UNIVERSITY OF TORONTO

Faculty of Arts and Science

April 2013 Examinations

CSC258H1S: Computer Organization

Duration: 3 hours

Permitted Aids: one ruler

Last Name: _____

First Name: _____

Student Number: _____

Instructor: Steve Engels

Instructions:

- **Write your name on every page of this exam.**
- **Do not open this exam until you hear the signal to start.**
- Have your student ID on your desk.
- No aids permitted other than writing tools. Keep all bags and notes far from your desk before the exam begins.
- There are 6 questions on 18 pages. When you hear the signal to start, make sure that your exam is complete before you begin.
- Read over the entire exam before starting.
- If you use any space for rough work or have to user the overflow page, clearly indicate the section(s) that you want marked.

Mark Breakdown

Part A:	**/ 22**
Part B:	**/ 18**
Part C:	**/ 12**
Part D:	**/ 58**
Part E:	**/ 12**
Part F:	**/ 30**
Total:	**/ 152**

Part A: Short Answer (22 marks)

Answer the following questions in the space provided. When providing a written answer, write **as clearly and legibly as possible**. Marks will not be awarded to unreadable answers.

1. How many bytes can be stored in a memory unit that uses 6 address bits and a 32-bit architecture? **(2 marks)**

a) 256 **b)** 64

c) 128 **d)** 192

2. How many address bits are needed to specify each instruction in a 1024 byte memory unit, given a 32-bit architecture? **(2 marks)**

a) 512 **b)** 8

c) 32 **d)** 9

3. What is the voltage value in a circuit that is commonly associated with "high voltage"? **(1 mark)**

4. True or False? A pn junction won't have a depletion layer unless a forward or reverse bias is applied to the junction. **(1 mark)**

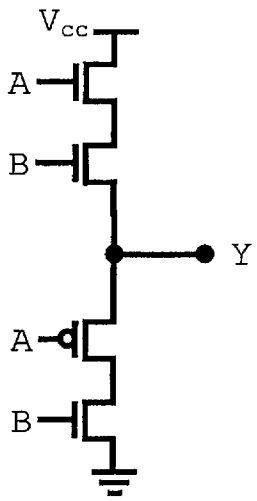
True **False**

5. True or False? When assigning flip-flop values to states, it's better to use more flip-flops in your design if it avoids having multiple flip-flops change at the same time. **(1 mark)**

True **False**

6. What are the two techniques that are used to measure the cost of a circuit layout? **(2 marks)**

7. What logic gate results from the transistor circuit on the right? **(2 marks)**



8. True or False? An accumulator circuit performs multiplication faster than a multiplication circuit made up of layers of adders. **(1 mark)**

True **False**

9. Assuming that `A` and `B` are one-bit wire values, which of the following Verilog statements will cause `B` to have a high value? Circle all that apply. **(2 marks)**

a) `xor(B, A, ~A);`

b) `and(B, A, ~A);`

c) `or(B, A, ~A);`

d) `nand(B, A, ~A);`

10. If a finite state machine has 60 states, what is the minimum number of flip-flops that you would need to implement it in a circuit? **(1 mark)**

