Спецкурс

Языки описания схем. Проблемы верификации.

Часть 1.

Лекция 2 марта 2010 г.

Корухова Юлия Станиславовна

Этапы проектирования с использованием HDL

1. Разработка иерархической блок-схемы проекта
2. Программирование
3. Компиляция
4. Моделирование
5. Синтез
6. Компоновка, монтаж, разводка
7. Временной анализ

Проектирование в WebPack ISE

www.xilinx.com

Литература:

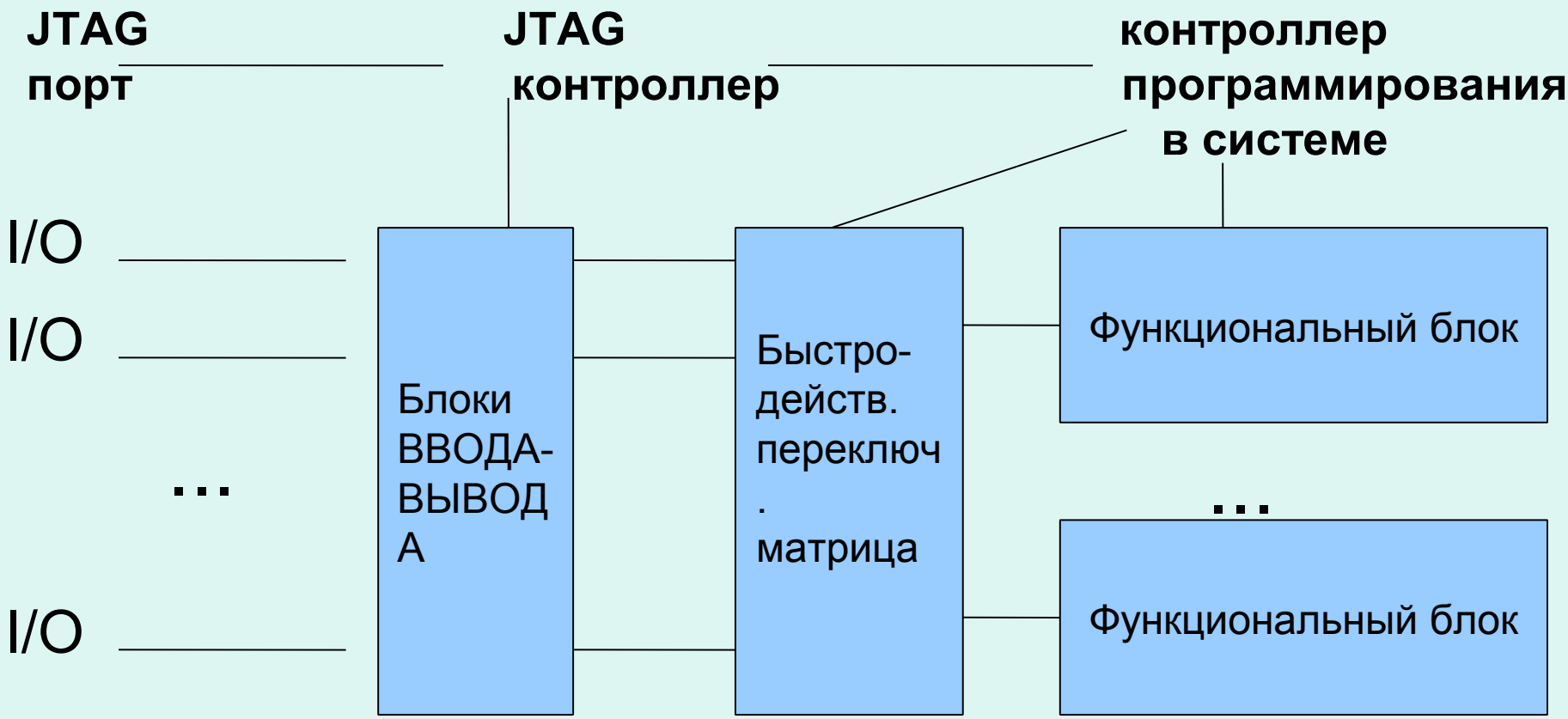
Зотов В.Ю. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPack ISE
М.: Горячая линия — Телеком., 2003

ПЛИС (программируемая логическая интегральная схема) — электронный компонент, используемый для создания цифровых устройств. Логика ПЛИС не определяется при изготовлении, а задается как программа.

