HAND IN

answers recorded on question paper

Queen's University
Faculty of Engineering and Applied Science
Department of Electrical and Computer Engineering
ELEC 371 Microprocr. Interfacing & Embedded Systems

Final Examination

15 December 2018

Instructor: Dr. N. Manjikian

Student ID (please print clearly)

✱

- **No aids are permitted (NO CALCULATORS, no notes, no textbooks).**
- **The duration of the exam is three hours.**
- **There are six sections (questions). Attempt all of them.**
- **The pages of this examination must remain stapled together.**
- **Proctors are unable to respond to queries about the interpretation of exam questions. Do your best to answer exam questions as written.**

1	/15
2	/15
3	/10
4	/15
5	/15
6	/30
Total	/100

Question 1: Bus Interface and Address Decoding (15 marks)

(a) A system with a 16-bit address space consists of a five-step processor, ROM, RAM, and three I/O devices. ROM is 16 kbytes, RAM is 8 kbytes, and each I/O is 4 kbytes. ROM is at address 0x0000. RAM is *immediately after* ROM. Design & show address decoding logic, and fill in the table.

Device	Start	End

(b) Within an FPGA, *briefly* describe the nature of the address line connections.

(c) Within an FPGA, *briefly* describe data line connections *FROM* the processor.

(d) Within an FPGA, describe or show the data line connections *TO* the processor.

Question 2: Bus/Memory Interface Timing (15 marks)

This question uses the configuration from *Question 1(a)* *of this examination.*

Consider the following instruction at the specified address in memory.

address	*contents*	*corresponding instruction*
0x00002B34	0x018ACD17	ldw r6, 0x2B34(r0)

Complete the timing diagram below for five cycles constituting the execution of the *ldw* instruction above. Use '——' for unknown or irrelevant waveform values.

	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5
clk					
mem_addr_out					
mem_read					
mem_write					
rom_active					
ram_active					
1st I/O device active					
2nd I/O device active					
3rd I/O device active					
data_from_procr					
data_to_procr					

Question 3: Parallel Port Programming (10 marks)

A system based on the Nios II has an *output parallel port* with a *data register* and a *status register*. Bit 2 is the flag bit in the status register. Both registers in the parallel port interface are word-sized. Assume that there are symbols PORT_DATA and PORT_STATUS defined in an assembly-language source file, with associations to the relevant addresses for the interface registers. *The addresses fit in 16 bits.*

Write a modular subroutine `SendDataToPort(list_ptr, n)` in *Nios II assembly-language* which accepts a pointer to a list of word-sized elements and the number of elements in the list. The subroutine should use a loop to transfer elements of the list to the external device attached to the output port described above.

add/addi/sub/subi, and/andi/or/ori/andhi/orhi, ldw/ldb/ldwio/ldbio, stw/stb/stwio/stbio, mov/movi/movia, br/beq/bne/blt/bgt/ble/bge, slli/srli, call/ret, trap/eret, rdctl/wrtcl, registers r0 to r31 (r0 always 0), sp/ra/ea/et are register aliases, other special registers are status/estatus/ienable/ipending, directives .equ/.text/.org/.global/.word/.byte/.skip

Question 4: Interrupt Hardware and Programming (15 marks)

This question concerns interrupt hardware and software for the Nios II processor.

(a) Briefly, in one line, indicate the purpose of the processor registers below.

ienable:

ipending:

status:

estatus:

(b) Describe or show the logic in a parallel port that just generates an IRQ signal, where the port has a data register, a control register, and a status register.

(c) Assume that a system has two parallel ports A, B that generate interrupts. Assume asm.-lang. subroutines *HandlePortA* and *HandlePortB* that perform the necessary processing to respond to port A, B interrupts. Show ***complete*** and ***modular*** asm.-lang. code for an *exception handler* that checks hardware interrupt sources, invokes subroutines, and returns to the main program.

