THE UNIVERSITY OF BRITISH COLUMBIA
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
EECE 379 : Design of Digital and Microcomputer Systems
2000/2001 Winter Session, Term 1

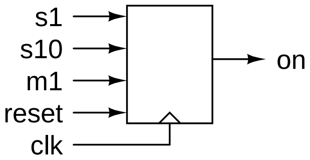
FINAL EXAMINATION
3:30 AM – 6:30 PM
December 11, 2000

This exam has five (5) questions on five (5) pages. The marks for each question are as indicated. There are a total of 44 marks. Answer all questions. Write all answers in the exam book provided. Show your work. You may answer the questions in any order. Books, notes and calculators are allowed. Show your work. You may keep this exam paper.

Question 1 (10 marks)

This question asks you to design a timer circuit that controls a microwave oven.

The controller has three push-button inputs that are used to increase the "remaining cooking time." `s1`, `s10`, and `m1` are high when the user is pushing a button that should increase the "remaining cooking time" by 1, 10 or 60 seconds respectively. Another push-button input resets the "remaining cooking time" to zero. The frequency of the clock input, `clk`, is 1 Hz. The controller has one output, `on`, which turns on the microwave power. The `on` output is high whenever the "remaining cooking time" is greater than zero.



The device has the following VHDL entity declaration:

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_arith.all ;

