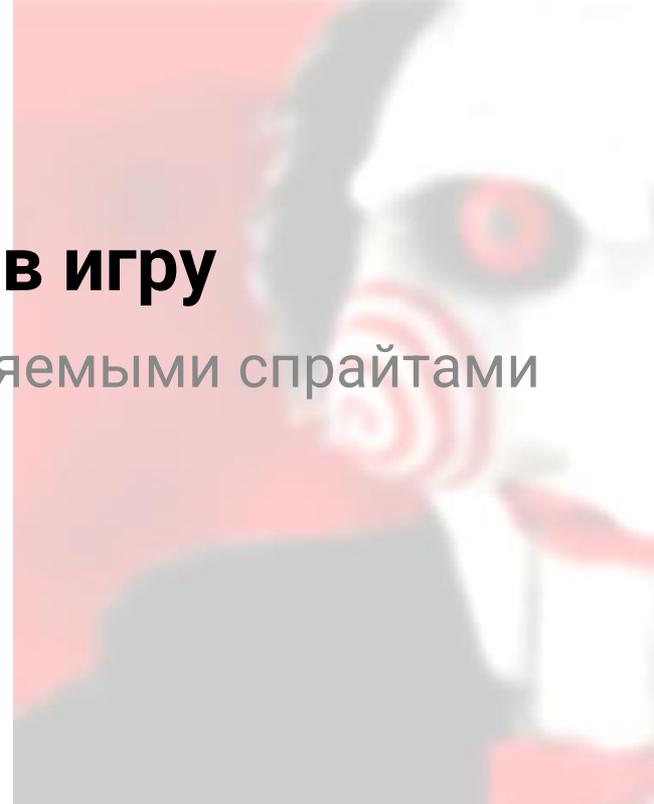


# FPGA-SYSTEMS.RU

сообщество FPGA разработчиков

## Я хочу сыграть с тобой в игру

Графическая игра с параллельно вычисляемыми спрайтами



# Где найти FPGA комьюнити?



[fpga-systems.ru](http://fpga-systems.ru)



[t.me/fpgasystems](https://t.me/fpgasystems)  $\Leftrightarrow$  [@fpgasystems](https://t.me/fpgasystems)



[youtube.com/c/fpgasystems](https://youtube.com/c/fpgasystems)



[admin@fpga-systems.ru](mailto:admin@fpga-systems.ru)



III КОНФЕРЕНЦИЯ FPGA РАЗРАБОТЧИКОВ

# FPGA-Systems 2021.2

Доступно в записи на Youtube

Конференция в Москве



Конференция в  
Санкт-Петербурге



# Приходи на следующую конференцию FPGA разработчиков

[fpga-systems.ru/meet](https://fpga-systems.ru/meet)

Москва

МИНСК

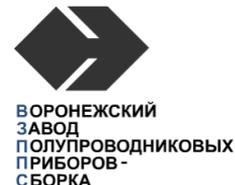
ТОМСК

Санкт-Петербург



# Не только Intel FPGA

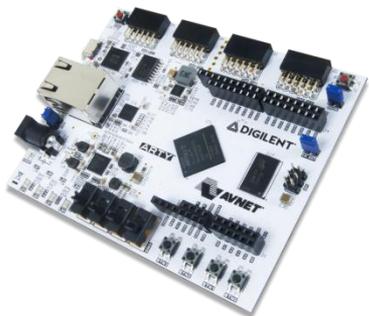
## Другие производители FPGA



<https://github.com/FPGA-Systems/fpga-awesome-list>

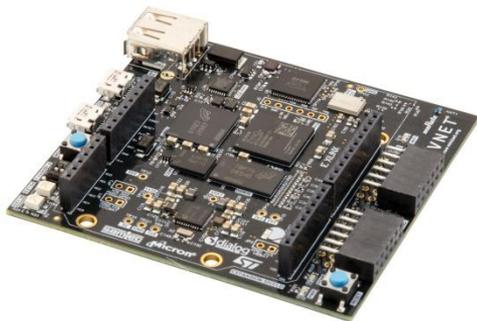
# Xilinx начального уровня

## Доступные платы с Xilinx FPGA



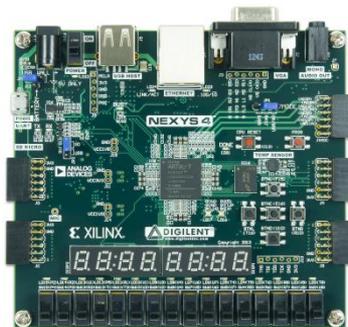
Digilent Arty-A7

[Видео обзор на русском](#)



Avnet Minized

[Возьми бесплатно](#)



Digilent Nexys4



PYNQ-Z2

Но их гораздо больше...

# Xilinx Vivado и Xilinx Vitis

Что использовать для работы с Xilinx?



Основная среда разработки RTL проектов для ПЛИС Xilinx на языках VHDL/Verilog/SystemVerilog



Проектирование с использованием высокоуровневого синтеза (High Level Synthesis - HLS) на языках C/C++/OpenCL/SystemC и написания программ для процессорной части Zynq/Zynq UltraScale+ и софтверного процессора MicroBlaze

Процесс установки максимально прост. Vitis содержит в себе Vivado, поэтому рекомендуется скачивать именно Xilinx Vitis.

Рекомендуется иметь не менее 100ГБ свободного места на диске

[UG973](#) Vivado Design Suite User Guide. Release Notes, Installation, and Licensing

1. Пройдите по [ссылке](#)
2. Выберите Vitis (SW Developer) и версию 2020.1

Downloads

Licensing Help Alveo Accelerator Card Downloads

Vivado (HW Developer) **1** Vitis (SW Developer) Vitis Embedded Platforms PetaLinux Device Models

Version **2**

**2020.1**  
2019.2  
SDSoC Archive  
SDAccel Archive  
SDK/PetaLinux Archive

Vitis Core Development Kit Update 1 - 2020.1 Product Update

**Important Information**

This is a common updater. You do not need to re-run it for Vitis if you have already run it for Vivado and vice versa.

Note: Download verification is only supported with Google Chrome and

Download Includes	Vitis Core Development Kit
Download Type	Product Update
Last Updated	Aug 12, 2020
Answers	<a href="#">2020.x - Vitis Known Issues</a>

3. Воспользуемся установщиком. Прокрутите вниз до ссылки на установщик



 [Xilinx Unified Installer 2020.1: Windows Self Extracting Web Installer \(EXE - 66.73 MB\)](#)

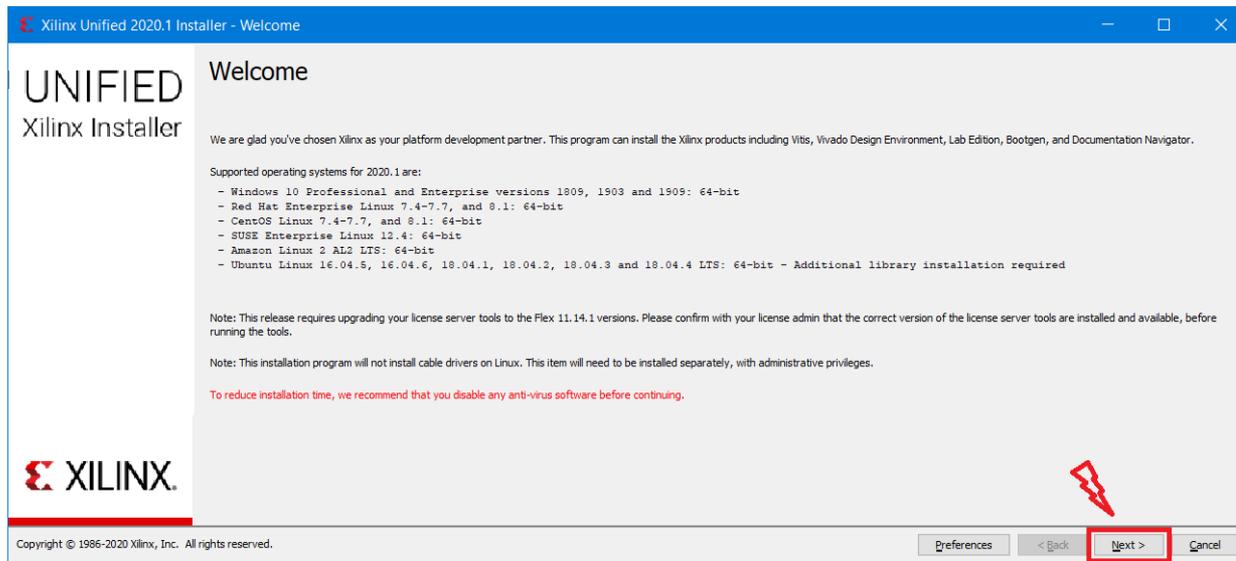
MD5 SUM Value : e4339ae3bcad478d7130edd669aac786

Download Verification 

[Digests](#) [Signature](#) [Public Key](#)

4. Скачайте установщик и запустите его (потребуется учетная запись на сайте Xilinx)

## 5. Нажмите “Next”



## 4. Скачайте установщик и запустите его (потребуется учетная запись на сайте Xilinx)

# Подготовка

## Устанавливаем Vitis/Vivado

5.1. Введите данные вашего аккаунта на сайте Xilinx (\*почту, указанную при регистрации)

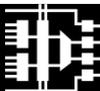
5.2 Выберите загрузку и отдельную установку

5.3 Укажите место для загрузки

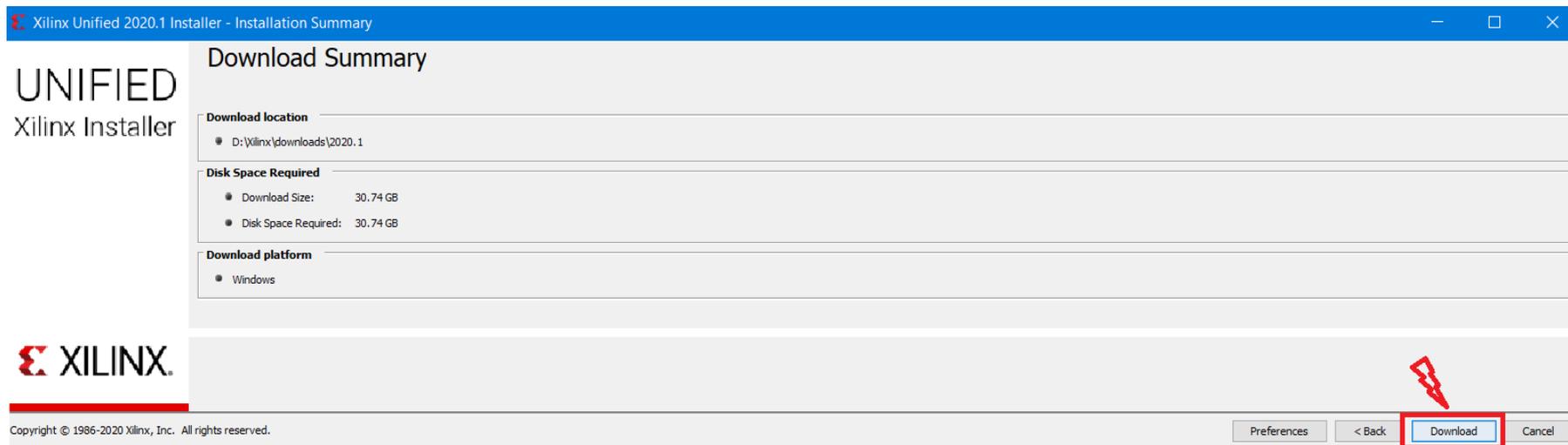
5.4 Выберите операционную систему

5.5 Выберите «Full Image»

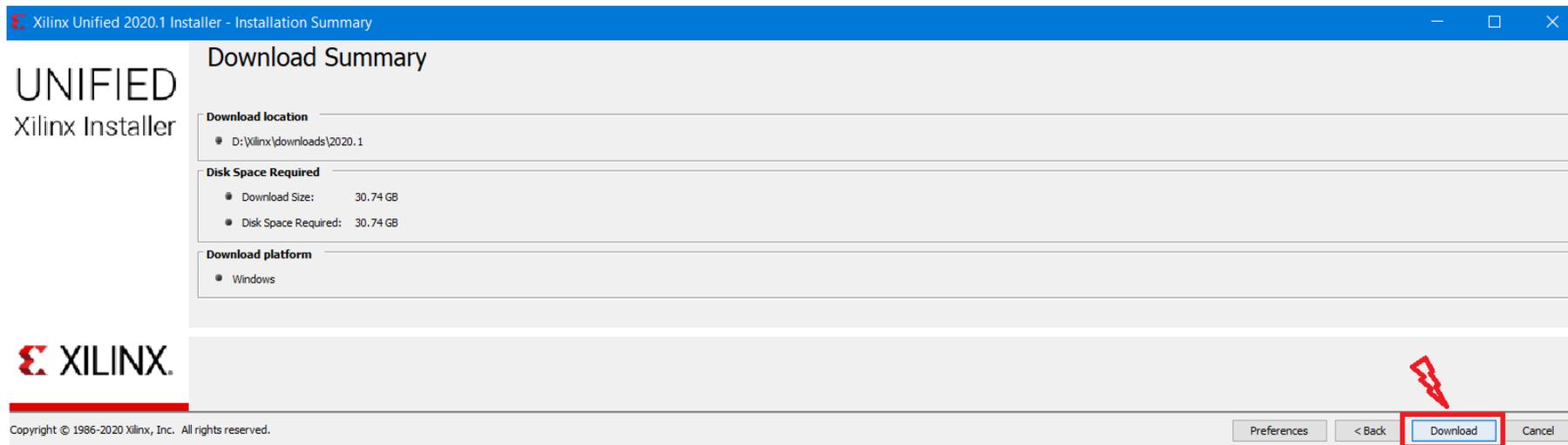
5.6 Нажмите «Next»



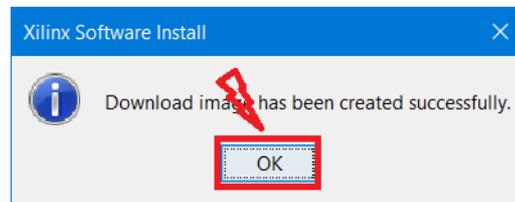
### 6. Нажмите «Download»



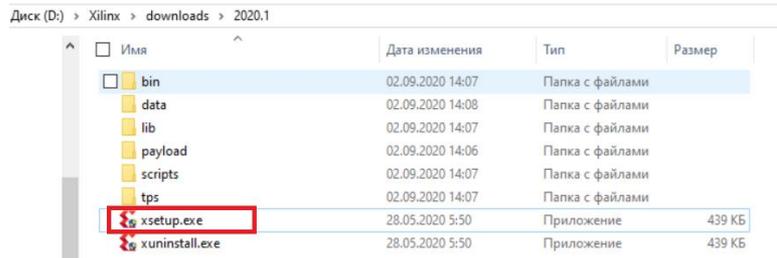
### 6. Нажмите «Download»



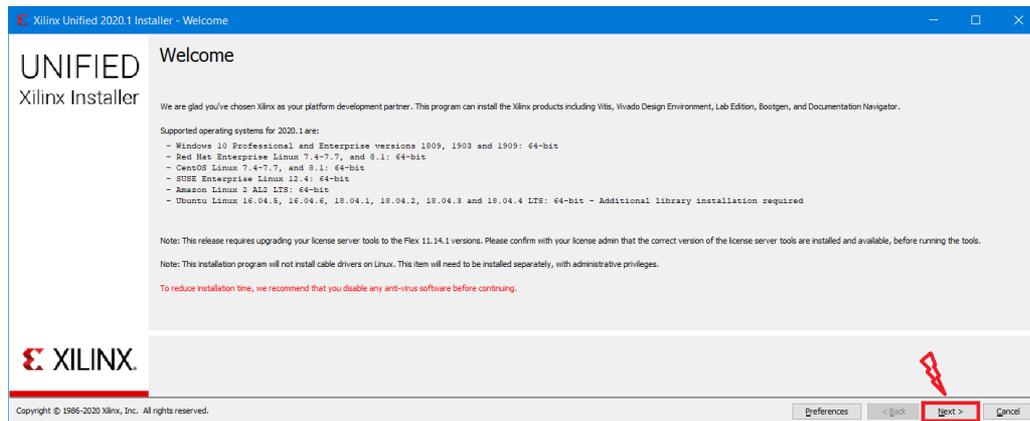
### 7. Дождитесь окончания загрузки



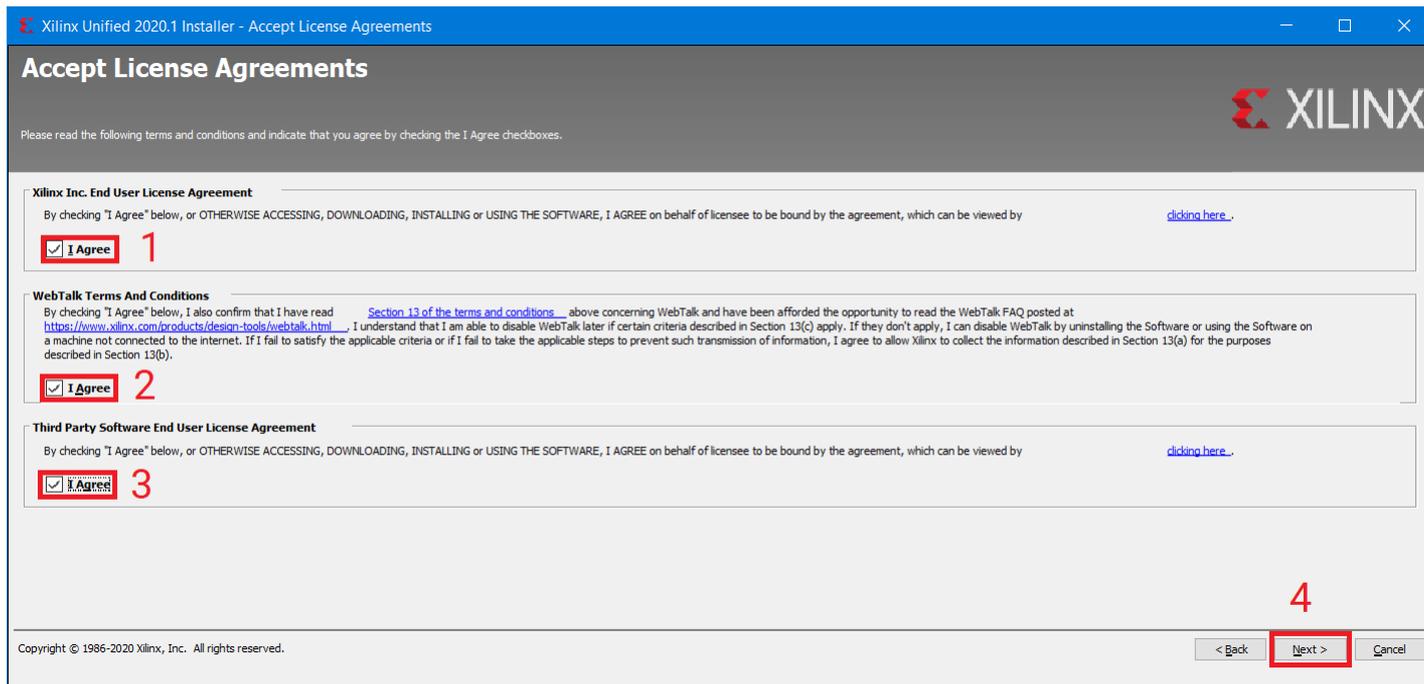
## 8. Перейдите в папку со скаченными файлами и запустите файл xsetup.exe



Нажмите «Next»



Соглашаемся с условиями использования и нажимаем «Next»



Xilinx Unified 2020.1 Installer - Accept License Agreements

## Accept License Agreements

Please read the following terms and conditions and indicate that you agree by checking the I Agree checkboxes.

**Xilinx Inc. End User License Agreement**

By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of licensee to be bound by the agreement, which can be viewed by [clicking here](#).