ПЛИС фирмы XILINX
с архитектурой FPGA (Field Programmable Gate Array)
CPLD (Complex Programmable Logic Device)

Пример архитектуры ПЛИС

Семейство XC9500 CPLD



Проектирование в WebPack ISE 1

1. Создание проекта
Задается имя, описание.
Выбираем семейство ПЛИС, а также средств синтеза.

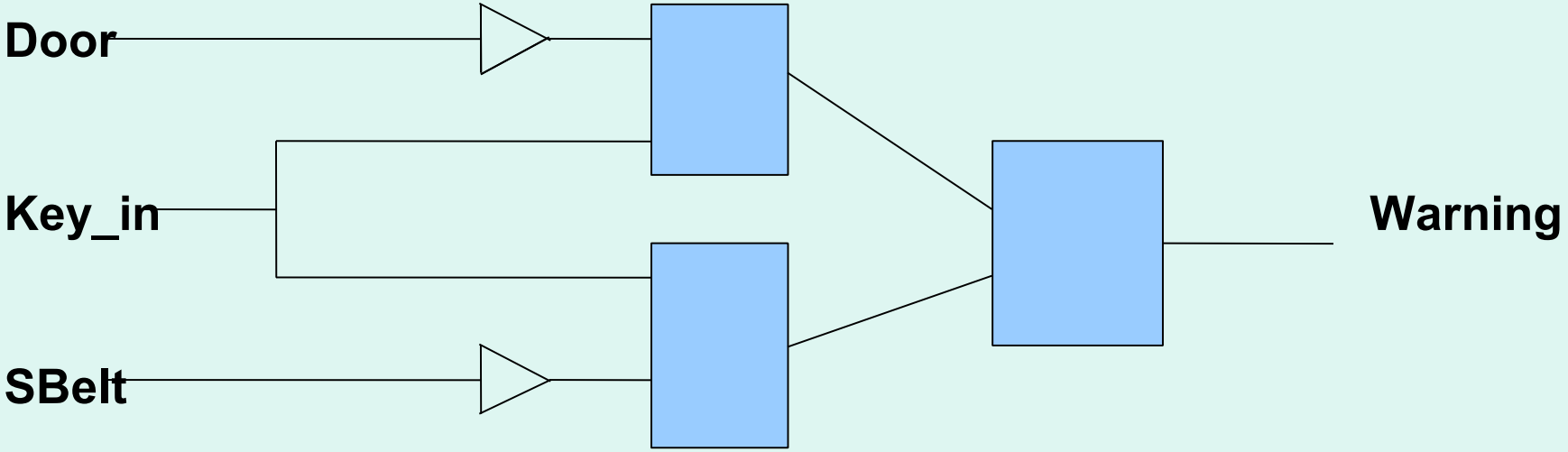
ПРИМЕР

ЗАДАЧА
Запрограммировать цифровое устройство ALARM,
которое выдает сигнал предупреждения (Warning) всякий раз, когда автомобильный ключ вставлен в замок зажигания (Key_in), а при этом открыта любая дверь (Door) или
отстегнут ремень безопасности водителя (Sbelt1).

ОПИСАНИЕ ПОВЕДЕНИЯ СИСТЕМЫ
Warning = Key_in and (not Door or not SBelt1)

Проектирование устройства

1. Структурное описание устойства —
в виде набора логических элементов и компонентов,
которые связаны таким образом, что выполняют нужную функцию.



Проектирование в WebPack ISE

2. Описание на поведенческом уровне

в виде потока данных описываются межрегистровые пересылки

в алгоритмическом виде с помощью операторов

Создаем модуль на языке VHDL

entity

architecture

ПРИМЕР

Проектирование в WebPack ISE 4

Добавлено средой разработки:

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

Описание типа (перечислимого)

```
type STD_ULOGIC is (
 'U', --неинициализированный сигнал
 'X', --неизвестное значение сигнала (сильная неопределенность)
  '0', --сильный 0
  '1', --сильная 1
  'Z', --высокий импеданс
  'W',--слабая неопределенность
  'L', --слабый 0
  'H', --слабая 1
  '~'); --безразлично
```

