Основы VerirogHDL/SystemVerilog (синтез и моделирование)

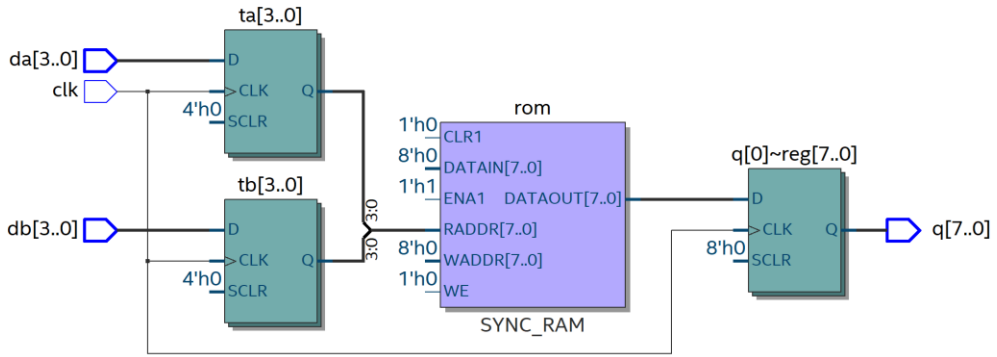
Задание lab7 (4 часа лабораторных занятий)

Часть lab7_1

- На языке Verilog создайте описание параметризированного без знакового умножителя двух чисел разрядностью N(параметр).
- Умножение должно быть реализовано на ROM памяти: адресные входы – множимое и множитель; выход памяти ROM – результат умножения.
- Инициализацию ROM следует выполнить в процедурном блоке initial, с использованием циклов.
- Базовое значение параметра N: 4
- Выводы устройства для моделирования и реализации на плате
  - ✓ вход множимого – [N-1:0]da (при реализации на плате подать на sw[7:4]); На входе следует использовать регистр.
  - ✓ вход множителя – [N-1:0]db (при реализации на плате подать на sw[3:0]); На входе следует использовать регистр.
  - ✓ clk – вход тактового сигнала
  - ✓ выход – результат умножения - [N-1:0]q (при реализации на плате подать на led[7:0]); На выходе следует использовать регистр.

Часть lab7_1 продолжение описания

- Ожидаемая структура модуля (при базовом значении параметра N)

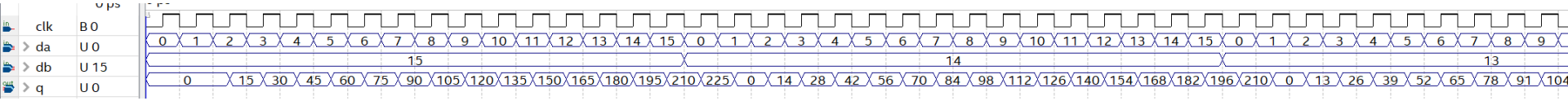


- Пример текстового описания модуля

```
1  module lab7_1
2  #(parameter DATA_WIDTH=4)
3  (  input [(DATA_WIDTH-1):0] da, db,
4     input clk,
5     output reg [(2*DATA_WIDTH-1):0] q);
6  // ROM memory array
7  reg [2*DATA_WIDTH-1:0] rom[2**(2*DATA_WIDTH)-1:0];
8  // Internal varaibles
9  reg [DATA_WIDTH-1:0] ta, tb, ia, jb;
10 // Specify the initial contents of the ROM.
11 initial begin : INIT
12    integer i,j;
13    for(i = 0; i <= 2**DATA_WIDTH-1; i = i + 1)
14       for(j = 0; j <= 2**DATA_WIDTH-1; j = j + 1)
15       begin
16          ia = i;
17          jb = j;
18          rom[{ia,jb}] = ia * jb;
19       end
20 end
21 // store data from inputs.
22 always @ (posedge clk)
23    begin
24       ta <= da;
25       tb <= db;
26    end
27 // read ROM and return the result
28 always @ (posedge clk)
29       q  <= rom[{ta, tb}];
30 endmodule
```

