Name: _____

First Name: _____

Student ID: _____

1st session examination

Design of Digital Circuits SS2014

(252-0014-00S)

Srdjan Capkun, Frank K. Gürkaynak

Examination Rules:

1. Written exam, 90 minutes total.
2. No books, no calculators, no computers or communication devices. Six pages of hand-written notes are allowed.
3. Write all your answers on this document, space is reserved for your answers after each question. Blank pages are available at the end of the exam.
4. Put your Student ID card visible on the desk during the exam.
5. If you feel disturbed, immediately call an assistant.
6. Answers will only be evaluated if they are readable
7. Write with a black or blue pen (no pencil, no green or red color).
8. Show all your work. For some questions, you may get partial credit even if the end result is wrong due to a calculation mistake.

Question:	1	2	3	4	5	6	7	Total
Points:	5	9	10	10	15	14	12	75
Score:								

This page intentionally left blank

1. (a) (2 points) For the following four numbers given in hexadecimal notation, write the corresponding binary number.

   $0x42$: _____

   $0xF9$: _____

   $0x85$: _____

   $0x7E$: _____

   (b) (3 points) Sort the four numbers given above from the *smallest to largest* assuming that the binary numbers are coded using:

   Unsigned: _____

   Two's complement: _____

   Sign Magnitude: _____

2. In this question, you will be asked to derive the Boolean Equations for two 4-input logic functions. Please use the Truth Table below for this part.

Input				**Output**	
D_3	D_2	D_1	D_0	A	B
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

(a) (4 points) The output A is *one* when there are at least three consecutive 1's or 0's in the word D_3, D_2, D_1, D_0 (for example 1110), output is *zero* otherwise. Write the *simplified* Boolean equation for A using the *Product of Sums* form. (*Hint: use a Karnaugh map*)

(b) (4 points) The output B is *one* when there are *three or more zeroes* in the 4-bit input word D_3, D_2, D_1, D_0, output is *zero* otherwise (for example 0100). Write the *simplified* Boolean equation for B using the *Sum of Products* form. (*Hint: use a Karnaugh map*)

(c) (1 point) In general, how many different 4-input Boolean functions can be defined?

3. In this question you will be asked to design a Finite State Machine that implements a simple vending machine for a product that costs 3 CHF. The vending machine accepts only two types of coins 1 CHF and 2 CHF. There are two inputs to the state machine: the signal *coin* is 1 when a new coin is inserted and 0 when there is no new coin. The signal *value* is 1 when the machine has detected a 2 CHF coin and 0 when a 1 CHF coin has been detected.

   The state machine has two outputs. A logic-1 on the *product* signal activates the mechanism to release a product. The signal *return* will activate a system that will return the customer 1 CHF should he chose to enter two 2 CHF coins in a row.

   Simplifying assumptions

   - It is assumed that there is an infinite supply of 1 CHF coins available to return the money.
   - The system has been designed so that while *product* and/or *return* signals are active, no new coins will be accepted and the user will not be able to add more coins during this time. In other words, as soon as the user enters either 3 CHF or 4 CHF, the machine will no longer accept new coins until the product is delivered.

   (a) (4 points) Draw a State Transition Diagram using a Moore type FSM.
   (*Hint: Label the states "INIT" for the initial state and "ONE", "TWO", "THREE", and "FOUR" for the remaining states which correspond to the total amount of coins entered by the customer.*).

(b) (4 points) Draw State Transition Diagram using a Mealy type FSM. (*Hint: A simple solution would have only three states: "INIT", "ONE", and "TWO"*)

(c) (2 points) Complete the following extended State Transition Table with outputs of the Mealy type FSM. The first entry has been provided for you as an example.

