**Прожирко Владислав, группа P41193**

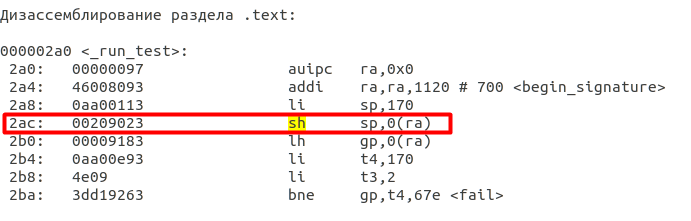
**Lab SCR1 pipeline**

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Команда | Arch #1 | Arch #2 |
| 5 | SH | RVIC -IFU\_BYPASS | RVIC +IFU\_BYPASS |

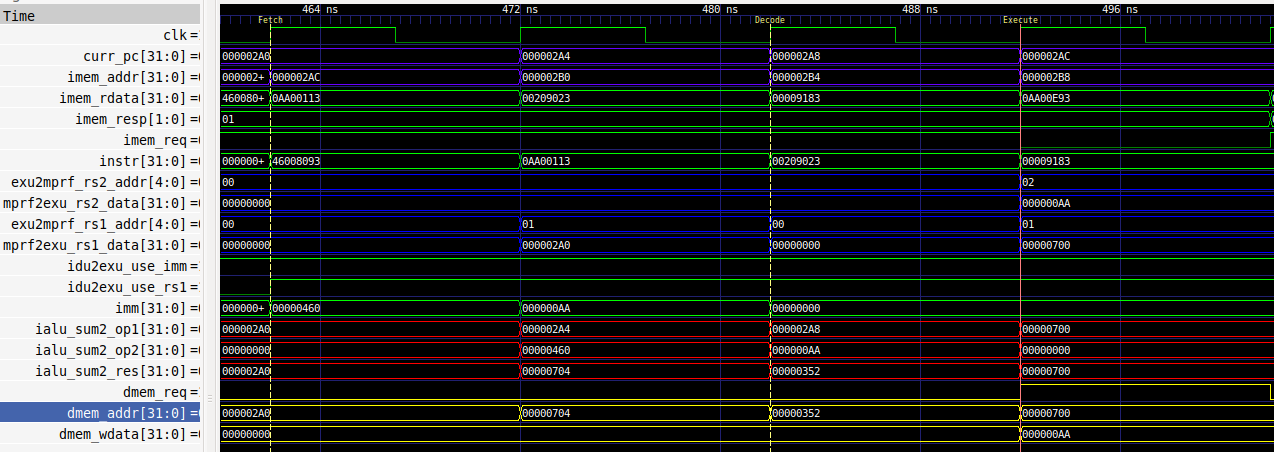
**1 Часть**

Для разборки был выбран файл с тестом «sh.S».

Отрывок из dump-файла, где используется команда sh:



Результирующая wave-form:

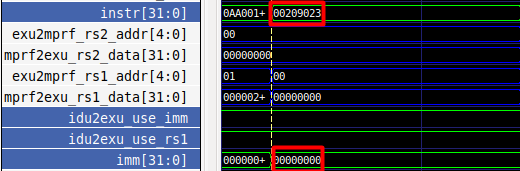


Описание сигналов:

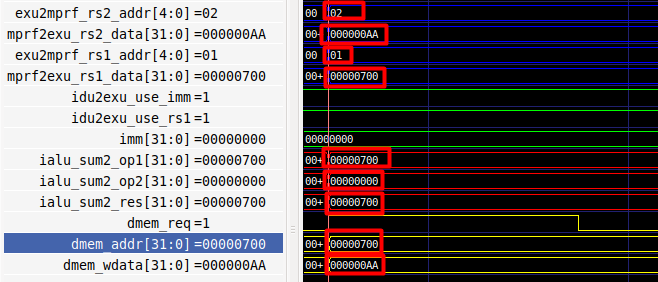
* clk - тактовый импульс
* curr\_pc - текущее значение счетчика команд, соответствует стадии Execution (000002AC)
* набор сигналов для Instruction Fetch:
  + imem\_req - запрос от процессора в память инструкций (1)
  + imem\_addr - адрес запроса памяти инструкций (000002AC)
  + imem\_resp - ответ памяти инструкций (01)
  + imem\_rdata - данные чтения памяти инструкций (00209023)



* набор сигналов для Decode:
  + instr – закодированная инструкция (00209023)
  + idu2exu\_use\_rs1 – сигнал о том, что в команде есть rs1 (1)
  + idu2exu\_use\_imm – сигнал о том, что в команде есть immediate (1)
  + imm – значение immediate (00000000)



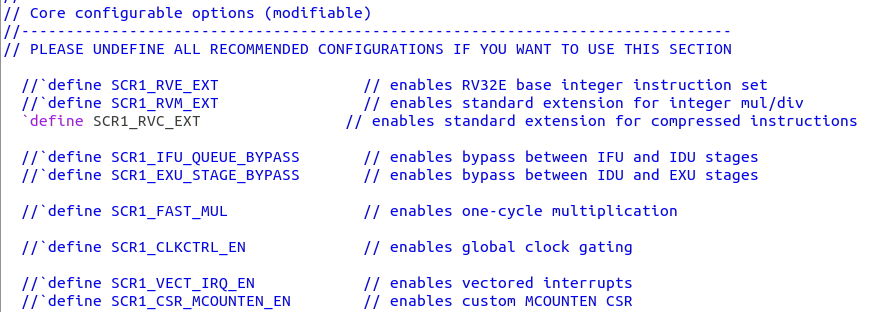
* Набор сигналов для Execute:
  + exu2mprf\_rs2\_addr – номер регистра в котором лежит адрес для записи в память. (02), x2 = sp
  + exu2mprf\_rs2\_data – значение для записи в регистровый файл (000000AA).
  + exu2mprf\_rs1\_addr – номер регистра в котором лежит адрес. (01) x1;
  + exu2mprf\_rs1\_data – значение регистра x1 (00000700);
  + imm – значение смещения (00000000);
  + ialu\_sum2\_op1 – первый операнд АЛУ (00000700) Это значение регистра x1;
  + ialu\_sum2\_op2 – второй операнд АЛУ (00000000) Это смещение;
  + ialu\_sum2\_res – результат АЛУ (00000700);
  + dmem\_req – запрос от процессора в память данных;
  + dmem\_addr – адрес, по которому будет записано значение в память (00000700);
  + dmem\_addr – значение которое нужно положить в память (000000AA).