Часть lab7_1 продолжение описания

- Пример временной диаграммы для моделирования



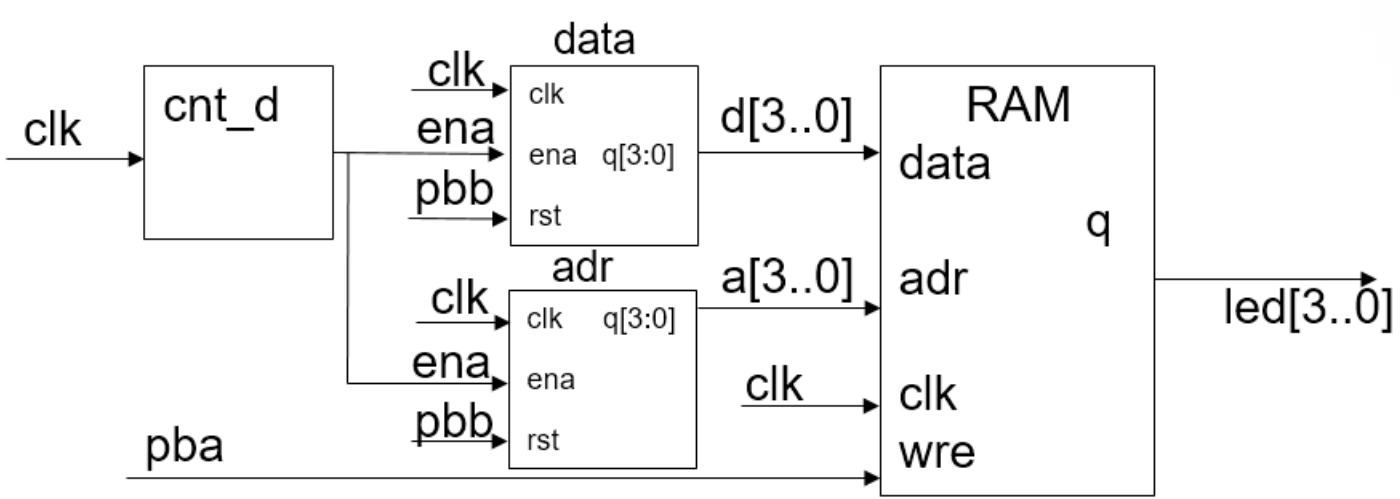
Часть lab7_1 продолжение описания

- Дополнительные требования:
  - ✓ стандарты и номера выводов СБИС для платы miniDiLaB_CIV задайте с помощью атрибутов.
- Осуществите функциональное моделирование.
- ~~Подготовьте конфигурационный файл для реализации на плате.~~

- Имя проекта – lab7_1. Имя модуля верхнего уровня – lab7_1

Часть lab7_2

- На языке Verilog создайте **структурное** описание устройства, приведенного на рисунке (*разрядность на рисунке показана при базовых значениях параметров data_W и word_num – см. следующий слайд*)



Часть lab7_2 продолжение описания

- В состав устройства входят:
- cnt_d - счетчик делитель (параметризированный, параметр – DIV)
  - ✓ счетчик-делитель, обеспечивает счет по модулю DIV (базовое значение - 3) и формирование синхронного сигнала переноса (активный уровень сигнала – 1, длительность один такт тактовой частоты) по достижению счетчиком значения DIV-1.
- Параметризированный счетчик cnt_N

  – двоичный счетчик на сложение с параметризированной разрядностью (параметр N, базовое значение - 4), имеющий вход тактовых сигналов (clk), вход разрешения работы (ena), вход асинхронного сброса (rst) и выход – q[N-1:0]
- data – формирователь данных для модуля памяти
  - ✓ реализован на базе параметризированного счетчика cnt_N
- adr – формирователь адреса для модуля памяти
  - ✓ реализован на базе параметризированного счетчика cnt_N
- RAM – модуль памяти (простая одно портовая память с **чтением старых данных** в процессе записи) параметризированный, параметры:
  - ✓ word_num – число слов – базовое значение 16,
  - ✓ data_W – разрядность данных – базовое значение 4

Часть lab7_2 продолжение описания

- RAM – модуль памяти
  - ✓ простая одно портовая память с чтением старых данных в процессе записи
  - ✓ параметризированный, параметры:
    - word_num – число слов – базовое значение 16,
      - *Для расчета разрядности шины адреса следует использовать функцию с постоянным значением для вычисления log2 word_num (пример был в лекции 6).*
    - data_W – разрядность данных – базовое значение 4
  - ✓ Вход wre – вход разрешения записи (=0 – запись в память разрешена)
  - ✓ Модуль RAM памяти должен быть инициализирован данными:
    - 0000 – четные адреса; 1111 – нечетные адреса
    - Инициализацию реализовать с помощью $readmemb или $readmemh

- Выводы устройства:
  - ✓ Входы:
    - ARSTn- вход асинхронного сброса (‘0’– сброс), при реализации на плате – кнопка pbb .
    - WEn– вход разрешения записи в память (‘0’– запись разрешена), при реализации на плате – кнопка pba.
    - clk – вход тактового сигнала (clk), при реализации на плате подается от тактового генератора 25МГц (см. описание стенда).
  - ✓ Выходы
    - [data_W-1:0] q, при реализации на плате – светодиоды led[3:0]. На выходах должен быть использован регистр, который по сигналу ARSTn = 0 устанавливается, во всех разрядах, в 1.

