**Прожирко Владислав, группа P41193**

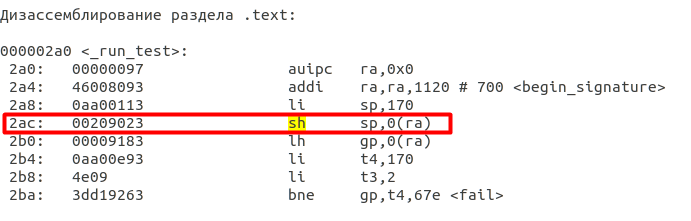
**Lab SCR1 pipeline**

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

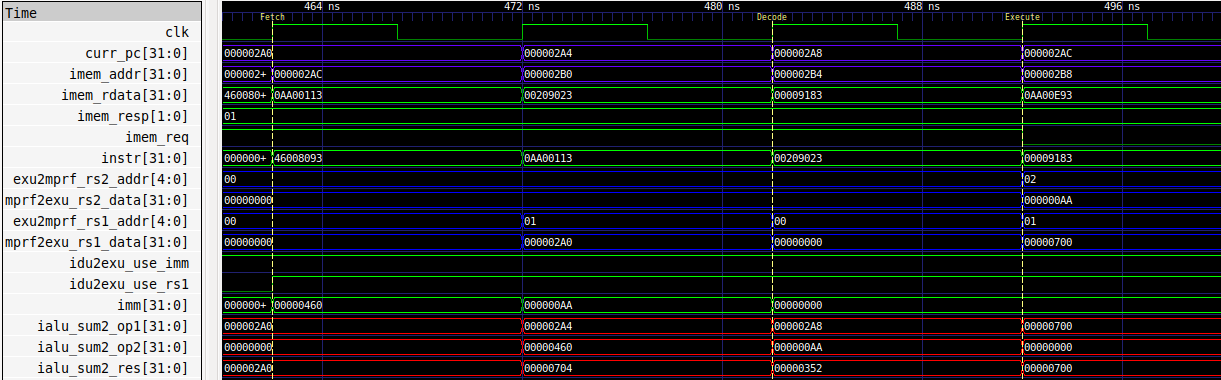
**1 Часть**

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

Отрывок из dump-файла, где используется команда sh:



Результирующая wave-form:

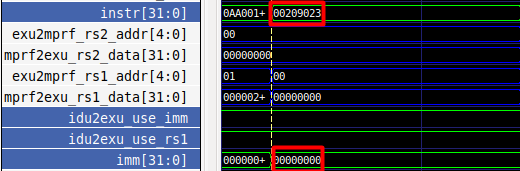


Описание сигналов:

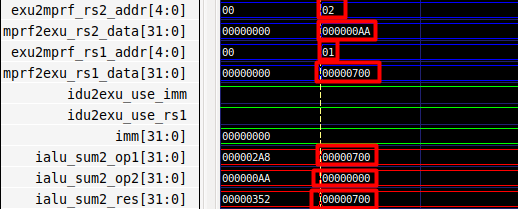
* clk - тактовый импульс
* curr\_pc - текущее значение счетчика команд, соответствует стадии Execution (000002AC)
* набор сигналов для Instruction Fetch:
  + imem\_req - запрос от процессора в память инструкций (1)
  + imem\_addr - адрес запроса памяти инструкций (000002AC)
  + imem\_resp - ответ памяти 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\_data – 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 x2(sp);
  + ialu\_sum2\_op2 – второй операнд АЛУ (00000000) Это смещение;
  + ialu\_sum2\_res – результат АЛУ (00000700)



**2 Часть**

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

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

`define SCR1\_RVC\_EXT \\ Для добавление С архитектуры

И при запуске указывалась IM / IMC архитектура

make run\_verilator\_wf BUS=AHB ARCH=IM/ IMC IPIC=0

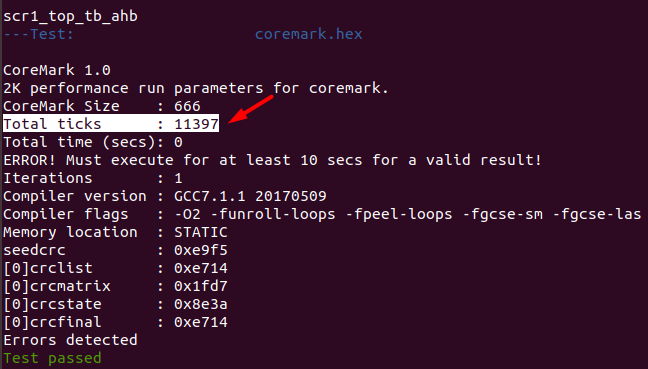
Были получены следующее результаты:

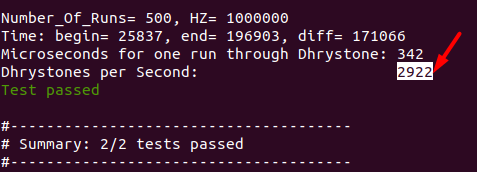
|  |  |  |
| --- | --- | --- |
| Тест | RVIC -IFU\_BYPASS | RVIC +IFU\_BYPASS |
| Coremark (Total ticks) | 11397 Total ticks | 9490 Total ticks |
| Coremark (size memory) | 78,6 Кбайт | 78,6 Кбайт |
| Dhrystone (per Seconds) | 2922 per Seconds | 3133 per Seconds |
| Dhrystone (size memory) | 45,3 Кбайт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