**Самойлов Владислав, группа P41193**

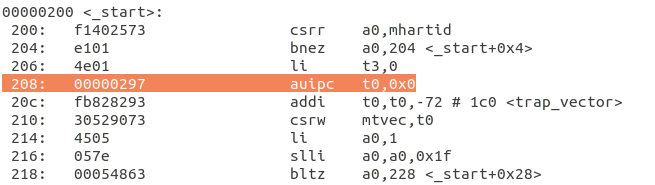
**Lab SCR1 pipeline**

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Команда | Arch #1 | Arch #2 |
| 6 | AUIPC | RVE | RVI |

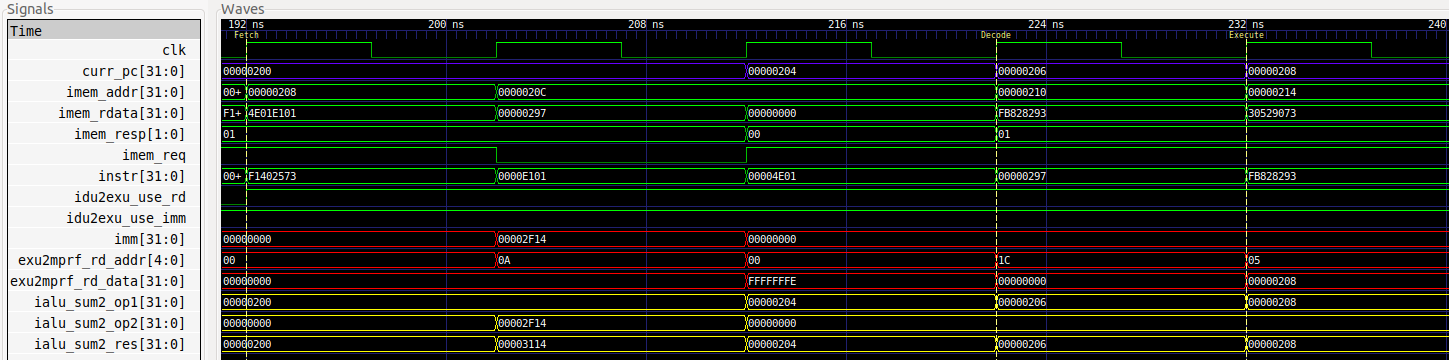
**1 Часть**

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

Отрывок из dump-файла, где используется команда auipc:

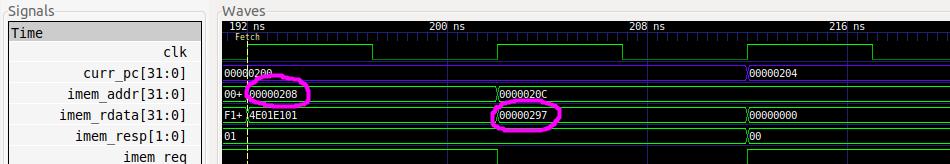


Результирующая wave-form:

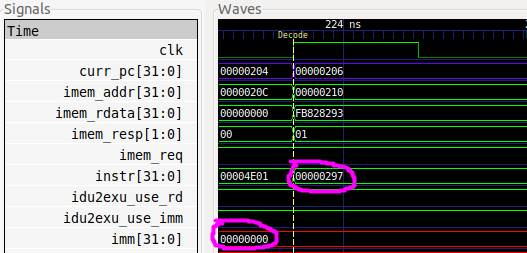


Описание сигналов:

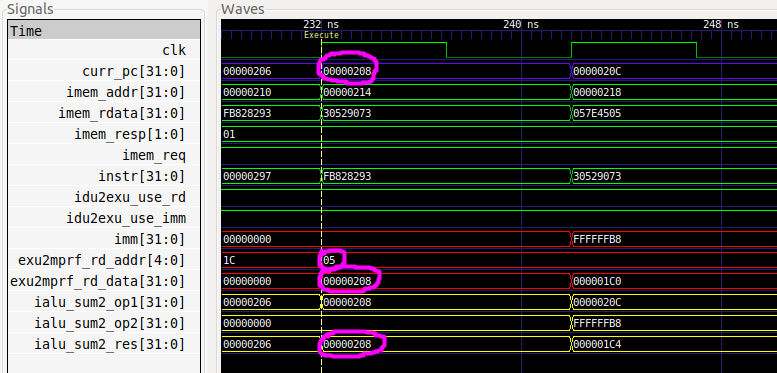
* clk - тактовый импульс
* curr\_pc - текущее значение счетчика команд, соответствует стадии Execution (00000208)
* набор сигналов для Instruction Fetch:
  + imem\_req - запрос от процессора в память инструкций (1)
  + imem\_addr - адрес запроса памяти инструкций (00000208)
  + imem\_resp - ответ памяти инструкций (01)
  + imem\_rdata - данные чтения памяти инструкций (00000297)



