Projekt: UART – přijímací část

Cílem projektu je získat základní dovednosti potřebné pro návrh a implementaci číslicových obvodů. Naučit se tyto obvody správně popisovat v jazyce VHDL a získat zkušenosti s jejich simulací a syntézou s využitím profesionálních nástrojů.

Jako zvolený příklad komponenty, na které uvedené dovednosti získáte, nám poslouží komponenta pro příjem a vysílaní dat po asynchronní sériové lince (anglicky UART – Universal Asynchronous Receiver-Transmitter).

Pro jednoduchost budeme v projektu vytvářet pouze vybranou část UART řadiče, konkrétněji se zaměříme na přijímací část. Ta je zodpovědná za zpracování dat ze sériové linky a jejich rekonstrukci (deserializaci jednotlivých bitů). V porovnání s plnohodnotným řadičem UART budeme uvažovat i řadu dalších zjednodušení, aby celková složitost vypracování projektu nebyla velká.

Vypracování projektu je rozděleno na dvě části. V první části jde o návrh fungování obvodu na úrovni RTL a také logiky jeho řídícího automatu. Odevzdává se dokument s popisem vámi navrženého řešení. Druhá část je zaměřena na implementaci navrženého obvodu v jazyce VHDL a odladění správné funkcionality pomocí nástrojů pro simulaci a syntézu. Odevzdávají se zdrojové kódy implementace a doplněná zpráva s popisem fungování vytvořeného obvodu.

Po zkušenostech z minulých let ještě jedno důležité upozornění!

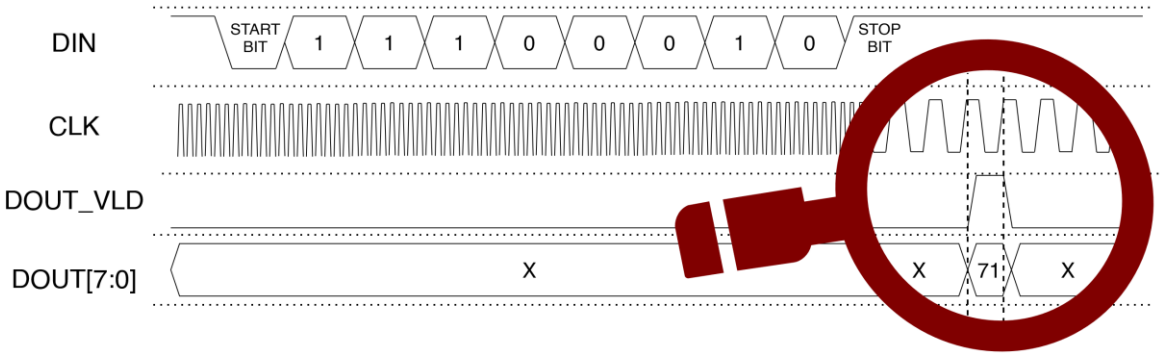
Podle Směrnice děkana FIT doplňující Studijní a zkušební řád VUT (Rozhodnutí děkana FIT č. 34/2010), K článku 11, odst. 4: *„Veškeré testy, projekty a další hodnocené úlohy vypracovává student samostatně, pokud projekt nebo úloha nebyly výslovně zadány pro stanovenou skupinu studentů.“*

V případě odhalení plagiátorství nebo nedovolené spolupráce na projektu nebo jeho částech, bude proto student odměněn neudělením zápočtu z předmětu INC (0 bodů za projekt). Dále může také dojít k předvolání před disciplinární komisi podle Disciplinárního řádu pro studenty Fakulty informačních technologií Vysokého učení technického v Brně.

První část projektu – Návrh obvodu

Navrhněte obvod pro příjem datových slov po asynchronní sériové lince (UART_RX).

- Vycházejte ze základních informací o fungování a zpracování asynchronní sériové komunikace uvedených v následující kapitole.
- Uvažujte vstupní tok dat v pevném formátu: jeden START bit, 8 bitů dat, jeden STOP bit, zasílaných rychlostí 9600 baudů za sekundu. Přijímací obvod bude pracovat na 16x vyšší frekvenci (signál CLK) ve srovnání s přenosovou rychlostí jednotlivých datových bitů. Vašim úkolem bude snímat datové bity uprostřed přenášeného intervalu (viz. obrázek 3).
- Obvod UART_RX bude přijímat jednotlivé bity na vstupním datovém portu DIN, provede jejich de-serializaci a výsledné 8-bitové slovo zapíše na datový port DOUT. Platnost datového slova na portu DOUT potvrďte nastavením příznaku DOUT_VLD na úroveň logické 1 po dobu jednoho taktu hodinového signálu CLK. Příklad časového diagramu ukazujícího očekávaný průběh signálů na vstupně/výstupních portech komponenty UART_RX je znázorněn na obrázku 1.