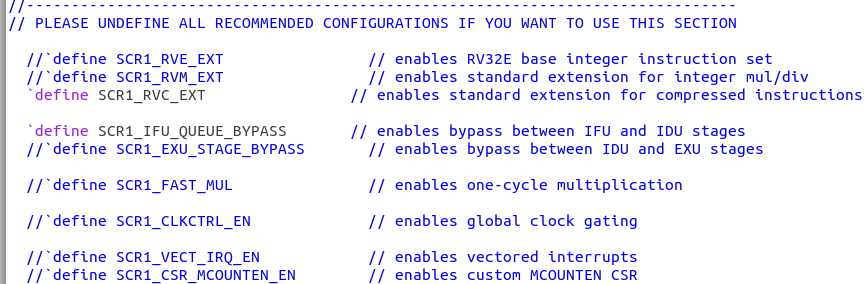
**2 Часть**

Были запущены два теста-бенчмарка Coremark и Dhrystone для двух архитектур по заданию: RVIC -IFU\_BYPASS и RVIC +IFU\_BYPASS. Для этого был изменён файл «scr1\_arch\_description.svh»:

RVIC -IFU\_BYPASS



RVIC +IFU\_BYPASS



Вызов: make run\_verilator\_wf BUS=AHB ARCH=IC IPIC=0

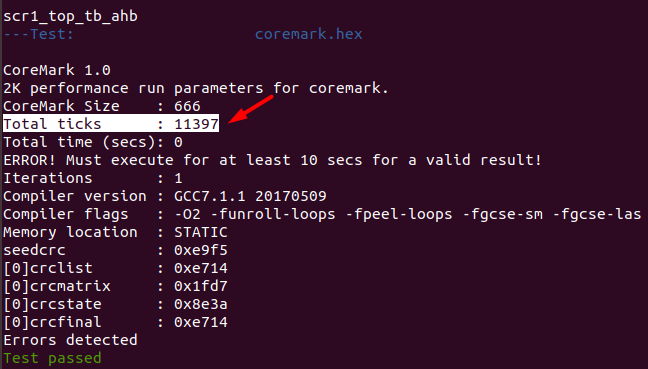
Были получены следующее результаты:

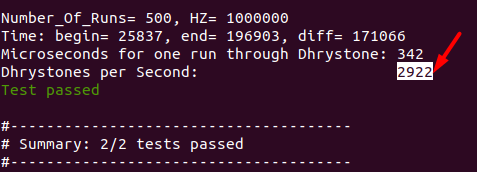
|  |  |  |
| --- | --- | --- |
| Тест | RVIC -IFU\_BYPASS | RVIC +IFU\_BYPASS |
| Coremark (Total ticks) | 11397 Total ticks | 9490 Total ticks |
| Coremark (size memory) | 78,6 Кбайт | 78,6 Кбайт |
| Dhrystone (per Seconds) | 2922 per Seconds | 3133 per Seconds |
| Dhrystone (size memory) | 45,3 Кбайт | 45,3 Кбайт |

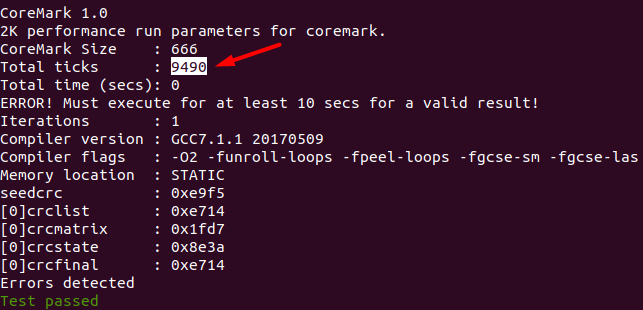
При выборе разных архитектур, для Dhrystone, количества итераций в секунду (изменилось на 211), а размер занимаемой памяти не изменился. Для Coremark количество итераций в секунду изменяется на 1907, при это размер занимаемой памяти не увеличился.

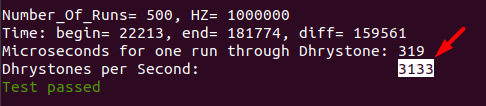
Из полученных результатов можно сделать вывод, что Coremark занимает почти в два раза больше памяти, чем тест Dhrystone, но при этом количество итераций в секунду выше, по сравнению с Dhrystone.

