

Lab 11

- Lab 11: DMA (Direct Memory Access)

Computing Memory Architecture Lab.

Lab 11: Overview

- Analysis
 - Performance measurement tips) time, gettimeofday()
- Performance Tuning
 - memcpy, DMA
- Report
 - DUE: 6월 2일 (수) 23시 59분

Analysis

- [Linux Debugging and Performance Tuning: Tips and Techniques, 2005]
 - Profiling tools
 - stopwatch, date, **time**, clock, **gettimeofday**, gprof, kprof, ...

```
zed@debian-zynq:~/lab9$ time sudo ./a.out
real    0m0.054s  // elapsed real time
user    0m0.040s  // total CPU time in user mode
sys     0m0.000s  // total CPU time in kernel mode
```

```
#include <sys/time.h>
struct timeval start, finish;

int main()
{
    gettimeofday (&start, NULL);
    /* ... */
    gettimeofday (&finish, NULL);
    //msec = finish.tv_sec * 1000 + finish.tv_usec / 1000;
    //msec -= start.tv_sec * 1000 + start.tv_usec / 1000;
}
```

Performance Tuning

Performance Tuning 1. MEMCPY

- memcpy: copy block of memory

- void * memcpy (void * destination, const void * source, size_t num);
 - destination: pointer to the destination array
 - source: pointer to the source of data to be copied
 - num: number of bytes to copy

```
gettimeofday (&st[2], NULL);  
// memory load  
for (i = 0; i < SIZE * (SIZE + 1); i++)  
{  
    *(fpga_bram + i) = flat[i];  
}  
gettimeofday (&st[3], NULL);
```

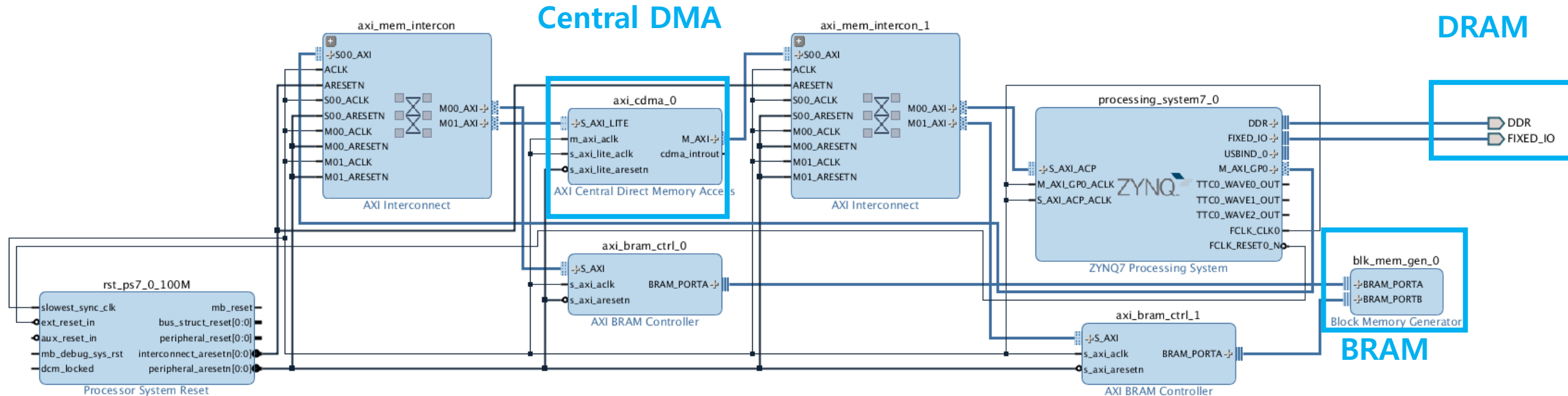
-> **852us**

```
gettimeofday (&st[2], NULL);  
// memory load  
memcpy( fpga_bram, flat, 16640 );  
gettimeofday (&st[3], NULL);
```

-> ?

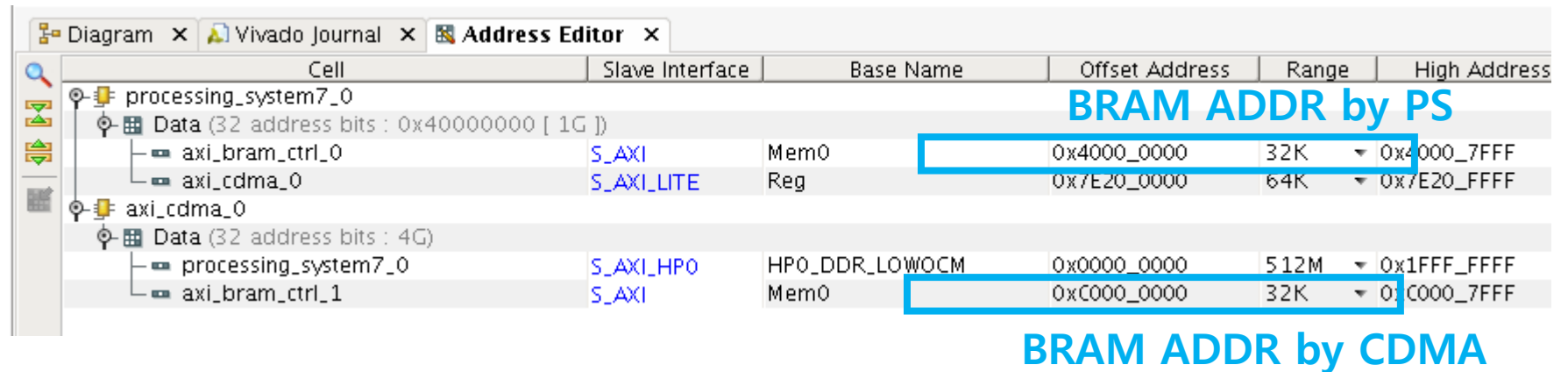
2. DMA – Block Design

- Tutorial on Direct Memory Access
- Create a Vivado project and run block_design_dma.tcl
- Create HDL Wrapper / Generate Bitstream file



DMA – Programmer's Perspective

- BRAM is @ 0x4000_0000 (by PS) and 0xC000_0000 (by CDMA)
 - CDMA operation (PG034)
 - Assign source address (SA)
 - Assign Destination address (DA)
 - Assign bytes to transfer (BTT)
 - Poll if the operations is done (CDMASR.IDLE)

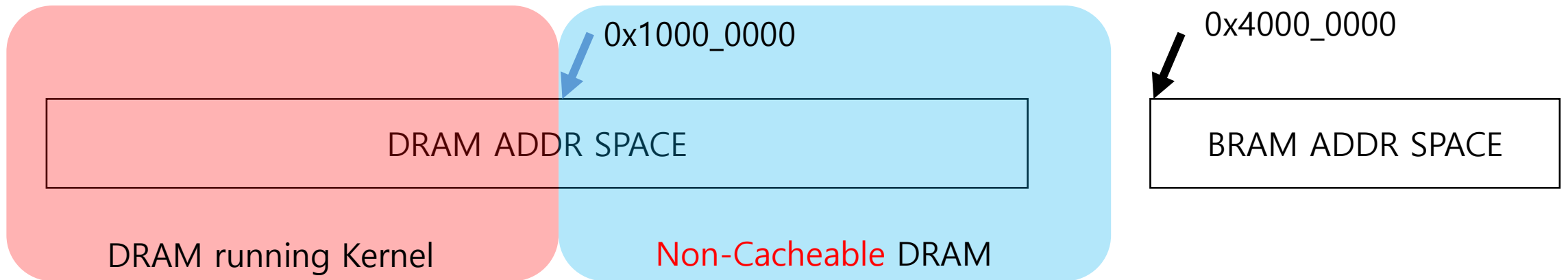


The screenshot shows the Vivado Address Editor window with a tree view on the left and a table of memory addresses on the right. The tree view shows a hierarchy starting with 'processing_system7_0', which contains 'Data (32 address bits : 0x40000000 [1G])' and 'axi_cdma_0'. The 'Data' node contains 'axi_bram_ctrl_0' and 'axi_cdma_0'. The 'axi_cdma_0' node contains 'Data (32 address bits : 4G)' which in turn contains 'processing_system7_0' and 'axi_bram_ctrl_1'. The table on the right lists the memory addresses for these components. The 'axi_bram_ctrl_0' entry is highlighted with a blue box and labeled 'BRAM ADDR by PS'. The 'axi_bram_ctrl_1' entry is highlighted with a blue box and labeled 'BRAM ADDR by CDMA'.

Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
axi_bram_ctrl_0	S_AXI	Mem0	0x4000_0000	32K	0x4000_7FFF
axi_cdma_0	S_AXI_LITE	Reg	0x7E20_0000	64K	0x7E20_FFFF
axi_cdma_0					
Data (32 address bits : 4G)					
processing_system7_0	S_AXI_HP0	HP0_DDR_LOWOCM	0x0000_0000	512M	0x1FFF_FFFF
axi_bram_ctrl_1	S_AXI	Mem0	0xC000_0000	32K	0xC000_7FFF

DMA – New Bootloader

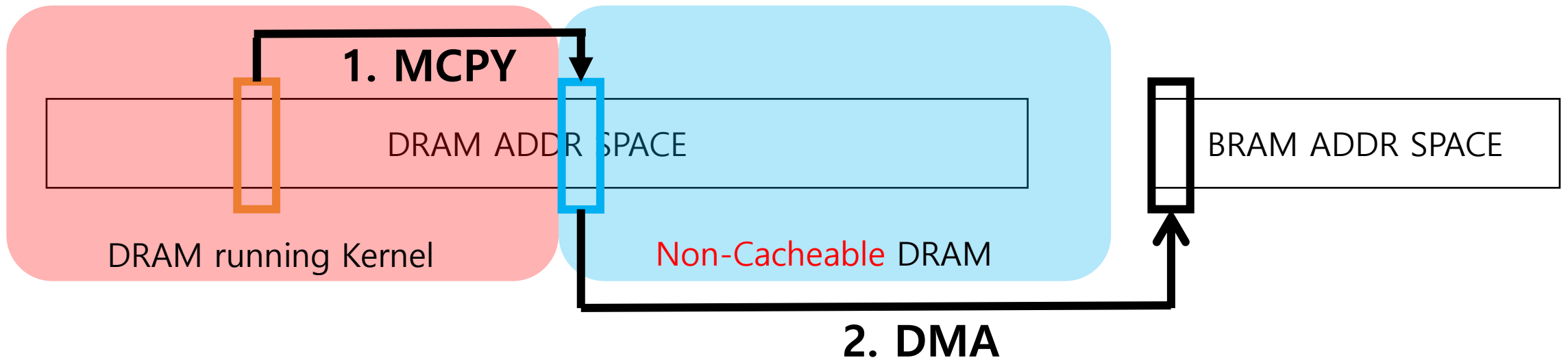
- Update the boot loader to enable non-cacheable DRAM
 - Switch u-boot.img with u-boot.img.nc
 - mv u-boot.img.nc u-boot.img
 - Ref) cat /proc/meminfo



DMA – Application

- Run a sample c-code to execute DMA operation
 - BOARD\$ git clone https://github.com/tahsd/hsd21_lab11.git
 - BOARD\$ make

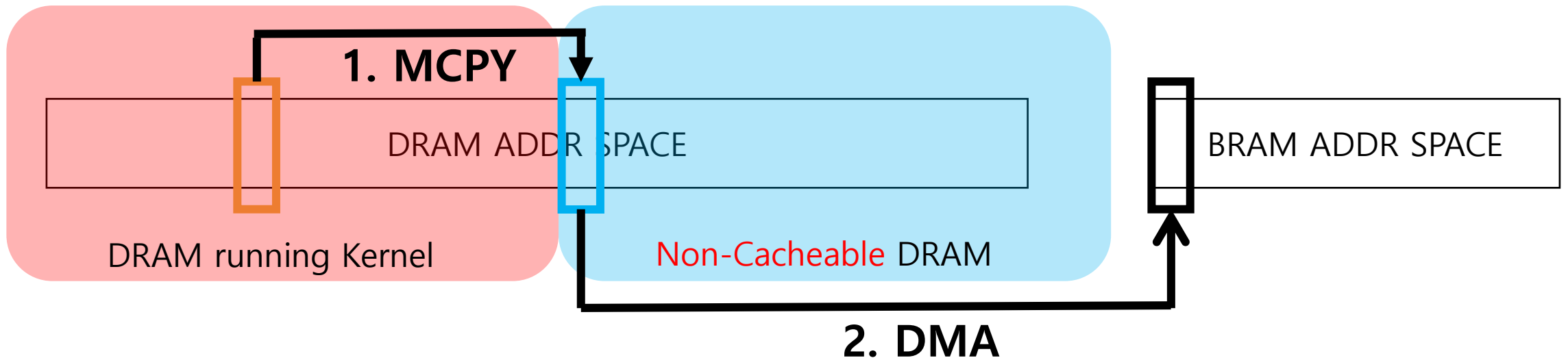
```
// 1. MCPY  
gettimeofday (&st[2], NULL);  
memcpy( ps_dram, flat, SIZE * (SIZE + 1) * sizeof(float) );  
gettimeofday (&st[3], NULL);
```



DMA – Application

- Run a sample c-code to execute DMA operation

```
// 2. DMA
gettimeofday (&st[2], NULL);
*(fpga_dma+6) = 0x10000000; // SA
*(fpga_dma+8) = 0xC0000000; // DA
*(fpga_dma+10) = SIZE * (SIZE + 1) * sizeof(float); // BTT
while ((*fpga_dma+1) & 0x00000002) == 0; // CDMASR.IDLE
gettimeofday (&st[3], NULL);
```



Main Practice

Practice

- Follow tutorial & understand the DMA functionality
 - Run a given block design in your FPGA
 - Run and understand DMA function
 - Refer to the CDMA product guide and analyze the main code
 - In our term project optimization, you will modify the block design
 - Term project = PS + BRAM + Connectivity + Custom IP (MM multiplier)
+ CDMA
- Compare data transfer using CDMA versus transferring data through CPU
 - You should modify main.c for transferring data through CPU.

Homework

- Requirements

- Result

- Attach a screenshot of main.c result with [student_number, name]

- Report

- Write down all the points mentioned on the previous page
 - In your own words
 - Either in Korean or in English
 - # of pages does not matter
 - **PDF only!!**

- **Result + Report to one .zip file**

- Upload (.zip) file on ETL

- Submit one (.zip) file

- zip file name : [Lab11]name.zip (ex : [Lab11]홍길동.zip)

- Due: 6/2(WED) 23:59

- **No Late Submission**