

Světelné efekty na FPGA a FITkitu

projekt do předmětu IVH 2022 / 2023

Zadání

Implementujte světelné efekty na maticovém displeji pouze s využitím FPGA. Aplikace se bude ovládat pomocí vestavěné vnitřního stavového automatu. Zobrazování bude realizováno pomocí maticového displeje 16x8 bodů, který bude možné zapůjčit. Na displeji se zobrazí obrázek, který se bude posouvat (rotovat) doleva či doprava. Směr se bude po třech iteracích měnit a na závěr proběhne jednoduchá animace

Odevzdávání

Celý projekt je nutné odevzdat do 8. května 2022; jednotlivé podúkoly do termínů uvedených níže.

Ovládání

bez ovládání uživatelem. Projekt bude obsahovat ROM paměť se 4 stavy, které se budou cyklicky procházet. Pro každý počáteční stav se dokola bude pouštět tato smyčka

- 3x rotace vašeho obrázku doprava (tj. 3x16 kroků)
- 3x rotace vašeho obrázku doleva (tj. 3x16 kroků)
- zmizení nahoru (tj rolování nahoru s tím, že se bude doplňovat odspodu nulami)
- spuštění vlastní animace

smyčka by měla proběhnout **do dvou minut**.

Specifikace

Při implementaci je možné použít nebo se inspirovat libovolným kódem z repozitáře projektu FITkit. Při přednáškách předmětu IVH budou všechny nutné prerekvizity vysvětleny.

Aplikace nebude mít žádný centrální řídicí prvek, ale bude obsahovat soustavu 16 sloupců, které budou navzájem propojeny a které spolu budou lokálně komunikovat. Každý sloupec bude mít na starosti řízení svitu osmi LED diod v jednom sloupci LED diody maticového displeje. Chování jednotlivých součástí je popsáno následovně. Od chování se můžete odchýlit (mimo podúkoly), je však nutné, aby byl dodržen koncept 16 samostatných sloupců tvořených konstrukcí `for-generate`.

Sloupec

- synchronní prvek
- generický popis pro N řádků
- stav buňky bude vyjádřen pomocí jednoho bitu (svítí / nesvítí)

- pokud bude posun vlevo / vpravo, tak se převezme stav od sousedů. Pokud se bude vyžadovat posun nahoru, provede se operace shift ve vnitřních stavech.
- signály entity
 - CLK - hodinový signál
 - RESET - resetovací signál
 - STATE - výstupní stav (N-bitový vektor)
 - INIT_STATE - vstupní stav (N-bitový vektor)
 - NEIGH_LEFT - stav souseda nalevo (N-bitový vektor)
 - NEIGH_RIGHT - stav souseda napravo (N-bitový vektor)
 - DIRECTION - posun vlevo nebo vpravo nebo nahoru - typu DIRECTION_T (viz úkol 1)
 - EN - povolení provedení akce posunu

Ovladač chování

Navrhněte patřičný stavový automat.

Časovač

Pro generování povolovacích signálů, použijte generický zápis a správně vypočtěte, jakou hodnotu pro jakou frekvenci změn potřebujeme.