I Agree 1

**WebTalk Terms And Conditions**

By checking "I Agree" below, I also confirm that I have read [Section 13 of the terms and conditions](#) above concerning WebTalk and have been afforded the opportunity to read the WebTalk FAQ posted at <https://www.xilinx.com/products/design-tools/webtalk.html>. I understand that I am able to disable WebTalk later if certain criteria described in Section 13(c) apply. If they don't apply, I can disable WebTalk by uninstalling the Software or using the Software on a machine not connected to the internet. If I fail to satisfy the applicable criteria or if I fail to take the applicable steps to prevent such transmission of information, I agree to allow Xilinx to collect the information described in Section 13(a) for the purposes described in Section 13(b).

I Agree 2

**Third Party Software End User License Agreement**

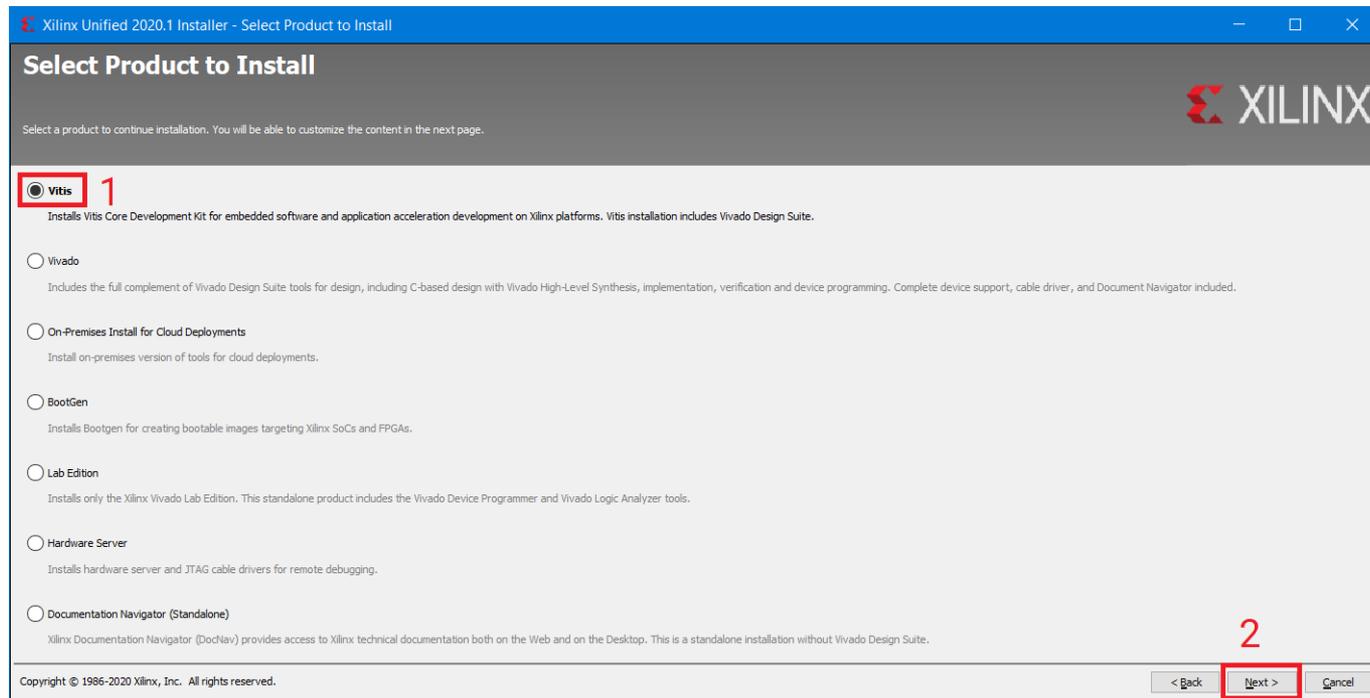
By checking "I Agree" below, or OTHERWISE ACCESSING, DOWNLOADING, INSTALLING or USING THE SOFTWARE, I AGREE on behalf of licensee to be bound by the agreement, which can be viewed by [clicking here](#).

I Agree 3

Copyright © 1986-2020 Xilinx, Inc. All rights reserved.

< Back **Next >** Cancel

Выбираем Vitis, нажимаем «Next»



# Подготовка

## Устанавливаем Vitis/Vivado

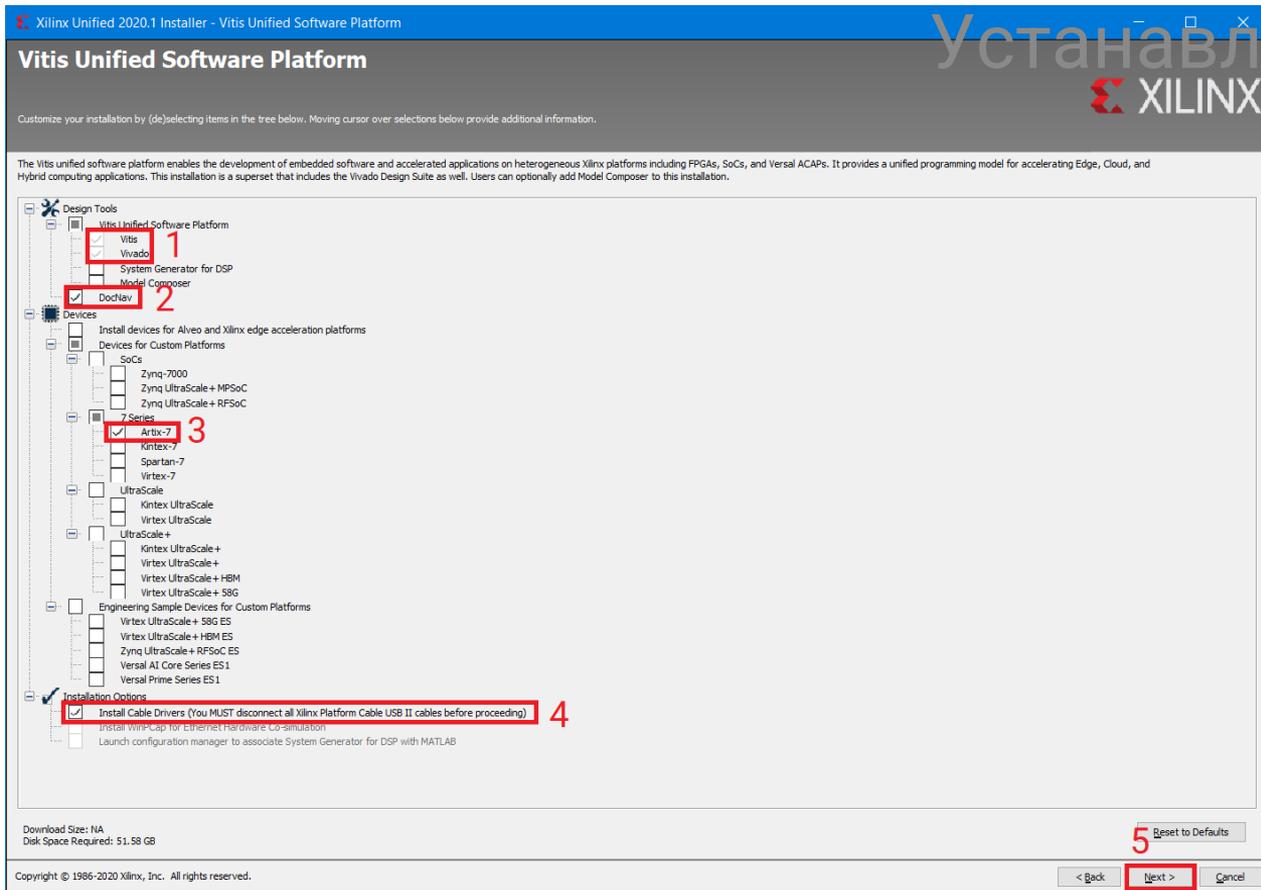


Минимальный набор для установки (60ГБ):

- Vitis/Vivado (1)
- DocNav (2)
- Artix-7 (3)
- Драйверы (4)

Нажмите «Next» (5)

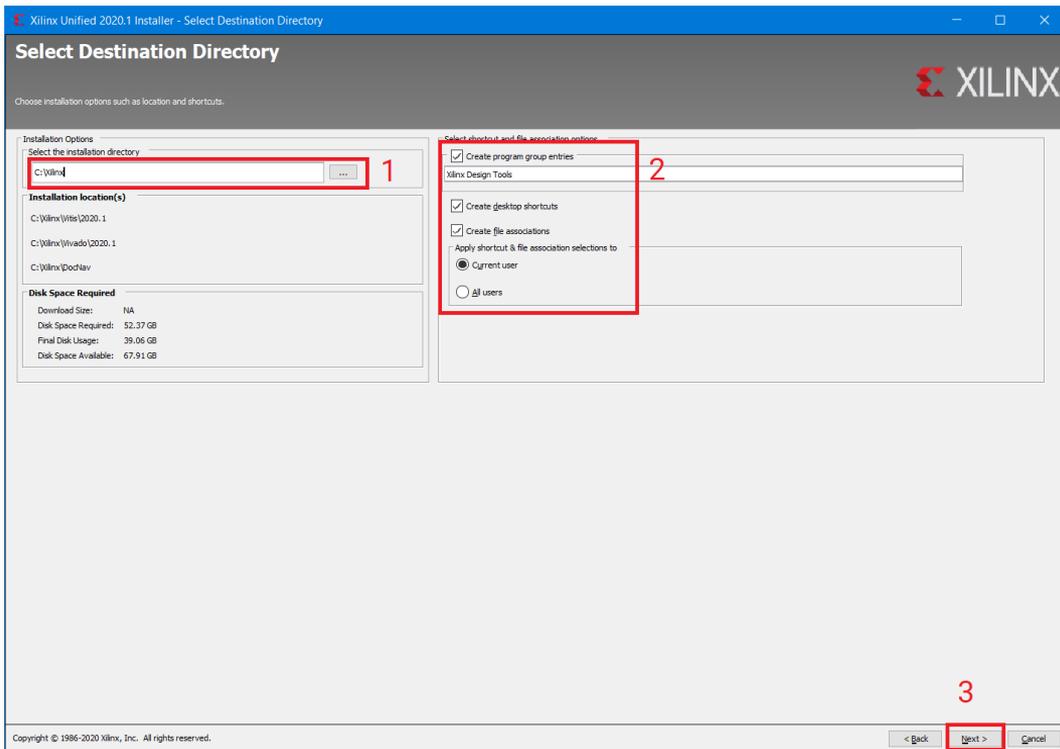
\*Вы можете установить полный комплект ПО и микросхем. Займет ~90ГБ



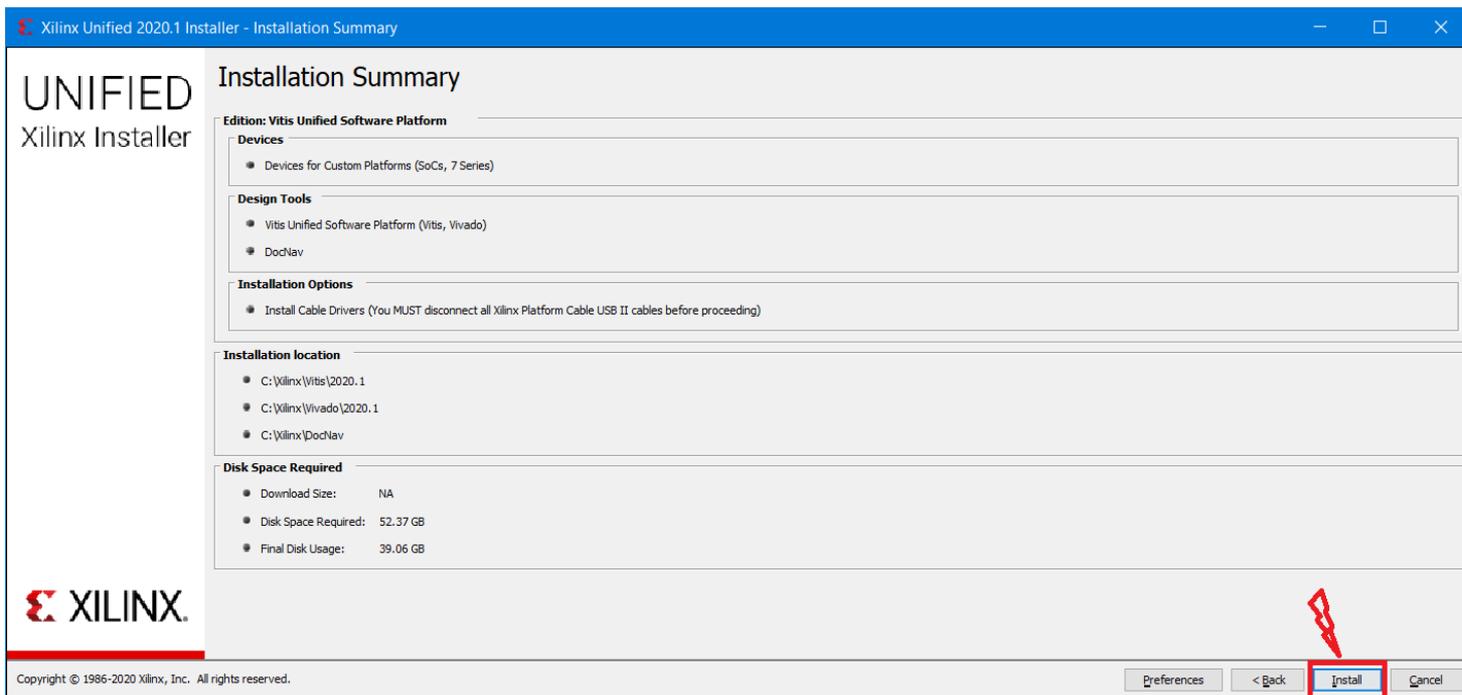
# Подготовка

## Устанавливаем Vitis/Vivado

Выберите место для установки (1),  
наличие иконок (2), нажмите далее (3)



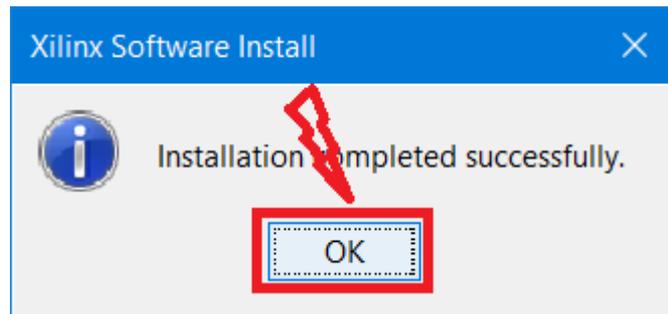
Нажмите «Install»



# Подготовка

## Устанавливаем Vitis/Vivado

Дождитесь окончания установки

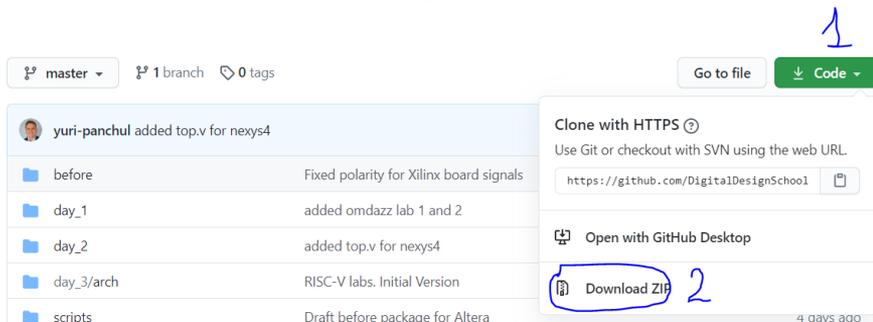


# Подготовка

## Скачиваем исходники проекта с github

Мы скачаем исходный код игры с github и разберем далее как он устроен

1. Пройдите по [ссылке](#) в репозиторий
2. Скачайте репозиторий

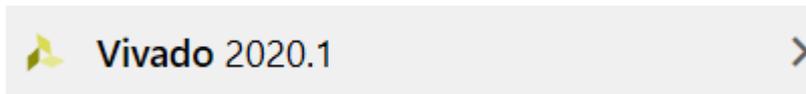


3. Распакуйте архив. Путь до распакованной папки не должен содержать русских букв, а длина пути минимальна. Например, D:/projects/ce2020labs-master

# Приступая к работе

## Запускаем Vivado

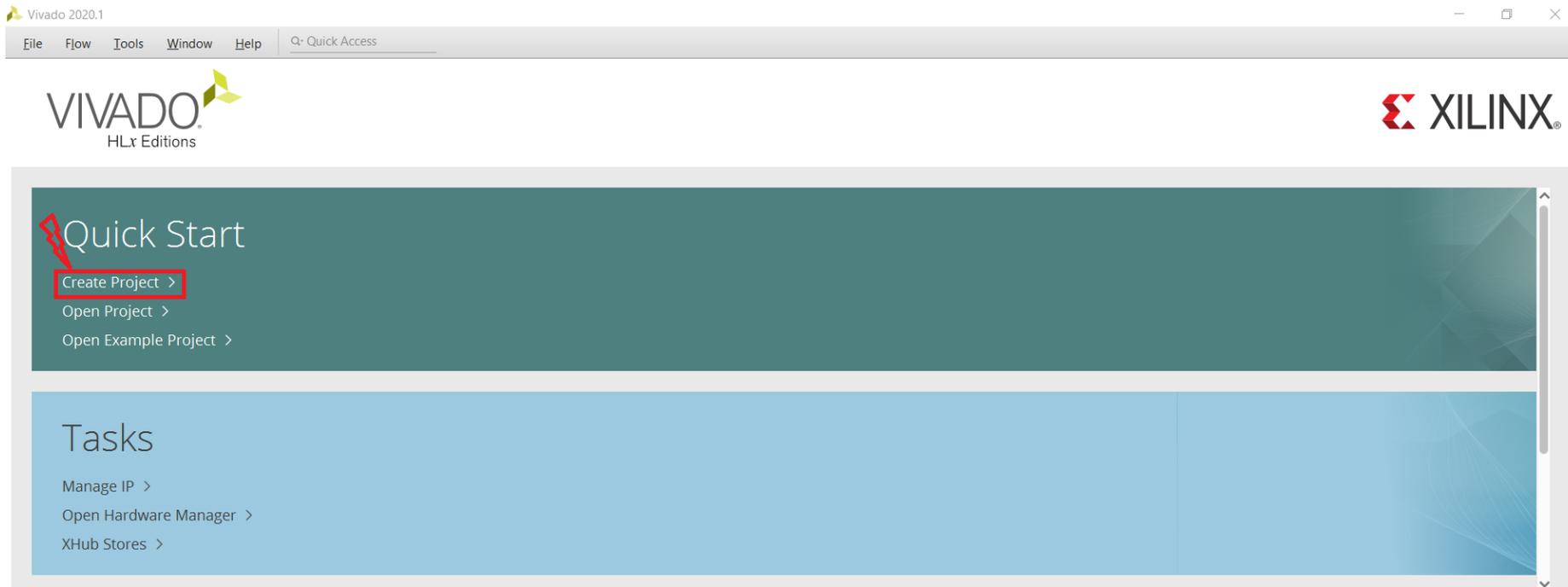
Откройте Vivado: Пуск - > Xilinx Design Tools -> Vivado 2020.1



