubor mnoha různě položených elementárních elektrických dipólů, pro centimetrové a kratší vlny se využívají antény plošné, které lze považovat za soubor Huygensových zdrojů vlnění. [3] Pro vlnové délky pod milimetr se typicky už nevyužívají antény, ale kombinace zářícího a přijímacího prvku, např. luminiscenční dioda jako zdroj, u kterého vzniká elektromagnetické záření s vlnovou délkou odpovídající energii šířky zakázaného pásu polovodiče, a fotodioda, u které dopadem elektromagnetického záření do oblasti prostorového náboje dochází ke vzniku párů elektron-díra, které jsou působením elektrického pole od sebe oddáleny a dají se detekovat jako proud diodou v závěrném směru, případně jako napětí na svorkách, pokud je dioda zapojena na prázdno. [4]

Množství přenášených informací rovněž závisí na vlnové délce a na zvoleném druhu modulace vlnění. Pod pojmem modulace rozumíme úpravu nosného signálu určitým způsobem tak, aby bylo možné přenášenou informaci na přijímači zrekonstruovat. Tradičním příkladem je modulace amplitudová (AM), u které dochází ke změně amplitudy nosného signálu na základě amplitudy přenášeného signálu, nebo kmitočtová (FM) modulace, u které na základě amplitudy přenášeného signálu dochází ke změně kmitočtu. Následkem modulace je rozšíření původně čárového spektra nosného signálu o postranní pásma, tato pásma nesou samotnou přenášenou informaci a jejich šířka typicky koresponduje s množstvím a kvalitou přenášených informací. Kratší vlnové délky (vyšší kmitočty) dovolují přenášet širší kmitočtová pásma nebo více pásem v rámci jednotlivých kanálů. [5]

Možnost vysílání na jednotlivých kmitočtových pásmech je přísně regulována. Na mezinárodní úrovni je přidělování oprávnění pro vysílání na jednotlivých kmitočtových pásmech prováděno Mezinárodní telekomunikační unií, a ta má pro účely průmyslových, vědeckých a zdravotnických (ISM) aplikací vyhrazena určitá kmitočtová pásma, pro která není zapotřebí mít výhradní licenci. Konkrétní hodnoty těchto pásem se u některých zemí liší, ale pásmo 2,4 GHz lze využívat bez licence ve všech zemích. [5] Stejně tak nejsou kladeny restrikce na využití optické komunikace, jelikož zdroje světla jsou všude kolem nás, a tak se tato práce věnuje rozboru možností komunikace na kmitočtovém pásmu 2,4 GHz a pomocí optické komunikace.

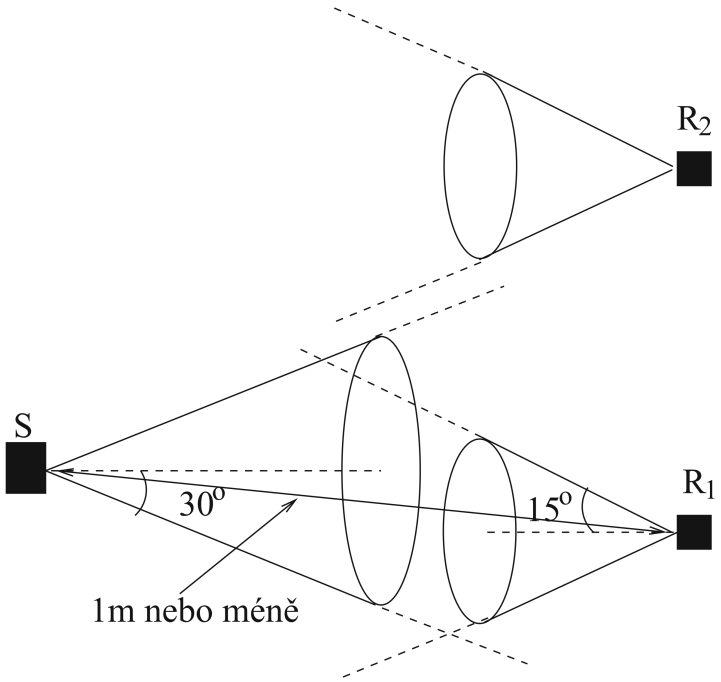
1.2 Komunikační normy krátkého dosahu

Jelikož není cílem práce navrhnout úplně nový způsob komunikace, byl proveden rozbor existujících řešení pro komunikaci na krátkou vzdálenost opticky nebo pro kmitočtové pásmo 2,4 GHz.

1.2.1 IrDA

Norma IrDA (Infrared Data Association) popisuje poloduplexní bezdrátovou komunikaci za použití infračerveného světla o vlnové délce 859-900 nm. Maximální vzdálenost komunikačních prvků je stanovena na 1 m, s maximální rychlostí přenosu 4 Mbps. Je to technologie velice levná a odolná vůči elektromagnetickému rušení a je tak vhodná pro průmyslové, vojenské a letecké použití.

Typické využití IrDA je jako náhrada drátové sériové linky RS232, pro tyto účely lze využít protokolové řadiče MCP21xx sloužící jako přímá náhrada budících obvodů linky RS232, konektor DB9 lze pak nahradit optickým transceiverem. Tyto transceivery jsou ale omezeny z hlediska možností prostorového uspořádání pro komunikaci, vysílání je omezeno na prostor kruhového kuželu s 30° úhlem pláště od osy souměrnosti, přijímání je omezeno na prostor kruhového kuželu s 15° úhlem pláště od osy souměrnosti. Příklad uspořádání je znázorněn na obr. 1.1, kde přijímač R_1 dokáže přijímat data z vysílače S, ale R_2 už ne. [6]

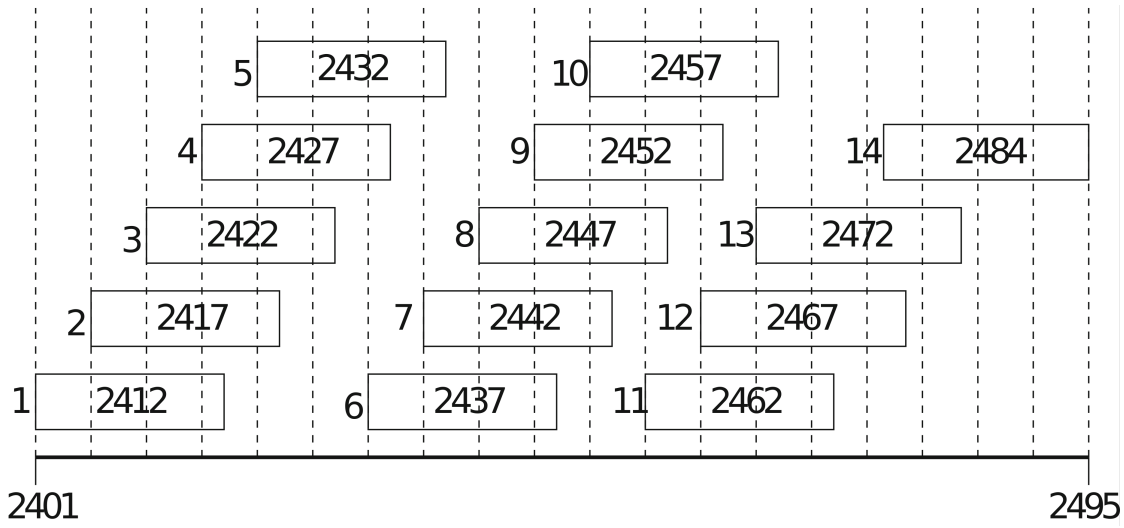


Obr. 1.1: Infračervené komunikační kužely (zdroj: [6])

Komunikace na IrDA lince má klient-server architekturu, kde primární zařízení (klient) provádí identifikaci všech sekundárních zařízení (serverů) ve svém komunikačním kuželu, následně se svoleným sekundárním zařízením jedná o podmínkách komunikace, dle kterých se pak komunikace přepíná z výchozí přenosové rychlosti 9600 baud na nejvyšší společnou přenosovou rychlost. Sekundární zařízení může poskytovat několik nezávislých serverových aplikací, které si klient volí na základě jejich identifikačního kódu. [6]

1.2.2 WiFi

Jedná se o technologii bezdrátového Ethernetu, založena na normách IEEE 802.11, která slouží na rozšíření existující drátové sítě LAN o bezdrátové prvky přes přístupový bod (angl. *access point*, AP). Původní varianta WiFi, IEEE 802.11b z roku 1999, pracuje na kmitočtovém pásmu 2,4 GHz s šířkou kanálu 22 MHz a rychlostí přenosu 5 až 11 Mbps. Norma definuje 14 překrývajících se kanálů vzájemně od sebe posunutých o 5 MHz (obr. 1.2). Pro zajištění spolehlivého provozu by měly být využity kanály od sebe posunuty alespoň o 30 MHz, tomuto vyhovuje kombinace kanálů 1, 6 a 11. Plyne z toho omezení na 3 přístupové body v rámci jedné místnosti. [6]



Obr. 1.2: Rozložení WLAN kanálů na kmitočtovém pásmu 2,4 GHz (zdroj: [6])

V roce 1999 vyšla také norma IEEE 802.11a, nazývaná WiFi5, která pracuje na kmitočtovém pásmu 5 GHz a využívá efektivnější způsob kódování zvaný OFDM (angl. *Orthogonal Frequency Division Multiplexing*), který poskytuje teoretickou maximální hodnotu přenosové rychlosti 54 Mbps. Šířka kanálu je 40 MHz, rozsah využívaných kmitočtových pásem je 5,15 GHz až 5,25 GHz, 5,25 GHz až 5,35 GHz a 5,725 GHz až 5,825 GHz. V Evropě lze volně využít spodní i střední pásmo, které

dohromady poskytují 8 nepřekrývajících se kanálů. Šířka pásma samotného signálu je 20 MHz. [6]

Vyšší pracovní kmitočet této technologie znamená ale i kratší vlnovou délku a z toho plynoucí vyšší tlumení signálu, přístupové body typu IEEE 802.11a poskytují signál typicky jen na čtvrtinu plochy poskytovanou IEEE 802.11b, kvůli čemu se na spotřebitelském trhu tato technologie neujala. To byl jeden z důvodů, proč v roce 2003 vyšla norma IEEE 802.11g, která využívá stejné OFDM kódování jako WiFi5, a poskytuje tak teoretickou maximální hodnotu přenosové rychlosti 54 Mbps, pracuje ale na kmitočtovém pásmu 2,4 GHz, jedná se o kompromis mezi délkou dosahu a počtem dostupných kanálů. [6]

V roce 2009 vyšla norma IEEE 802.11n, která pro dosažení rychlejšího přenosu dat využívá více než jednu anténu (nanejvýš 4 vysílací a 4 přijímací antény) pro přenášení více datových toků najednou, a využívá i 2,4 GHz i 5 GHz pásmo. [6]

WiFi síť může fungovat v infrastrukturním nebo nezávislém režimu. Pro infrastrukturní režim je charakteristické zapojení bezdrátových zařízení do existující kabelové sítě přes přístupové body, každý z přístupových bodů poskytuje základní sadu služeb (angl. *Basic Service Set*, BSS) a je identifikován MAC adresou BSSID. Jelikož jsou MAC adresy pro lidi těžce zapamatovatelné, je poskytnut také textový identifikátor SSID, pomocí kterého lze od sebe jednotlivé přístupové body odlišit a připojit se k nim. Identifikátor SSID je typicky přístupovým bodem vysílán, z bezpečnostních důvodů lze ale tento mechanizmus vypnout, v tom případě musí klient tento identifikátor předem znát, aby se mohl připojit. V případě nezávislého režimu se síť skládá ze skupiny bezeztrátových zařízení, která musí být ve vzájemném dosahu, a spolu tvoří nezávislou základní sadu služeb (IBSS). U takovéto sítě vysílá identifikátor SSID nejdříve první zařízení v síti, následně v pseudonáhodném pořadí ostatní zařízení. [6]

1.2.3 Bluetooth

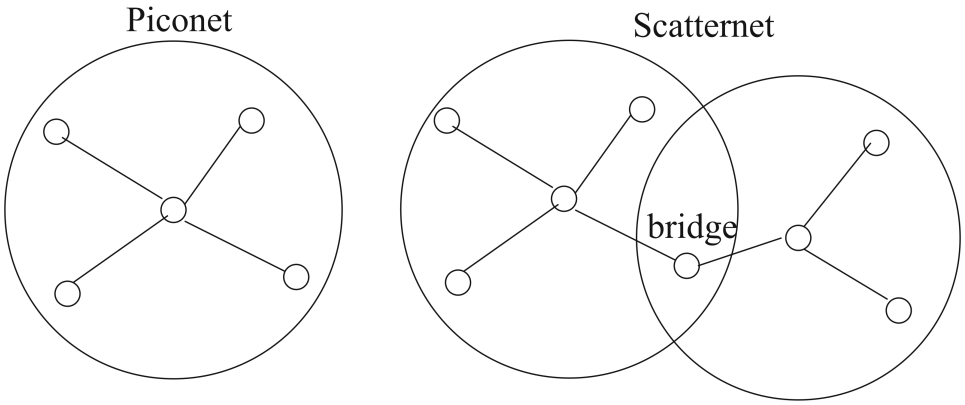
Bluetooth, původně vyvinut společností Ericsson jako přímá náhrada kabelového připojení periferií, je rádiová komunikační norma krátkého dosahu fungující na kmitočtovém pásmu 2,4 GHz, definující 79 kanálů o šířce 1 MHz na kmitočtech 2,402 GHz až 2,48 GHz, mezi kterými dochází k nepřetržitému cyklickému přepínání (1600 krát za vteřinu), s maximální rychlostí přenosu 3 Mbps, určena pro vytvoření osobní sítě (angl. *Personal Area Network*, PAN). Norma je vyvíjena skupinou *Bluetooth Special Interest Group*. [5], [6]

Bezdrátová zařízení Bluetooth se na základě vysílacího výkonu dělí do třech tříd: Třída 1 s maximálním vysílacím výkonem 100 mW a dosahem 100 metrů, třída 2 s maximálním vysílacím výkonem 2,5 mW a dosahem 10 metrů a třída 3

s maximálním vysílacím výkonem 1 mW a dosahem 1 metr. [5]

V jedné Bluetooth síti se může nacházet maximálně 8 zařízení ve hvězdové topologii. Ve středu topologie se nachází nadřízené zařízení (angl. *master*) a okolní zařízení jsou podřízená (angl. *slave*). Takováto síť je nazývaná *piconet* a pomocí přemosťujícího zařízení lze propojit více piconetů v *scatternet* (obr. 1.3). Přemosťující zařízení může být buď v obou piconetech podřízené, nebo v jednom piconetu podřízené a v druhém nadřízené. [6]

Komunikace s podřízenými zařízeními je po komunikačním kanálu časově multiplexována, s časovým intervalem 625 µs pro komunikaci s jednotlivými zařízeními. Pro formát komunikace existuje celá řada profilů (nad 40) [5], které popisují, o jaký druh zařízení se jedná a jaké služby poskytuje. Každé zařízení má unikátní MAC (*Media Access Control*) adresu, pomocí které dokáže nadřízené zařízení jednotlivá podřízená zařízení identifikovat. Pomocí protokolu SDP (*Service Discovery Protocol*) může nadřízené zařízení zjistit, jaké služby konkrétní podřízené zařízení poskytuje. [6]



Obr. 1.3: Bluetooth sítě typu piconet a scatternet (zdroj: [6])

Důležitou vlastností Bluetooth je možnost kombinovat a využívat zařízení od různých výrobců, tzv. *interoperabilita*. Proto musí každé Bluetooth zařízení implementovat základní sadu protokolů a držet se předem definovaných profilů služeb. Každá služba má unikátní identifikátor UUID, pomocí kterého ji může nadřízené zařízení rozpoznat. Zařízení může také poskytovat více služeb (a více instancí dané služby) najednou pomocí L2CAP (*Logical Link Control Adaptation Protocol*), který dokáže vytvářet logické spoje mezi nadřízeným zařízením a jednotlivými aplikacemi na podřízeném zařízení. [6]

1.2.4 Bluetooth Low Energy

Bluetooth Low Energy je rádiová komunikační norma krátkého dosahu fungující na kmitočtovém pásmu 2,4 GHz definující 40 kanálů s šířkou pásma 2 MHz a maximální přenosovou rychlostí 1 Mbps, která se stala součástí specifikace Bluetooth ve verzi 4.0. Jedná se o normu, která není zpětně ani dopředně kompatibilní s klasickou technologií Bluetooth, lze ji ale implementovat na zařízení společně s klasickou technologií, takovým zařízením se říká dvourežimová (angl. *dual-mode devices*). Technologie je určena primárně pro ovládací a monitorovací prvky umístěné na krátkou vzdálenost od přijímače, se zaměřením na nízkou spotřebu. [7]

Architektura Bluetooth Low Energy je podobná architektuře tradičního Bluetooth, využívá ale jednodušší modulaci (gaussovské klíčování frekvenčním posuvem, GFSK) a podporuje také vysílání informací bez nutnosti navazování spojení. Tři kanály ze 40 definovaných normou Bluetooth Low Energy jsou tzv. reklamní kanály (angl. *advertising channels*), které jsou využívány na identifikaci zařízení a vytváření spojení, ale také na jednosměrné zasílání informací, např. hodnot ze senzoru. Pro získání těchto informací není zapotřebí explicitně vytvářet se zařízením spojení, provoz může fungovat v tzv. odposlechovém (angl. *scanning*) režimu. Poloha reklamních kanálů na spektru byla zvolena tak, aby nezasahovala do WLAN kanálů 1, 3 a 11, na kterých se typicky nachází přístupové body WiFi. Pro oboustrannou komunikaci lze využít ostatních 37 kanálů. [7]

Další typickou vlastností Bluetooth Low Energy je, že podřízena zařízení typicky tráví většinu svého času v režimu spánku, ze kterého se v časovém intervalu stanoveném nadřazeným zařízením pravidelně probouzí a kontrolují, zdali pro ně nadřízené zařízení něco vysílá. [7]

Podobně jako klasická technologie Bluetooth i Bluetooth Low Energy využívá pro definici služeb profily. Charakteristickým pro tuto technologii je profil GATT (*Generic Attribute Profile*), který definuje způsob přijímání a zasílání krátkých datových položek zvaných *atributy*. [7]

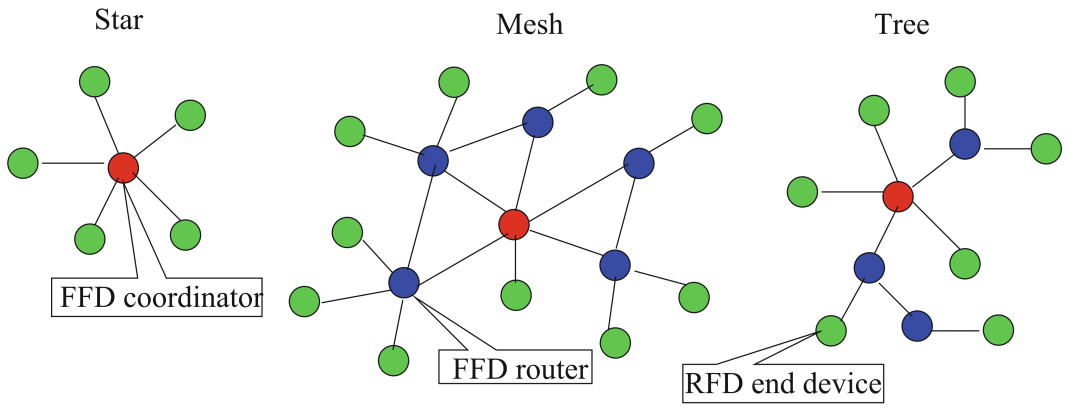
1.2.5 Zigbee

Základem komunikační normy Zigbee je norma IEEE 802.15.4, vydaná v roce 2003, která poskytuje popis fyzické a MAC (*Media Access Control*) vrstvy pro nízkoenergetickou bezdrátovou osobní síť. Tato norma využívá několik kmitočtových pásem, konkrétně jeden 2 MHz kanál na kmitočtu 868,3 MHz s přenosovou rychlostí 20 Kbps, deset 2 MHz kanálů na kmitočtech 902 MHz až 928 MHz s přenosovou rychlostí 40 Kbps a šestnáct 5 MHz kanálů na kmitočtovém pásmu 2,4 GHz s rychlostí přenosu 250 Kbps. Výhodou kanálů na nižších kmitočtech je značně delší dosah.

Norma podporuje 2 druhy adresování na MAC vrstvě, konkrétně pomocí 64-bitové nebo 8-bitové adresy. [6]

Zigbee, vyvíjen Zigbee aliancí, poskytuje nad normou IEEE 802.15.4 síťové, bezpečnostní a aplikační protokoly. Podobně jako u Bluetooth jsou normou Zigbee definovány aplikační profily, které určují druh zpráv a způsob komunikace pro zařízení implementující určitý profil. [6]

Norma definuje tři druhy zařízení, konkrétně zařízení plné funkce (angl. *Full Function Device*, FFD), zařízení zredukované funkce (angl. *Reduced Function Device*, RFD) a zařízení pro koordinaci sítě (angl. *Network Coordinator*). Topologie sítě může být hvězdová, stromová, nebo pletivová (obr. 1.4) s maximálním počtem síťových prvků 254. Každá síť obsahuje alespoň jedno FFD a kombinaci FFD a RFD zařízení. Jedno z FFD zařízení funguje v roli síťového koordinátora, ostatní jako směrovače, které umožňují prostorově rozšířit síť nad vzdálenost dosahu koordinátora, a v případě duplicity poskytují náhradní trasy komunikace pro případ selhání článku sítě. [6]



Obr. 1.4: Topologie sítě Zigbee (zdroj: [6])

Zařízení RFD fungují typicky v režimu spánku, ze kterého se pravidelně probouzejí za účelem komunikace. Tento interval lze pro snížení latence nastavit vysíláním majákových rámců (angl. *Beacon frame*), probouzení RFD se s intervalem vysílání těchto rámců sesynchronizuje. Další podporované druhy komunikačních rámců jsou *MAC rámce* sloužící na peer-to-peer MAC konfiguraci, *potvrzovací rámce* sloužící na potvrzení úspěšného přijmu rámce a *datové rámce* pro datovou komunikaci. Jeden datový rámec dokáže přenést 127 bajtů uživatelských dat. [6]

1.3 Moduly, řadiče, brány

Pro komunikaci s výše popsanými normami existují moduly a řadiče, které lze využít pro rozšíření digitálních systémů o schopnost zapojit se do daných sítí a komunikovat po nich. Existují jak brány, které převádí bezdrátový provoz na provoz kompatibilní se starší drátovou normou, tak integrované moduly, na kterých lze spouštět vlastní kód a pracovat s bezdrátovou sítí na softwarové úrovni.

