LAB VHDL – Pamięć RAM

Przedmiotem niniejszego projektu laboratoryjnego jest zaprojektowanie i przetestowanie w zaproponowanym testbench'u modelu pamięci statycznej RAM o strukturze jak układ 6116. Wielkość pamięci należy zdefiniować jako generic na poziomie definicji entity. Pamięć należy zaprojektować jako jedno entity RAM_STATIC, które będzie miało dwie wersje architecure:

- RAM_Ideal – jest architekturą, która modeluje pamięć bez uwzględniania parametrów czasowych ;
- RAM_Timing – jest architekturą, która modeluje pamięć z uwzględnieniem parametrów czasowych.

RAM_STATIC Model

Interfejs entity powinien mieć następującą postać **(proszę zachować kolejności nazwy)**:

```
entity RAM_STATIC is
generic(
constant n_address: integer := 7; -- number of address lines 0..7
constant m_data:    integer := 7; -- number of data lines 0..7
constant tACS:      time := 120 ns; -- symbol meaning as in RAM6116 datasheet
constant tCLZ:      time := 10 ns;
constant tCHZ:      time := 10 ns;
constant tOH:       time := 10 ns;
constant tWC:       time := 120 ns;
constant tAW:       time := 105 ns;
constant tWP:       time := 70 ns;
constant tWHZ:      time := 35 ns;
constant tDW:       time := 35 ns;
constant tDH:       time := 0 ns;
constant tOW:       time := 10 ns);

port(CSn, WEn, OEn: in std_logic;
        Address: in std_logic_vector(n_address downto 0);
        Data: inout std_logic_vector (m_data downto 0) := (others => 'Z'));

end entity RAM_STATIC;
```

Opis struktury i działania pamięci zawiera plik 6116.pdf (Moodle lub Internet).

Architecture Ideal

Działanie układu, bez uwzględniania parametrów czasowych przedstawia tabela "Truth Table" na stronie 2 dokumentacji 6116.pdf.
Użyteczne wskazówki do modelu „"Ideal":

- Wykorzystaj strukturę ARRAY do modelowania pamięci;
- Adresowanie tablicy jest możliwe typem INTEGER, stąd konieczna odpowiednia konwersja szyny adresowej typu STD_LOGIC_VECTOR;
- gdy CSn = '1' wówczas DATA<='ZZZZZZZZ"
- gdy CSn='0' i narastające zbocze WEn wówczas zapisz do pamięci dane z DATA. Aby zabezpieczyć poprawne działanie w przypadku, gdy Address i Wen zmieniają się w tym samym czasie można zastosować ADDRESS określający miejsce zapisu w pamięci/tablicy (ARRAY) zmodyfikowany o atrybut 'delayed (ADDRESS'delayed). Wówczas brana jest wartość sygnału ADDRESS opóźniona o delta.
- Należy pamiętać o dodaniu odpowiedniego opóźnienia w celu uaktualnienia zawartości pamięci (wewnątrz procesu potrzebne jest minimum „wait for 0 ns" .

Architecture Timing

Działanie z uwzględnieniem zależności czasowych przedstawione jest na stronie 7 i 9 pliku 6116.pdf odpowiednio dla cyklu READ i WRITE.

READ mode:
W przypadku przedstawionym na wykresie "Timing Waveform of Read Cycle No. 2" (str.7), sygnały CSn i OEn są aktywne (LOW) przed zmianą na linii adresowej ADDRESS. Wówczas poprzednie dane pozostają na szynie danych przez czas tOH (Output Hold from Address Change), a następnie na szynie danych zmieniają się dane , aby ustabilizować się po czasie tAA (Address Acsess Time). Adres musi być stabilny przez czas tRC (Read Cycle Time).

W przypadku przedstawionym na wykresie "Timing Waveform of Read Cycle No. 3" (str.7), sygnał OEn jest aktywny (LOW) i adres był/jest stabilny przed zmianą CSn na LOW. W czasie gdy CSn jest HIGH szyna danych jest w stanie wysokiej impedancji 'Z' (na wykresie linia pomiędzy LOW i HIGH). Gdy CSn zmienia się na LOW linia danych wychodzi ze stanu wysokiej impedancji 'Z' po czasie tCLZ (Chip Select to Output in Low-Z) I wchodzi w stan przejściowy (dane przejściowe), aby po czasie tACS (Chip Select Access Time) liczonym od zmiany CSn ustabilizowały się z nową wartością. Szyna danych wraca do stanu wysokiej impedancji po czasie tCHZ (Chip Deselect to Output in High-Z) od momentu powrotu sygnału CSn do stanu HIGH.

