

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное автономное образовательное учреждение высшего образования  
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Кафедра вычислительных систем и сетей  
(наименование)

ОТЧЁТ ПО ПРАКТИКЕ  
ЗАЩИЩЁН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Доцент, д-р т.н., доцент

должность, уч. степень, звание

*Обиши*

*[Подпись]*

подпись, дата

С.Т. Хвош

инициалы, фамилия

ОБЩИЙ ОТЧЁТ ПО ПРАКТИКЕ

вид практики производственная

тип практики технологическая (проектно-технологическая)

на тему индивидуального задания Стандартное задание

выполнен Тегай Екатериной Дмитриевной, Пономаревым Дмитрием Валерьевичем,  
Шалевым Степаном Андреевичем  
фамилия, имя, отчество обучающегося в творительном падеже

по направлению подготовки 09.03.01  
код

Информатика и вычислительная  
наименование направления

техника

наименование направления

направленности

03

код


Компьютерные технологии, системы и сети  
наименование направленности

наименование направленности

Обучающиеся группы №

4143

номер



20.07.2024

подпись, дата

Тегай Е. Д.

инициалы, фамилия

4143

номер



20.07.2024

подпись, дата

Пономарев Д. В.

инициалы, фамилия

4143

номер



20.07.2024

подпись, дата

Шалев С. А.

инициалы, фамилия

## Цель работы

Разработать проект ПЛИС для формирования 32-битного (для формирования слова ARINC429) тактирующего сигнала с частотами 12.5 кГц, 48 кГц и 100 кГц (и паузой длительностью 4 периода) при входной частоте тактирования 48 МГц для программируемой логической интегральной схемы (ПЛИС) семейства Cyclone II с использованием среды разработки Quartus.

## Ход работы

Начало выполнения работы заключалось в создании нового проекта в среде Quartus, где в качестве семейства ПЛИС был выбран Cyclone II согласно заданию, а также сама ПЛИС модели EP2C8Q208C8.

В задании фигурирует такое понятие, как ARINC429. Это стандарт авионики, который определяет протокол обмена данными между различными системами авионики и компонентами в самолёте. Стандарт поддерживает две основные скорости передачи данных: 12,5 кбит/с (низкая) и 100 кбит/с (высокая). Каждое сообщение представляет собой 32-битное слово. Первые 8 бит используются для адреса получателя (идентификатор канала), следующие 19 бит – для данных, 2 бита для управления и 3 бита для контроля ошибки (контроль чётности). Стандарт использует принцип однонаправленной передачи данных, также данные передаются циклически, а приёмники непрерывно прослушивают шину данных, фильтруя нужные им сообщения по идентификатору канала.

Основным этапом работы является разработка программы. Для начала в программе объявляется модуль, который и будет использоваться для формирования выходного сигнала с определённой частотой и заданной последовательностью импульсов. Модуль принимает несколько входных сигналов и формирует три выходных сигнала. Рассмотрим их подробнее.

- **clk\_48MHz** – входной тактовый сигнал с частотой 48 МГц, который используется для синхронизации работы всех внутренних блоков модуля.

- **freq\_sel** - двухбитный входной сигнал для выбора частоты выходного сигнала. В зависимости от значения этого сигнала, модуль будет формировать выходной сигнал с частотой 12,5 кГц, 48 кГц или 100кГц.
- **ENA** - входной сигнал разрешения. Если этот сигнал активен, то модуль формирует выходные импульсы. А если неактивен, то формирование импульсов останавливается.
- **arinc\_clk** – выходной тактовый сигнал, частота которого выбирается в зависимости от значения freq\_sel. Этот сигнал используется для синхронизации генерации выходного сигнала out.
- **out** - основной выходной сигнал, который формируется в виде последовательности из 32 импульсов, за которыми следует пауза из 4 тактов, в зависимости от выбранной частоты.
- **masked\_out** – дополнительный выходной сигнал, который является результатов логического «И» сигналов out и arinc\_clk. Этот сигнал может быть использован для дальнейшей обработки или маскирования выходного сигнала.

Далее идёт определение регистров. Рассмотрим их подробнее.

- **counter** – этот регистр используется для подсчёта тактовых импульсов от входного сигнала clk\_48MHz. он помогает в реализации деления частоты для создания выходного сигнала с нужной частотой.
- **max\_count** – этот регистр хранит значение, до которого counter должен досчитать перед тем, как сброситься и инвертировать состояние arinc\_clk. Значение этого регистра зависит от выбранной частоты.
- **pulse\_counter** – этот регистр используется для подсчёта числа выходных импульсов arinc\_clk. Он помогает в реализации вывода именно 32 импульсов с 4 паузными тактами.
- **en\_active** – этот регистр хранит состояние сигнала разрешения ENA. Он используется для завершения текущего цикла из 32 импульсов, даже если ENA деактивирован до завершения цикла.

