Luke Jaffe
ID: 001182531
Microprocessor-Based Design
Due 03/14/16

Homework 3
Interrupts, Communication Systems

Problem 3.1: Operation Modes

a. The ARM processor distinguishes between a “User” and a “Supervisor” mode. Assume that you are implementing a file system driver for a system that can execute multiple applications at the same time. Which of the two modes would you use for executing your sound driver? Please show your reasoning for choosing one particular mode.

Since device drivers, including a sound driver, have close interaction with the kernel, they should be run in kernel mode. User mode should be reserved for higher level applications with fewer system calls. Code with many system calls will incur a large overhead for user-kernel mode transitions. Further, since the sound driver is executed in kernel mode, it must be written carefully to avoid unintended vulnerabilities because of its privilege level.

Problem 3.2: Interrupt Concept

a. What is an interrupt?

An interrupt is a signal to the processor produced by hardware of software which indicating an event that should be immediately serviced.

b. What happens to the execution of the currently running task once an interrupt occurs? What happens after interrupt processing has finished?

When an interrupt occurs, the process is suspended, the current machine registers are saved, the privilege is upgraded from user to kernel mode, and execution jumps to the interrupt handler. When the interrupt has finished processing, exeuction jumps back to the interrupted process, the privilege is downgraded from kernel mode to user mode, and the machine registers are restored.

c. What is the difference between an IRQ and an ISR?

The IRQ is the interrupt request, and the ISR is the interrupt service routine. When the processor receives an interrupt request, the current process is suspended and execution jumps to the interrupt service routine.

Problem 3.3: Polling

a. This polling has several disadvantages. Please name 3 disadvantages and briefly explain the effect onto embedded systems.

1. Waste CPU: While a processor iterates through the loop, it is wasting cycles which code be used on other code. When using interrupts instead, the processor will be able to execute useful code most of the

time, and only handle interrupts when they are generated.
2. Waste power: The core executing the loop would be 100% consumed for the duration of the program, which would be a tremendous waste of power as compared to using interrupts, which require comparatively little CPU time.
3. Inefficient: In the looping method, the code must check the data register at each iteration (or simply set the led values in the data register to the switch values). Either way, this is a lot of additional code being executed since the loop is iterating as fast as possible. In the interrupt method, this code is only executed when an interrupt is triggered.

Problem 3.4: Interrupt Latency

a. Define the concept of interrupt latency.

Interrupt latency is the period from when an interrupt request is generated to when the interrupt service routine begins executing.

b. Why is it for embedded systems particularly important to have a known, bounded interrupt latency?

Embedded systems are frequently used for hard real-time applications, which must be responsive to events within rigid intervals. Interrupt latency must be known and bounded because certain interrupts could correspond to critical events. These critical events must be serviced within the described rigid intervals, or the system may fail and cause danger to people.

c. Please name and explain 4 contributors to interrupt latency starting with the largest contributor.

1. Waiting for pending instructions to finish
    -depends on data access of current instruction, could take up to milliseconds for a disk access
2. Higher priority interrupts
    -higher priority interrupts must be finish being serviced before lower priority ones can begin
3. Saving machine registers
    -the machine registers of the current process must be saved before the ISR can begin
4. Transitioning to kernel mode
    -the processor must transition from user to kernel mode before the ISR can begin

Problem 3.5: Interrupt Impact

a. What is the total IRQ execution time (incl. interrupt overhead) over the CPU execution time? In other words, how much is the CPU utilization purely due to this one interrupt type? Define the formula, show your calculation.

If the interrupts of this type occur periodically with period t_{period} and duration $t_{duration}$, then the % utilization of the CPU due to this interrupt type is $t_{duration} / t_{period}$.

b. At what IRQ period would the CPU reach 100% utilization for processing the interrupt?

When $t_{duration} = t_{period}$, the CPU reaches 100% utilization for processing the interrupt.

c. Name and briefly outline a principle to protect an embedded system against a denial of service attack with too many interrupts.

Too protect an embedded system against a denial of service attack with too many interrupts, rate limiting can be used. This means limiting the rate at which interrupts may be received by the system. This rate should be set significantly slower than the CPU clock frequency.

Problem 3.6: Zynq SoC Interrupt Handling

a. An interrupt from a PSGPIO can be blocked at three different levels. Please name these three levels and show how to enable and disable the interrupt processing.

1. Low-level interrupt handler
2. System interrupt handler
3. Execute user interrupt handler

Interrupt processing is enabled and disabled through the GPIO Interrupt Enable/Disable registers. Setting a given bit high in either register will complete the corresponding action, enabling or disabling the interrupt for that pin.