# Создание проекта в Vivado

## Создание нового проекта

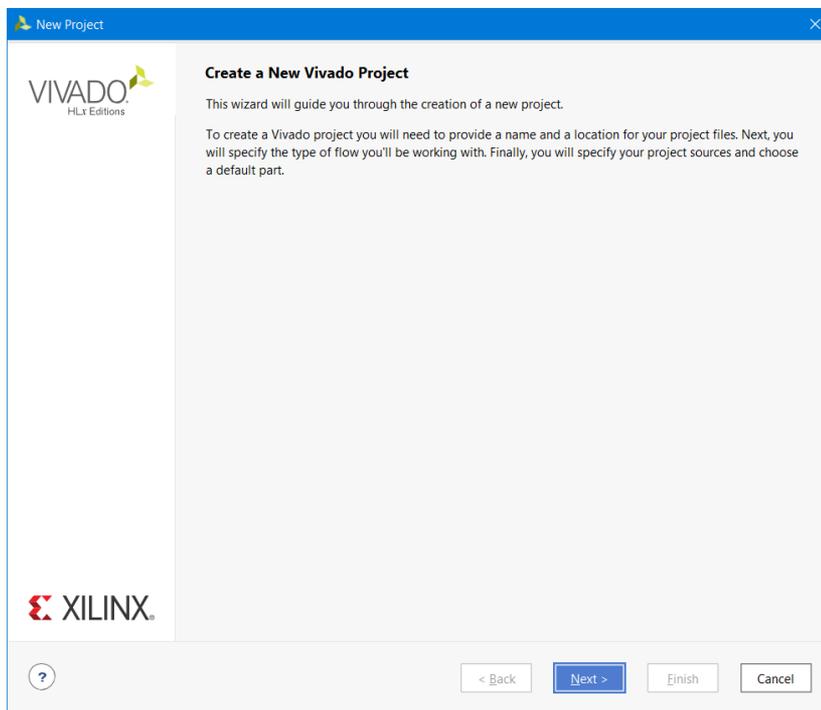
Нажмите Create project



# Создание проекта в Vivado

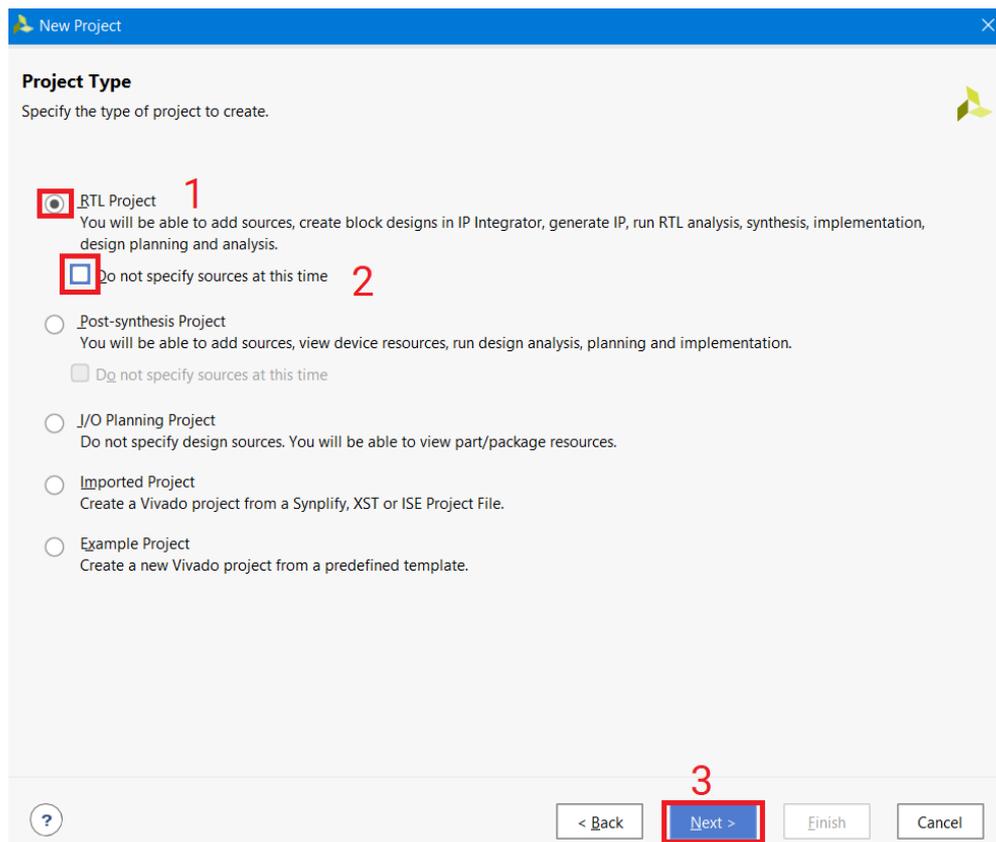
## Создание нового проекта

Нажмите «Next» в мастере создания нового проекта





# Создание проекта в Vivado

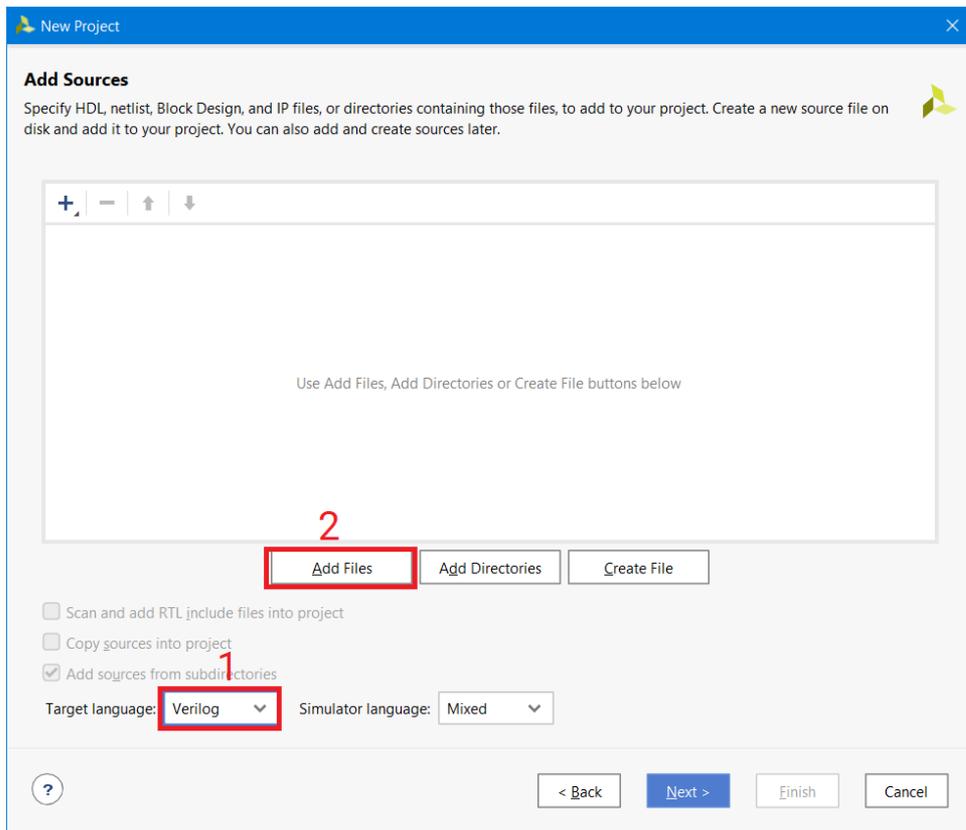


## Создание проекта

Выберите:

- Тип проекта – RTL (1)
- Снимите галочку (2)
- Нажмите далее (3)

# Создание проекта в Vivado

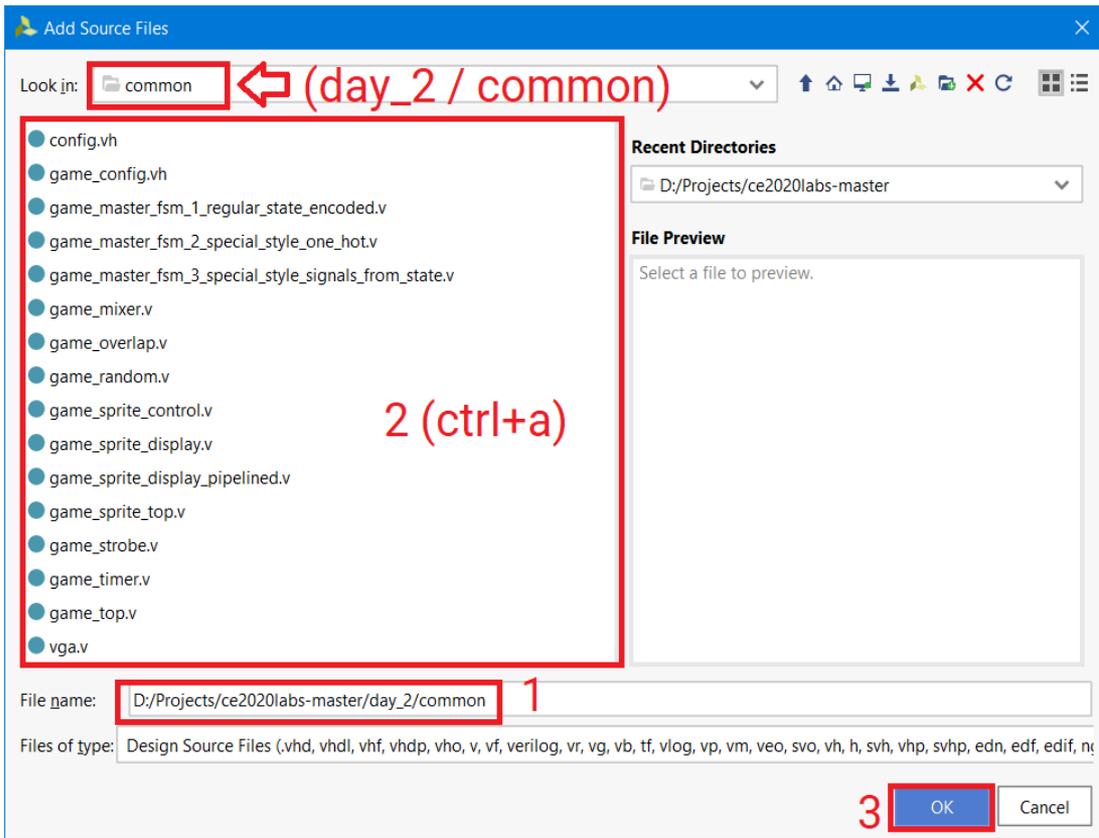


## Создание проекта

Добавим исходники:

- Тип по умолчанию – Verilog (1)
- Нажимаем кнопку добавить файлы (2)

# Создание проекта в Vivado

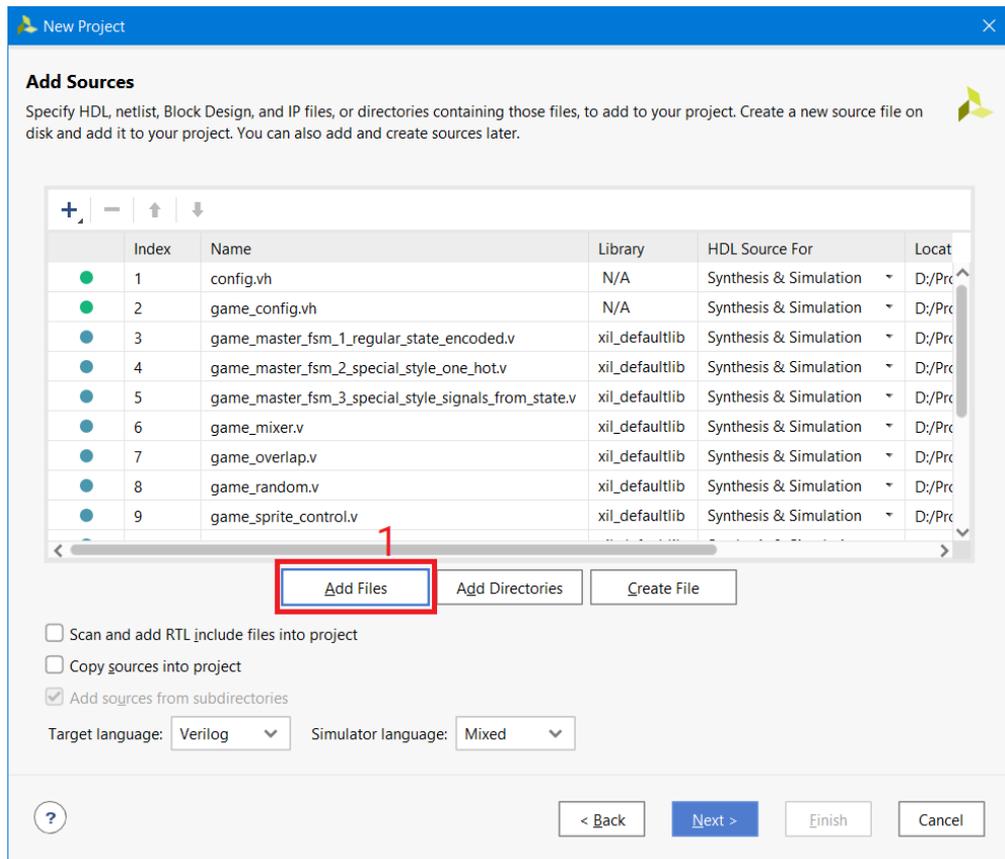


## Создание проекта

Перейдите в папку скачанного репозитория

- ce2020labs-master/day\_2/common
- Выберите все файлы нажав ctrl+a
- Нажмите OK (3)

# Создание проекта в Vivado



## Создание проекта

Добавим файл модуля верхнего уровня:

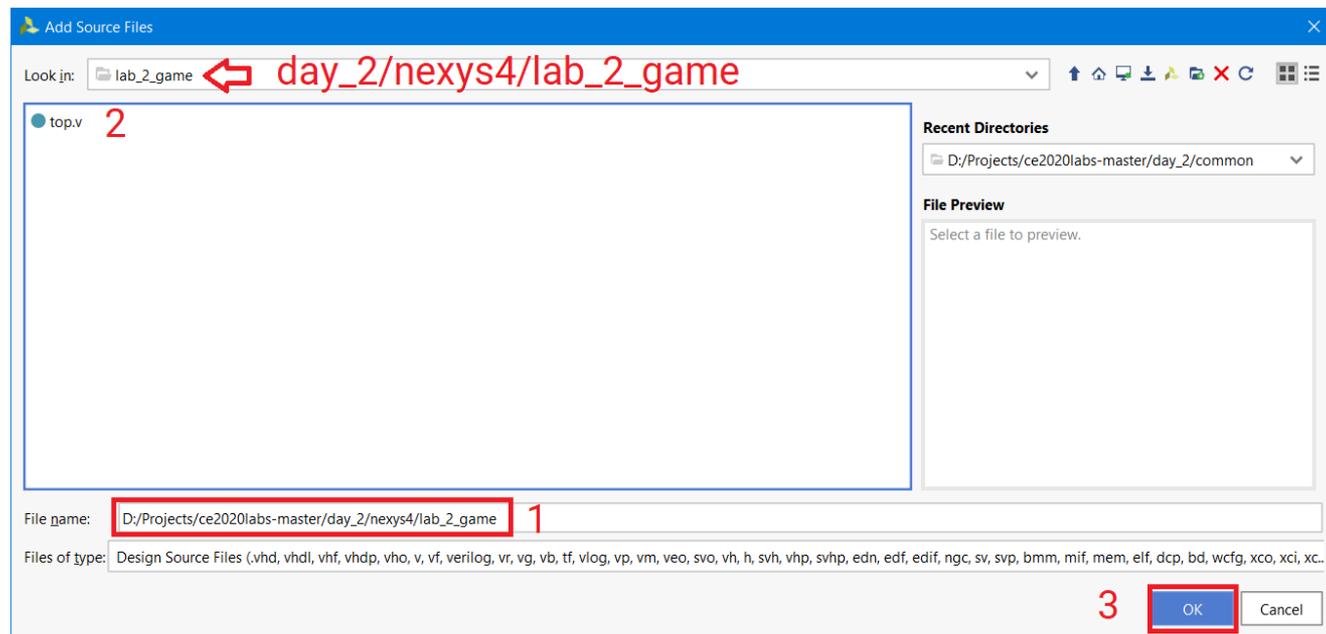
- Нажмите добавить файл (1)

# Создание проекта в Vivado

## Создание проекта

Добавим файл модуля верхнего уровня:

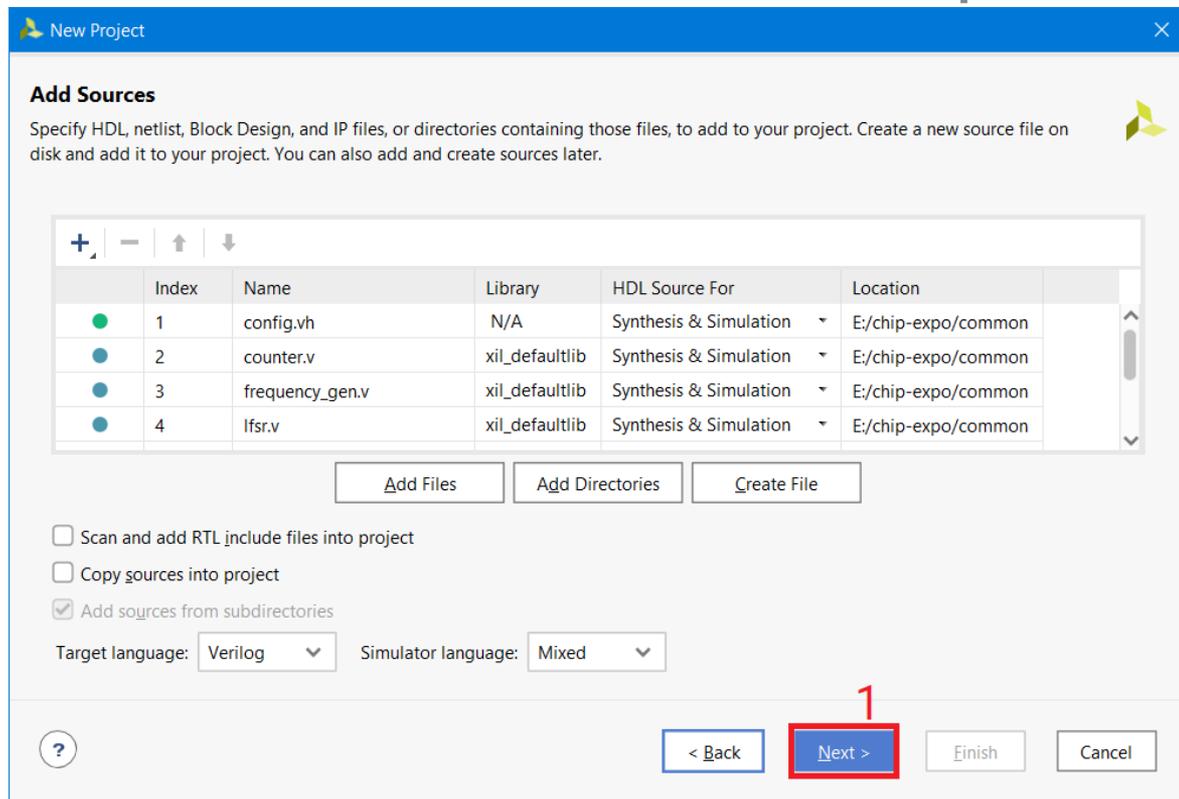
- Перейдите в папку скачанного репозитория ce2020labs-master/day\_2/nexus4/lab\_2\_game
- Выберите файл top.v
- Нажмите ОК (3)



# Создание проекта в Vivado

## Создание проекта

Нажмите далее (1)



**Add Sources**

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

	Index	Name	Library	HDL Source For	Location
●	1	config.vh	N/A	Synthesis & Simulation	E:/chip-expo/common
●	2	counter.v	xil_defaultlib	Synthesis & Simulation	E:/chip-expo/common
●	3	frequency_gen.v	xil_defaultlib	Synthesis & Simulation	E:/chip-expo/common
●	4	lfsr.v	xil_defaultlib	Synthesis & Simulation	E:/chip-expo/common

