NYCU-EE IC LAB – SPRING2023

Lab02 Exercise

Design: 12-QUEEN

Data Preparation

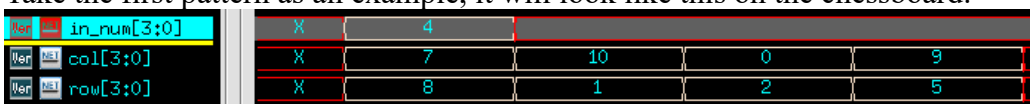
1. Extract test data from TA's directory:
   % tar -xvf ~iclabta01/Lab02.tar
2. The extracted Lab02/ directory contains:
   a. Practice
   b. Exercise

Design Description

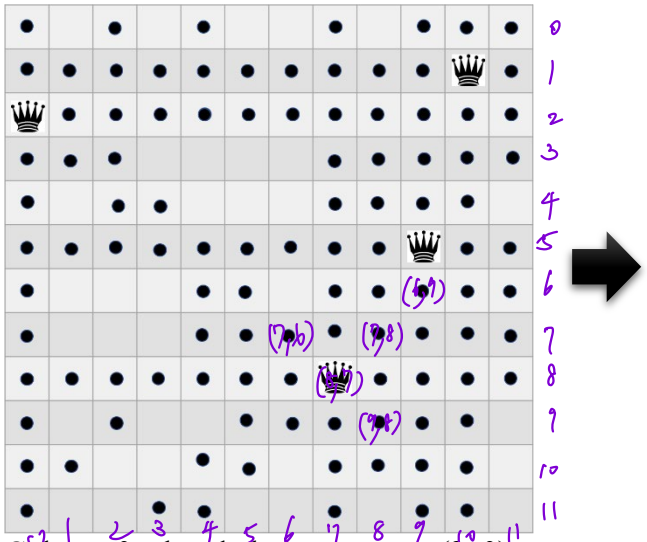
I took the opportunity to play Chinese chess and stumbled upon the classic mathematical problem of the 8x8 chess board. However, I felt that it was too small for me and I wanted something bigger and more challenging. That's why I chose the 12 Queen's problem.

In this lab, you need to place **12 queens** in the **12x12 chessboard** so that none of any two queens threaten each other; that is, you need to find a solution of **12 queen's positions** that **none of any two queens share the same row, column, or diagonal.** All the test patterns have at least one solution, you need to find **the first solution** due to the priority.

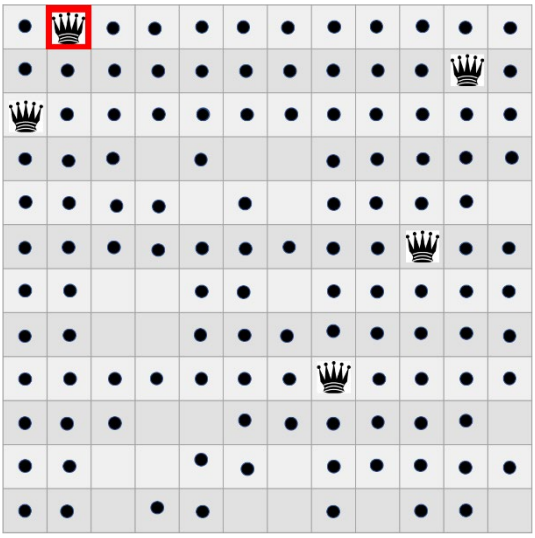
Take the first pattern as an example, it will look like this on the chessboard.



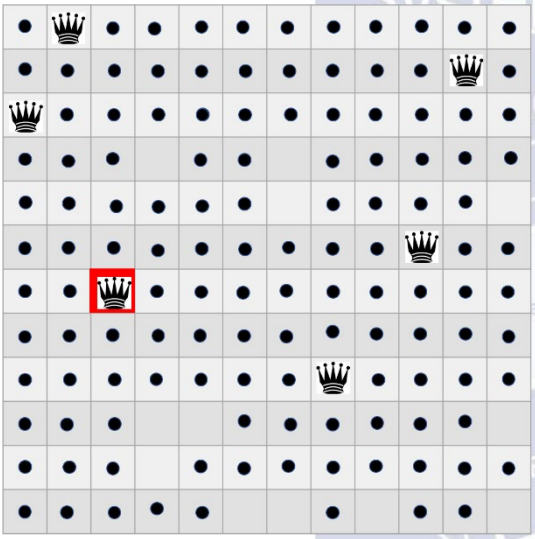
The user gets 4 positions.
I will be using (row, col) as the notation for representation later.
The user gets the positions (8,7). (1,10). (2,0) and (5,9).
Choose positions column by column, "smaller column values will choose smaller row values first."
Initial: we can choose (0,1), (4,1), (6,1), (7,1), (9,1), (11,1), and we will choose (0,1) first, and so on……



