Projekt: UART – přijímací část

1. Úvod

Cílem projektu je osvojit si základní dovednosti při návrhu a implementaci číslicových obvodů. Naučit se tyto obvody popisovat v jazyce VHDL a získat zkušenosti s jejich simulací s využitím nástroje Modelsim.

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

Pro jednoduchost budeme implementovat pouze vybranou část tohoto řadiče, konkrétněji se zaměříme na přijímací část zodpovědnou za zpracování dat ze sériové linky a jejich rekonstrukci (paralelizaci). Oproti plnohodnotnému řadiči UART budeme uvažovat i řadu dalších zjednodušení, aby celková složitost projektu nepřesáhla únosnou mez.

2. 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 nám vystačí jeden datový vodič, po kterém jsou postupně zasílány jednotlivé datové bity od významově nejnižšího bitu (LSB) po významově nejvyšší bit (MSB).

Asynchronní sériová komunikace znamená, že přenášené bity nejsou synchronizovány žádným dodatečným signálem jako je např. CLK signál. Přijímač je schopen rozpoznat příchozí bity na základě použitého zakódování dat.

Přenosové linka je vždy před začátkem přenosu každého vícebitového slova (obvykle bajtu) nastavena na úroveň log. 1.

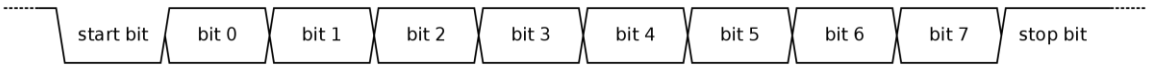
Přenos vícebitového slova začíná tzv. START bitem nastaveným na úroveň log. 0. Odvysílání tohoto START bitu (tj. přechod z log. 1 do log. 0) umožní přijímači spolehlivě identifikovat začátek přenosu.

Za START bitem jsou následně odvysílaný jednotlivé bity datového slova od významově nejnižšího bitu (LSB) po významově nejvyšší bit (MSB).

Za posledním bitem datového slova (LSB) následuje jeden nebo více tzv. STOP bitů, které jsou vždy nastaveny na úroveň log. 1.

Za STOP bitem může začít přenos dalšího datového slova počínaje START bitem. Prosím všimněte si, že STOP bit předchozího datového slova v kombinaci se START bitem umožňují spolehlivou detekci následujícího datového slova (přechod z log. 1 do log. 0).

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

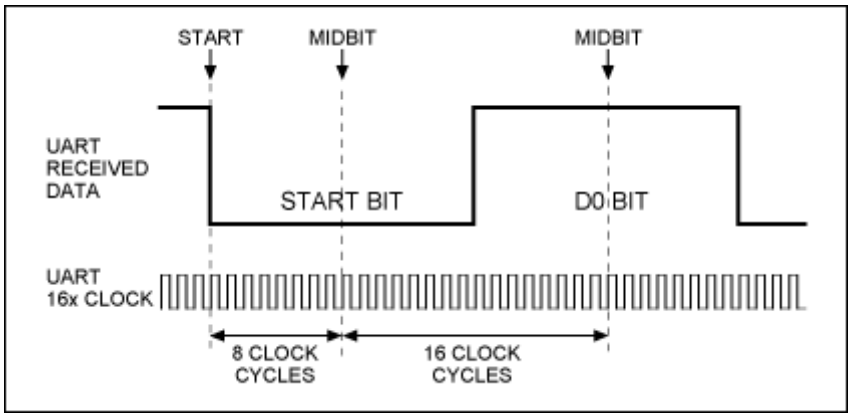


Obrázek 1. Příklad sériového přenosu 8-bitového datového slova s jedním STOP bitem

Pro spolehlivou detekci přenášeného slova na straně přijímače je potřeba nejen identifikovat začátek přenosu (skrze přechod z log. 1 do log. 0), ale také vědět na jaké rychlosti jsou data přenášena. Vysílač i přijímač se proto musí nejprve nastavit/nakonfigurovat 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 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 rychlosti, než je vybraná přenosová rychlost. Datový bit by následně měl být snímán uprostřed intervalu pro přenos jednoho bitu, jak je naznačeno na obrázku 2[1].