Затем идёт блок кода, который определяет локальные параметры, которые представляют количество тактов, необходимых для генерации нужных частот.

После этого идёт описание выбора частоты, на которую будет делиться тактовый сигнал `clk_48MHz`, в зависимости от значения сигнала `freq_sel`. Описание заключено в блок `always`, который реагирует на каждый положительный фронт тактового сигнала `clk_48MHz`. Внутри этого блока есть ещё один блок `case`, с помощью которого реализуется конструкция выбора в зависимости от значения переменной `freq_sel`, которая является двухбитным входом. Также в случае, когда ни одна строка не подошла в отношении введённого значения, по умолчанию устанавливается значение 12,5кГц.

Далее идёт блок кода, который отвечает за генерацию сигнала `arinc_clk` с учётом внешнего сигнала `ENA` и внутренней переменной `en_active`. Здесь снова используется конструкция `always`, а также используется условный оператор `if`, которой проверяет, активен ли сигнал `ENA` или включён ли внутренний флаг `en_active`. В положительном исходе идёт переход к ещё одному условию, которое проверяет, не достигло ли значение счётчика `counter` максимального значения. Если не достигло, то значение инкрементируется, а иначе он сбрасывается в 0 и происходит инверсия сигнала `arinc_clk`. Если же сигнал `ENA` неактивен или флаг равен 0, то счётчик и `arinc_clk` сбрасываются в ноль.

Затем идёт описание генерации сигнала `out` на основе сигнала `arinc_clk` и внешнего сигнала `ENA`. Здесь сначала устанавливается внутренний флаг `en_active`, когда сигнал `ENA` активен. Затем счётчик `pulse_counter` увеличивается до 35, после чего сбрасывается, чтобы начать новый цикл. Это позволяет контролировать периоды активности и паузы сигнала `out` в зависимости от состояния сигнала `arinc_clk` и `ENA`. После этого идёт описание управления генерации выходного сигнала `out` в зависимости от текущего состояния счётчика `pulse_counter`, флага `en_active` и сигнала `ENA`. Он

обеспечивает генерацию импульсов с учётом заданного количества и пауз между ними, как требуется спецификацией ARINC429.

Завершающим блоком является описание формирования сигнала `masked_out`, который представляет собой логическое «И» между сигналами `out` и `arinc_clk`. Сигнал `masked_out` используется для указания моментов, когда происходит передача данных. Это обеспечивает корректную синхронизацию и формирование выходного сигнала в соответствии с требованиями протокола ARINC429.

## Текст программы

```
module frequency_selector (  
    input wire clk_48MHz,  
    input wire [1:0] freq_sel,  
    input wire ENA,  
    output reg arinc_clk,  
    output reg out,  
    output reg masked_out);  
  
    reg [15:0] counter;  
    reg [15:0] max_count;  
    reg [5:0] pulse_counter;  
    reg en_active;  
  
    localparam FREQ_12_5_KHZ = 16'd1920;  
    localparam FREQ_48_KHZ = 16'd500;  
    localparam FREQ_100_KHZ = 16'd240;  
  
    always @(posedge clk_48MHz) begin  
        case (freq_sel)  
            2'b00: max_count <= FREQ_12_5_KHZ;  
            2'b01: max_count <= FREQ_48_KHZ;  
            2'b10: max_count <= FREQ_100_KHZ;  
            default: max_count <= FREQ_12_5_KHZ;  
        endcase  
    end  
  
    always @(posedge clk_48MHz) begin  
        if (ENA || en_active) begin  
            if (counter < max_count - 1) begin  
                counter <= counter + 1;  
            end else begin  
                counter <= 0;  
                arinc_clk <= ~arinc_clk;  
            end  
        end else begin  
            counter <= 0;  
        end  
    end
```

```
arinc_clk <= 0;  
end  
end
```

```
always @(posedge arinc_clk) begin  
  if (ENA) begin  
    en_active <= 1;  
    if (pulse_counter < 35) begin  
      pulse_counter <= pulse_counter + 1;  
    end else begin  
      pulse_counter <= 0;  
    end  
  end  
end
```

```
  if (pulse_counter < 32) begin  
    out <= 1;  
  end else begin  
    out <= 0;  
  end  
  end else if (en_active) begin  
    if (pulse_counter < 32) begin  
      out <= 1;  
      pulse_counter <= pulse_counter + 1;  
    end else begin  
      out <= 0;  
      pulse_counter <= 0;  
      en_active <= 0;  
    end  
  end else begin  
    out <= 0;  
  end  
end  
end
```

```
always @(posedge clk_48MHz) begin  
  masked_out <= out & arinc_clk;  
end
```

```
endmodule
```