11. What do the following abbreviations stand for? **(4 marks)**

PC: _____ **ALU:** _____

IR: _____ **MIPS:** _____

12. What special function do the `$ra`, `$sp` and `$zero` registers serve in the processor? **(3 marks)**

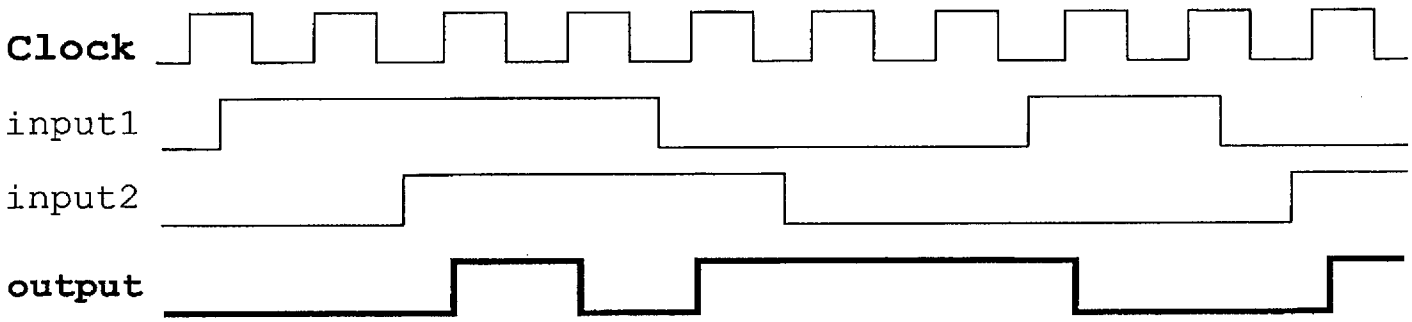
$ra : _____

$sp: _____

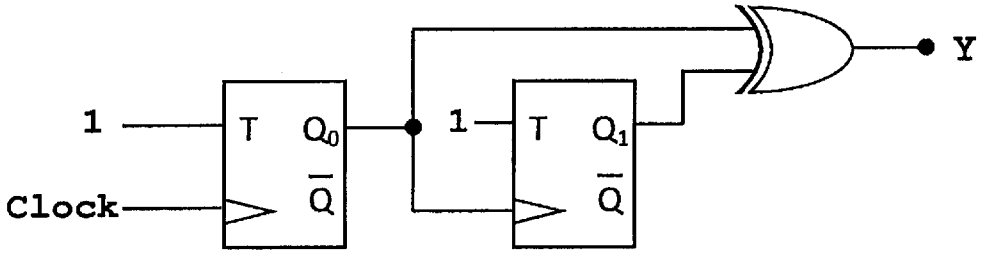
$zero: _____

Part B: Design and Analysis (18 marks)

1. In the space below, draw a circuit whose behaviour matches the following waveform. **(4 marks)**



4. Given the following circuit, show what the output value of Q_0, Q_1 and Y will be in the waveform diagram. Assume that Q_0 and Q_1 start with initial values of zero. **(6 marks)**