Скриншоты тест-бенчмарков Coremark и Dhrystone для ARCH = RVIMC



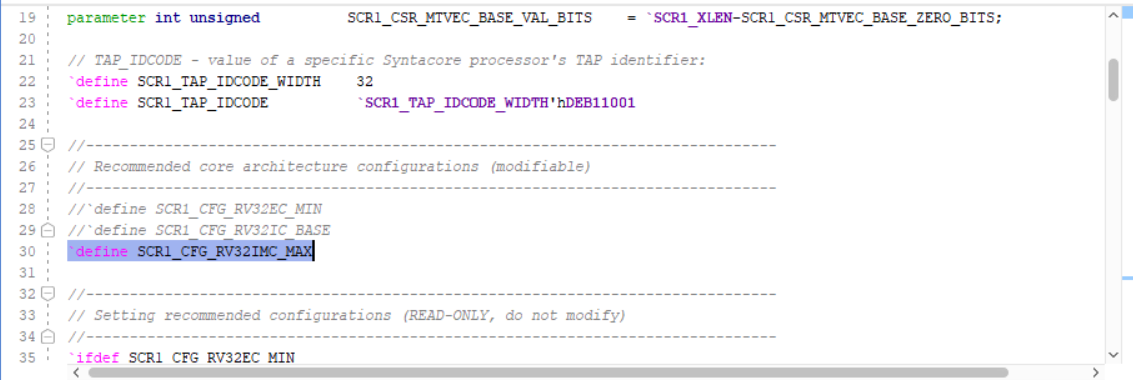


Скриншоты тест-бенчмарков Coremark и Dhrystone для ARCH = RVIM 



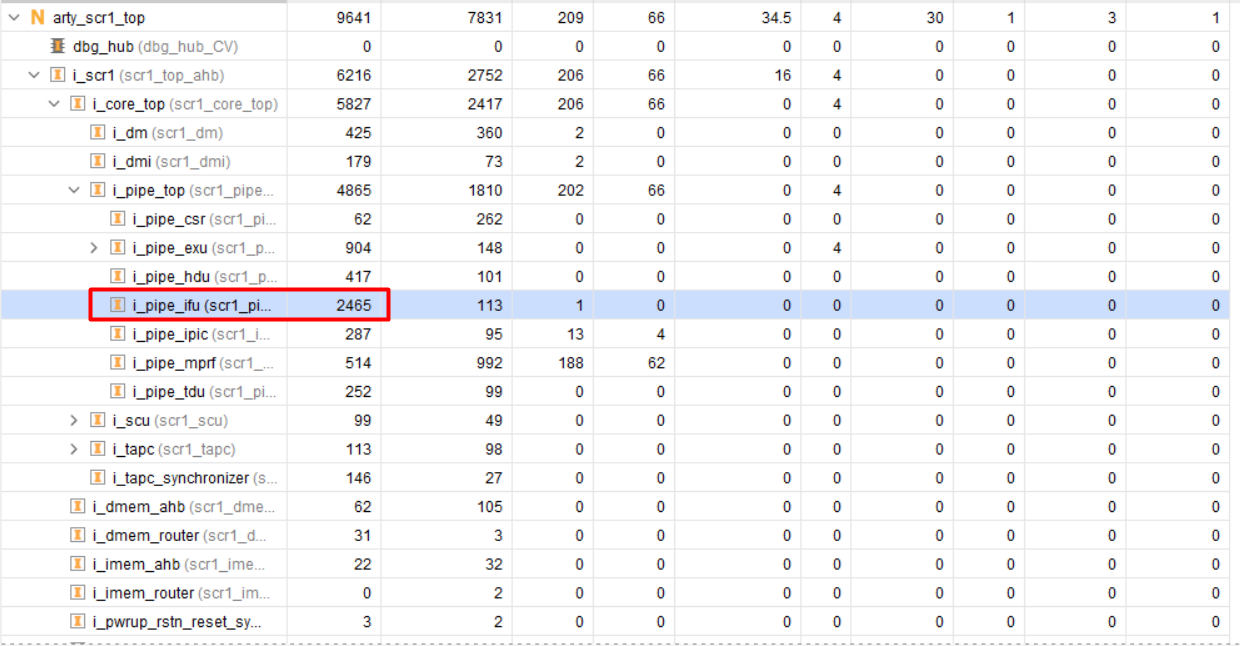
Запуск c конфигурацией IMC\_MAX.

Конфигурация была выбрана в файле scr1\_arch\_description.svh

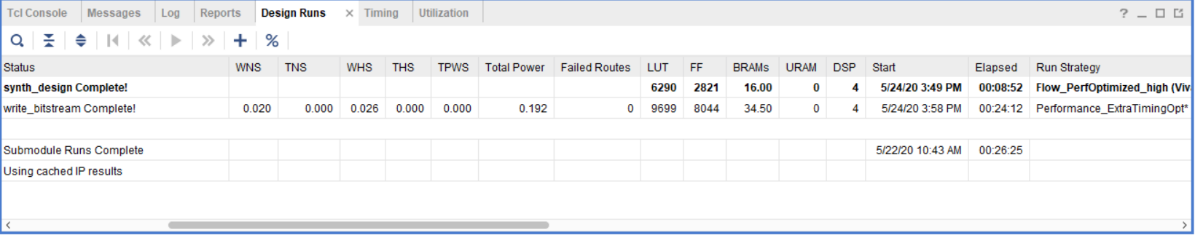


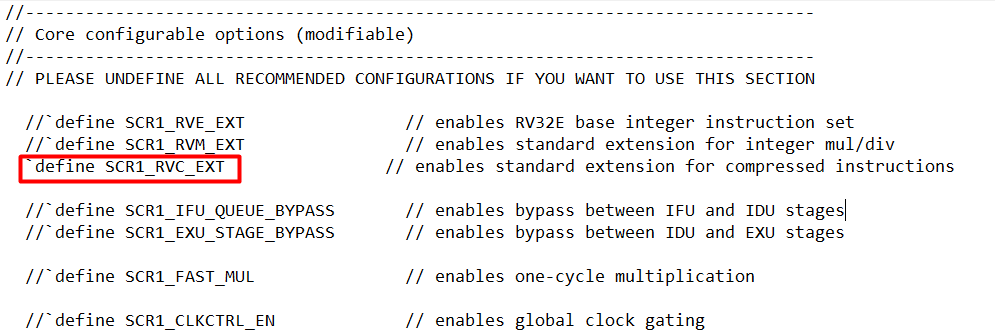
Fmax = 33.333 MHz/30 ns

1. Report Utilization. Модуль ядра, занимающий наибольшую площадь на кристалле: i\_pipe\_ifu = 2465.



1. Значение WNS = 0.020 c Perfomance\_ExtraTimingOpt.