* набор сигналов для Decode:
  + instr – закодированная инструкция (00000297)
  + idu2exu\_use\_rd – сигнал о том, что в команде есть rd (1)
  + idu2exu\_use\_imm – сигнал о том, что в команде есть immediate (1)
  + imm – значение immediate (00000000)



* Набор сигналов для Execute:
  + exu2mprf\_rd\_addr – номер регистра в который будет записываться PC + Imm. (05), x5 = t0
  + exu2mprf\_rd\_data – значение регистра rd = х5(00000208).
  + imm – значение смещения (00000000);
  + ialu\_sum2\_op1 – первый операнд АЛУ (00000208) Это текущее значение PC;
  + ialu\_sum2\_op2 – второй операнд АЛУ (00000000) Это смещение;
  + ialu\_sum2\_res – результат АЛУ (00000208)



**2 Часть**

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

`define SCR1\_RVE\_EXT \\ Для добавление M архитектуры

И при запуске указывалась I / E архитектура

make run\_verilator\_wf BUS=AHB ARCH=I/ E IPIC=0

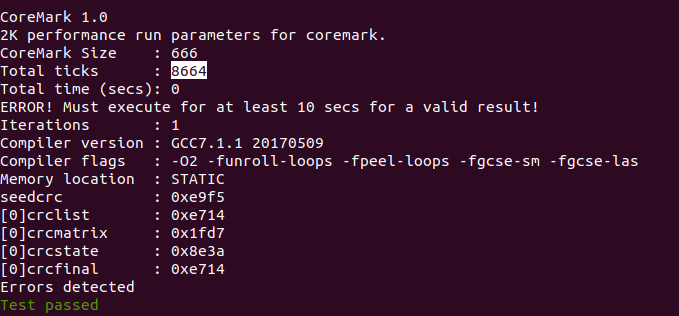
Были получены следующее результаты:

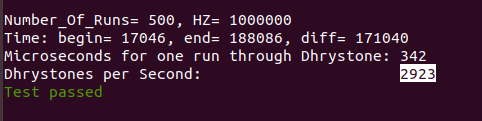
|  |  |  |
| --- | --- | --- |
| Тест | RVE | RVI |
| Coremark (Total ticks) | 8664 Total ticks | 7879 Total ticks |
| Coremark (size memory) | 100,2 Кбайт | 93,6 Кбайт |
| Dhrystone (per Seconds) | 2923 per Seconds | 3133 per Seconds |
| Dhrystone (size memory) | 53,8 Кбайт | 49,5 Кбайт |

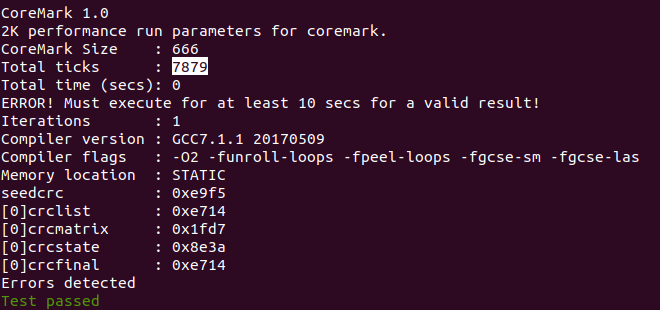
При выборе разных архитектур, для Dhrystone, количества итераций в секунду (изменилось на 210), размер занимаемой памяти уменьшился на 4,3 Кбайта. Для Coremark количество итераций в секунду изменяется на 785, при это размер занимаемой памяти уменьшился на 6,6 Кбайта.