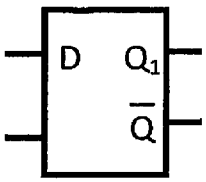


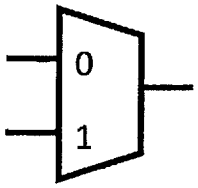
Q_0

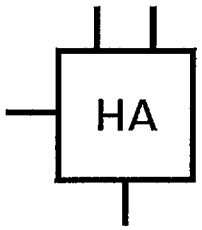
Q_1

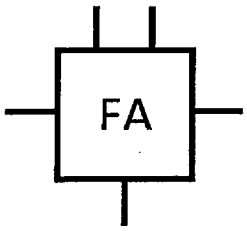
Y

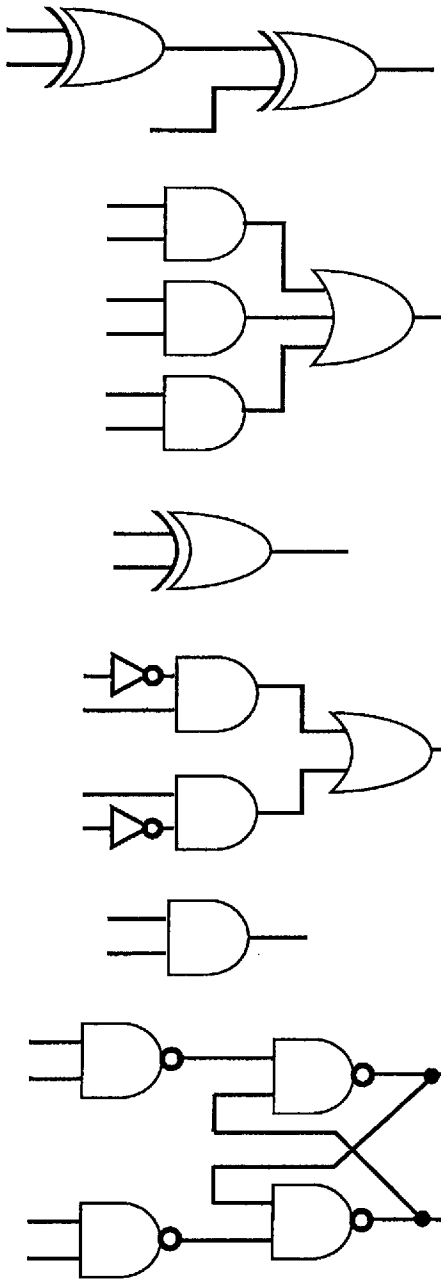
3. Match the logic devices on the left to the circuits on the right that implement them. Draw lines to indicate connections between devices and circuits. If you feel that a circuit is missing, indicate what the missing circuit would be by also drawing and connecting it in the space below. **(6 marks)**











2. In one sentence, what is the difference between a D latch and a D flip-flop? **(2 marks)**

Part C: Sequential Circuit Design (12 marks)

1. A sequential circuit has two flip-flops, whose values are set according to the K-map below. Circle the largest minterm groupings possible, and write the equations that result. **(4 marks)**

F_1:

	$\overline{F_1} \cdot \overline{F_0}$	$\overline{F_1} \cdot F_0$	$F_1 \cdot F_0$	$F_1 \cdot \overline{F_0}$
$\overline{X}$	0	0	1	1
X	0	1	1	0

F_0:

	$\overline{F_1} \cdot \overline{F_0}$	$\overline{F_1} \cdot F_0$	$F_1 \cdot F_0$	$F_1 \cdot \overline{F_0}$
$\overline{X}$	0	1	1	0
X	0	0	1	1

2. Given the K-map groupings from the previous part, draw the circuit diagram that implements this behaviour in the space below. **(4 marks)**

3. Given the K-maps from the previous page, what would the finite state machine diagram have to look like to generate those K-maps? Draw your answer in the space below. **(4 marks)**

Part D: Processors (58 marks)

1. In the space below, perform Booth's Algorithm on the binary values A=0110 and B=0101. Show your steps in the space provided. **(6 marks)**

P =

Step 1:

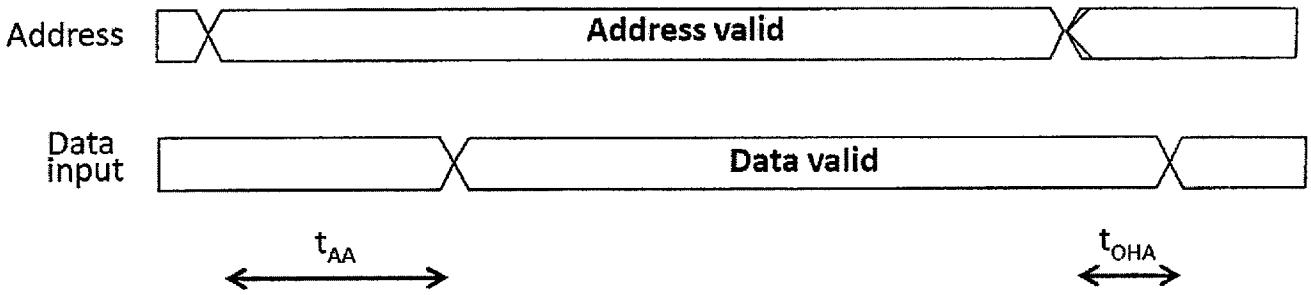
Step 2:

Step 3:

Step 4:

P =

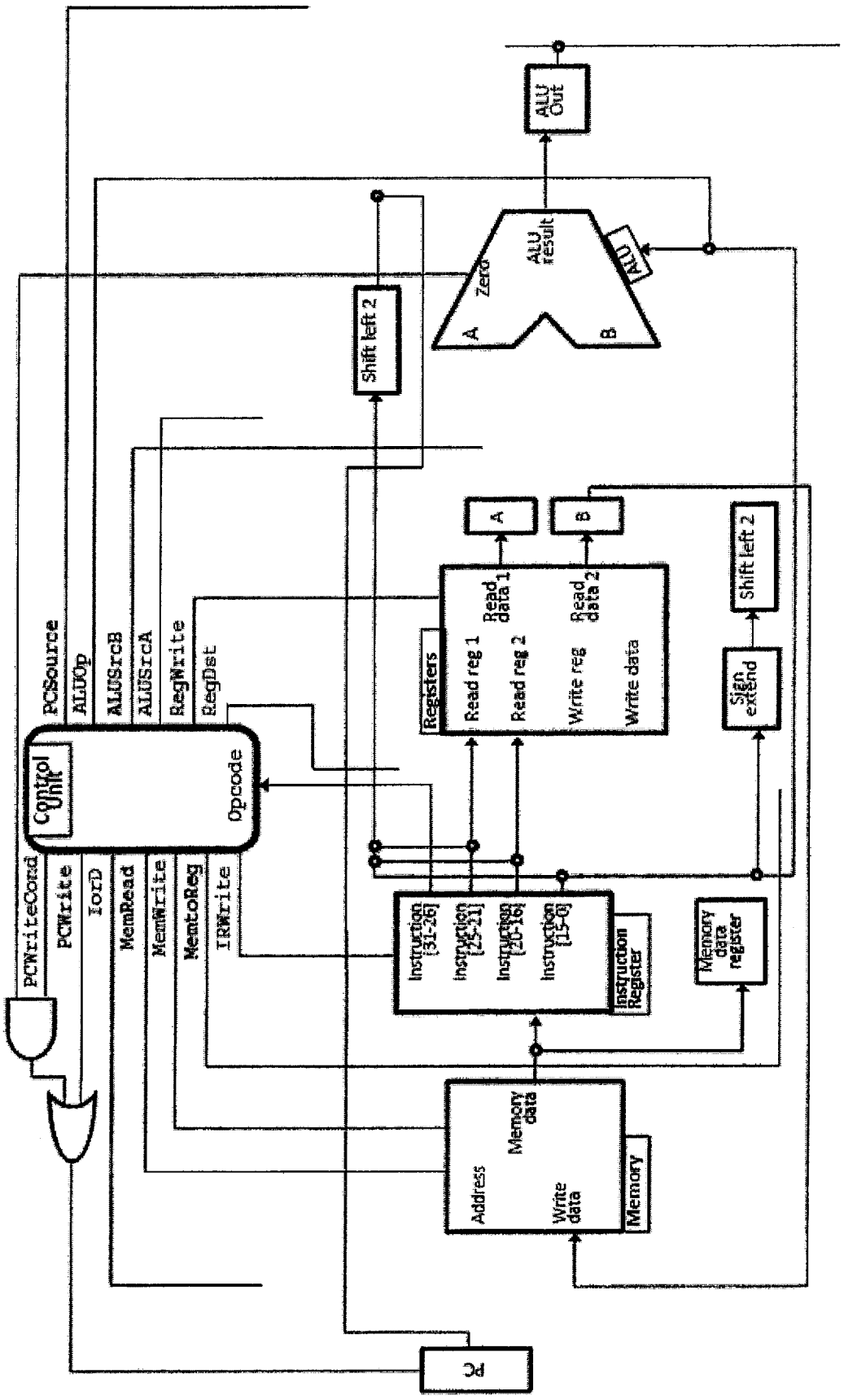
2. For the following read signal diagram, describe what each labeled time segment is called, and the purpose of each segment during a memory read operation. **(4 marks)**



t_{AA}: _____

t_{OHA}: _____

3. The datapath below is missing its muxes! Draw the missing muxes in the correct locations, and make sure that they're connected to the right input, output and select lines. **(20 marks)**