Byl proveden rozbor několika modulů dostupných na trhu v čase psaní práce.

1.3.1 Infračervené řadiče MCP21XX

Jedná se o rodinu řadičů od společnosti Microchip pro bezdrátovou komunikaci pomocí infračerveného záření dle normy IrDA. Dělí se na dvě základní kategorie, a to na jednoduché převodníky MCP212X, které přímo převádějí UART signály na elektrické impulsy pro infračervený transceiver a zpět, a na protokolové řadiče MCP2140 a MCP215X, které pracují na linkové a prezenční vrstvě IrDA normy. [8]

Převodníky MCP2120 a MCP2122 pracují na fyzické vrstvě IrDA normy a využívají UART rozhraní bez hardwarového řízení toku (jen RX a TX signály). Hardwarové řízení toku se u UART rozhraní využívá typicky při kombinaci rychlejšího a pomalejšího zařízení, aby mohl být provoz pozastaven, pokud by pomalejší zařízení už nestíhalo data zpracovat. U IrDA se jedná o poloduplexní komunikaci, tak je zapotřebí dbát při využití těchto řadičů na pečlivé dodržení protokolů linkové vrstvy, aby nedošlo ke ztrátě dat. [9], [6]

Hlavním rozdílem mezi MCP2120 a MCP2122 je způsob odvození přenosové rychlosti. Zatím co MCP2122 má jeden digitální hodinový vstup a přenosová rychlost je daná kmitočtem tohoto signálu poděleným 16, MCP2120 má vývody pro zapojení externího krystalu a digitální rozhraní (hardwarové i softwarové) pro nastavení kmitočtové děličky pro odvození přenosové rychlosti. Převodník MCP2122 má stejné rozložení vývodů jako Agilent HSDL-7000 a je s ním zpětně kompatibilní. [10], [11]

Protokolové řadiče MCP215X mají podobně jako MCP2120 vývody pro zapojení externího krystalu a nastavení přenosové rychlosti, hodnota krystalu je ale výrobcem fixně stanovena na 11,0592 MHz a vstupy pro nastavení přenosové rychlosti nastavují jen rychlost UART rozhraní, přenosovou rychlost infračerveného signálu si řadiče odvozují samy. Řadiče také poskytují hardwarové řízení toku pro UART rozhraní, podporu protokolů linkové vrstvy a části protokolu IrCOMM poskytující emulaci 9 - vodičové sériové linky RS232. Tyto řadiče fungují v režimu sekundárního zařízení a liší se od sebe způsobem řízení a interpretace signálů hardwarového řízení. Zatímco MCP2150 emuluje připojení typu *null modem* pro zapojení k zařízením typu *Data Terminal Equipment*, řadič MCP2155 emuluje modemové připojení

vhodné pro připojení k zařízením typu *Data Communicating Equipment*. V praxi to znamená, že MCP2150 signalizuje vytvoření spojení na vývodu CD (*Carrier Detect*) a MCP2155 na vývodu DSR (*Data Set Ready*). [12], [13]

Pro aplikace s požadavkem na nízký proudový odběr je výhodný protokolový řadič MCP2140, který je podobný řadiči MCP2155, ale funguje jen na výchozí přenosové rychlosti 9600 a v případě nepřenášení dat automaticky přechází do úsporného režimu s minimální proudovou spotřebou (řádově desítky µA). [14]

Všechny zmíněné součástky fungují s napájecím napětím 3,3 V nebo 5 V (maximální napájecí napětí 6,5 V) a odebírají proud řádově v jednotkách až desítkách mA. Převodník MCP2120 je dostupný v 14-vývodovém PDIP nebo SOIC pouzdru [10], MCP2122 v 8-vývodovém PDIP nebo SOIC pouzdru [11] a protokolové řadiče MCP2140, MCP2150 a MCP2155 v 18-vývodovém PDIP nebo SOIC pouzdru nebo 20-vývodovém SSOP pouzdru. [12], [13], [14]

1.3.2 Moduly ESP32

U platformy ESP32 se jedná o systémy na čipu (angl. *System on Chip*, SoC) čínské firmy Espressif obsahující 32-bitový procesor harvardské architektury XTensa LX6, řadiče sběrnic I^2C, I^2S, SPI, CAN, Ethernet, eMMC/SDIO a UART, převodníky DAC a ADC, PWM generátory pro luminiscenční diody a motory, Hallovu sondu a také obvody pro bezdrátovou komunikaci na kmitočtovém pásmu 2,4 GHz technologií Bluetooth, Bluetooth Low Energy (dvourežimové zařízení) a WiFi dle norem IEEE 802.11 b/g/n, v provedení QFN pouzdra s 49 vývody. [15]

Pro jednoduché zapojení na DPS poskytuje firma Espressif integrované moduly obsahující SoC ESP32, paměť FLASH (případně PSRAM), integrovanou anténu nebo přípojku na anténu a potřebné pasivní komponenty a stínící prvky pro zajištění správné funkce a elektromagnetické kompatibility obvodu. Tyto moduly, konkrétně ESP32-WROOM-32 a ESP32-WROVER, mají všechny signálové vývody na hranách modulu, ze spodní strany mají jen zemnící plošku, jejíž zapojení není pro plnohodnotnou funkci modulů vyžadováno, a tak je lze pájet i ručně. [16], [17]

Jedná se o univerzální programovatelné moduly, s kterými se pracuje podobně jako s mikrokontrolérem. Návrhář výsledného zařízení si připraví svůj vlastní program, který do modulu uvnitř zařízení nahraje, v případě modulů ESP32-WROOM-32 a ESP32-WROVER je to přes UART rozhraní při startu modulu v programovacím režimu, čehož lze dosáhnout uzemněním vývodu GPIO0. Pokud není vývod GPIO0 zapojen, vnitřní pull-up rezistor ho nastaví do hodnoty logické 1, což znamená spuštění programu z FLASH paměti na sběrnici SPI. [16], [17]

Moduly ESP32-WROOM-32 a ESP32-WROVER jsou certifikovány *WiFi aliancí* i skupinou *Bluetooth Special Interest Group* a podporují WiFi dle norem IEEE 802.11

b/g/n se zabezpečením WPA a WPA2, a Bluetooth (tradiční i Low Energy) dle Bluetooth specifikace 4.2. [18], [19], [20], [21]

Fyzické rozměry modulu ESP32-WROOM-32 jsou 18 mm šířka, 25,50 mm délka a 3,1 mm výška. Pro ESP32-WROVER je šířka 18 mm, délka 31,40 mm a výška 3,3 mm. Oba moduly mají 39 vývodů. Moduly obsahují 4 MB FLASH čip a ESP32-WROVER obsahuje také 4 MB Pseudo-SRAM čip, oba čipy jsou připojeny na SPI sběrnici. Moduly pracují na napájecím napětí 3,3 V (rozsah 3,0 V až 3,6 V pro ESP32-WROOM-32, 2,3 V až 3,6 V pro ESP32-WROVER) s maximálním proudovým odběrem 500 mA. [16], [17]

1.3.3 Modul Silicon Labs MGM220P

Jedná se o integrovaný modul americké společnosti Silicon Labs založen na SoC EFR32MG22. Daný systém na čipu obsahuje procesorové jádro ARM Cortex-M33, 512 kB programové FLASH paměti, řadiče sběrnic USART, EUART a I^2C, čítače, časovače, generátory PWM, systém pro reakci modulu na elektrické podněty bez softwaru *Peripheral Reflex System*, a také obvody pro bezdrátovou komunikaci na kmitočtovém pásmu 2,4 GHz technologií Bluetooth Low Energy a Zigbee v režimu RFD. [22], [23]

Podobně jako u ESP32 modulů obsahuje modul MGM220P všechny potřebné pasivní i stínící prvky pro zajištění správné funkce a elektromagnetické kompatibility obvodu. Obsahuje také vestavěnou anténu. Modul má všechny vývody ze spodní strany, takže není možné ho pájet ručně. Vývody lemují okraje modulu, takže lze alespoň pomocí testovacích plošek jednoduše provést elektrický test pro odhalení zkratů. [23]

Stejně jako u ESP32 běží na modulu program vytvořen návrhářem výsledného zařízení. Programování se provádí pomocí dvouvodičového rozhraní *Serial Wire Debug* (SWD), pomocí kterého se přistupuje k ladícímu rozhraní ARM. Toto rozhraní je součástí specifikace *ARM Debug Interface v5*. [22], [23], [24]

Modul je certifikován skupinou *Bluetooth Special Interest Group* a podporuje Bluetooth Low Energy dle Bluetooth specifikace 5.2. Tato certifikace se konkrétně vztahuje na hardwarový řadič fyzické vrstvy a lze ho kombinovat s certifikovaným softwarovým řadičem linkové vrstvy *Wireless Gecko Link Layer* a hostitelským řadičem *Wireless Gecko Host* pro vytvoření plně certifikovaného Bluetooth Low Energy zařízení. [23], [25], [26], [27]

Fyzické rozměry modulu MGM220P jsou 12,9 mm šířka, 15 mm délka a 2,2 mm výška. Počet vývodů je 31. Dovolený rozsah napájecích napětí je 1,71 V až 3,8 V, s typickým proudovým odběrem kolem 5 mA při přijímání a 10 mA při vysílání pro napájecí napětí 3,0 V. [23]

1.3.4 Modul Lantronix xPico 250

Jedná se o modul typu síťové brány americké společnosti Lantronix v řadě xPico 200, podporující 10/100 Mbps Ethernet s rozhraním RMII (pro připojení k externímu řadiči fyzické vrstvy), WiFi dle norem IEEE 802.11 a/b/g/n a kombinaci Bluetooth a Bluetooth Low Energy (dvourežimové zařízení). Na modulu dvě U.FL přípojky pro antény, jedna pro WiFi a druhá pro Bluetooth. Modul má také rozhraní UART, SPI Master, SPI Slave, a USB rozhraní s podporou režimu zařízení i hostitele. [28]

Pro WiFi je podporován režim přístupového bodu i klientského zařízení a tyto režimy mohou oba fungovat ve stejný čas. Modul je certifikován *WiFi aliancí* i skupinou *Bluetooth Special Interest Group* a podporuje WiFi dle norem IEEE 802.11 a/b/g/n se zabezpečením WPA, WPA2 a WPA3 a Bluetooth (tradiční i Low Energy) dle Bluetooth specifikace 5.0. [29], [30], [31]

Na rozdíl od modulů ESP32 a MGM220P si modul xPico 250 nevyžaduje uživatelský program, nedílnou součástí modulu je firmware s konfiguračním rozhraním, pomocí kterého se nastavuje mapování komunikačních kanálů. Jsou dostupná dvě konfigurační rozhraní: rozhraní příkazové řádky (CLI) přes UART a webové rozhraní přes WiFi. Tato rozhraní jsou navržena tak, aby byla uživatelsky přívětivá, ale podporují i možnost efektivní strojové konfigurace pomocí XML konfiguračních dat. Firmware je chráněn technologií *Secure Boot*, která dovoluje spustit jen kód s vhodným digitálním podpisem. Tato technologie pomáhá i se spolehlivostí aktualizací, jsou aplikovány pouze v případě, že byly do modulu přeneseny bezchybně. [28], [29]

Modul má všechny vývody ze spodní strany, takže není možné ho pájet ručně. Signálové vývody lemují okraje modulu, ve středu modulu jsou jen zemnící plošky, takže lze alespoň pomocí testovacích plošek jednoduše provést elektrický test pro odhalení zkratů u signálových vývodů. Fyzické rozměry modulu jsou 17 mm šířka, 25 mm délka a 1,78 mm výška. Signálových plošek je na modulu 66. Dovolený rozsah napájecích napětí je 3,15 V až 3,45 V, s typickým proudovým odběrem kolem 600 mA při startu, 123 mA v klidovém režimu v režimu přístupového bodu a 66 mA bez přístupového bodu, 30 mA v režimu spánku, a 380 mA při vysílání. [28]

2 Návrh konfiguračního modulu

Emulační platforma má tři konektory, do kterých by se dal bezdrátový konfigurační modul zapojit. Jedná se o dvouřadové 12-vývodové konektory X5 a X6 a jednořadový 6-vývodový konektor X7. Na všech třech konektorech je dostupná zem a napájecí napětí 5 V s maximálním proudovým odběrem 1 A (případně 3 A po výměně napájecího modulu), ostatní vývody vedou na vstupy a výstupy digitálních izolátorů ADuM160N, které pomocí elektromagnetické indukce provádí galvanické oddělení signálů od zbytku desky. Izolátory dokážou data přenášet rychlostí 150 Mbps, s propagačním zpožděním řádově v desítkách nanosekund. [32]

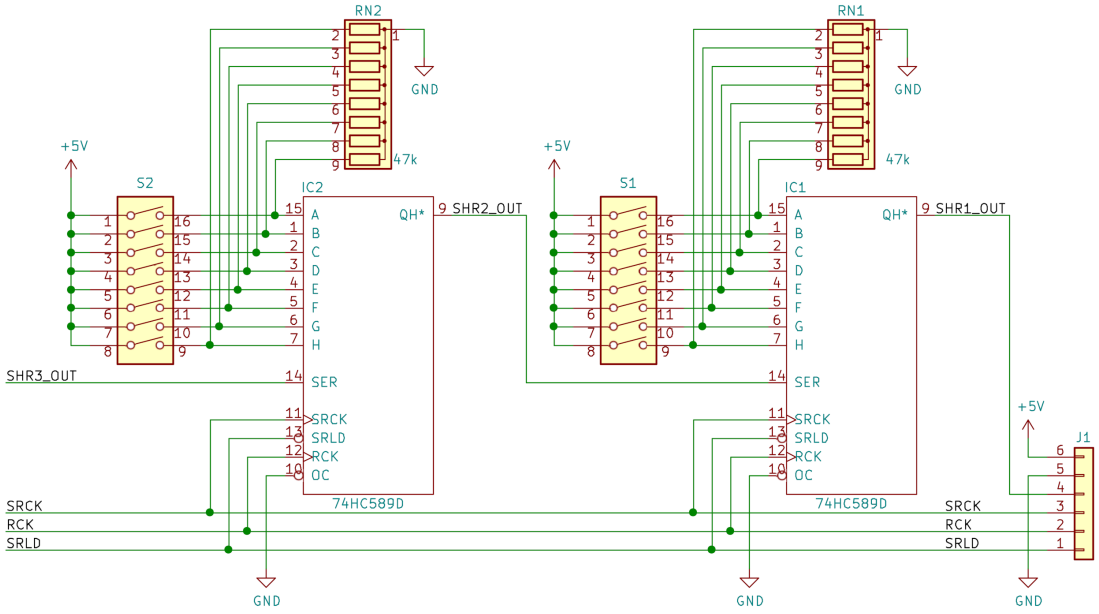
Konektory X5 a X6 nejsou zatím pro žádný účel využity a poskytují přes izolátory přímý přístup na vývody X0 až X17 vývojového kitu Trenz Electronic TE0720-03-61C33MA, na kterém je umístěn SoC Xilinx Zynq XC7Z020-1CLG484C, který je součástí emulační platformy. [33]

Konkrétně jsou tyto vývody přivedeny na CPLD firmy Lattice LCMX02-1200HC, které slouží jako systémový kontrolér pro daný vývojový kit. Úpravou firmwaru by bylo možné připojit všech 18 vývodů na některé z volných vývodů SoC, a to jak na straně ARM procesoru, tak na straně FPGA. U výchozího firmwaru jsou z těchto vývodů se systémem na čipu spojeny jen vývody X16 (nastaven jako vstup směrem k SoC vývodu A6, neboli MIO13) a X17 (nastaven jako výstup od SoC vývodu C5, neboli MIO12). Ve výchozím nastavení SoC se jedná o rozhraní UART2 ARM procesoru s maximální přenosovou rychlostí 1 Mbps. Vývody A6 a C5 by se daly překonfigurovat, aby vedly do FPGA části SoC, ale využití vestavěného UARTu, pomocí kterého lze přímo komunikovat s ARM jádrem je výhodné z hlediska neplýtvání FPGA zdrojů určených pro digitální část emulovaného obvodu a pro relativní jednoduchost vytvoření programu, který by mimo konfigurace dokázal provádět i monitorování systému. [33], [34], [35], [36]

Konektor X7 je využíván existující implementací konfiguračního modulu s fyzickými spínači pro každou jednu emulovanou pojistku. Principiální schéma je na obr. 2.1. Modul se skládá z řetězce 8-bitových posuvných registrů se sériovým i paralelním vstupem, kde na paralelním vstupu jsou spínače a na sériovém vstupu výstup předchozího registru. Na jednom modulu je těchto registrů 16, což odpovídá 128 konfiguračním bitům, a moduly lze řadit do řetězce pro teoreticky neomezený počet konfiguračních bitů.

Konektor má 1 vstup a 3 výstupy, vstup je pro výčet konfiguračních bitů, výstupy jsou pro řízení posuvných registrů v modulu. Konkrétně se jedná o hodinový signál pro bitový posuv SRCK, kterého náběžná hrana způsobuje posuv dat v řetězci registrů o jeden bit, hodinový signál pro navzorkování dat z paralelního vstupu RCK, kterého náběžná hrana vede k navzorkování konfigurace spínačů do vnitřního klop-

ného obvodu, a signál SRLD, který ve stavu logické nuly vede k nastavení hodnot posuvných registrů na hodnoty z vnitřního vzorkovacího klopného obvodu. [37]



Obr. 2.1: Zjednodušené schéma původního konfiguračního modulu

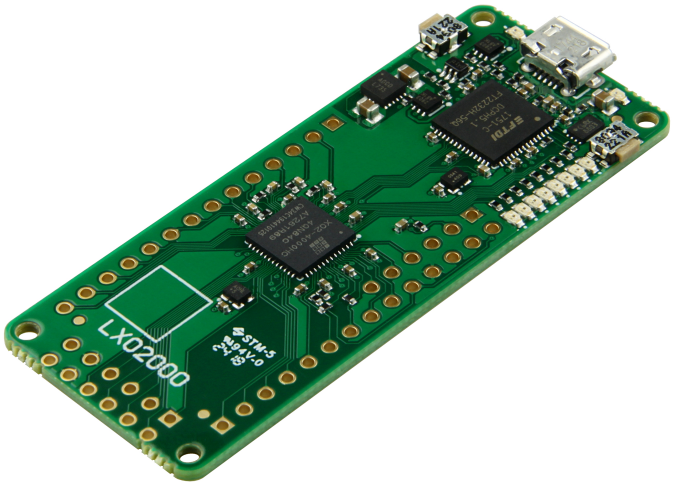
Po zvážení výhod a nevýhod jednotlivých možností byl pro bezdrátový konfigurační modul zvolen konektor X7 s rozhraním pro posuvné registry. Nakonec se ukázala být kompatibilita s předchozím konfiguračním modulem velice důležitým faktorem, který dovoluje využít modul bez jakýchkoliv úprav emulační platformy, což je výhodné z hlediska spolehlivosti i bezpečnosti, jelikož se jedná o citlivý obvod, který pracuje s vysokým napětím.

2.1 Volba součástek

Pro vytvoření rozhraní elektricky kompatibilního s rozhraním posuvných registrů dává smysl využít programovatelný logický obvod, zamezí se tím možným chybám vycházejícím ze sekvenčního charakteru mikrokontrolérového programu. Teoreticky by bylo možné využít diskrétní posuvný registr zapojen na výstupu mikrokontroléru, tam by ale také mohly nastat komplikace s tím, zda se registr po využití všech osmi bitů povede včas naplnit novými hodnotami, a tak byla zvolena alternativa místo mikrokontroléru využití programovatelného logického obvodu.

Jelikož se nejedná o složitý obvod, bylo původně uvažováno s jednoduchým CPLD obvodem firmy Lattice třídy ispMACH4000, u kterého je výhodou nenáročná montáž 44 nebo 48-vývodového TQFP pouzdra. Nakonec se ale ukázalo, že pro tyto obvody

firma Lattice už neposkytuje zdarma vývojové prostředí a cena pro roční licenci je $ 590,79 US. Jelikož autor práce nemá plány, které by cenu této licence dokázaly opodstatnit, bylo rozhodnuto, že bude využit novější čip, pro který je vývojové prostředí poskytnuto zdarma, konkrétně LCMXO2-4000HC třídy MachXO2. Čip byl zvolen kvůli tomu, že ho v Onsemi používají a mají ho k dispozici. Konkrétně jsou k dispozici desky Trenz Electronic TEL0001 (obr. 2.2), které obsahují daný obvod společně s napěťovým regulátorem, řadičem USB, generátorem hodinového signálu a podpůrnými obvody pro analogové vstupy. Tyto desky mají plošky, pomocí kterých se dají připájet na jinou DPS, a takovýto postup byl zvolen i u tohoto projektu, kde deska TEL0001 bude využita jako integrovaný modul na desce bezdrátového konfiguračního modulu. [38], [39], [40], [41], [42]



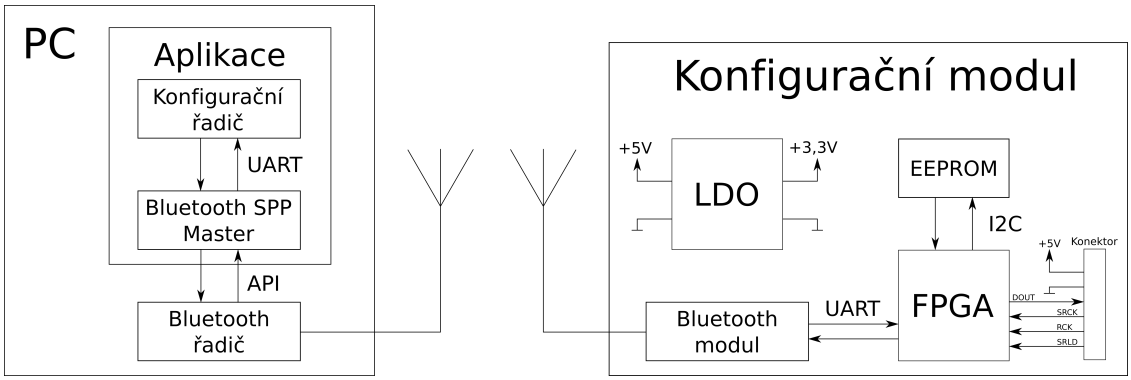
Obr. 2.2: Trenz Electronic TEL0001 (zdroj: [42])

Následně byla provedena volba bezdrátové komunikační technologie a odpovídajícího řadiče. Kvůli široké kompatibilitě s existujícími zařízeními byla zvolena technologie Bluetooth s profilem sériového portu a modul typu brána, který zprostředkovává UART sériovou komunikaci mezi nadřízeným Bluetooth zařízením a nějakým jiným obvodem na desce, v našem případě s obvodem typu FPGA. Výhodou tohoto přístupu je možnost výměny bezdrátové technologie v budoucnu. Jediné, co se bude muset změnit, je způsob zprostředkování UART komunikace, samotná data zasílaná po lince zůstanou stejná. Nevýhodou je o něco složitější hardware a u volby Bluetooth vyšší proudová spotřeba. Jelikož se jedná o vývojový prostředek a je pro něj k dispozici dostatečně silný zdroj napájení, byla širší kompatibilita vyhodnocena jako důležitější vlastnost vůči proudové spotřebě. Technologie WiFi, která má také vynikající kompatibilitu, byla vyloučena kvůli poměrné složitosti zapojení zařízení do WLAN sítě v porovnání s jednoduchou PAN sítí. [6]

Konkrétně byl pro tento účel zvolen modul Lantronix xPico 250. Hlavní motivací pro volbu právě tohoto modulu vůči ESP32 jsou vnitřní směrnice a požadavky na bezpečnost v Onsemi, které nedovolují provozovat ve firemní síti čínská zařízení, dobré zkušenosti konzultanta práce s tímto konkrétním modulem a dostupnost modulu.