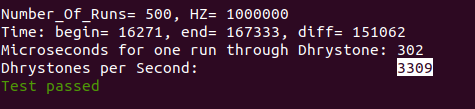
Из полученных результатов можно сделать вывод, что Coremark занимает почти в два раза больше памяти, чем тест Dhrystone, но при этом количество итераций в секунду выше, по сравнению с Dhrystone.

Скриншоты тест-бенчмарков Coremark и Dhrystone для ARCH = RVE





Скриншоты тест-бенчмарков Coremark и Dhrystone для ARCH = RVI 

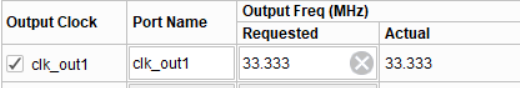


**3 часть**

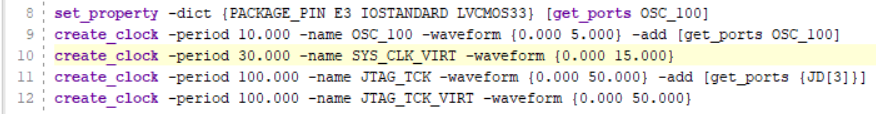
|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Arch #1 | Arch #2 | Конфигурация |
| 6 | RVI | RVE | IMC\_MAX |

**IMC\_MAX**

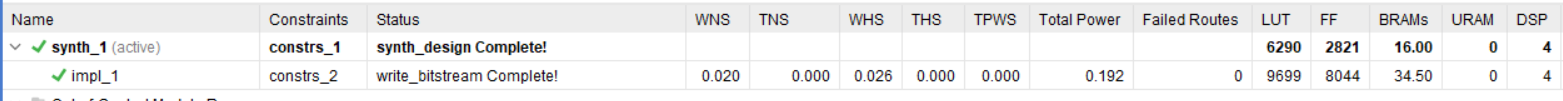
Для проверки максимальной частоты установил частоту 33.333Mhz в sys\_pll.bd в Properties:



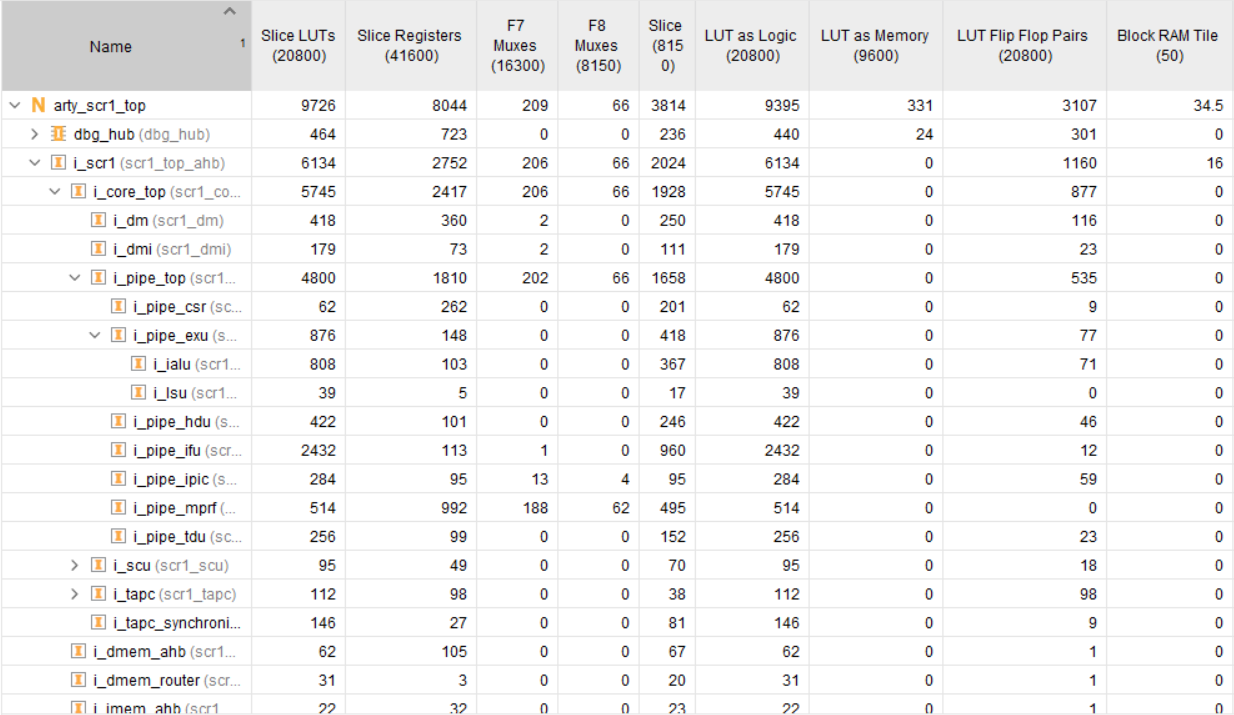
А также период 30ns для SYS\_CLK\_VIRT:



В результате тестирования получили WNS = 0.020ns, следовательно из этого принимаем максимальную частоту Fmax = 1/30ns = 33.333Mhz

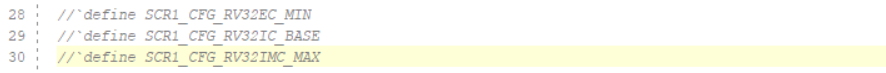


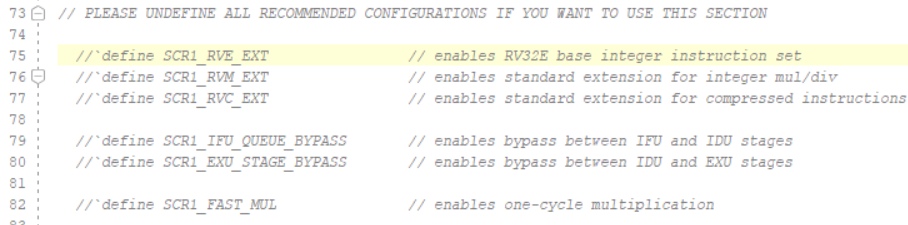
Модуль, который занимает больше всего места на кристалле – i\_pipe\_ifu = 2432 LUT



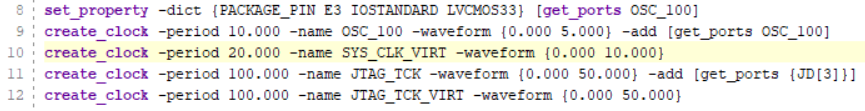
**RVI**

Для выбора RVI архитектуры изменил файл «scr1\_arch\_description.svh» (отключил RV32IMC\_MAX, по умолчанию, при отключенных других аритектурах используется RVI):

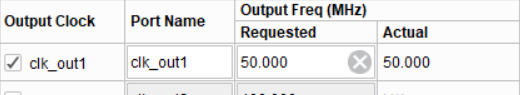




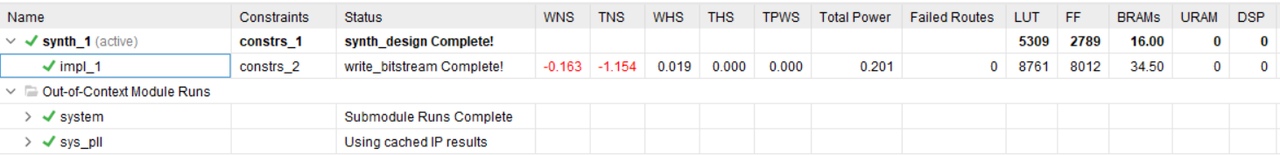
Проведем тестирование с частотой F = 50Mhz. Соответственно, период SYS\_CLK\_VIRT = 20ns:

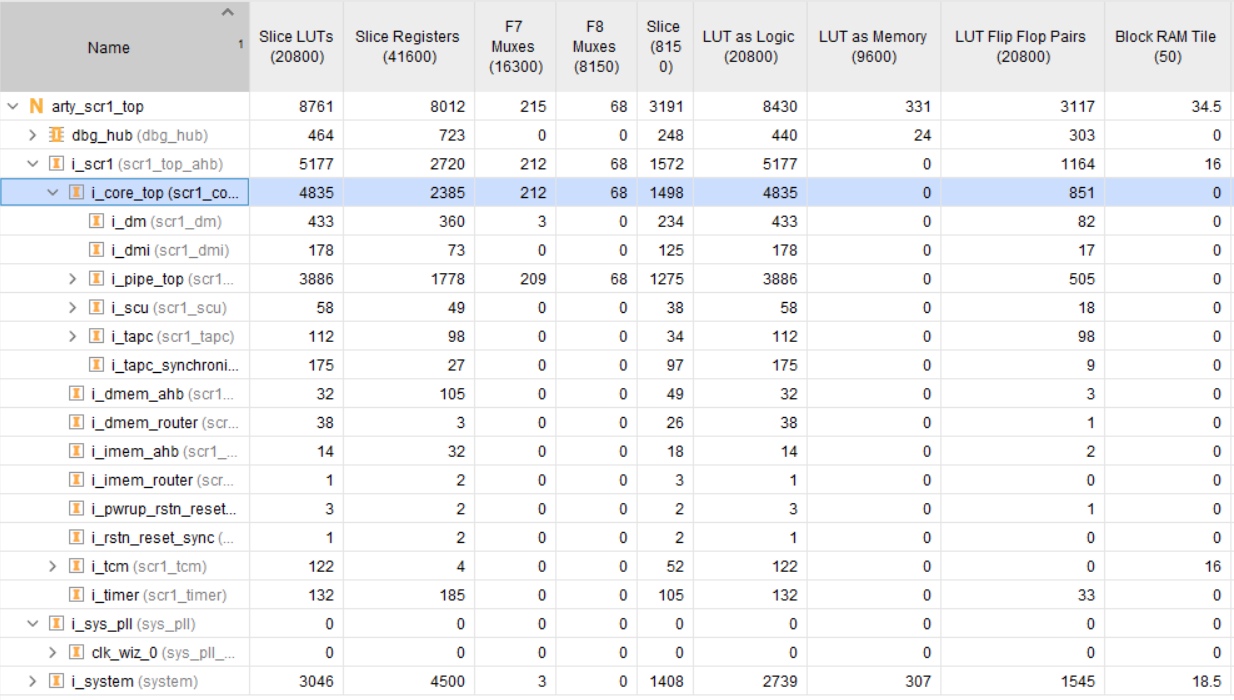


А также установил частоту 50Mhz в sys\_pll.bd в Properties.



Получил WNS = -0.163ns.