Scan and add RTL include files into project

Copy sources into project

Add sources from subdirectories

Target language: Verilog Simulator language: Mixed

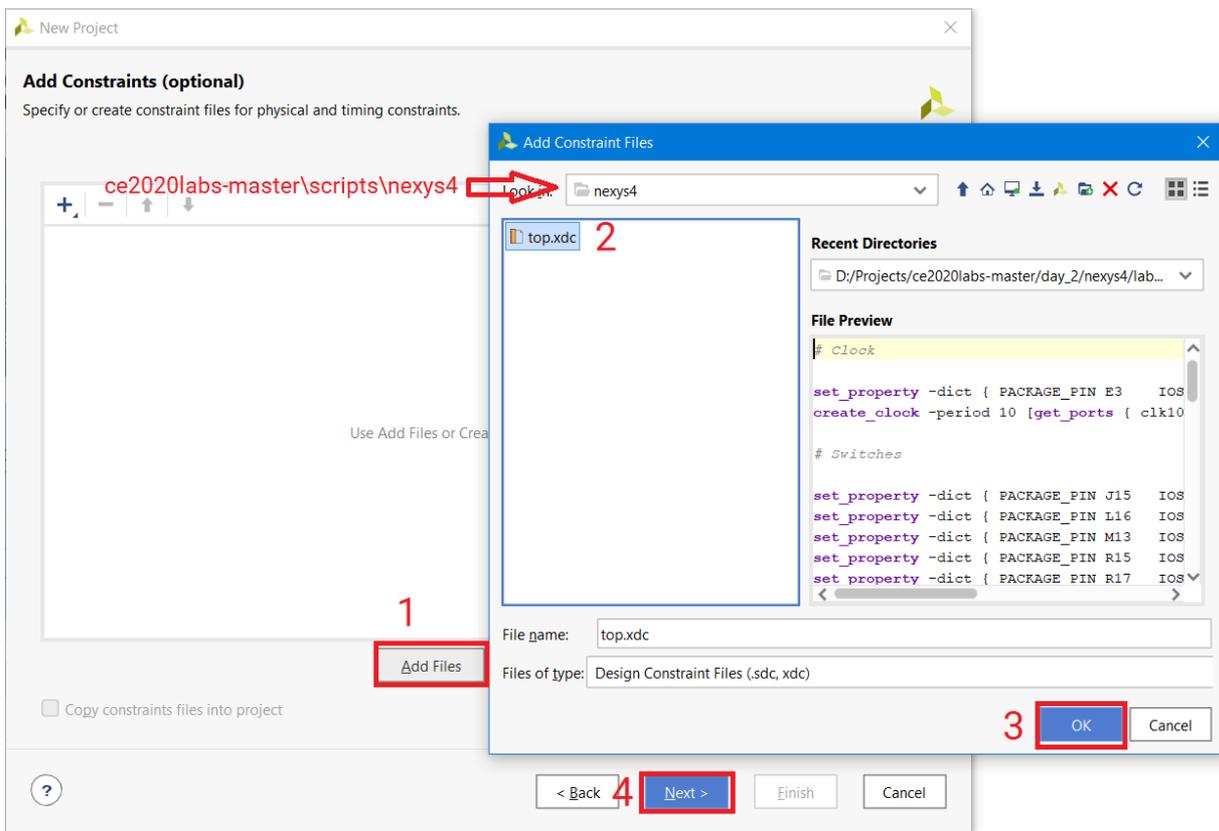
**Next >**

# Создание проекта в Vivado

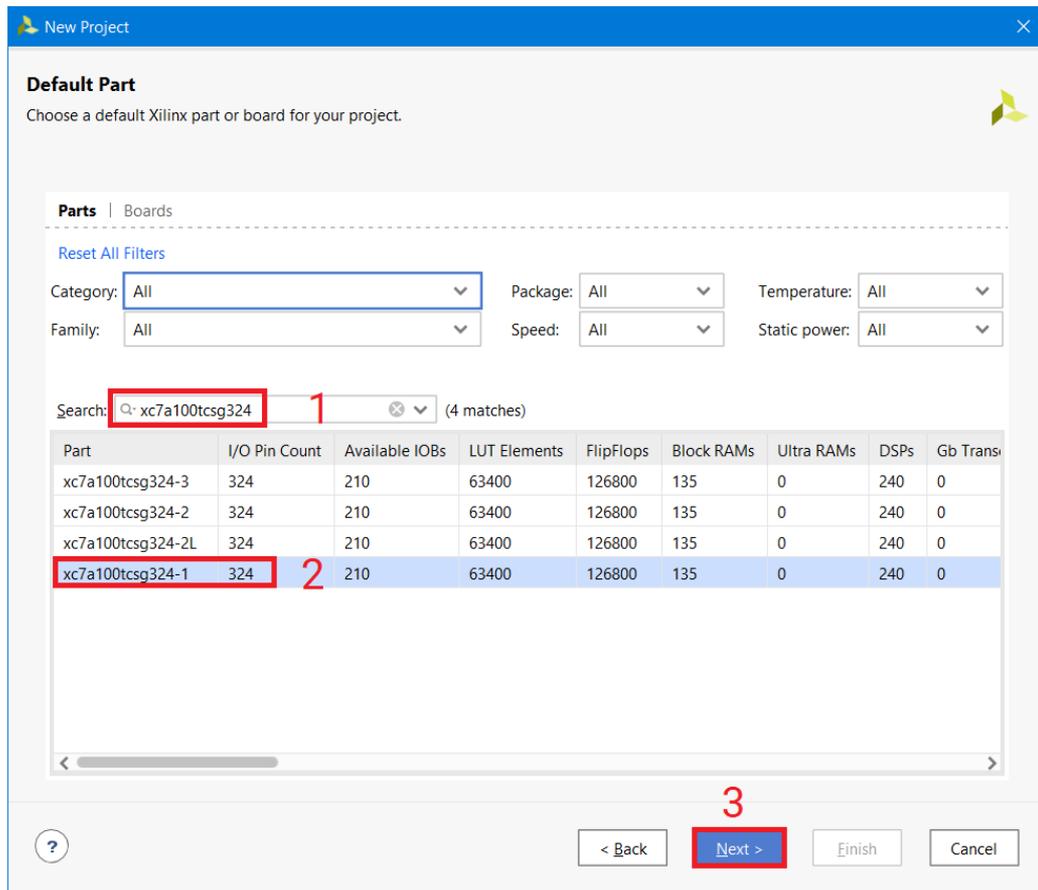
## Создание проекта

Добавим файл проектных ограничений

- Нажмите добавить файл (1)
- Перейдите в папку ce2020labs-master/scripts/nexus4
- Выберите файл top.xdc (2)
- Нажмите ОК (3)
- Нажмите далее (4)



# Создание проекта в Vivado

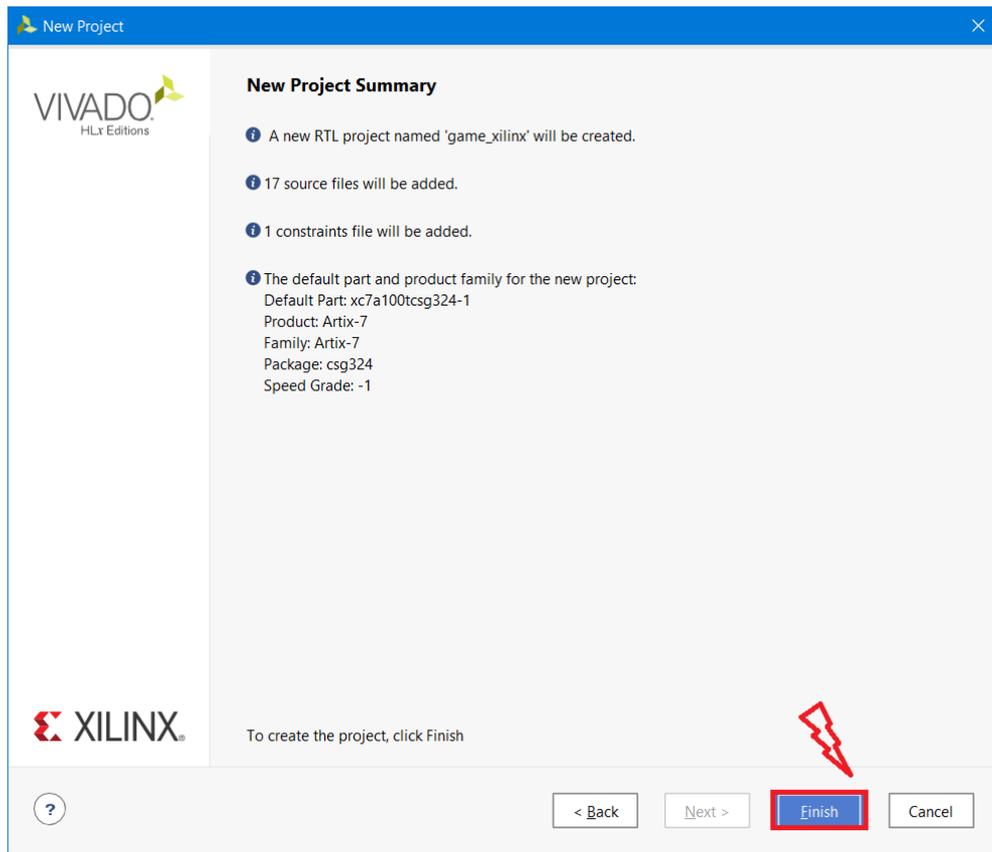


## Создание проекта

Выберем ПЛИС, установленную на плате

- Введите в поле (1) xc7a100tcsq324
- Выберите быстродействие -1 (2)
- Нажмите далее (3)

# Создание проекта в Vivado



## Создание проекта

Создание проекта завершено,  
нажмите Finish

# Как спроектировать игру?

## Сформулируем цель

**Цель** – разработать устройство, выводящее на экран игру с двумя движущимися объектами (спрайтами). Движение объекта 1 осуществить по горизонтали с постоянной скоростью. Движение объекта 2 постоянно по вертикали и дополнительно корректируется кнопками вправо/влево для движения вправо/влево соответственно. При столкновении объектов или выходе объектов за поле экрана вывести экран с «белым шумом». Видео поток сформировать для дисплея с интерфейсом VGA 640x480 60кадров/с

# Как спроектировать игру?

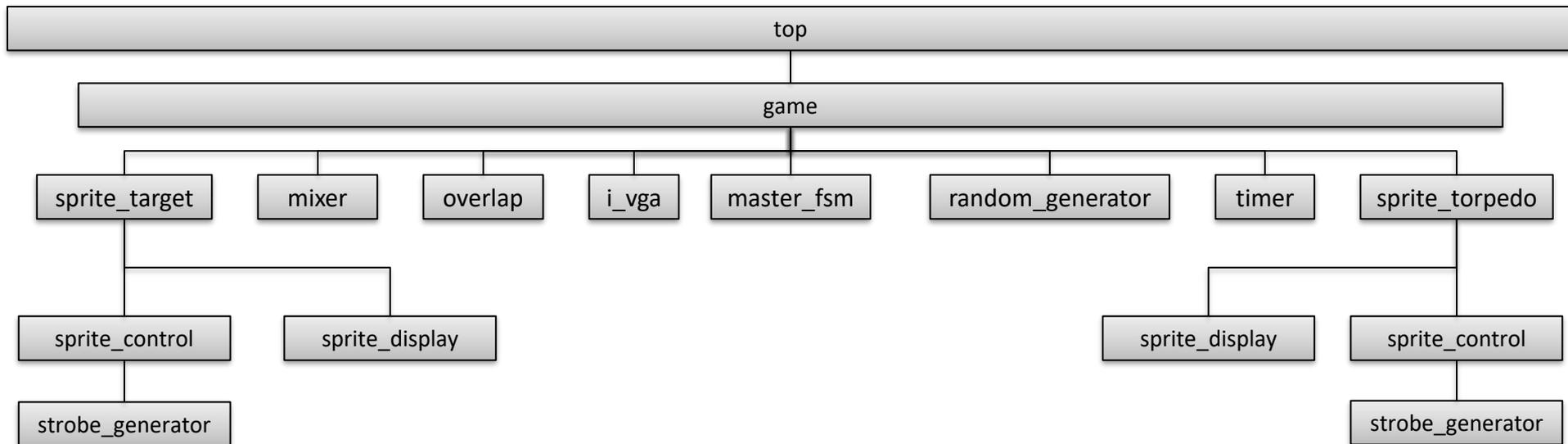
## Сформулируем задачи

### **Задачи, решаемые для достижения цели:**

1. Разработать модуль формирования развёртки VGA заданного формата
2. Определить размер объектов и вариант их графического представления
3. Определить пересечение объектов 1 и 2
4. Определить достижение объектом 1 или 2 границ экрана
5. Обеспечить ввод и считывание информации с кнопок для контроля объекта 2
6. Создать конфигурационный файл для ПЛИС, установленной на плате Nexus A7

# Иерархия проекта

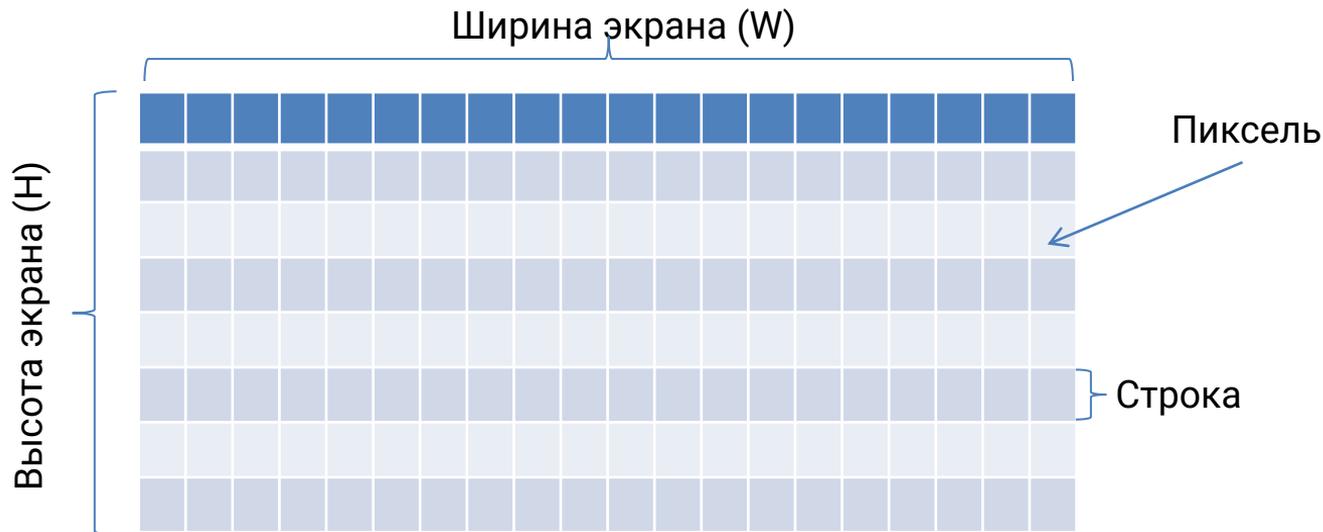
Что за чем?



# Как изображение выводится на экран?

## ОСНОВНЫЕ ОСНОВЫ ОСНОВ

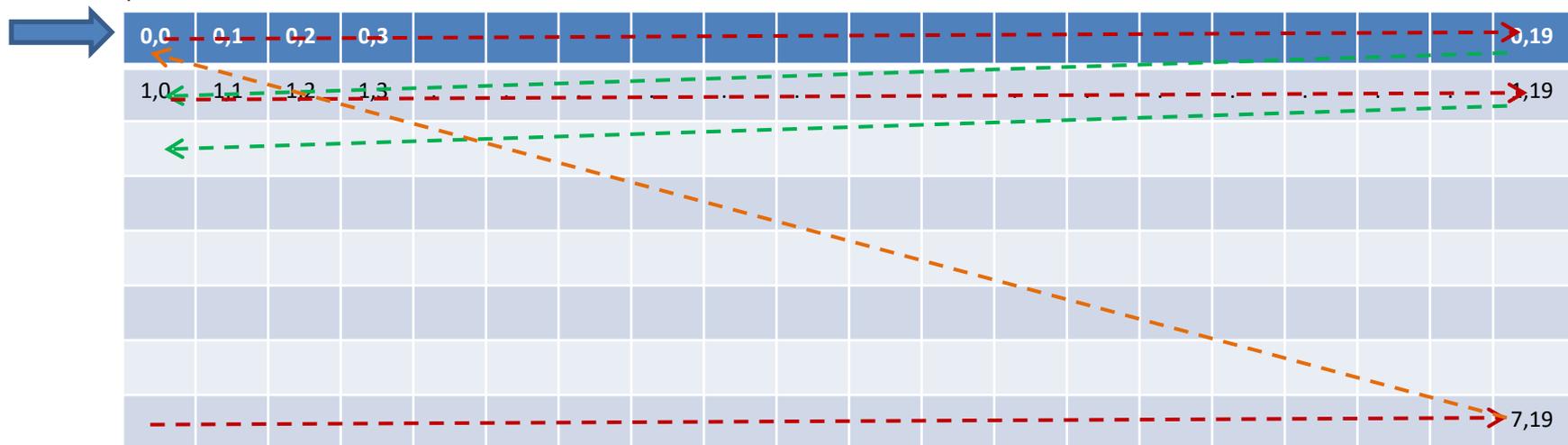
Задача №1 Разработать модуль формирования развёртки VGA заданного формата



# Как изображение выводится на экран?

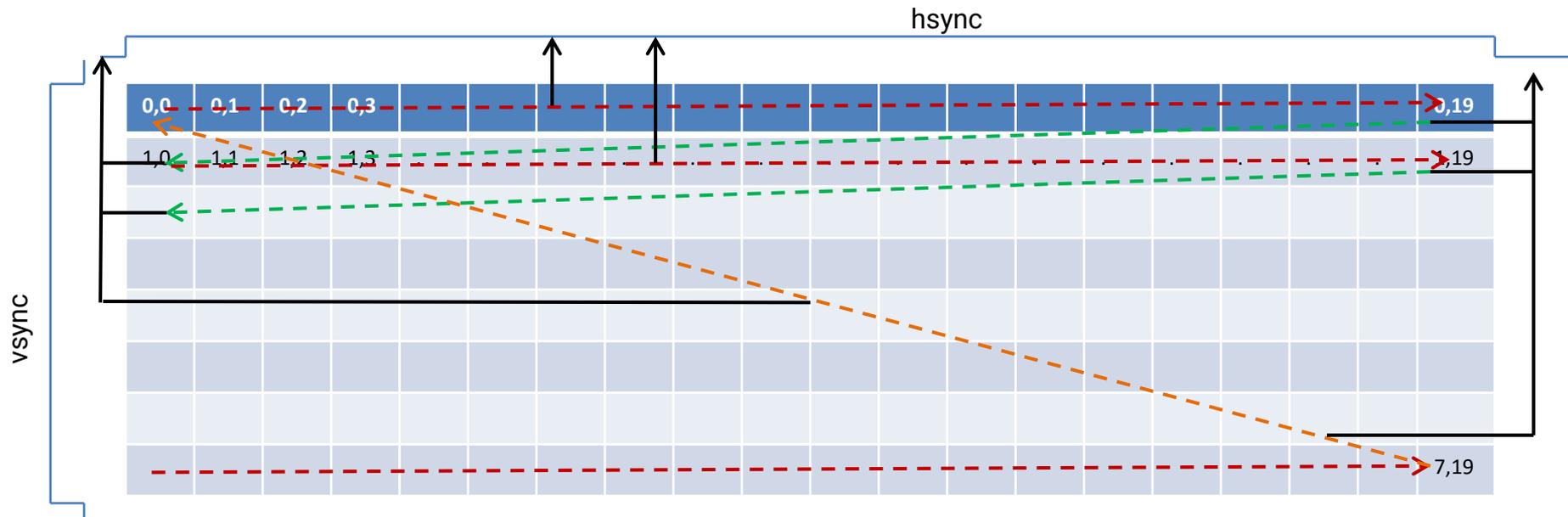
## ОСНОВНЫЕ ОСНОВЫ ОСНОВ

Начало кадра



# Как изображение выводится на экран?

## ОСНОВНЫЕ ОСНОВЫ ОСНОВ



# Как изображение выводится на экран?

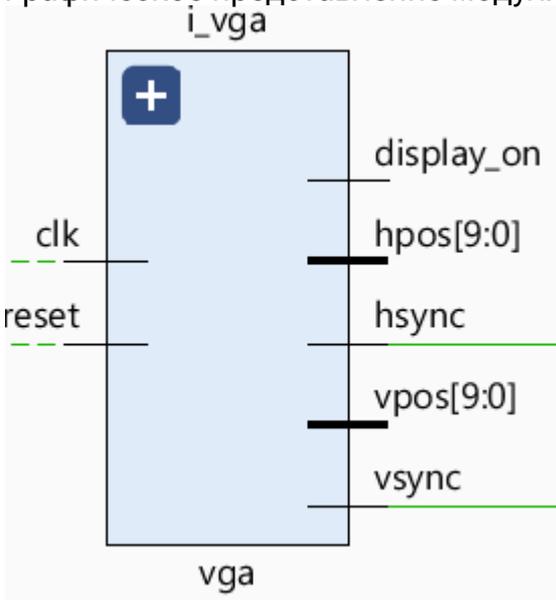
## ОСНОВНЫЕ ОСНОВЫ ОСНОВ

Разрешение	Частота обновления (Гц)	Частота пикселей (МГц)	Строки				Кадры				Полярность	
			Пиксели	Передний порожек	Синхронизация	Задний порожек	Строки	Передний порожек	Синхронизация	Задний порожек	h_sync *	v_sync *
640x350	70	25.175	640	16	96	48	350	37	2	60	p	n
640x350	85	31.5	640	32	64	96	350	32	3	60	p	n
640x400	70	25.175	640	16	96	48	400	12	2	35	n	p
640x400	85	31.5	640	32	64	96	400	1	3	41	n	p
640x480	60	25.175	640	16	96	48	480	10	2	33	n	n
640x480	73	31.5	640	24	40	128	480	9	2	29	n	n
640x480	75	31.5	640	16	64	120	480	1	3	16	n	n
640x480	85	36	640	56	56	80	480	1	3	25	n	n

