**Выполнил: студент гр. P41193 Морозов О. В.**

**Вариант задания**

|  |  |  |  |
| --- | --- | --- | --- |
| 3 | MUL | RVI -EXU\_BYPASS | RVI +EXU\_BYPASS |

**Часть I**

Фрагмент dump-файла с выполнением инструкции mul на 0x2b0.

000002a0 <\_run\_test>:  
2a0: 000080b7 lui ra,0x8  
2a4: e0008093 addi ra,ra,-512 # 7e00 <\_\_global\_pointer$+0x6ef8>  
2a8: b6db7137 lui sp,0xb6db7  
2ac: db710113 addi sp,sp,-585 # b6db6db7 <\_\_global\_pointer$+0xb6db5eaf>  
2b0: 022081b3 mul gp,ra,sp  
2b4: 00001eb7 lui t4,0x1  
2b8: 200e8e93 addi t4,t4,512 # 1200 <\_\_global\_pointer$+0x2f8>  
2bc: 02000e13 li t3,32  
2c0: 37d19c63 bne gp,t4,638 <fail>

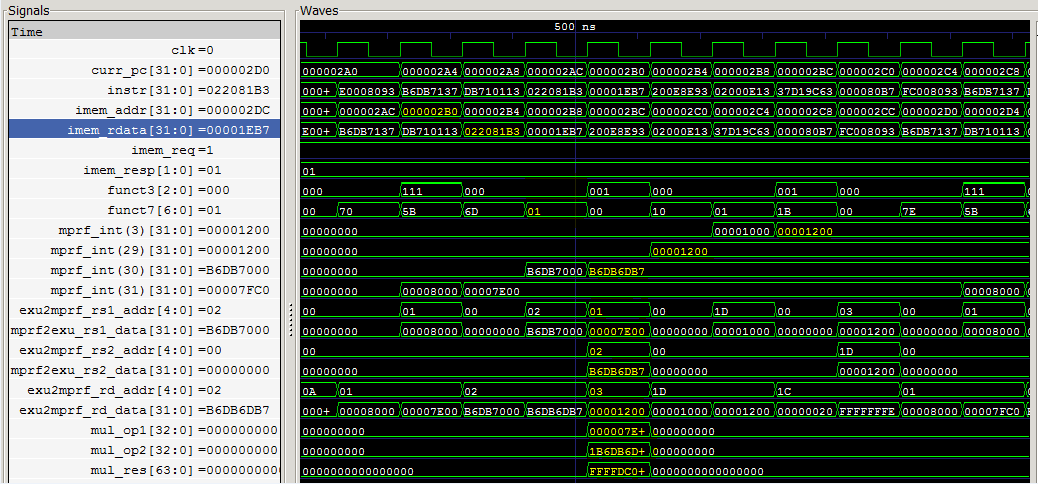
Из рисунков 1 и 2 видно, что из памяти для pc=0x2b0 (выделено синим цветом) выбирается инструкция умножения с rs2=2 (sp), rs1=1 (ra), rd=3 (gp).

  
Рисунок 1. Формат инструкции MUL.

  
Диаграмма 1. Бинарное представление инструкции MUL.

Расшифровка диаграммы 2.

|  |  |
| --- | --- |
| curr\_pc | Объяснение |
| 0x2A4 | Выборка из памяти по адресу 0х2b0. (imem\_addr) |
| 0x2A8 | По адресу 0х2b0 из памяти загружена инструкция 022081b3. (imem\_data) |
| 0x2AC | Инструкция декодируется, получаем код умножения (func7/func3) |
| 0x2B0 | Выставлены адреса регистров sp, ra, gp (exu2mprf\_rs1/rs2/rd\_addr), данные из регистров передаются на вход АЛУ (exu2mprf\_rs1/rs2/rd\_data), в умножителе устанавливаются значения операндов и вычисляется результат. (mul\_op1/op2/res) |
| 0x2B4 | В регистр gp (mprf\_int(29)) записан результат умножения. Стоит отметить, что в коде регистры пронумерованы в обратном порядке: x1 это mprf\_int(31), x2 это mprf\_int(30) и т. д. |
| 0x2BC | Тест успешен в случае, если gp = t4 (mprf\_int(3)). На это такте видно, что значения сравнялись и остались неизменными до инструкции bne. |

  
Диаграмма 2. Выполнение инструкции MUL.