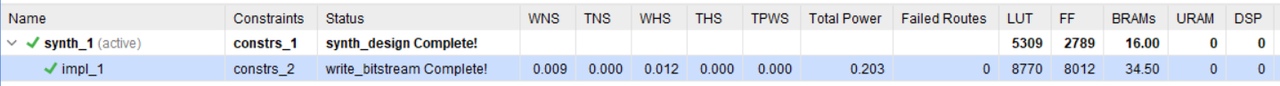


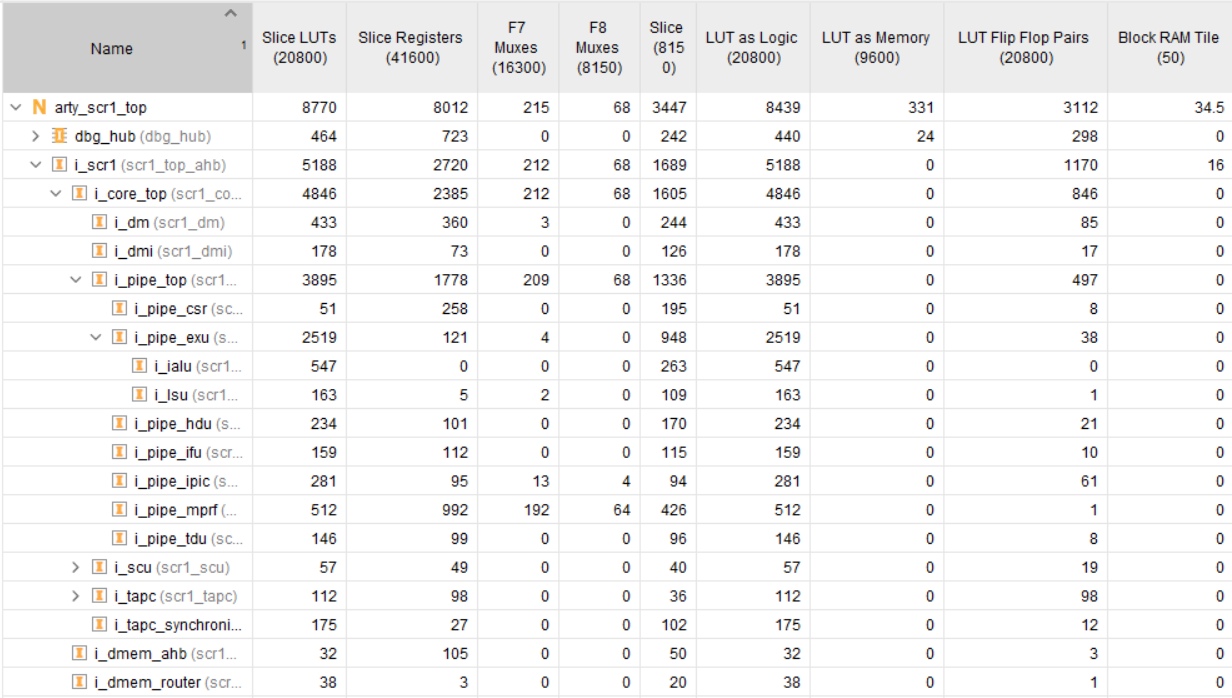


Для того чтобы WNS стал ближе к 0, подключил strategy Perfomence\_ExtraTimingOpt + phys\_opt\_design.



Провел тестирование и получил следующие результаты:



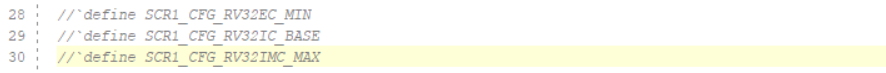


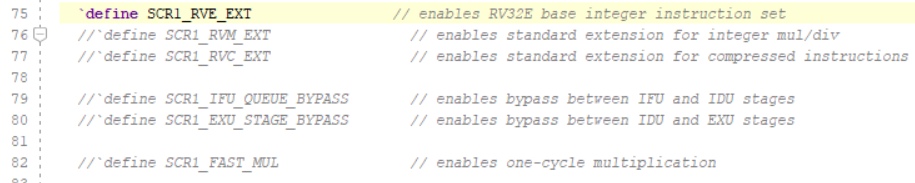
Получил WNS = 0.009. Модуль, который занимает больше всего места на кристалле – i\_pipe\_exu = 2519 LUT. При этом i\_alu = 547 LUT.

Следовательно, максимальная частота для RVIM= 50Mhz.

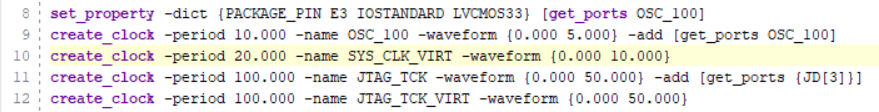
**RVE**

Для выбора RVE архитектуры изменил файл «scr1\_arch\_description.svh» (отключил RV32IMC\_MAX и выбрал RVE):

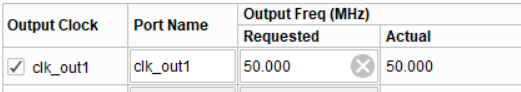




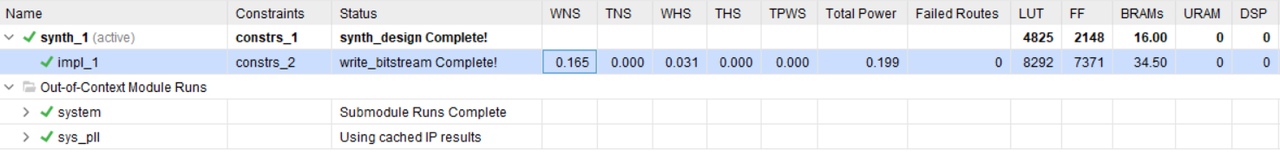
Также проведем тестирование при частоте F = 50Mhz и установим, соответственно, период SYS\_CLK\_VIRT = 20ns:



Установил частоту 50Mhz в sys\_pll.bd в Properties.