Aby si obvod dokázal zachovat konfigurační stav i po odpojení napájení, je zapotřebí využít paměťový prvek. Kvůli nízké ceně, jednoduché montáži a nízkého počtu vodičů pro komunikaci s FPGA byla zvolena paměť EEPROM s rozhraním I^2C, a to konkrétně AT24C08D v 8-vývodovém pouzdru SOIC, z kterých pro komunikaci s FPGA slouží dva. [43]

Modul xPico a EEPROM čip si vyžadují napájecí napětí 3,3 V místo 5 V, které je dostupné na konektoru. Modul TEL0001 obsahuje spínaný regulátor poskytující napětí 3,3 V s maximálním proudovým výstupem 1 A, který využívá pro napájení vlastních obvodů a poskytuje ho i možným periferiím, tak by se dal využít pro napájení xPico modulu i EEPROM. Dokumentace xPico klade ale důraz na požadavek nízkého šumu napájecího napětí, z čehož plyne volba diskrétního lineárního regulátoru. Maximální proudový odběr xPico modulu je kolem 600 mA a u AT24C08D paměti je to maximálně 1 mA, což vedlo k volbě regulátoru MC33269 firmy Onsemi s výstupním proudem 800 mA. [29], [42], [43], [44] Blokové schéma výsledného uspořádání je znázorněno na obr. 2.3. Schéma zapojení je v elektronické příloze.



Obr. 2.3: Blokové schéma bezdrátového konfiguračního modulu

2.2 Firmware, software a komunikační protokol

U FPGA firmwaru se jedná o náhradu řetězce posuvných registrů. V původní realizaci konfiguračního modulu vzorkují tyto registry hodnoty fyzických spínačů, v nové realizaci si přejeme poskytnout konfigurační data bezdrátově pomocí Bluetooth modulu a mít je k dispozici i po odpojení napájecího napětí.

Pro tento účel máme k dispozici UART rozhraní s hardwarovou podporou řízení toku poskytující přímé datové spojení s konfiguračním programem a EEPROM čip pro úschovu dat při vypnutém napájení, jak je znázorněno na obr. 2.3. Hardwarové řízení toku je výhodné z hlediska zabránění ztráty dat, pokud by je FPGA nebo bezdrátový modul nestíhaly zpracovat, a dovoluje zjednodušit komunikační protokol, který by jinak musel zahrnovat opatření, aby nedošlo k přetečení zásobníků nebo jiných mechanizmů pro ukládání a zpracování dat.

Emulační platforma přistupuje ke konfiguračnímu modulu jako k podřízenému zařízení. Nástupní hranou RCK signálu dává příkaz k navzorkování konfigurace ze spínačů, stavem logické 0 signálu SRLD dává pokyn posuvným registrům aplikovat navzorkované hodnoty a nástupní hranou SRCK dává příkaz k vysunutí jednoho bitu z řetězce registrů.

Toto rozhraní by se na FPGA dalo implementovat pomocí jednoho posuvného registru, který by byl plněn hodnotami z paměti typu RAM, např. block RAM. Rozhraní by bylo inicializováno stavem logické 0 signálu SRLD, do posuvného registru o předem stanovené šířky (např. 8 bitů) by byly načteny hodnoty odpovídající konfiguračním spínačům umístěným k posuvnému registru s výstupem zapojeným na konektor u původního modulu a do adresovacího registru by byla uložena adresa následujícího konfiguračního slova. Při vysunutí celého slova z registru by bylo načteno slovo v paměti na adrese udané adresovým registrem, a registr by byl inkrementován.

Hodnoty konfiguračních slov v block RAM by po přivedení napájecího napětí byly vyčteny z EEPROM paměti. Aktualizaci hodnot v block RAM by bylo možné pomocí UART rozhraní, nad kterým by byl implementován stavový automat zpracovávající textovou komunikaci mezi FPGA a konfiguračním softwarem na počítači typu PC. Tento stavový automat by byl schopen zapisovat, a případně i vyčíst data z block RAM (pro účely zjištění aktuální konfigurace). Je vhodné také provádět kontrolu zapsaných dat, čehož lze dosáhnout vyhodnocováním kontrolního součtu a následným porovnáním s kontrolním součtem zaslaným konfiguračním softwarem. V případě nalezení chyby by bylo možné požádat o zopakování zápisového příkazu, v případě úspěšného ověření by mohla být zapsaná data označena pro zápis do EEPROM, aby byly dostupné i po odpojení napájecího napětí.

V praxi často využívaným formátem pro zápis dat do paměti přes sériovou linku je formát Intel HEX. Data ve formátu se skládají ze záznamů obsahující informaci o druhu záznamu, o počtu přenášených bajtů dat (s maximem 255 pro jeden záznam), o počáteční adrese, na kterou mají být data uložena, samotná data v hexadecimálním zápisu, a také kontrolní součet dat v hexadecimálním zápisu. Kontrolní součet je vypočten jako dvojkový doplněk součtu všech bajtů záznamu. [45]

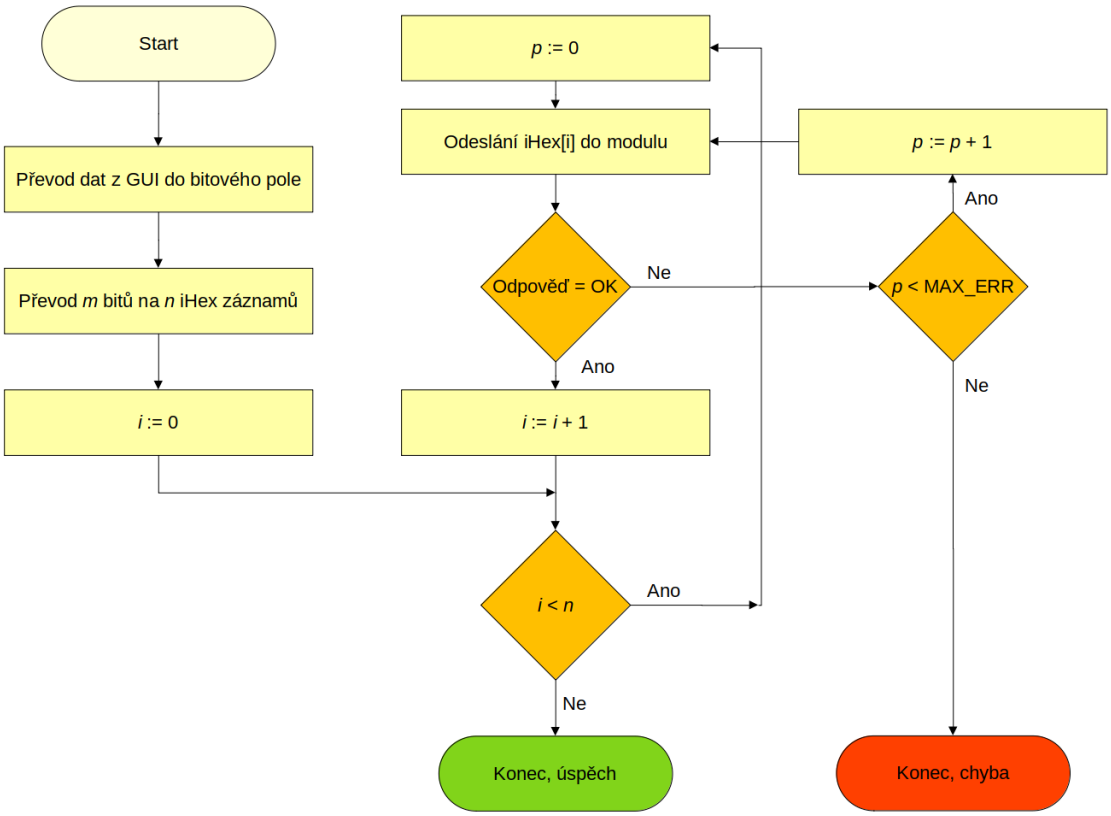
Komunikační protokol modulu by mohl být založen na formátu Intel HEX, záznamy tohoto formátu by se daly využít jako příkazy pro zápis dat a pro mož-

nost průběžné kontroly úspěšnosti by firmware pro každý záznam mohl generovat odpověď. Tímto způsobem by byla zachována kompatibilita pro různé obecné programovací aplikace, které formát Intel HEX využívají, a odpovědi na záznamy by konkrétní konfigurační aplikaci modulu dovolily v případě neúspěšného zápisu zkusit zápis zopakovat. Pokusů o zápis tím samým příkazem by mělo být omezené množství, aby se zabránilo zacyklení programu v případě, že by se jednalo o chybu, která by nebyla způsobena chybným přenosem, ale chybou v softwaru nebo ve firmwaru, tento postup je znázorněn na obr. 2.4.

Pro identifikaci modulu by měl konfigurační protokol obsahovat i identifikační příkaz, na který by modul odpověděl svým názvem nebo identifikačním kódem a kódem revize firmwaru. Tato funkcionalita zajistí, že si konfigurační aplikace dokáže ověřit, zda se připojila na kompatibilní modul, a ověřit si na základě identifikačních informací technické parametry modulu a podporované komunikační příkazy.

Pro výčet dat z modulu by mohl posloužit příkaz obsahující počáteční adresu výčtu a počet vyčtených bajtů. Jako odpověď by mohl modul generovat záznamy typu Intel HEX, stejný formát jako pro zápis. Hodnoty kontrolních součtů záznamů dovolí konfigurační aplikaci ověřit si správnost přenosu a v případě odhalení chyby může aplikace požádat o opětovný výčet části dat, která byla poskytnuta vadným záznamem.

Jelikož by data byla uložena v block RAM, která je přímo využitá rozhraním s posuvným registrem, byly by změny konfigurace dostupné emulační platformě ihned. Stačilo by přivést na signál SRLD logickou 0, čímž by došlo k inicializaci rozhraní a následně data vysunout hodinovým signálem SRCK. Jelikož toto rozhraní nemá žádný způsob signalizace čekání, zatímco na UARTu je dostupné hardwarové řízení toku, mělo by rozhraní s posuvným registrem mít vyšší prioritu pro přístup k block RAM než příkazový interpret UARTu a synchronizační obvod pro EEPROM, případně by mohlo být následující konfigurační slovo načteno dříve a dočasně uloženo do pomocného registru, ze kterého by po vysunutí celého aktuálního konfiguračního slova bylo přesunuto do posuvného registru.



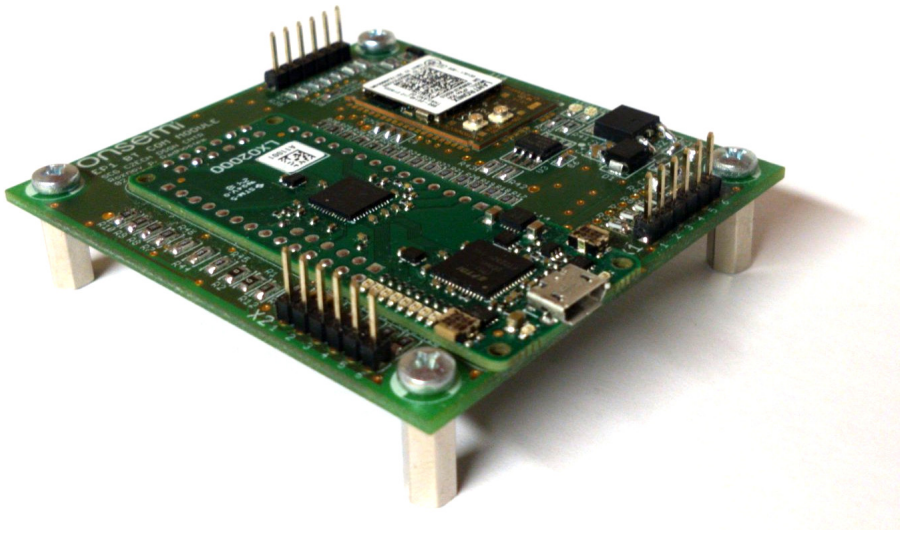
Obr. 2.4: Vývojový diagram popisující zápis konfiguračních dat ze strany softwaru

3 Realizace konfiguračního modulu

V této kapitole jsou popsány jednotlivé kroky k vytvoření konfiguračního modulu.

3.1 Návrh a výroba DPS

U DPS modulu se v podstatě jedná jen o propojení FPGA kitu Trenz Electronic TEL0001 a bezdrátové komunikační brány Lantronix xPico 250. Deska také obsahuje EEPROM čip připojen k FPGA kitu pro uchování konfigurace po dobu vypnutí modulu a napěťový regulátor pro převod napájecího napětí 5,0 V na 3,3 V pro napájení bezdrátové brány. Návrh DPS byl proveden v prostředí Autodesk EAGLE 9.6.2. Schéma zapojení a motiv DPS jsou v elektronické příloze. Foto výsledného modulu je na obr. 3.1.



Obr. 3.1: Foto navrženého a vyrobeného modulu

Na modulu jsou umístěny tři konektory: X2 a X3 jsou pro zpřístupnění ladícího rozhraní JTAG FPGA kitu, resp. bezdrátové brány, a X1 slouží k připojení k emulační platformě. Přes konektor X1 je na desku přivedeno i napájení. Pro ochranu desky před přepólovaným napájecím napětím, a také pro ochranu před naindukovanými rušivými napěťovými špičkami, byla na desku umístěna ochranná dioda typu SMBJ5.0A a nadproudová PTC pojistka 1206L150TH s jmenovitým proudem 1,5 A, a vypínacím proudem 3,0 A.

Ke všem analogovým a digitálním vstupům a výstupům modulů byly umístěny ochranné sériové rezistory, které slouží k potlačení vlivu naindukovaných rušivých napěťových špiček (jelikož se jedná o výrobek určený do rušivého prostředí) a také

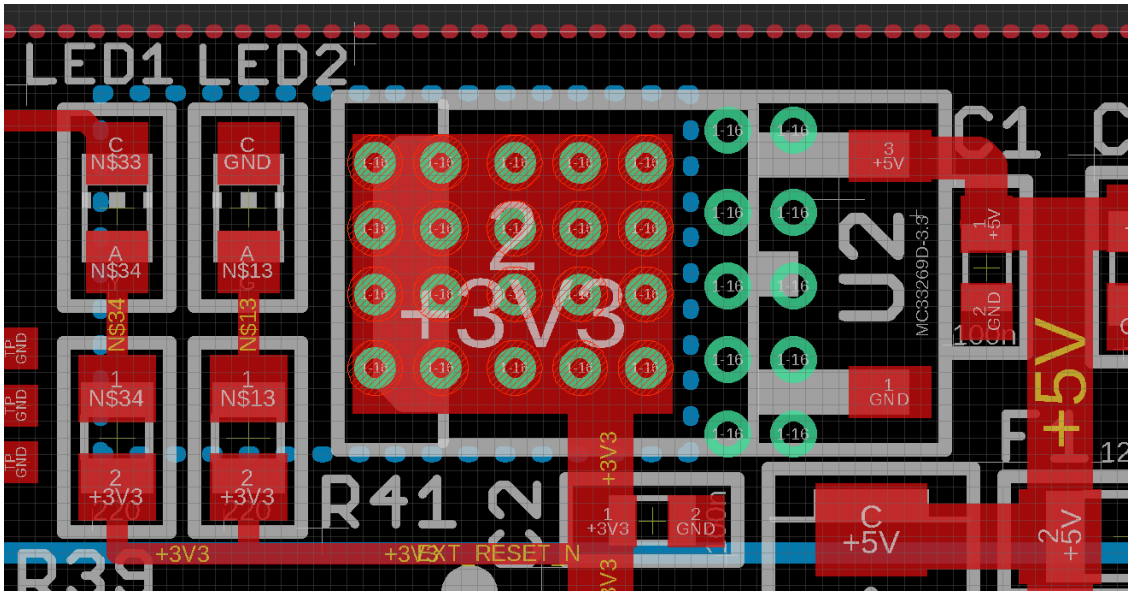
pro účely ladění: poskytují možnost jednoduchého odpojení jednotlivých vývodů. Slouží také jako nadproudová ochrana v případě zkratu a jako testovací plošky k ověření dobrého elektrického spojení pájených spojů a vyloučení zkratů.

Testovací plošky byly také realizovány kolem celé bezdrátové brány xPico 250, jelikož má tato brána vývody jen ze spodní strany, a bylo by jinak obtížné ověřit kvalitu pájených spojů.

Deska byla realizována v dvouvrstvém provedení, s třídou přesnosti 6 (minimální šířka vodiče 0,15 mm, minimální mezera 0,15 mm, minimální velikost otvoru 0,3mm). Samotný návrh počítal s určitou rezervou pro zajištění vyšší spolehlivosti, a po téměř celou dobu návrhu byla využita minimální šířka vodiče a mezery 0,3 mm. Minimální šířka mezery byla snížena až v posledním stádiu návrhu, u optimalizace, při zvětšování napájecích vodičů a rozměrů prokovů.

U návrhu desky byla také využita rozlitá měď a pro zajištění kvalitního signálového propojení všech částí rozlité mědi byl na desku na návrh konzultanta přidán hojný počet prokovů. Prokovy byly také využity jako držáky hrotových sond osciloskopu.

Rozlitá měď byla také využita pro realizaci chladiče pro napěťový regulátor. Jelikož nebylo na vrchní straně desky dostatek místa pro chladící plochu, byla chladící plocha realizována na spodní straně desky a spojena s vrchní stranou hojným počtem prokovů zajišťujících kvalitní tepelný odvod z regulátoru (obr. 3.2).

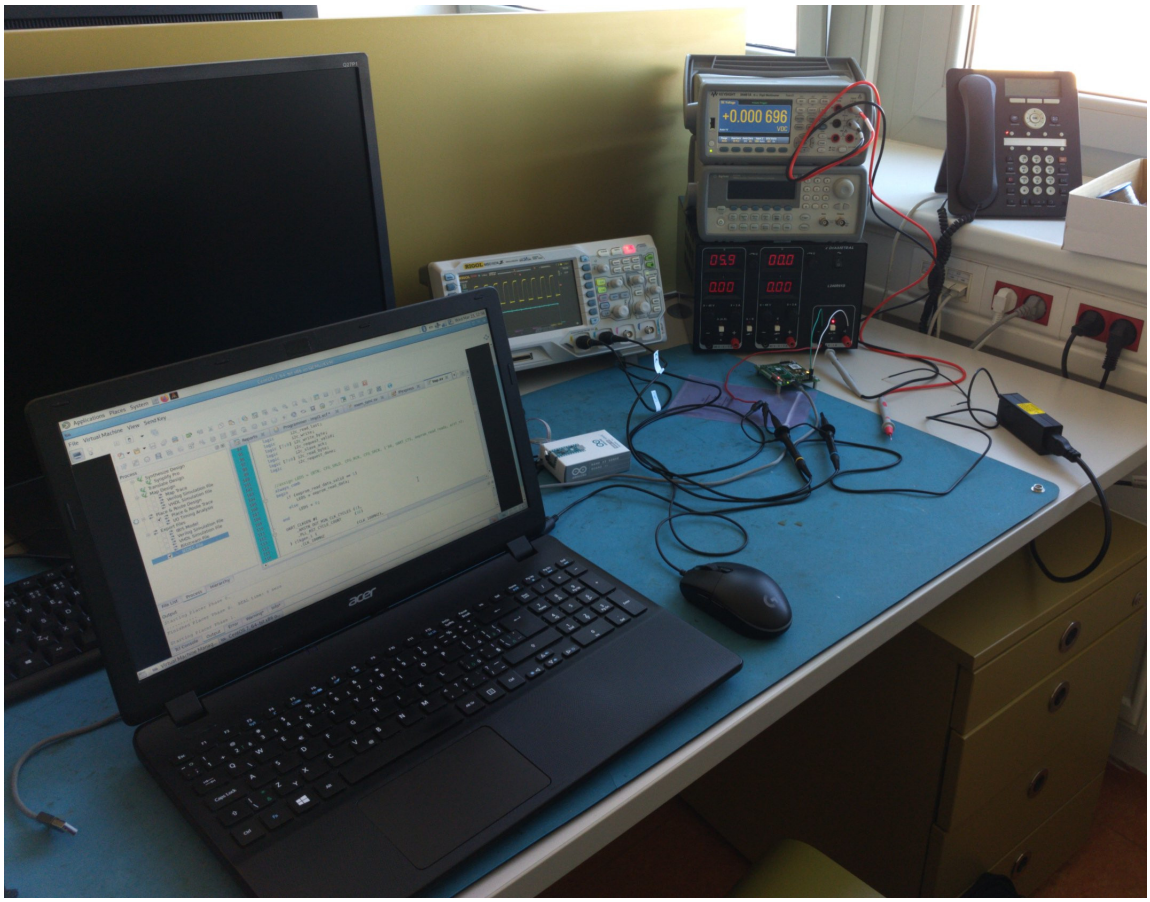


Obr. 3.2: Napěťový regulátor 3,3 V s chladící ploškou na spodní straně desky

Samotná výroba DPS byla provedena firmou Gatema PCB a.s. formou pool service na substrátu typu FR4 (Tg135) 1,5 mm s tloušťkou mědi 18 µm a povrchovou

úpravou imersním zlatem. Osazení a pájení bylo provedeno v design centru Onsemi v Rožnově pod Radhoštěm metodou pájení přetavením, s nanesením pájecí pasty přes šablonu. Šablona byla potřebná pro kvalitní zapájení bezdrátové brány, jejíž vývody jsou malých rozměrů (čtvercové se stranou 0,80 mm) a ze spodní strany součástky.

