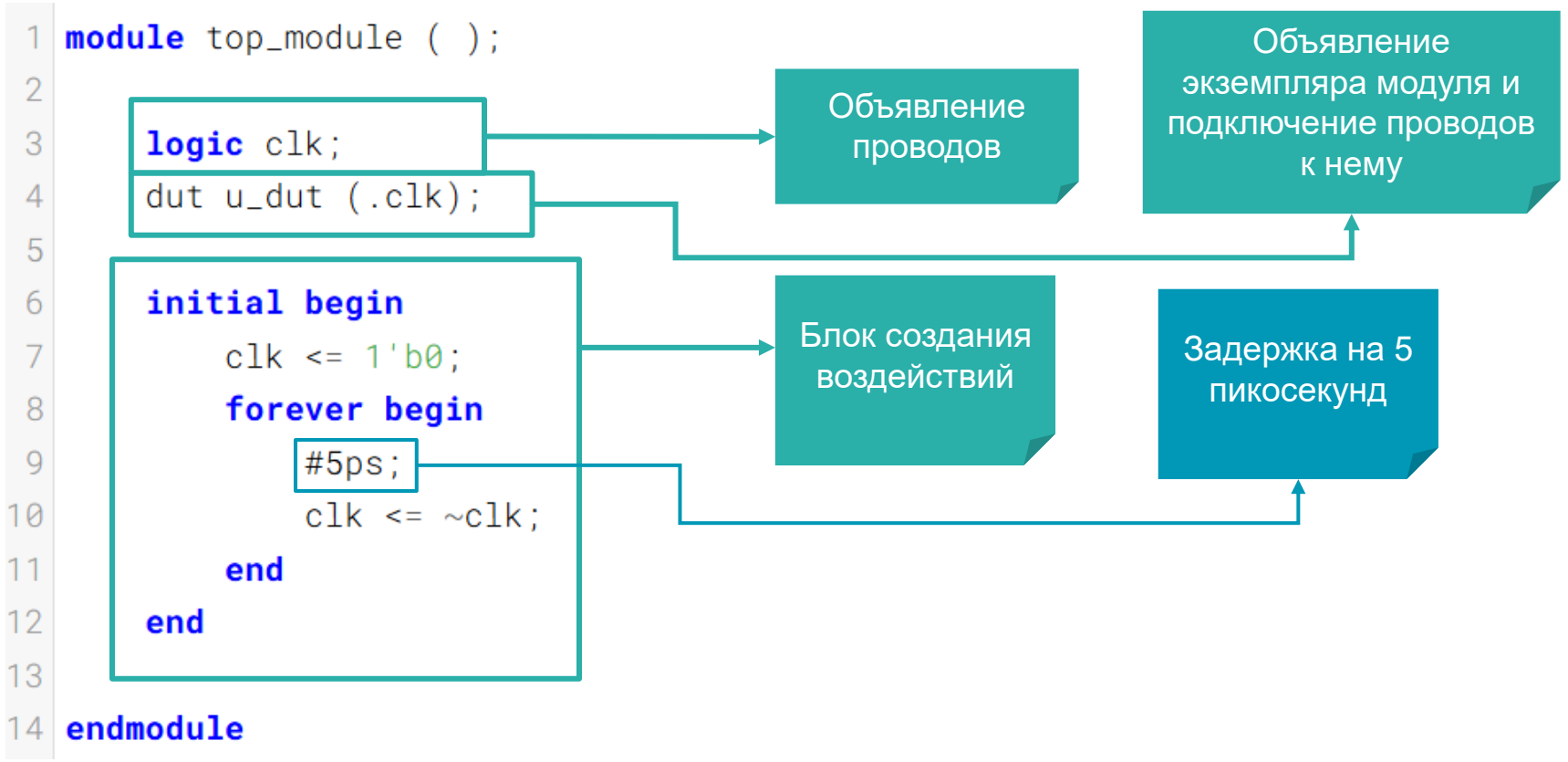
AQUARIUS

Лабораторная 1. Симуляция комбинационных схем.