[Ссылка на таблицу](#)

# Как же сформировать развертку на экран?

Графическое представление модуля VGA



Параметры модуля vga

```
// Horizontal constants
H_DISPLAY      = 640, // Horizontal display width
H_FRONT       = 16,  // Horizontal right border (front porch)
H_SYNC        = 96,  // Horizontal sync width
H_BACK        = 48,  // Horizontal left border (back porch)

// Vertical constants
V_DISPLAY     = 480, // Vertical display height
V_BOTTOM     = 10,  // Vertical bottom border
V_SYNC       = 2,   // Vertical sync # lines
V_TOP        = 33,  // Vertical top border
```

Условия формирования управляющих сигналов

```
begin
  hsync    <= ~ ( d_hpos >= H_SYNC_START
                && d_hpos <= H_SYNC_END );

  vsync    <= ~ ( d_vpos >= V_SYNC_START
                && d_vpos <= V_SYNC_END );

  display_on <= ( d_hpos < H_DISPLAY
                && d_vpos < V_DISPLAY );

  hpos     <= d_hpos;
  vpos     <= d_vpos;
end
```

## Посчитаем

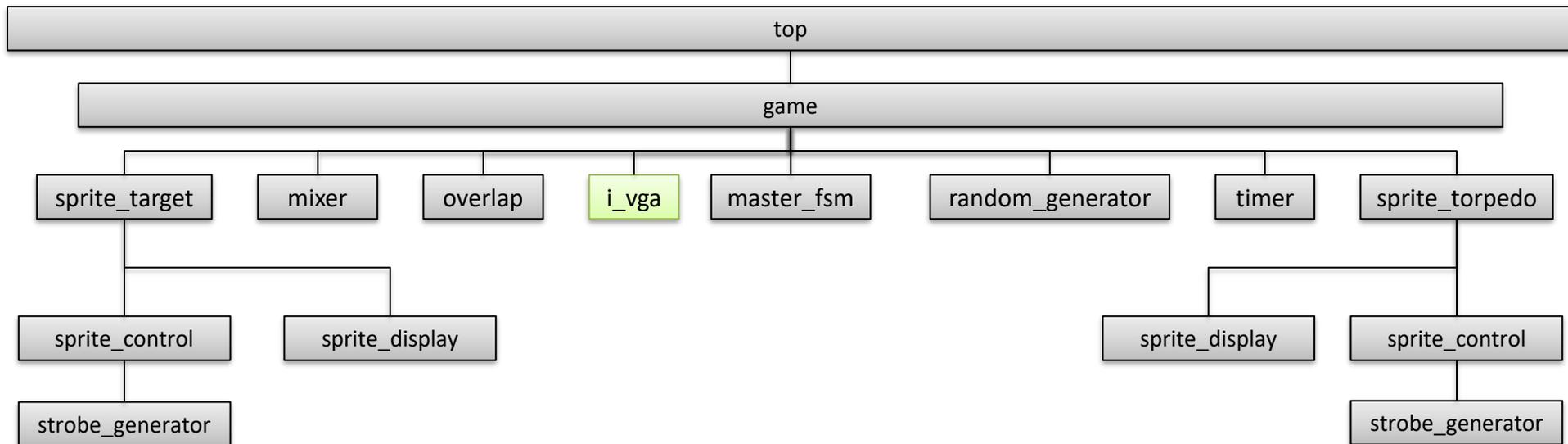
Счетчики пикселей и строк

```
always @*
begin
  if (hpos == H_MAX)
  begin
    d_hpos = 1'd0;

    if (vpos == V_MAX)
      d_vpos = 1'd0;
    else
      d_vpos = vpos + 1'd1;
  end
  else
  begin
    d_hpos = hpos + 1'd1;
    d_vpos = vpos;
  end
end
```

# Иерархия проекта

Что за чем?



# Объекты 1 и 2

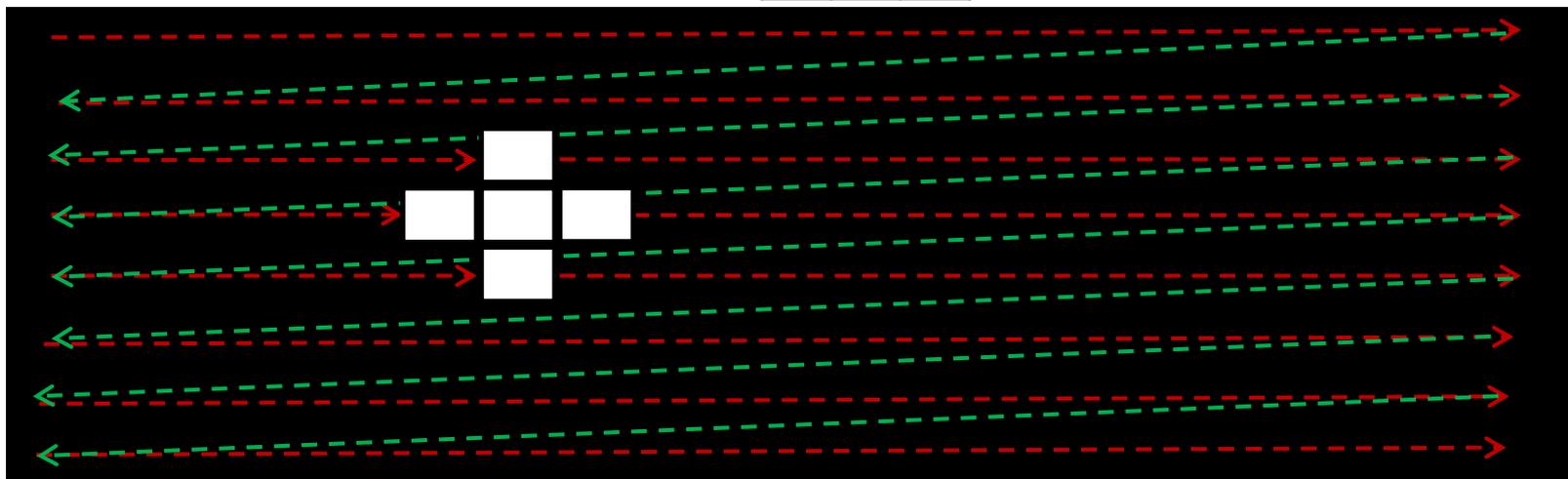
## Спрайты

Задача №2. Определить размер объектов и вариант их графического представления

ROM

0	1	0
1	1	1
0	1	0

\*Где «1» там цвет пикселя белый



# Объекты 1 и 2

Шаблон объекта (game\_sprite\_top.v)

Создание объектов

# Спрайты

```
module game_sprite_top
#(
    parameter SPRITE_WIDTH  = 8,
              SPRITE_HEIGHT = 8,

              DX_WIDTH      = 2, // X speed width in bits
              DY_WIDTH      = 2, // Y speed width in bits

              ROW_0         = 32'h000cc000,
              ROW_1         = 32'h000cc000,
              ROW_2         = 32'h000cc000,
              ROW_3         = 32'hcccccccc,
              ROW_4         = 32'hcccccccc,
              ROW_5         = 32'h000cc000,
              ROW_6         = 32'h000cc000,
              ROW_7         = 32'h000cc000,

              strobe_to_update_xy_counter_width = 20
)

```

```
game_sprite_top
#(
    .SPRITE_WIDTH  ( 8 ),
    .SPRITE_HEIGHT ( 8 ),

    .DX_WIDTH      ( 2 ),
    .DY_WIDTH      ( 3 ),

    .ROW_0 ( 32'h000cc000 ),
    .ROW_1 ( 32'h00cccc00 ),
    .ROW_2 ( 32'h0cceecc0 ),
    .ROW_3 ( 32'hcccccccc ),
    .ROW_4 ( 32'hcc0cc0cc ),
    .ROW_5 ( 32'hcc0cc0cc ),
    .ROW_6 ( 32'hcc0cc0cc ),
    .ROW_7 ( 32'hcc0cc0cc ),

    .strobe_to_update_xy_counter_width
    (strobe_to_update_xy_counter_width)
)
sprite_torpedo

```

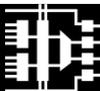
```
game_sprite_top
#(
    .SPRITE_WIDTH  ( 8 ),
    .SPRITE_HEIGHT ( 8 ),

    .DX_WIDTH      ( 2 ),
    .DY_WIDTH      ( 1 ),

    .ROW_0 ( 32'h000bb000 ),
    .ROW_1 ( 32'h00099000 ),
    .ROW_2 ( 32'h00099000 ),
    .ROW_3 ( 32'hb99ff99b ),
    .ROW_4 ( 32'hb99ff99b ),
    .ROW_5 ( 32'h00099000 ),
    .ROW_6 ( 32'h00099000 ),
    .ROW_7 ( 32'h000bb000 ),

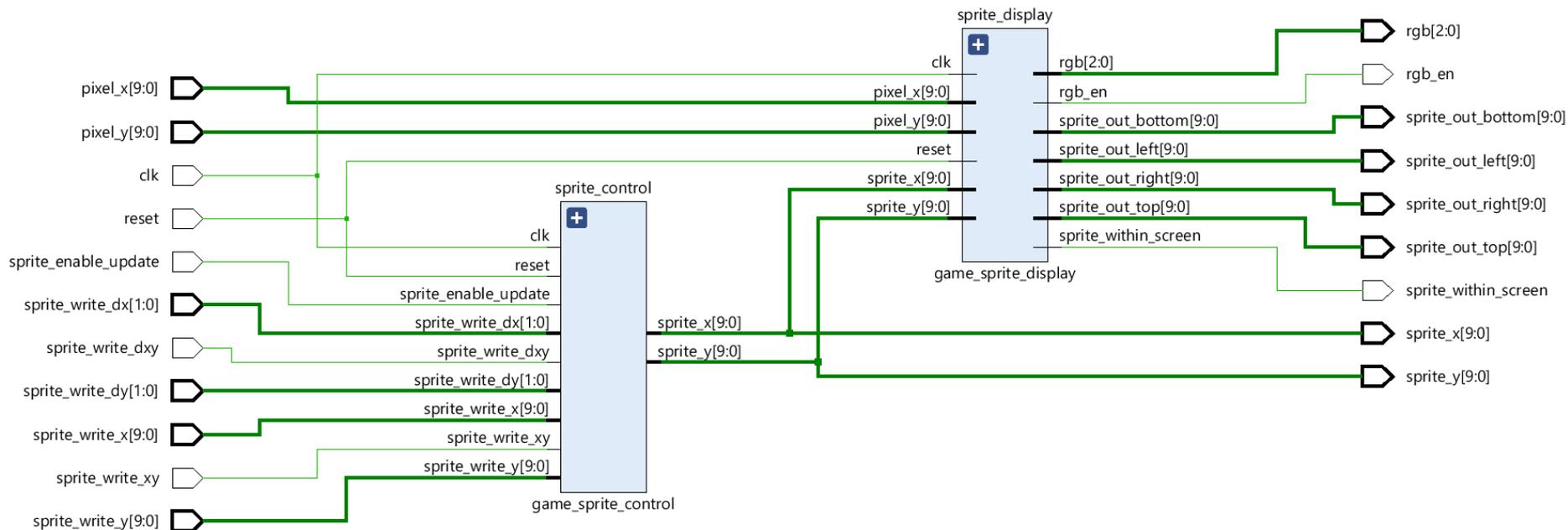
    .strobe_to_update_xy_counter_width
    (strobe_to_update_xy_counter_width)
)
sprite_target

```



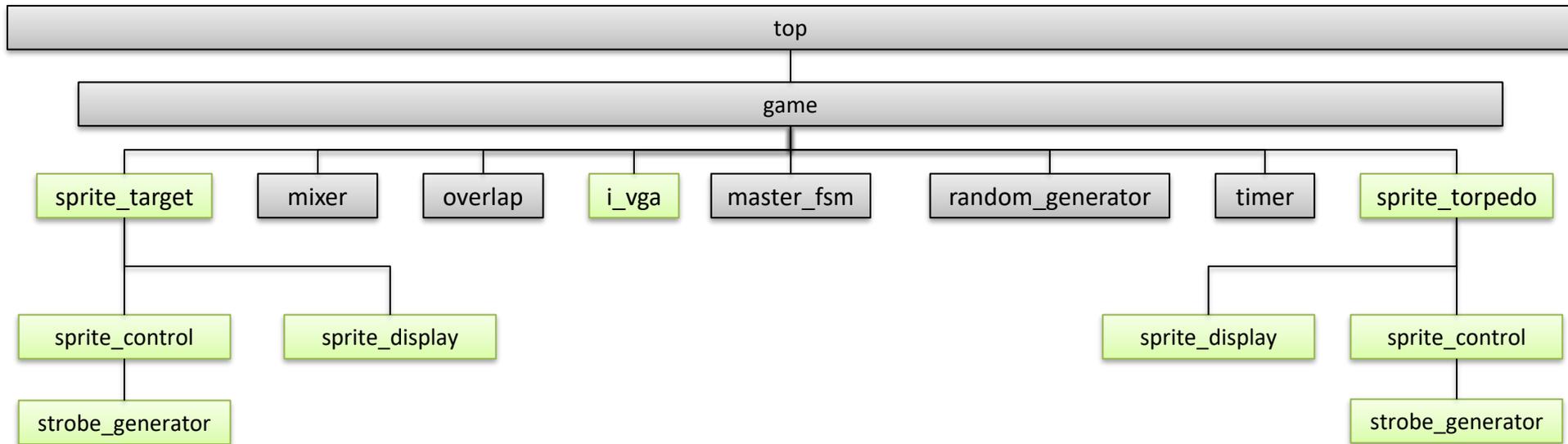
# Объекты 1 и 2

## Спрайты



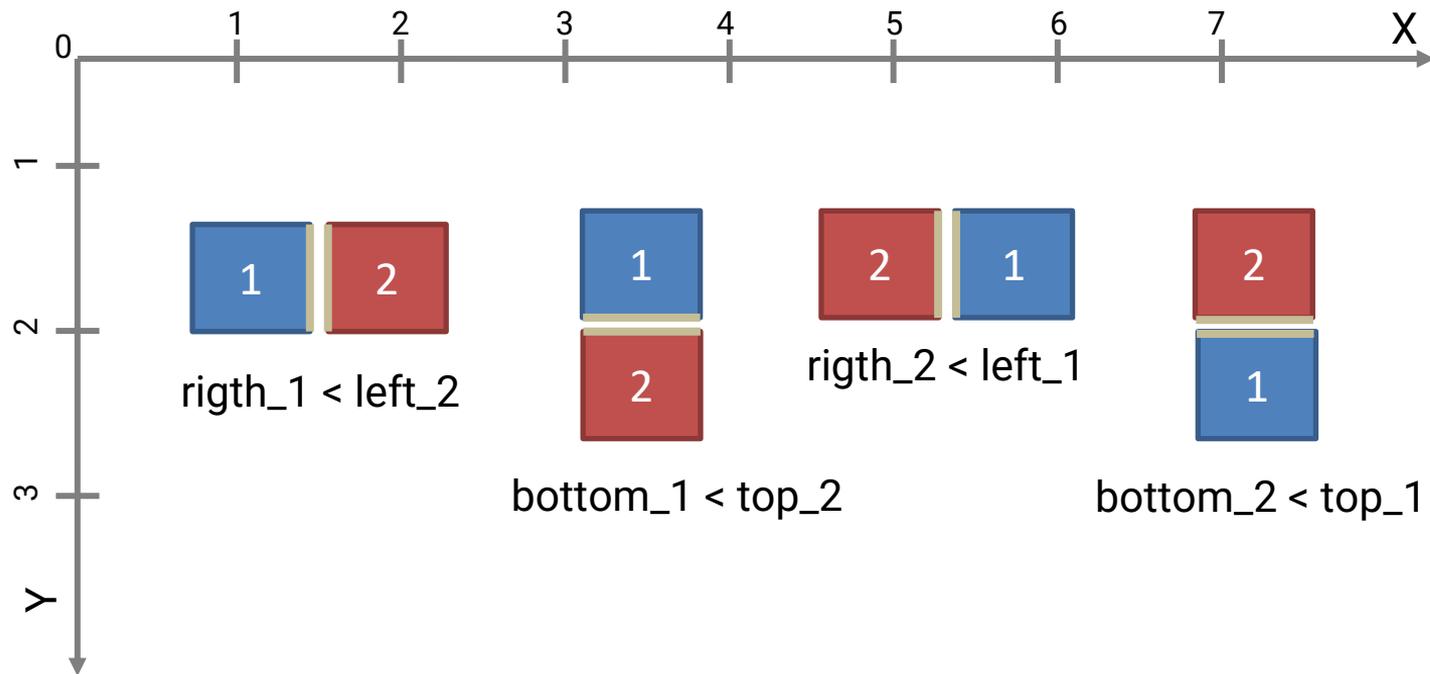
# Иерархия проекта

Что за чем?



