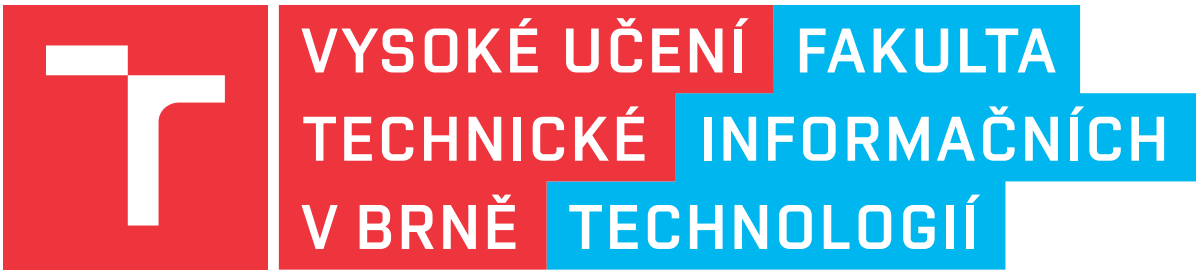


Dokumentácia k projektu IVH

8. mája 2023

Peter Ďurica

Obsah

1 Odôvodnenie konštantných hodnôt v čítačoch

V implementácii som použil 6 rôznych druhov čítačov. Boli to:

- Čítač na zmenu vykresľovanej pozície v displeji,
- Čítač na posun obrázka,
- Čítač na počítanie pohybov do strany,
- Čítač na počítanie pohybov hore,
- Čítač na načítanie animácie,
- Čítač na zobrazenie animácie.

1.1 Čítač na zmenu vykresľovanej pozície v displeji

Tento čítač slúži na zaistenie ilúzie obrázka na obrazovke, keďže vie displej naraz vykresľovať len jeden stĺpec. Preto treba tieto stĺpce vykresľovať dostatočne rýchlo, aby to vyzeralo že je na obrazovke celistvý obrázok. Keď čítač dopočíta svoju hodnotu, prepne sa pozícia vykresľovaného stĺpcu na displeji.

Pre čítač sa zvolila hodnota podľa frekvencie procesora, počtu stĺpcov na displeji a rýchlosti vykresľovania. Túto rýchlosť som si zvolil na 400 prejdení displeja za sekundu. Displej má 16 stĺpcov a frekvencia fitkitu je 25 Mhz. Výpočet hodnoty v čítači je teda:

$$\frac{25\,000\,000}{400 \cdot 16} \doteq 3096.$$

1.2 Čítač na posun obrázka

Hodnota v tomto čítači značí po koľkých cykloch na procesore sa posunie obrázok na displeji. Chcel som pozíciu obrázku meniť 6 krát za sekundu, aby zbytočne dlho netrval presun doprava a doľava. Preto sa hodnota v čítači počíta:

$$\frac{25\,000\,000}{6} \doteq 4\,166\,667.$$

1.3 Čítač na počítanie pohybov do strany

Tento čítač označuje koľko krokov spraví obrázok do strany, pred tým ako zmení smer. Inkrementuje sa len vtedy, ak predchádzajúci čítač na posun dopočíta svoju hodnotu. Aby sa obrázok posunul 3 krát do strany potrebuje 3 krát prejsť cez celú obrazovku. Teda:

$$3 \cdot 16 = 48.$$

1.4 Čítač na počítanie pohybov hore

Funguje na rovnakom princípe ako predchádzajúci čítač, ale počíta len počet riadkov na displeji. Plus sa k týmto riadkom pripočítava aj jednotka, aby bolo na jeden snímok vidieť aj prázdnu obrazovku. Takže tento čítač počíta do hodnoty 9.

1.5 Čítač na načítanie animácie

Animácia sa vykresľuje po stĺpcoch zľava doprava. Tento čítač počíta koľko stĺpcov sa už vykreslilo, aby sa nevykresľovalo mimo obrazovku. Preto hodnota v tomto čítači je 16.

