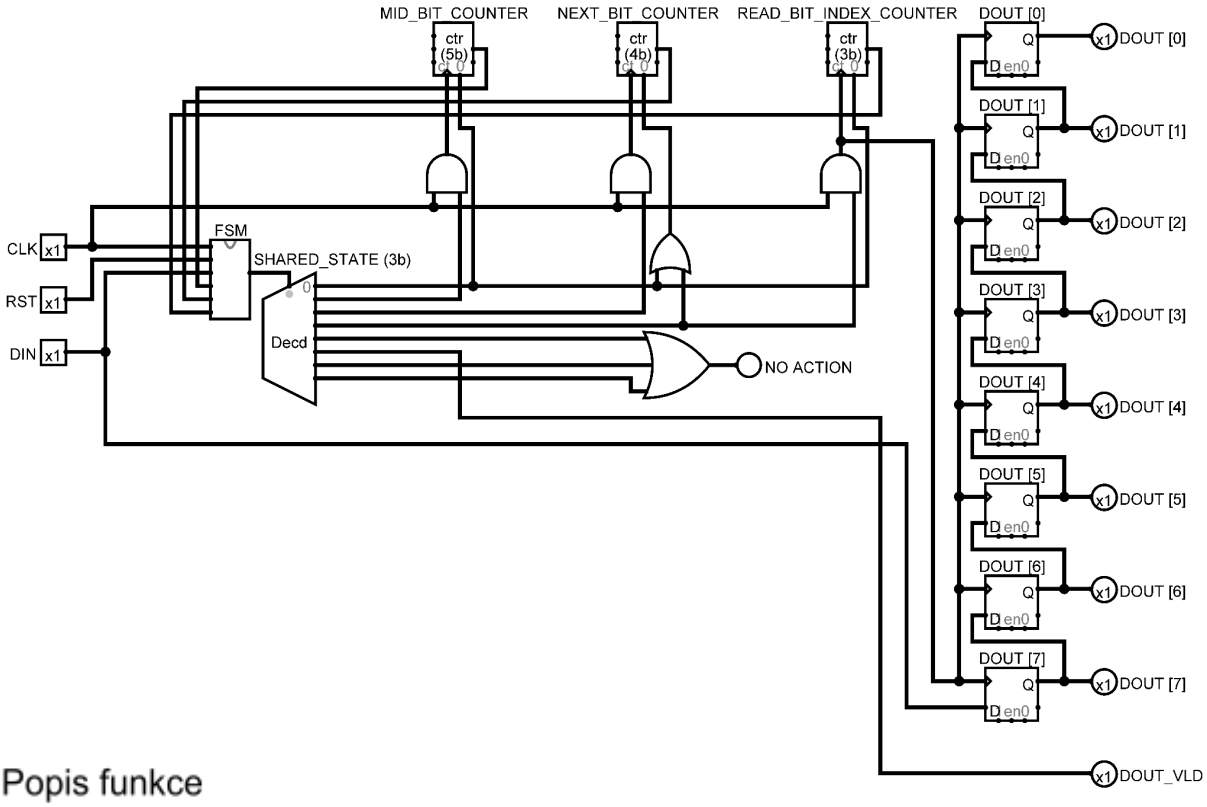
Výstupní zpráva

Jméno: Janšta Jakub
Login: xjanst02

Architektura navrženého obvodu (na úrovni RTL)

