

Laboratorium #II-5. Praca na ekranie VGA w trybie tekstowym.

- Obsługa pamięci czcionek.
- Wypisywanie czcionek na ekranie.

Używane elementy: Basys3

Temat ćwiczenia.

Zadanie do wykonania w ramach ćwiczenia to opracowanie programu, który będzie na ekranie wyświetlał prostokąt z określoną zawartością tekstową. Rozmiar prostokąta będzie wynosił 16 x 16 znaków, a rozmiar każdego znaku to 8 x 16 pikseli.

Przebieg ćwiczenia

W ćwiczeniu dodasz do swojego projektu nowe trzy moduły:

pamięć ROM czcionek (**font_rom**) – dostępny na UPEL,

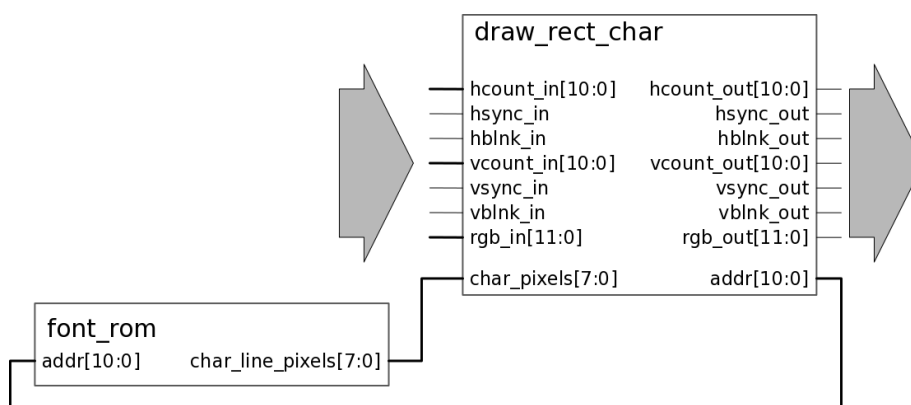
pamięć ROM wyświetlanego tekstu (**char_16x16**) – do napisania,

moduł kontrolny **draw_rect_char** – do napisania.

Docelowy efekt ćwiczenia najlepiej osiągnąć stosując następujące kroki.

Krok 1 – wyświetlanie czcionek na ekranie.

Do swojego dotychczasowego projektu dodaj dwa nowe moduły, wg schematu poniżej.



Zawartość modułu **font_rom** znajdziesz na UPELu. Przeglądnij kod aby zrozumieć, jakich danych moduł oczekuje i co otrzymujemy na wyjściu modułu.

Moduł **draw_rect_char** budujemy standardowo, podobnie jak moduły z poprzednich ćwiczeń (będziesz musiał opóźnić wszystkie sygnały sterujące VGA, tak aby wyrównać opóźnienie sygnałów **rgb_out**). Możesz do tego celu wykorzystać dostępny na UPEL moduł **delay**.

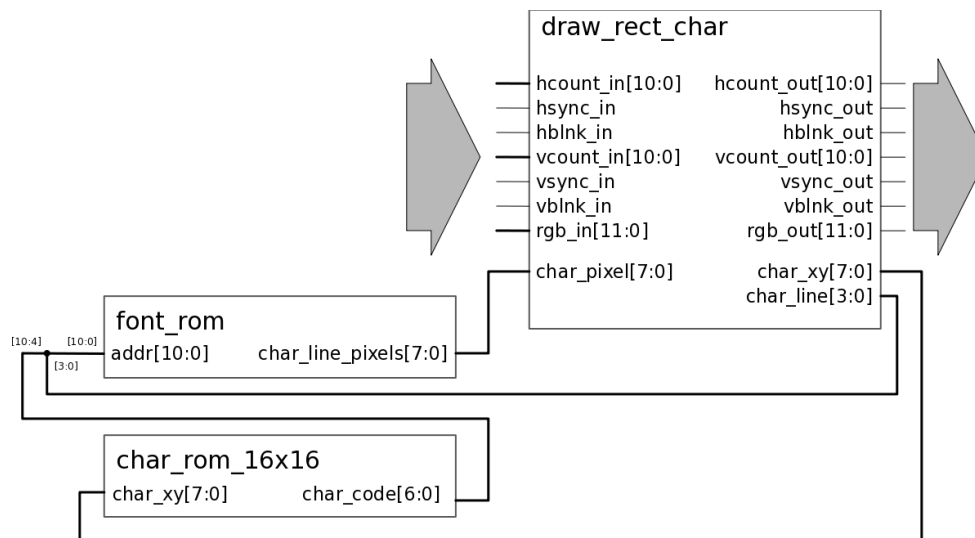
Moduł należy skonstruować tak, żeby wyświetlał na ekranie linijkami kolejne znaki wg narastających kodów ASCII. Czyli: każde 16 linii na ekranie wyświetla kolejne znaki o kodach ASCII od zera w górę, na całej szerokości ekranu.

Zwróć uwagę ponieważ znaki mają szerokość 8 pikseli szerokości, więc możesz wykorzystać bity **hcount_in[9:3]** jako kod ASCII znaku. Podobnie, możesz wykorzystać bity **vcount_in[3:0]** do wyznaczenia linii znaku, która ma być odczytana z ROMu.

Krok 2 – budowa kwadratu 16 x 16

Ogranicz rozmiar wyświetlanego obszaru do 16 x 16 znaków.

Dodaj nowy moduł **char_rom_16x16** wg rysunku poniżej. Moduł ten powinien zawierać dowolne 256 znaków, które należy linia po linii umieścić. Zwróć uwagę, że moduł **draw_rect_char** ma teraz osobne wyjścia wskazujące pozycję znaku w kwadracie (**char_xy**), oraz nr linii wyświetlanego znaku.



Po prawidłowym wykonaniu tego kroku powinieneś na ekranie widzieć prostokąt wypełniony określonym tekstem.

Krok 3 – parametryzacja modelu.

Dodaj do modelu parametry określające pozycję wyświetlanego prostokąta na ekranie.

Wyniki ćwiczenia

Jako wynik ćwiczenia należy:

- załadować projekt na UPEL (bez katalogu symulacyjnego i implementacyjnego).

ZA BRAK ZAŁADOWANEGO PROJEKTU NA UPEL OTRZYMUJEMY -1 pkt. DO OCENY.

- zaprezentować działanie programu na następnych zajęciach laboratoryjnych i wyjaśnić słownie zasadę działania.

UWAGA: Kod projektowanego modułu powinien być napisany zgodnie z zasadami opisanymi w pliku „Zasady pisania kodu w języku Verilog”, dostępnego na UPEL. Będzie to oceniane!

Wyniki będą ocenione bezpośrednio na kolejnych zajęciach.