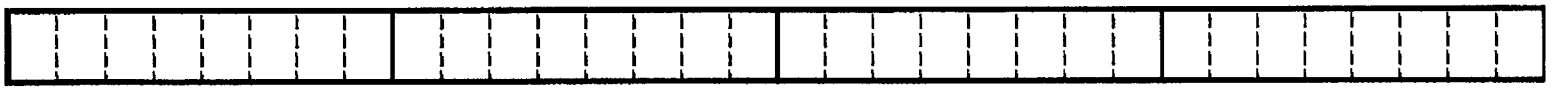


5. For the following assembly language instructions, write the equivalent machine code instruction in the space provided. You might find the reference information in the appendix helpful for this question. Fill in the space with an X if the value doesn't matter. **(10 marks)**

a) `nor $v1, $t0, $t1`

b) `bne $a0, $a1, end` (where "end" is 6 instructions ahead)

c) `sb $s0, 255($t8)`



6. For the following machine code instructions, provide the equivalent assembly language instruction in the space provided. **(6 marks)**

a) 00111001000010010000000011111111

b) 00000011111000110011101000001000

c) 10001110000010000000000000001100

7. For each of the processor tasks below, indicate what the values of the following control unit signals will be by filling in the boxes next to each signal with the signal values. **(12 marks)**

- If a control signal doesn't affect the operation, fill in its value with an X.
- For ALUOp, if you don't know the values, just write what kind of operation is taking place.

Store the instruction whose address is in $a0 into the instruction register.

PCWrite ☐ PCWriteCond ☐ IorD ☐ MemRead ☐ MemWrite ☐
MemToReg ☐ IRWrite ☐ PCSource ☐ ALUOp ☐
ALUSrcA ☐ ALUSrcB ☐ RegWrite ☐

Add 24 to the program counter and write the result to $s0.

PCWrite ☐ PCWriteCond ☐ IorD ☐ MemRead ☐ MemWrite ☐
MemToReg ☐ IRWrite ☐ PCSource ☐ ALUOp ☐
ALUSrcA ☐ ALUSrcB ☐ RegWrite ☐

Branch to the new PC value from the previous part if $t0 is equal to $t1.

PCWrite ☐ PCWriteCond ☐ IorD ☐ MemRead ☐ MemWrite ☐
MemToReg ☐ IRWrite ☐ PCSource ☐ ALUOp ☐
ALUSrcA ☐ ALUSrcB ☐ RegWrite ☐

Part E: Verilog (12 marks)

Consider the piece of Verilog code on the right.

1. In one sentence, describe what function this code performs. **(4 marks)**

2. Given your answer to part 1, what functions do input signals `e` and `w` perform? **(2 marks)**

```
module X (clk, e, w, a, di, do);
  input   clk, w, en;
  input  [7:0] a;
  input  [3:0] di;
  output [3:0] do;

  reg    [3:0] M [255:0];
  reg    [3:0] do;

  always @(posedge clk)
  begin
    if (e) begin
       if (w)
          M[a] <= di;
       else
          do <= M[a];
    end
  end
endmodule
```

3. In the space below, write a Verilog module called `multiplier` with input signals called `A` and `B` (four bits each), and `clock` and `reset` (one bit each). The output signal is called `product`. **(2 marks)**

- Make the `product` output be the result of `A` x `B` when the clock goes high **(2 marks)**
- Also, implement an asynchronous reset that is negative-edge triggered. **(2 marks)**

Part F: Assembly Language (30 marks)