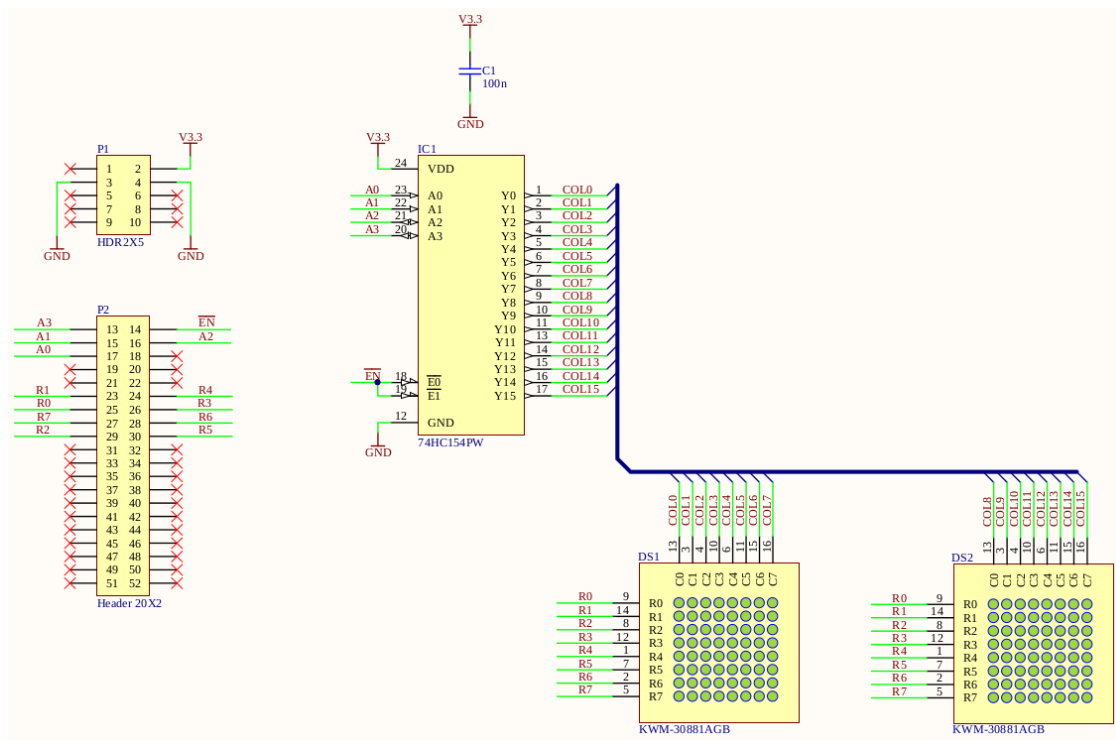
Výstupní řadič

Je použit pro řízení maticového displeje.

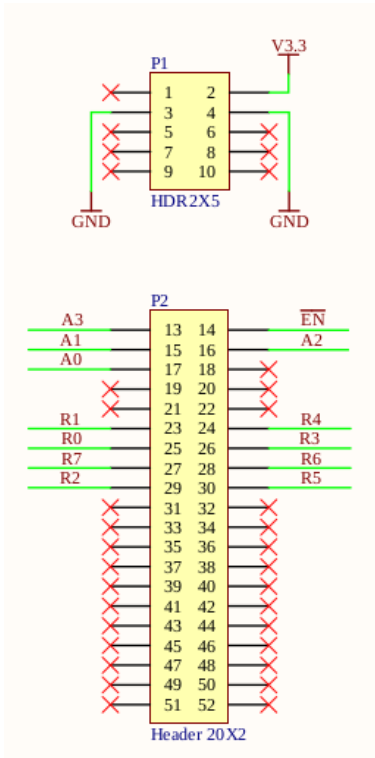
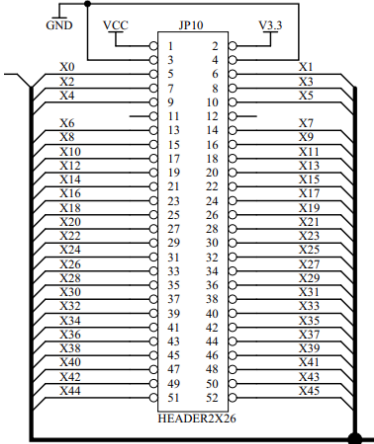
Maticový displej má vstupy:

- A0-A3: 4bitová identifikace sloupce, se kterým se pracuje - řádky jsou propojeny přes hardwarový dekodér 74HC154W (4-to-8 decoder) - plňte čísla 000 až 111
- R0-R7: 8bitová hodnota udávající, který řádek (ve vybraném sloupci) má svítit

Zobrazování využívá tzv. multiplexingu, neboli zobrazování jednotlivých sloupců za sebou. V jednom taktu nastavíte A3-A0 na 000 a R7-R0 na požadovanou konfiguraci a určitou dobu počkáte (zvolte vhodnou obnovovací frekvenci, aby displej neblíkal, ale byl čas pro rozsvícení LED). Pak nastavíte A3-A0 na 001 a R7-R0 na (jinou) hodnotu. A tak dále. Periodicky tento děj opakuje.



Pro propojení na fitkitu využijte mapování vektorů pro adresu sloupce a řádku na výstupní port X. Vycházejte ze schématu zapojení FITkitu.

Přípravek	FitKit 2.0
	

Animace

Tato funkce může být implementovaná pomocí speciálního signálu do buňky (povolení animace) a např. generické proměnné, která říká, jestli buňka bude při animaci svítit, nebo bude zhaslá. Příkladem animace je to, že se např. na 1 sekundu zobrazí “šachovnice” nebo “sudé řádky budou svítit”.

Dokumentace

Součástí řešení je stručná dokumentace, obsahující zejména odůvodnění použití různých **konstant** (zejména pro časovače). Dále by měla obsahovat výstup (screenshot) **simulace**, kde budou zvýrazněny určité stavy a v textu zdůvodněno, jestli se komponenta chová správně. Simulace budou minimálně 4 - pro entitu sloupce, FSM řadič, ROM paměť a pro celý systém. Tyto simulace nemusí nutně obsahovat konstrukci assert. Součástí dokumentace bude odkaz na vámi natočené video funkční implementace s komentářem o délce max 2 minuty. Pro uložení můžete použít např. <http://nextcloud.fit.vutbr.cz/>, youtube.com, drive.google.com, atd.

Hodnocení

Projekt bude hodnocen následovně:

- První podúkol

až 10 bodů

- Druhý podúkol až 15 bodů
- Ovládání sloupce a instanciací (for-generate) až 20 bodů
- Funkčnost posuvů až 15 bodů
- Funkčnost výstupní animace až 10 bodů
- Paměť a její testbench až 10 bodů
- Testbench FSM (obrázek v dokumentaci) až 10 bodů
- Další rozšíření projektu až 10 bodů

Hodnocení bude zohledňovat funkčnost, efektivitu, přehlednost, dodržení zvyklostí jazyka VHDL a případná implementovaná rozšíření nad rámec zadání. V případě nejasností může být vyžadována osobní prezentace řešení.