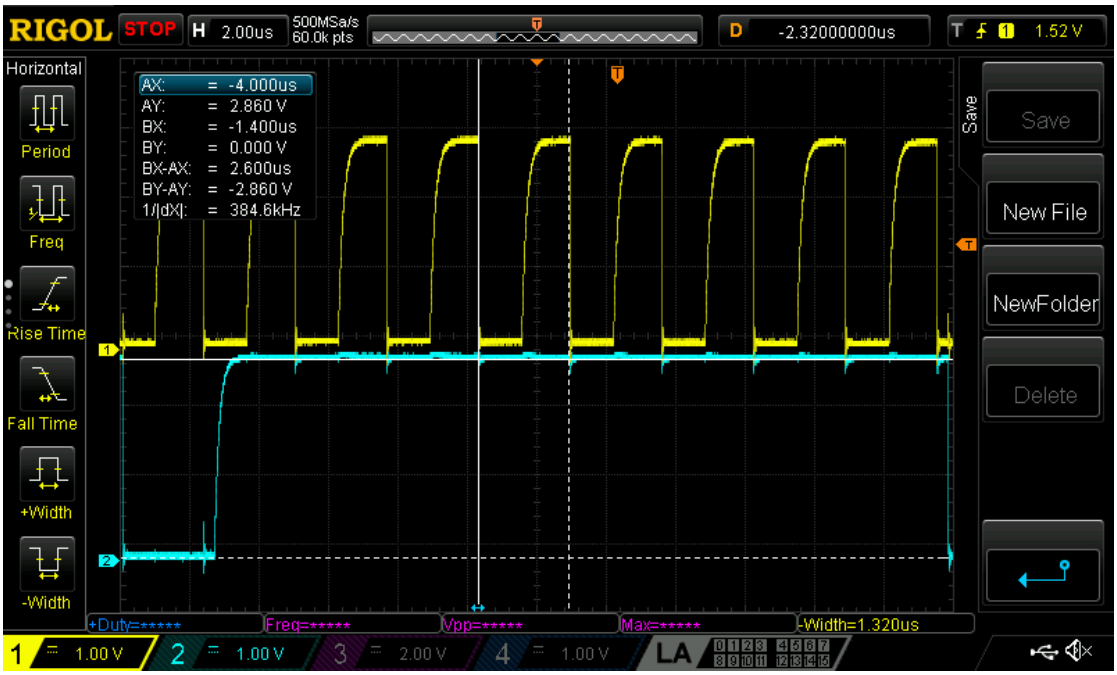
Oživování desky bylo provedeno z části v design centru Onsemi a z části v mikrobastnírně na pátém podlaží budovy T10 FEKTu VUT v Brně (obr. 3.3 a 3.4). Zatímco v Onsemi bylo testováno hlavně spojení s emulační platformou, v mikrobastlírně byla zprovozněna Bluetooth SPP a I^2C EEPROM komunikace.



Obr. 3.3: Oživování I^2C EEPROM komunikace v mikrobastlírně

U oživování desek se vyskytl problém s firmwarem bezdrátové brány, nebylo možné nastavit mapování Bluetooth připojení typu SPP (Serial Port Profile) k vnitřnímu komunikačnímu kanálu. Bylo zapotřebí provést aktualizaci firmwaru na verzi 5.1.0.0R4 vydanou 26. 1. 2022, poté se možnost namapovat SPP připojení na kanál v konfiguračním rozhraní zpřístupnila.

Na propojení UART rozhraní připojeného k FPGA a Bluetooth SPP rozhraní určeného pro konfigurační aplikaci na PC bylo zapotřebí nastavit u obou rozhraní



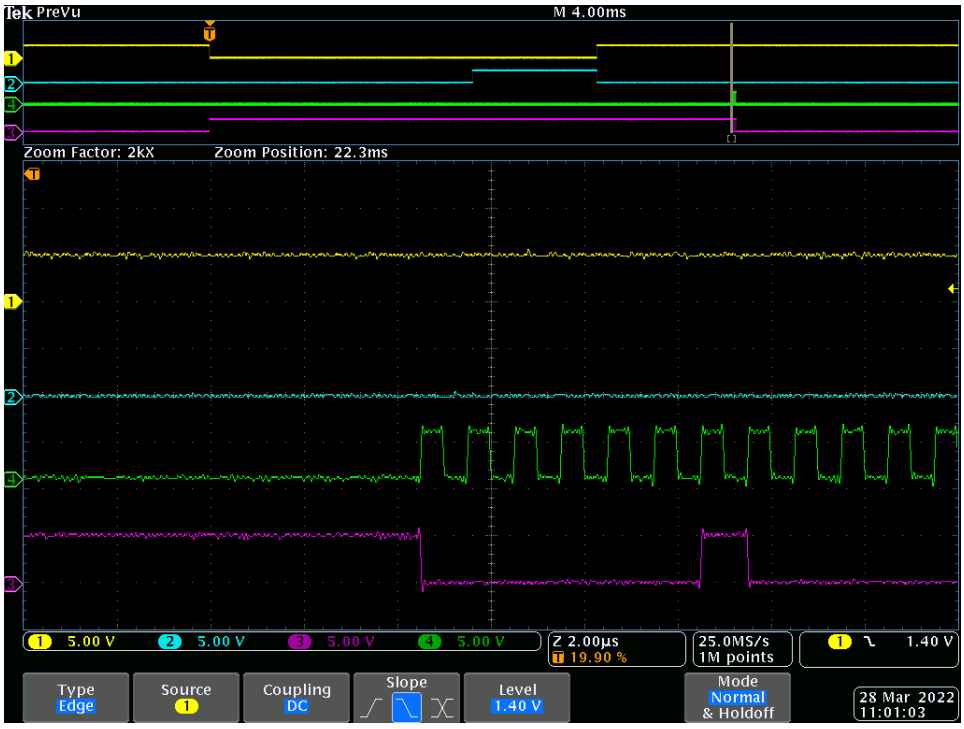
Obr. 3.4: Zobrazení I^2C komunikace na osciloskopu (čtení 0xFF)

mapování typu tunel, a pak v záložce nastavení tunelů zvolit u jednoho rozhraní režim TCP serveru a u druhého režim TCP klienta, který se připájí na server na IP adrese 127.0.0.1 (místní stroj) se stejným číslem portu.

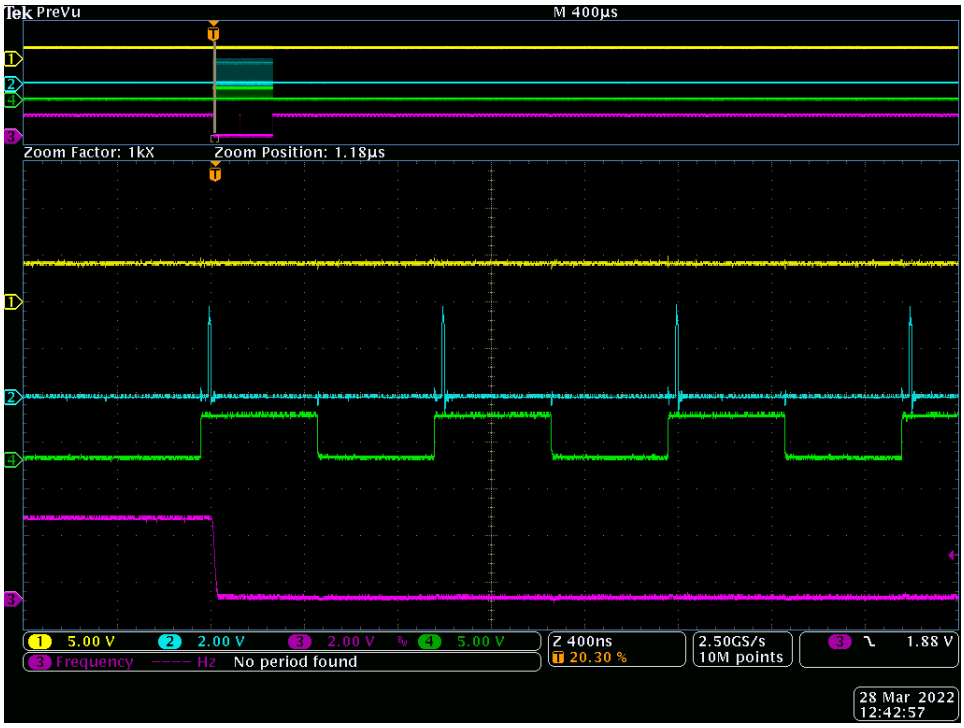
Na obrázcích 3.5, 3.6, 3.7 a 3.8 jsou zobrazeny časové průběhy signálu konfiguračního rozhraní mezi emulační platformou a navrženým konfiguračním modulem. Žlutý signál je SRLD, zelený je SRCK, a fialový je datový výstup DOUT. Na obrázcích 3.5 a 3.6 je tyrkysový signál RCK, a na 3.7 a 3.8 je to vnitřní signál *shift_condition* FPGA modulu CFG_MEM udávající detekci podmínky vysunutí bitu z posuvného registru.

Z těchto snímků je vidět, že konfigurační modul s emulační platformou opravdu dokáže komunikovat a přenášet do něj konfigurační bity odpovídající emulovaným konfiguračním tavným pojistkám.

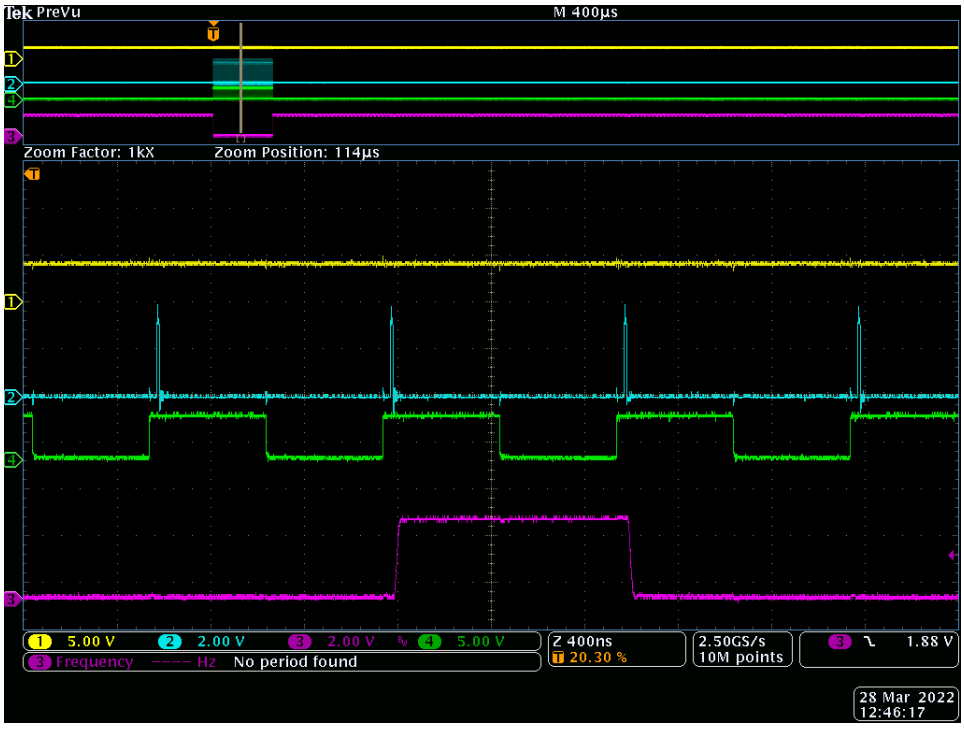
Obr. 3.5: Zobrazení signálů konfiguračního rozhraní na osciloskopu



Obr. 3.6: Detail signálů konfiguračního rozhraní na začátku přenosu bitů



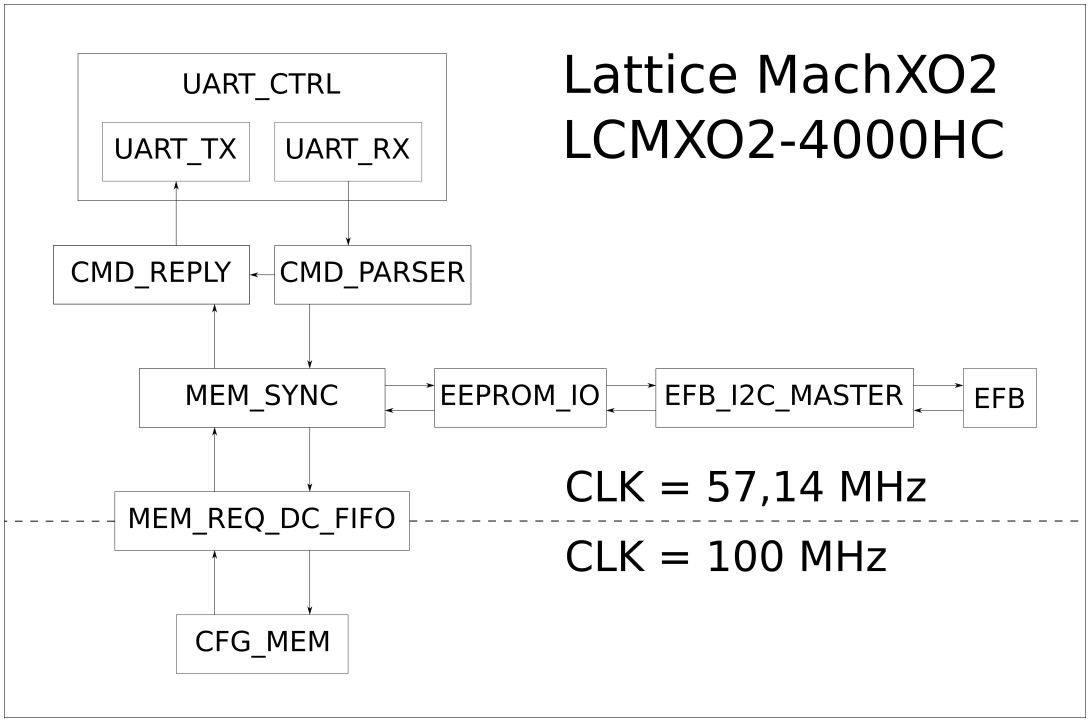
Obr. 3.7: Detail konfiguračních signálů na začátku přenosu bitů se signálem vysunutí



Obr. 3.8: Detail konfiguračních signálů uprostřed přenosu bitů se signálem vysunutí

3.2 Návrh a implementace FPGA firmwaru

Vývoj FPGA firmwaru byl proveden v prostředí Lattice Diamod firmy Lattice Semiconductor Corporation. Pro vypracování projektu byl zvolen HDL jazyk SystemVerilog, jelikož se jedná o jazyk, který pro návrh digitálních obvodů využívají v design centru Onsemi v Rožnově. Při vypracování byl kladen důraz na modularitu kódu, využitelnost komponent i v jiných projektech, jasně definované rozhraní komponent a možnost změn a úprav pro jednoduché přenesení designu na jinou FPGA platformu. Obrázek 3.9 představuje blokové schéma firmwaru. Zdrojový kód, společně s bit streamem pro FPGA na modulu, se nachází v elektronické příloze.



Obr. 3.9: Blokové schéma navrženého FPGA firmwaru

Jádrem firmwaru je paměť typu block RAM, která obsahuje data odpovídající konfiguračním pojistkám emulovaného čipu. Na tuto paměť je napojen posuvný registr, jehož rozhraní je vyvedeno na konektor X1 modulu a také rozhraní vnitřní pro výčet a zápis konfiguračních dat.

K tomuto rozhraní je napojena synchronizační jednotka, která slouží pro inicializaci dat v konfigurační block RAM po přivedení napájecího napětí výčtem z EEPROM a také pro zápis do EEPROM, když probíhá zápis do konfigurační block RAM. Cílem této komponenty je udržení konfigurační block RAM a paměti EEPROM v identickém stavu, aby bylo možné konfigurační stav zachovat po odpojení a následném připojení napájecího napětí.

Na tuto synchronizační jednotku jsou kromě EEPROM a konfigurační block RAM napojeny příkazový parsér a jednotka generace odpovědí na příkazy. Parsér slouží na zpracování textových příkazů z ovládacího softwaru na PC a jednotka generace odpovědí na příkazy vhodně odpovídá. Obě jednotky pracují s textovým formátem Intel HEX, který slouží jako rámcový formát komunikace, a byl rozšířen o příkazy identity a čtení a také o odpovědi na dané příkazy a signalizaci poruch a úspěchu komunikace.

Pro komunikaci s vnějším světem slouží jednotka UART (Universal Asynchronous Receiver/Transmitter), tato jednotka převádí textové znaky na elektrické impulzy s definovaným časováním a zpět. Tato jednotka je na modulu spojena s bezdrátovou bránou xPico 250, která poskytuje připojení k tomuto rozhraní pomocí Bluetooth SPP (Serial Port Profile).

Pro komunikaci s EEPROM čipem slouží dvě komponenty: Komponenta „EEPROM I/O" pracující na vyšší úrovni, která zohledňuje specifické vlastnosti zvoleného EEPROM čipu, jako jsou způsob adresování, čtecí a zápisové protokoly a stránkový zápis, a komponenta nižší úrovně „EFB I2C Master" která ovládá vestavěnou I^2C jednotku na FPGA čipu a poskytuje jednoduché rozhraní pro generaci I^2C signálů a čtení/zápis dat. Rozdělení řadiče na dvě komponenty zjednodušuje kód a dovoluje nahradit komponentu nižší úrovně jiným řadičem I^2C v případě přesunu projektu na jinou FPGA platformu.

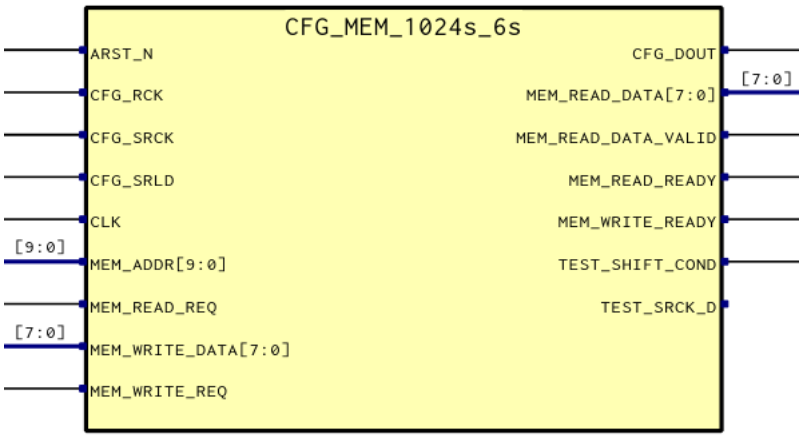
Následuje podrobný popis jednotlivých části firmwaru.

3.2.1 Paměť konfigurace

Komponenta CFG_MEM, obsažena v souboru *cfg_mem.sv*, slouží pro uchování dat odpovídajících konfiguračním pojistkám a zpřístupnění těchto dat emulační platformě a konfiguračnímu programu na PC. Schematická značka komponenty je na obr. 3.10.

Obě rozhraní komponenty jsou slave rozhraní a jelikož mechanizmus pro výčet dat do emulační platformy neobsahuje podporu čekání, musí mít toto rozhraní vyšší prioritu k přístupu ke konfiguračním datům než rozhraní pro PC. Mechanizmem priority je multiplexor, který na základě signálu *ext_io_allow* volí, zda je na konfigurační block RAM napojeno vnější rozhraní, nebo vnitřní signály pro naplnění posuvného registru.

Posuvný registr se skládá ze tří paměťových prvků: z prvního konfiguračního bajtu, z aktuálního konfiguračního bajtu a následujícího konfiguračního bajtu. První bajt musí být k dispozici vždy, jelikož emulační platforma může kdykoliv nařídit „načtení dat z paralelního vstupu do posuvného registru", což pro modul znamená přivedení nejvýznamnějšího bitu prvního bajtu na vývod DOUT.



Obr. 3.10: Schematická značka FPGA komponenty konfigurační paměti

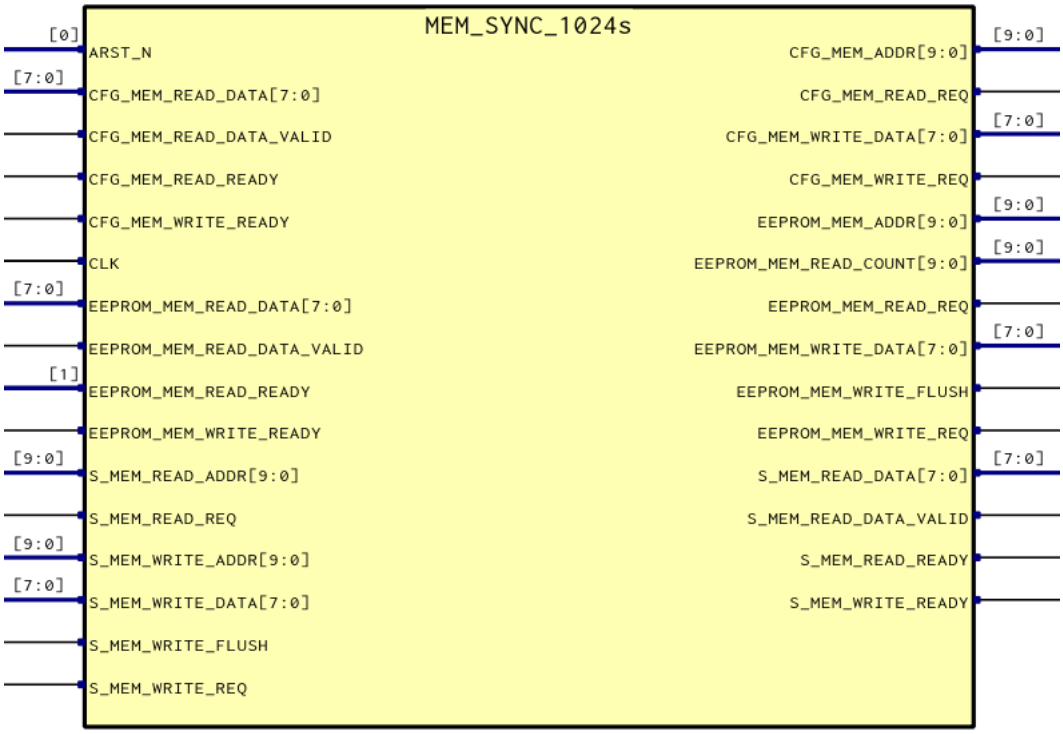
Paměť pro následující bajt je vždy načtena s určitým předstihem, aby byl bajt také k dispozici, když si ho emulační platforma vyžádá. Aktuálně je pomocí konstanty NEXT_BYTE_LOAD_BIT nastavena tato podmínka na bit 6 aktuálního bajtu, což je předposlední bit.