Получил WNS = 0,165 ns



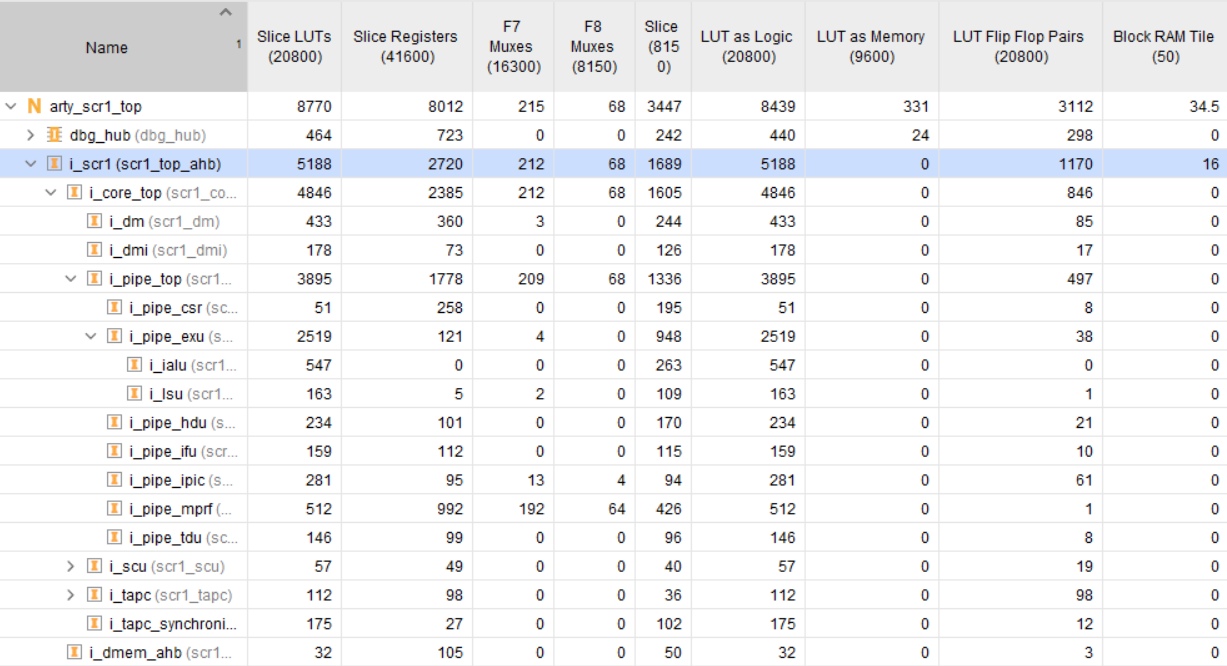
Для того чтобы WNS стал ближе к 0, включил strategy Perfomence\_ExtraTimingOpt + phys\_opt\_design.



Результаты STA получились следующие:



WNS = 0.006. Модуль, который занимает больше всего места на кристалле – i\_pipe\_exu = 2519 LUT. При этом i\_alu = 547 LUT.



Следовательно, максимальная частота для RVIMC= 50Mhz.

**Выводы**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № |  | IMC\_MAX | RVI | RVE |
| 1 | LUT | 9699 | 8770 | 8298 |
| Flip-Flop | 8044 | 8012 | 7371 |
| i\_pipe\_hdu | 516 LUT | 234 LUT | 234 LUT |
| 2 | Fmax, Mhz | 50 | 50 | 50 |
| 3 | WNS | 0.20 | 0.09 | 0.06 |

Наибольшое различие занимаемой площади между IMC\_MAX, RVI и RVE в модуле **«i\_pipe\_hdu»** (Hart debug unit), потому что, при использовании E и I архитектур убираются функции отладки для архитекуры IMC\_MAЧ и соответствено, уменьшается интерфейс модуля отладки, следовательно необходимо меньше LUT (Look up table).

Блок отладки Hart (HDU) - это компонент внутри HART, реализующий контроль над своими функциями отладки и предоставляющий интерфейс для модуля отладки.

Максимальная частота для архитектур RVI и RVE примерно равна 50.000 Mhz, но не одинакова для каждой из них. При проведении тестирования запаса WNS при 50.000 Mhz хватает для обоих случаев. При тестировании IMC\_MAX с частотой 33.333 Mhz, запас WNS будет составлять 0,20 ns, следовательно, пути между модулями укорачивать не нужно, повышение частоты не требуется. Аналогичная ситуация возникла при тестировании RVE и RVI архитектур. Запаса WNS при использовании ExtraTiming равного 0.06 и 0.09, соответственно, вполне достаточно, пути укорачивать не нужно, повышение частоты не требуется.