entity controller is
   port (
      s1, s10, m1, reset : in std_logic ;
      clk : in std_logic ;
      on : out std_logic ) ;
end controller ;
```

You design must satisfy *all* of the following requirements.

The "remaining cooking time" changes only on the rising edge of `clk` (it is a synchronous design).

The "remaining cooking time" can vary between 0 and 600 seconds (10 minutes) and should *never* be set to a value outside this range.

If only *one* of the s1, s10, or m1 buttons are being pushed, then the "remaining cooking time" should be incremented by the the amount indicated by the button (but never to more than 600).

If only the reset button is being pushed, then the "remaining cooking time" should be set to zero.

If no buttons are being pushed then the "remaining cooking time" should be decremented by 1 (but never to less than 0).

For other input conditions the "remaining cooking time" should not change.

Write an architecture that implements this device and is synthesizeable by MaxPlus+II. You may use only std_logic, std_logic_vector or unsigned types. Use type conversion functions as necessary. Any process in your VHDL code may only contain exactly one if statement controlling one or more simple signal assignment statements.

You need not include comments, library or use statements. You need not synchronize inputs or register outputs (for glitch removal). You may ignore the initial (uninitialized) state of the device.

Hints: The VHDL operators "+" and "<=" can be applied to certain combinations of unsigned *and* integer *values.* $2^9 = 512$. $2^{10} = 1024$.

Question 2 (10 marks)

Write a function, checkbuf:, in 8086 assembly language that validates an array of 200 unsigned 16-bit integers.

When your function is called, the register BX will contain the address (offset portion) of the buffer where the values are stored.

Your function should check each and every value in the buffer: if a value is less than 1024, then that value should be set to 1024; if a value is greater than 64512 then that value should be set to 64512.

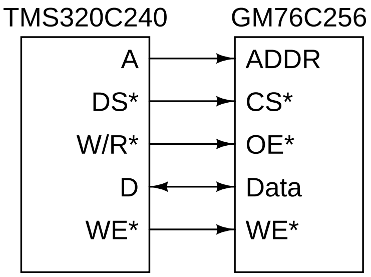
Your function must save any registers it modifies and restore them before returning with a RET instruction.

You must declare storage for any variables. You need not include comments or assembler directives such as segment, assume or org.

Hints: Write a C or pseudo-code solution before you start coding.

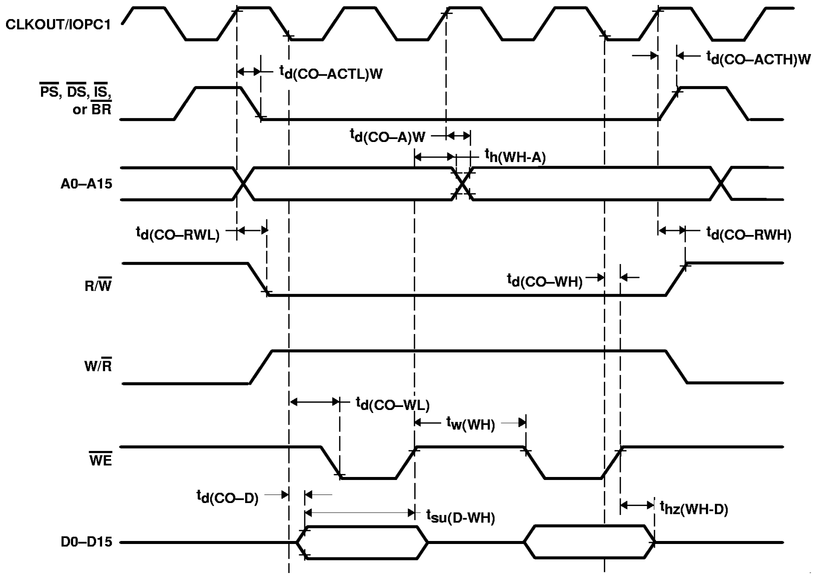
Question 3 (8 marks)

The diagram below shows part of the interface between a TI TMS320C240 microprocessor and a Hyundai GM76C256 SRAM:



The following figure shows the CPU write cycle timing diagram. This timing diagram shows two successive write cycles (to allow more space for indicating timing specifications). Timing specifications relative to the clock (—CLKOUT—) are the same on both cycles. You may assume the —CLKOUT— signal period is exactly 50ns, and that each half-cycle takes exactly 25 ns. Delay times $t_{d(\ldots)}$ may be assumed to have minimum values of zero (0).

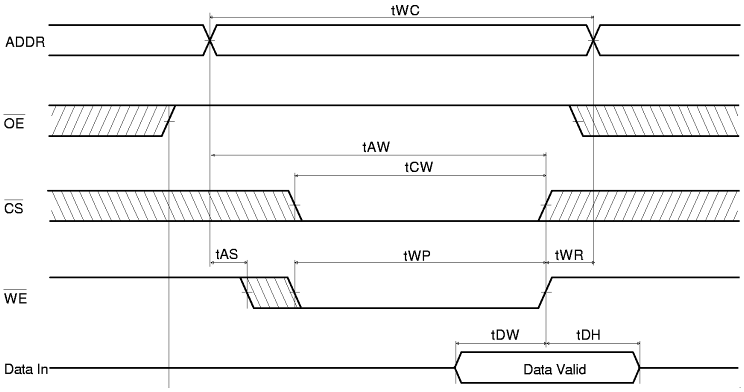
Note that the WE* pulse-low duration, $t_{W(WL)}$, is not labeled in the diagram, (it is the width of the low portion of the WE* signal).



The following table shows some of the CPU timing specifications. Assume H is 25ns and that $t_{h(WH\text{-}A)}$ is $H - 10$.

PARAMETER		MIN	MAX	UNIT
$t_{d(CO\text{-}A)W}$	Delay time, CLKOUT/IOPC1 high to address valid		17	ns
$t_{d(CO\text{-}D)}$	Delay time, CLKOUT/IOPC1 low to data bus driven		15	ns
$t_{h(WH\text{-}A)}$	Hold time, address valid after $\overline{\text{WE}}$ high	H – 8 H – 10‡		ns
$t_{w(WH)}$	Pulse duration, $\overline{\text{WE}}$ high	2H – 11		ns
$t_{w(WL)}$	Pulse duration, $\overline{\text{WE}}$ low	2H – 11		
$t_{d(CO\text{-}WL)}$	Delay time, CLKOUT/IOPC1 low to $\overline{\text{WE}}$ low		9	ns
$t_{d(CO\text{-}WH)}$	Delay time, CLKOUT/IOPC1 low to $\overline{\text{WE}}$ high		9	ns
$t_{su(D\text{-}WH)}$	Setup time, write data valid before $\overline{\text{WE}}$ high	2H – 8		ns