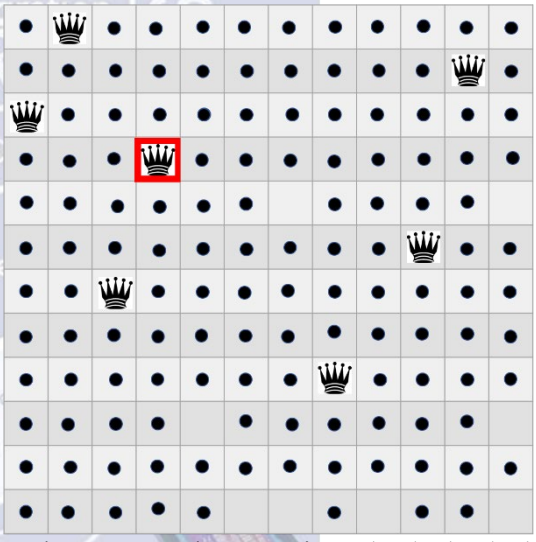
Column 0: already has a queen at (2, 0)



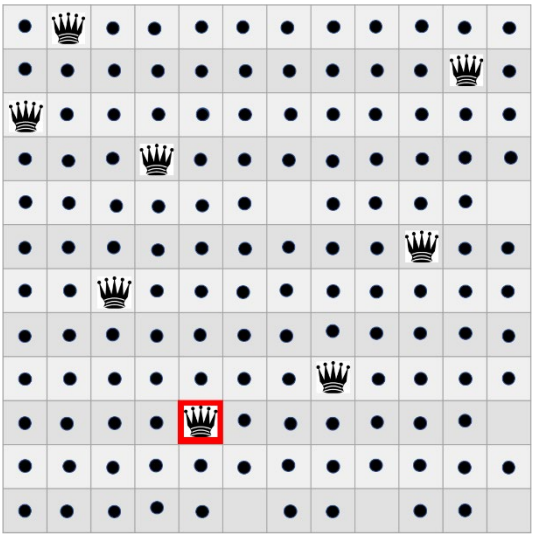
Column 1: we have options (0,1), (4,1), (6,1), (7,1), (9,1), (11,1), choose (0,1) first.



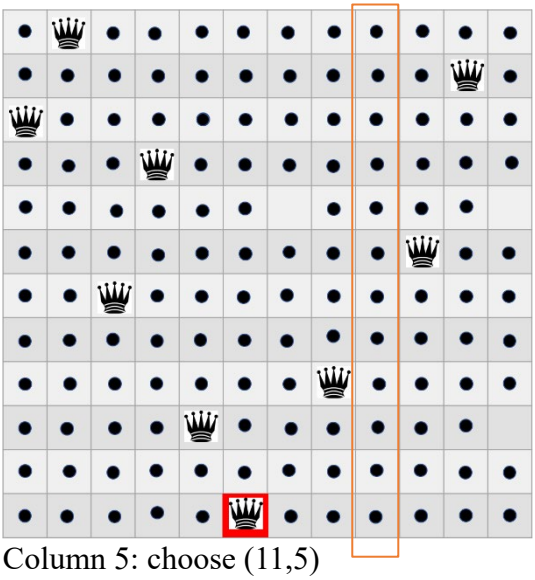
Column 2: we have options (6,2), (7,2), (10,2), (11,2), choose (6,2) first.



Column 3: we have options (3,3), (9,3), (10,3), choose (3,3) first.