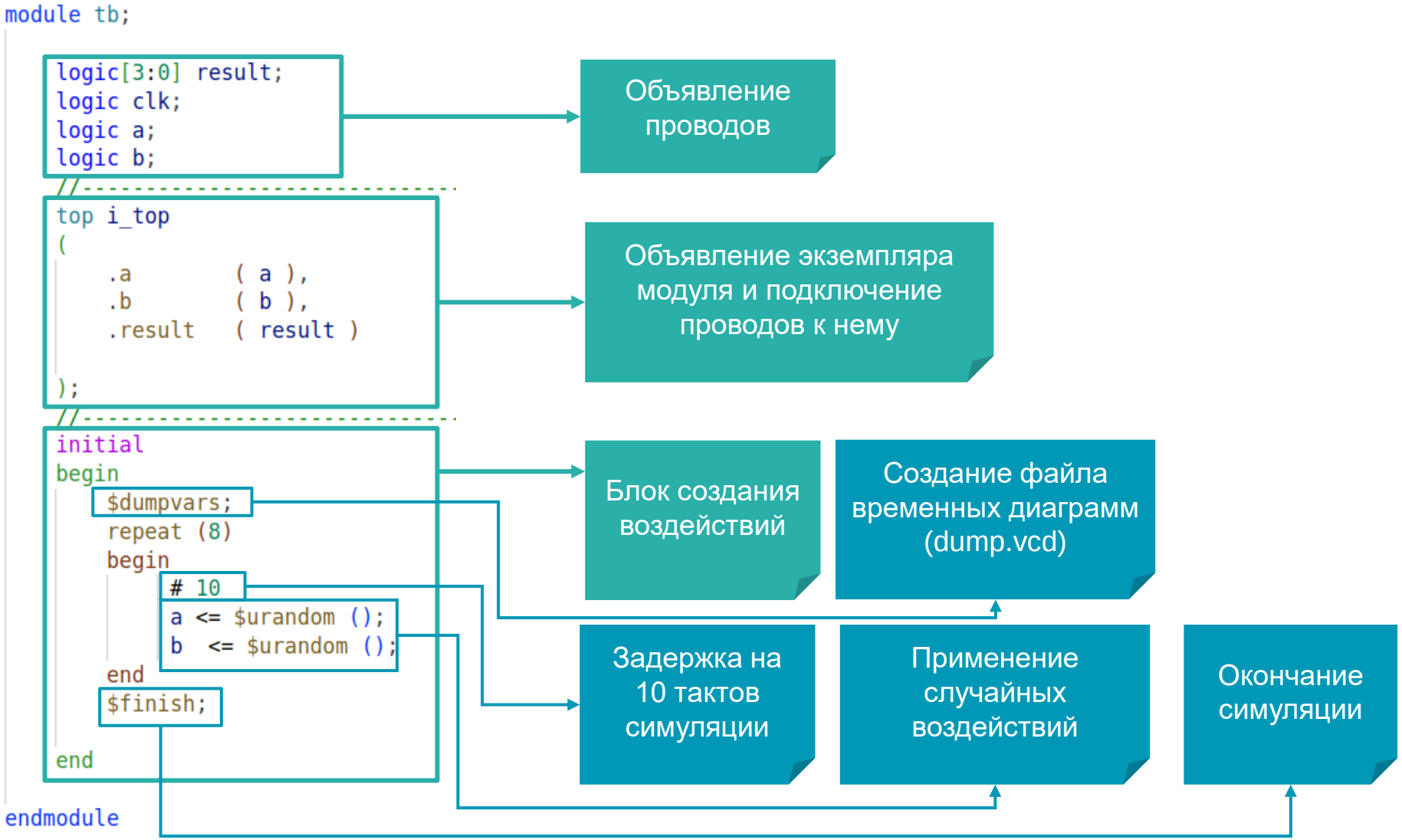
Выполнение упражнений hdlbits

Тестовые окружения

https://hdlbits.01xz.net/wiki/Tb/clock



Тестовое окружение



UNIX – системы

Семейство переносимых, **многозадачных** и **многопользовательских** операционных систем

- Каждая задача выполняется отдельной утилитой
- Взаимодействие осуществляется через единую файловую систему
- Для работы с утилитами используется командная оболочка.

UNIX – системы

Семейство переносимых, **многозадачных** и **многопользовательских** операционных систем

- Каждая задача выполняется отдельной утилитой
- Взаимодействие осуществляется через единую файловую систему
- Для работы с утилитами используется командная оболочка.

Windows – «тостер»

Windows – коммерческая пользовательская персональная система.
Концепция — максимально облегчить вхождение пользователя в задачу.