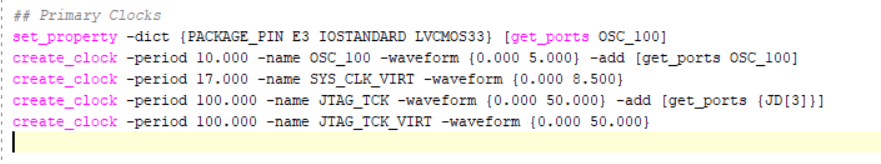


В файле scr1\_arch\_description.svh по варианту оставил **RVIC**.

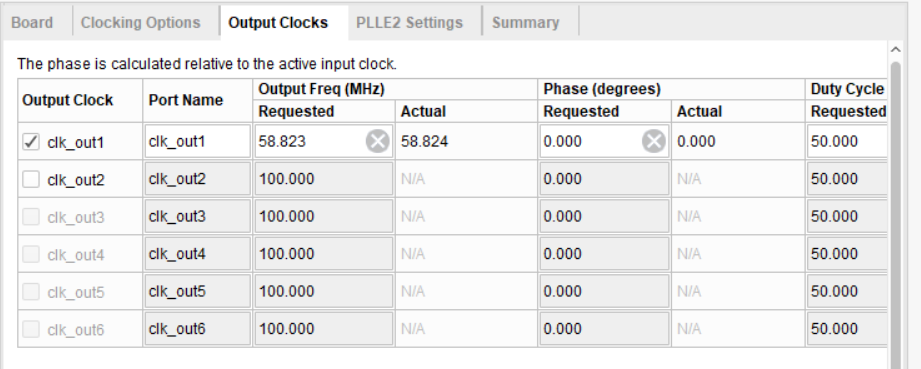
Определение Fmax:

Изначально поставил частоту F = 33.333 MHz/ 30 ns.

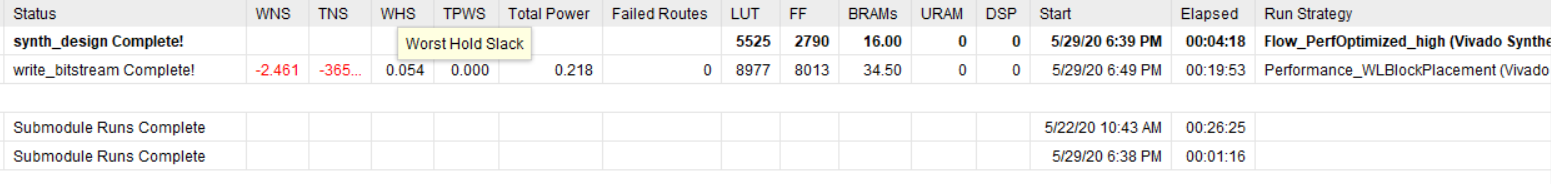
WNS получилось 12.928. Исходя из этого значения, я понял, что у меня есть запас 13 ns и поэтому уменьшил период на 13 ns (период SYS\_CLK\_VIRT = 17 ns).



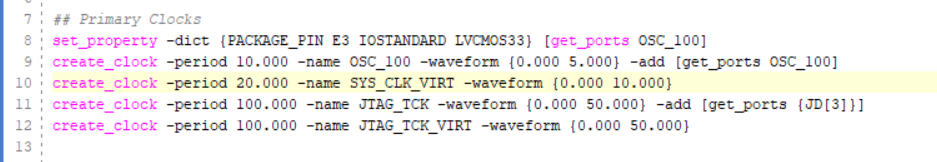
Следовательно, F равно (1/17)\*1000 = 58,823 MHz.



