

# Домашнее задание №7

---

Домашнее задание №7 состоит из 7 упражнений:

- Первое в папке `07_01_circular_buffer_with_valid`
- Второе в папке `07_02_fifo_with_counter_baseline`
- Третье в папке `07_03_fifo_pow2_depth`
- Четвёртое в папке `07_04_fifo_empty_full_optimized`
- Пятое в папке `07_05_fifo_with_reg_empty_full`
- Шестое в папке `07_06_sqrt_formula_pipe`
- Седьмое в папке `07_07_sqrt_formula_pipe_using_fifo`

У некоторых упражнений есть секция `Example` с модулем для примера.

Во всех упражнениях есть секция `Task` с описанием задания и местом, где необходимо описать ваше решение.

## Предисловие

---

В файле Testbench любого из заданий можно убрать комментарий у строк `$dumpfile;` и `$dumpvars;` для генерации `dump.vcd` файла. В файле будут содержаться текстовые описания временной диаграммы, описывающей изменения на всех проводах и регистрах во время симуляции.

Можно воспользоваться командой `gtkwave dump.vcd` для просмотра файла, либо добавить опцию `--wave` или `-w` к скрипту `run_linux_mac` или `run_windows`.

Так же, возможно использовать более современную программу [Surfer](#) для просмотра временных диаграмм.

Surfer доступен на системах Linux, Windows и macOS, а так же в качестве [расширения редактора VS Code](#).

## Упражнение 1. Кольцевой буфер

---

Ознакомьтесь с примером кольцевого буфера с шириной в 1 бит. Также, ознакомьтесь с примером кольцевого буфера с произвольной шириной и глубиной, но принимающего и передающего данные каждый такт.

Задание: Реализуйте вариант кольцевого буфера, который перемещает указатель только в том случае когда на вход поступают корректные данные (сигнал `in_valid` активен).

## Упражнение 2. Базовое FIFO с счётчиком (Пример)

---

Данное упражнение является примером имплементации базового FIFO произвольной глубины с использованием счётчика текущего количества элементов.

В данном упражнении отсутствует задание. Исходные коды приведены в ознакомительных целях.

## Упражнение 3. FIFO с глубиной кратной степени двойки

---

Задание: Завершите реализацию модуля FIFO. В недостающих местах заполните код обновления расширенного указателя чтения и сигнал `empty` для полноценного функционирования FIFO.

## Упражнение 4. Оптимизированное FIFO

---

Примечание: Оптимизация FIFO заключается в отсутствии счётчика зависящего от глубины, но наличие только 2 битов для определения взаимного расположения указателей чтения и записи.

Задание: Реализуйте код обновления указателя чтения и чётности круга у соответствующего бита, а так же сформировать сигнал `full` с учётом констант `equal_ptr` и/или `same_circle`.

## Упражнение 5. FIFO с регистрами `empty` и `full`

---

В данном упражнении сигналы `empty` и `full` являются регистрами, внутренние сигналы же формируются комбинационно и далее записываются в соответствующие регистры.

Задание: Реализуйте код формирования комбинационного сигнала `rd_ptr_d`, а так же описать логику формирования сигналов `empty_d` и `full_d` при выставлении сигнала `pop`

## Упражнение 6. Formula 2 с кольцевым буфером

---

Структура папки упражнения идентична упражнению с формулами в Домашнем задании 5. Для лучшего понимания, рекомендуется ознакомиться со статьей Юрия Панчула "Что умеют и не умеют писать на SystemVerilog для ASIC и FPGA американские студенты?" в [журнале FPGA-Systems Magazine](#).

Задание: Реализуйте вычисление Формулы 2 используя конвейеризованные модули `isqrt` и готовый модуль `circular_buffer_with_valid` из Упражнения 1.

## Упражнение 7. Formula 2 с FIFO

---

Структура папки упражнения идентична упражнению с формулами в Домашнем задании 5. Для лучшего понимания, рекомендуется ознакомиться со статьей Юрия Панчула "Что умеют и не умеют писать на SystemVerilog для ASIC и FPGA американские студенты?" в [журнале FPGA-Systems Magazine](#).

Задание: Реализуйте один из последних случаев описанных в статье — вычисление Формулы 2 используя конвейеризованные модули `isqrt` и готовый модуль из файла `flip_flop_fifo_with_counter`.

PIPELINED SQRT (A + SQRT (B + SQRT (C)))H  
WITH 3 ISQRT PIPELINED MODULES  
AND TWO FIFOs