Present State	**Inputs**		**Next State**	**Outputs**	
name	*coin*	*value*	name	*product*	*return*
`INIT`	0	X	`INIT`	0	0

This page intentionally left blank

4. You have been asked to design a single-port SRAM memory for a microcontroller with 1024 entries of 32-bit words each.

   (a) (2 points) Answer the following short questions about this memory:

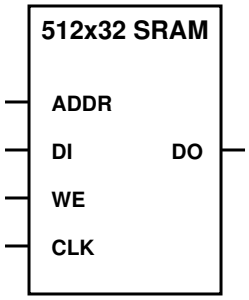
   How many kiloBytes can this memory store? _____

   How many bits will the address input have? _____

   How does the SRAM know when there is a write access? _____

   Is an SRAM a combinational or a sequential circuit? _____

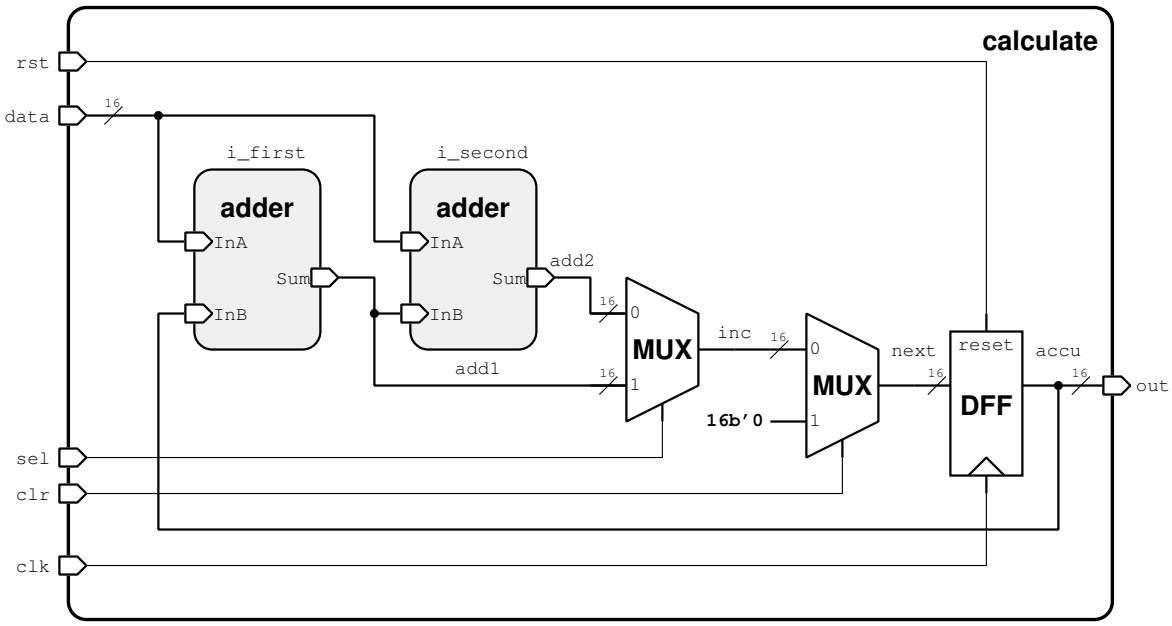
   (b) (8 points) To build your 1024-entry, 32-bit word memory, you only have access to a single-port SRAM memory with 512-entries of 32-bit words that has the following connections.



   Draw a schematic that shows how the 1024-entry memory can be designed by combining two 512-entry memories. You can use any combinational logic gates such as AND, OR, NOT gates and multiplexers as necessary. Make sure to label the address and data connections properly.
   (*Hint:Consider the read and write operations separately*)

This page intentionally left blank

5. (a) (8 points) In this question you are required to write the Verilog code that implements the following block diagram.