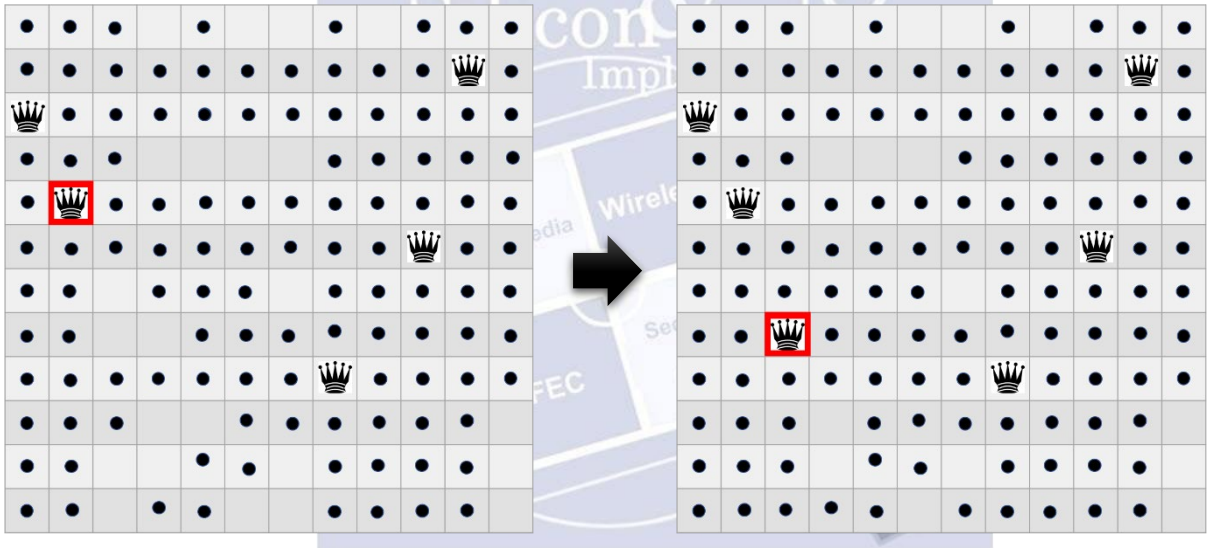


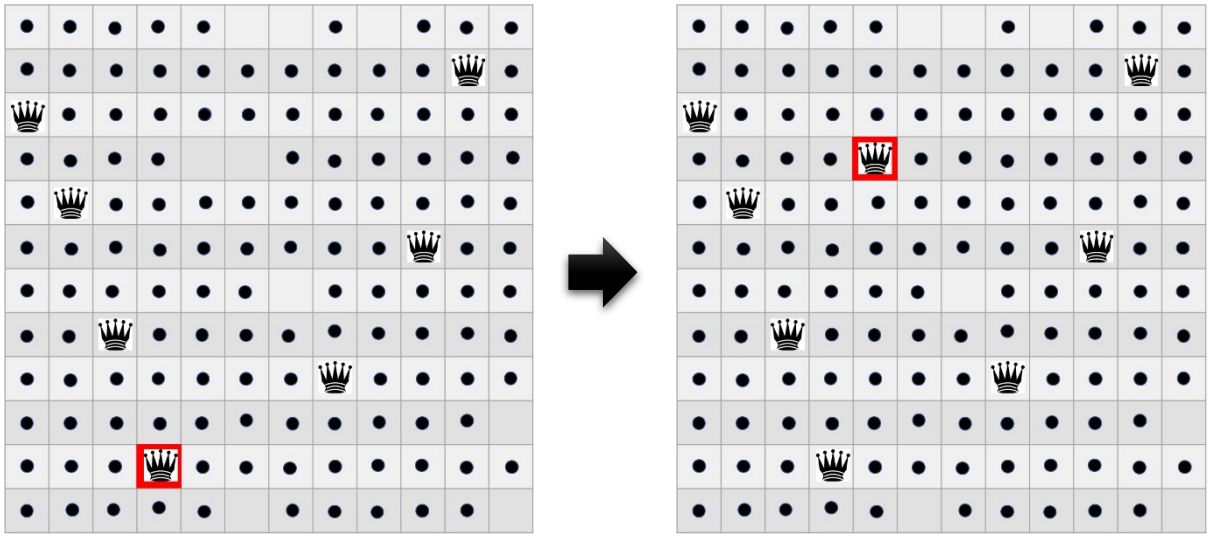
Column 4: choose (9,4)

