School of Computer Science
Computer Fundamentals (COMP1036) Autumn 2024

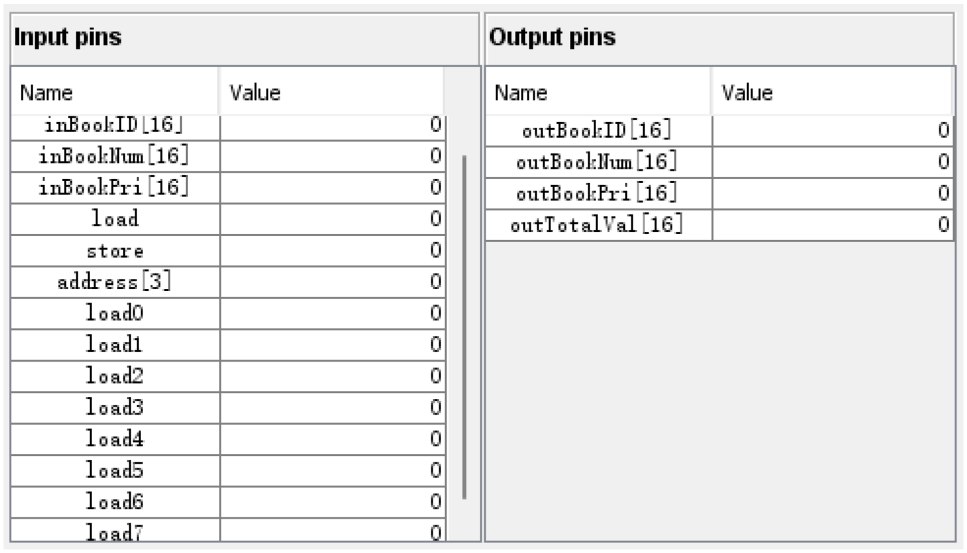
Custom Library System
Deadline: 17:00 Friday 15th of November, 2024

Collaborating in small groups of up to three students is permitted, but you must implement your own programs (absolutely *do not* copy and paste from others) and provide your own answers where appropriate.

Note that lacking proper comments will lose mark.

Description

You have used combinatorial logic to construct various logic circuits that form the basis of a computer in previous exercises. In this exercise, you will implement a library system with some simple functionalities. Additionally, you will incorporate arithmetic operations into your design. The inputs and outputs of the library system can be found in Figure 1.



Input pins

Name	Value
inBookID[16]	0
inBookNum[16]	0
inBookPri[16]	0
load	0
store	0
address[3]	0
load0	0
load1	0
load2	0
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins

Name	Value
outBookID[16]	0
outBookNum[16]	0
outBookPri[16]	0
outTotalVal[16]	0

Figure 1: Library System

1 Load Data to Registers

Turn on **Load** (load = 1) to start loading **inBookID**, **inBookNum**, and **inBookPri** into the Library System's registers, respectively.

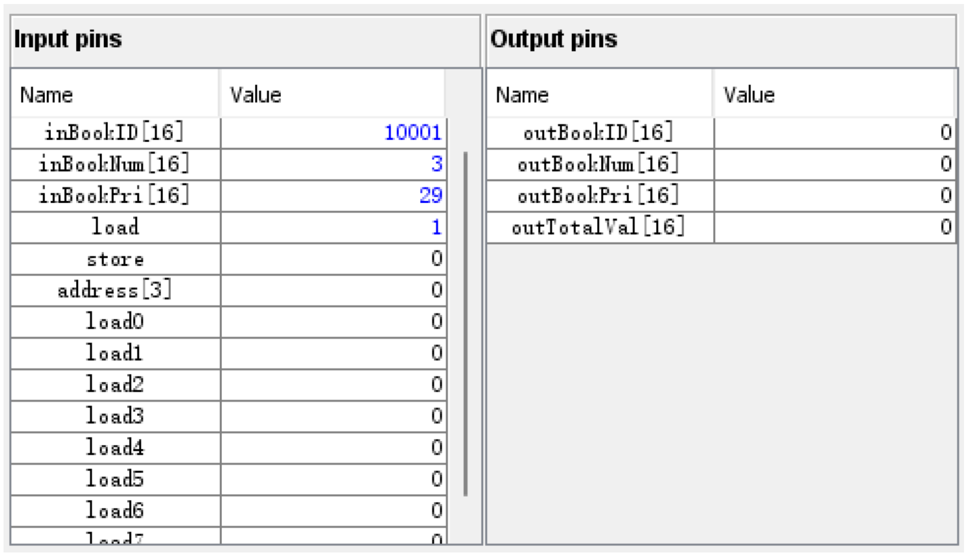
Input:

1. A 16-Bit version of a book ID number called **inBookID**.
2. A 16-Bit version of a book quantity called **inBookNum**.
3. A 16-Bit version of a book price called **inBookPri**.
4. A 1-Bit switch for loading the data called **load**.

Turn on the **load** (load = 1), then input **inBookID**, **inBookNum**, and **inBookPri**, which should be stored in the registers, respectively.

Output:

1. A 16-Bit version of a book ID number called **outBookID**.
2. A 16-Bit version of a book quantity called **outBookNum**.
3. A 16-Bit version of a book price called **outBookPri**.
4. A 16-Bit version of total book value called **outTotalVal**.



Input pins

Name	Value
inBookID[16]	10001
inBookNum[16]	3
inBookPri[16]	29
load	1
store	0
address[3]	0
load0	0
load1	0
load2	0
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins

Name	Value
outBookID[16]	0
outBookNum[16]	0
outBookPri[16]	0
outTotalVal[16]	0

Figure 2: Load data to register.

(6 marks)

2 Store Data to RAM

Based on **Task 1**, turn on the **store** (store = 1) with the assigned address; the data should be stored in the RAMs, respectively.

Input:

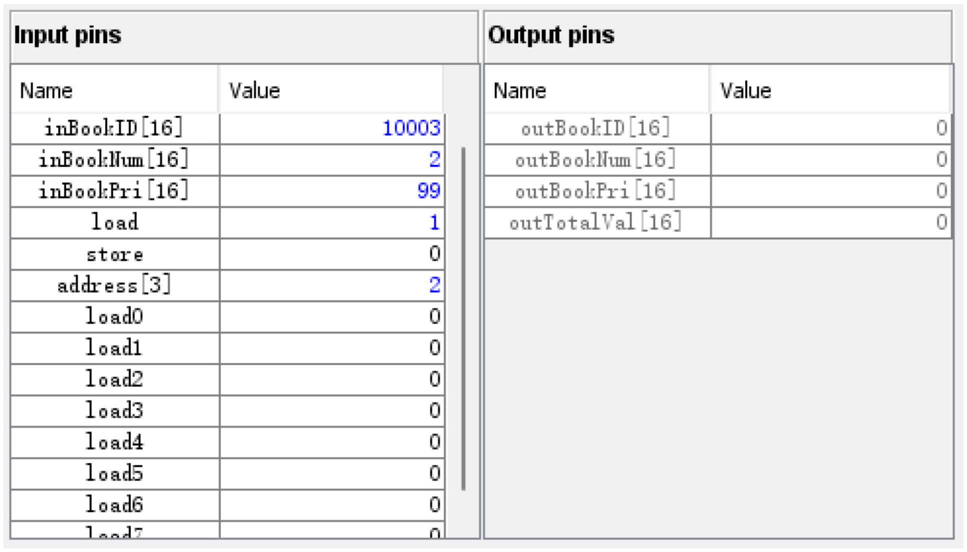
1. A 1-Bit switch for storing the data called **store**.
2. A 3-Bit value assigned to **address** that stores each value in RAM.

Turn on the **store** (store = 1) and turn off **load** (load = 0) with the assigned address; **inBookID**, **inBookNum**, and **inBookPri** should be stored in the RAMs. Meanwhile, **outBookID**, **outBookNum**, and **outBookPri** will display the stored data, and the default value of **outTotalVal** is 0. For example, after loading the above input into the register, you should set **store = 1** and **address = 2**. The input data will be stored in the RAMs, with **outBookID** = 10003, **outBookNum** = 2, **outBookPri** = 99, and **outTotalVal** = 0. Figures 3 and 4 illustrate the loading of data to the register and RAM, respectively.

