Спецкурс

Языки описания схем. Проблемы верификации.

Часть 1.

Лекция 16 марта 2010 г.

Корухова Юлия Станиславовна

Пример (сигналы и переменные)

Фрагмент 1

```
Y<=A – B * C;
 Z<=A + B * C;
```

Фрагмент 2

```
 V:=  B * C;
 Y<= A – V;
 Z<= A + V;
```

Фрагмент3

```
 V<= B * C;
 Y<=A – V;
 Z<= A + V;
```

Операции языка VHDL (продолжение)

mod rem - остаток от деления

В стандарте VHDL определены следующим образом

L rem R должно выполняться соотношение L = (L/R) *R + (L rem R)
где L/R — целая часть частного
(L rem R) — остаток от деления, имеет знак операнда L

L mod R должно выполняться соотношение L = N *R + (L mod R)
где N - целое
(L mod R) — остаток от деления, имеет знак операнда R

Примеры:

13 rem 5 = 3	**13 mod 5 = 3**
13 rem (-5) =3	**13 mod (–5) = –2**
(–13) rem 5 = – 3	**(–13) mod 5 = 2**
(–13) rem (–5) = – 3	**(–13) mod (–5) = –3**

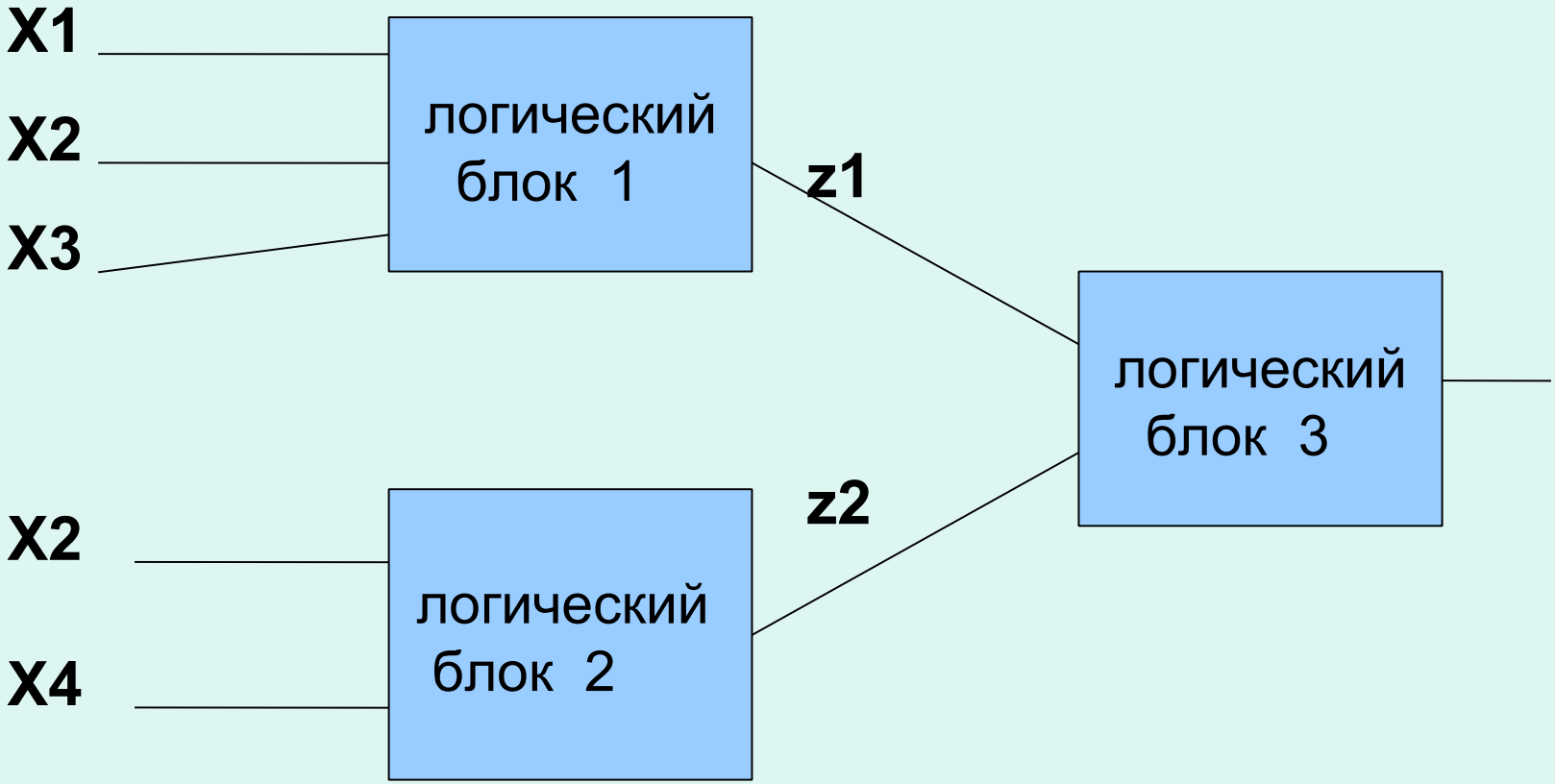
Для векторов: результирующий вектор для операций rem и mod имеет размерность делителя.