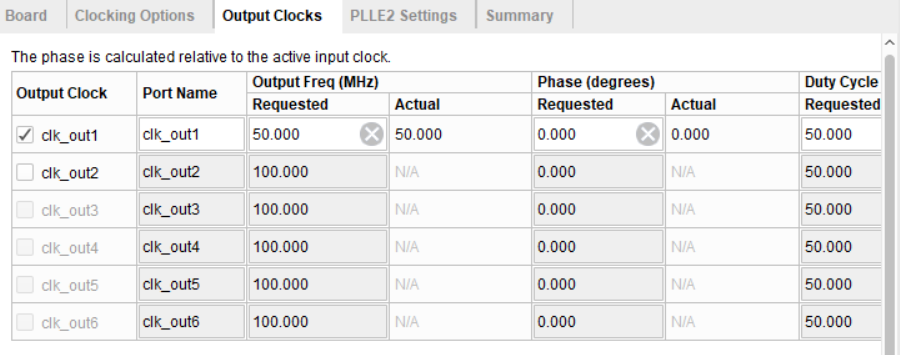
Получил WNS = -2.451 ns



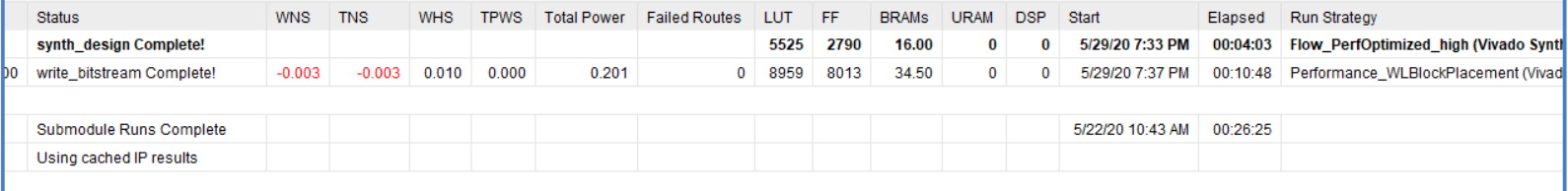
Из-за того, что WNS = -2.45 ns можно сделать вывод, что запаса по времени удержания и установки недостаточно приблизительно на 3 ns. Необходимо увеличить период SYS\_CLK\_VIRT на 3 ns. Следовательно, период будет равен 20 ns.



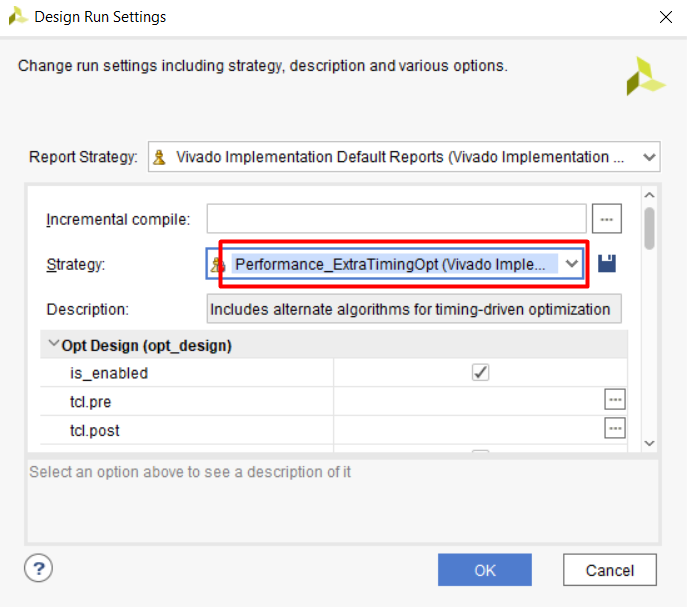
А частота (1/20)\*1000 = 50MHz.



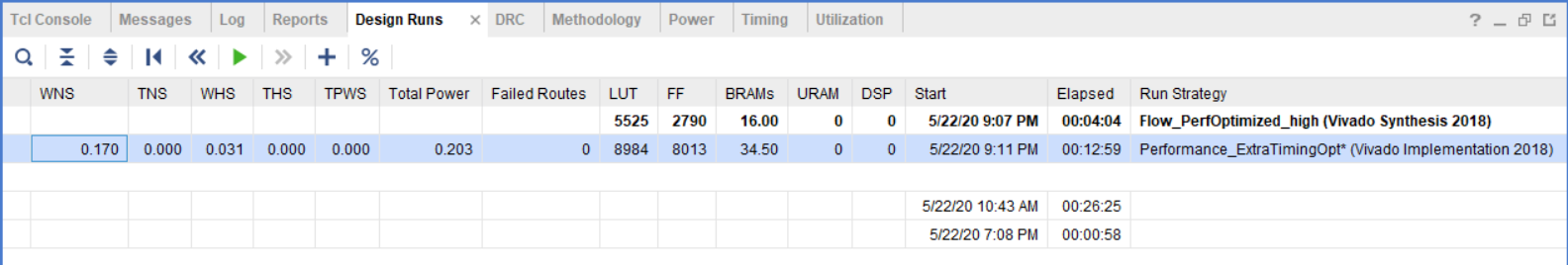
WNS получилось -0,003.



Чтобы убрать это отрицательное значение использовал стратегию Perfomance\_ExtraTimingOpt.