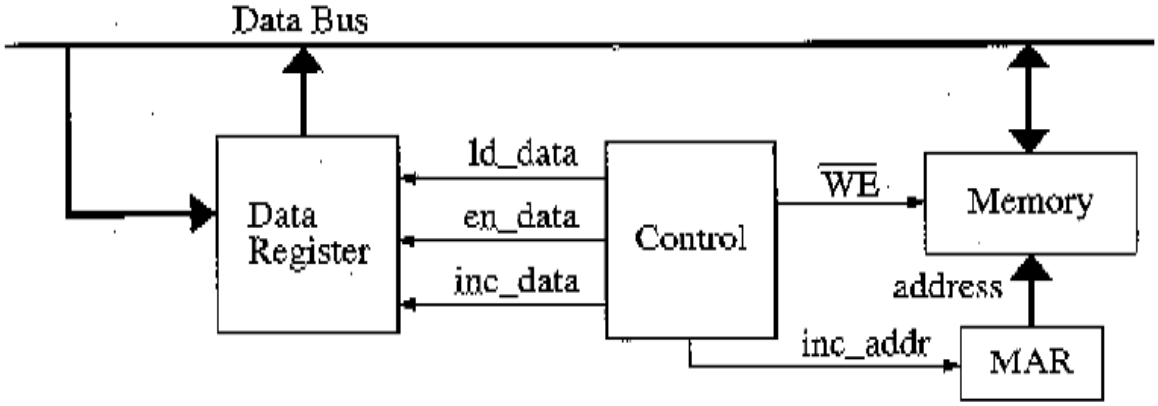
WRITE mode:
Wykres „Timing Waveform of Write Cycle No. 1" (str.9) przedstawia cykl zapisu do pamięci w trybie „WE Controlled Timing". W tym trybie OEn jest LOW w całym cyklu zapisu, który jest sterowany sygnałem WEn. Zakładamy, że CSn zmienił się na LOW przed lub równolegle ze zmianą WEn na LOW i WEn zmienia się na HIGH przed lub razem ze zmianą CSn na HIGH. Zakreskowany obszar na wykresie sygnału CSn oznacza czas, w którym sygnał ten może zmieniać się na LOW i odpowiednio na HIGH. Adres musi być stabilny w czasie tAS (Address Set-up Time), zanimWEn zmieni się na LOW. Po czasie tWHZ szyna danych wyjściowych (wewnątrz pamięci) przechodzi w stan wysokiej impedancji, co umożliwia podanie danych wejściowych na szynę DATA. Dane przeznaczone do zapisu muszą być stabilne przez czas tDW (Data to Write Time Overlap = SetupTime) zanimWEn zmieni się na HIGH i powinny być

stabilne jeszcze przez HoldTime tDH (Data Hold from Write Time). Adres musi być stabilny przez tWR (Write Recovery Time) po przejściu WEn w HIGH. Po przejściu WEn w stan HIGH, pamięć wraca do trybu READ i po czasie minimum tOW (Output Active from End-of-Write), po przejściu przez stan przejściowy, na wyjściu pojawią się zapisane dane. Dalsze zmiany na szynie danych wyjściowych mogą nastąpić w konsekwencji zmiany adresu bądź zmiany CSn na HIGH.

Drugi tryb cyklu zapisu (CS Controlled Timing) przedstawiony jest na wykresie "Timing Waveform of Write Cycle No. 2" (str.9). W tym trybie OEn jest utrzymywane w stanie LOW. Zakładamy, że w tym przypadku WEn zmienia się na LOW przed lub w tym samym czasie, gdy CSn zmienia się na LOW. CSn zmienia stan na HIGH przed, lub w tym samym czasie, gdy w stan HIGH przechodzi sygnał WEn. Adres musi być stabilny co najmniej tAS (Address Setup Time) przed zmianą CSn na LOW. Dane muszą być stabilne przez setup time, czyli tDW (Data to Write Time Overlap) przed zmianą CSn na HIGH i muszą być nadal stabilne („trzymane") przez hold time tDH po zmianie CSn na HIGH. Adres ma być stabilny jeszcze przez czas tWR po zmianie CSn na HIGH. Ten tryb zapisu jest podobny do poprzedniego, gdyż w obu przypadkach zapis do pamięci odbywa się gdy CSn i WEn są w stanie LOW i zapis jest zakończony, gdy jeden z tych sygnałów zmieni stan na HIGH.