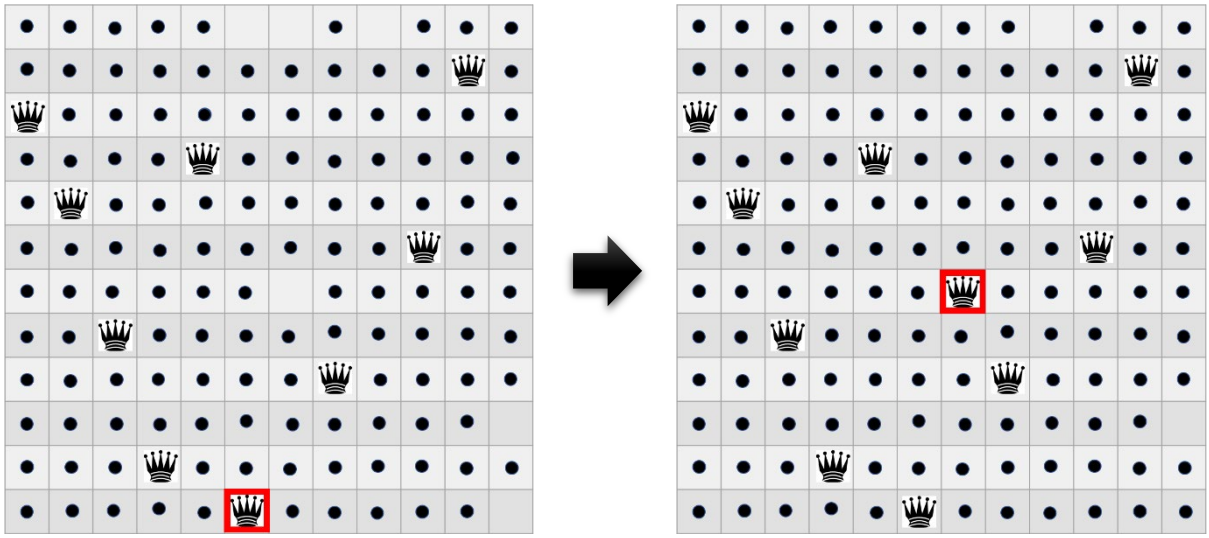


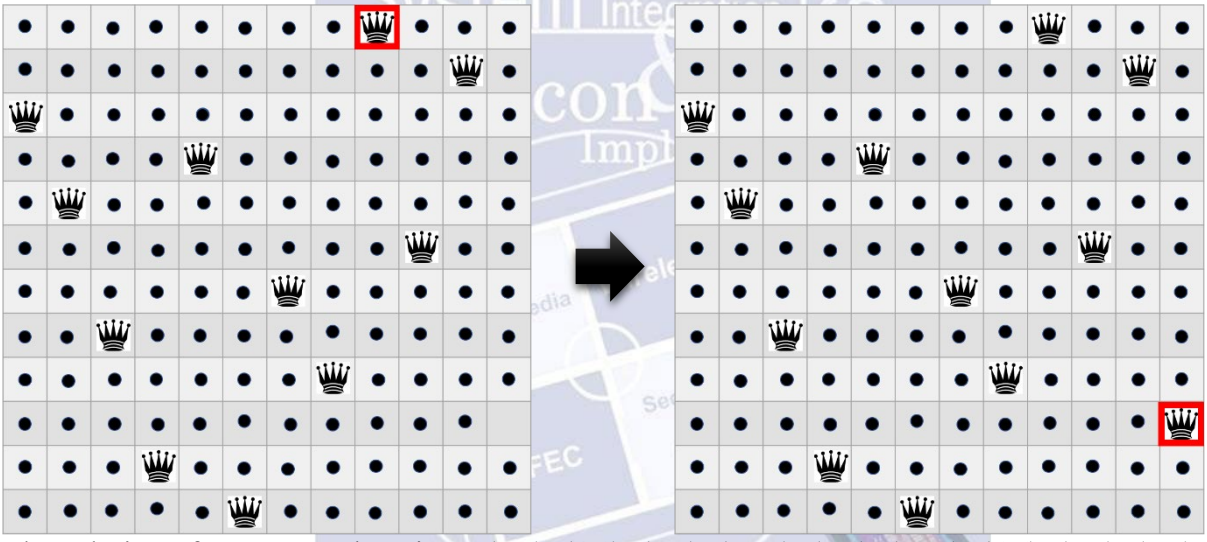
Column 5: choose (11,5)
No solution on column 8

Follow the priority and repeat the above process several times, you can find the first correct answer
Final ans:









The solution of 12 queen's locations: (2,0), (4,1), (7,2), (10,3), (3,4), (11,5), (6,6), (8,7), (0,8), (5,9), (1,10), (9,11). You need to output the **row positions of the 12 queens column by column**; that is, 2 → 4 → 7 → 10 → 3 → 11 → 6 → 8 →0 → 5 → 1 → 9.