Результаты выполнения работы

Искомые результаты продемонстрированы на рисунках 1 –

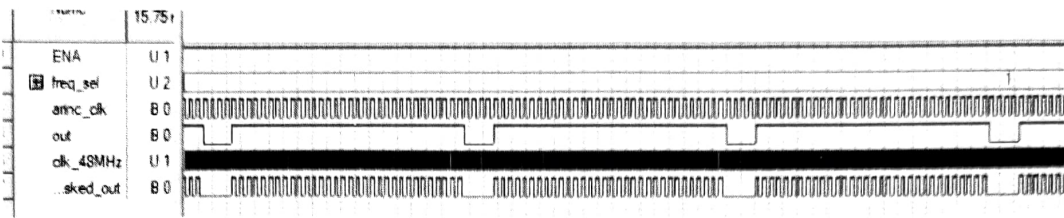


Рисунок 1 – Временная диаграмма (часть 1)

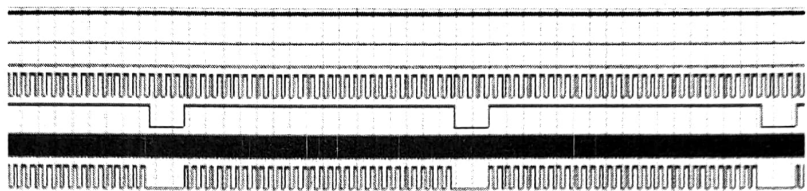


Рисунок 2 - Временная диаграмма (часть 2)

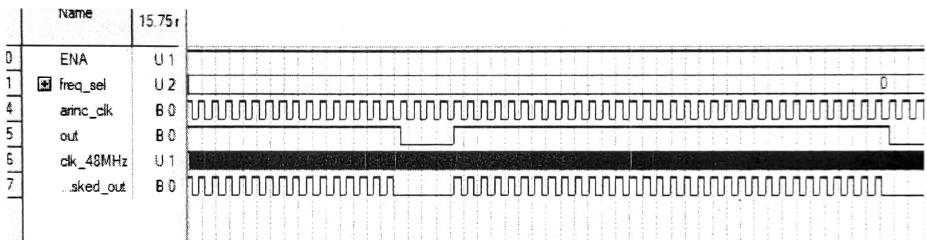


Рисунок 3 - Временная диаграмма (часть 3)

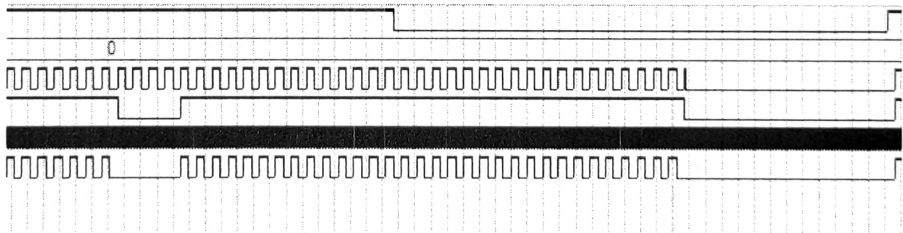


Рисунок 4 - Временная диаграмма (часть 4)

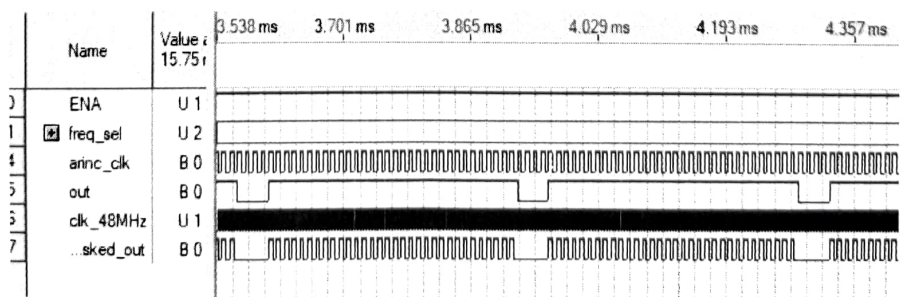


Рисунок 5 - Временная диаграмма (часть 5)

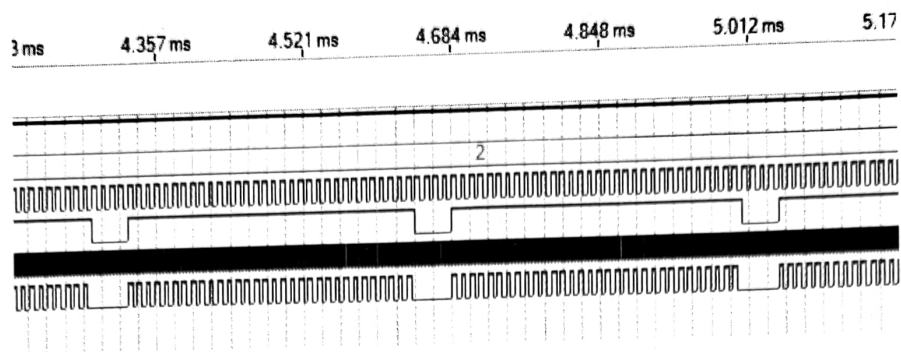


Рисунок 6 - Временная диаграмма (часть 6)

