Summer training 2020

Lab02 Homework

Design: LU Matrix Factorization

Data Preparation

1. Extract test data from TA's directory:

   % tar xvf ~train_ta/Lab02.tar

Design Description

LU decomposition is a basic concept in linear algebra, it indicates the process and the result of Gaussian elimination, and factors a matrix as:

$$A = LU$$

, where L stands for unit lower triangular matrix, and U stands for upper triangular matrix. L and U have the form:

$$L = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix}$$

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

Partial Pivoting

Before doing LU factorization, proper matrix permutation can prevent us from getting ill-conditioned U or zero pivot. In this exercise, we use permutation in rows, which is called **LU factorization with partial pivoting**. Before doing each Gaussian elimination, find the maximum absolute values in r_{th} column, for example, k_{th} row contains this element. Then switch r_{th} row with k_{th} row. So now the element is at pivot position (r, r). And we can get:

$$PA = LU$$

, where

$$L = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix}$$, an unitriangular matrix

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$, a triangular matrix, and $u_{ii} \neq 0$

Example1:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

$$A' = L_1 A = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$A'' = L_2 A' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 2 \end{bmatrix} = U$$

$$L = L_1^{-1} L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

In this case, $P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Example2:

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Since a_{11} is 0, swap row_1 and row_2,

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

$$A' = L_1 A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & -1 \end{bmatrix}$$

$$A'' = L_2 A' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -2 \end{bmatrix} = U$$

In this case, $P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $PA = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

L_1 and L_2 have to be recalculated since the order of rows in A is changed.

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -2 \end{bmatrix}$$

In this exercise, you have to determine whether a given matrix A is invertible first. If A isn't invertible, then you don't have to do LU decomposition, set both invertible and decomposable to zero(but notice that it's possible to have non-invertible matrix decomposed, but for simplicity, we do nothing if the matrix isn't invertible). If A is invertible, then you have to do LU decomposition and determine whether A is decomposable or not (L and U have to be both invertible).

Note that to get a unique P that matches TA's pattern, the order of row exchange matters. For example, if A(1,1) = 0 and A(2,1) = A(3,1) = 1, row exchange should be operated on row 1 and row 2 instead of row 1 and row 3, which will lead to another P (if possible).

Input Signal	Bit Width	Definition
clk	1	Clock.
rst_n	1	Asynchronous active-low reset.
in_valid	1	0 for input invalid, 1 for input valid.
in_data	1	Element of a 3-by-3 matrix A, in raster scanning order.

Output Signal	Bit Width	Definition
out_valid	1	0 for output invalid, 1 for output valid.
invertible	1	0 if A is not invertible.
decomposable	1	0 if A cannot be decomposed to LU.
out_l	3	Element of a 3-by-3 matrix L, in raster scanning order.
out_u	3	Element of a 3-by-3 matrix U, in raster scanning order.

Inputs

1. Input signals：**clk**, **rst_n, in_valid, in_data.**
2. The in_data is valid only when in_valid is high, and is delivered 9 cycles continuously.
3. All input signals will be synchronized at negative edge of the clock.

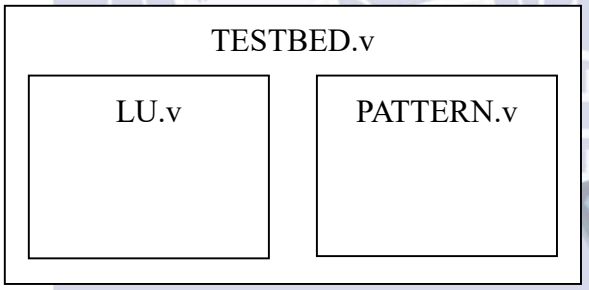
Outputs

1. You should set all your outputs to 0 when **rst_n** is low.
2. Output signals are synchronized at clock positive edge.
3. out_valid should be low after initial reset.
4. out_valid should not be raised when in_valid is high.
4. If L, U is available, out_valid should stay high for 9 cycles, otherwise, stay high for 1 cycle.
5. The TA's pattern will capture your output for checking at clock negative edge.