Vstupy z emulační platformy jsou před využitím převedeny do hodinové domény konfigurační paměti, a to pomocí sériového řazení tří D klopných obvodů. Paměť aktuálně funguje s hodinovým taktem 100 MHz, což je rychlejší než zbytek systému, který funguje na 57,14 MHz, a to hlavně pro dosažení podobných časových charakteristik, jaké dosahují původní posuvné registry 74HC589D, konkrétně možnost zpracování signálu řídících hodin 20 MHz s propagačním zpožděním v desítkách nanosekund. Nakonec se ale ukázalo, že pro konkrétní emulační platformu nemusí tyto parametry splňovat dané meze, jelikož je rychlost přístupu jen 1 MHz, a výčet hodnoty je prováděn v polovině periody, takže by postačilo i propagační zpoždění do 500 ns.

3.2.2 Synchronizační jednotka

Komponenta MEM_SYNC, obsažena v souboru *mem_sync.sv*, má za úkol zajistit, že data v konfigurační paměti a v EEPROM jsou identická. Řeší to takovým způsobem, že po resetu načte data z EEPROM do konfigurační paměti a u požadavků na zápis do konfigurační paměti provádí identický zápis do EEPROM. Schematická značka komponenty je na obr. 3.11.

Logika této komponenty je docela jednoduchá. Skládá se ze stavového automatu, který provádí inicializaci konfigurační paměti po resetu, a dále pak už jen kontroluje čtecí a zápisové požadavky. Zápisové požadavky si komponenta ukládá do vnitřního registru, jelikož je musí umět zreplikovat pro dvě rozhraní.



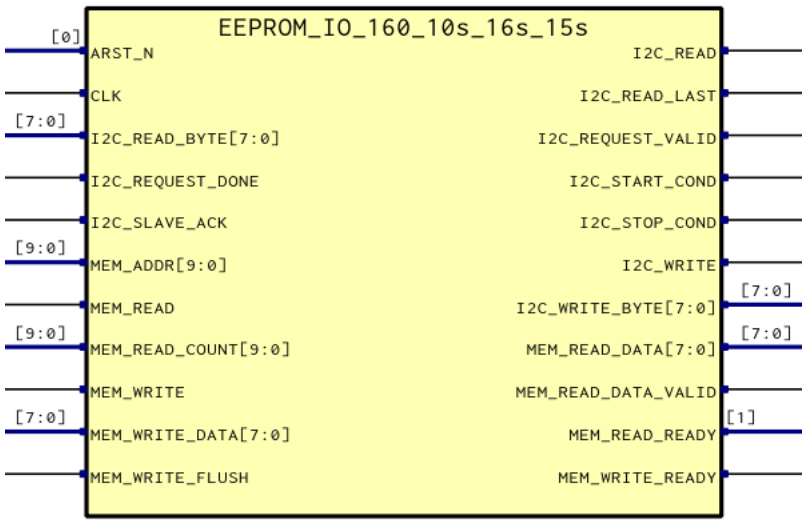
Obr. 3.11: Schematická značka FPGA komponenty synchronizační jednotky

Komponenta také zpracovává a propaguje signál pro zápis stránky dat do EEPROM, který je generován v případě posledního bajtu zápisové transakce z parsru. Komponenta zajišťuje, že je tento signál přiveden na kontrolér EEPROM paměti ve vhodnou chvíli.

3.2.3 Řídící jednotka EEPROM

Řídící jednotka EEPROM čipu je rozdělena do dvou částí, do komponenty EEPROM_IO v souboru *eeprom_io.sv*. která je speciálně určena pro řízení čipu AT24C08D s rozhraním pro zápis a čtení bajtu dat, a do komponenty EFB_I2C_MASTER v souboru *efb_i2c_master.sv*, která pomocí EFB bloku pro I^2C generuje a sleduje signály na I^2C sběrnici. Schématické značky komponent jsou na obr. 3.12 a obr. 3.13

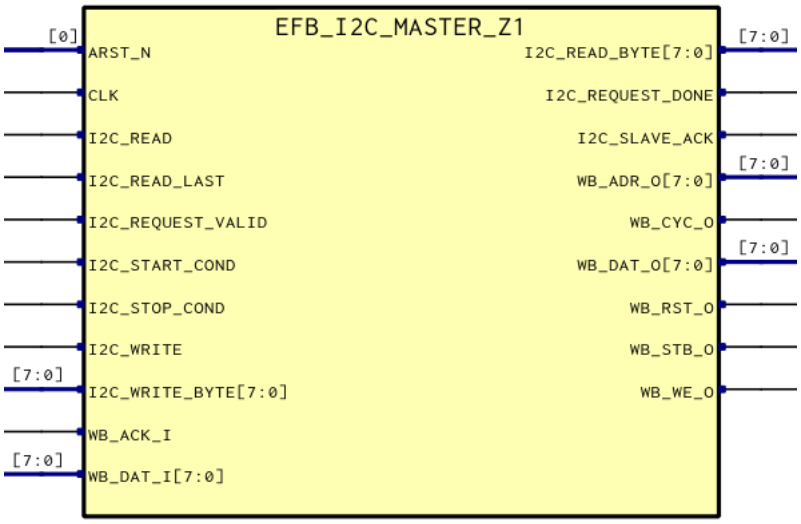
Čip AT24C08D se z hlediska adresování chová na I^2C sběrnici jako 4 nezávislé 256-bajtové paměti, každá s vlastní hardwarovou adresou. Z hlediska čtení se ale chová jako jednotný čip a čtecí přístup začínající v první 256-bajtové části dokáže vyčíst data až do posledního bajtu. Výčet dat funguje nastavením vnitřního adresového registru pomocí zápisového požadavku bez dat a následného čtecího požadavku. Číst lze řetězec bajtů libovolné délky, čtení je ukončeno zasláním NACK



Obr. 3.12: Schematická značka FPGA komponenty EEPROM_IO

stavu a následně STOP stavu.

U zápisu do AT24C08D lze zapisovat řetězec maximálně 16 bajtů, jedná se o tzv. stránku paměti a stránkový zápis musí být zarovnán na adresu dělitelnou 16. Po ukončení zápisu STOP stavem je zapotřebí před dalším požadavkem vyčkat až se zápis provede, buď vyčkáním doby 5 ms, nebo pomocí techniky ACK polling.



Obr. 3.13: Schematická značka FPGA komponenty EFB_I2C_MASTER

Řídící jednotka EEPROM_IO zohledňuje tyto vlastnosti čipu a poskytuje jednoduché paměťové rozhraní, pomocí kterého lze zapisovat a číst libovolně dlouhé řetězce bajtů. Čtení je jednoduché a u zápisu komponenta automaticky skládá data

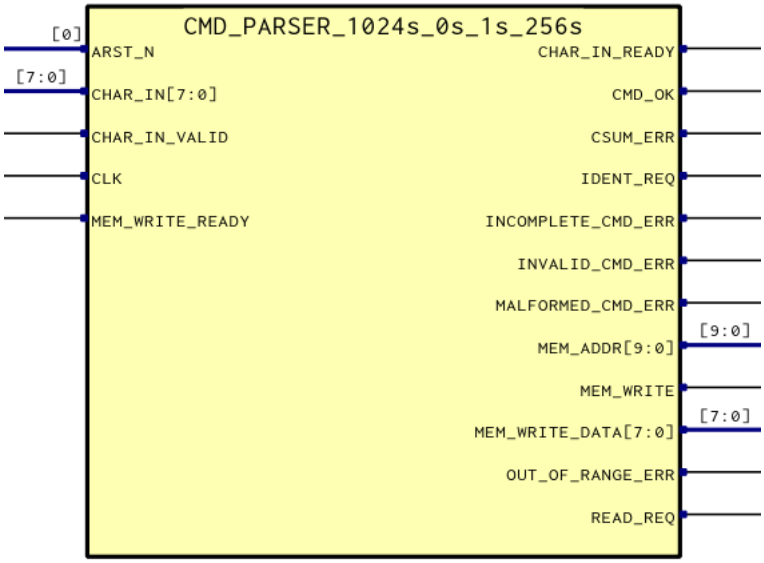
pro zápis do stránek a využívá ACK polling (sledování odpovědi čipu na pokus vykonat další požadavek) pro zjištění, zda je zápisový požadavek dokončen.

Z toho plyne, že když uživatel zapisuje přes rozhraní komponenty data a požaduje, aby byly v EEPROM co nejdříve, musí využít signál FLUSH, pomocí kterého přinutí komponentu zapsat aktuální stránku paměti do EEPROM, i když není plná. Komponenta automaticky provádí zápis aktuální stránky, pokud se naplní, nebo pokud adresa bajtu zápisového požadavku nenavazuje na adresu předchozího zápisového požadavku.

Komponenta EFB_I2C_MASTER slouží jako rozhraní mezi komponentou EEPROM_IO a vestavěným řadičem I^2C. S vestavěným řadičem komunikuje pomocí rozhraní Wishbone a přistupuje k registrům takovým způsobem, aby zrealizovala požadavky přivedené na své vstupy. Dovoluje generovat stavy START/STOP, provádět bajtový zápis/čtení a sledovat a generovat stavy ACK/NACK. Jedná se o rozhraní typu Master a nepodporuje funkci v režimu Slave.

3.2.4 Parsér příkazů

Komponenta CMD_PARSER, obsažena v souboru *cmd_parser.sv*, hraje klíčovou roli v zprostředkování komunikace mezi konfiguračním PC softwarem a pamětí konfigurace. Tato komponenta představuje stavový automat, který akceptuje textové znaky a vyhodnocuje jejich význam. Schematická značka komponenty je na obr. 3.14.



Obr. 3.14: Schematická značka FPGA komponenty příkazového parsru

Formát příkazů odpovídá formátu Intel HEX, skládající se ze znaku záznamu (dvojtečka), dvou hexadecimálních čísel udávajících počet datových bajtů v transakci, čtyř hexadecimálních číslic udávajících adresu, dvou hexadecimálních číslic udávajících typ záznamu, pak číslic odpovídajících datům záznamu a poslední dvě číslice jsou kontrolní součet.

Ze standardních příkazů podporuje modul jen příkaz datového záznamu a označení konce souboru (které je ignorováno). Podpora rozšířeného adresování je v parsru naimplementována, ale je vypnuta, jelikož konfigurační paměť, stejně jako paměť EEPROM, má kapacitu 1 KB a standardní Intel HEX zápisy bez rozšířeného adresování dokážou zapisovat až do 64 KB, což pro účely modulu bohatě postačí.

Parsér také podporuje upravené příkazy, které se od standardních příkazů liší záměnou znaku záznamu z dvojtečky na vykřičník. Jsou definovány dva upravené příkazy, a to konkrétně příkaz identifikace (typ 01) a příkaz čtení (typ 02).

Samotná komponenta parsru neprovádí ani čtení, ani identifikaci, jen na svém výstupu udává, že takové požadavky existují a že je zapotřebí je obsloužit. Zápis dat ale provádí, obsahuje paměť typu block RAM, do které si ukládá datové bajty Intel HEX záznamu, a po úspěšném vyhodnocení kontrolního součtu a ověření, zda se jedná o zápisový požadavek, provádí zápis do konfigurační paměti.

Komponenta také vyhodnocuje a informuje o úspěšnosti příkazů. Má výstupní signály, které určují, zda se příkaz povedlo zpracovat úspěšně, a pokud ne, tak o jakou chybu se přesně jedná. U čtecího požadavku se objeví adresa čtení a počet bajtů na výčet v signálech MEM_ADDR pro adresu zápisu a MEM_WRITE_DATA pro data zápisu, samozřejmě bez signálu MEM_WRITE který by signalizoval zápisový požadavek. Informace o čtecím požadavku jsou platné, když je výstupní signál READ_REQ v logické 1.

Když je výstupní signál IDENT_REQ v logické 1, jedná se o příkaz identity. Tento příkaz je důležitý pro konfigurační PC aplikaci z hlediska ověření, zdali se jedná o kompatibilní modul, a pro získání základních metadat o modulu.

Délka příkazu je pevně určena hodnotou prvního bajtu záznamu, počtem datových bajtů záznamu, který udává délku části záznamu, který obsahuje data pro zpracování, např. data pro zápis, vyšší slovo rozšířené adresy, atd. Pokud nemá záznam požadovaný počet znaků, tj., pokud se na vstupu objeví znak nového záznamu uvnitř existujícího záznamu, na výstupu komponenty se do stavu logické 1 uvede po dobu jednoho hodinového taktu signál INCOMPLETE_CMD_ERR. Je-li poskytnuto více znaků, než je délka záznamu, jsou tyto znaky odignorovávány.

Ověření kontrolního součtu záznamu je prováděno součtem všech bajtů záznamu, kde bajt odpovídá dvou hexadecimálním číslicím. Je-li součet 0, kontrolní součet je v pořádku. Hodnota kontrolního součtu je součet všech bajtů kromě kontrolního součtu převeden na záporné číslo ve formátu doplňku dvojky. Je-li kontrolní součet v

pořádku, parsér pokračuje ve zpracování příkazu. Není-li v pořádku, výstupní signál CSUM_ERR je uveden do stavu logické 1 po dobu jednoho hodinového taktu.

Následně je provedena kontrola typu záznamu. Pokud číslo typu záznamu není podporováno parsrem, do logické 1 je uveden signál INVALID_CMD_ERR. V případě, že příkaz podporován je, ale neobsahuje vhodný počet datových bajtů, je do logické 1 uveden signál MALFORMED_CMD_ERR. Pokud je všechno v pořádku, proběhne vykonání příkazu a signál CMD_OK (případně společně se signálem IDENT_REQ nebo READ_REQ) je po dobu jednoho hodinového taktu uveden do stavu logické 1.

V případě zápisu je možné, že se vyhodnotí adresa, která leží mimo adresový prostor konfigurační paměti a EEPROM čipu. V tom případě je uveden signál OUT_OF_RANGE_ERR do stavu logické 1 a zápis je přerušen. Je nutno poznamenat, že část zápisového požadavku mohla být zpracována úspěšně, tj., když je OUT_OF_RANGE_ERR v logické 1, neznamená to, že by paměť nebyla upravena.

3.2.5 Jednotka generace odpovědí

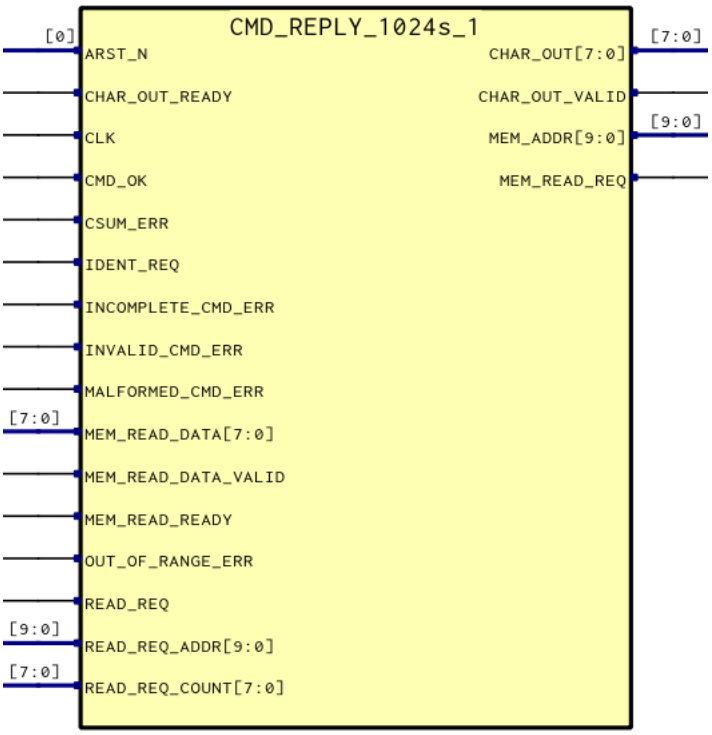
Komponenta CMD_REPLY, obsažena v souboru *cmd_reply.sv*, navazuje přímo na parsér příkazů a generuje odpovědi na zpracované příkazy na základě výstupních signálů parsru. Schematická značka komponenty je na obr. 3.15.

Podobně jako u parsru se i u jednotky generace odpovědí jedná o stavový automat zpracovávající Intel HEX příkazy, ale tentokrát jsou znaky generovány. Stejně jako u parsru jsou využívány standardní Intel HEX záznamy zápisu dat a upravené záznamy pro informování o stavu vykonání zpracovaných příkazů a identifikaci modulu. V případě jednotky generace odpovědí jsou upravené příkazy označeny znakem tečka. Byl zvolen jiný znak než u parsru, aby nedošlo k záměně příkazů a aby nebylo nutno řešit zpracování zpráv odpovědí v parsru.

Záznam pro zápis dat je generován jako odpověď na čtecí požadavek. V podstatě se jedná o příkaz, který by při zaslání do modulu nastavil stav vyčtené paměti na aktuální hodnotu. Výčet dat z konfigurační paměti provádí samotná komponenta generace odpovědí.

Pro upravené odpovědi, konkrétně pro záznam udávající úspěšné vykonání příkazu (typ 01), neúspěšné vykonání příkazu (typ 02) a identifikaci modulu (03), je využito adresové pole záznamu pro pořadové číslo odpovědi. Pomocí tohoto čísla lze ověřit, zda náhodou nedošlo někde ke ztrátě příkazu, což se dá využít pro zopakování daného příkazu.

Zpráva o identifikaci modulu může obsahovat libovolná data, jako např. typ desky, verze podporovaného příkazového rozhraní a šířka adresového prostoru. Jediným omezením je omezení maximálního počtu 255 datových bajtů v rámci jednoho



Obr. 3.15: Schematická značka FPGA komponenty jednotky generace odpovědí

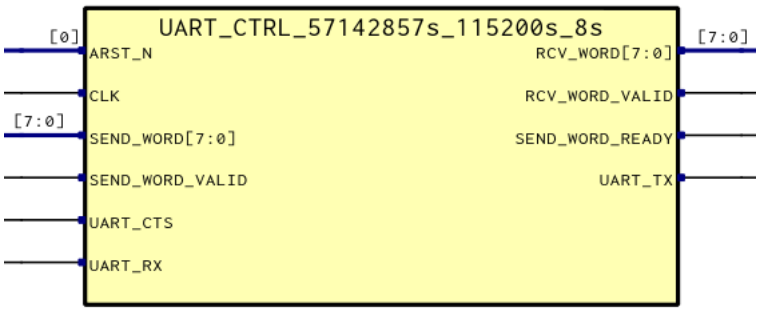
Intel HEX záznamu, který je dán šířkou pole záznamu udávajícího počet datových bajtů. V aktuální implementaci obsahuje identifikační zpráva datové hodnoty „00” a „01”, odpovídající verzi 0.1 firmwaru.

3.2.6 Jednotka UART

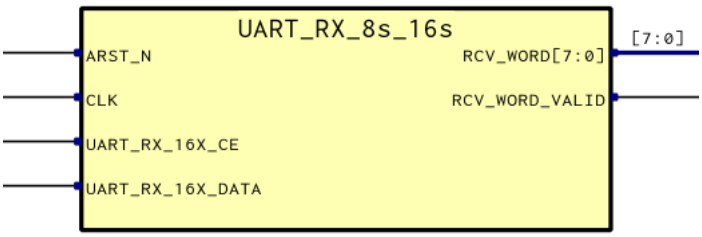
Komponenta UART_CTRL, skládající se z podkomponent UART_RX a UART_TX, realizuje obousměrnou textovou komunikaci pomocí elektrických impulsů, které jsou typicky využívány pro sériovou linku RS232. Schematická značka komponenty je na obr. 3.16, podkomponenty jsou na obr. 3.17 a obr. 3.18.

Slovo asynchronní v názvu UART znamená, že datové výstupy nemají vyveden hodinový signál, který by přesně stanovoval, kdy je na datové lince jaká hodnota. K tomuto účelu musí být využito předem dohodnutého časování přenosu. K dosažení co nejlepší spolehlivosti datového přenosu musí být časování přijímače i vysílače v co nejlepší shodě.

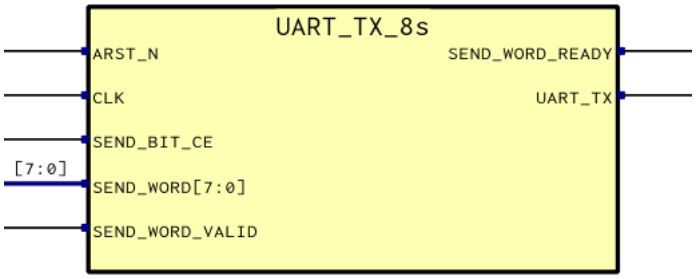
Z tohoto důvodu byl zvolen hodinový kmitočet systému 57,14 MHz, odvozen z vnějšího hodinového signálu 100 MHz dělícím poměrem 4 ku 7. Ze všech možných kmitočtů získaných celočíselným podílem a vynásobením kmitočtu 100 MHz v jednotce PLL na zvoleném FPGA čipu dovoluje kmitočet 57,14 MHz získat nejlepší



Obr. 3.16: Schematická značka FPGA komponenty jednotky UART



Obr. 3.17: Schematická značka FPGA přijímací podkomponenty jednotky UART



Obr. 3.18: Schematická značka FPGA vysílací podkomponenty jednotky UART

shodu se standardními přenosovými rychlostmi rozhraní UART. Nejedná se o přesnou shodu, ale např. pro přenosovou rychlost 115 200 bitů za sekundu dokážeme z kmitočtu $100MHz \cdot 4/7 = 57,142857MHz$ získat podílem 496 přenosovou rychlost 115 207,4 bitů za sekundu, u které se jedná jen o 0,006% odchylku od standardní hodnoty.

