VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Seminář VHDL

Projekt – finální verze

22. dubna 2021

Michal Pyšík (xpysik00)

1 Úvod

Jedná se o projekt pro Fitkit 2.0, pracující s externím maticovým displejem o velikosti 16x8 světelných LED buněk. Po spuštění se uprostřed scény vykreslí postavička, která provede 3 rotace doprava, 3 rotace doleva, a na závěr se vykreslí posloupnost 3 obrázků (obsahující text). Tato smyčka se poté stále opakuje (a stiskem `RESET`u se kdykoli vrátí na svůj začátek).

Komentovanou úkázku ve formě videa lze nalézt zde: https://youtu.be/d9GLYtgNLVI

2 Implementace

Implementace se skládá ze souborů `matrix_pack.vhd`, `counter.vhd`, `display.vhd`, `cell.vhd` a hlavní entity `top.vhd`. Dále jsou popsány pouze poslední 2 soubory, které se týkají tohoto finálního úkolu.

2.1 Entita buňky (cell.vhd)

Jedná se o synchronní komponentu, která představuje právě jednu LED diodu maticového displeje. Při každé náběžné hraně vstupního signálu `CLK` zkontroluje, zda je aktivní vstupní signál `EN` (v hlavní entitě namapován na signál `EN_frame`). Pokud aktivní je, rozhoduje se podle vstupního signálu `DIRECTION` typu `DIRECTION_T`. Pokud je jeho hodnota `DIR_RIGHT`, do výstupního signálu `STATE` zkopíruje hodnotu svého levého souseda, která je hodnotou vstupního signálu `NEIGH_LEFT`. Pro hodnotu `DIR_LEFT` provede symetricky opačnou operaci.

2.2 Hlavní entita včetně ovladače chování (top.vhd)

Hlavní entita se kromě svých konstant a signálů skládá z entity `display`, čítače `counter_frame`, 128 instancí buňek, a procesu ovladače celého děje `event_controller`. Entita `display` z třetího podúkolu má v sobě zapouzdřenou svou vlastní instanci čítače, který je nastaven na výstupní frekvenci 1600 Hz, což odpovídá obnovovací frekvenci 100 Hz pro celý maticový display, jelikož se skládá z 16 řádků, mezi kterými se „multiplexuje". Displej vždy vykresluje obsah uložený ve vektoru `DATA`.

Čítač `counter_frame`, který reguluje signál `EN_frame`, řídí počet snímků vykreslených za vteřinu. Po různém experimentování jsem se rozhodl pro hodnotu 8 Hz (= 8 FPS), jelikož jsem si oblíbil rychlejší animace ($1600 \ mod \ 8 = 25000000 \ mod \ 8 = 0$ ✓). Signál `EN_frame` je enable signálem jak pro buňky, tak pro samotný ovladač chování.

Celá smyčka se skládá ze 192 snímků (má tedy 24 sekund), přičemž prvních 48 snímků provádí postavička rotace doprava, dalších 48 snímků rotace doleva, a zbývajících 96 snímků je rovnoměrně rozděleno mezi mé statické obrázky. To že zobrazení mých snímku trvá stejně dlouho jako 6 rotací samozřejmě není náhoda. Původně jsem prováděl různé složité pokusy s impementací, ale měl jsem výrazné problémy se synchronizací. Rozhodl jsem se tedy pro „inženýrské" řešení – při posledních 96 snímcích postavička ve skutečnosti rotuje 6x doprava, my to pouze nevidíme, protože do vektoru `DATA` jsou přesměrovány statické obrázky, místo výstupu buňek `cell_DATA` (vše jsou 128-bitové vektory).