**Часть II**

Результат для архитектуры RVI -EXU\_BYPASS

Dhrystones per Second: 3010



Результат для архитектуры RVI +EXU\_BYPASS

Dhrystones per Second: 3153

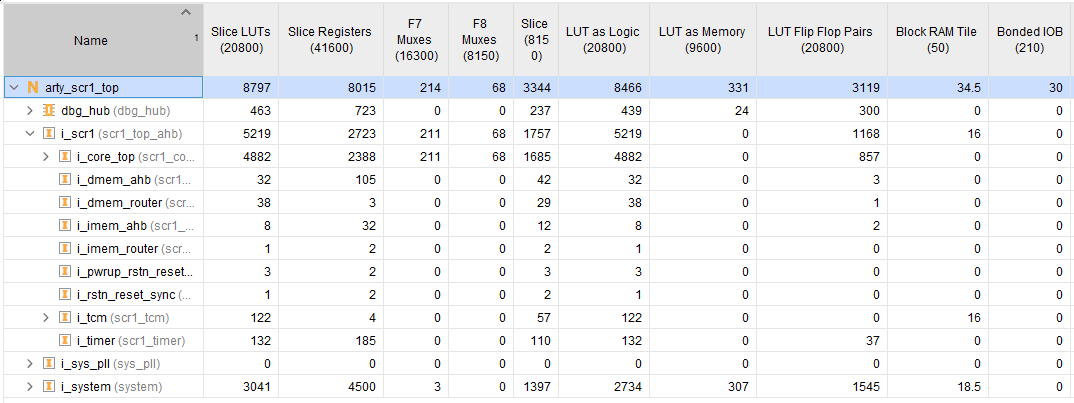


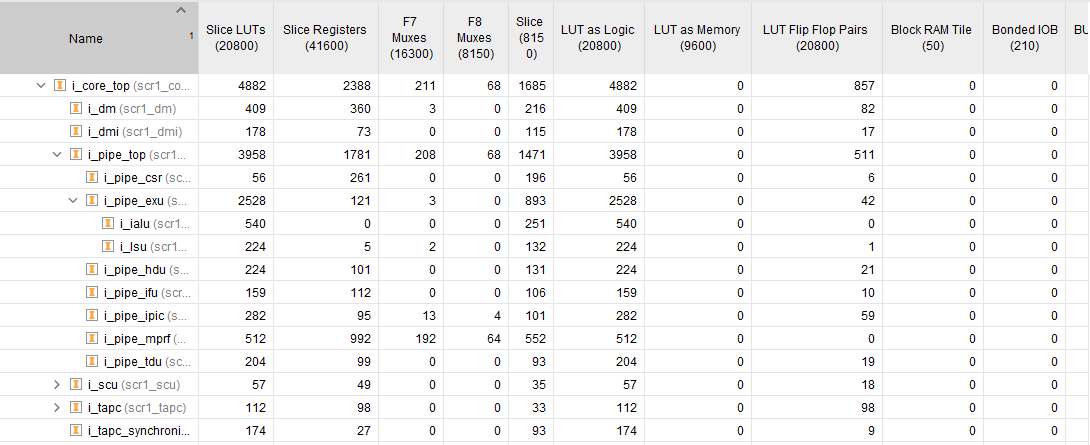
**Часть III**

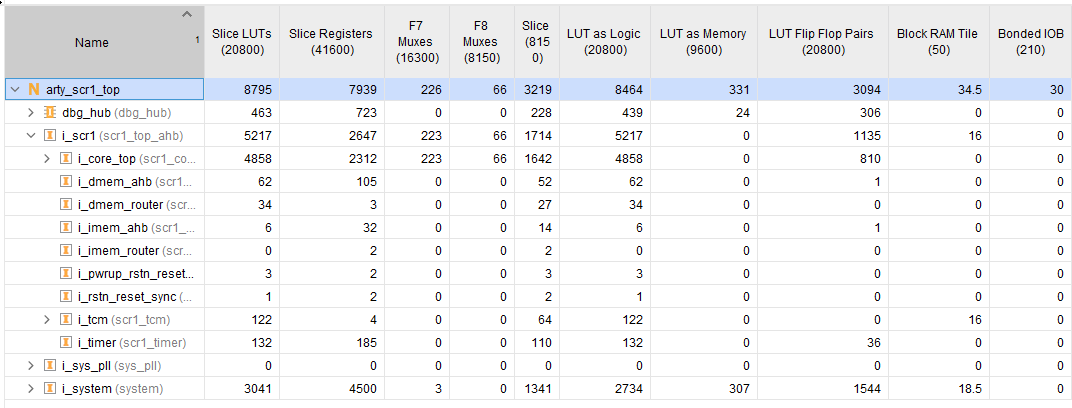
При изменении наличие параметра SCR1\_EXU\_STAGE\_BYPASS добавляет bypass перед стадией exu, согласно scr1\_eas.pdf конвеер состоит из 3 стадий. (SCR1\_IFU\_QUEUE\_BYPASS отключен) Без EXU\_BYPASS конвеер состоит из 2 стадий.