Dalším aspektem dohody mezi přijímačem a vysílačem je šířka slova, počet stop bitů, a zda je nebo není využita parita. Vyšší počet stop bitů může pomoct se spolehlivostí přenosu, ale způsobuje snížení propustnosti. Stejně dokáže zvýšit spolehlivost přenosu parita, která udává, zda slovo obsahuje lichý nebo sudý počet jedniček, a dokáže identifikovat poškozené datové slovo. Šířka slova bývá typicky 8 bitů pro bajt,

ale pro specifické aplikace je možné využít i jinou šířku, např. pro dosažení vyšší rychlosti přenosu znaků, je-li využita užší sada znaků.

Pro konkrétní aplikaci byla zvolena přenosová rychlost 115 200 bitů za sekundu, jelikož nejsou přenášeny velké objemy dat a nižší rychlost dovoluje získat lepší spolehlivost přenosu z hlediska parazitních jevů na DPS. Byl také zvolen jen jeden stop bit a žádný paritový bit, jelikož se zvolený rámcový formát dokáže s poškozením přenášených dat vypořádat.

Vysílací jednotka UART_TX je jednoduchý stavový automat přecházející mezi stavy rychlostí 115 207,4 stavů za sekundu, jenž generuje pro každý vstupní bajt START bit, 8 datových bitů a následný STOP bit. Zapojení této jednotky v UART_CTRL také respektuje vnější signál CTS, který stanovuje, zda je zasílání dat povoleno, a v případě, že je tento signál ve stavu logické 1 (přenos není povolen), nezahajuje stavový automat vysílání dalšího bajtu.

Přijímací jednotka UART_RX pracuje na rychlosti 16krát vyšší než vysílací jednotka. Tato vyšší rychlost je využita pro tzv. nadvzorkování přicházejících dat z vysílače a dovoluje jednotce vypořádat se s fázovým posuvem hodin přijímače a vysílače a také do určité míry s odchylkami přenosové rychlosti. Jedná se o stavový automat, který hlídá logický stav na datovém vstupu, a detekuje-li přechod z logické 1 do logické 0, vyčká určitou dobu a kontroluje ve střední části potenciálního START bitu, zda se opravdu jedná o START bit nebo jen o rušení. Je-li stav datového vstupu pořád v logické 0, zahajuje se čtení bitů v jejich střední části.

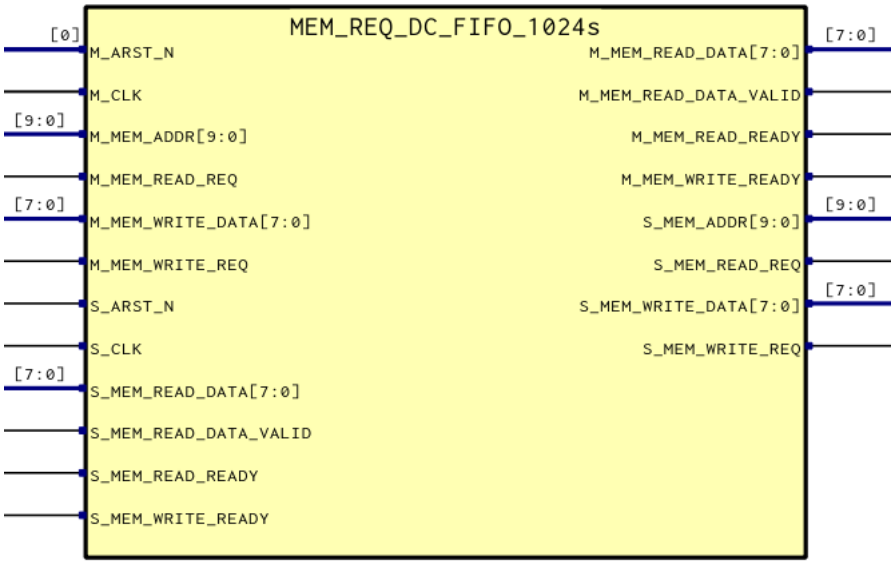
3.2.7 Generace hodin a resetu

Komponenty popsány v předchozích sekcích jsou sice ty nejdůležitější, nedokážou ale fungovat správně bez vhodně natvarovaných hodinových a resetovacích signálů.

Hodiny jsou na FPGA kitu Trenz Electronic TEL0001 generovány čipem SiTime SiT8008AI-73-XXS-100.0 a přivedeny na vývod A31 FPGA čipu. Tento vývod ale není určen pro hodinový vstup a daný FPGA čip ho nedokáže propojit s hodinovou sítí uvnitř čipu. Z toho důvodu je zapotřebí přidat do designu nějaký způsob zpracování hodin. Byly zavedeny dvě jednotky PLL, jedna pro 1:1 replikaci hodinového signálu pro konfigurační paměť a jedna pro 4:7 replikaci hodinového signálu pro vytvoření 57,14 MHz hodin vhodných pro UART.

Jelikož UART a konfigurační paměť musí se sebou vzájemně umět komunikovat, byla mezi komponentu CFG_MEM a MEM_SYNC zavedena paměť typu FIFO se dvěma hodinovými vstupy. Jelikož se jedná o poměrně složitou komponentu s implementací náchylnou na chyby, byla využita existující implementace poskytnuta firmou Lattice jako IP Core ve vývojovém prostředí Diamond. Tento IP Core byl využit uvnitř komponenty MEM_REQ_DC_FIFO (schematická značka na obr.

3.19) ve formě dvou pamětí FIFO, paměť *mem_mts_dc_fifo*, do které synchronizační jednotka zapisuje požadavky a z které si je konfigurační paměť čte, a paměť *mem_stm_dc_fifo*, do které konfigurační paměť zapisuje odpovědi na čtecí požadavky a z které jsou zaslány do synchronizační jednotky.



Obr. 3.19: Schematická značka FPGA komponenty MEM_REQ_DC_FIFO

V designu byl také využit asynchronní reset. Výhodou tohoto resetu je, že působí okamžitě a v případě FPGA Lattice MachXO2 je pro něj vyhrazen vnitřní signál GSR (Global Set/Reset), který ho dokáže velice rychle propagovat po celém čipu. Nevýhodou jsou možné problémy se stavem paměťových prvků a stavových automatů po odebrání resetu, a proto je zapotřebí sestupnou hranu asynchronního resetu zesynchronizovat s hodinovým signálem.

Jelikož jsou v systému dvě hodinové domény, jsou generovány i dva asynchronní resety, každý zesynchronizován s hodinami dané domény. Synchronizace resetu je provedena pomocí sériového zapojení D klopných obvodů, kde na vstupu mají tyto klopné obvody logickou 0 a na resetovacím vývodu mají přiveden vstupní reset, který je okamžitě převede do stavu logické 1. Změna zpět do logické 0 proběhne až když registry navzorkují logickou 0 na vstupu řetězce. Je využito více klopných obvodů, aby se zamezilo problémům způsobených možným vznikem metastabilního stavu na výstupu prvního D klopného obvodu, jelikož mezi časem odebrání resetu a náběžnou hranou hodin nemusí mít dostatek času pro spolehlivé navzorkování signálu.

Vstup do jednotek synchronizace resetu je výstup LOCK jednotek PLL. Tento výstup udává, zda je generovaný hodinový signál stabilní, a zdali je možné s ním pracovat. Z dokumentace Lattice také plyne, že PLL jednotky by měly dostat re-

setovací puls v případě pádu výstupu LOCK z logické 1 do 0, a z toho důvodu je puls i generován. Tento resetovací puls musí být dostatečně široký na to, aby v případě návratu stavu LOCK do logické 1 nevznikla nekonečná smyčka, kde reset by způsoboval krátký výpadek stavu LOCK a ten by generoval reset. Šířka tohoto resetu je daná generickým parametrem PLL_RST_CYCLE_COUNT komponent CFG_MEM_CLKGEN a UART_CLKGEN.

3.3 Návrh a implementace konfiguračního softwaru

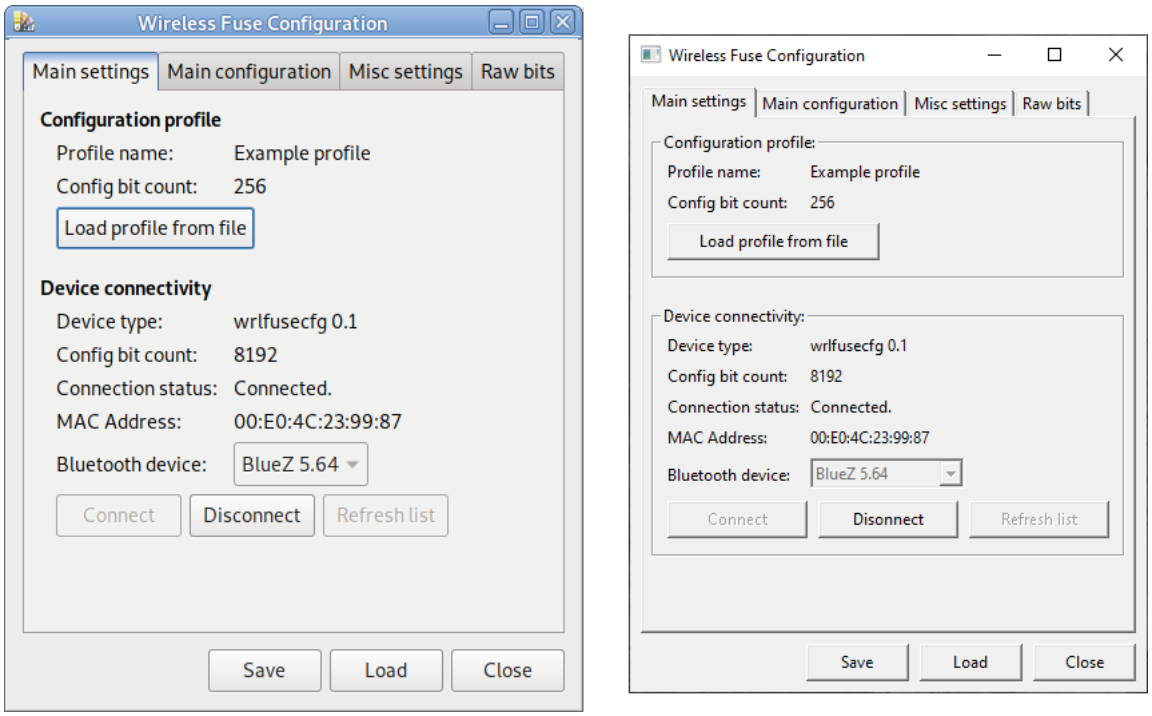
Hlavním cílem návrhu konfiguračního softwaru bylo vytvoření aplikace, která by byla uživatelsky přívětivá, flexibilní z hlediska možnosti změny konfiguračních položek pro jiný emulovaný čip, nápomocná při ladění, nezabírala příliš mnoho systémových zdrojů a byla přenositelná.

Výsledkem je aplikace s dynamickým uživatelským rozhraním, která si informace o čipu, o jeho konfiguračních pojistkách a o způsobu zacházení s pojistkami, čte z textového souboru formátu JSON. Poskytuje možnost vytváření zaškrtávacích políček, rozbalovacích seznamů a textových políček pro úpravu bitových hodnot. Zaškrtávací políčka, jako i rozbalovací seznamy, mají pro každý definovaný stav bitovou masku a hodnotu libovolné šířky a textové políčka pro úpravu bitových hodnot mají udán index nejvyššího a nejnižšího zpracovaného bitu, s možností zpracování nanejvýš 64 bitů jedním políčkem. Dále lze definovat podmínky, kdy mohou být jednotlivé konfigurační prvky aktivní. Zdrojový kód, společně s 32 a 64-bitovou verzí pro Microsoft Windows, se nachází v elektronické příloze.

Aplikace funguje na systému Microsoft Windows a na GNU/Linux, na obou systémech se chová jako nativní aplikace, která si nevyžaduje žádné balíky kompatibility nebo nezvyklé knihovny. Uživatelské rozhraní realizuje aplikace na systému Microsoft Windows pomocí API rozhraní win32 a na Linuxu pomocí knihovny GTK+ verze 3. Komunikace s modulem přes Bluetooth SPP je realizována pomocí API Winsock 2.2 na systému Microsoft Windows a pomocí BlueZ na Linuxu. Na obrázku 3.20 je vidět hlavní stránku aplikace běžící na obou systémech.

Aplikace byla napsána v programovacím jazyku C, s využitím kompilátoru GCC na obou platformách (na systému Microsoft Windows pomocí balíku mingw-w64, který poskytuje kompilátory generující kód pro Windows, a je dostupný na tomto systému v rámci vývojového prostředí msys2). Neexistuje zásadní důvod, proč by nemohl být využit i kompilátor prostředí Microsoft Visual Studio, autor s daným prostředím jen nemá moc zkušeností a využití GCC zamezuje vzniku možných právních komplikací způsobených komerčním kompilátorem.

Programovací jazyk C byl zvolen hlavně kvůli dobrým zkušenostem autora práce s daným jazykem. Autor uvažoval i o volbě programovacího jazyka Rust, který je



Obr. 3.20: Hlavní stránka konfigurační aplikace na systémech Linux a Windows

výhodný z hlediska systematického odhalování a zamezování chyb paměťových přístupů, ale nakonec z důvodu časového nátlaku a nedostatku praktických zkušeností s jazykem dospěl k závěru, že by mohla nastat situace, kde by si kód kvůli nedorozumění vyžadoval zásadní změnu struktury, a ta by mohla trvat dlouho a zamezit splnění zadání včas. Navíc je u programovacího jazyka C výhodou, že je obecně známý, hlavně v oboru elektrotechniky, kde se často pracuje s mikrokontroléry a s obslužným softwarem hardwaru, a tak využití jazyku C umožní i ostatním vývojářům zapojit se do projektu a rozšířit ho, nebo ho upravit pro své vlastní účely.

Pro zpracování dat ve formátu JSON byla využita knihovna json-c. Jedná se o Open Source knihovnu s licencí MIT, která nepředstavuje problém s možným komerčním využitím aplikace.

3.3.1 Konfigurační profily

Pro zajištění univerzálnosti vytvořené aplikace byl zaveden koncept konfiguračního profilu. Jedná se v podstatě o popis konfiguračních pojistek čipu emulovaného emulační platformou obsahující konfigurační prvky, které dovolují přirozeným a uživatelsky přívětivým způsobem stav těchto pojistek manipulovat.

Konfigurační profily jsou načteny z externích souborů. V aktuální implementaci je formát konfiguračních profilů založen na datovém formátu JSON. Jedná se

o pro člověka čitelný textový formát, určen pro úschovu a přenos strukturovaných dat. Zvolen byl hlavně kvůli své jednoduchosti a přehlednosti zápisu, obzvlášť v porovnání s formátem XML, který se také využívá pro podobné účely. [46], [47]

K tomu, aby mohl být JSON soubor považován za konfigurační profil aplikace, musí splňovat určité požadavky, konkrétně se jedná o přítomnost povinných atributů a vhodných datových typů pro jednotlivé atributy.

Atribut *type* s povinnou hodnotou „wrlfusecfg-config" určuje, že daný soubor opravdu je konfiguračním profilem navrhované aplikace. Jelikož se formát konfiguračního profilu může v budoucnu měnit, povinný atribut *cfg-fmt-ver* udává verzi formátu, aktuálně existuje jen verze 1. Budoucí verze navrhovaného programu mohou tento atribut využít pro účely zpětné kompatibility a starší verze si zase mohou ověřit, zda dokážou se souborem pracovat, nebo ne.

Následuje atribut *cfg-name*, který udává název konfiguračního profilu, a *cfg-bytes*, který udává počet konfiguračních bajtů. Název by měl být výstižný, měl by popisovat, o jaký emulovaný čip (případně rodinu čipů) se jedná a případně nějaká specifika konfigurace. Počet konfiguračních bajtů v podstatě odpovídá počtu posuvných registrů a spínačů na předchozí konfigurační desce, každý bajt dokáže popsat 8 konfiguračních bitů.

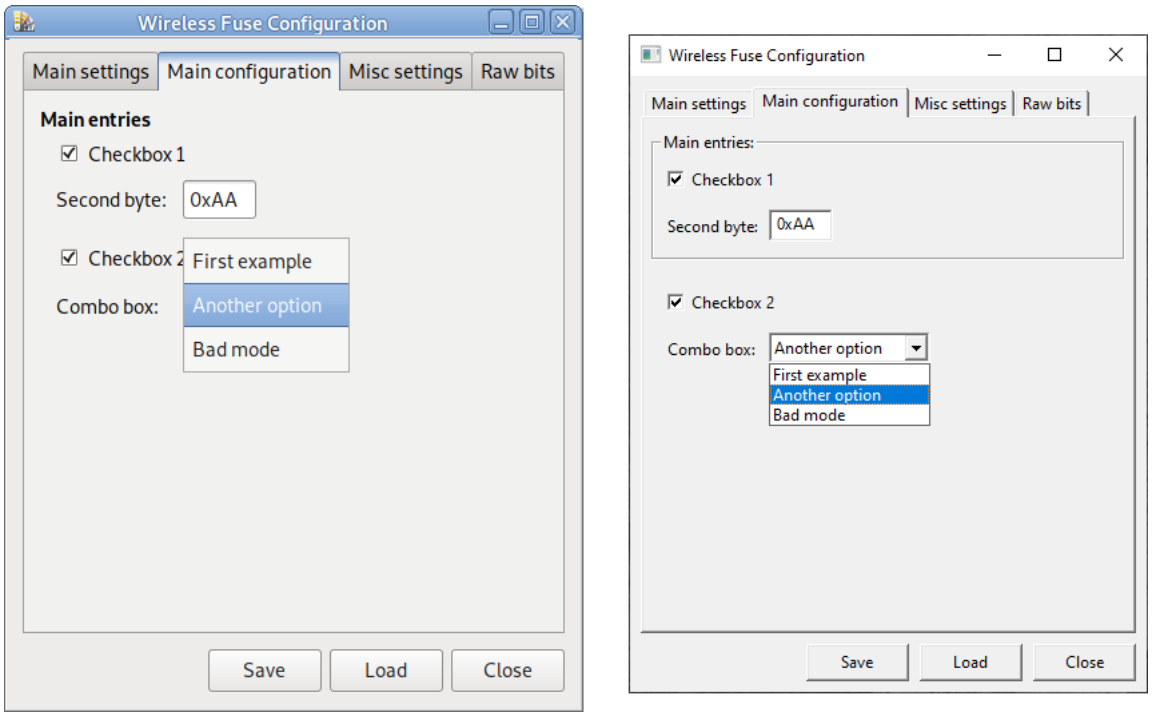
Tyto bajty musí mít nějakou výchozí hodnotu, tato hodnota je udána atributem *cfg-bytes-default*, který obsahuje pole číselných hodnot jednotlivých bajtů. Číslování začíná od 0, kde bajt s číslem 0 je bajt nejblíž k emulační platformě.

Následuje atribut *cfg-pages*, který obsahuje popis konfiguračních prvků. Prvky jsou rozděleny do stránek a uvnitř stránek jsou rozděleny do skupin. Jedná se o pole objektů popisujících jednotlivé stránky obsahující konfigurační prvky, kde atribut *page-label* určuje název v záhlaví stránky, a *page-groups* je pole obsahující objekty skupin prvků.

Skupiny prvků mají také název určen atributem *group-label* a v případě verze programu pro Microsoft Windows i orámování prvků. Orámování a název lze schovat nastavením atributu *frame-shown* na hodnotu „no", v případě hodnoty „yes" je název i rám zobrazen. Atribut *elements* je pole objektů konfiguračních prvků. Obrázek 3.21 ukazuje příklad dvou skupin konfiguračních prvků, jednu bez rámu a druhou s rámem.

Každý konfigurační prvek musí obsahovat atribut *type*, který udává, o jaký prvek se jedná, aktuálně jsou podporovány hodnoty „check-box" pro zaškrtávací políčko, „combo-box" pro rozbalovací seznam a „bit-entry" pro textová políčka pro úpravu bitových hodnot.

U každého z těchto prvků musí být také atribut *label*, který udává popisný text prvku. V případě zaškrtávacího políčka je text na pravé straně od políčka, v případě rozbalovacího seznamu a textového políčka je na levé straně.



Obr. 3.21: Stránka konfiguračních prvků se skupinou s rámem a bez rámu

Dále může mít každý konfigurační prvek volitelný atribut *active-cond*, který udává, za jakých podmínek má být prvek přístupný. Základní podmínkou je nějaký konkrétní stav konfiguračních pojistek, obsahuje atribut *cond-type* s hodnotou „bit-match" a atribut *value* obsahující pole objektů specifikujících požadovanou hodnotu bitů uvnitř bajtu. Konkrétní volba bitů je provedena pomocí atributu *mask*, obsahující bitovou masku, a atributu *value*, obsahující požadovanou číselnou hodnotu bajtu, když uvažujeme jen o bity stanovené maskou. Atribut *addr* určuje, o který bajt se přesně jedná. Pokud je specifikován víc než jeden bajt, musí se najít shoda u všech, aby byl konfigurační prvek zpřístupněn.

Nepostačuje-li tento mechanizmus, je možné vytvořit i složitější chování pomocí podmínek typu „and", „or" a „xor", které mají atribut *terms*, obsahující pole objektů podmínek, na které je aplikována daná logická operace (součin, součet, neekvivalence). Je dostupná i podmínka typu „not", která akceptuje v atributu *term* jenom jednu podmínku a obrací její logickou hodnotu.

U zaškrtávacích políček a rozbalovacích seznamů je také možnost specifikovat stav, do kterého se mají přepnout, změní-li se jejich aktivační stav. Atribut *activate-state* udává stav, který má prvek zaujmout, přechází-li z neaktivního stavu do stavu aktivního a atribut *deactivate-state* udává stav, který má prvek zaujmout, přechází-li z aktivního stavu do stavu neaktivního. Pokud nejsou tyto stavy specifikovány,

ponechává si prvek konfigurační stav před změnou aktivačního stavu. Pro zaškrtávací políčka jsou povolené hodnoty těchto atributů „checked" pro zaškrtnutý stav a „unchecked" pro prázdný stav. Pro rozbalovací seznamy jsou povolené hodnoty těchto atributů názvy prvků konkrétních seznamů.