Плюсы	Минусы
Легкость вхождения для неподготовленного пользователя	Сложность автоматизации и объединения программ
Интерфейс с меню и иконками интуитивно понятен	Программы часто коммерческие и используют закрытые форматы
Быстрый старт без необходимости изучения командной строки	Ограниченная гибкость: выполняется только то, что предусмотрено производителем ПО
Подходит для выполнения стандартных, типовых задач	Пользователь не мотивирован изучать и осваивать глубже
	Часто нарушаются стандарты: пробелы вместо форматирования, слабая безопасность, вирусы

UNIX – toolbox

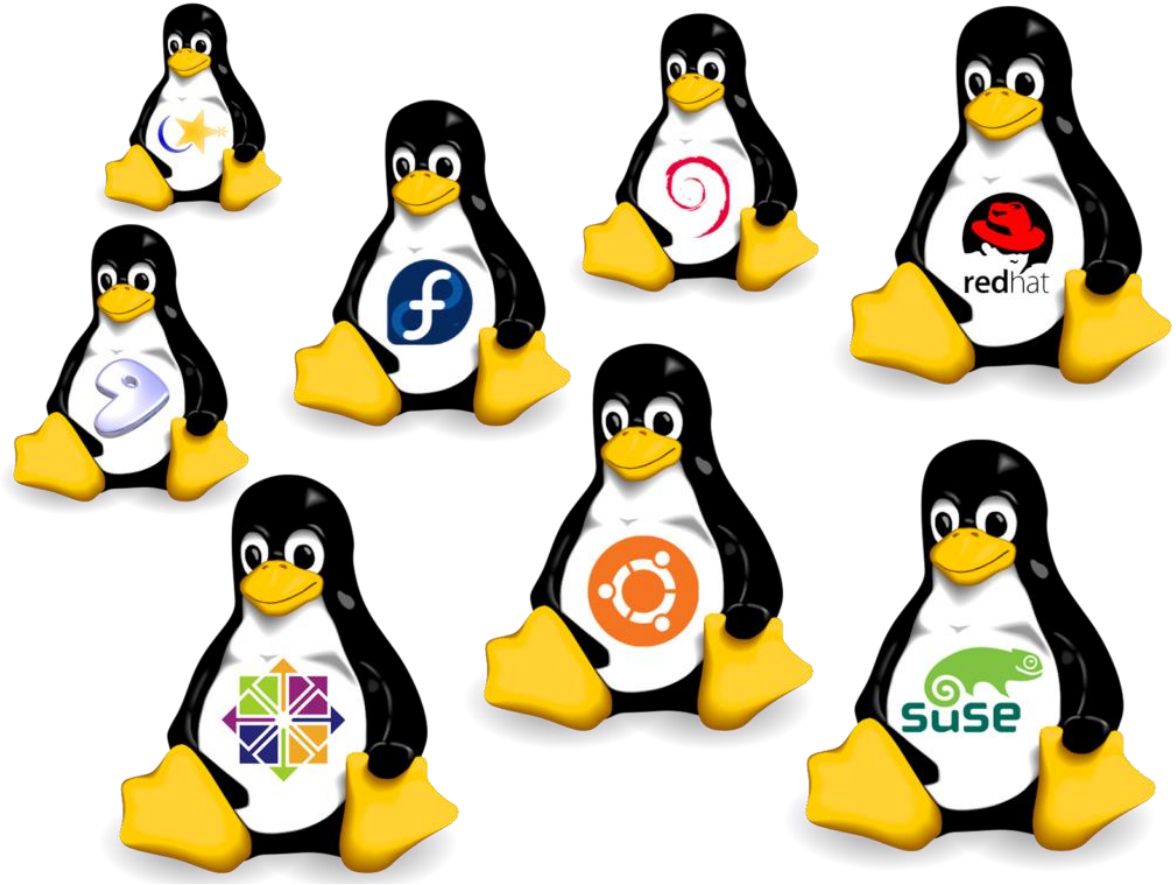
UNIX системы разрабатывались инженерами для инженеров и представляли собой «коробку с инструментами»

Плюсы	Минусы
Концепция "toolbox": каждая утилита делает одно дело, но хорошо	Нужно самому комбинировать утилиты для выполнения задач
Легко создавать цепочки утилит (автоматизация рутинных операций)	Высокий порог входа
Использование текстовых файлов упрощает обмен данными	Могут возникать проблемы с производительностью при больших объемах
"Pipes" (цепочки) позволяют удобно передавать данные между программами	Отсутствие универсальных решений требует больше ручной настройки
Гибкость и мощность в руках опытного пользователя	Ошибки в одной утилите могут сломать всю цепочку

UNIX - подобная система Linux

«В мире без стен – окна не нужны»

- ОС полностью бесплатная сс момента основания.
- Linux имеет полностью открытый исходный код.
- На Windows или Mac возможно создавать программу
- На Linux - возможно создавать свои ОС



Терминал

Историческая справка

Компьютерный терминал, **оконечное устройство** (**дисплейная станция**, **дисплей-консоль**, **консоль оператора**, **пульт оператора**) — устройство для взаимодействия пользователя с компьютером или компьютерной системой, локальной или удалённой-

Концепции

- **Linux – всё есть файл.**

«всё» действительно означает всё:
жесткий диск, раздел на жестком диске, параллельный порт,
подключение к веб-сайту, карта Ethernet, даже сами каталоги.