Schéma obvodu

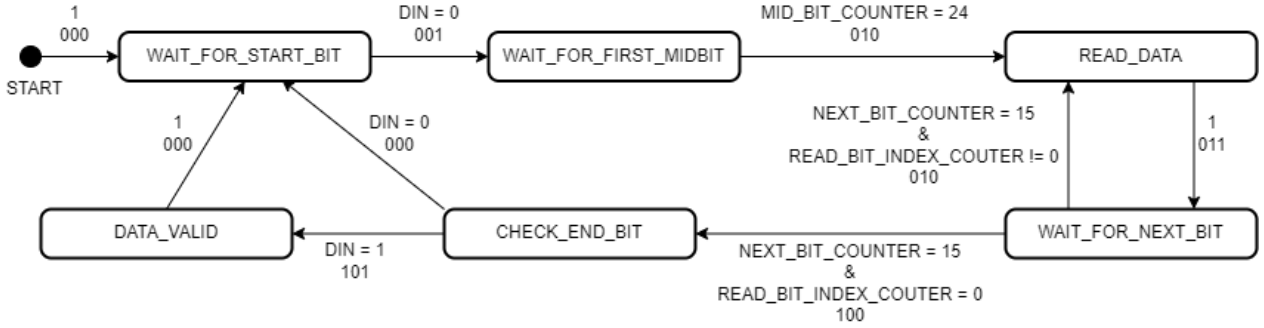


Popis funkce

Obvod realizuje akce podle výstupu FSM. FSM má jako výstup 3 bitové číslo, reprezentující stav, který je pomocí dekodéru převeden na jednotlivé vodiče, které povolují jednotlivé akce. **Vodič 0** pouze resetuje všechny čítače do nulového stavu (registr není nutné nulovat, protože se jeho hodnota přepíše). **Vodič 1** povoluje čítači MID_BIT_COUNTER, aby se inkremental každý takt hodin CLK. Tento čítač slouží jako zpětná vazba pro FSM pro přechod do dalšího stavu. Akce **vodiče 2** je funkčně totožná s vodičem 1 s rozdílem čítače, kdy je inkrementován NEXT_BIT_COUNTER. Tyto dva čítače by šly nahradit jedním, pokud by byla důležitá úspora součástek, ale z důvodu čitelnosti jsem se rozhodl pro dva explicitní čítače.
Vodič 3 inkrementuje čítač READ_BIT_INDEX_COUNTER každý takt, ale zároveň povoluje vstup DIN do registu DOUT a resetuje čítač NEXT_BIT_COUNTER, aby se vzorkovací bod neposouval v čase. **Vodič 4** je bez funkce (vysvětlení v další části). A nakonec **vodič 5** je napojen přímo na DOUT_VLD, který značí úspěšný přenos (logika je celá v FSM).
Vodiče 6 a 7 jsou navíc, kvůli rozsahu reprezentace čísel na třech bitech a jsou bez funkce.

Návrh automatu (Finite State Machine)

Schéma automatu



Mealyho automat tak, jak je implementován v uart_rx_fsm.vhd

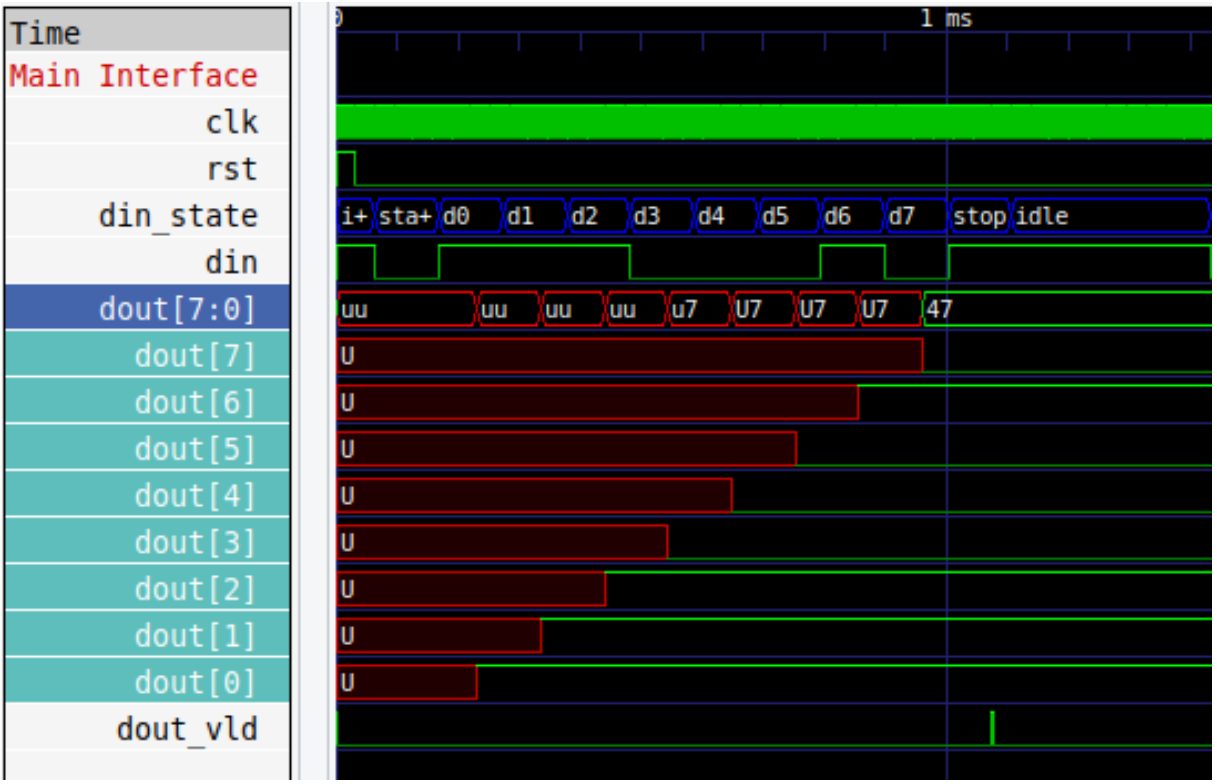
Pozn.: Pokud není splněna podmínka pro změnu stavu, tak automat implicitně zůstává v současném stavu.