ISR:

Question 5: Embedded Systems and System-on-a-Chip Design (15 marks)

(a) Embedded systems are characterized as being __________, which means that

____

____.

(b) Microcontroller chips are typically designed to achieve three objectives:

____________________, ____________________, and ____________________.

(c) Identify and briefly describe *three design issues* related to embedded systems.

(d) Explain two ways in which embedded systems *differ* from general computers.

(e) Identify and describe the two forms of processors found within FPGA chips. (This question is *not* about different variations of the Nios II processor.)

(f) Identify the two ways in which the circuitry within an FPGA chip is configured. Indicate the situation/circumstances associated with each configuration method.

Question 6: Embedded Application Programming in C (30 marks)

A system-on-a-chip FPGA hardware configuration has been defined as follows:

Nios II processor	(no address range)
ROM	(0x00000000-0x00003FFF)
timer 0	(0x00005000-0x0000500F)
timer 1	(0x00006000-0x0000600F)
8-bit input port data reg.	(0x00006A00)
8-bit output port data reg.	(0x00006B00)
JTAG UART	(data reg. 0x00006C00, status reg 0x00006C04)
RAM	(0x00007000-0x00007FFF)

Address	15 ... 3	2	1	0	
0x00005000			RUN	TO	Status
0x00005004	STOP	START	CONT	ITO	Control
0x00005008					Start (lo)
0x0000500C					Start (hi)

USE BACKS OF PAGES TO DRAFT A SOLUTION

Use the specifications below to write a ***C*** program for the above system hardware.

- The vendor-provided exception handler will call a function *interrupt_handler()*.
- Write *interrupt_handler()* to do *full checking to identify the interrupt source.*
- Assume that macros NIOS2_WRITE_IENABLE(), NIOS2_WRITE_STATUS(), and NIOS2_READ_IPENDING() are available in the header file *nios2_control.h.*
- Bit 0 of ipending/ienable is for timer 0. Bit 1 of ipending/ienable is for timer 1.
- The 8-bit parallel *input* port is connected to an external analog-to-digital chip that *continuously* gives 8-bit data that is from a temperature measurement. For simplicity, the 0-255 range corresponds directly to Celsius temperature.
- The 8-bit parallel *output* port controls a heating element. For simplicity, only two output values are to be used: 0 for heater off and 128 for 50% heat.
- The control algorithm is as follows: if the temperature is below 98 deg. Celsius, turn on 50% heat; if the temperature is above 102 deg. Celsius, turn heat off.
- With a 50-MHz clock input, an interrupt interval of 1 sec is required for timer 0. On each timer 0 interrupt, have code *within the ISR* apply the above algorithm.
- An interrupt interval of 0.25 sec (12,500,000 cycles) is required for timer 1. On each timer 1 interrupt, notify the main program that 0.25 sec has elapsed.
- In the main program, the notification it receives every 0.25 sec should cause it to print a backspace character '`\b`' followed by a single character to reflect the current temperature: '`L`' if below 98 deg. Celsius, '`-`' if between 98 and 102, '`H`' if above 102 deg. Celsius. (Print a blank space *before* entering main loop.)
- For the JTAG UART, the upper 16 bits of the *status* register indicate the amount of space available in the output buffer. A character is sent to the output buffer by writing a *word* to the *data* register with the character in the low 8 bits. *Check for space in the JTAG UART output buffer before sending a character.*

- Structure the code such that **`interrupt_handler()`** calls a separate function for performing the needed work to respond to each detected interrupt source.

- *Required functions*:

```
void Init (void);/*hardware/software initialization*/
void PrintChar (unsigned int ch); /* print a char */
void HandleTimer0 (void); /* called ONLY by ISR */
void HandleTimer1 (void); /* called ONLY by ISR */
void interrupt_handler (void); /* the ISR */
int  main (void); /* main program with main loop */
```

- On the following pages, present the functions in the order listed above.