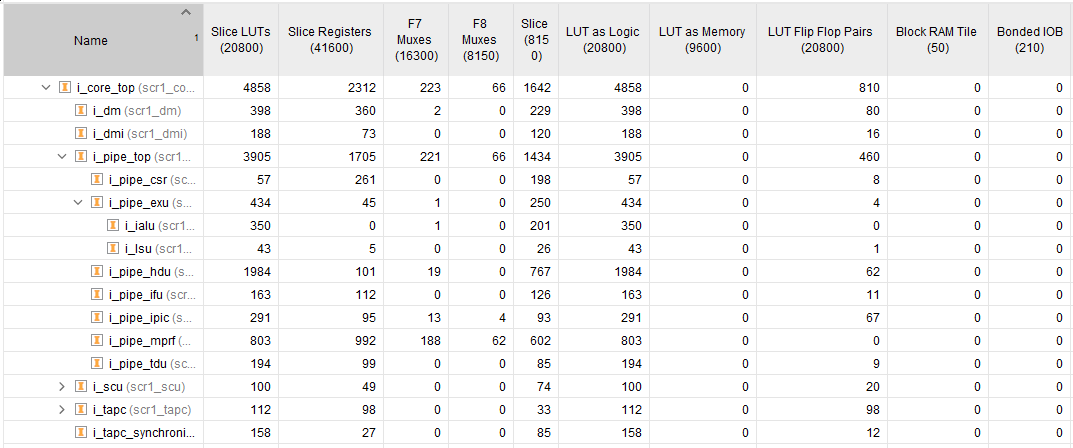
Занимаемая площадь для архитектуры для проекта меняется незначительно. Однако модули MPRF и HDU задействуют значительно больше LUT в конфигурации с EXU\_BYPASS, модуль EXU значительно больше без. Количество регистров значительно меняется только в EXU.

По всей видимости, bypass, который разделяет EXU и IDU, переносит управляющие сигналы в HDU и MPRF.