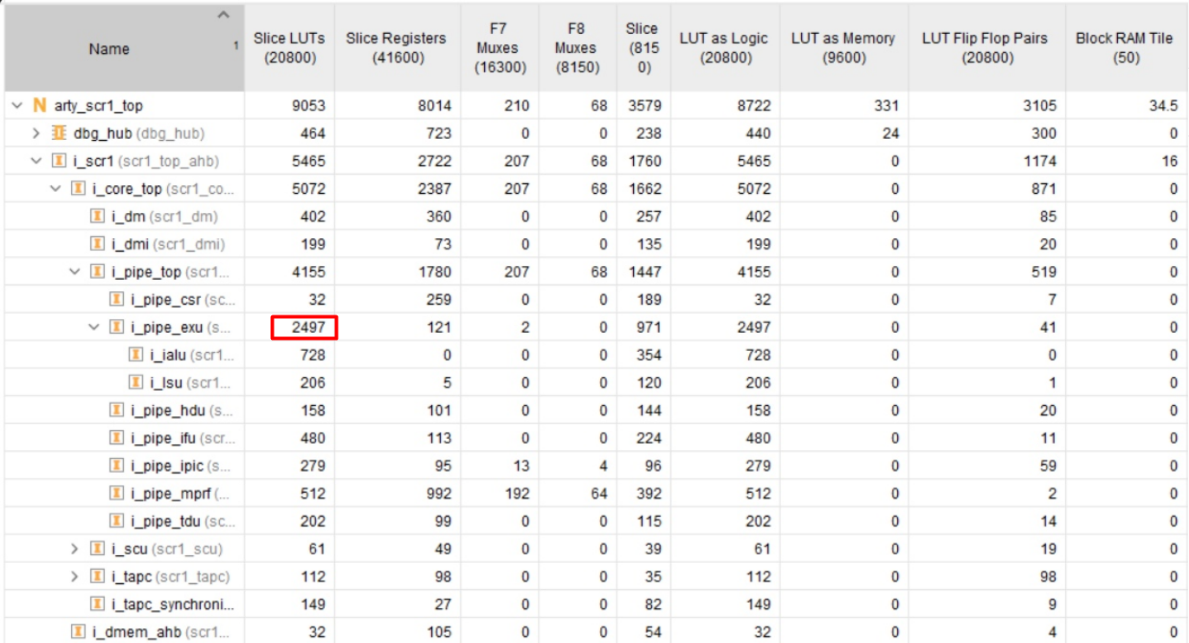


С помощью Perfomance\_ExtraTimingOpt удалось добиться WNS = 0.170.

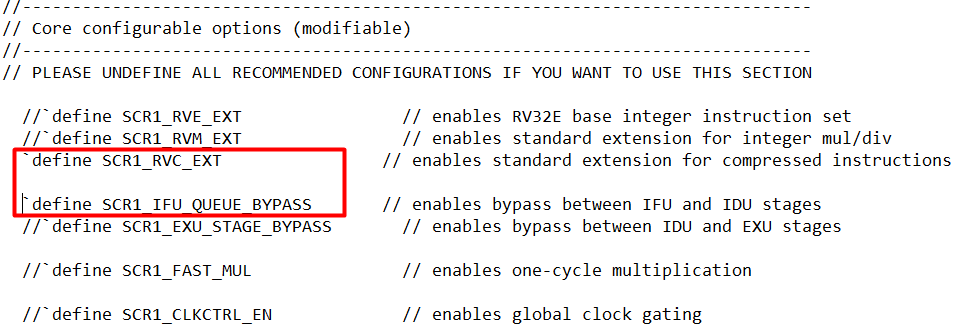


Следовательно, для RVIC Fmax = 50 MHz.

После успешного прохождения Generate Bitstream, был запущен Report Utilization и определил модуль ядра, занимающий наибольшую площадь на кристалле i\_pipe\_exu = 2497.

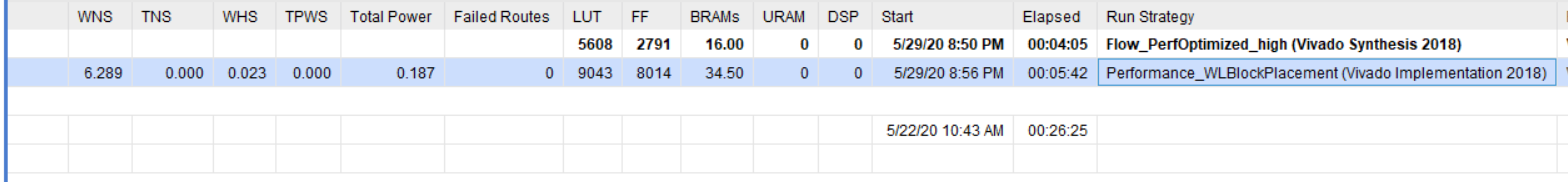


В файле scr1\_arch\_description.svh по варианту поставил **RVIC +IFU\_BYPASS**.

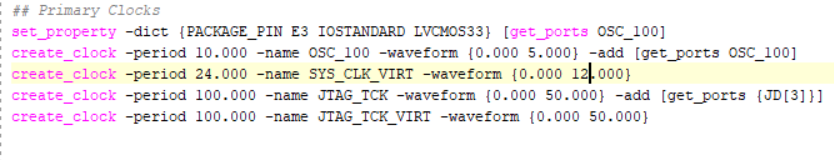


Изначально поставил частоту F = 33.333 MHz/ 30 ns.

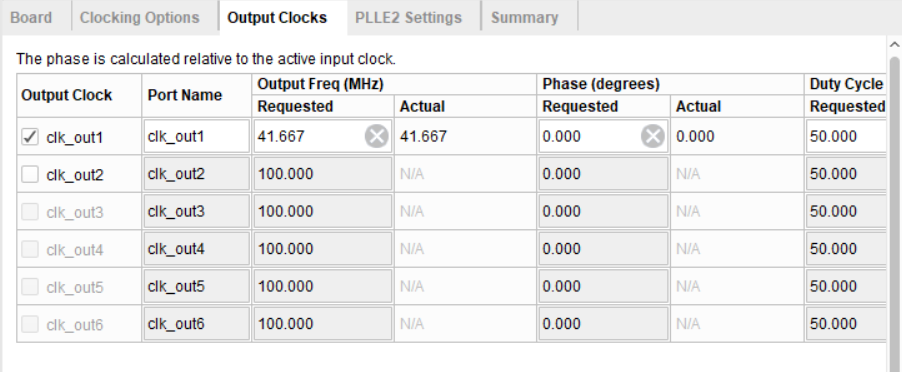
WNS получилось 6.289.