Problem 3.7: Communication Structures

a. Compare and contrast serial and parallel buses. What are the strong points of each and how are they typically used.

A bus is a set of wires with a protocol. A serial bus sends one bit at a time, whereas a parallel bus sends multiple bits at a time. Serial buses have a single data wire, but can transport data long distances (off-chip). They have more complex interfacing logic and communication protocol than parallel busses. Parallel buses multiple data wires, so can achieve higher data throughput, but must be used for communication over shorter distances (on-chip).

b. What is the distinction between master and slave in a communication system?

The master defines the clock signal, and determines when data is sent. The slave uses this clock signal to respond to requests issued by the master.

c. What is arbitration?

Arbitration is the policy which determines which master has access to the bus.

Problem 3.8: I2C

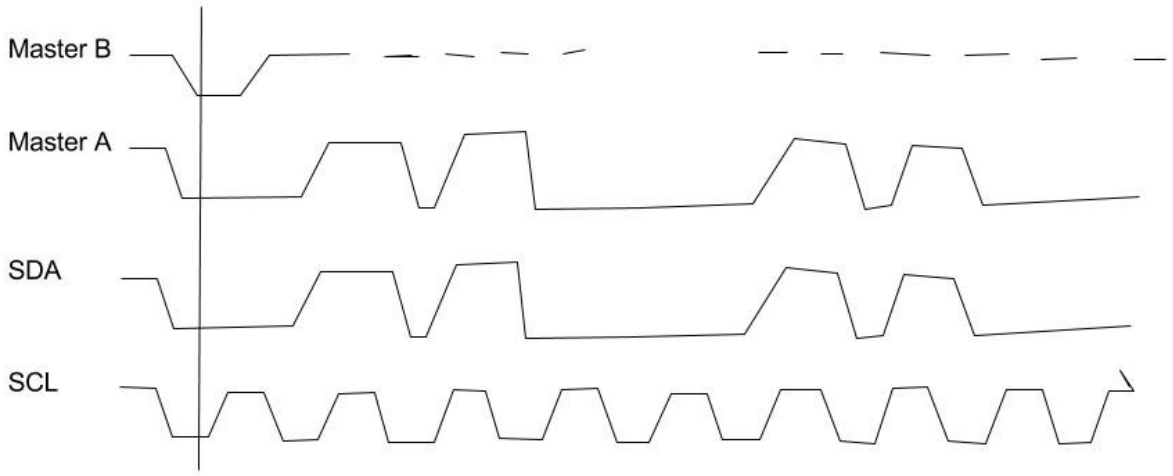
a. The I2C uses two wires. How does it handle access by multiple masters / multiple bus nodes? Touch upon the physical access, arbitration and how these two are tightly linked.

To handle access by multiple masters/ multiple bus nodes, I2C employs the following method: first, the sender listens when sending and address. If the sender hears a conflict, it stops signaling. Because of wired-and logic, the one sending a '1' will hear the conflict and stop sending. The one sending a '0' will not hear the conflict and keep sending.

b. Two masters (A and B) access the same I2C bus. One is sending a message with the id 0100110b and the other uses 1100110b. Both masters start communication at the same time. Draw a timing diagram

for the first byte of the data frame that shows the clock (SCL), what each node attempts to write (SDL A, SDL B), as well as the resulting status of the data line SDL. Which message wins the arbitration?

Assuming bits are placed on the line starting with the most significant bit, there is a conflict in the first cycle, and Master A wins the arbitration since it sent the '0'.



c. Assume an I2C EEPROM of type AT24C64B (http://www.atmel.com/devices/AT24C64B.aspx). It operates on an I2C bus at 400kHz clock frequency. Assume “Random Read” transactions as outlined in “Figure 9-2. Random Read” of its datasheet. What is the effective data bandwidth considering back to back transactions (without any cycle delay in between and no wait cycles)? Show your calculations.

It takes 39 clock cycles to read a single byte with the Random Read method. If the clock cycle is 400kHz, the effective data bandwidth is 400,000/39 = 10,256 bits/second.

d. Assume the access above happens to be sequentially increasing. Is there a better mode of operation? What is the achievable bandwidth? Show your calculations.

If the operation above happens to be sequentially increasing, then the Sequential Read operation would be more effective. This has peak efficiency of only 9 cycles to read a single byte, meaning the max achievable bandwidth is 400,000/9 = 44,444 bits/second.