The block called *calculate* has one 16-bit input (*data*), two 1-bit control signals (*sel*, *clr*) as well as two signals for clock and reset. It is a sequential circuit that generates a single 16-bit output called *out*. The circuit contains two instances of a combinational adder block called (*adder*). The following is the declaration part of this module:

```
1 module adder ( input [15:0] inA , input [15:0] inB,
2                output [15:0] Sum);
3   // definition of the adder
4
5 endmodule
```

Notes:

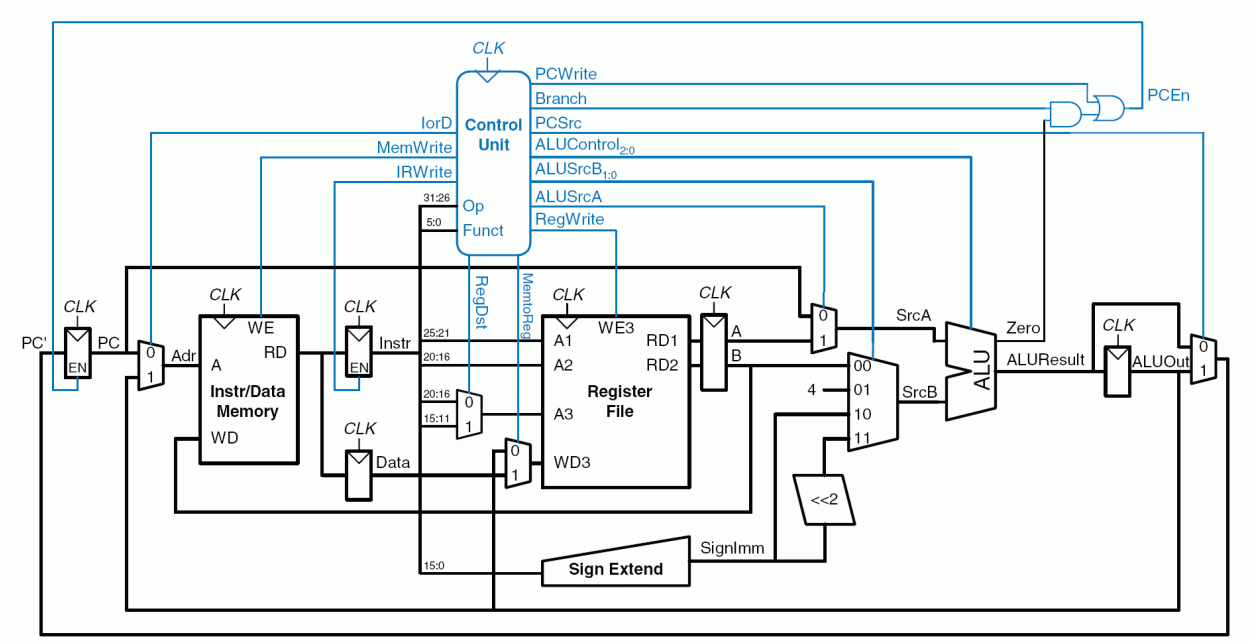
- The flip-flop and the multiplexers are not instantiated, you have to write the corresponding Verilog code.
- The flip-flop uses an asynchronous reset, the output is zero when reset is one.
- Note that Verilog is case sensitive.
- Write legibly.

```
module calculate (
  input clk, rst,
  input [15:0] data,
  input sel, clr,
  output [15:0] out);









// Output assignment
  assign out = accu;

endmodule
```

(b) (2 points) In one sentence or two, briefly explain what this circuit does.

(c) (5 points) Derive a more efficient version of the circuit that does exactly same thing using only one instance of the adder and without using additional clock cycles. Draw either a block diagram or write the corresponding Verilog code (it is sufficient to draw/write only the parts that will change).

6. In this question, you are required to answer questions about the Multicycle architecture of the MIPS processor and calculate different timing paths for the implementation given below.



Use the following table to calculate timing related questions. Assume that the control unit is not part of any timing path.