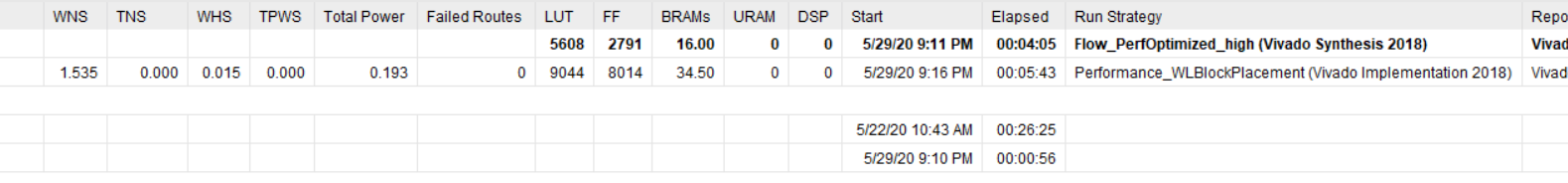
Исходя из этого значения, я понял, что у меня есть запас 6 ns и поэтому уменьшил период на 6 ns (период SYS\_CLK\_VIRT = 24 ns).



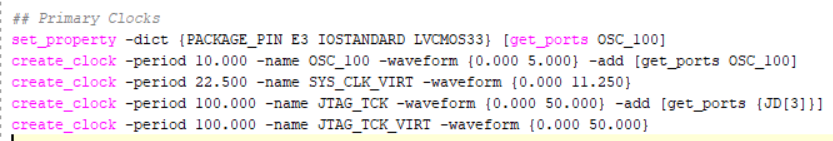
Следовательно, F равно (1/24)\*1000 = 41,667 MHz.



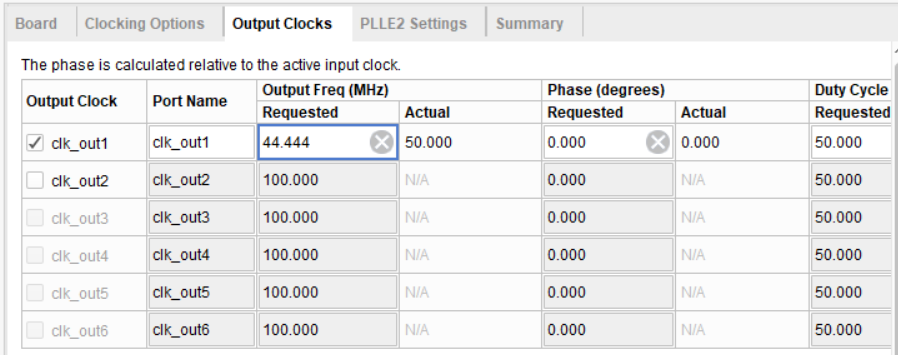
Получил WNS = 1.535 ns



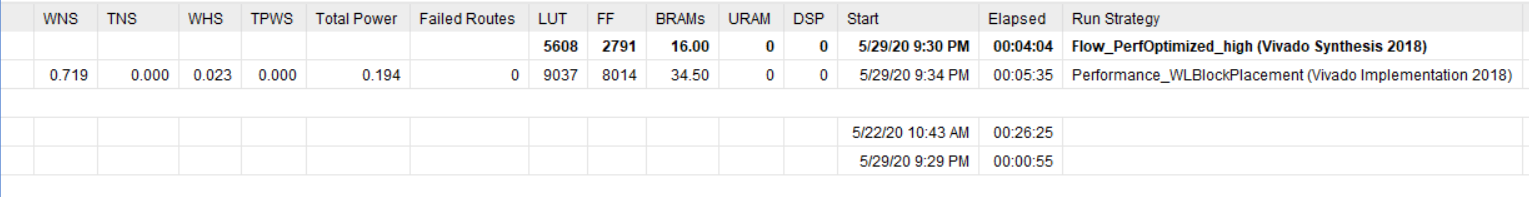
WNS = 1.535 ns можно сделать вывод, что запаса по времени удержания и установки с избытком. Можно попробовать уменьшить период SYS\_CLK\_VIRT на 1.5 ns. Следовательно, период будет равен 22.5 ns.



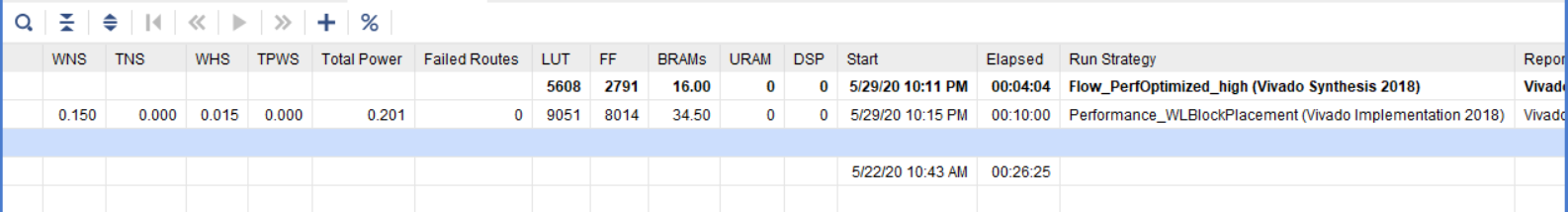
Следовательно, F равно (1/22.5)\*1000 = 44,444 MHz.



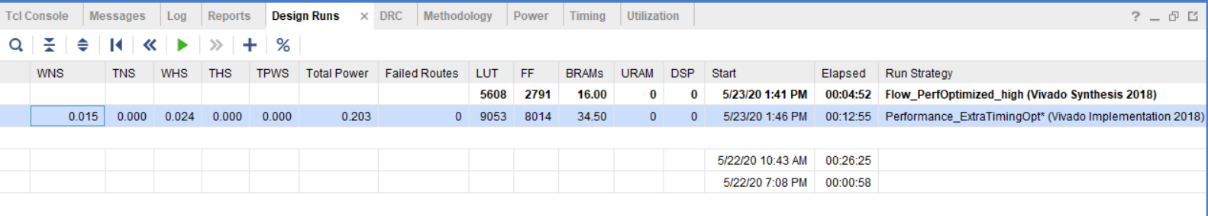
Получилось WNS = 0.719



Я решил попробовать оптимизировать и уменьшить значение WNS. Поставив SYS\_CLK\_VIRT 20 ns, а F равно (1/20) \*1000 = 50,000 MHz.

