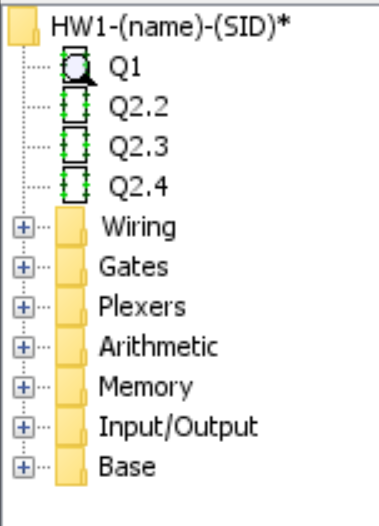
**HOMEWORK 2**

**Instruction:**

* The solutions should be submitted to Canvas. Email submissions (or other form of submissions) will not be accepted.
* Your submission should contain one **“circ” file** (the **Logisim** file format) and one **WORD/PDF** document.
  + The WORD or PDF document contains the answer of Question 0 and Question 2-(1). This PDF document should be named as “**HW2-(*name*)-(*SID*).pdf”**, where *name* is replaced by your own name and SID is replaced by your own SID.
  + The “circ” file should contains the answers of Question 1, Question 2-(2), Question 2-(3), Question 2-(4). This file should be named as “**HW2-(*name*)-(*SID*).circ”.** In Logisim, the circuit structure should look like the following picture.



*Hint: you can use “Menu->Project->Add Circuit” to add multiple circuits in one “circ” file.*

**Question 0:**

**Your SID:**

**Logisim version and OS:**

*(e.g., logisim-generic-2.7.1 on Windows)*

**Question 1 (20%):**

Build a D flip-flop in Logisim. Use the basic gates (e.g., AND, OR, NOR, NAND, XOR, NOT), input/output pins, and a clock to build your circuit. You can reuse the circuit components built by yourself, but should not use the built-in flip-flop components provided by Logisim.

**Question 2 (80%):**

A student’s SID has eight places (place 0 to place 7). For example, if a student’s SID is 45738429, the number at place 0 is 4, the number at place 1 is 5, …, the number at place 7 is 9.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| place | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
| number | **4** | **5** | **7** | **3** | **8** | **4** | **2** | **9** |

Correspondingly, we can define a function f(x) which maps the place to number. With the above example, we have:

f(0) = 4, f(1) = 5, …, f(7) = 9.

Answer the following questions:

1. The input x of *f(x)* can be represented by a 3-digit binary number, and its output is represented by a 4-digit binary number. Write the truth table of function *f(x)* ***of your own SID***, and derive the corresponding logic expression of *f(x)* using the K-Map approach.

**(10%)**

*For example, the truth table for the above example is (where i1, i2, i3 are input bits, and o1, o2, o3, o4 are output bits):*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *i1* | *i2* | *i3* | *o1* | *o2* | *o3* | *o4* |
| *0* | *0* | *0* | *0* | *1* | *0* | *0* |
| *0* | *0* | *1* | *0* | *1* | *0* | *1* |
| *0* | *1* | *0* | *0* | *1* | *1* | *1* |
| *0* | *1* | *1* | *0* | *0* | *1* | *1* |
| *1* | *0* | *0* | *1* | *0* | *0* | *0* |
| *1* | *0* | *1* | *0* | *1* | *0* | *0* |
| *1* | *1* | *0* | *0* | *0* | *1* | *0* |
| *1* | *1* | *1* | *1* | *0* | *0* | *1* |

1. Build circuits in Logisim with 3-bit input and 4-bit output to realize function *f(x)* according to the logic expression in (2).

**(20%)**

1. Build a 4-bit ripple carry adder in Logisim. You should directly use basic gates to build the ripple carry adder by yourself, but not use the built-in adder in Logisim.

**(20%)**

1. Based on the circuits you build in (2) and (3), add other basic gates and wires to build circuits to implement the function *g(x)* which is defined as follows:

The input of this circuit is the binary representation of *x*, and the output is the binary representation of the value of *g(x)*. You should decide the number of bits of the input and output of *g(x)* by yourself.

**(30%)**