- **Bash – все входные данные – строки.**

Строки – последовательности символов (байтов).

Термины

Корень файловой системы – это начальная точка иерархической структуры файлов и папок в операционной системе.

Это самый верхний уровень, от которого разветвляются все остальные каталоги и файлы.

В Linux корневой каталог обозначается символом «/».

Домашний каталог – это личный каталог пользователя в операционной системе, где находятся его данные, настройки и т. д.

В Linux домашний каталог обозначается символом «~».

Директория = каталог = папка

Термины

Абсолютный путь – полный и точный путь к файлу или каталогу из корневого каталога системы.

```
/home/user/pictures/funny_picture.png
```

Относительный путь – расположение файла или каталога относительно текущего рабочего каталога. Символ «./» означает текущий каталог.

```
// Для рабочего каталога /home/user/pictures/:
funny_picture.png
./funny_picture.png

// Для рабочего каталога /home/user:
./pictures/funny_picture.png
```

Основные команды

- pwd – print working directory – вывести рабочую директорию
- ls <путь> - вывести содержимое каталога
- cd <путь> - перейти в указанную директорию
- mkdir <имя> – создание директории
- touch <имя> – создать пустой файл
- cp <путь/имя_что> <путь/имя_куда> - скопировать файл в указанную директорию под указанным именем
- mv <путь/имя_что> <путь/имя_куда> - переместить файл
- rm – удаление файла БЕЗ ВОЗМОЖНОСТИ ВОССТАНОВЛЕНИЯ
- rm -r – удаление директории БЕЗ ВОЗМОЖНОСТИ ВОССТАНОВЛЕНИЯ
- echo “Hello world!” – вывод сообщения в консоль
- cat <имя> - вывести содержимое файла в консоль
- grep – поиск внутри файлов
- find – поиск самих файлов или каталогов
- Используйте флаг **–h (--help)** для вызова окна помощи
- Используйте **man** <команда> для вызова руководства пользователя *(не работает в git bash windows)

Материалы, изучение команд

https://itproger.com/course/linux/4
https://itproger.com/course/linux/5
~~https://itproger.com/course/linux/6~~ - не работает в git bash windows
https://itproger.com/course/linux/7

Переменные в bash

- Переменная — это то макрос, который может быть подставлен в строку команды перед её разбором.
- Объявление: **`имя=значение`** `(без пробелов)`.
  - Для объявления в терминале используется команда export.
- Использование: **`$имя`** `или` **`${имя}`**.

```
User@DESKTOP-LURTA51 MINGW64 ~
$ export name="Alice"

User@DESKTOP-LURTA51 MINGW64 ~
$ echo $name
Alice
```

Переменная окружения PATH

PATH - переменная окружения (ОС), представляющая собой набор каталогов, в которых расположены исполняемые файлы.

```
User@DESKTOP-LURTA51 MINGW64 ~
$ echo $PATH
/c/Users/User/bin:/mingw64/bin:/usr/local/bin
:/usr/bin:/bin:/mingw64/bin:/usr/bin:/c/Users
/User/bin:/c/Program Files (x86)/VMware/VMwar
e Workstation/bin:/c/Windows/system32:/c/Wind
ows:/c/Windows/System32/Wbem:/c/Windows/Syste
m32/WindowsPowerShell/v1.0:/c/Windows/System3
2/OpenSSH:/c/Program Files/7-Zip:/cmd:/usr/bi
n/vendor_perl:/usr/bin/core_perl
```

Пример применения команд

```
User@DESKTOP-LURTA51 MINGW64 ~
$ pwd
/c/Users/User

User@DESKTOP-LURTA51 MINGW64 ~
$ ls ./Pictures/
Screenshots/  desktop.ini  funny_picture.png

User@DESKTOP-LURTA51 MINGW64 ~
$ cd ./Pictures/

User@DESKTOP-LURTA51 MINGW64 ~/Pictures
$ pwd
/c/Users/User/Pictures

User@DESKTOP-LURTA51 MINGW64 ~/Pictures
$ ls
Screenshots/  desktop.ini  funny_picture.png
```

Linux pipe

Linux

Pipe (конвеер) – это однонаправленный канал межпроцессного взаимодействия.