  
Рис. 2 Report Utilization *RVI -EXU\_BYPASS*

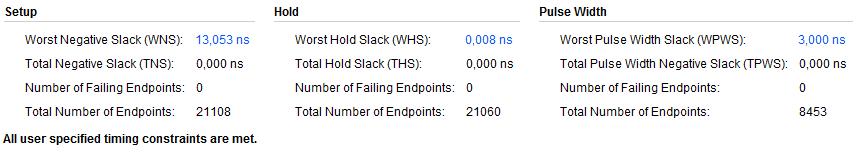
  
Рис. 3 Report Utilization *RVI -EXU\_BYPASS* (подробно)

  
Рис. 4 Report Utilization *RVI +EXU\_BYPASS*

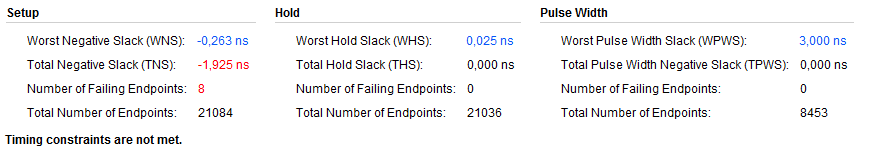
  
Рис. 5 Report Utilization *RVI +EXU\_BYPASS* (подробно)

**Временной анализ**

Результат для архитектуры *RVI -EXU\_BYPASS*.

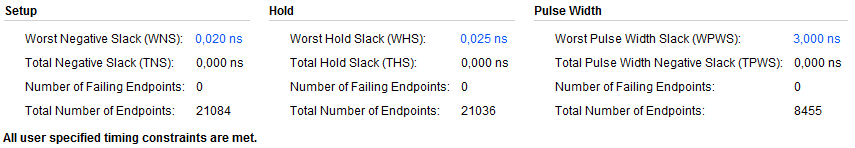
  
Рис. 6 Изначальный Timing summary при 40ns

В файле arty\_scr1\_synth.xdc период SYS\_CLK\_VIRT установлен 20ns, параметр частоты в sys\_pll установлен 50MHz.

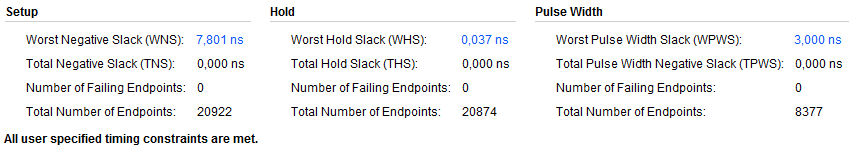
Рис. 7 Timing summary при 20ns

При включенном post-route phys opt design и post-place phys opt design с директивой AggressiveExplore получилось добиться положительного значения WNS.

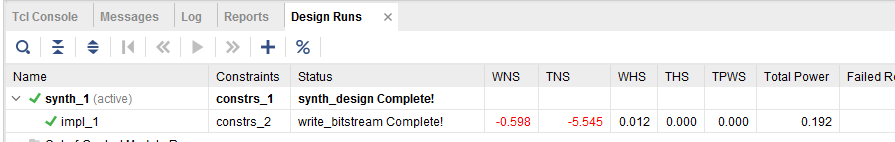
Fmax=50MHz

 Рис. 8 Timing summary при 20ns с настройками

Результат для архитектуры *RVI +EXU\_BYPASS*.

  
Рис. 9 Изначальный Timing summary

В файле arty\_scr1\_synth.xdc период SYS\_CLK\_VIRT установлен 25ns, параметр частоты в sys\_pll установлен 40MHz.

  
Рис. 10 Timing summary при 25ns

При включенном post-route phys opt design и post-place phys opt design с директивой AggressiveExplore получилось добиться положительного значения WNS.

Fmax=40MHz

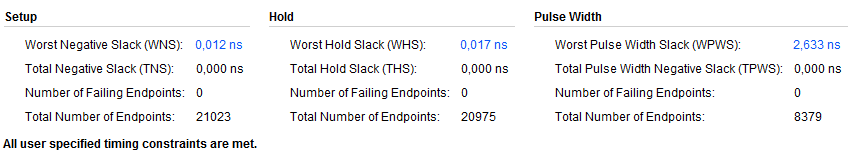


Рис. 11 Timing summary при 25ns с настройками