CS 241 - Foundations of Sequential Programs **Spring 2017**

Lecture 5: May 16th, 2017

Lecturer: Kevin Lanctot *Notes By: Harsh Mistry*

Arrays

Arrays are stored as a sequence with each element being 4 bytes. So, given a starting address, often referred to as the **Base Address**. So, you need to add 4 for every element in the array

We spent majority of today's lecture on unnecessary examples, Please refer to the course slides if you wish to read these example.

Memory Mapped I/O

Input/output from devices are treated just like reading and writing to memory. For CS241, to output a char to the screen, store the ASCII value of the character to memory address `FFFF000C`$_{hex}$.

- Bytes written to this address appear on screen
- You must write one ASCII character at a time.

Subroutines

Subroutines are similar to functions, but require additional work such as creating labels, monitoring PC, and loading/storing values in memory.

- `jr $s` : Jumps to address stored in `$s`.
- `jalr $s` : Will jump to address stored in `$s`, but will store the address of the next instruction (PC) in `$31`

Storing Essential Data

When a new subroutine is called, any data that is required after the subroutine returns, should be backed up into memory. This is referred to as **current execution context**

Call Stack

To store store values in memory, a stack should be used. Stacks by convention grow downward. To ensure the bottom of the stack can be accessed, `$30` is used in CS241 to hold the memory location of the bottom of the stack (`$31` is often referred to as the stack pointer). As a result, values should be stored where the stack pointer resides before calling a subroutine.