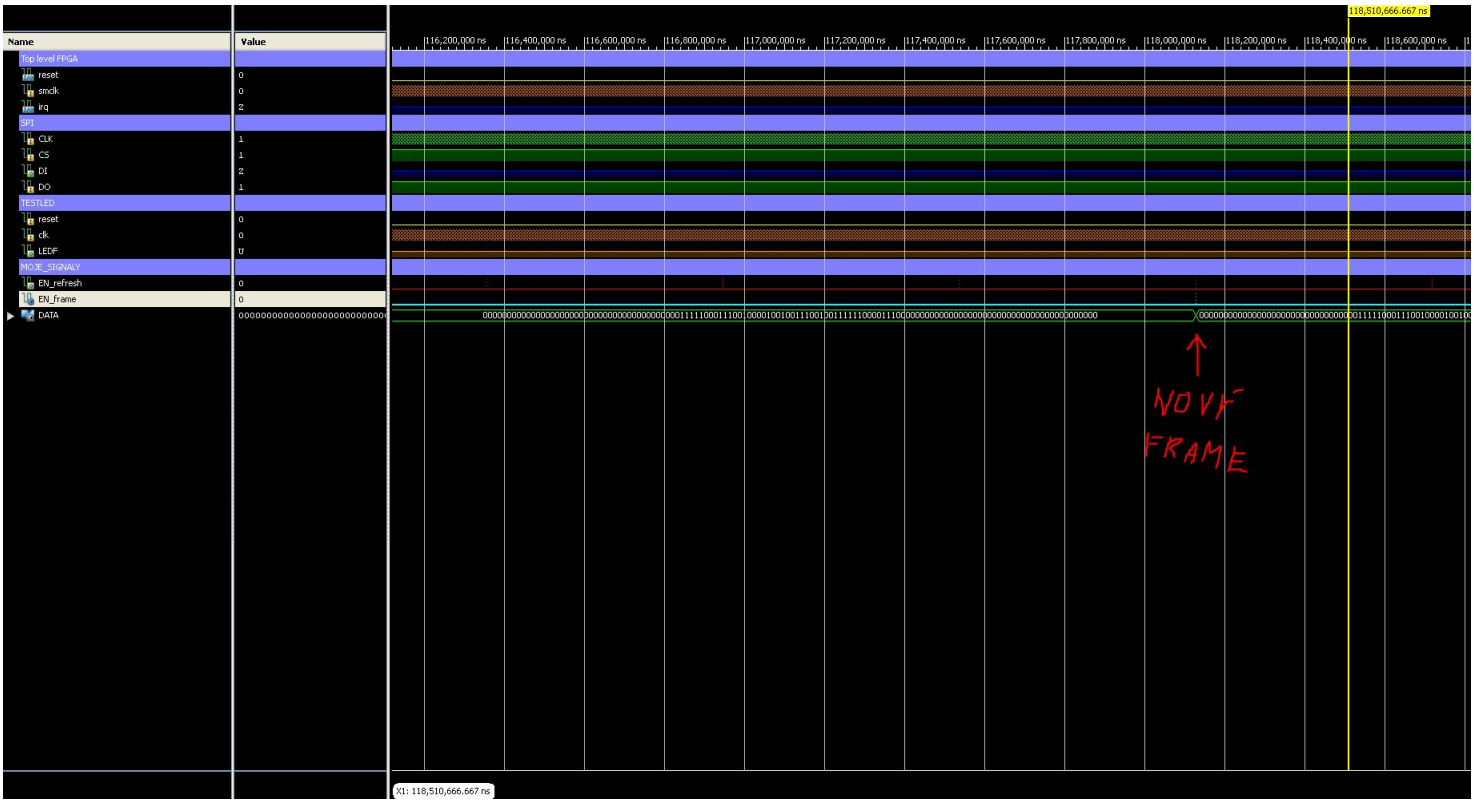
Instancí buňek je vytvořeno přesně 128, iniciální hodnotu získají pomocí funkce `GETID` z 16x8 matice obsahující statický obrázek postavičky. Mapování portů probíhá tak, aby byly schopné své další hodnoty odvozovat pouze z hodnot daného souseda. Při každé aktivaci signálu `EN_frame` generují nový posunutý snímek do vektoru `cell_DATA`.

Samotný ovladač chování je implementovaný jako konečný stavový automat (není ale psán formou enumerovaných stavů). Obsahuje čítač, který se cyklý přes 192 hodnot reprezentující snímky, přičemž řídí přechody mezi stavy popsanými o 2 odstavce výše. Mezi jeho režii patří například změna směru pro buňky a to, zda se do vektoru `DATA` přesměrovává výstup buňek, nebo nějaký z mých statických obrázků. Je to opět synchronní komponenta, která provede akci jen když je aktivní signál `EN_frame` společně s náběžnou hranou hodinového signálu `CLK`.

3 Testování a simulace

Tentokrát jsem použil přímo soubor `tb.vhd` z kostry třetího podúkolu, k němuž jsem do složky `sim` přidal i testbenche z minulých podúkolů. Přidal jsem si pouze labely pro 3 signály, které považuji z vnějšího pohledu za nejdůležitější:

- EN_refresh odpovídá signálu `EN` zapouzdřeném v entitě `display`, je tedy aktivován 1600x za sekundu a řídí přepínání mezi sloupci displeje.
- EN_frame je signál aktivovaný pouze 8x za sekundu, řídí vykreslení dalšího snímku. Odsimuloval jsem pouze 150 ms interval obsahující jediný přechod mezi snímky (viz. obrázek), celých 24 sekund by se simulovalo nesmírně dlouhou dobu.
- DATA je vektor odpovídající tomu, co displej zobrazuje. Z posloupnosti 128 binárních číslic si samozřejmě člověk obrázek moc nepředstaví, je zde uveden spíše proto, aby šel viděl přechod mezi snímky při aktivaci `EN_frame`



Obrázek 1: Screenshot simulace, doporučuji zoom na náběžné hrany obou EN signálů (tečkované)