

# Языки описания схем

(mk.cs.msu.ru → Лекционные курсы → Языки описания схем)

## Блок 13

Verilog:  
синтезируемость кода

Лектор:  
**Подымов Владислав Васильевич**

E-mail:  
**valdus@yandex.ru**

## Вступление

Не всё, что написано на языке Verilog,  
имеет аппаратную семантику

## Вступление

```
module M(input x, output y);  
    assign y = !x;  
endmodule
```

```
module test();  
    reg x;  
    M testee(.x(x));  
    initial begin  
        #1 x = 0; #1 x = 1; $finish;  
    end  
endmodule
```

Модуль М имеет аппаратную семантику:

это схема с одноразрядным входом  $x$  и одноразрядным выходом  $y$ , эквивалентная комбинационной схеме из одного элемента НЕ

А схемы, выполняющейся в точности так, как записано в модуле `test`, не существует

## У: синтезируемость кода

С точки зрения синтеза схем по коду, элементы синтаксиса У делятся на поддерживаемые, игнорируемые и неподдерживаемые

**Поддерживаемые** элементы имеют аппаратную семантику и используются для синтеза схем

**Игнорируемые** элементы могут встречаться в записи схем, но в аппаратной семантике полностью игнорируются

**Неподдерживаемые** элементы не имеют аппаратной семантики:

- ▶ Если такой элемент записан внутри игнорируемого элемента, то он игнорируется в аппаратной семантике
- ▶ Иначе код схемы считается **несинтезируемым**: аппаратная семантика кода не определена

## У: примеры поддерживаемых конструкций

Всё, что обсуждалось в *блоке 11*, имело аппаратную семантику:

- ▶ объявления точек и шин точек типов `wire`, `reg`
- ▶ комбинационные выражения  
(*в том виде, как они обсуждались в блоке 11*)
- ▶ непрерывные присваивания
- ▶ входы и выходы, модули и их экземпляры

Все эти языковые конструкции являются **поддерживаемыми**

## У: примеры игнорируемых конструкций

Большая часть конструкций, обсуждавшихся в *блоке 12*, предназначена **исключительно** для программной симуляции:

- ▶ все команды, начинавшиеся с “\$”
- ▶ задержки
- ▶ начальная процедура
  - ▶ (кроме одного исключительного случая, не обсуждаемого в курсе: инициализация двумерной памяти, доступной только для чтения (ROM))

Все эти конструкции являются **игнорируемыми**: с точки зрения аппаратной семантики, их не существует в коде

## У: примеры неподдерживаемых конструкций

Некоторые конструкции У являются безусловно неподдерживаемыми

В *блоке 12* встретились две таких конструкции: *\$time* и *\$stime*

Если хочется сохранить синтезируемость кода,  
то следует использовать такие конструкции **только** внутри  
игнорируемых

Некоторые конструкции У могут расцениваться как поддерживаемые,  
так и неподдерживаемые в зависимости от способа их использования

Основная такая конструкция в *блоке 12* — **постоянная процедура**:

- ▶ в этом блоке обсуждался **неподдерживаемый** способ применения процедуры, пригодный только для программной симуляции
- ▶ соблюдением строгих синтаксических ограничений и использованием дополнительных “ограничивающих” конструкций можно добиться того, что постоянной процедурой будет описываться поведение известных схемных элементов — в этом случае постоянная процедура оказывается **поддерживаемой**