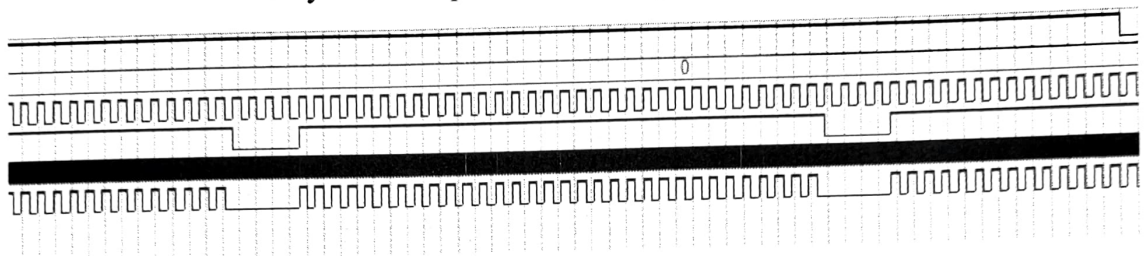


Рисунок 7 – Временная диаграмма (увеличенная, часть 1)

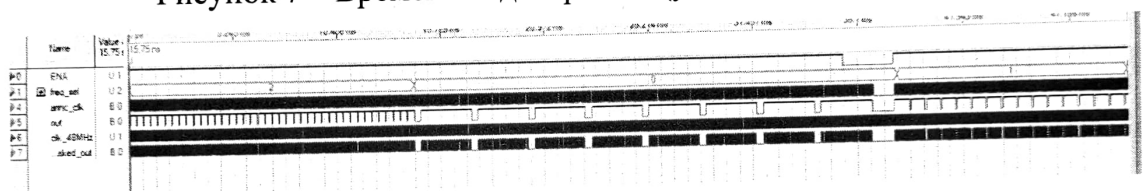


Рисунок 8 - Временная диаграмма (увеличенная, часть 2)

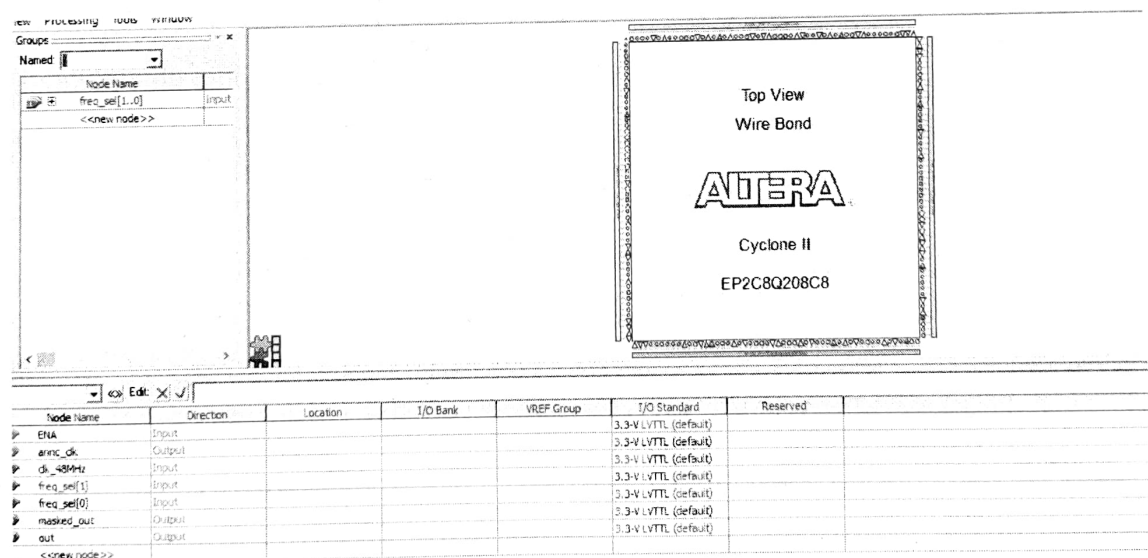


Рисунок 9 – Выбранная ПЛИС

## **Выводы**

В результате выполнения работы был разработан модуль для формирования 32-битного кода ARINC429 с паузой и частотами 12.5 кГц, 48 кГц и 100 кГц при входной частоте тактирования 48 МГц для программируемой логической интегральной схемы (ПЛИС) семейства Cyclone II с использованием среды разработки Quartus.