Logic Design

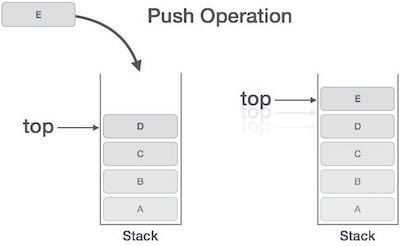
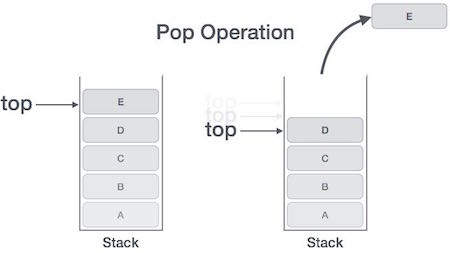
EECS101001

Lab 5: Stack Machine – Report

104021215 熊 磊

1. Foreword

This Lab is trying to design a Stack Machine, which is a **Last In First Out** structure. And also support basic operation: addition, subtraction and multiplication.

SM

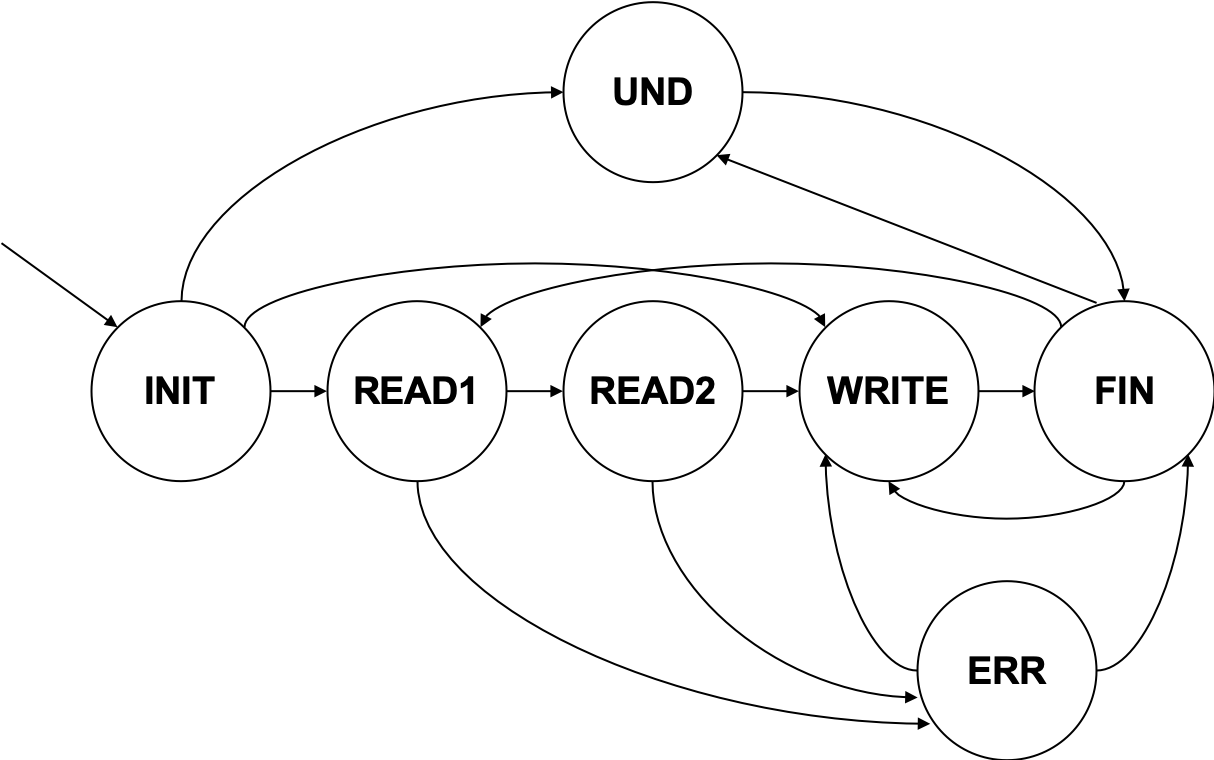
|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **size** | **function** | **I/O** |
| clk | 1 | 時間訊號 | input |
| rst\_n | 1 | rst\_n = 0時進行reset | input |
| instr | 13 | instruction | input |
| pc | 10 | instruction address，instruction的總長度放在pc = 1023的位置 | output |
| err\_code | 3 | 錯誤訊息 | output |
| d\_valid | 1 | 當輸出的資料為add, sub, mul的最終結果時時把d\_valid拉起 | output |
| out\_data | 20 | 輸出的資料 | output |
| fin | 1 | 當處理完所有的instruction之後拉成1 | output |