1.6 Čítač na zobrazenie animácie

Animáciu som chcel mať zobrazenú na displeji sekundu a preto je v tomto čítači nastavená hodnota 25 000 000. Táto hodnota zodpovedá frekvencii fitkitu a teda kým sa napočíta tak prejde presne jedna sekunda.

2 Ostatné komponenty

2.1 FSM

V mojej implementácii má FSM 6 stavov. Program rotuje medzi týmito stavmi až do vypnutia. Operácie vrámci stavov sa vykonávajú v stavovom ovládači. Stavy v FSM sú:

1. RESET – Ak je stlačený reset alebo začínajuci stav cyklu, presun prebieha ak RESET = 0,
2. LOAD IMAGE – Načítanie obrázku z pamäte, presun do ďalšieho stavu prebehne po načítaní obrázka,
3. RIGHT – Presun obrázku doprava 3 krát, presun do ďalšieho stavu prebehne po dopočítaní čítača pohybu,
4. LEFT – Presun obrázku dolava 3 krát, presun do ďalšieho stavu prebehne po dopočítaní čítača pohybu,
5. TOP – Odchod obrázku hore, presun do ďalšieho stavu prebehne po dopočítaní čítača pohybu nahor
6. ANIMATION – Načítacia animácia, v rámci tohto stavu sa aj generuje. Po dopočítaní čítača sa vracia na RESET.

2.2 Pamäť

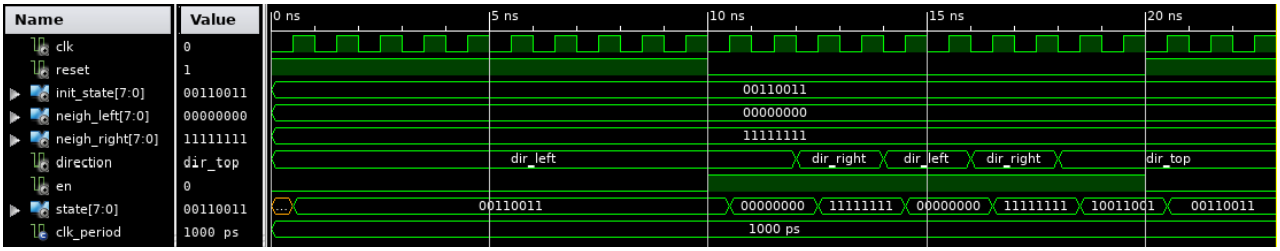
Pämať sa skladá zo 64 polí hodnôt. Každé pole sa skladá z vektoru 8 hodnôt. Teda každé pole znázorňuje stĺpec na displeji. Pamäť má 64 polí kvôli tomu, že obsahuje 4 obrázky a každý obrázok je cez celý displej, teda 16 stĺpcov. Komponent má jeden vstupný a jeden výstupný signál. Vstupný signál je integer a znázorňuje polohu stĺpca v poli. Výstupný je vektor 8 hodnôt a znázorňuje dáta v poli na ktoré ukazuje vstupný signál.

2.3 Stĺpec

Stĺpec znázorňuje komponent stĺpca na displeji. Pre každý stĺpec sa tento komponent generuje samostatne. Každý stĺpec si uchováva hodnoty stĺpcov vedľa seba. Hodnoty stĺpcov sa získavajú z vektora obrazovky pomocou funkcie `GETCOLUMN` z prvej podúlohy. Podľa smeru `DIRECTION_T` sa následne tieto hodnoty z vedľajších stĺpcov presúvajú do hlavného stĺpca.