Popis funkce

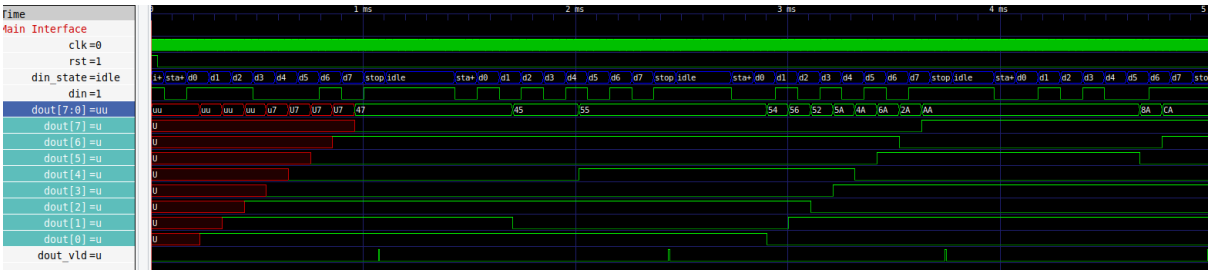
Kvůli časování bylo pro mne jednodušší v uart_rx_fsm.vhd implementovat Mealyho automat a proto je zde jako schéma použit. Automat se skládá z 6 stavů, podle jejichž výstupů na přechodech samotný obvod vykonává obsluhu signálů a zpětnou vazbu. Automat zahájí svou činnost ve stavu **WAIT_FOR_START_BIT**, kdy čeká na změnu signálu DIN na 0, což znamená začátek přenosu. Stav **WAIT_FOR_START_BIT** je současně i reset stavem, který se nastaví při 1 na RST. Hned po 0 na DIN se stav přepne **do WAIT_FOR_FIRST_MIDBIT**, kdy se čeká 24 taktů na střed prvního datového bitu (16 taktů na první datový bit + 8 taktů na jeho střed). Automat se rozhoduje podle čítače **MID_BIT_COUNTER**, který se v tomto stavu inkrementuje a slouží tak jako zpětná vazba. Po dosažení středu prvního datového bitu se přejde do stavu **READ_DATA**, kdy se přečte hodnota DIN, uloží se do výstupního registru a inkrementuje čítač **READ_BIT_INDEX_COUNTER**, který slouží k tomu, aby automat věděl, který datový bit byl právě načten. Hned poté se přejde do stavu **WAIT_FOR_NEXT_BIT**, který pomocí čítače **NEXT_BIT_COUNTER** čeká 16 taktů na další midbit. Po dosažení midbitu se přesune **do CHECK_END_BIT**, pokud se **READ_BIT_INDEX_COUNTER** rovná 0 (counter overflow), což znamená, že bylo načteno právě 8 bitů. V jiném případě se vrací zpět do **READ_DATA**. Stav **CHECK_END_BIT** slouží pouze k čitelnosti kódu a diagramu a na jeho výstupu nezáleží žádná operace. Šel by tedy implementovat jako součást stavu **WAIT_FOR_NEXT_BIT** za cenu složitějších podmínek a menší čitelnosti. Automat se v tomto stavu rozhoduje podle hodnoty DIN pro následující stav. V případě 0, kdy je datové slovo nevalidní se přesouvá zpět na začátek do stavu **WAIT_FOR_START_BIT** a v případě 1 do **DATA_VALID**, kdy je **DOUT_VLD** nastaveno na 1 a přenos byl úspěšný. Ze stavu DATA_VALID se ihned (po 1 taktu) automat přesune do **WAIT_FOR_START_BIT** a může nastat další přenos.

Výstupy ve všech stavech korespondují s číslem daného stavu v tom pořadí, jak byly popsány, od 0 do 5 jako 3 bitové binární číslo. Čísla 6 (110b) a 7 (111b) nejsou použita a jsou nevalidními stavy, které by neměly nastat (není pro ně žádná akce a obvod by mohl uvíznout v nekonečné smyčce).

Snímky obrazovky ze simulací



1) Snímek přenosu prvního datového slova 0x47 (01000111b)



2) Snímek přenosu všech čtyř testovacích datových slov (0x47; 0x55; 0xAA; 0xCA)