$t_{hz(WH\text{-}D)}$	High-impedance time, $\overline{\text{WE}}$ high to data bus Hi-Z	0	5	ns
$t_{d(CO\text{-}SL)W}$	Delay time, CLKOUT/IOPC1 low to $\overline{\text{STRB}}$ low		10	ns
$t_{d(CO\text{-}SH)W}$	Delay time, CLKOUT/IOPC1 low to $\overline{\text{STRB}}$ high		6	ns
$t_{d(CO\text{-}ACTL)W}$	Delay time, CLKOUT/IOPC1 high to $\overline{\text{PS}}$, $\overline{\text{DS}}$, $\overline{\text{IS}}$, and $\overline{\text{BR}}$ low		10	ns
$t_{d(CO\text{-}ACTH)W}$	Delay time, CLKOUT/IOPC1 high to $\overline{\text{PS}}$, $\overline{\text{DS}}$, $\overline{\text{IS}}$, and $\overline{\text{BR}}$ high		10	ns
$t_{d(CO\text{-}RWL)}$	Delay time, CLKOUT/IOPC1 high to R/$\overline{\text{W}}$ low		10	ns
$t_{d(CO\text{-}RWH)}$	Delay time, CLKOUT/IOPC1 high to R/$\overline{\text{W}}$ high		10	ns

The following diagram shows the SRAM write-cycle timing diagram. The write cycle happens during the overlap of CS* or WE*. For the TMS320C240 CPU, WE* goes active after and inactive before DS* so the write cycle begins and ends with WE*.



The table below shows five SRAM write-cycle timing requirements that need to be met. Obtain expressions for these requirements in terms of the symbols for the CPU timing specifications given above. Obtain values for these expressions in nanoseconds. Find the amount by which the requirement is exceeded (in nanoseconds) and state whether the requirement is met or not (Y[es] or N[o]). Give your answer in the form of a table as shown below. You may omit the first column. Write your answer in your exam book – *not on the exam paper*.

Requirement			Guaranteed		Margin	Met
Parameter	Symbol	Min. (ns)	Expression	Value	(ns)	(Y/N)
Write Cycle Requirements						
Write Pulse Width	t_{WP}	45				
Data to Write Time Overlap	t_{DW}	25				
Address Valid to End of Write	t_{AW}	50				
Write Recovery	t_{WR}	0				

Hint: If the required specification is not given, use two specifications relative to clock edges plus the time between the clock edges.

Question 4 (6 marks)

This question asks you to design a memory system for a CPU that has a 20-bit address bus, `A0` to `A19` and an 8-bit data bus (`D0` to `D7`). The memory system provides a total of 256 kBytes using 128k by 4 ROMs. Each memory chip has data out pins (`Q0`, `Q1`, . . .), address inputs (`A0`, `A1`, ...), and an active-low chip-select signal (`CS*`). The `CS*` pins are driven from an address decoding circuit with an appropriate number of outputs.

(a) Draw the schematic of the memory system showing the connections between the CPU, the decoder and the memory chip pins. The connections must *unambiguously* show which CPU signals connect to which memory signals.

Hint: Use a complete page for your diagram.

(b) Given the following VHDL entity declaration:

```
library ieee ;
use ieee.std_logic_1164.all ;
```

```
entity decoder is
   port (
      a : in std_logic_vector (19 downto 0) ;
      cs0, cs1, ... : out std_logic ) ;
end decoder ;
```

write an architecture that sets the value(s) of the *active-high* chip-select signals `CS0`, `CS1`, . . . so that the 256 kByte memory region starts at address 80000H and addresses are fully decoded.

Question 5 (10 marks)

(a) For each term in column “A” select the best matching entry in column “B”. Write your answers in numerical order and show the number and the selected letter unambiguously. There is a different best answer for each entry in column ‘A’. No marks will be subtracted for incorrect answers. For this part you need not explain your answer. Make sure your answer is unambiguous.

A		B	
1	race condition	A	flash
2	RS-232	B	structural design
3	non-volatile	C	ignores CPU interrupt enable bit
4	tri-state	D	controls IRQ priority
5	schematic	E	random behaviour
6	NMI	F	data bus
7	PIC	G	bus request/grant
8	DMA	H	serial

(b) The keyboard interrupt handler in a PC is located at absolute memory location 26104H. Write a table showing the address (memory location) and value of each byte in the keyboard interrupt vector. Show your work.