CIS 350 – INFRASTRUCTURE TECHNOLOGIES
HOMEWORK #4

Student Name(s):_____
(You may do this homework in groups of 2 students maximum.)

Topics: The CPU and Memory (Chapters 7 and 8)

For short essay questions, your answers should capture the essence of the questions. There is no credit for 1- or 2-sentence answers.

Ex. 7.2, p. 230

A register is a component of the CPU that is permanent and is designed to complete a specific task. They are used to temporarily hold binary values to complete calculations, store, or manipulate. The calculator/accumulator in the LMC is similar to a general-purpose register on the CPU. Using a two-digit memory location in the LMC is like a memory address register. Other values in a register include instruction, memory data, and status registers.

Ex. 7.23, p. 231

A stack stores data that uses LIFO methodology. This means that the last thing that came onto the stack is the first thing to leave the stack. New data is 'pushed' to the top of the stack, and to retrieve data the last number is 'popped' from the top. This is an efficient method for storing intermediate data. [push/pop diagrams will be drawn]

Ex. 7.3, p. 232

Using the example in figure 7.18, a 32-bit memory address divides the binary digits into 8 bits for the op code and 24 bits for the address field. Using the same ratio, 12 bits should be used for the op code and 36 bits for the address field in a 48-bit computer. $2^{12} = 4096$ different instructions and $2^{36} = 68,719,476,736$ memory addresses. In total, there are $2^{48} = 281,474,976,710,656$ memory locations.

Ex. Suppose that the following instructions are found at memory locations 15 and 16. Suppose that the following data are found at memory 25 and 26.

Address	Instruction	
15	LDA 25	
16	SUB 26	Addresses 15-16 represent the program area
	Data	
25	120	Addresses 25-26 represent the data area
26	50	

Show the contents of the PC, the MAR, the MDR, the IR, and the A as each step of the fetch-execute cycle is performed for instructions at addresses 15 and 16. (The machine cycle for instruction LDA I worked in class on the white board would be helpful. See page 5 in the lecture notes for chapter 7. I asked students to take notes. Also, look at Assignment One in in-class small group activity #4.)

Instruction: 15 LDA 25

	PC	MAR	MDR	IR	A
1. PC→ MAR	15	15	LDA 25	?	?
2. MDR→ IR	15	15	LDA 25	LDA 25	?
3. IR[addr] → MAR	15	15	120	LDA 25	?
4. MDR→ A	15	15	120	LDA 25	120
5. PC+1→ PC	16	15	120	LDA 25	120

Instruction: 16 SUB 26

	PC	MAR	MDR	IR	A
1. PC→ MAR	16	16	SUB 26	LDA 25	120
2. MDR→ IR	16	16	SUB 26	SUB 26	120
3. IR[addr] → MAR	16	16	50	SUB 26	120
4. A – MDR → A	16	16	50	SUB 26	70
5. PC+1→ PC	17	16	50	SUB 26	70

Short essay questions. Your answers should capture the essence of the questions. There is no credit for 1- or 2-sentence answers.

Ex. 8.13, p. 263.

In cache write-through the data is immediately written back to the main memory when a change in made in the cache. This is the safer method because it ensures the data is stored somewhere other than the cache. In write-back the data changes are only reflected in the cache. This is faster as there are less steps involved, but there is more care required to prevent data loss.

Ex. 8.14, p. 263.

When cache memory is full, some data must be removed to make room for other data. The most common method is the least recently used method. It replaces cache objects that have are the oldest in the cache that have not been used by the CPU. It may also optionally be written to main memory if keeping the data is crucial to an application.

Ex. 8.15, p. 263.

The locality of reference principle states that most memory references will only use one or few small memory regions. By utilizing cache memory to its utmost efficiency, it creates hit ratio efficiencies upwards of 90%. This increases program execution speed by over 50%. This allows access to memory data at almost instantaneous speed that the instruction was executed.

Ex. 8.18, p. 263

The two different ways of configuring a multiprocessing system are master-slave multiprocessing and symmetrical multiprocessing. In master-slave, the master manages the system overall (resources/scheduling) and assigns work to the slaves. In symmetrical, each CPU works identically to the other and, communicating through the operating system, will schedule work for itself. For general purpose computing, symmetrical is best because it keeps each core on a balanced workload. Master-slave is more efficient for specialized tasks because the master must run game code and assign calculations to the other cores.