Inputs

1. Input signals：

Input Signals	Bit Width	Definition
clk	1	Clock.
rst_n	1	Asynchronous active-low reset.
in_valid	1	High when col and row signals are valid.
in_valid_num	1	High when in_num signals are valid.
col,row	4	(col, row) is queen's location
in_num	3	number of inputs

2. The **col** [3:0] and **row** [3:0] are valid only when **in_valid** is high.
3. All input signals will be synchronized at negative edge of the clock.
4. in_num is at most 3'd6

PS: all of input at least have one solution

- Example:

  Given input waveform as below:

  Chessboard is 12x12, which is from 0 to 11, in this waveform means that first chess is located on (7,7), second is located on (10,8), and so on.

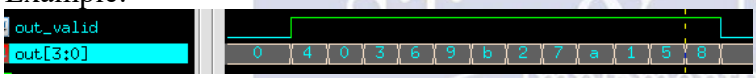
Outputs

1. Output signals:

Output Signals	Bit Width	Definition
out_valid	1	High when output is valid
out	4	Row positions of 12 queens column by column

2. All outputs should be low after initial reset.
3. Output signals should be synchronized at clock positive edge.
4. **out_valid** should not be raised when **in_valid** is high.
5. **out_valid** is set to high when **out**[3:0] is valid, and will be high for 12 cycle when be triggered.

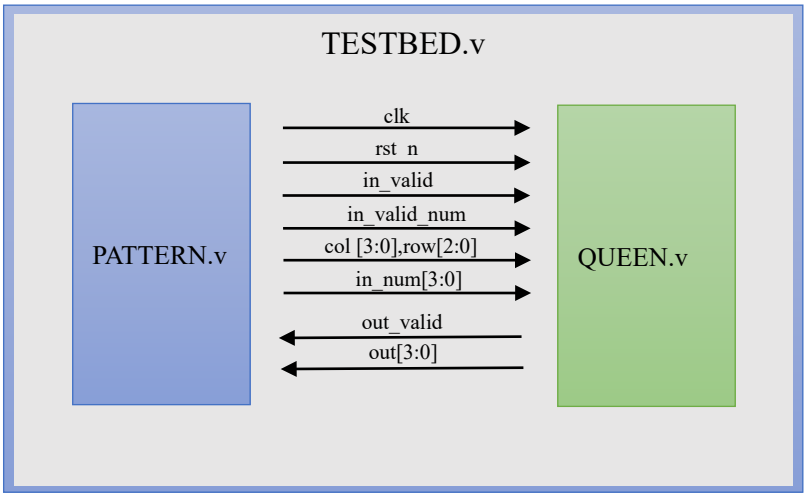
6. TA's pattern will capture your output for checking at clock negative edge.
7. Example:



Specifications

1. Top module name：QUEEN (Filename: QUEEN.v)
2. It is an asynchronous reset and active-low architecture. If you use synchronous reset (reset after clock starting) in your design, you may fail to reset signals.
3. The clock period of the design is fixed to 6ns.
4. The next group of inputs will come in 2~5 cycles after your out_valid pull down.
5. The synthesis result of data type cannot include any LATCH.
6. After synthesis, you can check QUEEN.area and QUEEN.timing.
7. The slack in the timing report should be non-negative and the result should be MET.
8. The gate level simulation cannot include any timing violation.
9. The latency of your design in each pattern should not be larger than 100000 cycles.
10. No **ERROR** is allowed in every simulation/synthesis.
11. No timing violation in gate level simulation
12. Any words with "error", "latch" or "congratulation" can't be used as variable name.
13. No Lookup table

Block Diagram



Note

1. **Grading policy:**
   RTL and gate-level simulation correctness: 70%
   Performance: 30%
   - (Area * Execution Cycle) 30%
   - The grade of the 2nd demo will be 30% off.
2. **Please submit following files under 09_UPLOAD:**
   - QUEEN.v
   - If uploaded files violate the naming rule, you will get 5 deduct points.
3. **Deadline:**
   a. 1st_demo deadline: 2023/03/6(Mon.) 12:00:00
   b. 2nd_demo deadline: 2023/03/8(Wed.) 12:00:00
4. **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/***QUEEN.timing***)
03_GATE / (Gate-level simulation) **./01_run**
09_UPLOAD/(submit files) **./01_submit**
09_UPLOAD/(check files) **./02_check**

4. **There is a pattern about the demo**
"Due to the difficulty in controlling the overall number of cycles and to prevent students from facing challenges in making trade-offs on the circuit, the final demo pattern data will randomly select input "in_num" in the range of 1 to 6."
Note: The input are all randomly generated.

Sample Waveform

- The next in_valid will come in 2~5 cycles after out_valid pulls down.