Obrázek 1. Příklad časového průběhu na vstupech a výstupech obvodu UART_RX

- Jednotlivé části datové cesty obvodu bude potřeba ovládat skrze konečný automat (*Finite State Machine*). Sestavte si nejprve graf přechodů tohoto automatu.
- Při návrhu nezapomeňte ošetřit asynchronní vstup do synchronní sítě obvodu UART_RX pro redukci možných metastabilních stavů.

Vytvořte technickou zprávu, která bude obsahovat:

- Jméno, příjmení a login.
- Schéma architektury navrženého obvodu UART_RX na úrovni RTL a její stručný popis.
- Nákres grafu přechodů konečného automatu a jeho stručný popis.

Ukázku formátu a obsahu výstupní zprávy naleznete v příloze. Rozsah vaší zprávy z první části projektu by neměl překročit dvě strany formátu A4. Zprávu odevzdejte ve formátu PDF jako soubor s názvem *zprava.pdf* skrze informační systém, termín pro první část projektu.

Před odevzdáním do informačního systému se prosím ujistěte, že finální podoba zprávy obsahuje všechny potřebné náležitosti. Zkontrolovanou verzi zprávy odevzdejte prostřednictvím informačního systému nejpozději do data uvedeného v informačním systému u termínu pro první část projektu. Pozdější odevzdání nebude bráno v úvahu.

Asynchronní sériová komunikace

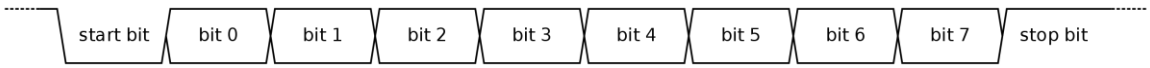
Asynchronní sériová komunikace se stala základním způsobem přenosu dat mezi počítači a periferními zařízeními. V současné době se používá zejména v oblasti vestavěných systémů. Pro přenos dat mezi dvěma uzly stačí při sériové komunikaci jeden datový vodič, po kterém jsou postupně zasílány jednotlivé datové bity. Asynchronnost komunikace pak znamená, že přenášené bity nejsou synchronizovány žádným dodatečným signálem jako je např. hodinový signál CLK. Přijímač je schopen rozpoznat příchozí bity a jejich synchronizaci na základě použitého zakódování na datovém vodiči.

Komunikační linka (vodič) je vždy před začátkem přenosu libovolného vícebitového slova (obvykle bajtu) nastavena na úroveň logické 1. Přenos vícebitového slova pak začíná tzv. START bitem s hodnotou logické 0. Odvysílání START bitu, tedy přechod datové linky z hodnoty logické 1 do 0, umožní přijímači spolehlivě identifikovat okamžik začátku přenosu.

Za START bitem jsou následně odvysílány jednotlivé bity datového slova od významově nejnižšího bitu (LSB) po významově nejvyšší bit (MSB). Za posledním bitem datového slova následuje jeden nebo více tzv. STOP bitů, které jsou vždy nastaveny na úroveň logické 1.

Za STOP bitem může začít přenos dalšího datového slova, začíná se opět START bitem. Všimněte si, že STOP bit předchozího datového slova v kombinaci se START bitem dalšího slova umožňují spolehlivou detekci začátku nového přenosu (přechod z log. 1 do 0).

Příklad přenosu 8-bitového datového slova s jedním STOP bitem je znázorněn na obrázku 1.