Pro získání zápočtu je nutné získat více jak 50 bodů celkem a také minimálně 1 bod z obou podúkolů.

Podúkoly

1. Podpůrné funkce

Navrhněte a odevzdejte do IS VUT implementaci balíku `effects_pack` v souboru `effects_pack.vhd`, který bude obsahovat

- funkci `GETCOLUMN`, která bere na vstupu jeden `STD_LOGIC_VECTOR`: `DATA` a dvě čísla typu `integer`: `COLID`, `ROWS` s tím, že bude vrátet `STD_LOGIC_VECTOR` vybraný z vektoru o délce `ROWS * COLS` (neboli mapování souřadnic 2D pole do souřadnic 1D pole). Tento vektor bude indexovaný směrem nahoru (`0 to ROWS*COLS-1`). Pokud `COLID` bude větší, než je maximální možný počet sloupců (odvozeno z velikosti pole), vrátíte první (nultý) sloupec, pokud bude menší než 0, vrátíte poslední sloupec. Tím vám tato funkce pomůže při výpočtu sousedních hodnot

Deklarace funkce bude vypadat následovně

```
function GETCOLUMN ( signal DATA : in std_logic_vector; COLID : in
integer; ROWS : in integer) return std_logic_vector;
```

Příklad:

```
SIGNAL DATA : std_logic_vector(0 to 11) := "101001100111";
```

pak platí, že:

- `GETCOLUMN(DATA, 0, 3) = "101"`
- `GETCOLUMN(DATA, 1, 3) = "001"`
- `GETCOLUMN(DATA, 0, 3) = "101"`
- `GETCOLUMN(DATA, 4, 3) = "101"`
- Výčtový typ `DIRECTION_T`, který bude obsahovat směr doleva (`DIR_LEFT`), doprava (`DIR_RIGHT`) a nahoru (`DIR_TOP`).
- Funkci navracející `NEAREST2N` nejbližší vyšší mocninu dvou k zadanému číslu s deklarací:

```
function NEAREST2N ( DATA : in natural ) return natural;
```

Příklad: pro 6 vrací 8, pro 42 vrací 64 atd.

Dále odevzdejte soubor [effects_pack_tb.vhd](#), který bude pomocí konstrukce ASSERT testovat správnou funkci funkcí GETCOLUMN a NEAREST2N. Pro zjednodušení můžete počítat, že velikost vstupního vektoru GETCOLUMN je dělitelná počtem sloupců.

Termín: 13. 3. 2023

2. Čítač / časovač

Navrhněte a odevzdejte do IS VUT implementaci časovače ([counter.vhd](#)) a jeho testbench ([counter_tb.vhd](#)), který bude testovat jeho funkčnost. Kostru entit stáhněte z dokumentového skladu e-learningu. Čítač bude mít 3 enable výstupy (bude tříkanálový).

Čítač bude mít 3 generické parametry: OUT1_PERIOD, OUT2_PERIOD, a OUT3_PERIOD (všechny typu positive). Například čítač s periodou OUT1_PERIOD = 3 (t.j., každé 3 cykly) aktivuje na **právě na jeden** hodinový cyklus signál EN1. Analogicky to bude fungovat i pro další dva kanály.

Napište testbench, který otestuje, že čítač opravdu za zvolený časový interval aktivuje signál EN1, EN2 i EN3. Testbench musí obsahovat funkci ASSERT pro testování. Počítejte s tím, že při zkoušení poběží testbench 100 ms.

Termín: 3. 4. 2023

Výsledný projekt

Výsledný odevzdávaný projekt bude projektem pro Xilinx ISE. Jeho kostra bude vložena v dokumentovém skladu ve e-learningu. Vaším úkolem bude implementovat celý projekt pro platformu Fitkit. Tento kit si můžete půjčit v knihovně (pozor, žádejte **FITKIT verze 2!**). Programování a práce s kitem bude představena v samostatném semináři. Projekt pojmenujte **ivh_projekt**.

Pro řízení časového děje byste měli využít čítač, který jste navrhli ve druhém úkolu. Pokud nebyl správně, je nutné jej opravit. Odevzdávejte soubor [display.zip](#), který bude obsahovat

- všechny soubory VHD z projektu a soubor **ivh_projekt.xise** - **neodevzdávejte složku build!**
- všechny testbenche
- soubor **doc.pdf**, který bude obsahovat dokumentaci

Termín: 8. 5. 2023

Konzultace

Všechny potřebné informace k řešení projektu se dozvíte na přednáškách - některé přednášky budou organizované jako demonstrační cvičení, kde si na příkladu detailně ukážeme, jak při řešení projektu postupovat.

Další dotazy je možné řešit po přednášce nebo online po domluvě předem s Vojtěchem Mrázkem (mrizek@fit.vutbr.cz)