Pro nastavení hodnot stavů zaškrtávacích políček se využívají atributy *state-on* a *state-off*. U obou těchto atributů je hodnota stejná jako u podmínky bitové shody při tvorbě podmínek: pole popisů bitové shody uvnitř libovolné skupiny bajtů. Stejný formát hodnot je také využit pro jednotlivé volby rozbalovacího seznamu, které jsou dané atributem *options*, jenž obsahuje pole objektů s atributy *name* pro název možnosti a *value* pro odpovídající hodnotu.

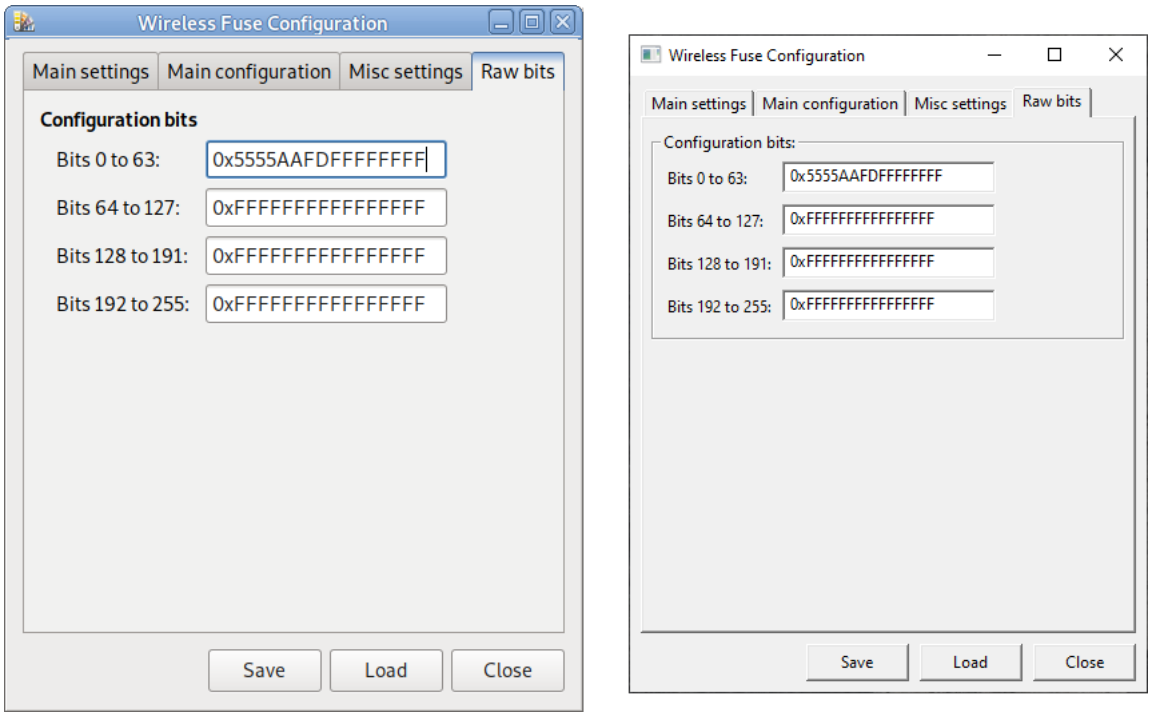
Při přechodu konfiguračního prvku z jednoho stavu do druhého nedochází k automatickému odnastavení původního stavu, tzn., že pokud dva rozličné stavy nastavují hodnoty jiných bajtů, při přechodu z jednoho stavu do druhého si bajty, které první stav nastavil, ale druhý stav nenastavuje, ponechávají svoji původní hodnotu.

Textová políčka pro úpravu bitových hodnot mají stav definován aktuální hodnotou konfigurační paměti na daném bitovém rozsahu. Dokážou zpracovávat čísla v hexadecimálním zápisu (se znaky 0x před číslem), v osmičkovém zápisu (se znaky 0o, nebo jen 0 před číslem) a v desítkovém zápisu (bez znaků před číslem). V případě změny hodnoty políčka jako následek změny stavu jiného konfiguračního prvku přechází text políčka na hexadecimální zápis. Rozsah bitů, které políčko manipuluje, je dán atributy *ms_bit* a *ls_bit*. Atribut *ms_bit* obsahuje číselný index bitu nejvyšší váhy a *ls_bit* číselný index bitu nejnižší váhy. Číslování bitů začíná na 0 u bitu nejblíž emulační platformě, jedná se o bit nejvyšší váhy a bity s vyšším indexem jsou považovány za bity nižší váhy. Tento nezvyklý způsob číslování byl zvolen z důvodu, že z principu funkce člověk předem neví, kolik přesně bajtů bude konfigurační paměť mít, a dává smysl umístit bit nejvyšší váhy na známou pozici, takže byl umístěn na adresu 0.

Kromě stránek ze souboru generuje program jednu stránku i automaticky, jedná se o stránku bitové manipulace všech konfiguračních pojistek profilu. Data pojistek jsou zobrazena po 64 bitech v textových polích, kde je uživatel může přímo upravovat a nastavovat, nebo je využít k diagnostickým účelům. Ukázka této stránky je na obr. 3.22.

3.3.2 Uživatelské rozhraní

Hlavním cílem návrhu uživatelského rozhraní bylo dosažení dynamického vytváření prvků rozhraní na základě popisu v konfiguračním profilu a oddělení kódu uživatelského rozhraní od vnitřního kódu aplikace, který provádí vyhodnocení konfigurace a její nahrávání přes Bluetooth.



Obr. 3.22: Vygenerovaná stránka bitové manipulace konfigurační aplikace

Oddělení kódu bylo důležité z hlediska možnosti změny implementace uživatelského rozhraní, což může být užitečné např. v případě požadavku na vytvoření mobilní aplikace. Její vytvoření by nemělo být problematické, jelikož stačí jen naimplementovat definované programové rozhraní a zbytek kódu s ním bude umět pracovat.

V tomto duchu bylo uživatelské rozhraní implementováno se dvěma rozličnými knihovnami pro tvorbu grafických aplikací: API win32 na Microsoft Windows a GTK+ na Linuxu. Obě tato uživatelská rozhraní implementují programové rozhraní v souboru *gui.h*.

Původně bylo zamýšleno vytvoření aplikace čistě jen s GTK+, jelikož je práce s touto knihovnou mnohem jednodušší než s API win32, a knihovna funguje i na systému Microsoft Windows, ale je na tomto operačním systému problematická z hlediska distribuce binární verze a z hlediska vzhledu, který je atypický pro aplikace na systému Microsoft Windows. Kdyby byla knihovna GTK+ na systému Microsoft Windows využita, musel by program mít svůj vlastní instalátor, zabíral by řádově desítky až stovky megabajtů a musely by být vyřešeny i právní požadavky autorů knihovny, jelikož je vydaná pod licencí GNU LGPL. Nebylo by to praktické.

Soubor *gui_gtk.c* obsahuje implementaci uživatelského rozhraní pomocí knihovny GTK+ verze 3. Implementace je přímočará, vytváří se okno obsahující prvek *GtkNo-*

tebook a skupinu tlačítek pro načtení konfigurace, uložení konfigurace a ukončení programu. Prvek *GtkNotebook*, který představuje záložkové zobrazení jednotlivých stránek konfiguračních prvků, obsahuje po startu programu jenom jednu stránku, sloužící pro načtení konfiguračního profilu a pro připojení se přes Bluetooth ke konfiguračnímu modulu. Po úspěšném načtení konfiguračního profilu se vytvářejí stránky prvků pro daný profil. GUI rozhraní také přebírá vlastnictví nad strukturou konfiguračního profilu a zbytek programu musí využít dostupného programového rozhraní k získání přístupu k informacím ohledně konfiguračního profilu.

Verze GUI rozhraní pro API win32 je rozdělena do dvou souborů, do *gui_win32.c* a *gui_win32_layout.c*. Na rozdíl od knihovny GTK+ neposkytuje API rozhraní win32 kód pro řešení rozložení grafických prvků. Každý grafický prvek je v tomto rozhraní své vlastní okno, které je na obrazovku umístěno určením jeho pozice a rozměrů v rodičovském okně. Navíc je rozhraní docela nerovnoměrné v tom, jak se tato okna chovají, zejména rozbalovací seznam se chová docela zvláštně v tom, že jeho rozměry, určeny při vytvoření, zahrnují i samotný seznam v rozbaleném stavu, a výška prvku se nastavuje samostatnou funkcí *ComboBox_SetItemHeight()*.

Soubor *gui_win32_layout.c* obsahuje kód pro řešení rozložení grafických prvků v rámci definovaného prostoru na okně. Rozhraní bylo z části inspirováno rozhraním knihovny GTK+. Vytváření grafických prvků se provádí pomocí funkcí *gui_*_alloc()*. Tyto funkce vracejí strukturu, pomocí které lze daný prvek vykreslit na obrazovce, nebo ho přidat do skupiny s jinými prvky pro rozložení na obrazovce. Pro tento účel jsou k dispozici řádky prvků, sloupce prvků a skupinové rámy. Zajímavostí skupinových rámů je, že se z hlediska API rozhraní win32 jedná o tlačítka, které mají styl BS_GROUPBOX. [48]

Po vytvoření lze grafický prvek umístit na obrazovku. Předtím bývá smysluplné ověřit, zda je na obrazovce dostatek místa pro daný prvek nebo skupinu prvků, na to slouží funkce *gui_elem_get_min_size()*. Po zajištění dostatku místa na obrazovce (např. zvětšením okna) lze u prvku nastavit jeho polohu a rozměry pomocí funkce *gui_elem_set_pos()*. Při tomto volání dochází k vytvoření nebo přesunu okna grafického prvku na požadovanou polohu a rozměry.

Nastavení minimálních rozměrů okna v API win32 lze dosáhnout pomocí zprávy WM_GETMINMAXINFO, která je zasílána programu v případě nadcházející změny rozměrů okna. Tato funkce poskytuje strukturu typu MINMAXINFO, která obsahuje prvek *ptMinTrackSize*, u něhož se jedná o ukazatel na strukturu udávající minimální velikost okna. Tyto rozměry jsou ale vnější rozměry okna včetně záhlaví a rámu a pro převod mezi vnitřními a vnějšími rozměry je zapotřebí využít funkci *AdjustWindowRect()*.

Pro realizaci záložkového rozložení je využit prvek třídy WC_TABCONTROL. Na rozdíl od *GtkNotebook* se ale nejedná o prvek, který by sám obsahoval stránky

prvků na zobrazení a přepínal mezi nimi, tento prvek poskytuje jen mechanizmus přidání záložek (pomocí funkce *TabCtrl_InsertItem()*) a při volbě záložky jen informuje okno pomocí zprávy typu WM_NOTIFY s kódem TCN_SELCHANGE že taková změna nastala a je na programátorovi, aby prvky požadované stránky vykreslil. Funkce *TabCtrl_AdjustRect()* pomáhá s výpočtem prostoru uvnitř prvku WC_TABCONTROL a také s určením potřebné velikosti tohoto prvku pro obsažení určitého množství podřazených prvků.

Pro jednoduchost realizace volby mezi stránkami je doporučeným postupem využití samostatných oken (poznámka: ve Windowsu je každý jeden grafický prvek plnohodnotné okno), která by obsahovala prvky odpovídající zvolené stránce, a při změně stránky by se aktuální okno schovalo a nově zvolené okno zobrazilo. Tento postup je využit i v navrhované aplikaci, je definována třída oken GUI_PAGE_CLASS_NAME, které instance jsou vkládány do prvku na přepínání záložek a které obsahují konfigurační prvky. Součástí implementace je mechanizmus, který zajišťuje zvětšení rozměrů hlavního okna v případě, že by se obsah zvolené záložky nevešel do okna aktuálních rozměrů, k tomu je využita funkce *SetWindowPos()*.

Dalším důležitým aspektem uživatelského rozhraní je využitá tabulka znaků. GTK+ i win32 obě podporují tabulku znaků Unicode, která obsahuje znaky všech jazyků světa, ovšem neshodují se na kódování této tabulky znaků. Zatím co GTK+ využívá kódování UTF-8, které je zpětně kompatibilní s tabulkou znaků ASCII, API rozhraní systému Microsoft Windows využívají kódování UTF-16, které dovoluje jednodušší znakovou manipulaci se znaky mimo tabulky ASCII, ale využívá 16-bitové znaky. Jelikož oba způsoby kódování popisují stejnou tabulku znaků, lze je mezi sebou navzájem a beze ztrát převádět.

Pro hlavní program bylo zvoleno vnitřní kódování UTF-8, díky své zpětné kompatibilitě s tabulkou znaků ASCII a širokému využití na internetu a v textových souborech, a v případě práce s API rozhraními systému Microsoft Windows jsou řetězce znaků UTF-16 převáděny na UTF-8 a zpět. Z této volby plyne, že i JSON soubory konfiguračních profilů jsou ve formátu UTF-8.

3.3.3 Funkce konfiguračních prvků

Součástí programového rozhraní modulu uživatelského rozhraní jsou funkce pro získání popisu konfiguračních prvků načtených z JSON souboru. Ke každému prvku je přiřazen ukazatel dovolující uživateli nastavovat stav prvku a přiřadit mu funkci, kterou má zavolat v případě změny stavu, a ukazatel na strukturu s informacemi z JSON souboru odpovídajícími konkrétnímu konfiguračnímu prvku.

Tyto dva ukazatele využívá hlavní program, implementován v souboru *tool.c*, pro realizaci konfigurace. Po úspěšném načtení konfiguračního profilu ze souboru

hlavní program posbírá ukazatele na konfigurační prvky a nastaví u nich funkce pro řešení změny stavu. Následně u každého jednoho prvku vyhodnotí aktuální stav v závislosti na datech výchozí konfigurace z JSON souboru. V této fázi vyhodnocení stavu není řešena problematika aktivačního a deaktivačního stavu, aby bylo možné do JSON souboru zapsat opravdu libovolnou hodnotu konfiguračních bitů a program je interpretoval a neměnil. V případě, že výchozí hodnota neodpovídá hodnotám, které konfigurační prvky nastavují, jsou uvedeny do neurčitého stavu, který uživateli naznačuje, že skutečná hodnota konfiguračních pojistek neodpovídá ani jedné variantě, které dané prvky poskytují.

Následně je hlavní program volán v případě změny stavu prvků. Jako první krok provede změnu dat konfiguračních pojistek a zaznamená si aktuální stav konfiguračního prvku. Následně iteruje přes všechny existující konfigurační prvky a kontroluje, zda u nich náhodou nedošlo ke změně stavu. V případě, že se konfiguračnímu prvku změnil aktivační stav, a u této změny byl definován stav, do kterého má prvek přejít, je stav prvku změněn, pojistky jsou aktualizovány a funkce *tool_update_gui_elems_try()* vrací návratovou hodnotu *false*. V tomto případě je nutno zavolat funkci znovu, jelikož se mohl změnit stav prvků, které funkce už zpracovala. V případě úspěšného aktualizování stavu všech konfiguračních prvků vrací funkce návratovou hodnotu *true*. Funkci je vhodné volat v kontextu negované podmínky podmíněné smyčky *while()* bez těla. Z tohoto popisu plyne, že je možné vytvořit stav nekonečné smyčky v programu, pokud by v konfiguračním profilu existoval kruh prvků, které by si navzájem měnily při změně aktivačního stavu stav (funkce by nikdy nevrátila hodnotu *true*), takovémuto stavu je nutno při návrhu konfiguračního profilu zabránit.

3.3.4 Bluetooth komunikace

Stejně jako u grafického uživatelského rozhraní je část kódu pro práci s Bluetooth SPP zařízeními, která je závislá na platformě, rozdělena do vlastního souboru s programovým rozhraním, které na platformě závislé není. Takovýmto způsobem byla implementována Bluetooth SPP komunikace na systému Microsoft Windows v souboru *tool_bt_ws2.c* pomocí API rozhraní Winsock 2.2 a na Linuxu v souboru *tool_bt_bluez.c* pomocí API rozhraní BlueZ. Oba tyto soubory mají i hlavičkový soubor obsahující datové typy specifické pro tyto dva systémy, a pak spojený hlavičkový soubor *tool_bt.h*, který obsahuje rozhraní nezávislé na platformě.

U obou implementací se pracuje se zařízeními Bluetooth s profilem SPP podobně jako s internetovým připojením typu TCP: Po připojení na server (v našem případě za server považujeme konfigurační modul) získá program zásuvku (socket), do které může zasílat a číst z ní sekvenční data pomocí funkcí *send()* a *recv()*. Hlavní roz-

díly jsou v procesu navázání komunikace a získání zásuvky a ve způsobu získání a hodnotách chybových kódů v případě neúspěchu. Zatímco na Linuxu a jiných UNIXových operačních systémech se pro získání chybových kódů využívá globální proměnná *errno* a pro získání textového popisu chyby slouží funkce *strerror()*, na systému Microsoft Windows s API rozhraním Winsock 2.2 slouží pro získání kódu chyby funkce *WSAGetLastError()* a pro získání textového popisu chyby lze využít funkci *FormatMessage()* s parametrem FORMAT_MESSAGE_FROM_SYSTEM.

Způsob detekce okolních Bluetooth zařízení je také na obou systémech odlišný. Zatímco na Linuxu s BlueZ na to slouží funkce *hci_inquiry()* vyplňující pole struktur *inquiry_info* obsahující adresy okolních zařízení a pro získání názvu zařízení s danou adresou slouží funkce *hci_read_remote_name()*, na systému Microsoft Windows probíhá hledání okolních zařízení iterativně. Funkce *WSALookupServiceBegin()* s parametrem pro hledání Bluetooth zařízení ve formě WSAQUERYSET struktury s položkou *dwNameSpace* s hodnotou NS_BTH zahajuje proces hledání a funkce *WSALookupServiceNext()* slouží pro iterativní získávání popisu nalezených zařízení.

V rámci kódu závislého na platformě je implementována funkce *tool_bt_devices_query()* pro vytvoření a aktualizaci seznamu okolních zařízení, funkce *tool_bt_plat_connect()* pro vytvoření spojení s konkrétním zařízením nalezeným předchozí funkcí, funkce *tool_bt_send_full()* pro zaslání určitého počtu znaků, funkce *tool_bt_recv_full()* pro přijetí určitého počtu znaků a funkce *tool_bt_recv()* pro přijetí nanejvýš určitého počtu znaků.

Při vytvoření komunikační zásuvky je také pomocí konstanty TOOL_BT_RCV_TIMEOUT_SEC a funkce *setsockopt()* nastaven časový limit pro přijetí znaků těmito funkcemi, aktuální hodnota jsou 2 sekundy. Časový limit je využit pro zabránění možného zaseknutí programu v případě, že by zařízení neodpovědělo na příkaz (nebo kdyby kvůli ztrátě na přenosové lince odpověď nepřišla v plném rozsahu), a i z praktického hlediska pro detekci odpovědí v situacích, kdy není jasné, zda odpověď přijde nebo ne.

Na těchto funkcích je postavena část Bluetooth kódu programu nezávislá na platformě, implementována v souboru *tool_bt.c*. Jejím jádrem je mechanizmus s vyrovnávací pamětí pro zasílání a přijímání textu přes Bluetooth SPP po bajtech, obdobně jak fungují funkce *fgetc()* a *fputc()* pro práci se soubory a mechanizmus pro zasílání a přijímání Intel HEX záznamů.

Kód pro parsování a serializaci Intel HEX záznamů byl původně napsán v rámci programu emulujícího Bluetooth rozhraní konfiguračního modulu, jenž je součástí balíku zdrojových kódů konfigurační aplikace, a sdílí s aplikací některé funkce včetně zmíněných funkcí pracujících s Intel HEX záznamy. Cílem emulačního programu bylo zrychlení vývoje a ladění konfiguračního softwaru, jelikož dovoluje jednodušeji ana-

lyzovat datový přenos a generovat různé chybové stavy pouhou úpravou C kódu a spuštěním na Linuxovém stroji. Emulátor má podporu Bluetooth jen na Linuxu, rozšíření o podporu Bluetooth přes rozhraní Winsock 2.2 na systému Microsoft Windows by bylo triviální a nebylo provedeno jen z důvodu časového nátlaku.

Před zahájením hlavní komunikace je uvnitř funkce *tool_bt_connect()* provedena synchronizace komunikačního toku a identifikace zařízení. Synchronizace se skládá ze čtyř kroků: smazání všech příchozích dat, která modul vyslal před zahájením procesu synchronizace, zaslání příkazu identifikace, přijetí odpovědi a v případě, že se jedná o odpověď typu „chyba: neúplný příkaz" přijetí další odpovědi. Pokud tento proces proběhne v pořádku, modul by měl být připraven na přijímání příkazů a konfigurační program by měl být připraven na přijímání odpovědí. Následuje kontrola odpovědi, zpracování identifikačních dat a uložení pořadového čísla odpovědi, jenž poslouží pro kontrolu, zda odpovědi na zaslané příkazy opravdu jsou odpovědi na dané příkazy.

Zápis a čtení dat z konfigurační paměti modulu provádí funkce *tool_bt_send_cfg_write_req()* a *tool_bt_send_cfg_read_req()*. Obě funkce využívají v podstatě totožný algoritmus (obr. 2.4), kde zasílají do modulu příkaz, získávají zpět odpověď, a v případě neúspěchu tento postup opakují, s maximálním počtem dovolených opravných pokusů udaných konstantou TOOL_BT_RETRY_COUNT_MAX. Některé chybové stavy, konkrétně chybějící odpověď nebo odpověď se špatným pořadovým číslem, si vyžadují zopakování synchronizačního postupu popsaného v předchozím odstavci. Nepovede-li se požadavek vyřídit ve stanoveném počtu pokusů, funkce zobrazí chybovou hlášku odpovídající poslední chybě a vrací chybový kód. Aktuálně je dovolený počet opravných pokusů nastaven na 5.

3.4 Zhodnocení výsledků

Byl navržen a zrealizován funkční prototyp bezdrátového komunikačního modulu pro nahrávání konfiguračních bitů do emulační platformy, včetně FPGA firmwaru a ovládacího softwaru pro PC. Výroba i testování samotného modulu proběhlo z větší části v design centru Onsemi v Rožnově pod Radhoštěm, kde byla ověřena základní funkcionalita modulu a jeho schopnost interagovat s emulační platformou a nahrát do ní konfiguraci zaslanou přes Bluetooth.