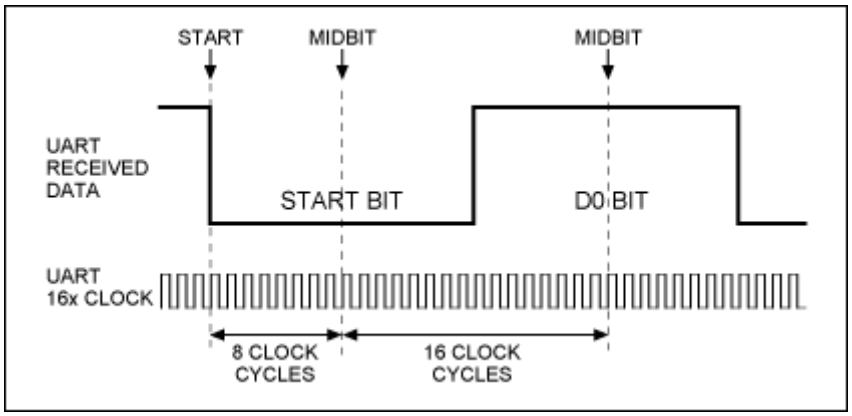


Obrázek 2. Příklad sériového přenosu 8-bitového datového slova s jedním STOP bitem

Pro spolehlivé rozpoznání jednotlivých bitů přenášeného datového slova na straně přijímače je potřeba nejen identifikovat začátek přenosu (přechod z logické 1 do 0), ale také vědět na jaké rychlosti komunikace probíhá. Vysílač i přijímač se proto musí nejprve nastavit na stejnou přenosovou rychlost.

Přenosová rychlost se udává v počtu přenesených baudů za sekundu, přičemž jeden baud odpovídá v tomto případě jednomu bitu. Základní a také nejčastěji používanou přenosovou rychlostí je rychlost 9600 baudů za sekundu. Pokud uvažujeme přenos 8-bitových datových slov ohraničených jedním START bitem a alespoň jedním STOP bitem (celkem 10 bitů), potom jsme schopni na rychlosti 9600 baudů přenášet až 960 bajtů za sekundu (9600/10).

Aby přijímač spolehlivě identifikoval hodnoty (logické úrovně) přenášených datových bitů je navíc doporučeno, aby tento obvod pracoval na 16x větší frekvenci, než je vybraná přenosová rychlost. Každý datový bit by následně měl být snímán uprostřed časového intervalu pro přenos bitů, jak je naznačeno na obrázku 2. Specifikace doporučuje při 16x větší vzorkovací frekvenci snímat logickou úroveň vstupních dat v 7., 8. a 9. hodinovém cyklu a jako výsledný bit použít majoritu z těchto tří hodnot. Pro jednoduchost se však zde spokojme s hodnotou nasnímanou na konci 8. hodinového cyklu označenou jako MIDBIT.



Obrázek 3. Příklad vzorkování datového bitu uprostřed[1]

[1] Převzato z https://electronics.stackexchange.com/questions/207870/uart-receiver-sampling-rate

Druhá část projektu – Implementace a ladění

1. Pro vypracování druhé části projektu budete potřebovat nástroje pro simulaci a syntézu číslicových obvodů popsaných v jazyce VHDL. Vyžadovány jsou konkrétně nástroje ModelSim a Xilinx ISE. Více informací o tom, jak nástroje získat a používat je uvedeno v následující kapitole.
2. Z informačního systému si stáhněte archív zdrojových souborů *projekt.zip* a seznamte se s jeho obsahem. V archívu naleznete hlavně tyto tři VHDL zdrojové soubory:
   - *uart.vhd* – zdrojový soubor v jazyce VHDL obsahující definici rozhraní (entity) komponenty UART_RX a prázdnou architekturou na doplnění vaši implementace,
   - *uart_fsm.vhd* – zdrojový soubor v jazyce VHDL, který bude sloužit pro popis konečného automatu řídicího ostatní komponenty vašeho obvodu UART_RX,
   - *uart_tb.vhd* – zdrojový soubor v jazyce VHDL, který reprezentuje ukázkové zapojení pro testování (tzv. testbench) základní funkcionality vámi navržené a implementované komponenty UART_RX,
3. Navržený obvod z první části projektu implementujte v jazyce VHDL a uložte do předpřipraveného souboru *uart.vhd*. Kód odpovídající konečnému automatu vložte pro přehlednost do samostatného souboru *uart_fsm.vhd*.
4. Proveďte simulaci vašeho VHDL kódu pomocí programu ModelSim a ověřte tak jeho správnou funkčnost. Postupně opravujte nalezené funkční chyby a simulaci opakujte.
5. Vyzkoušejte si syntézu vašeho VHDL kódu pomocí programu Xilinx ISE a ověřte tak jeho připravenost pro cílovou technologii FPGA. Odstraňte a vhodně nahraďte v kódu odhalené problematické konstrukce. Vraťte se pak na předchozí krok 4 (simulace funkce).
6. Doplňte technickou zprávu z první části projektu o snímek obrazovky (Print Screen) programu ModelSim s ukázkou časových průběhů simulací zachycujících přenos jednoho datového slova. Jestliže jste se při implementaci odchýlili od vašeho původního návrhu z první části projektu, upravte také ilustrace a popis architektury obvodu a grafu přechodů automatu. Finální verze zprávy by měla přesně souhlasit s vytvořenou implementací. Zachovejte formát a obsah zprávy podle ukázky v příloze.
7. Výstupy druhé části projektu budou tvořit:
   - implementované zdrojové soubory v jazyce VHDL (*uart.vhd* a *uart_fsm.vhd*),
   - soubor *zprava.pdf* s výstupní zprávou (ve formátu PDF).