Output: Book information is based on the assigned address of RAM.

1. A 16-Bit version of a book ID number called **outBookID**.
2. A 16-Bit version of a book quantity called **outBookNum**.
3. A 16-Bit version of a book price called **outBookPri**.
4. A 16-Bit version of total book value called **outTotalVal**.

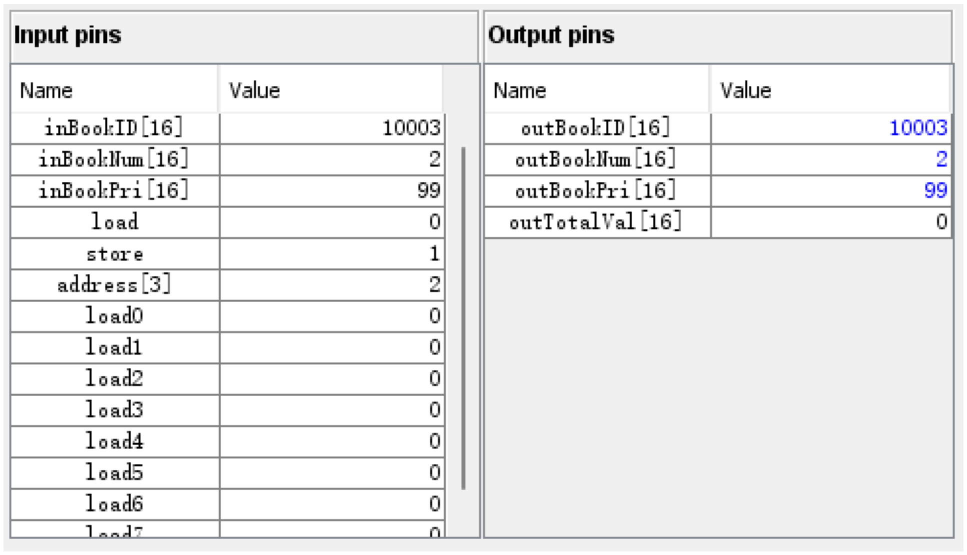
(6 marks)



Input pins	
Name	Value
inBookID[16]	10003
inBookNum[16]	2
inBookPri[16]	99
load	1
store	0
address[3]	2
load0	0
load1	0
load2	0
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins	
Name	Value
outBookID[16]	0
outBookNum[16]	0
outBookPri[16]	0
outTotalVal[16]	0

Figure 3: Load data to register.



Input pins	
Name	Value
inBookID[16]	10003
inBookNum[16]	2
inBookPri[16]	99
load	0
store	1
address[3]	2
load0	0
load1	0
load2	0
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins	
Name	Value
outBookID[16]	10003
outBookNum[16]	2
outBookPri[16]	99
outTotalVal[16]	0

Figure 4: Store data to RAM with assigned address 2 and display the stored data in output pins.

3 Store inBookNum × inBookPri to RAM

Based on **Task 2**, **inBookNum** × **inBookPri** should be calculated and stored in the RAM as well.

Input:

1. A 16-Bit version of a book quantity called **inBookNum**.
2. A 16-Bit version of a book price called **inBookPri**.

For example, if we have the following input data:

1. **10001 3 29**
2. **10002 5 19**
3. **10003 2 99**

Then the data stored in the RAM should be as follows:

1. **10001 3 29 87**
2. **10002 5 19 95**
3. **10003 2 99 198**

(5 marks)

4 Sequential Load from RAM

Based on **Task 3**, implement sequential load functionality. The values of **inBookNum** $\times$ **inBookPri** will be loaded from RAM according to the assigned address and stored to registers. load0-load7 are the switches that control loading data from addresses 0 to 7 of RAM to the registers, respectively.