Block	Prop. Delay (pd)	Cont. Delay (cd)	Setup Time (su)	Hold Time (ho)
Register	0.3 ns	0.1 ns	0.1 ns	0.2 ns
Instr/DataMemory (A)	2.5 ns	2.2 ns	n.a.	n.a.
Instr/DataMemory (WD)	n.a.	n.a.	0.2 ns	0.4 ns
Register File (A1, A2)	2.3 ns	1.0 ns	n.a.	n.a.
Register File (A3, WD3)	n.a.	n.a.	0.3 ns	0.3 ns
Control Unit	0.0 ns	0.0 ns	0.0 ns	0.0 ns
ALU	2.1 ns	1.2 ns	n.a.	n.a.
2:1 Multiplexer	0.5 ns	0.3 ns	n.a.	n.a.
4:1 Multiplexer	0.6 ns	0.4 ns	n.a.	n.a.
Sign Extend	0.0 ns	0.0 ns	n.a.	n.a.
«2	0.0 ns	0.0 ns	n.a.	n.a.
AND/OR	0.1 ns	0.0 ns	n.a.	n.a.

Hint: $\frac{1}{1\,ns} = 1\,GHz$, *a clock with 1 GHz has a period of 1 ns. 1 GHz = 1000 MHz*

(a) (2 points) Name two advantages of a multi-cycle architecture when compared to a single-cycle architecture.

(b) (1 point) Name at least one disadvantage of a multi-cycle architecture when compared to a single-cycle architecture.

(c) (6 points) We will attempt to determine the critical path. In such a circuit, there are several candidates; we therefore need to determine the length of all these paths and determine the longest one. Identify the critical path starting with the registers that drive the following signals. The first one has been completed for you, you need to determine the remaining three.

$ALUout \rightarrow pd_{reg,ALUout} \rightarrow pd_{Mux2} \rightarrow pd_{Mem,A} \rightarrow su_{reg,Instr}$
PC
$Instr$
$Data$

(d) (2 points) What is the critical path of this architecture and how long is this path? (*Hint: It will be one of the paths you have determined in the previous question*)

(e) (3 points) What is the *shortest path* of this circuit? What condition does this path have to satisfy? Is this condition satisfied in this circuit? If not, what can be done?

7. You are analyzing a performance problem of a software running on a 5-stage pipelined MIPS architecture (the same one covered in class) that uses a direct mapped cache with a capacity of 256 words and a block size of 16 words.

   The assembly code of the problematic part (the subroutine *comparr*) is given below.

   Answer the questions based on this code.

```
1
2  comparr: addi $sp, $sp, -4
3           sw   $ra, 4($sp)
4
5           addi $s0, $0,  1024   # initialize values
6           addi $s1, $0,  0
7           addi $s2, $0,  0x6000
8  loop:    beq  $s0, $0,  done
9           add  $s3, $s2, $s0
10
11          lw   $a0, 0x1000($s3) # load Y[$s1]
12          jal  round            # round (Y[$s1])
13          add  $a1, $v0, $0
14
15          lw   $a0, 0x2000($s3) # load Z[$s1]
16          jal  round            # round (Z[$s1])
17          add  $a2, $v0, $0
18
19          lw   $a0, 0($s3)      # load X[$s1]
20          jal  round            # round(X[$s1])
21          add  $a0, $v0, $0
22
23          jal  compute
24          add  $s1, $s1, $v0
25          addi $s0, $s0, -4
26          j    loop
27
28 done:    add  $v0, $s1, $0
29          lw   $ra, 4($sp)
30          addi $sp, $sp, 4
31          jr   $ra
32
33 round:   addi  $v0, $a0, 16
34          sll   $v0, $v0, 5
35          sra   $v0, $v0, 5
36          jr    $ra
37
38 compute:                       # code that does not interest us.
```

(a) (4 points) What is a *data hazard* in a pipelined processor? Which part of the code would be affected by a *data hazard*? Briefly explain why.

(b) (1 point) Consider the *lw* instructions in the main *loop* (lines 8 to 26). How many *lw* instructions will be executed within this loop in total?

(c) (2 points) Approximately, what percentage of the *lw* instructions will result in a cache miss? Why?

(d) (5 points) Is there any way to reduce the cache miss rate *without modifying the hardware and the cache structure*? Explain.