SM\_Mem

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **size** | **function** | **I/O** |
| r\_data | 20 | 從stack memroy讀出來的資料 | input |
| full | 1 | 當stack memory 滿的時候拉為一 | input |
| empty | 1 | 當stack memory 是空的時候拉為一 | input |
| cntrl | 2 | 控制stack memory的功能 | output |
| w\_data | 20 | 要寫入stack memory的資料 | output |

1. Implementation Details and Designs

First, we draw a State Transition Graph as follow:



We need at least 7 states to handle different situations and operations. Hence, we define these states in 3-bits. Also, we define instruction’s name.

|  |  |
| --- | --- |
| **State Name** | **Instruction** |
| READ1 | POP a number from stack. |
| READ2 | POP a number from stack. |
| WRITE | Do operation and push result to stack, or just push a number to stack. |
| FIN | Get a new address (pc), and change to next state. |
| ERR | If it is an invalid operation (pop from empty stack or push to full stack), will be in **ERR** state. And if we need to restore number back to stack, then it turns to **WRITE** state. |
| UND | If it is an undefined instruction, then it turns to **FIN** state, waiting for next instruction. |



Now we determine when to change states and change to which state. If it detects an error occurred, it will turn to corresponding state, **ERR (Error operation)** or **UND (Undefined Operation)**.

一張含有 螢幕擷取畫面 的圖片

自動產生的描述

Also, we need some wires to do restore (if error occurred), error code, read data, write data, control operation, and signal from the Stack Memory determine whether it is full or empty. Here is a list for each instruction of wire.

|  |  |
| --- | --- |
| **wire** | **function** |
| state | Transmit the current state, initial will be **INIT**. |
| pc | Transmit the address to get the instruction from input, initial will be **1023**, to get the total numbers of instructions. |
| len | Store the numbers of instructions. |
| data | Store the first data pop from SM\_Mem. |
| data2 | Store the second data pop from SM\_Mem. |
| restore | If it is an invalid operation, and we have to roll back to original situation. |
| cnt | cnt is a signal to transmit whether the writing data is the result of operation, so that d\_valid can raise up. |
| r\_data | r\_data is a data poping from SM\_Mem. |
| w\_data | w\_data is a data pushing to SM\_Mem. |
| full | A signal to transmit if SM\_Mem is full. |
| empty | A signal to transmit if SM\_Mem is empty. |

In **SM\_Mem**, we use num1, num2, …, num8 to store each number in stack memory. And using top to indicate top element of stack.



Finally, if we instantiate SM\_Mem in SM, we can push number to or do operation from stack memory.

1. Block Diagram

一張含有 籠, 建築物, 室外, 天空 的圖片

自動產生的描述

1. nWave Result

Without Error Detection

一張含有 電子用品, 螢幕擷取畫面 的圖片

自動產生的描述With Error Detection

一張含有 螢幕擷取畫面, 電子用品 的圖片

自動產生的描述

1. Report Area

一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

1. Encountered Problems and Solving

* To start this lab, I am confusing how to determine the states I need. Can I directly use instruction as state? I think it is not easy do define and might be ambiguous in changing state. Hence, I define another 7 states, and it make me more easily in later work.
* If I assign value without using DFF, e.g. **counter**, the value will be wrong value.

1. Questions

* I use 8 DFFs to store numbers in Stack Memory, is there any other to design a stack machine?

1. Impression and Experience

Stack Machine is a simple idea, however, in implementation we have to consider many aspects. And this is our last Lab, after these labs and the course, I think that I have basic concept in Verilog! Thank TAs.