3 Simulácia stĺpca

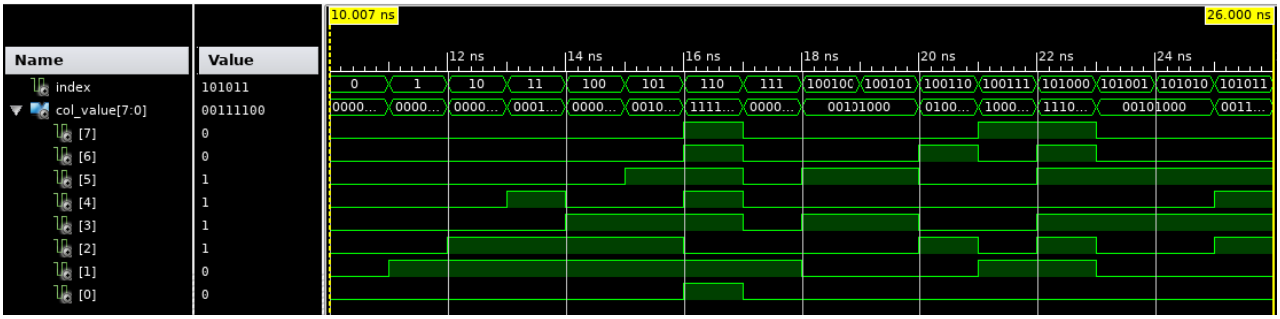
V obrázku 1 je vidieť ako sa mení hodnota vektore `STATE` podľa vstupných vektorov komponentu. Pokiaľ je `RESET` na jednotke, tak má hodnotu `INIT_STATE`. Následne ak je `EN` = 1, tak sa hodnota mení podľa smeru `DIRECTION`. Naberá vždy hodnoty susedných stĺpcov. Ak je nastavený smer hore, tak sa vrchná hodnota prehodí na spodok stĺpca.



Obr. 1: Simulácia stĺpca

4 Simulácia pamäte

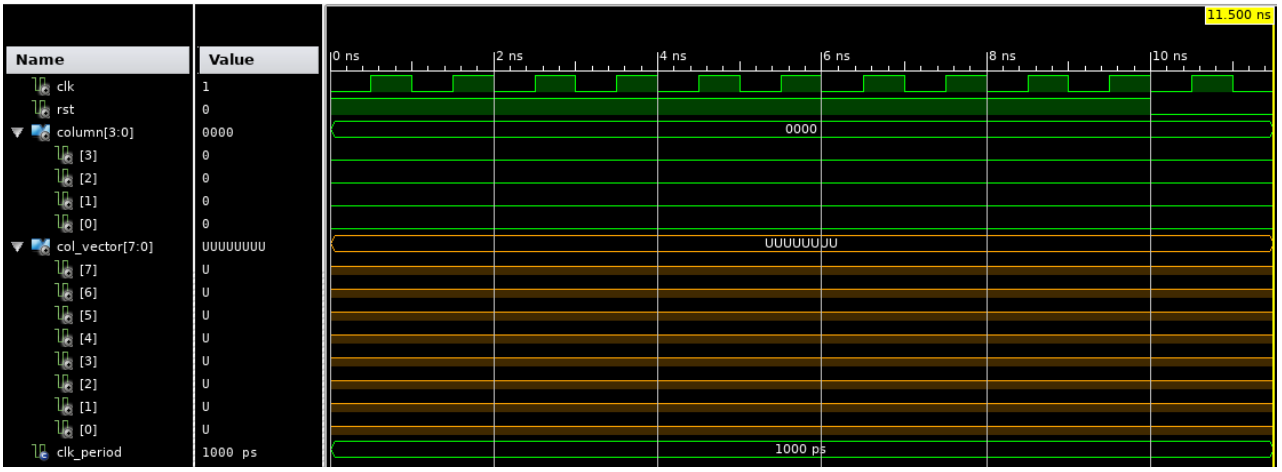
Na obrázku 2 je zobrazená simulácia pamäte. Hodnoty vo vektore `col_value` sa menia podľa toho, na akú polohu v pamäti ukazuje `index`. `Indexy` sa skladajú z dvoch poradí hodnôt. Prvé je od 0 do 7 a druhé je od 36 do 43. Na signáloch nižšie je patrne vidieť polovicu prvého obrázku a takmer celý tretí obrázok.