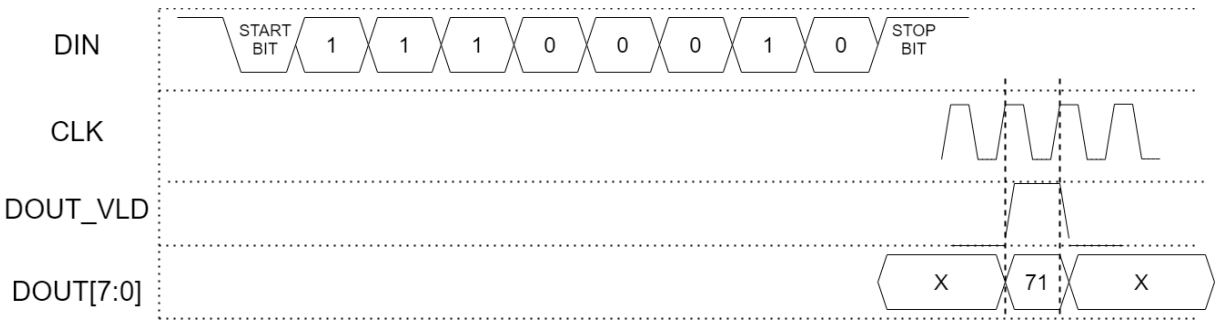
Obrázek 2. Příklad vzorkování datového bitu uprostřed[2]

[1] Specifikace doporučuje snímat log. úroveň vstupních dat v 7., 8. a 9. hodinovém cyklu a jako výsledný bit použít majoritu z těchto tří hodnot. Pro jednoduchost se však spokojme s hodnotou nasnímanou na konci 8. hodinového cyklu

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

3. Postup práce

1. Pro vypracování projektu budete potřebovat nástroj pro simulaci číslicových obvodů popsaných v jazyce VHDL. Preferován je nástroj Modelsim od společnosti Mentor Graphics, který bude použit i pro ohodnocení tohoto projektu. Více informací o tom, jak tento nástroj získat a používat je uvedeno v kapitole 4.
2. Z prostředí MS Teams (kanál Projekt, záložka Soubory) si stáhněte archív zdrojových souborů *projekt.zip* a seznamte se s jeho obsahem. V archívu naleznete čtyři soubory:
    - *uart.vhd* – zdrojový soubor v jazyce VHDL s definicí rozhraní komponenty UART_RX a prázdnou architekturou, kterou budete doplňovat.
    - *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 ukázkový Testbench soubor pro otestování základní funkcionality vámi navrhované komponenty UART_RX.
    - *uart.fdo* – pomocný skript v jazyce TCL, který slouží pro spuštění simulace vámi navrhovaného obvodu UART_RX v prostředí Modelsim.
3. Navrhněte obvod pro příjem datových slov po asynchronní sériové lince (UART).
    - Vycházejte ze základních informací o zpracování asynchronní sériové komunikace uvedených v kapitole 2.
    - Uvažujte vstupní tok dat ve formátu: START bit, 8 bitů dat, 1x 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 2).
    - Obvod bude přijímat jednotlivé bity na vstupním portu DIN, provede jejich de-serializaci a výsledné 8-bitové slovo zapíše na port DOUT. Platnost datového slova na portu DOUT potvrďte nastavením signálu DOUT_VLD na úroveň log. 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 3.



Obrázek 3. Příklad časového průběhu na vstupech a výstupech obvodu UART_RX

- Jednotlivé části 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.

4. Navržený obvod implementujte v jazyce VHDL a uložte do předpřipraveného souboru *uart.vhd*. Kód odpovídající konečnému automatu vložte pro přehlednost do samostatného souboru *uart_fsm.vhd*.
5. Proveďte simulaci VHDL kódu pomocí programu Modelsim a ověřte jeho správnou funkčnost.
6. Vytvořte technickou zprávu, která bude obsahovat:
   - Jméno, příjmení a login.
   - Architekturu navrženého obvodu UART_RX na úrovni RTL a její stručný popis.
   - Graf přechodu konečného automatu a jeho stručný popis.
   - Snímek obrazovky s ukázkou časových průběhů simulací zachycujících přenos jednoho datového slova.

   Ukázku výstupní zprávy naleznete v příloze č. 1. Rozsah zprávy by neměl překročit tři strany formátu A4.