Операции языка VHDL (продолжение)

Особенность операции abs и унарный –

```
 signal  a_s: SIGNED (3 downto 0) := "1000" ;   – – число - 8
 signal b_s,  c_s : SIGNED (3 downto 0);
b_s <= – (a_s); – –  результат -8, т.к. 8="01000" не поместилось в вектор
c_s <=abs(a_s); – – результат -8, т.к. 8="01000" не поместилось в вектор
```

Сигналы в VHDL



Операторы process, wait

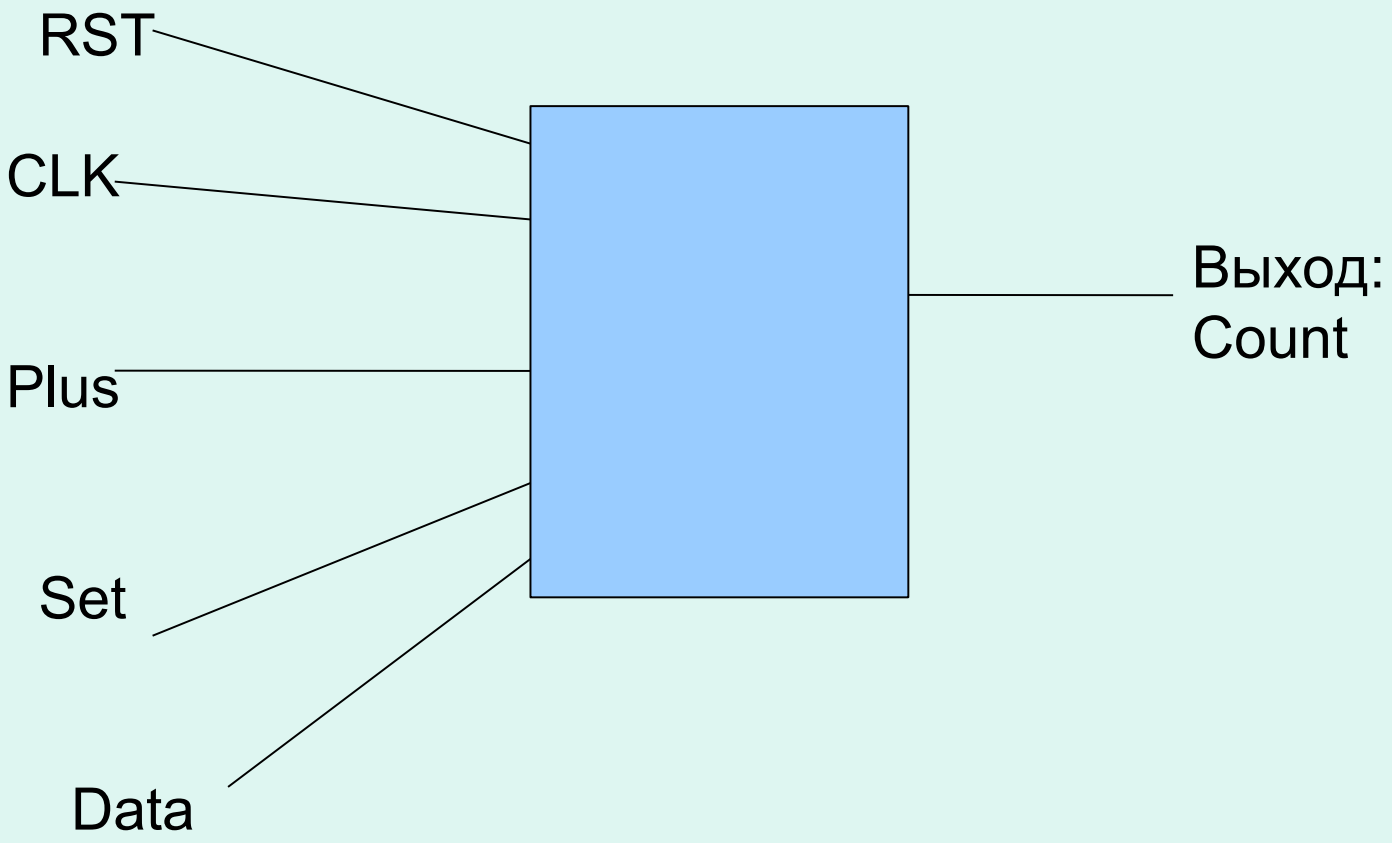
```
process (A,B,C)
begin
/*тело процесса*/
end process;

process
begin
/* тело процесса, содержащее wait */
end;
```