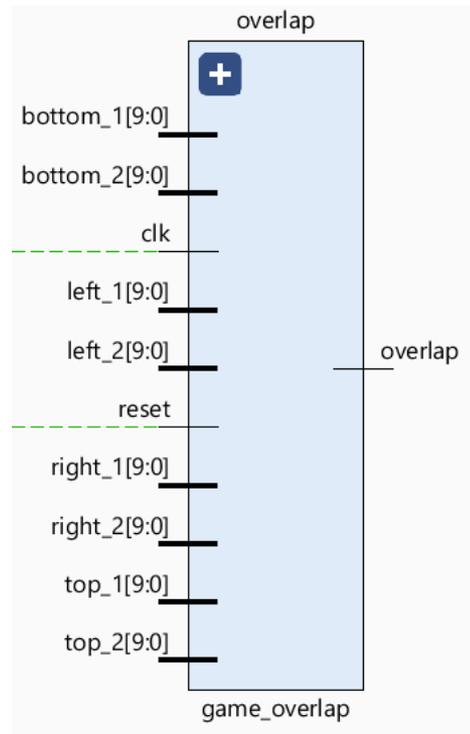
# Задача №3

Определить пересечение объектов 1 и 2



# Задача №3

## Определить пересечение объектов 1 и 2



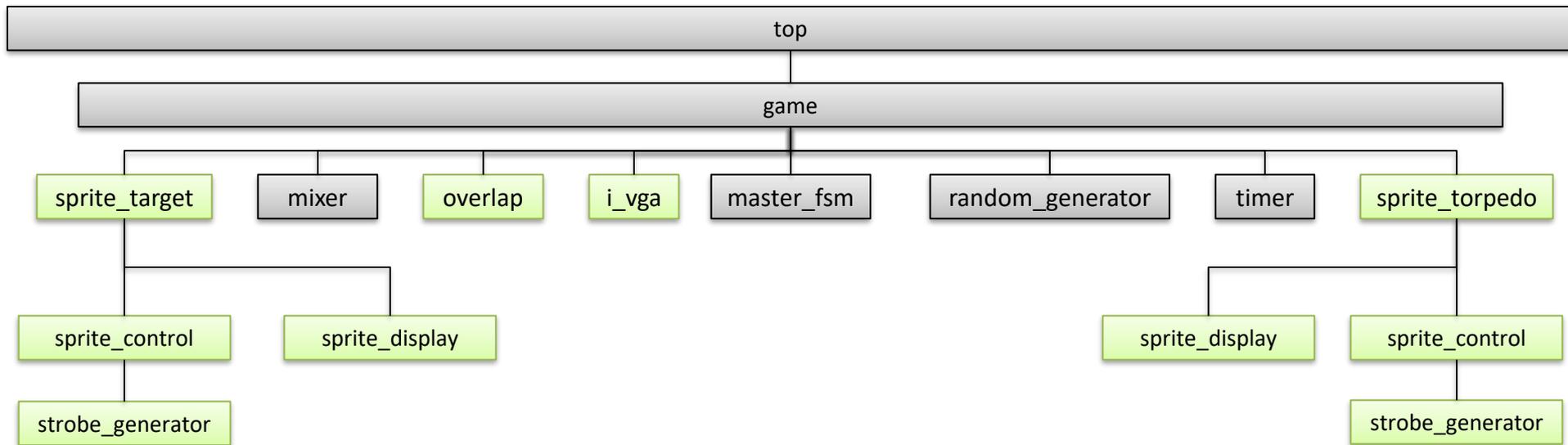
Определяем пересечение объектов

```
always @ (posedge clk or posedge reset)
    if (reset)
        overlap <= 1'b0;
    else
        overlap <= ! (    right_1 < left_2
                        || right_2 < left_1
                        || bottom_1 < top_2
                        || bottom_2 < top_1 );
```

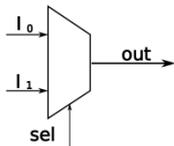


# Иерархия проекта

Что за чем?

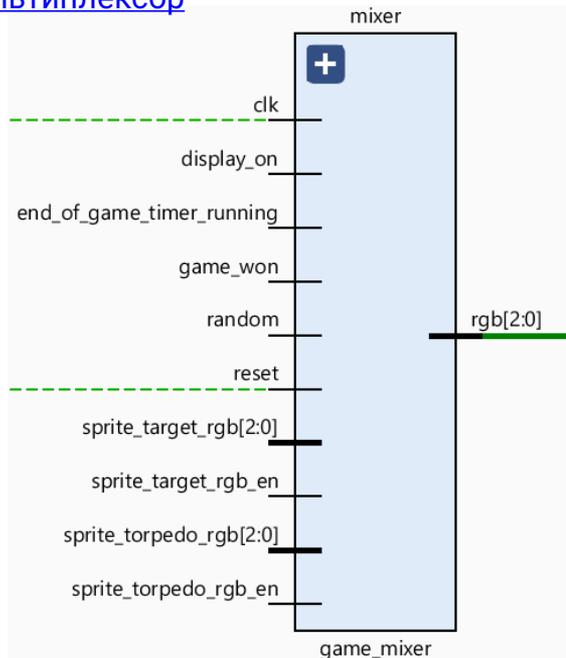


# Mixer



Взболтать, но не смешивать

Мультиплексор

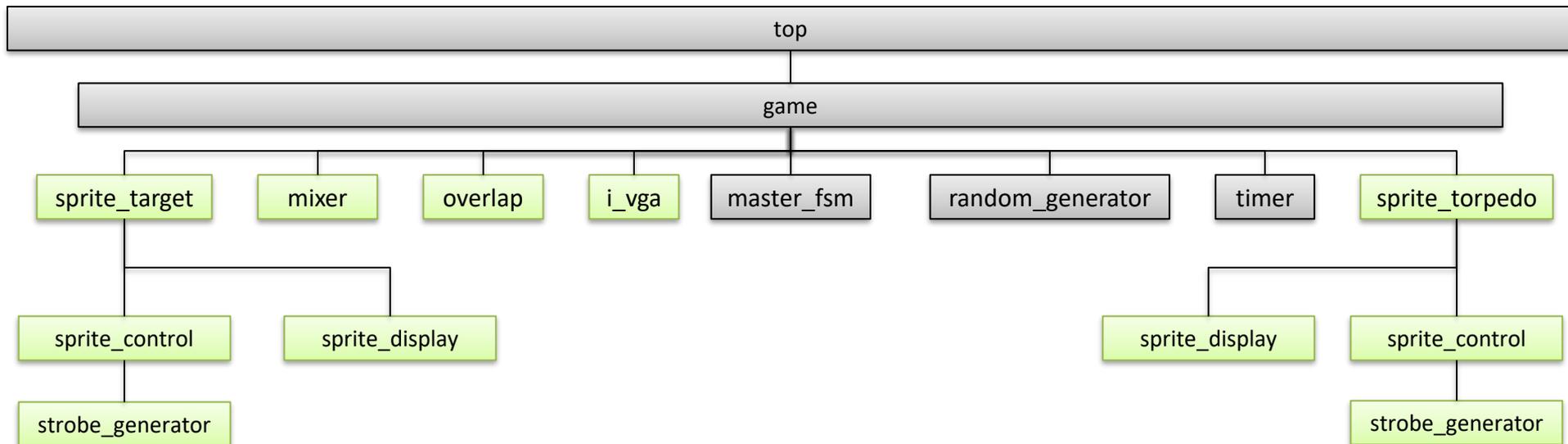


```
always @ (posedge clk or posedge reset)
    if (reset)
        rgb <= 3'b000;
    else if (! display_on)
        rgb <= 3'b000;
    else if (end_of_game_timer_running)
        rgb <= { 1'b1, ~ game_won, random };
    else if (sprite_torpedo_rgb_en)
        rgb <= sprite_torpedo_rgb;
    else if (sprite_target_rgb_en)
        rgb <= sprite_target_rgb;
    else
        rgb <= 3'b000;
```



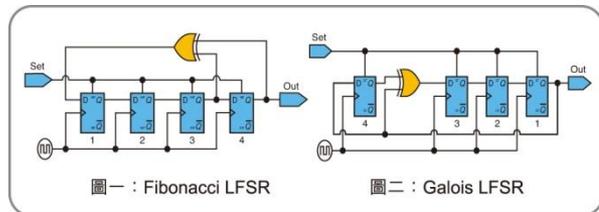
# Иерархия проекта

Что за чем?

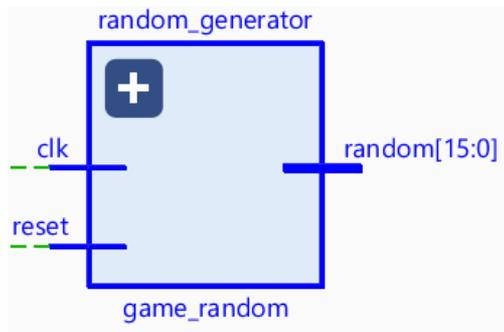


# Генератор псевдослучайной последовательности

## Случайности не случайны



Регистр сдвига с линейной обратной связью (РСЛОС, англ. **Linear Feedback Shift Register**, [LFSR](#))

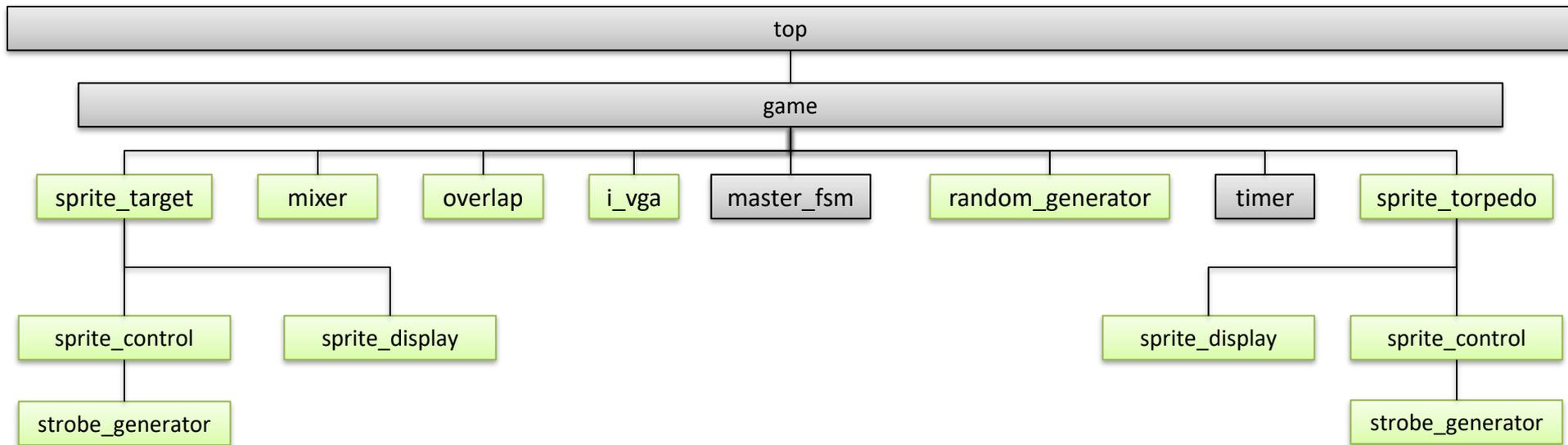


```
always @(posedge clk or posedge reset)
    if (reset)
        random <= 16'b11111111111111;
    else
        random <= { random [14:0], 1'b0 }
            ^ ( random [15] ? 16'b100000001011 : 16'b0);
```



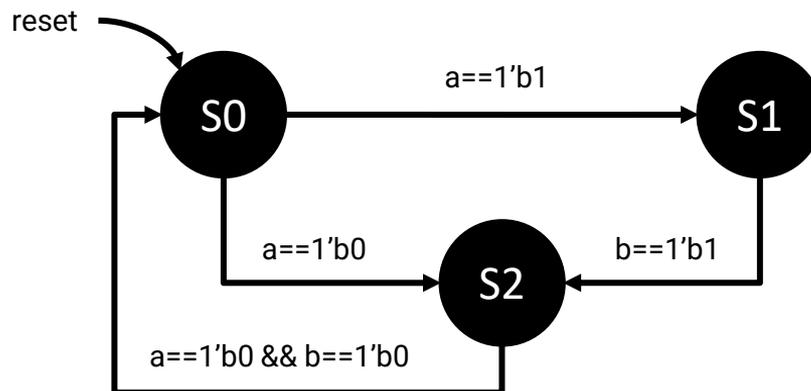
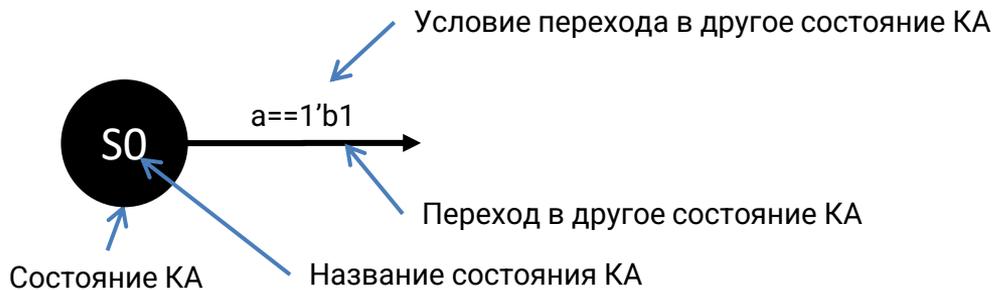
# Иерархия проекта

Что за чем?



# Конечные автоматы

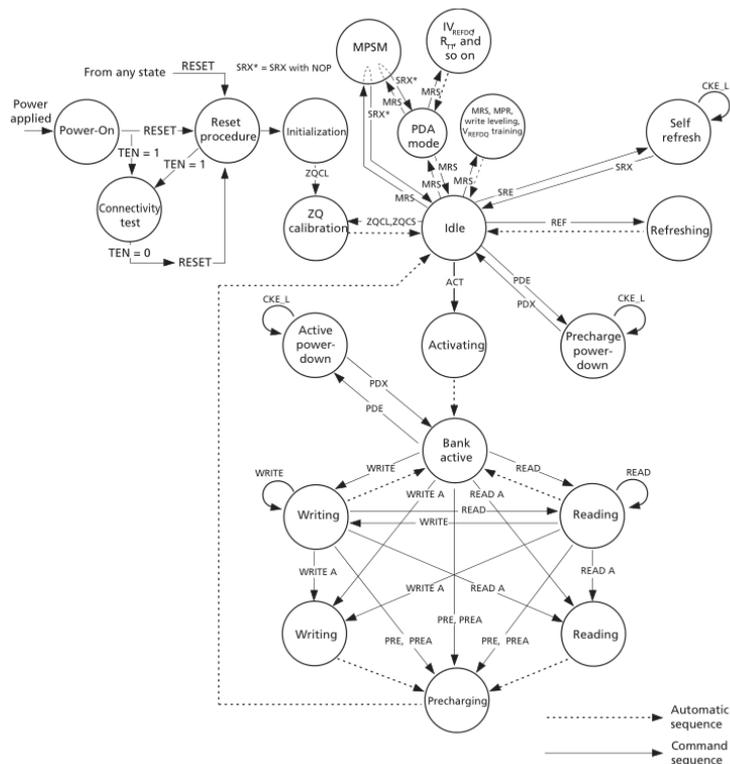
# Finite State Machine



Конечный автомат состояний абстрактного устройства

# Конечные автоматы

Figure 10: Simplified State Diagram

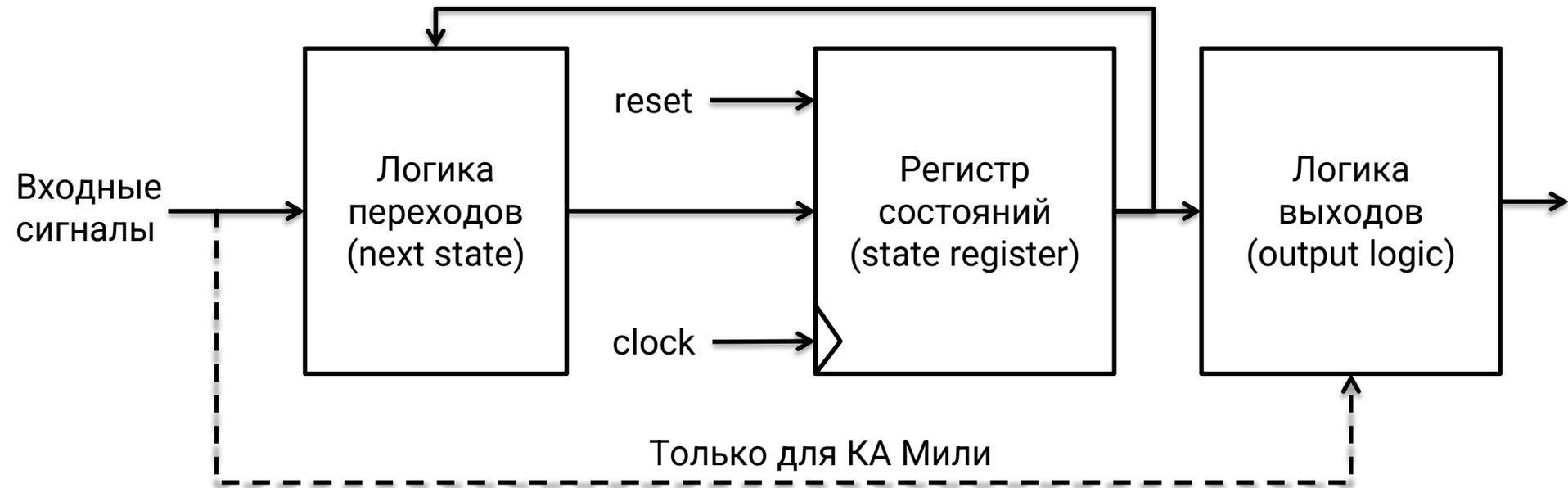


Везде и всюду

Граф состояний конечного автомата  
работы с памятью DDR4  
[MT40A1G8WE-083E](#)

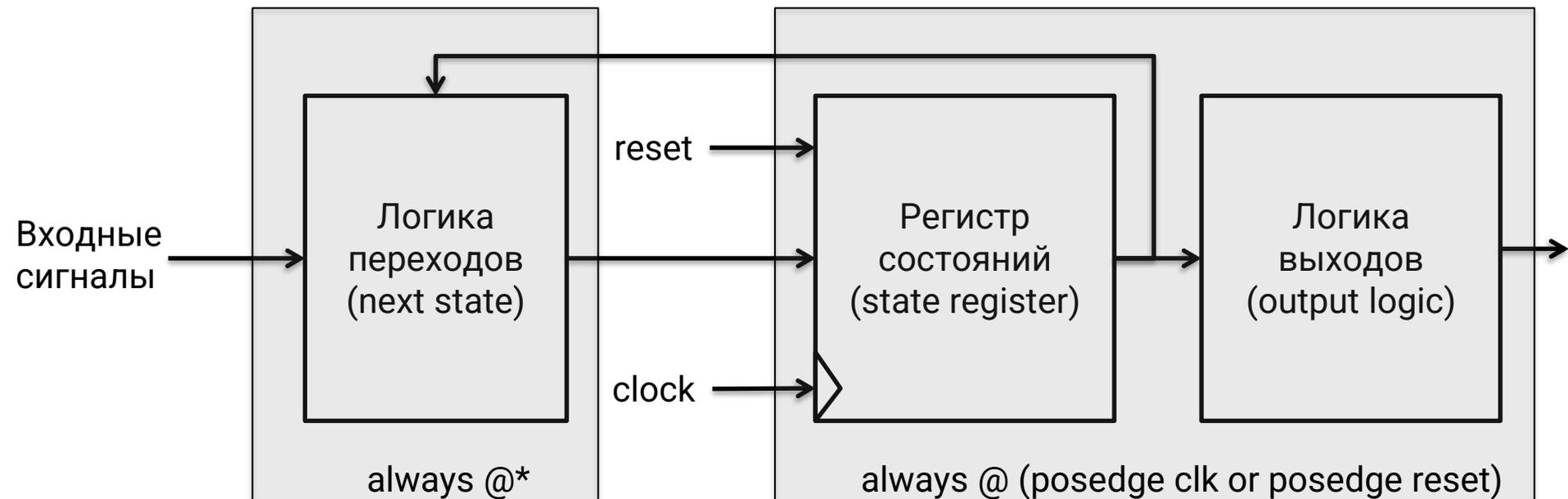
# Конечные автоматы

Как они устроены



# Конечные автоматы

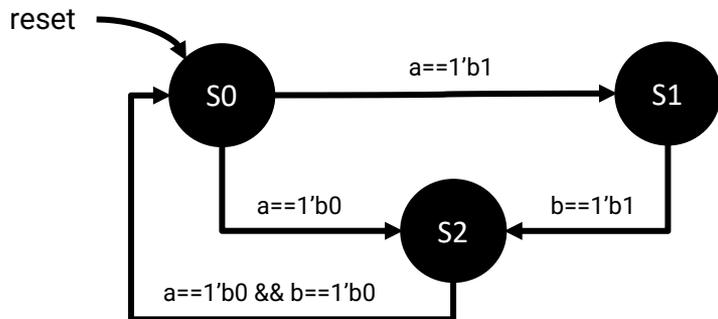
always x1, always x2 или always x3?