Input:

1. A 3-Bit value assigned to **address** that stores each value in RAM.
2. A 1-Bit switch for loading data, called **load0**, from address 0 in RAM.
3. A 1-Bit switch for loading data, called **load1**, from address 1 in RAM.
4. A 1-Bit switch for loading data, called **load2**, from address 2 in RAM.
5. A 1-Bit switch for loading data, called **load3**, from address 3 in RAM.
6. A 1-Bit switch for loading data, called **load4**, from address 4 in RAM.
7. A 1-Bit switch for loading data, called **load5**, from address 5 in RAM.
8. A 1-Bit switch for loading data, called **load6**, from address 6 in RAM.
9. A 1-Bit switch for loading data, called **load7**, from address 7 in RAM.

For example, we have following data stored in the RAM:

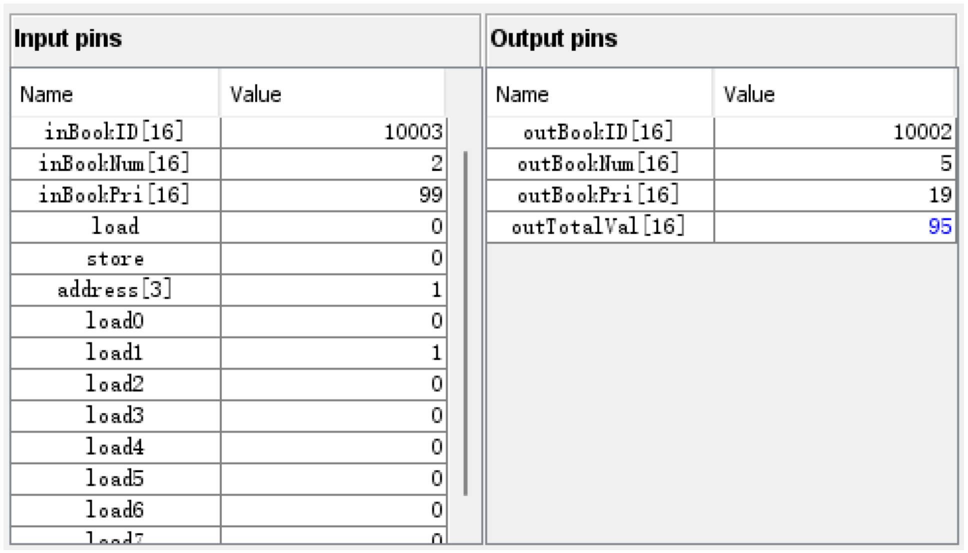
1. **10001 3 29 87**
2. **10002 5 19 95**
3. **10003 2 99 198**

In Figure 5, set **load1** = 1 and **address** = 1. The value **95** will be loaded to a register from RAM; meanwhile, **outBookID**, **outBookNum**, **outBookPri** and **outTotalVal** will display **10002**, **5**, **19**, and **95**, respectively.

Output: Book information is based on the assigned address of RAM.

1. A 16-Bit version of a book ID number called **outBookID**.
2. A 16-Bit version of a book quantity called **outBookNum**.
3. A 16-Bit version of a book price called **outBookPri**.
4. A 16-Bit version of total book value called **outTotalVal**.

(8 marks)



Input pins

Name	Value
inBookID[16]	10003
inBookNum[16]	2
inBookPri[16]	99
load	0
store	0
address[3]	1
load0	0
load1	1
load2	0
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins

Name	Value
outBookID[16]	10002
outBookNum[16]	5
outBookPri[16]	19
outTotalVal[16]	95

Figure 5: Load data from RAM to register.