wait until *условие*;
wait on *список сигналов*;
wait for *время*;
wait;

Пример: 5-разрядный счетчик

Входы:



Пример программы: 5-разрядный счетчик

```
entity count5 is
    port (RST, CLK, Plus, Set :in bit;
          Data: in integer range 0 to 31;
          count: out integer range 0 to 31);
end count5;
architecture c5  of count5 is
    signal val:integer range 0 to 31;
    begin
      p1: process(RST,CLK)
      begin
        if (RST='1') then val<=0;
           elsif (CLK'event and CLK='1') then
            if (Set='1') then val<=Data;
             elsif (Plus='1')then
                if (val=31) then val<=0;
                    else val<=val+1;
               end if;
           end if;
        end if;
  end process p1;
```

```
p2:process (val)
      begin
        count<=val;
      end process p2;

end c5;
```

Операторы report и assert

report строка; – – выдача сообщения

report строка [severity NOTE | WARNING | ERROR | FAILURE];

Примеры:

report "Str1";
report "Str"& "1"; – – & - оператор конкатенации строк

assert условие [report сообщение] [severity
NOTE | WARNING | ERROR | FAILURE]
проверка истинности условия и, если оно ложно, выдается сообщение

Пример:

assert X>=0 report "X is negative" severity WARNING;

Операторы цикла

exit [метка цикла] [when условие]; – – оператор выхода из цикла

next [метка цикла] [when условие]; – – завершение итерации цикла

1.

```
 [имя:] loop
    последовательность операторов
  end loop [имя];
```

Пример:

```
i:=1;
C1:loop
        X(i):=X(i+1);
         i:=i+1;
     exit C1 when i>10;
  end loop C1;
```

Операторы цикла (прод.)

2.

[имя:] for индекс in интервал loop
последовательность операторов
end loop [имя];

Пример:

```
C2:for i in 1 to 10 loop
       X(i):=X(i+1);
     end loop C2;
```

Операторы цикла (прод.)

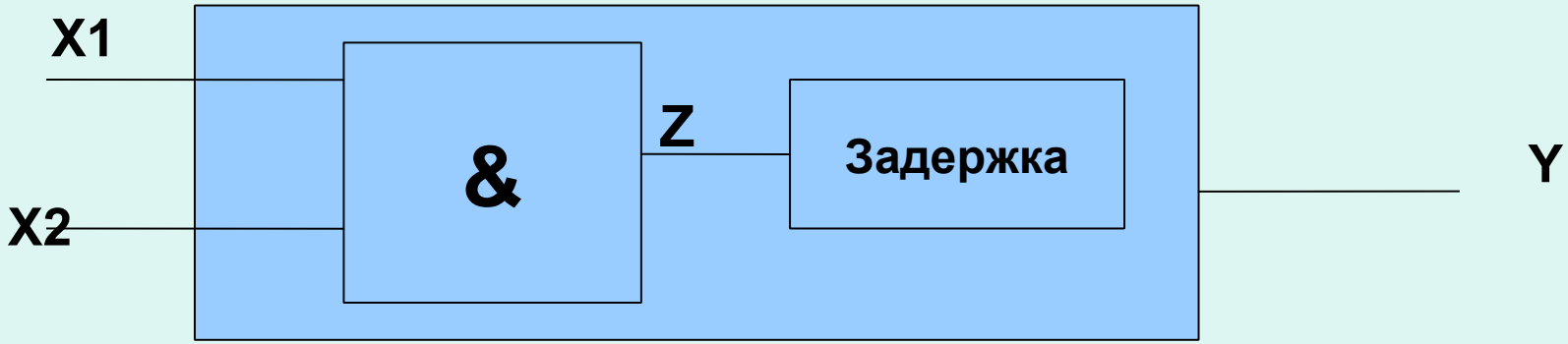
3.

