THE UNIVERSITY OF KANSAS

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

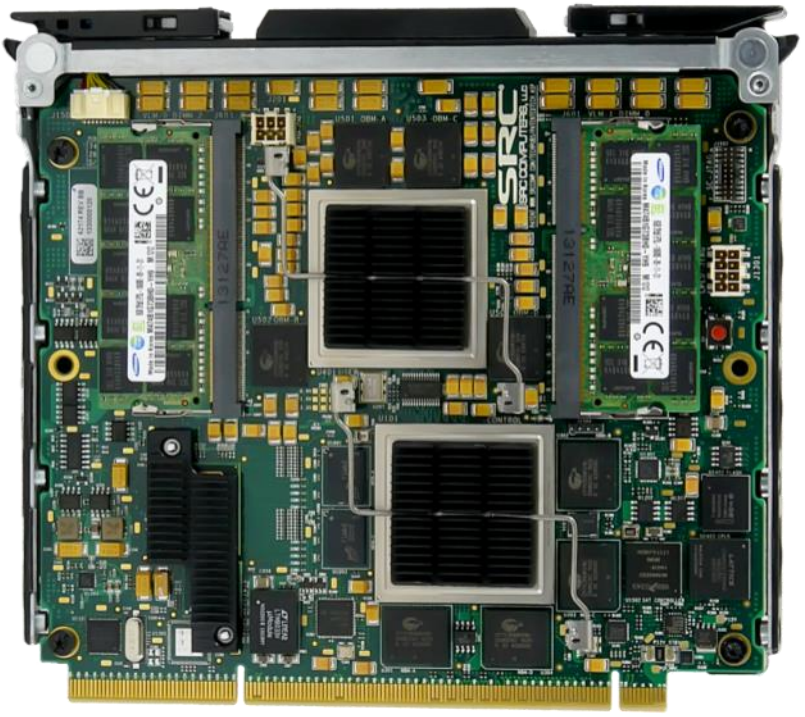
EECS 743 – Advanced Computer Architecture

Spring 2016

Homework 4

Student Name: Student ID:

SRC® Carte™ Saturn Exercises



Lesson 02a: Loop Slowdowns

Objectives:

In this exercise, you will learn what causes the MAP compiler to slow down a pipelined loop.

Background:

Loops containing both reads and writes to the same OBM bank cause some additional added clocks. These come from two sources:

- Interleaved loads and stores to OBM require some "space" between the load and the store → *Structure Hazard*.
- Non-sequential index patterns require enough time to guarantee that a stored value has reached memory before the next read, since the compiler has to assume that the next read might be accessing the value written by the previous iteration → *Read-After-Write (RAW) Data Hazard*.

Exercise 1:

You are given a MAP routine that has two reads from bank A.

V1 - MAP routine ex_loop_slowdowns.mc

```
#include <libmap.h>

void subr (int64_t I0[], int64_t Out[], int num, int64_t *time, int mapnum) {
    OBM_BANK_A (AL, int64_t, MAX_OBM_SIZE)
    OBM_BANK_B (BL, int64_t, MAX_OBM_SIZE)

    int64_t t0, t1;
    int i;

    buffered_dma_cpu (CM2OBM, PATH_0, AL, MAP_OBM_stripe (1,"A"), I0, 1, 2*num*8);

    read_timer (&t0);

    for (i=0; i<num; i++) {
         BL[i] = AL[i] + AL[num+i];
    }

    read_timer (&t1);

    *time = t1 - t0;

    buffered_dma_cpu (OBM2CM, PATH_0, BL, MAP_OBM_stripe (1,"B"), Out, 1, num*8); }
```

- Open the folder “loop_slowdowns_v1”.
- Compile this code in debug mode and note that the compiler reports:

```
###################################################################
##################        INNER LOOP SUMMARY      ##################
loop on line 15:
    clocks per iteration:    2
    pipeline depth:         29
    multiple reads of 'OBM bank A' required 1 additional clock
###################################################################
```

  This is because only *one reference to a bank* can take place *on each clock*. Thus two clocks per iteration are needed to execute the loop.
- Run the provided executable “ex_loop_slowdowns_hwexec_v1” on the MAP, varying the iteration count, and note the expected behavior.
- Derive an expression that predicts the behavior. **(5 Points)**
- Modify the code so that the slowdown is removed. Compile in debug mode and note that the loop is no longer slowed down. **(15 Points)**

  [Hint – use two OBM banks instead of one for the input data.**]**

Exercise 2:

You are given a MAP routine that has two writes to bank B.

V2 - MAP routine ex_loop_slowdowns.mc

```
#include <libmap.h>

void subr (int64_t I0[], int64_t Out[], int num, int64_t *time, int mapnum) {

    OBM_BANK_A (AL, int64_t, MAX_OBM_SIZE)
    OBM_BANK_B (BL, int64_t, MAX_OBM_SIZE)

    int64_t t0, t1;
    int i;

    buffered_dma_cpu (CM2OBM, PATH_0, AL, MAP_OBM_stripe (1,"A"), I0, 1, num*8);

    read_timer (&t0);

    for (i=0; i<num; i++) {
        BL[i] = AL[i] + 9;
        BL[num+i] = AL[i] * 5;
    }

    read_timer (&t1);

    *time = t1 - t0;

    buffered_dma_cpu (OBM2CM, PATH_0, BL, MAP_OBM_stripe (1,"B"), Out, 1, 2*num*8); }
```

- Open the folder “loop_slowdowns_v2”.
- Compile this code in debug mode and note that the compiler reports:

```
########################################################################
##################        INNER LOOP SUMMARY      ######################
loop on line 15:
    clocks per iteration:    2
    pipeline depth:         29
    multiple reads of 'OBM bank B' required 1 additional clock
########################################################################
```

  This is because only *one reference to a bank* can take place *on each clock*. Thus two clocks per iteration are needed to execute the loop.
- Run the provided executable “ex_loop_slowdowns_hwexec_v2” on the MAP, varying the iteration count, and note the expected behavior.
- Derive an expression that predicts the behavior. **(5 Points)**
- Modify the code so that the slowdown is removed **using only two OBM banks**. Compile in debug mode and note that the loop is no longer slowed down. **(15 Points)**

  [Hints:
  - use parallel sections and streamed DMA for data movement.
  - code in folder “streamed_dma_example” serves as a guide for using parallel sections and streamed DMA.**]**

Exercise 3:

You are given a MAP code that reads a 3-element “sliding window” across a vector.

V3 – ex_loop_slowdowns.mc

```
#include <libmap.h>

void subr (int64_t I0[], int64_t Out0[], int num, int64_t *time, int mapnum) {
    OBM_BANK_A (AL, int64_t, MAX_OBM_SIZE)
    OBM_BANK_B (BL, int64_t, MAX_OBM_SIZE)

    int64_t t0, t1;
    int i;

    buffered_dma_cpu (CM2OBM, PATH_0, AL, MAP_OBM_stripe (1,"A"), I0, 1, num*8);

    read_timer (&t0);

    for (i=0; i<num-2; i++) {
        BL[i] = AL[i] + AL[i+1] + AL[i+2];
    }

    read_timer (&t1);

    *time = t1 - t0;

    buffered_dma_cpu (OBM2CM, PATH_0, BL, MAP_OBM_stripe (1,"B"), Out0, 1, num*8); }
```

- Open the folder “loop_slowdowns_v3”.
- Compile this code in debug mode and note the loop slowdown caused by three reads to bank A:

```
########################################################################
##################        INNER LOOP SUMMARY      ####################
loop on line 14:
    clocks per iteration:    3
    pipeline depth:         32
    multiple reads of 'OBM bank A' required 2 additional clocks
########################################################################
```

  This is because only *one reference to a bank* can take place *on each clock*. Thus three clocks per iteration are needed to execute the loop.
- Run the provided executable “ex_loop_slowdowns_hwexec_v3” on the MAP, varying the iteration count, and note the expected behavior.
- Derive an expression that predicts the behavior. **(5 Points)**
- Modify the code so that the slowdown is removed. Compile in debug mode and note that the loop is no longer slowed down. **(15 Points)**

  [Hint – use a shift queue structure, i.e., a set of scalars to hold previously read values.**]**

Submission Instructions

After downloading and extracting the source files to the folder "HW04_source_files":

1) Rename the folder "HW04_source_files" to "HW04_<your last name>", for example "HW04_El-Araby".

2) Complete the code for the following files as you see necessary to implement the required function:
    a. "loop_slowdowns_v1/main.c"
    b. "loop_slowdowns_v1/ex_loop_slowdowns.mc"
    c. "loop_slowdowns_v2/main.c"
    d. "loop_slowdowns_v2/ex_loop_slowdowns.mc"
    e. "loop_slowdowns_v3/main.c"
    f. "loop_slowdowns_v3/ex_loop_slowdowns.mc"

3) Run the "./make_clbr" script in all folders to remove the unncessary files such that each folder contains only the modified source code.

4) Your written solution to exercise 1, exercise 2, and exercise 3 should be appended to the folder "HW04_<your last name>".

5) Compress the main folder to "HW04_<your last name>.zip", for example "HW04_El-Araby.zip" and upload it to blackboard before the due date and time.

NOTE: Homework submission is a "Single Attempt", i.e. carefully review everything that you want to submit before hitting the "submit" button and make sure that you have uploaded all documents you want to submit and have not missed anything.