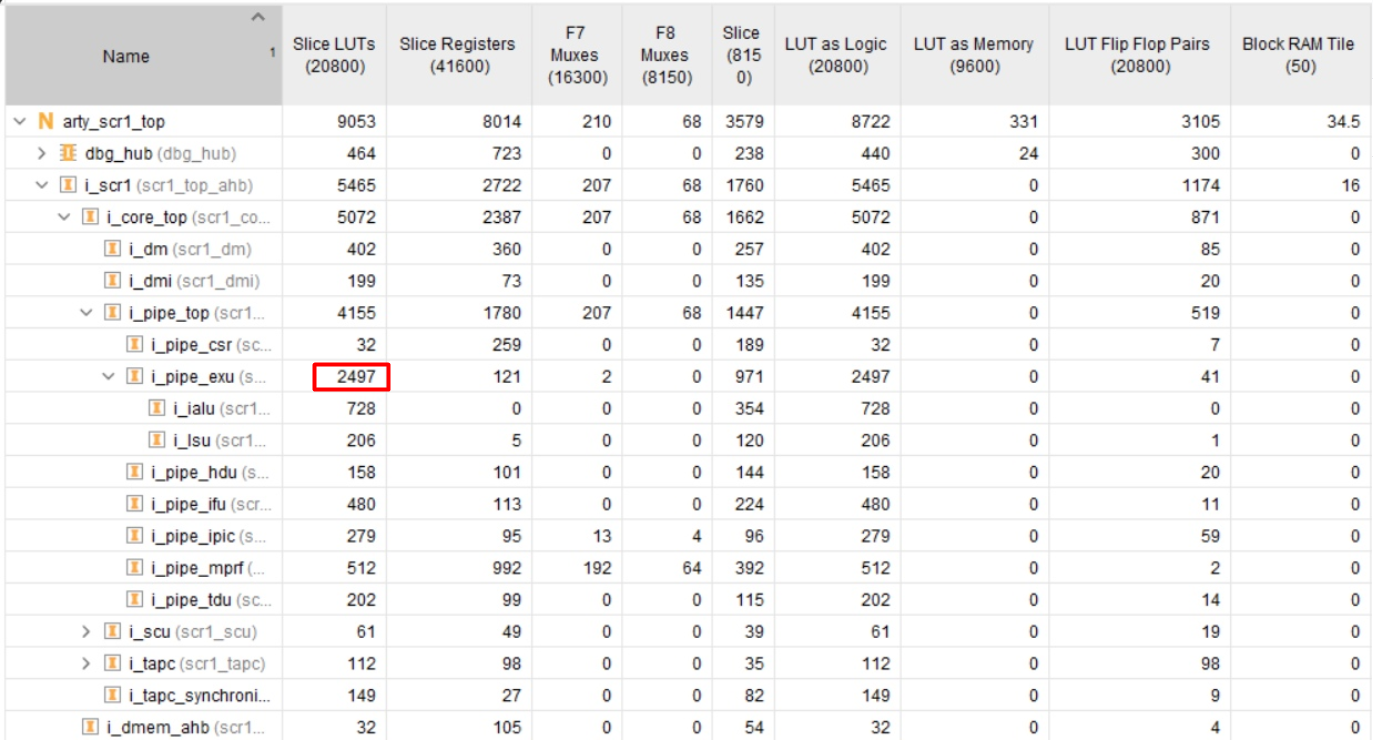


Для того чтобы WNS приблизился к 0 решил использовать стратегию Perfomance\_ExtraTimingOpt. И получил WNS = 0.015



Следовательно, для RVIC **+**IFU**\_**BYPASS Fmax = 50 MHz.

Определил модуль ядра, занимающий наибольшую площадь на кристалле i\_pipe\_exu = 2497.



Сравнение занимаемой площади на кристалле в зависимости от архитектур по варианту:

|  |  |  |  |
| --- | --- | --- | --- |
| Arch | LUT (Look Up Table) | FF (Flip-Flop) | Fmax |
| RVIC -IFU\_BYPASS | 8984 | 8013 | 50 MHz |
| RVIC +IFU\_BYPASS | 9053 | 8014 | 50 MHz |
| IMC\_MAX | 9699 | 8044 | 33.333 MHz |

Как заметно из таблицы значения LUT и FF увеличилось при добавлении IFU\_BYPASS. SCR1\_IFU\_QUEUE\_BYPASS обеспечивает обход очереди, т.е. команды передаются непосредственно в модуль декодирования команд.

Модули ядра, занимающие наибольшую площадь на кристалле для архитектуры RVIC -IFU\_BYPAS, RVIC +IFU\_BYPASS имеют одинаковые значения потому, что SCR1\_IFU\_QUEUE\_BYPASS включает, либо отключает очередь в модуле i\_pipe\_ifu, но занимаемая площадь на кристалле не меняется, так как этот модуль всегда должен присутствовать на кристалле.

Fmax для архитектур RVIC -IFU\_BYPASS, RVIC +IFU\_BYPASS одинакова и равна 50 Mhz, следовательно, запаса WNS хватает для обоих случаев. Если сравнить с IMC\_MAX, то при 33,333Mhz, будет большой запас WNS, следовательно, путь между модулями получится меньше, соответственно можно сделать частоту больше.