Часть lab7_2 продолжение описания

- Алгоритм работы устройства:
  - ✓ Модуль памяти RAM инициализируется данными 4’hf – четные адреса; 4’h0 – нечетные адреса.
  - ✓ Если ARSTn и WEn равны 0 (кнопки pba и pbb на плате не нажаты): осуществляется чтение данных из памяти. При реализации на плате данные отображаются на светодиодах: светодиоды переключаются (все включены-все выключены) ~ 1 раз в секунду
  - ✓ Если установить ARSTn =1 (на плате - нажать на кнопку pbb и удерживать ее), то счетчики адреса и данных сбрасываются в 0, выходной регистр данных устанавливается в 1 во всех разрядах и на выходах устройства во всех разрядах должны быть 1. (На плате – светодиоды должны быть выключены).
  - ✓ Если установить ARSTn =0 и WEn =1 (на плате нажать на кнопку pba) - разрешить запись данных в модуль памяти - в память по адресам, формируемым модулем adr, будут записываться данные формируемые модулем data. При этом на выходе устройства должны отображаться записываемые данные (модуль памяти реализует режим: чтение новых данных в процессе записи). При реализации на плате: для записи всей памяти потребуется около 16 секунд, записываемые данные будут отображаться на светодиодах.
  - ✓ Если дождаться записи во все 16 ячеек памяти, то в модуль памяти будут записаны значения от 0 до 15.
  - ✓ Если после записи данных в модуль памяти установить ARSTn =0 и WEn =0 (на плате кнопки pba и pbb не нажаты) - осуществляется чтение данных из RAM, значений от 0 до 15, и отображение их на выходах q. На плате: на светодиодах появляются значения от 0 до 15 с периодом равным ~ 1 секунде.

Часть lab7_2 продолжение описания

- Дополнительные требования:
  - ✓ стандарты и номера выводов СБИС для платы miniDiLaB_CIV задайте с помощью атрибутов.
- Осуществите функциональное моделирование при **DIV=3**, word_num = 16, data_W=4:
  - ✓ Режим 1: ARSTn =1 и WEn =1 (модуль памяти заполнен базовыми значениями 4’hf и 4’h0)
  - ✓ Режим 2: ARSTn =0 и WEn =1
  - ✓ Режим 3: ARSTn =1 и WEn =0 – следует дождаться заполнения всего модуля памяти новыми значениями
  - ✓ Режим 4: ARSTn =1 и WEn =1 (модуль памяти заполнен базовыми значениями от 0 до 15).
- Подготовьте конфигурационный файл для реализации на плате при **DIV=25 000 000**, word_num = 16, data_W=4
- Имя проекта – lab7_2. Имя модуля верхнего уровня – lab7_2

Часть lab7_3

- На языке Verilog создайте описание модуля, который, в зависимости от параметра dir, осуществляет преобразование двоичного 5-ти разрядного кода в:
  - ✓ Двоично-десятичный – при dir=“bd”
  - ✓ Код Грея – при dir=“grey”
  - ✓ Константное значение (равное номеру в списке группы) по адресу, равному номеру в списке группы (по остальным адресам – 0)– при любых других значениях dir.
- Преобразования должны быть реализованы на модуле памяти ROM. Начальные значения ROM для каждого преобразователя задать с помощью $readmemb или $readmemh
- Разрядность данных модуля ROM – 8 бит (две тетрады для двоично-десятичного преобразователя; Код Грея отображается в младших 5 разрядах, в старших – 0; константное значение отображается в 8 разрядах).
- Разрядность адреса модуля ROM – 5 бит (на вход адреса поступает преобразуемый двоичный код).
- Анализ параметра dir и формирование алгоритма работы модуля следует реализовать с помощью оператора **generate**.
  - ✓ *Например, можно с помощью generate case анализировать параметр dir и выбирать считываемый для заполнения ROM файл.*

Часть lab7_3 продолжение описания

- Выводы устройства:
  - ✓ Входы:
    - [4:0] bc – преобразуемый двоичный код (при реализации на плате подключен к sw[4:0])
    - clk – вход тактового сигнала (clk), при реализации на плате подается от тактового генератора 25МГц (см. описание стенда).
  - ✓ Выходы
    - [7:0] q (при реализации на плате подключены к светодиодам led[7:0]).

- Пример таблиц для файлов с начальным содержимым ROM

		содержимое ROM		
двоичный код	адрес ROM	2-10 код	Код Грея	Номер в списке (например для студента с номером 31)
00000	0	0000_0000	0000_0000	0000_0000
00001	1	0000_0001	0000_0001	0000_0000
00010	2	0000_0010	0000_0011	0000_0000
	3	…	…	0000_0000
	4	…	…	0000_0000
	5	…	…	0000_0000
	6	…	…	0000_0000
	7	…	…	0000_0000
	8	…	…	0000_0000
	9	…	…	0000_0000
01010	10	0001_0000	0000_1111	0000_0000
	11	…	…	0000_0000
	12	…	…	0000_0000
	13	…	…	0000_0000
	14	…	…	0000_0000
	…	…	…	0000_0000
11111	31	0011_0001	…	0001_1111

Часть lab7_3 продолжение описания

- Дополнительные требования:
  - ✓ стандарты и номера выводов СБИС для платы miniDiLaB_CIV задайте с помощью атрибутов.
- Осуществите функциональное моделирование (и приведите в отчете результаты) для всех трех вариантов реализации модуля.
- Подготовьте конфигурационный файл для реализации на плате
- Имя проекта – lab7_3. Имя модуля верхнего уровня – lab7_3