Конвейеры чаще всего используются в shell-скриптах для связи нескольких команд путем перенаправления вывода одной команды (stdout) на вход (stdin) последующей, используя символ конвеера ‘|’:

```
cmd1 | cmd2 | .... | cmdN
```

```
User@n-msk-16009136 MINGW64 ~
$ ls ./Pictures/
'Camera Roll'/   Screenshots/   desktop.ini   funny_picture.png   Фото/

User@n-msk-16009136 MINGW64 ~
$ ls ./Pictures/ | grep funny
funny_picture.png
```

Скрипты Bash Shell

Linux

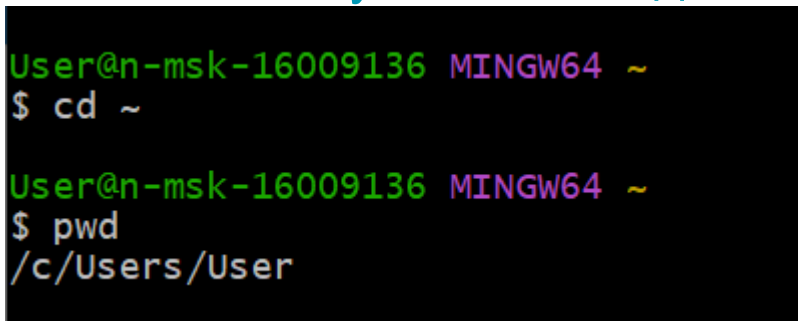
Bash скрипт – файл, содержащий список команд для выполнения в ОС Линукс.

При помощи Bash скрипта вы можете выполнять управление вашей операционной системой. Сам же Bash Shell является усовершенствованной вариацией командного терминала.

bash <name_of_script> – запуск скрипта

Подготовка к лабораторной работе

1. Выясните путь к вашей домашней директории:



2. Имя папки, отображаемой в проводнике может отличаться от имени в терминале (Пользователи = Users)

3. Скачайте папку **/basics-graphics-music-simple по ссылке** https://disk.yandex.ru/d/xFY7uhnfMr2CYw

4. Переместите скачанную папку в домашний каталог.

Выполнение лабораторной

Симуляция

1. Откройте терминал
2. Перейдите в директорию **cd ~/basics-graphics-music-simple/labs/01_and_or_not_xor_de_morgan/**
3. Для запуска VS code:
   а) введите в терминал **code**
   б) запустите VS code через иконку или из меню «пуск»
4. Откройте файл **top.sv** в VS code.
5. Выполните задание.
6. Сохраните файл сочетанием клавиш ctrl+s.
7. Запустите в терминале скрипт командой **bash 02_simulate_rtl.bash**.
8. Убедитесь в корректности выполнения задания.
9. Выполните лабораторные задания 01-04.

Клавиша **tab** позволяет автоматически дополнять имена файлов и команд

Например:
п.2: вводить путь не полностью, а частично и периодически нажимать tab несколько раз

п. 7: при запуске скрипта достаточно ввести:
“bash 02” и нажать **tab**

Особенность языка Verilog

Обратите внимание!

Код, кроме **assign** и объявления переменных, может исполняться только **внутри процедурных блоков.**

Вне процедурных блоков исполняемые части кода могут находиться в **task** и **function**, однако их исполнение происходит только в случае их вызова из процедурного блока.

```
module block (
    input in,
    output out
)

    logic a, b, c, d;

    a = 1;
    b = 0;

....

endmodule
```

```
module block (
    input in,
    output out
)

    logic a, b, c, d;

    assign a = 1;
    assign b = 0;

....

endmodule
```

```
module block (
    input in,
    output out
)

    logic a, b, c, d;

    initial begin
        a = 1;
        b = 0;
    end

....

endmodule
```

```
module block (
    input in,
    output out
)

    logic a, b, c, d;

    always @(*) //always_comb
    begin
        a = 1;
        b = 0;
    end

....

endmodule
```

```
module block (
    input in,
    output out
)

    logic a, b, c, d;

    task set_a_b(logic set_a,logic set_b);
        a = set_a;
        b = set_b;
        #10;
    endtask

....
    initial begin
        set_a_b(1,0);
    end
endmodule
```

Тестовое окружение