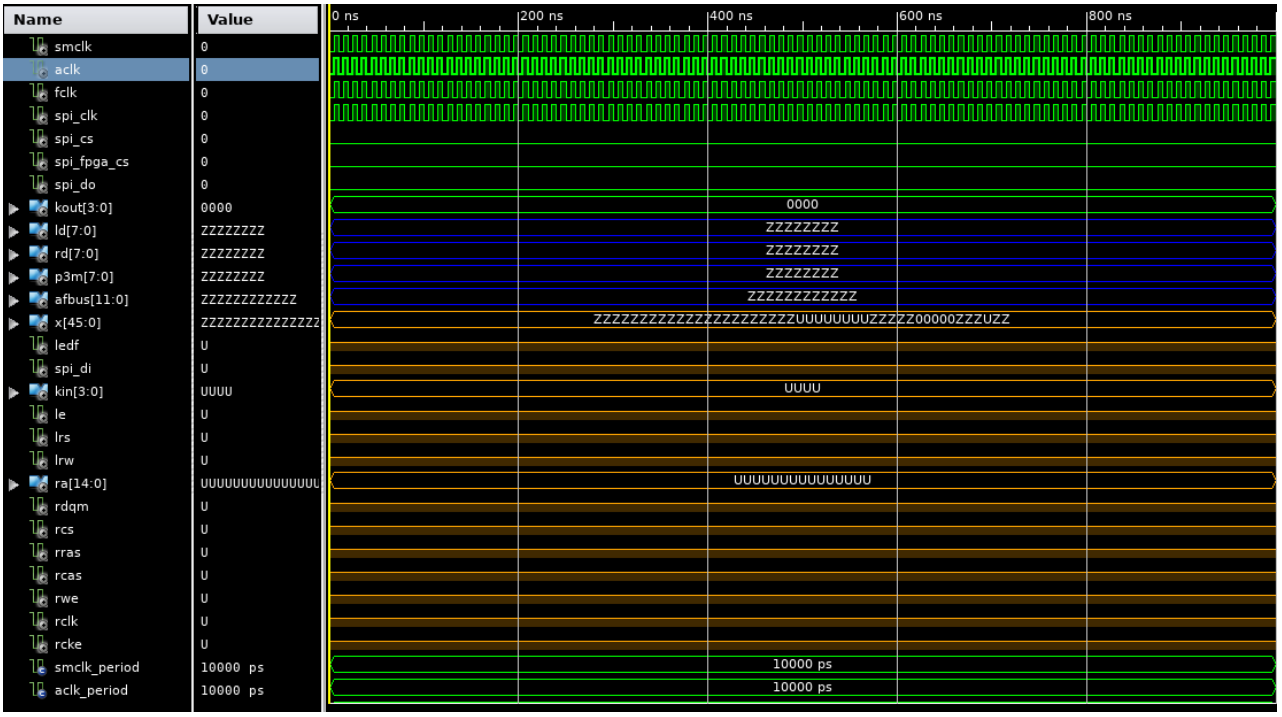
Obr. 2: Simulácia pamäte

5 Simulácia FSM a systému

Simulácie FSM a systému sa mi nepodarilo sprovozniť, za čo sa ospravedlňujem. Simulácia FSM skončila vždy po jeden a pol cykle mimo `RESET` = 1 a v systémovej simulácii sa nespustil `CLK`.



Obr. 3: Simulácia FSM



Obr. 4: Simulácia systému

6 Video s implementáciou

Video s implementáciou je dostupné v lokalite: video link
Vo videu sa najprv nahrá nanovo program do fitkitu a potom sa prejde cez 4 obrázky. Po skončení cyklu na 4 obrázku sa znova načíta prvý obrázok a cyklus pokračuje ďalej.