NCTU-EE IC LAB – Spring 2018

Midterm Project

Design: Breadth First Search

Data Preparation

1. Extract test data from TA's directory:

% tar xvf ~iclabta01/Mid.tar

2. The extracted LAB directory contains:
    a. EXERCISE/: your design

Design Description

In this lab, Breadth-First Search (BFS) will be implemented to find a solution path of a maze. There will be 3 prizes in the maze and BFS will be implemented to find all of the prize.

First, the prize nearest to the initial position will be found. Then the next prize is found such that it is nearest to the second prize and so on for the last prize.

Once the all prizes are found, the path starting from the initial point to the coordinate of the last prize will be returned.

This design will search through a 15-by-15 maze **with walls surrounded**. The **wall is denoted as 1 and the path is denoted as 0**. The coordinates of the **start point** and the **prizes** will be fed to the design during initialization.

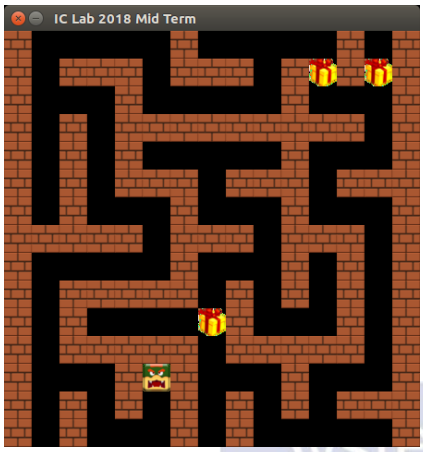
The coordinates of the maze are given such that (0, 0) refers to the top-left of the maze and (14, 14) refers to the bottom-right of the maze. The path output has to obey this restriction.

For more details from mathematical view point, please reference the following link: http://bryukh.com/labyrinth-algorithms/

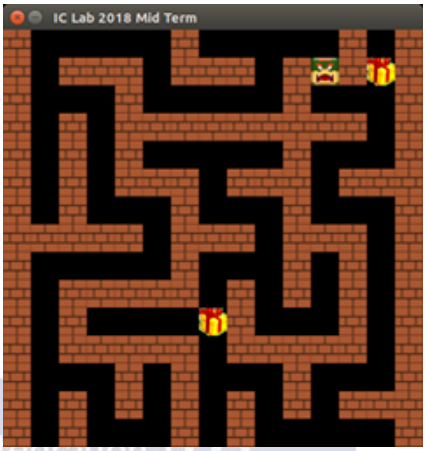
A software implementation made for this lab using Python can be found in this link: https://github.com/eugenelet/Maze-Traversing-using-BFS

Example

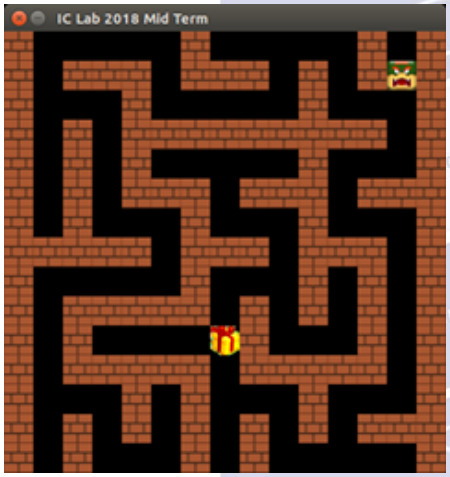
Example maze:



1



2



3

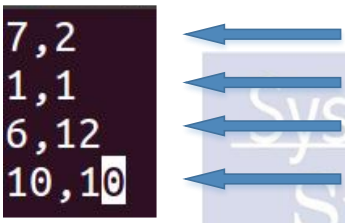
4

Files

All files are stored in ~iclabta01/mid_2018/… and are separated into 3 directories (ans, init, maze). All directories contain 5 files that can be worked on: 00[0~4].txt

The format of files in ***init*** and ***ans*** is:
[col,row]
* col increases in the horizontal direction and row in the vertical direction

~iclabta01/mid_2018/init/xxx.txt



~iclabta01/mid_2018/maze/xxx.txt

Top left of maze

```
1,0,1,0,0,0,0,0,0,0,1,0,0,0,1
1,0,1,0,1,1,1,1,1,0,1,0,1,0,1
1,0,0,0,1,0,1,0,0,0,1,0,1,0,1
1,0,1,1,1,0,1,1,1,1,1,1,1,0,1
1,0,1,0,0,0,0,0,0,0,0,0,0,0,1
1,0,1,1,1,0,1,1,1,0,1,1,1,1,1
1,0,0,0,1,0,0,0,1,0,1,0,0,0,1
1,1,1,0,1,0,1,0,1,1,1,0,1,0,1
1,0,0,0,1,0,1,0,1,0,0,0,1,0,1
1,0,1,1,1,1,1,0,1,0,1,1,1,0,1
1,0,1,0,0,0,1,0,0,0,0,0,1,0,1
1,0,1,0,1,0,1,1,1,1,1,1,1,0,1
1,0,1,0,1,0,0,0,0,0,0,0,0,0,1
1,0,1,0,1,1,1,1,1,1,1,1,1,0,1
1,0,0,0,1,0,0,0,0,0,0,0,0,0,1
```