   Všechny tři uvedené soubory zabalte do ZIP archivu s názvem ***<login>.zip***. Archiv odevzdejte skrze informační systém, termín pro druhou část projektu.

Před odevzdáním vašeho archivu do informačního systému si ho prosím otestujte skrze sadu testovacích skriptů připravených v souboru *student_test.zip*. Podrobný návod na otestování naleznete v přiloženém README souboru.

Otestovaný archiv odevzdejte prostřednictvím informačního systému nejpozději do data uvedeného v informačním systému u termínu pro druhou část projektu. Pozdější odevzdání nebude bráno v úvahu.

Simulace a syntéza obvodu

Pro účely vypracování projektu jsme pro vás připravili obraz disku virtuálního stroje, kde jsou nainstalovány potřebné nástroje. Obraz obsahuje zejména instalace ModelSim a Xilinx ISE.

Obraz disku virtuálního stroje si stáhněte z privátních stránek FITkitu. Konkrétně jde o archiv s názvem *fitkit-vbox-202103.7z*. VMDK soubor z archivu vybalte pomocí aplikace 7z.

Pro spuštění virtuálního stroje si nainstalujte volně dostupný program VirtualBox. Při tvorbě VM použijte nastavení Windows XP (32-bit), alespoň 2048 MB paměti, stažený VMDK disk.

Spuštění simulace

ModelSim je komerční nástroj, který se pravidelně dotazuje licenčního serveru a kontroluje platnost zakoupené licence. Tento licenční server je umístěn v doméně fit.vutbr.cz a vyžaduje připojení přímo ze sítě fakulty nebo vzdáleně skrze VPN.

Postupujte prosím dle návodu uvedeného na následujících stránkách a vždy před spuštěním virtuálního stroje a nástroje ModelSim mimo fakultu si VPN spojení aktivujte.

Samotné spuštění nástroje ModelSim (uvnitř běžícího virtuálního stroje) lze provést buď skrze ikonu na pracovní ploše, nabídku Start nebo z příkazové řádky pomocí příkazu *vsim*.

Po zobrazení grafického uživatelského prostředí je možné začít obvod simulovat. Je potřeba zkompilovat zdrojové soubory skrze příkaz *vcom*, přepnout ModelSim do režimu simulace skrze příkaz *vsim*, přidat do okna s označením *Wave* sledované signály, spustit simulaci skrze příkaz *run*. Abyste tuto sadu příkazů nemuseli zadávat při každé simulaci, nabízí ModelSim možnost vytváření a spouštění skriptů v jazyce TCL. Uvedené příkazy se jednoduše vloží do skriptu a tento skript se spustí skrze příkaz *do* ve formátu *do <název skriptu>*.

Samotné spuštění nástroje ModelSim lze navíc skombinovat se spuštěním tohoto pomocného TCL skriptu skrze následující příkaz z příkazové řádky: *vsim -do <název skriptu>*.

Pro pohodlnou práci jsme pro vás připravili ukázkový TCL skript *uart.fdo* určený přímo pro zadaný projekt. Spustíte jej voláním příkazu *do* jak bylo popsáno. Konkrétně tedy *do uart.fdo* z grafického prostředí programu ModelSim nebo *vsim -do uart.fdo* z příkazové řádky.

Spuštění syntézy

Nástroj Xilinx ISE pracuje s lokálním licenčním souborem nahraným přímo na připraveném obrazu virtuálního stroje. Není tedy potřeba spojení s licenčním serverem ani fakultní VPN.

Samotné spuštění syntézy nástrojem ISE funguje na úrovni projektů. Pro každý projekt je možné definovat celou radu parametrů, zejména seznam zdrojových souborů, hlavní (top) syntetizovanou entitu, cílový FPGA čip, nebo parametry optimalizace. Projekt a všechny tyto nastavení jsou pro vás připraveny v souborech *uart.{lso|prj|xst}*, není potřeba je upravovat. Spuštění syntézy proveďte z příkazové řádky voláním: *xst -intstyle ise -ifn uart.xst*.

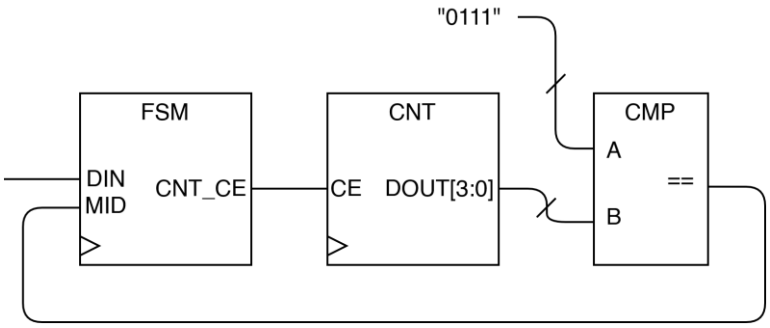
Průběh a výsledek syntézy bude zapsán přímo na příkazovém řádku a také v textovém soubor *uart.srp*. Zkontrolujte podle výpisu správnost syntézy vašeho obvodu a případně odstraňte nástrojem hlášené chyby nebo varování. Některé VHDL konstrukce podporované resp. fungující v simulaci mohou být nepoužitelné při syntéze do reálného FPGA hradlového pole. Nástroj vás kromě syntaktických a sémantických chyb tedy upozorní také na tyto problémy.

Příloha: Výstupní zpráva (Ukázka)

Jméno:
Login:

Architektura navrženého obvodu (na úrovni RTL)

Schéma obvodu



Poznámky:

- Pro přehlednost CLK a RST signály ve schématech uvádíme, ale nemusíme zapojovat.
- Několik jednobitových D-KO můžete pro přehlednost spojit do jednoho vícebitového registru, pokud tedy sdílí všechny kontrolní signály jako CLK, RST, nebo CE.
- Jednotlivé vodiče můžete spojovat do vícebitových sběrnic.

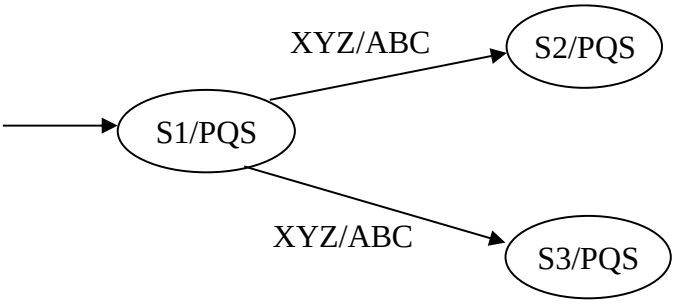
Popis funkce

Stručný slovní popis struktury a funkce obvodu (max. polovina strany A4).

Návrh automatu (Finite State Machine)

Schéma automatu

Legenda:
- Stavy automatu: S1, S2, S3
- Vstupní signály: X, Y, Z
- Mealyho výstupy: A, B, C
- Moorovy výstupy: P, Q, S



Poznámky:
- Použijte vhodné názvy pro stavy, vstupní a výstupní signály tak, aby byl snáze pochopitelný jejich význam.
- Za vstupně/výstupní signály XYZ, ABC a PQS dosaďte do grafu přímo hodnoty 0, 1 nebo X (don't care).
- Signály CLK a RST neuvádíme mezi vstupy automatu ani je nekreslíme do schémat.
- Automat může vhodně kombinovat jak Mealyho, tak Moorovi výstupy.
- Pokud je vstupním signálem vektor bitů, můžete s ním na hranách pracovat jako s vektorem.
- Připomeňte si konvence pro kreslení grafu automatu probírány na přednáškách.
- Nezapomeňte na označení počátečního stavu automatu.

Popis funkce

Stručný slovní popis funkce automatu (max. polovina strany A4).

Snímek obrazovky ze simulací

Zde prosím vložte obrázek (snímek obrazovky) z nástroje ModelSim, který demonstruje funkčnost vašeho obvodu na úrovni simulací. Zachyťte prosím přenos alespoň jednoho datového slova v okně Wave. Pro přehlednost můžete obrázek orientovat např. na šířku stránky A4.