Specifications

1. Top module name：LU (Filename: LU.v).
2. The clock period of the design is 5ns, you can adjust it by yourself.
3. The next group of inputs will come in 1~5 cycles after your out_valid is over.
5. The synthesis result of data type can not include any LATCH.
6. After synthesis, you can check LU.area and LU.timing. The area report is valid when the slack in the end of timing report is non-negative.
7. The gate level simulation cannot include any timing violation.
8. You have to use finite state machine to control your design and implement Gaussian Elimination, if not, you will get 0 point for this exercise.

Block Diagram



Note

1. **Grading policy:**

   RTL and Gate-level simulation correctness: 70%

   Performance: 30%

   - Latency 20%
   - Area 10%

2. **Please prepare the following files:**
   - LU_train_xx.v
   - xx.txt (ex 4.0.txt)
3. **Template folders and reference commands:**

   01_RTL/ (RTL simulation) **./01_run**

02_SYN/ (Synthesis) **./01_run_dc**

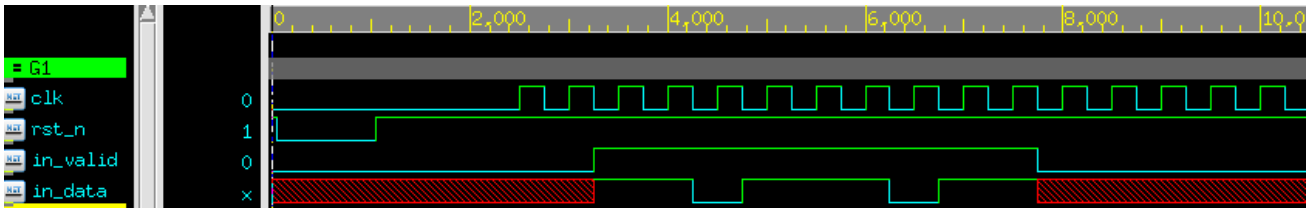
(Check the design if there's latch or not in ***syn.log*)**

(Check the design's timing in /***Report*/*LU.timing***)

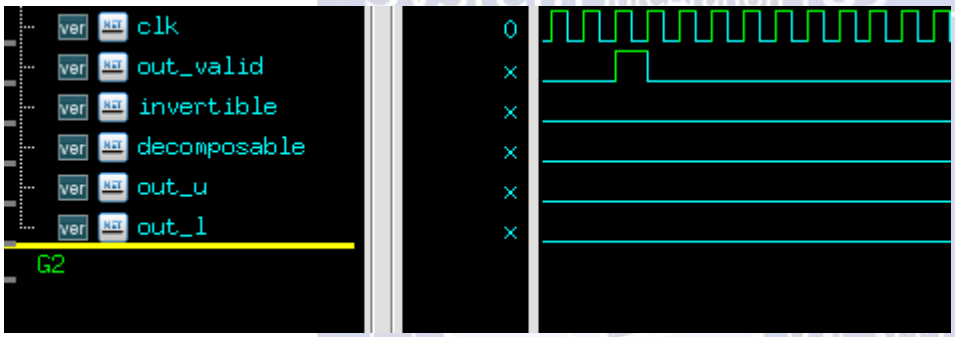
03_GATE / (Gate-level simulation) **./01_run**

Sample Waveform

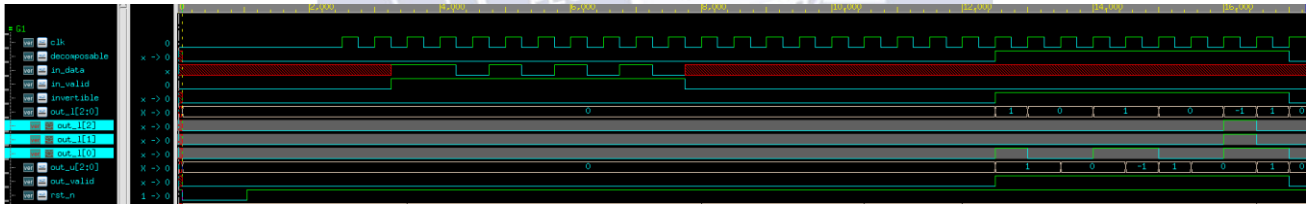
1. 9 cycles for in_valid & in_data.



2. 1 cycle if A is not invertible.



3. 9 cycles if A is both invertible and decomposable.



Reference Material

In the link, you can find useful information of how LU factorization works and why it is important.

http://www.math.iit.edu/~fass/477577_Chapter_7.pdf (P.59~P.65)