5 Sum of the inBookNum $\times$ inBookPri

Based on **Task 4**, calculate the total value of the books stored in RAM.
Input:

1. A 3-Bit value assigned to **address** that stores each value in RAM.
2. A 1-Bit switch for loading data, called **load0**, from address 0 in RAM.
3. A 1-Bit switch for loading data, called **load1**, from address 1 in RAM.
4. A 1-Bit switch for loading data, called **load2**, from address 2 in RAM.
5. A 1-Bit switch for loading data, called **load3**, from address 3 in RAM.
6. A 1-Bit switch for loading data, called **load4**, from address 4 in RAM.
7. A 1-Bit switch for loading data, called **load5**, from address 5 in RAM.
8. A 1-Bit switch for loading data, called **load6**, from address 6 in RAM.
9. A 1-Bit switch for loading data, called **load7**, from address 7 in RAM.

For example, we have following data stored in the RAM:

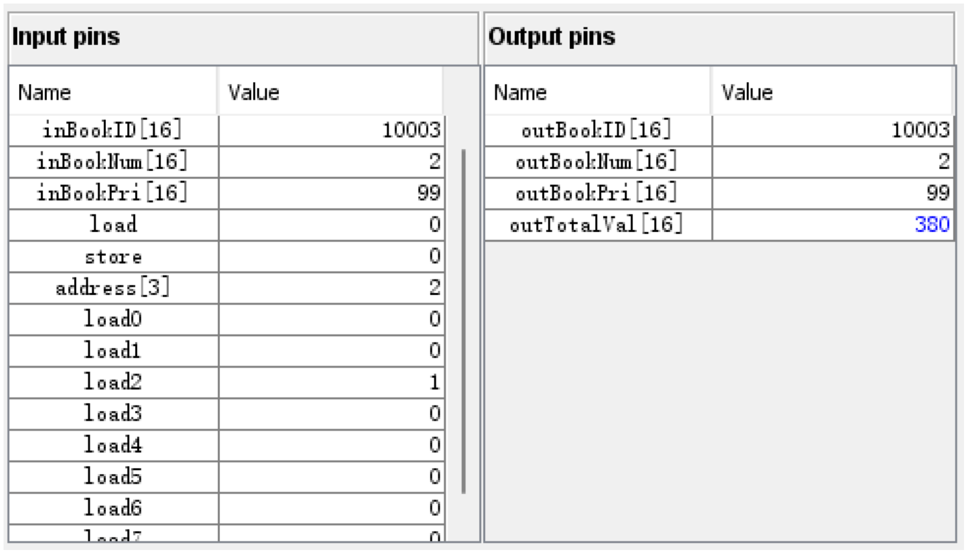
1. **10001 3 29 87**
2. **10002 5 19 95**
3. **10003 2 99 198**

Retrieve data from RAMs sequentially, calculate **87 + 95 + 198 = 380** and **outTotalVal** will display the sum **380**. Meanwhile, the most recently retrieved data (**address = 2**) will be displayed in **outBookID = 10003**, **outBookNum = 2**, and **outBookPri = 99**, respectively.

Output: Book information is based on the assigned address of RAM.

1. A 16-Bit version of a book ID number called **outBookID**.
2. A 16-Bit version of a book quantity called **outBookNum**.
3. A 16-Bit version of a book price called **outBookPri**.
4. A 16-Bit version of total book value called **outTotalVal**.

(5 marks)



Input pins

Name	Value
inBookID[16]	10003
inBookNum[16]	2
inBookPri[16]	99
load	0
store	0
address[3]	2
load0	0
load1	0
load2	1
load3	0
load4	0
load5	0
load6	0
load7	0

Output pins

Name	Value
outBookID[16]	10003
outBookNum[16]	2
outBookPri[16]	99
outTotalVal[16]	380

Figure 6: Calculate total value.

Submission

You should zip all your files into one zip file to “Coursework1 Assignment”. You should name your file as: `YOURSTUDENTID_YOURNAME`.zip. Your zip file should include:

1. one master hdl file called CW.hdl file
2. all additional hdl files used by CW.hdl
3. one readme.txt file if needed (optional)

Submit your zip file onto the Moodle submission page. Please note that every next submission overwrites all the files in the previous one. If you submit several times, make sure that your last submission includes all the necessary files. Include all required chips, instructions for use, and any instruction text file if necessary. For late submission, the standard late submission policy applies, i.e. 5% mark deduction for every 24 hours.