[имя:] while условие loop
последовательность операторов
end loop [имя];

Пример:

```
 i:=1;
C3: while   i<=10  loop
     X(i):=X(i+1); i:=i+1;
  end loop C3;
```

Присваивание значений сигналам



Инерционная задержка

X<= inertial Y after 3 ns;
X<= Y after 3 ns;

Транспортная задержка

X <=transport Y after 3 ns;

Разрешающая функция

Табличное задание разрешающей функции для сигналов (std_logic) из двух источников

	'U'	'X'	'0'	'1'	'Z'	'W'	'L'	'H'	'-'
'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'	'U'
'X'	'U'	'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X'
'0'	'U'	'X'	'0'	'X'	'0'	'0'	'0'	'0'	'X'
'1'	'U'	'X'	'X'	'1'	'1'	'1'	'1'	'1'	'X'
'Z'	'U'	'X'	'0'	'1'	'Z'	'W'	'L'	'H'	'X'
'W'	'U'	'X'	'0'	'1'	'W'	'W'	'W'	'W'	'X'
'L'	'U'	'X'	'0'	'1'	'L'	'W'	'L'	'W'	'X'
'H'	'U'	'X'	'0'	'1'	'H'	'W'	'W'	'H'	'X'
'-'	'U'	'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X'

Подпрограммы в VHDL: функции

Подпрограммы: процедуры и функции

Декларация функции:

[**pure** | **impure**] **function** имя_функции (параметр {, параметр})
  return тип_возвращаемого_значения **is**
  раздел_деклараций
begin
  тело_функции
end [имя функции];

Вызов функции

имя_функции(фактический параметр {,фактический параметр});

Функция имеет только входные параметры

Примеры функций

Функция преобразования логического типа в STD_ULOGIC

```
 function bool_to_sl ( X:boolean )  return STD_ULOGIC    is
begin
  if x then  return '1';
  else return '0';
end bool_to_sl;
```

Вызов функции: a = bool_to_sl (false);

Функции преобразования типов из пакета std_logic_1164:

to_Bit : std_ulogic -----> bit
to_BitVector : std_logic_vector---->bit_vector
to_BitVector : std_ulogic_vector---->bit_vector
to_StdULogic : bit -----> std_ulogic
to_StdLogicVector: bit_vector ------>std_logic_ vector
to_StdULogicVector: bit_vector ------>std_ulogic_ vector
to_StdLogicVector : std_ulogic -----> std_logic_vector
to_StdLogicVector: std_logic ------>std_ulogic_ vector

Подпрограммы в VHDL: процедуры

Декларация процедуры:

procedure имя_процедуры (параметр {, параметр}) **is**
  раздел_деклараций
begin
  тело_процедуры
end [имя процедуры];

Процедура может иметь входные (in), выходные (out) и параметры inout используемые как входные и как выходные.

Вызов процедуры

имя_процедуры(фактический параметр {,фактический параметр});

Пример процедуры VHDL

```
procedure example (signal X : in std_ulogic_vector,
                                  signal Y: out std_ulogic )  is
   variable TMP : std_ulogic :='1';
begin
   for i in X'range loop
    TMP :=TMP and  X(i);
   end loop;
   Y<=TMP;
end  example;
```

Пример программы на VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

– – Car Alarm
entity c is
    Port ( key_in : in  STD_LOGIC;
           door : in  STD_LOGIC;
           sbelt : in  STD_LOGIC;
           warning : out  STD_LOGIC);
end c;

