Spring 2020 Dated: Jan 21, 2020

CS G524: Advanced Computer Architecture
Assignment-1

Due Date: Jan 31, 2020

Birla Institute of Technology and Science (BITS) Pilani,
K K Birla Goa Campus
Second Semester 2019 – 2020

Instructions [10 Marks]. Students are advised to write their program with proper care. A program must have a header block consisting of programmer's name, roll no and date of creation [1.5 marks]. The objective of the program has to be written in the header block, also [1.5 marks]. The program should be properly indented [3 marks] and it is expected meaningful variable name [4 marks]. Use lowercase for variables and methods. If name consists of several words:
Make them into one

- First word lowercase
- Capitalize the first letter of each subsequent word
- Ex: radius, area and showMessageDialog

Capitalize every first letter in a constant, and use underscores between words

- Ex: PI and MAX_VALUE

Submission. Upload your assignment on Moodle in the link provided on the course homepage. Assignment should be a zip folder with six assignment " .v " extension files and one Word file which contains the descriptive solution expected for all 6 assignments. Zip file name must be your group number. E.g. group1.zip
Note. Member of each group must highlight their contribution (using comments)

Problem 1. Design a 3:8 decoder using Verilog and test the design with all possible inputs. Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, for a given input, which indicates a good approximation of the power consumption.

(Marks: 5 + 5)

Problem 2. Design a 32-bit 3x8 Multiplexer circuit using Verilog and test the design with all possible select inputs. Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, for a given input, which indicates a good approximation of the power consumption.

(Marks: 5 + 5)

Problem 3. Design a 32-bit register using the D-flipflop (f/f) with following specifications:

- Create a module of D-f/f
  - D-f/f must have a clock and a D input
  - It must have a Q and a *not*(Q) signal as a output
- Use the D-f/f module for design the 32-bit register

- The register must have a load enable (LE) input, an output enable (OE) input and data input

LE	CLK	Operation
High	High	Input data must be loaded into the register
High	Low	No operation
Low	High	No operation
Low	Low	No operation

OE	Operation
High	Content of the register must be shown in output terminal
Low	No operation

- Test the designed register with the four inputs cases (D, LE, OE, CLK):

  1) 1010 1100 1010 0110 1010 1100 1010 0110, 1, 0, 1
  2) 1010 1100 1010 0110 1010 1100 1010 0110, 0, 1, 1
  3) 1010 1110 1010 0111 1010 1110 1010 0111, 1, 0, 1
  4) 1010 1110 1010 0111 1010 1110 1010 0111, 0, 1, 1
  5) 1110 1110 1010 0110 1110 1110 1010 0110, 1, 0, 1
  6) 1110 1110 1010 0110 1110 1110 1010 0110, 0, 1, 1
  7) 1110 1110 1010 0110 1110 1110 1010 0110, 1, 0, 1
- Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, for a given input, which indicates a good approximation of the power consumption.

(Marks: 5 + 5 + 5)

Problem 4. Design a register file which contains the 8 number of registers, with each of width of 32 bits. The circuit must have load enable (LE), output enable (OE) inputs and data input. It also contains an output terminal. A register has to be identified, uniquely, for its operations: write/load or read.

LE	Operation
High	Input data must be loaded in the specific register
Low	No operation

OE	Operation
High	Specified register's content must be shown in output terminal
Low	No operation

Test Cases: Register names R0 to R7

Format for test case: (Register, data, LE, OE, CLK)

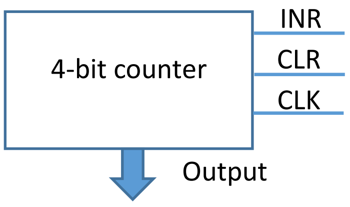
T1: (R0, 1010 1100 1010 0110 1010 1100 1010 0110, 1, 0, 1)
T2: (R0, 1010 1100 1010 0110 1010 1100 1010 0110, 0, 1, 1)
T3: (R1, 1010 1100 1010 0110 1010 1100 1111 0110, 1, 0, 1)
T4: (R1, 1010 1100 1010 0110 1010 1100 1111 0110, 0, 1, 1)
T5: (R2, 1010 1100 1010 0110 1010 1101 1011 0110, 1, 0, 1)
T6: (R2, 1010 1100 1010 0110 1010 1101 1011 0110, 0, 1, 1)
T7: (R3, 1010 1100 1010 0110 1011 1101 1011 0110, 1, 0, 1)
T8: (R3, 1010 1100 1010 0110 1011 1101 1011 0110, 0, 1, 1)
T9: (R4, 1011 1100 1010 0110 1011 1101 1011 0110, 1, 0, 1)
T10: (R4, 1011 1100 1010 0110 1011 1101 1011 0110, 0, 1, 1)
T11: (R5, 1010 1100 1010 0110 1010 1100 1010 0110, 1, 0, 1)
T12: (R5, 1010 1100 1010 0110 1010 1100 1010 0110, 0, 1, 1)
T13: (R6, 1010 1100 1010 0110 1010 1100 1111 0110, 1, 0, 1)
T14: (R6, 1010 1100 1010 0110 1010 1100 1111 0110, 0, 1, 1)

T15: (R7, 1010 1100 1010 0110 1010 1101 1011 0110, 1, 0, 1)
T16: (R7, 1010 1100 1010 0110 1010 1101 1011 0110, 0, 1, 1)

Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, for a given input, which indicates a good approximation of the power consumption.

(Marks: 5 + 5 + 5)

Problem 5. Design a 4-bit counter using Verilog and the test the design properly.



CLR	INR	CLK	Output
High	High	High	0000
High	High	Low	Previous value
High	Low	High	0000
Low	High	High	Increment the counter value by one
Low	Low	High	No change in the output
Low	Low	Low	No change in the output

Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, when CLK and INR are high, which indicates a good approximation of the power consumption.

(Marks: 5 + 5)

Problem 6. Design a 32-bit ALU. The ALU contains two input registers: Accumulator and Data. The ALU can perform these operations: Addition of two numbers, Compliment the content of accumulator and Logical AND of two inputs. Select the operation properly. Test the designed ALU properly. Count the number of 1-input, 2-inputs, …, n-inputs gate used for the design, which indicates a good approximation of the area requirement. Count the number of transitions (0→1 or 1→0) for each component's output of the design, for the complement operation with data, 1010 1100 1010 0110 1010 1101 1011 0110 and AND operation with data, 1010 1100 1010 0110 1010 1101 1011 0110, 1010 1100 1010 0110 1010 1100 1010 0110, which indicates a good approximation of the power consumption.

(Marks: 2 + 5 +2)