|  |  |  |  |
| --- | --- | --- | --- |
| Instructor |  | Due Date |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Part | **1** | **2** | **3** | **4** | Total |
| *Maximum Points* | **25** points | **25** points | **25** points | **25** points | **100**G101010 pointsG |
| ***Your Score*** |  |  |  |  |  |

**Textbook Reading Assignment**

Thoroughly read Chapter(s) 3 in your Computer Architecture and Organization textbook.

**Part 1 Glossary Terms - Boolean Algebra and Digital Logic**

Define, in detail, each of these glossary terms from the realm of computer architecture and computer topics, in general. If applicable, use examples to support your definitions. Consult your notes

or course textbook(s) as references or the Internet by visiting Web sites such as:

[**http://www.bing.com**](http://www.bing.com) or [**http://www.webopedia.com**](http://www.webopedia.com/)

**(a) Absorption Laws**

|  |
| --- |
| The absorption law is one of the basic identities of Boolean algebra and states that x(x+y)= x in the AND form and x+xy = x in its OR form. This is used when simplifying Boolean functions. |

**(b) Boolean Identity**

|  |
| --- |
| Boolean identities are laws of Boolean algebra that can be used to simplify Boolean functions when applied. These laws are essential when building and simplifying Boolean functions, which translate into circuit design and physical as well as digital performance and resource management. |

**(c) Combinational Circuits**

|  |
| --- |
| A combinatorial circuit is a circuit of combined digital logic chips whose output is determined by its current input. Combinatorial circuits can be used within sequential circuits, whose output is determined by its current input as well as its past inputs (memory). |

**(d) Digital Logic**

|  |
| --- |
| Digital logic is an implementation of Boolean algebra within a digital circuit which uses electric signal or pulses (as with a clock) as inputs with the presence of current within a threshold as an “on” value and the absence of current as an “off” value in the binary domain of Boolean algebra. Digital circuits are combined into combinational circuits, which are used within sequential circuits whose output is determined by current input as well as past inputs (memory). |

**(e) Finite State Machines**

|  |
| --- |
| Finite state machines (FSMs) describe computational systems (the control logic within a digital system). FSMs are used for designing computer circuitry. One example is the Deterministic Finite Automata FSM, which is a function that takes in a set of all possible states, a beginning state, and input alphabet or set of events the machine will recognize, and which contains a mapping function that maps a possible state to another or the same state in the set of all possible states, as well as a final set of all legal states. |

**Part 2 Exercises - Boolean Algebra and Digital Logic**

For each of the following, select the correct answer.

**(1)** If input *x* is true and if input *y* is true, then *x* · *y* is a \_\_\_\_\_\_\_\_\_\_ .

**(a) true statement ( 1 )** (b) false statement ( 0 )

**(2)** If *x* has a setting of 1 and *y* has a setting of 0 , then which of these will be false ( 0 ) ?

**(a) *y '* *'***  (b) *y '*  (c) *x '* + *y* (d) *x* ·  *y '* (e) *x* +  *y*

**(3)** If *x* is a false statement and *y* is a true statement, then which of these will be true?

(a) *y '* (b) *x* · *y '* (c) *x* + *y '* (d) *x* ·  *y* **(e) *x* +  *y***

**(4)** If *x* and *y* are true and *z* is false, find the truth value of: ( *z '* + *y* ) · *x*

**(a) true statement** (b) false statement

**(5)** If *x* is true and *y* and *z* are false, find the truth value of: ( *x '* · *z* ) + ( *z* · *y '* )

(a) true statement **(b) false statement**

**Part 3 Exercises - Boolean Algebra and Digital Logic**

Write a complete answer for each of these.

**(1)** Consider the truth table shown below. ( T represents true and F represents false )

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| *x* | *y* | *x '* | *x '* · *y* |
|  |  |  |  |
| T | T | F | F |
| T | F | F | F |
| F | T | T | T |
| F | F | T | F |

What are the missing entries in the rightmost column?

(a) T (b) F (c) F (d) F **(e)** **F**

F F F T **F**

F T F F **T**

T T T T **F**

**(2)** Consider the truth table shown below. ( 1 represents true and 0 represents false )

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| *x* | *y* | *x '* | *y '* | *x '* + *y '* | *x '*  + ( *x '* + *y '* ) |
|  |  |  |  |  |  |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 |

What are the missing entries in the rightmost column?

(a) 1 **(b)** **0** (c) 0 (d) 0 (e) 0

0 **1** 1 1 0

0 **1** 1 0 1

1 **1** 0 1 0

**(3)** The Boolean \_\_\_\_\_ function can be implemented with two switches, A and B . If a power lead is connected to switch A , and a wire connects switches A and B , then both A and B must be " on " in order for the output of the circuit to conduct electricity and provide power.

(a) NOT **(b) AND** (c) OR (d) NOR (e) NAND

**(4)** The Boolean \_\_\_\_\_ gate can be constructed from two switches, arranged so that if either switch is " on ", the output will also be " on ". Moreover, the output will still be on even if both switches are " on ".

(a) NOT (b) AND **(c) OR**  (d) NOR (e) NAND

**(5)** An XOR ( exclusive OR ) logic statement is false when both of its logic conditions are true. Note: an XOR statement can be simulated as: ( *x* · *y* )*'* · ( *x* + *y* )

**(a) True** (b) False

**(6)** A NAND ( i.e. NOT AND ) logic statement is true when one of its logic conditions is false.

**(a) True** (b) False

**Part 4 Exercises - Boolean Algebra and Digital Logic**

Write a complete answer for each of these.

**(1)** Using a Web site such as [**http://www.wolframalpha.com**](http://www.wolframalpha.com) construct a truth table for the logical expression given below. As an example of using WolframAlpha to construct a truth table, try this sample command:

truth table (NOT( x ) AND y) OR (x OR NOT( y ))

*x* ( *y '*  + *z* ) + *x y z*

|  |  |  |  |
| --- | --- | --- | --- |
| **x** | **y** | **z** | ***x* ( *y '*  + *z* ) + *x y z*** |
| T | T | T | T |
| T | T | F | F |
| T | F | T | T |
| T | F | F | T |
| F | T | T | F |
| F | T | F | F |
| F | F | T | F |
| F | F | F | F |

**(2)** Using Boolean identities, show that the following is a valid logical equivalence.

*x* = *x* *y* + *x* *y '*

x = (x + x)(y + y’) *Distributive*

x = x(y + y’) *Idempotent*  x = x(1) *Inverse* x = x *Identity*

**(3)** If *x* is set to 1 and *y* and *z* are both set to 0 , determine the result or truth value of this Boolean function.

*F* ( *x* , *y* , *z* ) = *x* *y ' z '*  + *y z* + *x y z*

***THE RESULT WOULD BE TRUE***

**(4)** According to De Morgan’s Principles, ( *x* · *y* )*'* is logically equivalent to *x '* + *y '* . Therefore, the logic expression *x* · *y '* is equivalent to which of these?

(a) *x '* · *y '*  (b) *x* + *y '* (c) *x '* + *y* (d) ( *x* · *y* ) **(e) ( *x '* + *y* ) *'***

**(5)** According to De Morgan’s Principles, ( *x* + *y* ) *'* is logically equivalent to *x '* · *y '* . Therefore, the logic expression ( *x '* · *y* ) *'* is equivalent to which of these?

(a) *x '* + *y '*  **(b) *x* · *y '*** (c) *x '* · *y* (d) *x* + *y '* (e) ( *x '* · *y* )