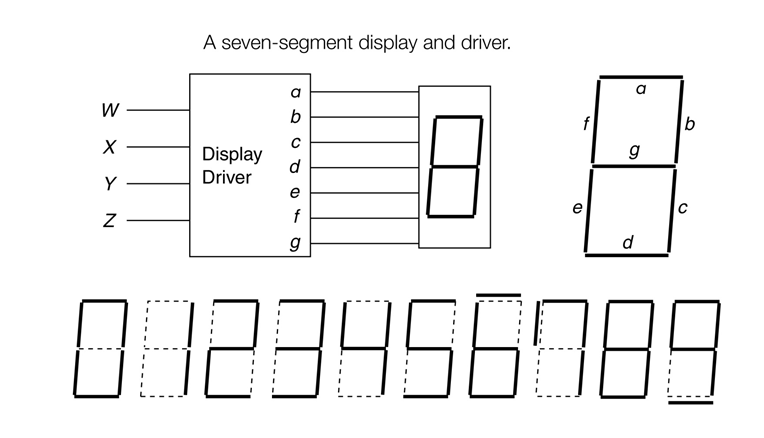
**Digital Logic – Project**

**LOGISM: Install LOGISM http://www.cburch.com/logisim/docs/2.1.0/guide/tutorial/index.html**

**Task 1: Design a seven-segment digit display circuit**

****

**Draw the Truth Table, Karnaugh map, and find a minimal function for each output while maximum sharing.**

**Implement the circuit in LOGISM (see the slides ch6[Adders&ALU]).**

**Task 2: Design a 4-bit ALU**

**Build a 1-bit ALU in LOGISM (or any other simulator).    You do not have to include all of the control logic inside the 1-bit ALU. Your 1-bit ALU must meet the following specifications:**

**Control Signals:**

* **ALUOP[1:0] -- encoded signal that specifies the function to be performed:**
* **Note: The first two bits of the instructions are allocated for other purpose.**

|  |  |
| --- | --- |
| ALUOP[1:0] | Operation |
| 00 | And |
| 01 | Or |
| 10 | Sub |
| 11 | Add |

**Operands:**

* **CIN -- carry in**
* **A -- first operand**
* **B -- second operand**

**OUTPUTS:**

* **S -- result of the operation**
* **COUT -- carry out**

**Major functions:**

* **The ALU must display the result of (A op B) on the output S; op represents the operation specified by the control signals.**
* **The ALU must use two's complement binary math for addition and subtraction.**
* **The ALU must display the correct carry bit on COUT for addition and subtraction; the carry bit is not defined for the other operations.**

**Simulate your 1-bit ALU thoroughly, and package it.**

**Design a 4-bit ALU using the 1-bit ALU block . Your 4-bit ALU must meet the following specifications:**

**INPUTS:**

* **A[3:0] -- first operand**
* **B[3:0] -- second operand**
* **ALUOP[1:0] -- encoded control signals specifying the ALU operation to be performed; coding the same as given in the 1-bit ALU design.**

**OUTPUTS:**

* **Result[3:0] -- result of operation**
* **Overflow**

**FUNCTION REQUIREMENTS:**

* **The ALU must perform the four math/logic functions listed above.**
* **The ALU must display the result of (A[3:0] op B[3:0]) on the output Result[3:0]; op represents operation specified by ALUOP[1:0].**
* **Overflow must be asserted if the result cause overflow.**