Логические операции для сигналов

Аргументы	AND
0 любой	0
любой 0	0
1 1	1
в ост. случаях	X

NOT
not(1)=0
not(0)=1
not(X)=X
not(Z)=X

Аргументы	OR
0 0	0
1 любой	1
любой 1	1
в ост. случаях	X

Пример: U01XWLHZ **AND** UU0UXLXZ

Лексические элементы VHDL

Идентификаторы

Ключевые и заразервированные слова

Расширенные идентификаторы \name#1\ \if\ \id\\1\

Числа

Integer 12 123E3 12e+2

real 1.2 1.8 E-2

Базированные литералы

база#число# 2#10010# 8#22# 16#12#

2#1001_1101_1100_0010#

Символы 'a' 'B' ','

Строки "string"

Битовые строки — последовательность значений, представленных в виде набора битов.

B"1100_1001" b"11001001" O"311" o"311" X"4B" x"c9"

Объявление элементов данных

Константы

```
constant SIZE: integer :=16;
constant delay2, delayX: time  :=2 ns;
```

Переменные

```
variable var1: boolean  :=false;
Variable Sum: integer  := range 0 to 256  :=16;
```

Модификация значения переменной

```
Sum:=0;
```

Сигналы

```
signal CLOCK: bit  :=0 ;
signal Value: integer range 0 to 100;
```

Модификация значения сигнала

```
 CLOCK <= 1 after 1 ns;
 Value <= '0', '1' after 5 ns, '0' after 10 ns, '1' after 20 ns;
```

Присваивание значений переменным и сигналам

```
architecture VAR of EXAMPLE is
  signal Inp, Res:  integer :=0;
begin
  process
    variable v1 : integer :=1;
    variable v2 : integer :=2;
    variable v3 : integer :=3;
  begin
    wait on Inp;
    v1 := v2;
    v2 := v1+v3;
    v3 := v2;
    Res <=v1+v2+v3;
  end process;
end VAR;
```

```
architecture SIGN of EXAMPLE is
  signal Inp, Res:  integer :=0;
  signal s1 : integer :=1;
  signal s2 : integer :=2;
  signal s3 : integer :=3;
begin
  process
    begin
      wait on Inp;
      s1 <= s2;
      s2 <= s1+s3;
      s3 <= s2;
    Res <=s1+s2+s3;
  end process;
end VAR;
```

Типы данных

```
Library std, work;
use std.standard.all;
```

тип	значения	пример
bit	0 1	signal A: bit :=1;
bit_vector	массив	signal In : bit_vector (7 downto 0)
boolean	true false	variable tst: boolean :=false;
character	символы	variable ch : character :='q';
integer	целые	variable i :integer :=123;
natural	цел. >=0	variable n : natural := 1;
positive	цел. >0	variable p : positive :=2;
real	веществ.	variable r : real :=54.2E-1;
string	строка	variable s : string(1 to 3) :="cat";
time	цел. + ед.	variable d :time :=5 ns;

Пользовательские типы данных

1. Перечислимый
Перечисляется список литер или идентификаторов
type имя is (список идентификаторов или символов)

2. Массив
type имя is array (индексы) of тип_элемента
type var is array (0 to 15) of integer

Открытый массив
type имя is array(тип range <>) of тип_элемента

3.Записи

```
type имя is record
  идентификатор: тип;
  идентификатор: тип;
  …
  идентификатор: тип;
end record;
```

Атрибуты

Специальные функции, позволяющие получить характеристики объектов в программе.

Атрибуты сигналов

Sign'Event (true если на сигнале Sign произошло событие, иначе — false)
Sign'Active (произошла ли установка сигнала)
Sign'quiet(T) (на сигнале в течение интервала T не было событий)

Скалярные атрибуты
Позволяют получить характеристики скалярных типов
my_type'left (крайнее левое значение диапазона)

Атрибуты массивов
Arr'high (правая верхняя граница индекса)
Arr'high(1) (правая верхняя граница первого индекса многомерного массива)

Операции

and or nand nor xor xnor

= /= < > <= >=

sll srl sla sra rol ror

+ - &(конкатенация векторов)

унарные + -

* / mod rem

** abs not