Bottom right of maze

~iclabta01/mid_2018/ans/xxx.txt

Results (path) are returned as follow:

starting position -> first prize -> second prize -> third prize

```
7,2
8,2
9,2
9,1
9,0
8,0
7,0
6,0
5,0
4,0
3,0
3,1
3,2
2,2
1,2
1,1
1,2
1,3
1,4
1,5
```

Starting position

First Prize

-
-
-

Input and Output

Input signal	Bit width	Definition
clk	1	Clock
rst_n	1	Asynchronous active-low reset
in_valid	1	Enable input signal
maze	15	Maze input (1: wall, 0: path)
player_x	4	Initial x coordinate of player
player_y	4	Initial y coordinate of player
prize_1_x	4	X coordinate of prize 1
prize_1_y	4	Y coordinate of prize 1
prize_2_x	4	X coordinate of prize 2
prize_2_y	4	Y coordinate of prize 2
prize_3_x	4	X coordinate of prize 3
prize_3_y	4	Y coordinate of prize 3

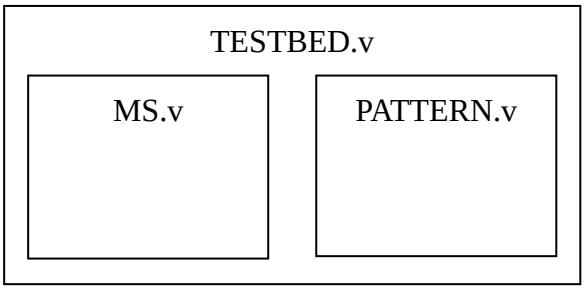
Output signal	Bit width	Definition
out_valid	1	Enable output check
out_x	4	Output coordinate x
out_y	4	Output coordinate y

Specification

1. Top module name: **MS** (design file name: **MS**.v)
2. It is **asynchronous** reset and **active-low** architecture.
3. The reset signal would be given only once at the beginning of simulation. All output signals should be reset after the reset signal is asserted.
4. The max clock period of the design is 5 **ns**. You can optimize the clock cycle.
5. After two cycles of rst_n and out_valid, in_valid will pull up.
6. The maze signal will be given when in_valid pull up.
7. The input delay is set to **0.5*clock cycle**.
8. The output delay is set to **0.5*clock cycle**, and the output loading is set to **0.05**.
9. The synthesis result of data type **cannot** include any latches.
10. After synthesis, you can check **MS**.area and **MS**.timing. The area report is valid when the slack in the end of timing report should be **non-negative**.
11. **Area** of your design must be **lower than 250000**.
12. Your design should output its result within **1000 cycles**.
13. The default memory is 256x4 (words x bits) with mux 4 (area = 45372.905), and

you can modify if you need it.

Block Diagram

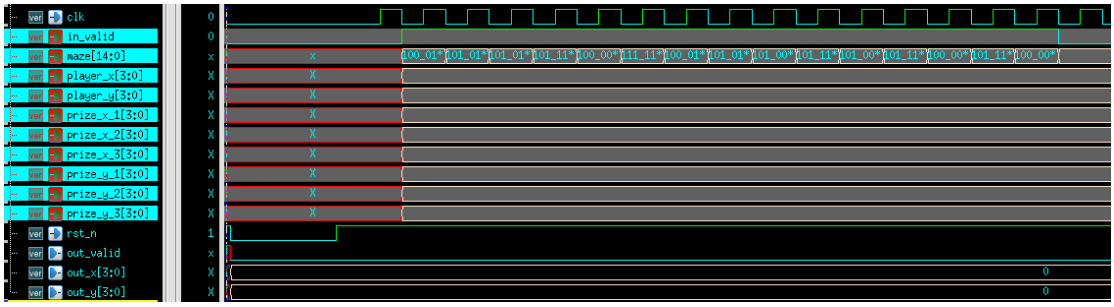


Note

1. **Grading policy:**
   RTL and Gate-level simulation correctness: 70%
   Performance: 30%
   - Latency 15%
   - Area 15%
2. **Please upload the following file on e3 platform before 12:00 p.m. on May. 16:**
   - **MS_iclabXX**.v
   - **iclabXX_your_clock_period.txt** (ex: iclab01_5.txt)
   - **memorysize.txt** (format: words_bits_mux.txt, ex: 256_2_4.txt)
3. **Template folders and reference commands:**
   01_RTL/ (RTL simulation) **./01_run**
   02_SYN/ (Synthesis) **./01_run_dc**
   (Check the design if there's latch or not in ***syn.log*****)**
   (Check the design's timing in /Report/**MS.*timing***)
   03_GATE/ (Gate-level simulation) **./01_run**
   04_MEM/ (Your RA1SH memory):
   ./01_mem_gen.sh (create the memory) (create the script by yourself)
   ./02_lib_gen_syntax_match.sh (make sure the format is correct)

Example wave form

Input:



Output: