Australian National University

Computer Science Courses
ANU College of Engineering & Computer Science

Week 11

Students are expected to attempt ALL of the questions in the Multi choice, Short answer and Lab quesitons. They will be discussed and marked at the beginning of the lab.

Multi choice

Question: Which one of these is the fastest form of memory? B	**A)** L1 Cache **B)** Registers **C)** DRAM **D)** L2 Cache

Question: Which of these are likely to use SRAM instead of DRAM? A	**A)** Registers **B)** CPU Caches **C)** Device buffers **D)** All of the above **E)** A and B

Question: Which of these is the correct formula to find the cache hit rate? B	**A)** hits/misses **B)** hits/accesses **C)** accesses/misses **D)** misses/hits **E)** accesses/hits

Question: What are typical lengths for cache lines? D	**A)** 4MiB would be a minimum of the typical length **B)** 4KiB would be typical **C)** 1 or 2 bytes **D)** 64, 128 bytes in length

Short answer

process is one application, thread is the executor with independent memory

Question: What is the difference between a process and a thread?

Question: Define the terms "direct-mapped cache" and "E-way set-associate cache". What are typical values for E?

The most common E is 64.

Question: Consider a program which repeatedly accesses all elements in an `int` array of of length `N` in a cyclic fashion (from first to last element). For example, the computation could be:

```
while (1) {
  int s = 0, i;
  for (i = 0; i <= N ; i++)
    s += a[i];
}
```

Suppose the first iteration has already been completed. Assume that the cache line size is 4 bytes, the cache holds 2^{16} bytes, and the cache is fully associative (note: this is not a realistic cache!), and the replacement policy is *least recently used*. Calculate the *cache hit rate* for

- (a) `N = 1 * 2^13`,
- (b) `N = 2 * 2^13`,
- (c) `N = 3 * 2^13` and
- (d) `N = 4 * 2^13`.

(a) 100%
(b) 100%
(c) 0%
(d) 0%
int occupies 4 bytes by default.

Hint: calculate the relative size of `a[]` to the cache first.

Explain whether a *random* replacement policy would perform better or worse in each case. perform better.

Recall that the *cache hit rate* is calculated as the percentage of (in this case `int`) load/store operations that occur when the data is in the cache.

Question: Repeat the above for a direct-mapped cache (note that the replacement policy plays no role in this case).

Lab questions

Excercise: Below is a simple program that prints Hello World! in x86 assembly language. Cut-and-paste this program into a file called "hello.s" and assemble and link it using the commands below.

```
# Simple Hello World in x86 assembler
# To compile and run execute:
#  %  gcc -c hello.s
#  %  ld -s -o hello hello.o
#  %  ./hello
```

```
.data
hellostr:
        .string "Hello World!\n"
.text
.globl _start
_start:
        movl    $4,%eax    # write(1,hellostr,13)
        movl    $1,%ebx
        movl    $hellostr,%ecx
        movl    $13,%edx
        int     $0x80

        movl    $1,%eax    # exit(0)
        movl    $0,%ebx
        int     $0x80
```

It works by making two system calls. The first is a write system call which writes the string to standard out. The second is the exit system call. The parameters of the system call are loaded into the general purpose register (%eax, %ebx, %ecx, and %edx) and then the "int $0x80" instruction is call (like the trap instruction rPeANUt). The %eax holds the system call number (4 for write, and 1 for exit). The "movl source,destination" instruction moves a long word from the source to the destination. Note, immediate values start with a $.

Add comments to the above code which states the meaning of each of the parameters in the system calls.

Modify the above program such that it prints "Hello World!" in an infinite loop. For this you will need to use a "jmp" instruction and add a label.

Modify the code again such that in an infinite loop it reads characters for the standard input stream and outputs them directly to the standard output stream. Note that, the system call number for read system call is 3 and it has similar parameters to that of write. Also the return value of the system calls is stored in the register %eax.

Test your program by typing something into the terminal and you should see what you typed in echoed back out.

(option) Modify the code such that it converts all the characters to upper case.

In-class Group Task

This group task should be done in groups of 2 or 3. Copy the following code into rPeANUt, then in the "Code" menu, select "Show cache simulator".

Run the program with different cache settings and record what times the program finishes with. Comment on what the differences are in the different types. In a realistic environment, why would set associative cache be chosen over fully associative cache?

```
0x0100:
loop:
        call subvar
        call fill
        call print
        load var R0
        jumpnz R0 loop

subvar:
        load var R0
        sub R0 ONE R0
        store R0 var
        return

fill:
        move R0 R1
        move R1 R2
        move R2 R3
        move R3 R4
        move R4 R5
        move R5 R6
        move R6 R7
        return

print:
        load #0 R0
printloop:
        load R0 #string R1
        jumpz R1 printend
        store R1 0xFFF0
        add R0 ONE R0
        jump printloop
```

```
printend:
        return

var:
        block #100

string:
        block #"Hello World"
```

This group task should be done in groups of 2 or 3. Obtain a copy the memoryrand.c code, compile and run it. Read through the code and work out what the program is intended to do. If every memory read took exactly the same amount of time what would you expect the output to report? Does changing the compiler optimization level change the results? What do the results tell you about the system you are running on?

UPDATED: **22 April 2013** / RESPONSIBLE OFFICER: **Head of School** / PAGE CONTACT: **COMP2300 Course Webmaster**