architecture Behavioral of c is
begin
  warning<= key_in and (not door or not sbelt);
end Behavioral;
```

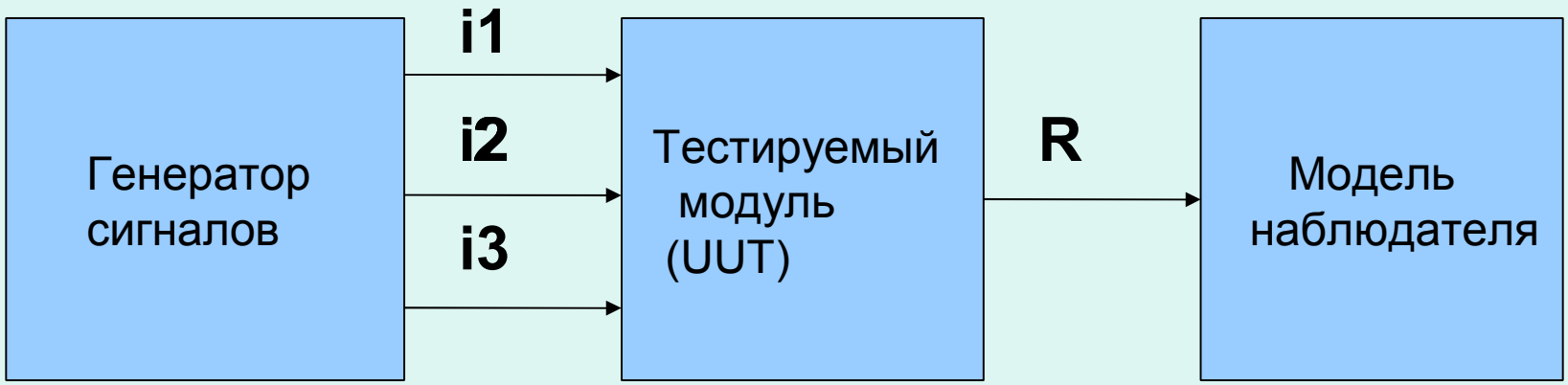
Тестирующие программы (testbench)

Верификация проекта — процесс доказательства того, что он функционирует согласно спецификации

Верификация

имитационная

формальная



Построение тестирующего модуля

```
ENTITY carTest IS
END carTest;

ARCHITECTURE ctest OF carTest IS
   COMPONENT c
      PORT(   key_in : IN  std_logic;      door : IN  std_logic;
              sbelt : IN  std_logic;    warning : OUT  std_logic
       );
    END COMPONENT;
    --Inputs
  signal i1,i2,i3 : std_logic := '0';
   --Outputs
  signal R : std_logic;

BEGIN
```

Построение тестирующего модуля

key_in (i1)	door (i2)	sbelt (i3)	warning (R)
1	0	0	1
1	1	0	1
1	0	1	1
1	1	1	0
0	любое	любое	0

Построение тестирующего модуля

(продолжение)

```
BEGIN
 uut: c PORT MAP (
          i1 ,  i2,  i3,  R ;
        );

GEN: process  – – генератор сигналов
   begin
      i1<='0';i2<='0';i3<='0';
      wait for 10 ns;
      i2<='1',
      wait for 10 ns;
.............................
      wait;
end GEN;
```

Построение тестирующего модуля
(продолжение)

```
– – для типа LINE и вывода : STD_TEXTIO.ALL

WRITER process (R)
  variable L:LINE;
    begin
       write (L, NOW); – – запись в строке модельного времени
       write (L,i1);
       write (L,i2);
       write (L,i3);
       writeline (output, L); – – строка L печатается в стандартный вывод
end process WRITER;

END ctest;
```

Стратегия функциональной верификации

Тестирование объекта:

– «черный» ящик

– «прозрачный» (белый) ящик

– «серый» ящик

Задания

1. Предложить вариант тестирующей программы на VHDL для 5-разрядного счетчика

2.Запрограммировать цифровое устройство реализующее систему лог. функций, заданных с помощью таблицы истинности. Привести VHDL- модель и тестирующую программу для всех наборов входных параметров.

x1 x2 x3 x4	y1 y2 y3
0 0 0 0	0 0 0
0 0 0 1	0 0 1
0 0 1 0	0 0 0
0 0 1 1	0 0 0
0 1 0 0	0 0 1
0 1 0 1	0 1 0
0 1 1 0	0 1 1
0 1 1 1	1 0 0

x1 x2 x3 x4 (продолжение)	y1 y2 y3
1 0 0 0	0 0 0
1 0 0 1	0 1 1
1 0 1 0	1 0 0
1 0 1 1	1 0 1
1 1 0 0	0 1 1
1 1 0 1	1 0 0
1 1 1 0	1 0 1
1 1 1 1	1 0 0