Testování ale z důvodu nedostatku času nebylo dostatečně důkladné a existují obavy z toho, že přenos konfiguračních bitů není 100% spolehlivý. Může to souviset s několika faktory, jako např. způsob vzorkování a vyhodnocování řídících signálů, konkrétní hodnoty ochranných rezistorů, možnost, že emulační platforma vyčte konfiguraci dřív, než si ji modul vyčte z EEPROM, atd.

Také nebylo provedeno dostatečně důkladné testování komunikačního protokolu, a obzvlášť ne na skutečném modulu. Většina testování komunikačního protokolu

proběhla pomocí emulátoru, který replikuje komunikační rozhraní bezdrátového modulu, jehož kód pro zpracování a generování Intel HEX záznamů je přímým překladem popisu v SystemVerilogu pro komunikační modul, a který dovoluje rychlejší a flexibilnější testování s širšími možnostmi diagnostiky. Tento postup byl ale riskantní z hlediska možného nenalezení některých situací, které u skutečné desky nastávají, ale u emulátoru ne.

Celkově byl u vypracování diplomové práce problém s nedostatkem času a projevil se i na kvalitě výstupů a nesplnění některých osobních cílů autora, jako např. chybějící FIFO na vstupu parsru, nedokončené testbenche a to, že uživatelské rozhraní se při vyhledávání Bluetooth zařízení zasekává (vyhledávání probíhá v hlavním programovém vláknu).

To ale neznamená, že výstupy nejsou užitečné. I v aktuálním stavu je bezdrátový modul spolu s konfiguračním programem zajímavou alternativou desky se spínači a posuvnými registry, poskytuje jednoduchý a uživatelsky přívětivý způsob nastavování emulovaných konfiguračních pojistek a také pomáhá zamezovat omylům, které by mohly nastat u ručního nastavování velkého množství fyzických spínačů. Všechny softwarové části projektu byly také navrženy s ohledem na modulárnost a možnost využití v jiných projektech, takže i v případě, že by bylo nutné provádět zásadnější změny v projektu, úprava kódu pro tyto změny by neměla být problematická.

Pro autora osobně je obzvlášť zajímavý kód pro práci s win32 API, jelikož se jedná o první plnohodnotnou grafickou aplikaci, kterou pomocí daného API napsal, a plánuje využít vytvořený kód i v jiných osobních projektech. Modulární struktura kódu s jasně definovanými vnitřními rozhraními také umožňuje jeho jednoduché rozšíření, např. v případě zájmu o vytvoření mobilní aplikace, nebo o přidání dalších konfiguračních prvků, jako jsou např. rádiová tlačítka.

Dalším problematickým aspektem návrhu a výroby modulu je jeho cena. Byly zvoleny docela drahé součástky a kvůli bezdrátové komunikační bráně Lantronix xPico 250 bylo zapotřebí vyrobit i šablonu pro nanášení pájecí pasty. Volba součástek byla z velké míry určena aktuální polovodičovou krizí a požadavky ze strany Onsemi, např. FPGA kit byl zvolen, protože ho konzultant měl ve skříni. Jelikož se jednalo o kusovou výrobu prvotního prototypu, tento aspekt nepředstavuje až tak zásadní problém a i v případě úplné změny modulu by mělo být pořád možné využít většinu softwarových výstupů práce.

Možná částečně problematická byla také volba programovacího jazyka C pro vypracování komunikačního softwaru. Výhodou jazyka C je rychlost kompilace kódu, dobrá přenositelnost a malá paměťová stopa standardní knihovny a základních jazykových konstruktů. Nevýhodou ovšem je pracnost programování a jednoduchost vytvoření chyby z nedbalosti a jazyk typicky není doporučován pro vytváření grafických aplikací. V posledních letech se značně vylepšily diagnostické nástroje pro

vývoj kódu v jazyku C, ale i tak existují jiné jazyky, které by mohly být pro projekt výhodnější.

Poznámka o pracnosti a jednoduchosti vytvoření chyby se také vztahuje na aplikační rozhraní win32, které neobsahuje mechanizmy řešící rozložení grafických prvků na obrazovce a které má značně nerovnoměrné a do jisté míry nepředvídatelné chování, autor ztratil několik dní práce laděním chyb způsobených tím, že se rozhraní chovalo jinak, než očekával, a že dostupná dokumentace z webových stránek firmy Microsoft nebyla moc detailní. Naštěstí lze toto rozhraní jednoduše nahradit a i na systému Microsoft Windows funguje verze aplikace s GTK+ rozhraním. Ta má ale na Windowsu svá vlastní specifika, která ji předurčují pro složitější aplikace, kde si programátor může dovolit použití instalátoru a distribuci celého balíku knihovny GTK+ a ostatních knihoven, na kterých GTK+ závisí. Pro porovnání aplikace s win32 rozhraním se zapnutými optimalizacemi a vypnutými ladícími symboly zabírá jen zhruba 200 KB prostoru na disku, což je vynikající z hlediska přenositelnosti.

Závěr

Cílem této práce byla realizace bezdrátového konfiguračního modulu, který by dovolil nahradit stávající konfigurační rozhraní emulační platformy založené na stovkách spínačů odpovídajících konfiguračním pojistkám emulovaného integrovaného obvodu bezdrátovým modulem s uživatelsky přívětivým konfiguračním programem pro počítač typu PC.

V práci byl proveden rozbor možností bezdrátové komunikace na krátké vzdálenosti. Na základě požadavku co nejširší kompatibility s periferní výbavou běžných laptopů a při zohlednění vnitřních norem pro bezpečnost sítě design centra Onsemi v Rožnově byla zvolena tradiční varianta technologie Bluetooth, pracující na volně využitelném kmitočtovém pásmu 2,4 GHz s profilem SPP určeným pro náhradu sériového portu typu RS232, a integrovaná WiFi a Bluetooth brána xPico 250 firmy Lantronix s UART rozhraním s hardwarovým řízením toku.

Následně byl proveden rozbor dostupných rozhraní na emulační platformě, do kterých by se nový modul dal zapojit, a na základě požadavku pro zpětnou kompatibilitu se starším modulem bylo zvoleno rozhraní postaveno na principu výčtu z posuvného registru. Charakter tohoto rozhraní vedl k využití programovatelného logického obvodu a požadavek na zachování konfigurace i po odpojení napájecího napětí vedl k využití EEPROM čipu.

Poté byl proveden rozbor problematiky komunikace s modulem a navržen jednoduchý komunikační protokol založený na datovém formátu Intel HEX, s podporou čtení a zápisu konfiguračních dat a identifikace modulu.

Modul byl následně vyroben, byl pro něj napsán FPGA firmware a vytvořen konfigurační program pro PC fungující jako nativní aplikace na systémech Microsoft Windows a GNU/Linux, s dynamickým uživatelským rozhraním a načítáváním popisu konfiguračních pojistek a způsobu jejich manipulace z textového souboru formátu JSON.

Hlavní nevýhody navrhovaného modulu jsou vysoká pořizovací cena součástek, vysoká proudová spotřeba a volba poměrně složitého hardwaru vzhledem k typu úlohy. Ty jsou kompenzovány skutečností, že se jedná o kusovou výrobu určenou pro vývojové účely ve firmě, kde bezpečnost a široká kompatibilita jsou důležitější faktory než pořizovací cena a spotřeba.

Také nebylo z časových důvodů provedeno dostatečně důkladné testování modulu, a nelze říct, zda funguje opravdu spolehlivě. Princip funkce byl ale ověřen, a všechen kód byl napsán modulárně, takže by teď neměl být problém v doladění a dokončení projektu, i kdyby si případně vyžadoval zásadnější změny modulu.

Literatura

[1] *Qi Specification* [online]. Piscataway, NJ: Wireless Power Consortium, 2021, poslední aktualizace ledna 2021 [cit. 26. 11. 2021]. Dostupné z URL: <https://www.wirelesspowerconsortium.com/data/downloadables/3/3/2/3/qi-v13-public.zip>

[2] STANLEY, R.: *Text-book on wireless telegraphy.* Nové vydání. London, New York: Longmans, Green, vol. 1, 1919. 471 s.

[3] NOVÁČEK, Z.: *Elektromagnetické vlny, antény a vedení Přednášky.* Elektronické učební texty. Brno: FEKT VUT v Brně, 2008. 143 s.

[4] URBAN, F., MIKEL B.: *Optoelektronika.* Elektronické učební texty. Brno: FEKT VUT v Brně, 2003. 304 s.

[5] TOULSON, R., WILMSHURST, T.: Chapter 11 - Wireless Communication - Bluetooth and Zigbee. *Fast and Effective Embedded Systems Design.* 2. vyd. Oxford: Newnes, 2017, s. 257–290. ISBN 978-008-10-0880-5. DOI 10.1016/B978-0-08-100880-5.00011-6.

[6] GHOSH, R.K.: *Wireless Networking and Mobile Data Management.* Singapore: Springer, 2017. 546 s. ISBN 978-981-10-3941-6. DOI 10.1007/978-981-10-3941-6.

[7] GOMEZ, C., OLLER, J., PARADELLS, J.: Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. *Sensors*, 2012, vol. 12, no. 9, s. 11734–11753. ISSN 1424-8220. DOI 10.3390/s120911734.

[8] PALMER, M.: *Selecting an MCP21XX Device for IrDA Applications* [online]. Chandler, AZ: Microchip Technology Inc., 2004, poslední aktualizace srpna 2004 [cit. 7. 12. 2021]. Dostupné z URL: <http://ww1.microchip.com/downloads/en/AppNotes/91073b.pdf>

[9] Silicon Laboratories, Inc.: *UART Flow Control* [online]. rev. 1.3. Austin, TX: 2017, poslední aktualizace ledna 2017 [cit. 7. 12. 2021]. Application note AN0059.0. Dostupné z URL: <https://www.silabs.com/documents/public/application-notes/an0059.0-uart-flow-control.pdf>

[10] Microchip Technology Inc.: *MCP2120 Infrared Encoder/Decoder* [online]. Chandler, AZ: 2007, poslední aktualizace února 2007 [cit. 7. 12. 2021]. Datasheet DS21618B. Dostupné z URL: <http://ww1.microchip.com/downloads/en/devicedoc/21618b.pdf>

[11] Microchip Technology Inc.: *MCP2122 Infrared Encoder/Decoder* [online]. Chandler, AZ: 2007, poslední aktualizace února 2007 [cit. 7. 12. 2021]. Datasheet DS21894C. Dostupné z URL: <https://ww1.microchip.com/downloads/en/DeviceDoc/21894c.pdf>

[12] Microchip Technology Inc.: *MCP2150 IrDA Standard Protocol Stack Controller Supporting DTE Applications* [online]. Chandler, AZ: 2013, poslední aktualizace ledna 2013 [cit. 7. 12. 2021]. Datasheet DS21655C. Dostupné z URL: <http://ww1.microchip.com/downloads/en/devicedoc/21655C.pdf>

[13] Microchip Technology Inc.: *MCP2155 IrDA Standard Protocol Stack Controller Supporting DCE Applications* [online]. Chandler, AZ: 2013, poslední aktualizace ledna 2013 [cit. 7. 12. 2021]. Datasheet D21690B. Dostupné z URL: <https://ww1.microchip.com/downloads/en/DeviceDoc/21690B.pdf>

[14] Microchip Technology Inc.: *MCP2140 IrDA Standard Protocol Stack Controller With Fixed 9600 Baud Communication Rate* [online]. Chandler, AZ: 2012, poslední aktualizace prosince 2012 [cit. 7. 12. 2021]. Datasheet DS21790B. Dostupné z URL: <https://ww1.microchip.com/downloads/en/DeviceDoc/21790B.pdf>

[15] Espressif Systems Co., Ltd.: *ESP32 Series Datasheet v3.8* [online]. Shanghai: 2021, poslední aktualizace října 2021 [cit. 8. 12. 2021]. Dostupné z URL: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>

[16] Espressif Systems Co., Ltd.: *ESP32-WROOM-32 Datasheet v3.2* [online]. Shanghai: 2021, poslední aktualizace srpna 2021 [cit. 8. 12. 2021]. Dostupné z URL: <https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf>

[17] Espressif Systems Co., Ltd.: *ESP32-WROVER Datasheet v2.5* [online]. Shanghai: 2021, poslední aktualizace srpna 2021 [cit. 8. 12. 2021]. Dostupné z URL: <https://www.espressif.com/sites/default/files/documentation/esp32-wrover_datasheet_en.pdf>

[18] WiFi Alliance: *Wi-Fi CERTIFIED™ Interoperability Certificate WFA78954* [online]. Austin, TX: 2018, datum certifikace 24. 8. 2018 [cit. 8. 12. 2021]. Certification ID: WFA78954. Dostupné z URL: <https://api.cert.wi-fi.org/api/certificate/download/public?variantId=66309>

[19] WiFi Alliance: *Wi-Fi CERTIFIED™ Interoperability Certificate WFA77915* [online]. Austin, TX: 2019, datum certifikace 18. 9. 2019 [cit. 8. 12. 2021]. Certification ID: WFA77915. Dostupné z URL: <https://api.cert.wi-fi.org/api/certificate/download/public?variantId=34068>

[20] Bluetooth SIG: *ESP32-WROOM-32 Bluetooth Module Declaration Details* [online]. Kirkland, WA: 2021, datum certifikace 16. 6. 2017 [cit. 8. 12. 2021]. Declaration ID: D035785. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/27858>

[21] Bluetooth SIG: *ESP32-WROVER Bluetooth Module Declaration Details* [online]. Kirkland, WA: 2021, datum certifikace 20. 4. 2018 [cit. 8. 12. 2021]. Declaration ID: D039625. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/58727>

[22] Silicon Laboratories, Inc.: *EFR32MG22 Wireless Gecko SoC Family Data Sheet* [online]. rev. 1.1. Austin, TX: 2021, poslední aktualizace června 2021 [cit. 9. 12. 2021]. Dostupné z URL: <https://www.silabs.com/documents/public/data-sheets/efr32mg22-datasheet.pdf>

[23] Silicon Laboratories, Inc.: *MGM220P Wireless Gecko Module Data Sheet* [online]. rev. 1.0. Austin, TX: 2020, poslední aktualizace června 2020 [cit. 9. 12. 2021]. Dostupné z URL: <https://www.silabs.com/documents/public/data-sheets/mgm220p-datasheet.pdf>

[24] Silicon Laboratories, Inc.: *Programming Internal Flash Over the Serial Wire Debug Interface* [online]. rev. 1.02. Austin, TX: 2013, poslední aktualizace listopadu 2013 [cit. 9. 12. 2021]. Application note AN0062. Dostupné z URL: <https://www.silabs.com/documents/public/application-notes/an0062.pdf>

[25] Bluetooth SIG: *RF-PHY Component Qualification of the EFR32BG22 and EFR32MG22 SoCs* [online]. Kirkland, WA: 2020, datum certifikace 13. 3. 2020 [cit. 9. 12. 2021]. Declaration ID: D044525. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/104412>

[26] Bluetooth SIG: *Wireless Gecko Link Layer based on Core Specification 5.2 Declaration Details* [online]. Kirkland, WA: 2020, datum certifikace 16. 4. 2020 [cit. 9. 12. 2021]. Declaration ID: D049533. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/105576>

[27] Bluetooth SIG: *Wireless Gecko Host based on Core Specification 5.2 Declaration Details* [online]. Kirkland, WA: 2020, datum certifikace 27. 3. 2020

[cit. 9. 12. 2021]. Declaration ID: D049534. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/104376>

[28] Lantronix, Inc.: *xPico 200 Series Embedded Wi-Fi Gateway Data Sheet* [online]. rev. M. Irvine, CA: 2021, poslední aktualizace září 2021 [cit. 10. 12. 2021]. Part Number 900-818. Dostupné z URL: <https://www.lantronix.com/download-xpico-200-series-data-sheet/>

[29] Lantronix, Inc.: *xPico 200 Series Integration Guide* [online]. Irvine, CA: 2021 [cit. 10. 12. 2021]. Dostupné z URL: <https://docs.lantronix.com/products/xpico-200/ig/>

[30] WiFi Alliance: *Wi-Fi CERTIFIED™ Interoperability Certificate WFA91445* [online]. Austin, TX: 2021, datum certifikace 6. 12. 2021 [cit. 10. 12. 2021]. Certification ID: WFA91445. Dostupné z URL: <https://api.cert.wi-fi.org/api/certificate/download/public?variantId=106471>

[31] Bluetooth SIG: *xPico 250, xPico 270, SGX 3100 IoT Gateway Declaration Details* [online]. Kirkland, WA: 2019, datum certifikace 29. 4. 2019 [cit. 10. 12. 2021]. Declaration ID: D044584. Dostupné z URL: <https://launchstudio.bluetooth.com/ListingDetails/84210>

[32] Analog Devices, Inc.: *ADuM160N/ADuM161N/ADuM162N/ADuM163N 3.0 kV RMS, 6-Channel Digital Isolators Datasheet* [online]. rev. A. Wilmington, MA: 2019, poslední aktualizace června 2019 [cit. 13. 12. 2021]. Dostupné z URL: <https://www.analog.com/media/en/technical-documentation/data-sheets/ADuM160N-161N-162N-163N.pdf>

[33] Trenz Electronic GmbH: *TE0720 TRM* [online]. 2021, poslední aktualizace 15. 11. 2011 [cit. 13. 10. 2011]. Dostupné z URL: <https://wiki.trenz-electronic.de/display/PD/TE0720+TRM>

[34] Trenz Electronic GmbH: *TE0703 CPLD - CC703S* [online]. 2019, poslední aktualizace 8. 11. 2019 [cit. 13. 10. 2011]. Dostupné z URL: <https://wiki.trenz-electronic.de/display/PD/TE0720+TRM>

[35] Xilinx, Inc.: *Zynq-7000 SoC Data Sheet: Overview* [online]. rev. v1.11.1. San Jose, CA: 2018, poslední aktualizace 2. 6. 2018 [cit. 13. 12. 2021]. Datasheet DS190. Dostupné z URL: <https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf>

[36] Xilinx, Inc.: *Zynq-7000 SoC Product Specification: Packaging and Pinout* [online]. rev. v1.9. San Jose, CA: 2021, poslední aktualizace

28. 6. 2021 [cit. 13. 12. 2021]. User Guide UG865. Dostupné z URL: <https://www.xilinx.com/support/documentation/user_guides/ug865-Zynq-7000-Pkg-Pinout.pdf>

[37] Motorola, Inc.: *MC54/74HC589 8-Bit Serial or Parallel-Input/Serial-Output Shift Register with 3-State Output* [online]. rev. 6. Chicago, IL: 1995, poslední aktualizace října 1995 [cit. 13. 12. 2021]. Dostupné z URL: <https://datasheet.datasheetarchive.com/originals/library/Datasheet-017/DSA00303462.pdf>

[38] Lattice Semiconductor: *ispMACH 4000V/B/C/Z Family* [online]. 2009, poslední aktualizace května 2009 [cit. 13. 10. 2011]. Datasheet DS1020_23.1. Dostupné z URL: <https://www.latticesemi.com/~/media/LatticeSemi/Documents/Solutions/Packaging%20Solutions/ispMACH4000VBCZ%20Family%20Data%20Sheet1020.pdf>

[39] Lattice Semiconductor: *Software Licensing - More Information on our Licensing* [online]. 2021 [cit. 13. 10. 2011]. Dostupné z URL: <https://www.latticesemi.com/Support/Licensing>

[40] Lattice Semiconductor: *Products - Development Software ispLEVER Classic 12-month Subscription License* [online]. 2021 [cit. 13. 10. 2011]. Dostupné z URL: <https://www.latticestore.com/products/tabid/417/searchid/1/searchvalue/lsc-sw-isplever/default.aspx>

[41] Lattice Semiconductor: *MachXO2 Family Data Sheet* [online]. rev. 3.3. 2017, poslední aktualizace března 2017 [cit. 13. 10. 2011]. Datasheet DS1035. Dostupné z URL: <https://www.latticesemi.com/~/media/LatticeSemi/Documents/DataSheets/MachXO23/MachXO2FamilyDataSheet.pdf>

[42] Trenz Electronic GmbH: *TEL0001 - LXO2000 2.5x6.15cm FPGA module with Lattice XO2-4000 Rev. 2 Resources* [online]. 2021 [cit. 13. 10. 2011]. Dostupné z URL: <https://shop.trenz-electronic.de/Download/?path=Trenz_Electronic/Modules_and_Module_Carriers/2.5x6.15/TEL0001/REV02>

[43] Microchip Technology Inc.: *AT24C08D I2C-Compatible (2-Wire) Serial EEPROM 8-Kbit (1,024 x 8)* [online]. Rev. A. Chandler, AZ: 2018, poslední aktualizace června 2018 [cit. 13. 12. 2021]. Datasheet DS20006022A. Dostupné z URL: <https://ww1.microchip.com/downloads/en/DeviceDoc/AT24C08D-I2C-Compatible-2-Wire-Serial-EEPROM-20006022A.pdf>

[44] Semiconductor Components Industries, LLC: *MC33269 800 mA Adjustable Output Low Dropout Voltage Regulator* [online]. Rev. 29. Phoenix, AZ: 2021,

poslední aktualizace října 2021 [cit. 13. 12. 2021]. Publication Order Number MC33269/D. Dostupné z URL: <https://www.onsemi.com/pdf/datasheet/mc33269-d.pdf>

[45] Intel Corporation: *Hexadecimal Object File Format Specification* [online]. Rev. A. Santa Clara, CA: 1998, poslední aktualizace 6. 1. 1998 [cit. 14. 12. 2021]. Dostupné z URL: <https://web.archive.org/web/20210303163219if_/http://grantronics.com.au/docs/intelhex.pdf>

[46] Ecma International: *The JSON Data Interchange Syntax* [online]. 2. vyd. Ženeva: 2017, poslední aktualizace prosince 2017 [cit. 23. 5. 2022]. Standard ECMA-404. Dostupné z URL: <http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>

[47] World Wide Web Consortium: *Extensible Markup Language (XML) 1.0* [online]. 5. vyd. Cambridge, MA: 2008, poslední aktualizace 26. 11. 2008 [cit. 23. 5. 2022]. Dostupné z URL: <https://www.w3.org/TR/2008/REC-xml-20081126/>

[48] Microsoft Corporation: *Button Types - Win32 apps* [online]. Redmond, WA: 2020, poslední aktualizace 21. 8. 2020 [cit. 23. 5. 2022]. Dostupné z URL: <https://docs.microsoft.com/en-us/windows/win32/controls/button-types-and-styles>