1. In the spaces provided below, write the assembly language instruction(s) that perform the following tasks. Full marks will be given for one-instruction answers. **(12 marks total)**

a) Perform a right arithmetic shift on the value in `$t0`. The number of bits to shift `$t0` by is stored in `$s0`. The result will be stored back into `$t0`. **(3 marks)**

b) Invert all the bits of the binary number stored in `$t2`. **(3 marks)**

c) Assuming that `$s0` stores the address of an integer in memory, set the lower two bytes of the integer at that address to zero. **(3 marks)**

d) If the value in `$a0` is greater than zero, loop infinitely. **(3 marks)**

Consider the assembly language program in the box below.

```
            .data
list:       .word     8, 1, -5, 9, 2, -4, 6, 8, 7, -2
            .text
main:       addi $s0, $zero, list     #
            addi $t2, $zero, 9        #
            lw $t0, 0($s0)            #
top:        lw $t1, 4($s0)            #
            sw $t1, 0($s0)            #
            addi $s0, $s0, 4          #
            addi $t2, $t2, -1         #
            bne $t2, $zero, top       #
end:        jr $ra                    #
```

2. For each line in the code, provide a short but insightful comment to describe it. **(5 marks)**

3. In the space below, provide a one-sentence description of the overall task this code is trying to perform. **(2 marks)**

4. This program is missing a key instruction. In the space below, state what that instruction would be, and where it would go. **(3 marks)**

5. In the space below, write a short assembly language program that is a translation of the program on the right. **(8 marks)**

- You can assume that `i` and `j` have been placed on the top of the stack, and that neither of them has a value of zero.
- The values of `i` and `j` should be replaced on the top of the stack by the return value before returning to the calling program.
- Make sure that you comment your code so that we understand what you're doing.

```
int remainder (int i, int j) {
   if (i < j)
      return j % i;
   else
      return i % j;
}
```

Reference Information

ALU arithmetic input table:

Select		Input	Operation	
S_1	S_0	Y	C_{in}=0	C_{in}=1
0	0	All 0s	G=A	G=A+1
0	1	B	G=A+B	G=A+B+1
1	0	B	G=A-B-1	G=A-B
1	1	All 1s	G=A-1	G=A

Register assignments:

Register values : Processor role

- Register 0 ($zero): reserved value.
- Register 1 ($at): reserved for the assembler.
- Registers 2-3 ($v0, $v1): return values
- Registers 4-7 ($a0-$a3): function arguments
- Registers 8-15, 24-25 ($t0-$t9): temporaries
- Registers 16-23 ($s0-$s7): saved temporaries
- Registers 28-31 ($gp, $sp, $fp, $ra)

Instruction table:

Instruction	Type	Op/Func	Syntax
add	R	100000	$d, $s, $t
addu	R	100001	$d, $s, $t
addi	I	001000	$t, $s, i
addiu	I	001001	$t, $s, i
div	R	011010	$s, $t
divu	R	011011	$s, $t
mult	R	011000	$s, $t
multu	R	011001	$s, $t
sub	R	100010	$d, $s, $t
subu	R	100011	$d, $s, $t
and	R	100100	$d, $s, $t
andi	I	001100	$t, $s, i
nor	R	100111	$d, $s, $t
or	R	100101	$d, $s, $t
ori	I	001101	$t, $s, i
xor	R	100110	$d, $s, $t
xori	I	001110	$t, $s, i
sll	R	000000	$d, $t, a
sllv	R	000100	$d, $t, $s
sra	R	000011	$d, $t, a
srav	R	000111	$d, $t, $s
srl	R	000010	$d, $t, a
srlv	R	000110	$d, $t, $s
beq	I	000100	$s, $t, label
bgtz	I	000111	$s, label
blez	I	000110	$s, label
bne	I	000101	$s, $t, label
j	J	000010	label
jal	J	000011	label
jalr	R	001001	$s
jr	R	001000	$s
lb	I	100000	$t, i ($s)
lbu	I	100100	$t, i ($s)
lh	I	100001	$t, i ($s)
lhu	I	100101	$t, i ($s)
lw	I	100011	$t, i ($s)
sb	I	101000	$t, i ($s)
sh	I	101001	$t, i ($s)
sw	I	101011	$t, i ($s)
trap	R	001100	i
mflo	R	010010	$d

This page is left blank intentionally for answer overflows.

This page is left blank intentionally for answer overflows.

Total Marks = 155

Total Pages = 18

End of exam