RAM_STATIC Testbench

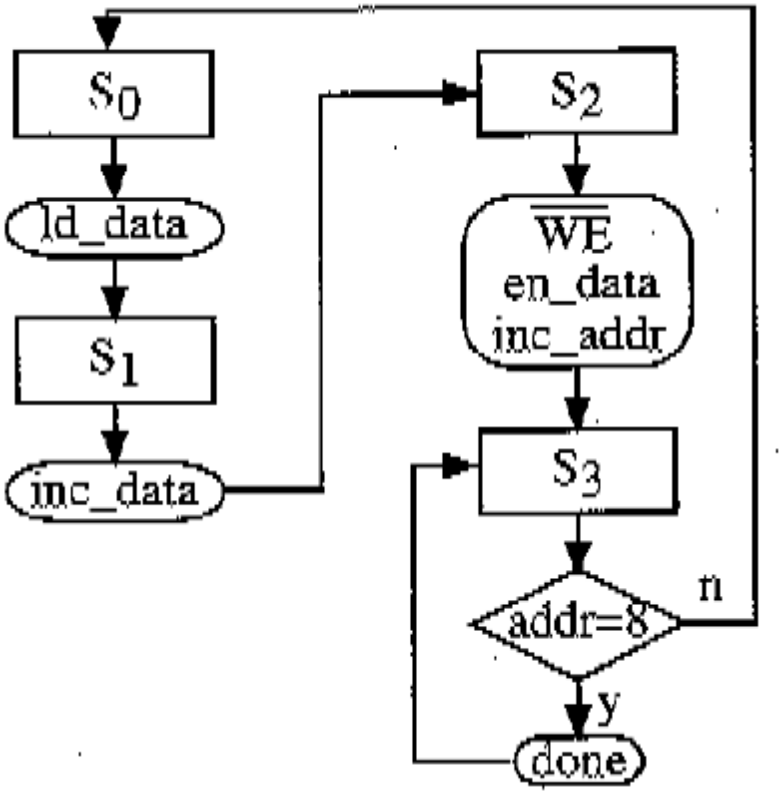
W celu przetestowania pamięci należy zastosować strukturę przedstawioną na Rys. 1. Zaproponowaną strukturę można/należy dopasować do swoich potrzeb. Wymagany jest podział na część sterującą (maszyna stanów) i pozostałe elementy.



Rys. 1 Struktura układu testbench dla RAM_STATIC.

Struktura testbench'u zawiera rejestr adresowy (Memory Address Register – MAR) oraz rejestr danych , gdzie będą trafiały dane odczytane z pamięci (Data Register). Idea testu polega na odczycie pamięci i zapisaniu danych do Data Register, zwiększeniu odczytanej wartości o „1" i ponownym zapisie po ten sam adres (początkowo pamięć była wyzerowana). Następnie wartość adresu zostaje zwiększona o 1, aż przetestujemy całą zawartość pamięci.

Głównym zagadnieniem do rozwiązania w tesbench'u jest zaprojektowanie układu generującego sekwencje sygnałów sterujących pamięcią oraz danych i adresu w taki sposób, aby pokazać sytuacje, w których niespełnione są wymogi czasowe pamięci. Wówczas za pomocą komend assert i report powinny pojawić się odpowiednie komunikaty (na ekranie i w pliku zawierającym wyniki testów. Szczegóły implementacyjne pozostawia się projektantowi (studentowi). Obraną procedurę testów proszę szczegółowo opisać z zaznaczeniem na przebiegach miejsc, gdzie są niespełnione testowane zależności czasowe. Diagram na Rys.2 przedstawia przykładowy algorytm testowania. Aby spełnić wymogi projektu należy go odpowiednio zmodyfikować.



Rys. 2 Przykładowy (do modyfikacji) algorytm testowania pamięci.