7. Výstupy projektu budou tvořit:
   - Zdrojové soubory v jazyce VHDL (*uart.vhd* a *uart_fsm.vhd*).
   - Soubor *zprava.pdf* s výstupní zprávou (ve formátu PDF)

   Všechny tři soubory zabalte do archívu s názvem <*login*>.zip.

   Před odevzdáním tohoto archivu do informačního systému si prosím tento archiv otestujte skrze sadu testovacích skriptů dostupných v informačním systému 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 označeného jako Projekt. Pozdější odevzdání projektu nebude bráno v úvahu.

   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, bude proto student odměněn neudělením zápočtu z předmětu INC (0 bodů za projekt). Případně i předvoláním před disciplinární komisi podle Disciplinárního řádu pro studenty Fakulty informačních technologií Vysokého učení technického v Brně.

4. Simulace obvodu s využitím nástroje Modelsim

Získání nástroje Modelsim

Pro účely vypracování projektů do kurzů zaměřených na návrh číslicových obvodů jsme pro vás připravili obraz virtuálního stroje, kde jsou nainstalovány všechny potřebné nástroje, včetně Modelsimu, který budete potřebovat v rámci tohoto kurzu.

Obraz tohoto virtuálního stroje si stáhněte z privátních stránek FITkitu (soubor fitkit-vbox-202103.7z).

Soubor si rozbalte/dekomprimujte pomocí aplikace 7z.

Pro spuštění tohoto stroje si nainstalujte volně dostupný program VirtualBox.

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 skrze VPN.

Postupujte prosím dle návodu uvedeného na následujících stránkách a vždy před spuštěním virtuálního stroje a nástroje Modelsim 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 skrze příkaz:

```
vsim
```

Po zobrazení grafického uživatelského prostředí Modelsimu je možné začít obvod simulovat. Je k tomu 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 spouštět 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 <název skriptu>
```

Samotné spuštění Modelsimu 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 určený přímo pro tento projekt, který se jmenuje `uart.fdo` a spustíte jej jedním z následujících způsobů:

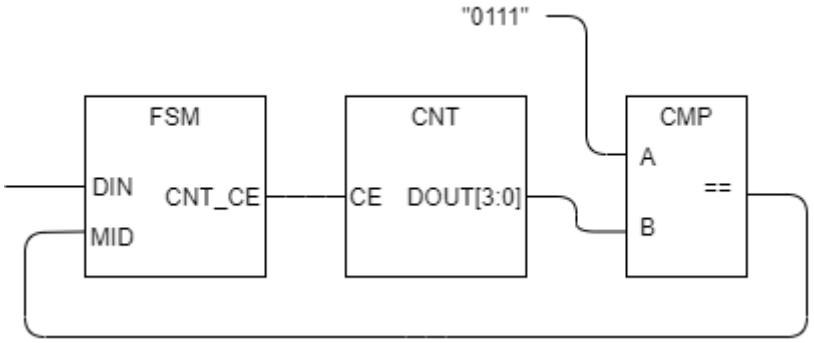
1. `do uart.fdo` (z prostředí Modelsimu)
2. `vsim -do uart.fdo` (z příkazové řádky)

Příloha č. 1: Ukázka výstupní zprávy

Jméno:
Login:

Architektura navrženého obvodu (na úrovni RTL)

Schéma obvodu



Poznámky:

- Pro přehlednost neuvádějte do schématu kontrolní signály typu CLK a RESET.

Popis funkce

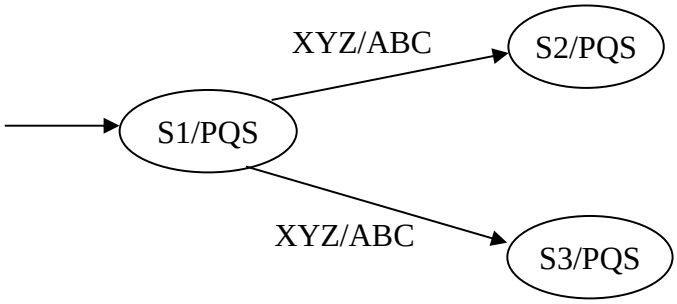
Stručný slovní popis struktury a funkce obvodu (max. polovina strany A4)

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 signály a výstupní signály, 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).

Popis funkce

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

Snímek obrazovky ze simulací

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