\*Подробнее о стилях описания конечных автоматов: [1](#), [2](#), [3](#), [4](#) (стр 270)

# Конечные автоматы

## Как это выглядит в Verilog



### Кодирование состояний КА

```
localparam [1:0] S0 = 2'b00,
                S1 = 2'b01,
                S2 = 2'b10;
```

### Регистр состояний КА

```
reg [1:0] state, n_state;
```

### Обновление состояний КА

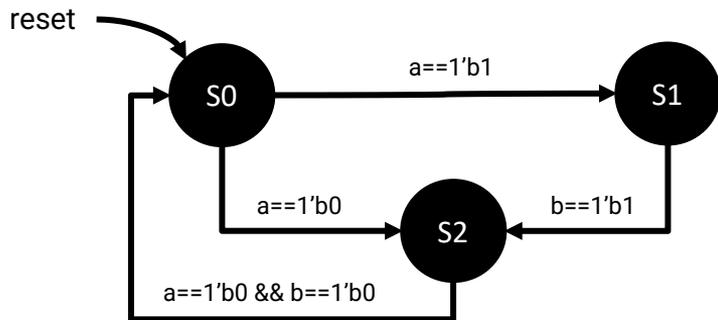
```
always@(posedge clk) begin
    if (reset) begin
        state <= S0;
    end else begin
        state <= n_state;
    end
end
```

### Логика следующего состояния

```
always @* begin
    n_state = state;
    case(state)
        S0 : if (a) begin
                n_state = S1;
            end else begin
                n_state = S2;
            end
        S1 : if (b) begin
                n_state = S2;
            end
        S2 : if (a==1'b0 && b==1'b0) begin
                n_state = S0;
            end
        default: n_state = S0;
    endcase
end
```



# Конечные автоматы



```
localparam [1:0] S0 = 2'b00,  
                S1 = 2'b01,  
                S2 = 2'b10;
```

Binary coding style

```
localparam [2:0] S0 = 3'b001,  
                S1 = 3'b010,  
                S2 = 3'b100;
```

One-hot

```
localparam [3:0] S0 = 4'b1110,  
                S1 = 4'b1101,  
                S2 = 4'b0110;
```

Пользовательский вариант

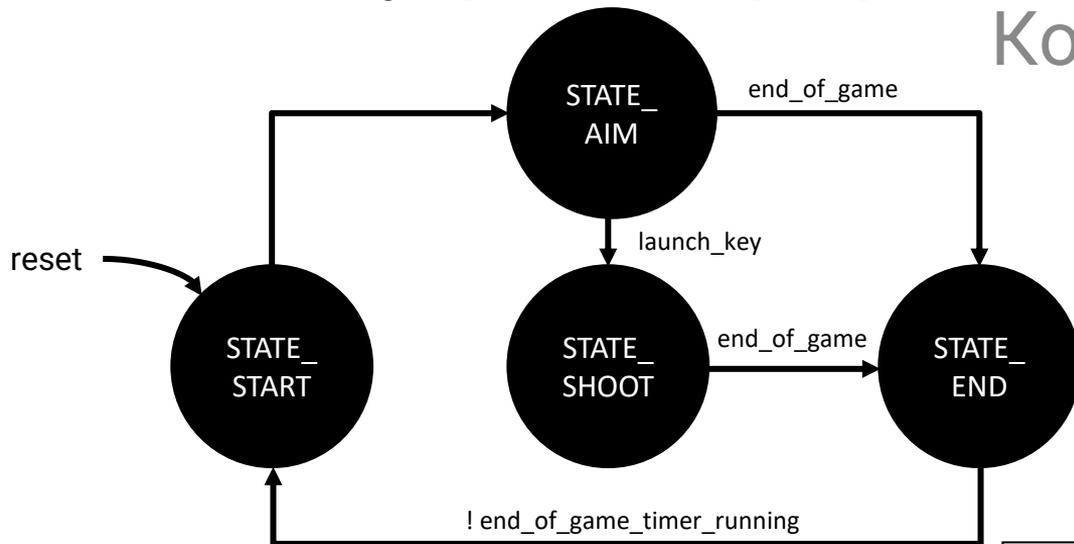
## Кодирование состояний

Отличие бинарного кодирования, кода Грея и кода Джонсона

Число	Бинарный код	Код Грея	Код Джонсона
0	0000	0000	0000
1	0001	0001	0001
2	0010	0011	0011
3	0011	0010	0111
4	0100	0110	1111
5	0101	0111	1110
6	0110	0101	1100
7	0111	0100	1000

# Главный управлятор проекта

## Конечный автомат в игре



```
localparam [1:0] STATE_START = 0,
                STATE_AIM   = 1,
                STATE_SHOOT  = 2,
                STATE_END    = 3;
```

```
wire end_of_game
    = ~ sprite_target_within_screen
      | ~ sprite_torpedo_within_screen
      | collision;
```

```
STATE_START:
begin
    d_sprite_target_write_xy   = 1'b1;
    d_sprite_torpedo_write_xy  = 1'b1;

    d_sprite_target_write_dxy  = 1'b1;

    d_game_won                 = 1'b0;

    d_state = STATE_AIM;
end
```

```
STATE_AIM:
begin
    d_sprite_target_enable_update = 1'b1;

    if (end_of_game)
    begin
        d_end_of_game_timer_start = 1'b1;
    end
    d_state = STATE_END;
end
else if (launch_key)
begin
    d_state = STATE_SHOOT;
end
end
```

```
STATE_SHOOT:
begin
    d_sprite_torpedo_write_dxy   = 1'b1;

    d_sprite_target_enable_update = 1'b1;
    d_sprite_torpedo_enable_update = 1'b1;

    if (collision)
        d_game_won = 1'b1;

    if (end_of_game)
    begin
        d_end_of_game_timer_start = 1'b1;
    end

    d_state = STATE_END;
end
```

```
STATE_END:
begin
    // TODO: Investigate why it needs collision detection here
    // and not in previous state

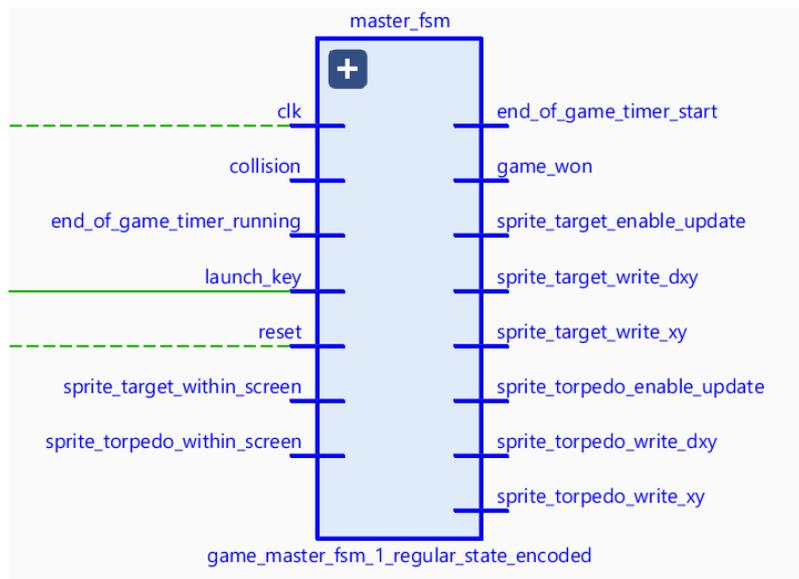
    if (collision)
        d_game_won = 1'b1;

    if (!end_of_game_timer_running)
        d_state = STATE_START;
end
```

# Главный управлятор проекта

## Конечный автомат в игре

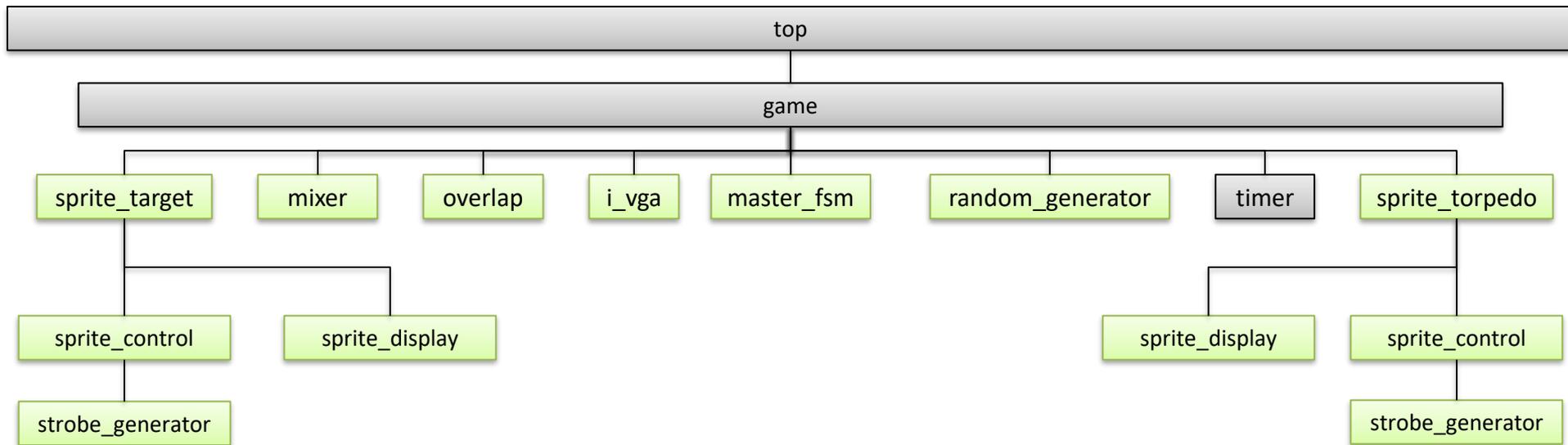
game\_master\_fsm\_1\_regular\_state\_encoded.v



- `game_master_fsm_2_special_style_one_hot` (`game_master_fsm_2_special_style_one_hot.v`)
- `game_master_fsm_3_special_style_signals_from_state` (`game_master_fsm_3_special_style_signals_from_state.v`)

# Иерархия проекта

Что за чем?



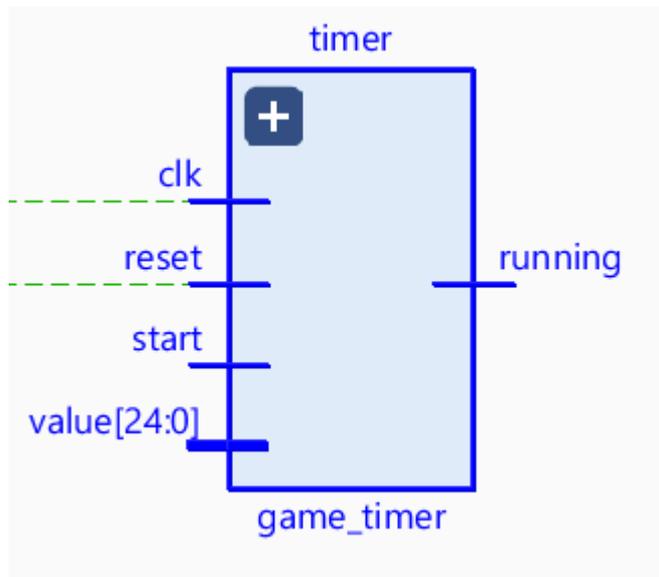
# Timer

## Сколько длится хаос?

game\_timer.v

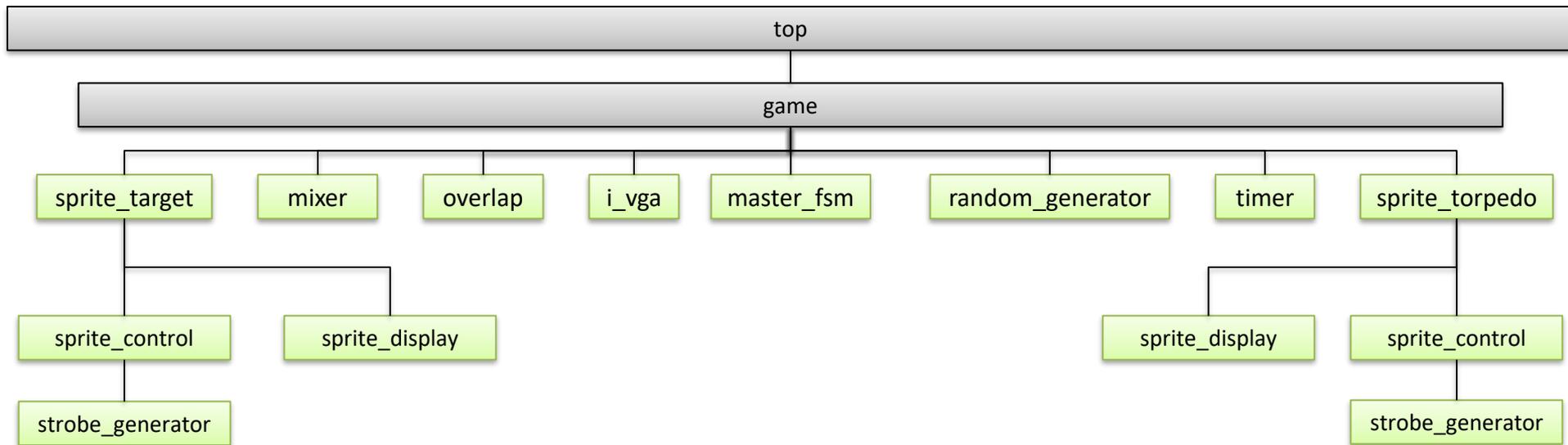
```
always @(posedge clk or posedge reset)
  if (reset)
    begin
      running <= 1'b0;
    end
  else if (start)
    begin
      counter <= value;
      running <= 1'b1;
    end
  else if (running)
    begin
      if (counter == { width, 1'b0 })
        running <= 1'b0;

      counter <= counter - 1;
    end
end
```



# Иерархия проекта

Что за чем?



# Модули верхнего уровня

## Модуль верхнего уровня top.v

```
//Nexus A7 wrapper
module top
# (
    parameter clk_mhz = 100,
               strobe_to_update_xy_counter_width = 20
)
(
    input      clk,
    input      reset_n,

    input  [3:0] key_sw,
    output [3:0] led,

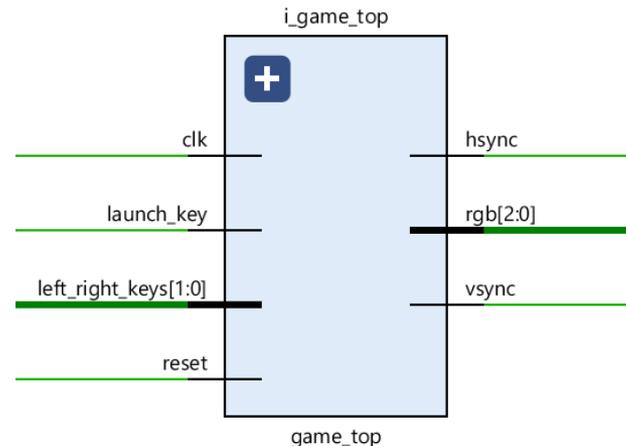
    //output [7:0] abcdefgh,
    //output [3:0] digit,

    //output      buzzer,

    output      hsync,
    output      vsync,
    output [2:0] rgb
);
```

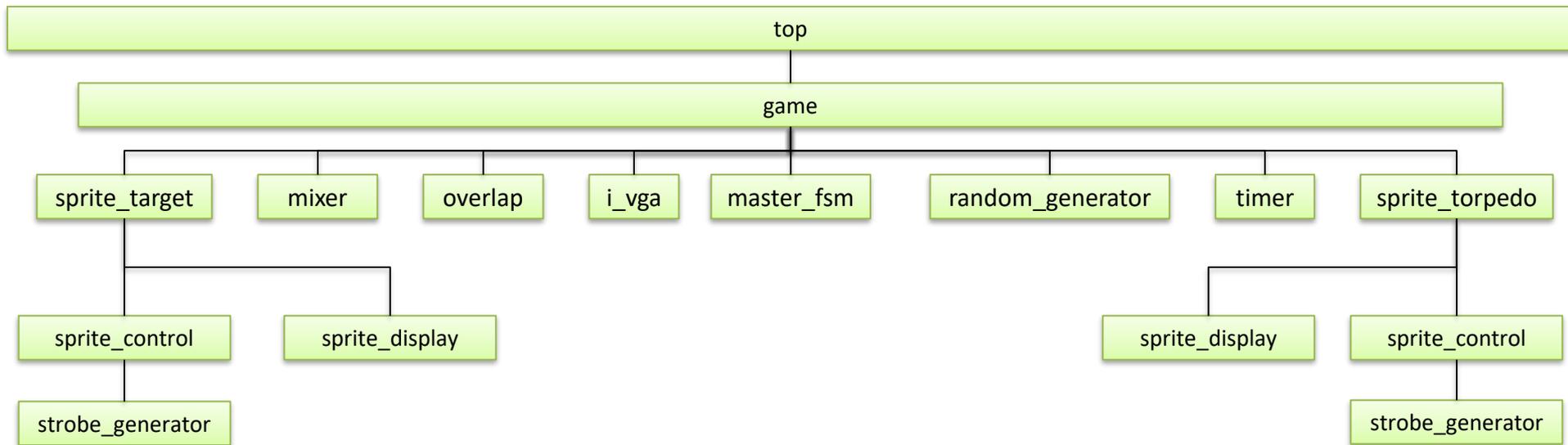
## top и game\_top

### Модуль игры game\_top.v



# Иерархия проекта

Что за чем?

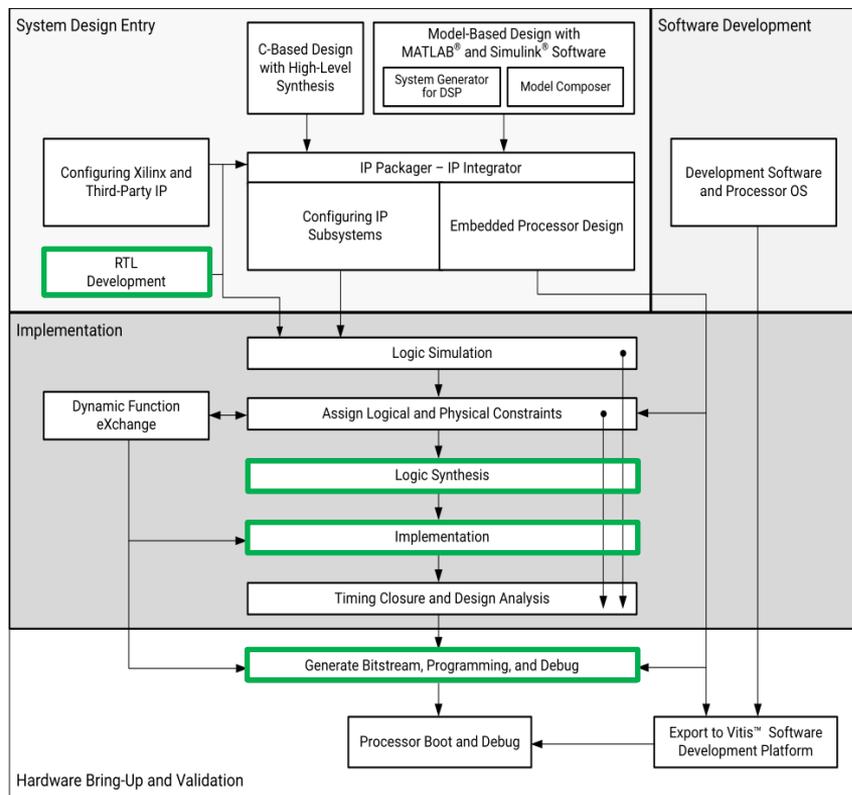


# Маршрут проектирования в Vivado

## Что за чем следует?

Это полный вариант маршрута проектирования в Vivado. Однако, в рамках школы мы рассмотрим только базовый маршрут проектирования для RTL, который включает в себя:

- создание проекта
- добавление исходников
- Elaborate представление
- Синтез
- Имплементация
- Создание конфигурационного файла



# Vivado – экспресс курс

## Графический интерфейс

The screenshot shows the Vivado IDE interface with several key components highlighted by red boxes and labeled with red text:

- Menu:** The top menu bar is labeled "Меню".
- Flow Navigator:** The left sidebar is labeled "Маршрут проектирования" (Design Flow).
- Project Manager:** The central pane shows the project hierarchy, labeled "Иерархия проекта".
- Properties:** The pane below the Project Manager is labeled "Свойства выбранного объекта" (Properties of the selected object).
- Project Summary:** The right pane is labeled "Основное рабочее поле" (Main workspace) and shows project details and synthesis/implementation status.
- Console:** The bottom pane is labeled "Консоль, ошибки, отчеты, логи" (Console, errors, reports, logs) and displays a table of design runs.

