Brown University
School of Engineering
EN164 Design of Computing Systems
Professor Sherief Reda
HW 06. Not due.

1. Communicating with I/O devices is achieved using combinations of polling, interrupt handing, memory mapping and special I/O command. Specific I/O commands are a mode of I/O communication in which specific instruction (e.g., `in` and `out`) Answer the following questions about communicating with I/O subsystems for the following applications using combinations of these techniques

a. Auto Pilot
b. Automated thermostat

1. Describe device polling. Would each application in the table be appropriate for communication using polling techniques? Explain.
2. Describe interrupt driven communication. For each application in the table, if polling is inappropriate, explain how interrupt driven techniques could be used.
3. For the applications listed in the table, outline a design for memory mapped communication. Identify reserved memory locations and outline their contents.

2. Direct Memory Access (DMA) allows devices to access memory directly rather than working through the CPU. This can dramatically speed up the performance of peripherals, but adds complexity to memory system implementations. Explore DMA implications by answering the questions about the following peripherals

a. Mouse controller
b. Ethernet controller

1. Does the CPU relinquish control of memory when DMA is active? For example, can a peripheral simply communicate with memory directly, avoiding the CPU completely?
2. Of the peripherals listed in the table, which would benefit from DMA? What criteria determine if DMA is appropriate?
3. Of the peripherals listed in the table, which could cause coherency problems with cache contents? What criteria determine if coherency issues must be addressed?
4. Describe what problems could occur when mixing DMA and virtual memory. Which of the peripherals in the table could introduce such problems? How can they be avoided?

3.

Assume you multiple a m x n matrix A by n x p matrix B to give the m x p matrix C = AB, where

$C_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \ldots + a_{i,n}b_{n,j}$

Assume that the matrices are stored in the memory in row order.

1. Assume C is computed on a single-core shared memory machine and a 4-core shared-memory machine. Compute the expected speed obtained on the 4-core machine, ignoring any memory issues.
2. Repeat the first question assuming that updates to C incur a cache miss due to false sharing when consecutive elements in a row are updated.
3. How would you fix the false sharing problem?