**Project Summary Details:**

**Overview** | Dashboard

**Settings** | Edit

Project name: game\_xilinx  
Project location: D:/Projects/ce2020labs-master/game\_xilinx  
Product family: Artix-7  
Project part: xc7a100tcsg324-1  
Top module name: top  
Target language: Verilog  
Simulator language: Mixed

Synthesis	Implementation	Summary	Route Status
Status: <span style="color: green;">✔</span> Complete	Status: <span style="color: green;">✔</span> Complete		
Messages: <span style="color: orange;">!</span> 16 critical warnings <span style="color: yellow;">!</span> 16 warnings	Messages: <span style="color: orange;">!</span> 16 critical warnings <span style="color: yellow;">!</span> 16 warnings		
Part: xc7a100tcsg324-1	Part: xc7a100tcsg324-1		
Strategy: Vivado Synthesis Defaults	Strategy: Vivado Implementation Defaults		
Report Strategy: Vivado Synthesis Default Reports	Report Strategy: Vivado Implementation Default Reports		

**Design Runs Console:**

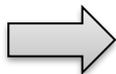
Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
✓ synth_1	constrs_1	synth_design Complete!								298	282	0.0	0	0	9/2/20, 9:33 PM	00:00:58	Vivado Synthesis Defaults*
✓ impl_1	constrs_1	route_design Complete!	4.860	0.000	0.155	0.000	0.000	0.125	0	296	282	0.0	0	0	9/2/20, 9:36 PM	00:01:45	Vivado Implementation De

# Vivado – экспресс курс

## Show Hierarchy

Flow Navigator

- PROJECT MANAGER
  - Settings
  - Add Sources
  - Language Templates
  - IP Catalog
- IP INTEGRATOR
  - Create Block Design
  - Open Block Design
  - Generate Block Design
- SIMULATION
  - Run Simulation
- RTL ANALYSIS
  - Open Elaborated Design**
  - Report Methodology
  - Report DRC
  - Schematic

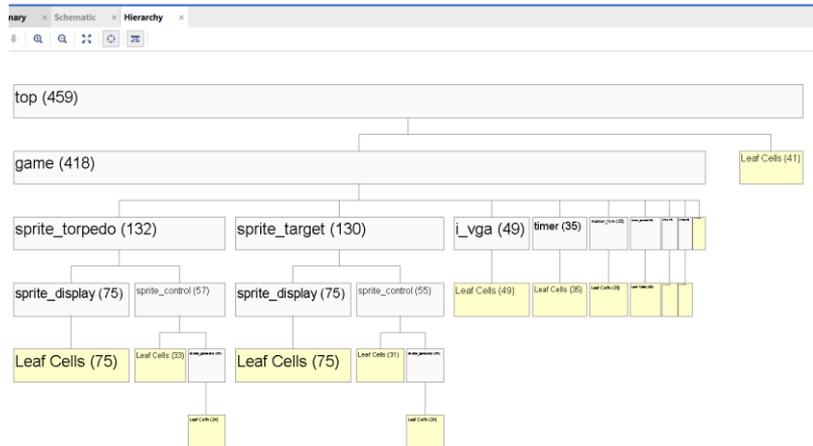
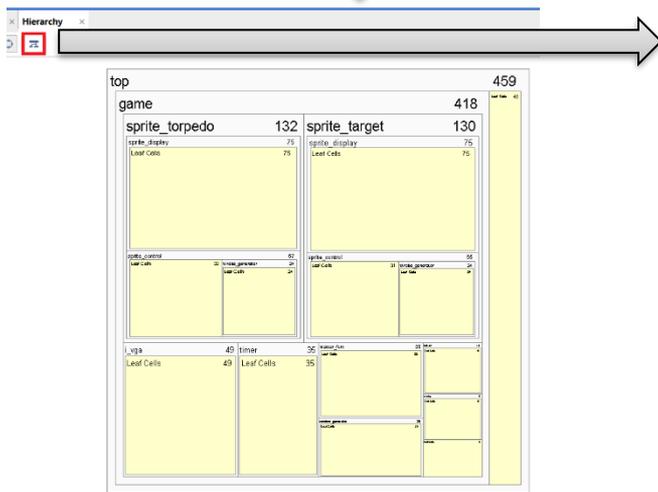


game\_xilinx - [D:/Projects/ce2020labs-master/game\_xilinx/game\_xilinx.xpr] - Vivado 2020.1

File Edit Flow Tools Reports Window Layout View Help

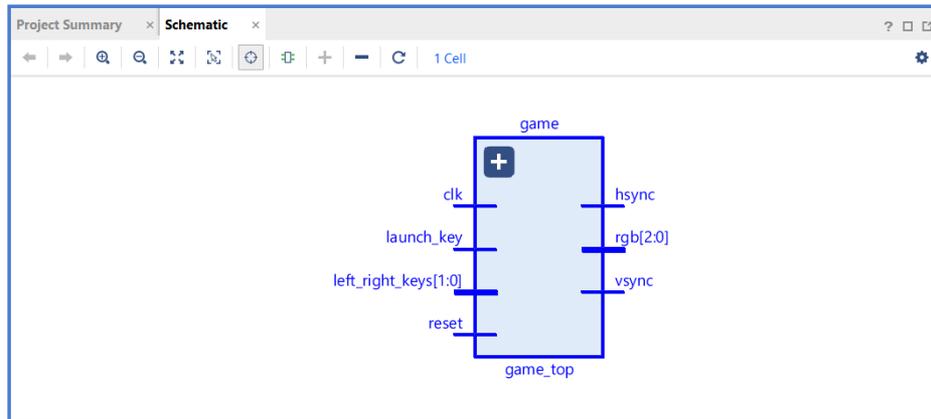
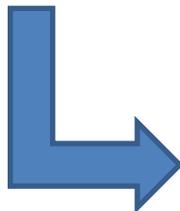
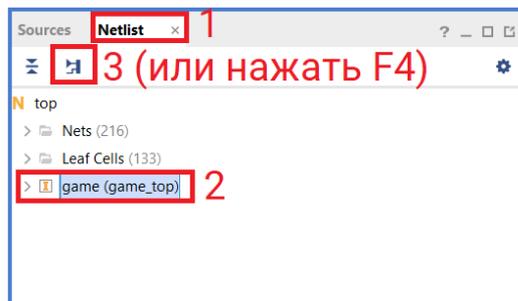
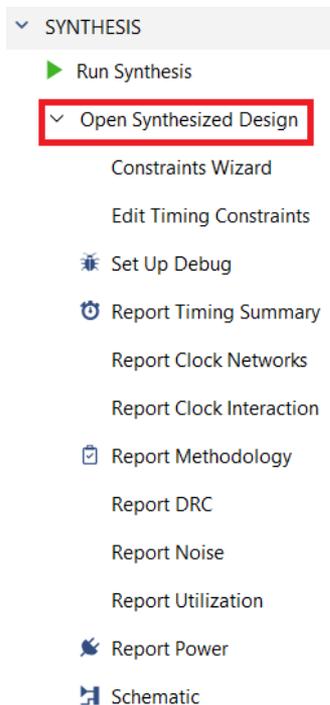
Tools

- Floorplanning
- I/O Planning
- Schematic F4
- Show Connectivity Ctrl+T
- Show Hierarchy F6



# Пост синтез

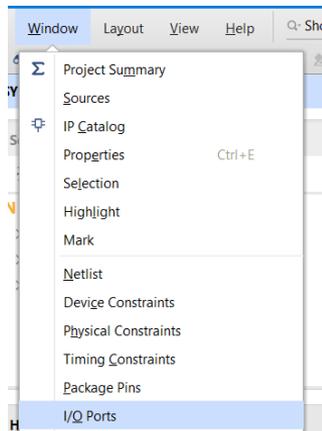
Из кода в регистры и логику



# Пост синтез

## Подключение ножек ПЛИС к модулю

- SYNTHESIS
- Run Synthesis
- Open Synthesized Design**
- Constraints Wizard
- Edit Timing Constraints
- Set Up Debug
- Report Timing Summary
- Report Clock Networks
- Report Clock Interaction
- Report Methodology
- Report DRC
- Report Noise
- Report Utilization
- Report Power
- Schematic



write\_bitstream Complete  
I/O Planning

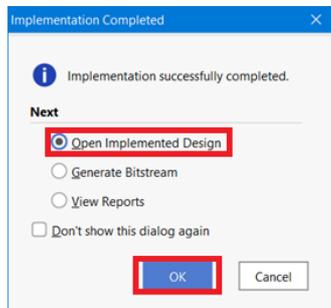
SYNTHESIZED DESIGN - xc7a100cpg324-1

Package: Device

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Te
> All ports (66)												
> an (8)	OUT			<input checked="" type="checkbox"/>	(Multiple)	LVC MOS33*	3.300	12	↓	SLOW	NONE	FP_VTT_50
> led (16)	OUT			<input checked="" type="checkbox"/>	(Multiple)	LVC MOS33*	3.300	12	↓	SLOW	NONE	FP_VTT_50
> rgb (3)	OUT			<input checked="" type="checkbox"/>	35	LVC MOS33*	3.300	12	↓	SLOW	NONE	FP_VTT_50
> sw (16)	IN			<input checked="" type="checkbox"/>	(Multiple)*	(Multiple)*	(Multiple)				NONE	NONE
> Scalar ports (23)												

# Пост имплементация

## Как проект выглядит внутри ПЛИС?



IMPLEMENTED DESIGN - xc7a100tcsq324-1  
Delete (Delete)

Sources Netlist

- top
  - Nets (192)
  - Leaf Cells (133)
  - game (game\_top)

Properties

Select an object to see properties

Project Summary x Device x

Timing

Design Timing Summary

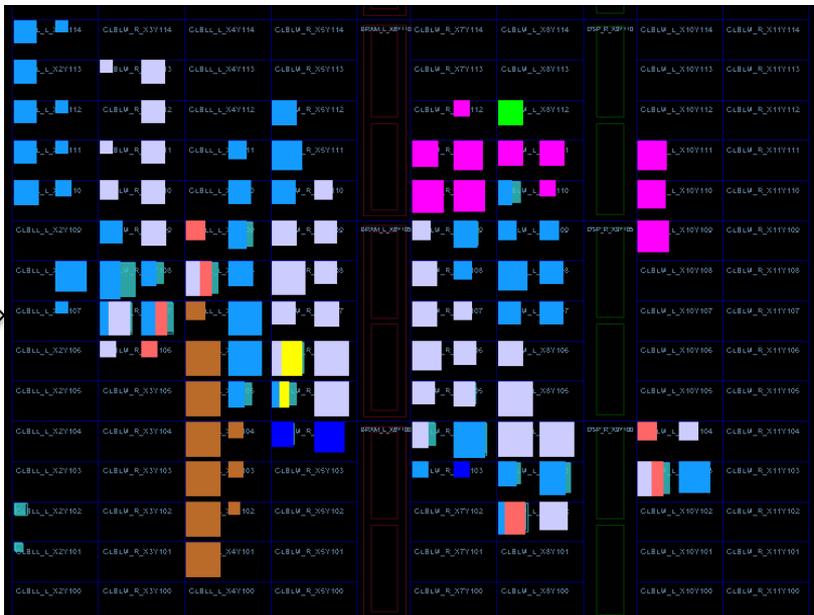
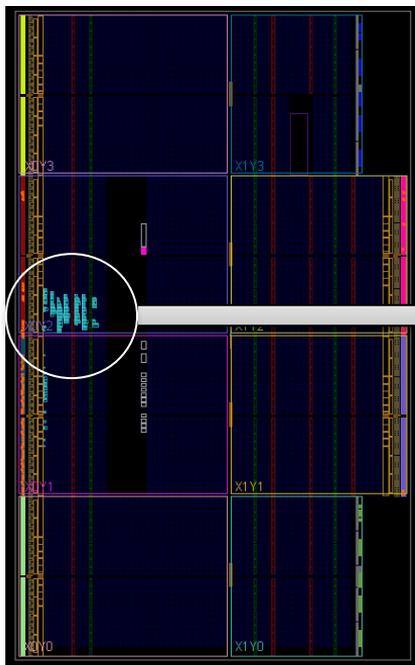
Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4,860 ns	Worst Hold Slack (WHS): 0,155 ns	Worst Pulse Width Slack (WPWS): 4,500 ns
Total Negative Slack (TNS): 0,000 ns	Total Hold Slack (THS): 0,000 ns	Total Pulse Width Negative Slack (TPWS): 0,000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 367	Total Number of Endpoints: 367	Total Number of Endpoints: 283

All user specified timing constraints are met.

Timing Summary - impl\_1 (saved)

# Пост имплементация

## Как проект выглядит внутри ПЛИС?



top

- > Nets (192)
- > Leaf Cells (133)
- ▼ game (game\_top)
  - > Nets (143)
  - > Leaf Cells (6)
  - > i\_vga (vga)
  - > master\_fsm (game\_master\_fsm\_1\_regular\_state\_encoded)
  - > mixer (game\_mixer)
  - > overlap (game\_overlap)
  - > random\_generator (game\_random)
  - > sprite\_target (game\_sprite\_top)
  - > sprite\_torpedo (game\_sprite\_top\_parameterized0)
  - > timer (game\_timer)

\*Для закрашки: правой кнопкой мыши по модулю в Netlist -> Highlight Leaf Cells -> Выбрать цвет

# Пост имплементация

## Сколько затрачено ресурсов ПЛИС?

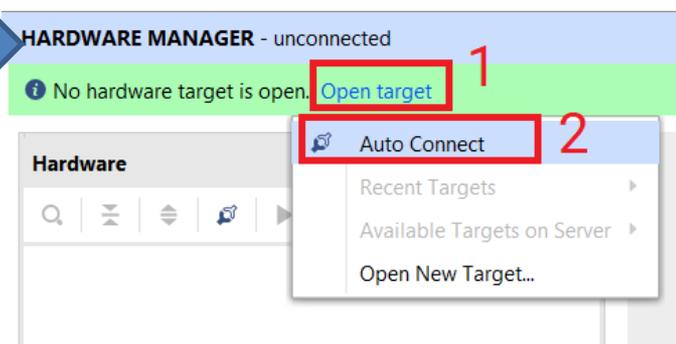
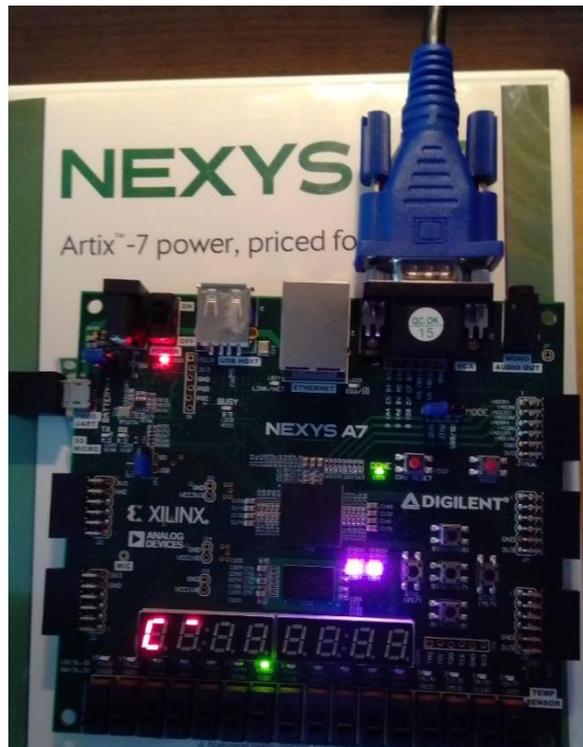
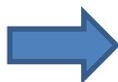
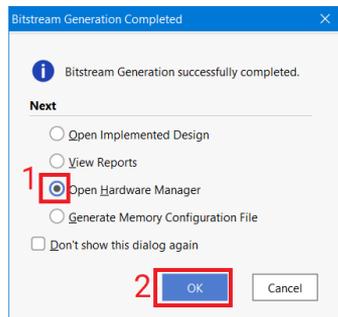
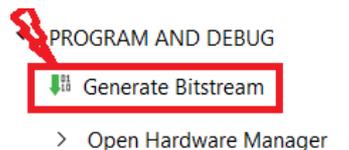
Сводная таблица занятых ресурсов (Report Utilization)

- ▼ IMPLEMENTATION
  - ▶ Run Implementation
  - ▼ Open Implemented Design
    - Constraints Wizard
    - Edit Timing Constraints
    - 🕒 Report Timing Summary
    - Report Clock Networks
    - Report Clock Interaction
    - 📄 Report Methodology
    - Report DRC
    - Report Noise
    - Report Utilization**
    - 🔌 Report Power
    - 🔍 Schematic

Name	Slice LUTs (63400)	Slice Registers (126800)	Slice (15850)	LUT as Logic (63400)	Bonded IOB (210)	BUFGCTRL (32)
▼ N top	296	282	126	296	66	1
▼ 📁 game (game_top)	278	250	106	278	0	0
📁 i_vga (vga)	39	28	12	39	0	0
📁 master_fsm (game_master_fsm_1_regular_state_encoded)	4	7	2	4	0	0
📁 mixer (game_mixer)	4	3	2	4	0	0
📁 overlap (game_overlap)	21	1	9	21	0	0
📁 random_generator (game_random)	2	16	5	2	0	0
> 📁 sprite_target (game_sprite_top)	84	80	42	84	0	0
> 📁 sprite_torpedo (game_sprite_top__parameterized0)	89	89	42	89	0	0
📁 timer (game_timer)	32	26	11	32	0	0

# Последний шаг

## Создание конфигурационного файла



# Последний шаг

## Создание конфигурационного файла

**HARDWARE MANAGER** - localhost/xilinx\_tcf/Digilent/210292AA7AE9A

There are no debug cores. **2** [Program device](#) Refresh device

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210292AA7AE9A	Open
<b>xc7a100t_0 (1)</b>	<b>1</b> Programmed
XADC (System Monitor)	

**Program Device**

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: **3**

Debug probes file:

Enable end of startup check

**4**

# Что будет дальше?

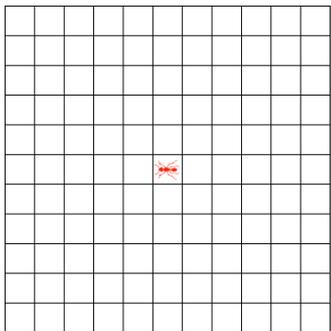
## Какие возможны улучшения?

1. Добавьте еще один спрайт == +1 к карме
2. Сделайте счетчик побед и поражений с выводом результата на экран
3. Добавьте второго игрока
4. Игра в догонялки
5. Расширьте функционал дополнительными модулями
6. Погрузитесь в исследования. Постройте генератор шума на физически неклонированной функции ([PUF](#))

# Клеточные автоматы

## Жизнь муравья

Муравей Лэнгтона — это двумерный клеточный автомат с очень простыми правилами, изобретенный Крисом Лэнгтоном[1]. [Wiki](#)



[Ссылка на стрим](#)



[Ссылка на стрим](#)

# Где найти FPGA комьюнити?



[fpga-systems.ru](http://fpga-systems.ru)



[t.me/fpgasystems](https://t.me/fpgasystems)  $\Leftrightarrow$  [@fpgasystems](https://t.me/fpgasystems)



[youtube.com/c/fpgasystems](https://youtube.com/c/fpgasystems)



[admin@fpga-systems.ru](mailto:admin@fpga-systems.ru)



# Приходи на следующую конференцию FPGA разработчиков

[fpga-systems.ru/meet](https://fpga-systems.ru/meet)

Москва

МИНСК

ТОМСК

Санкт-Петербург

