



Welcome to The Hardware Lab!

Fall 2018

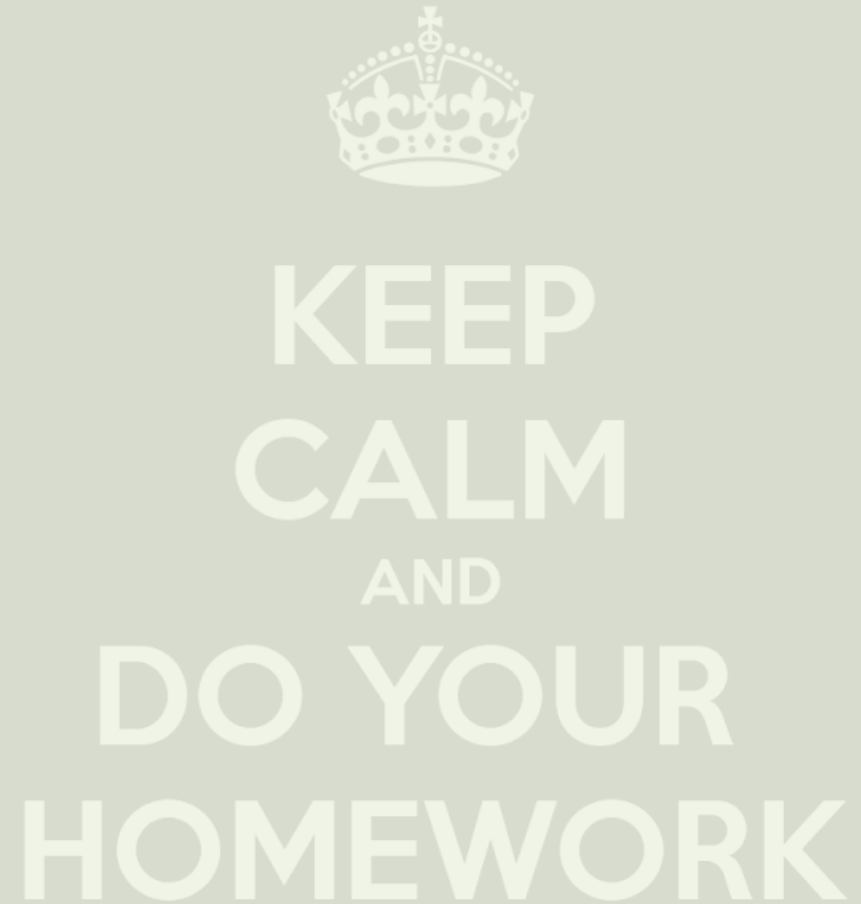
Lab 2: Advanced Gate-Level Verilog

Prof. Chun-Yi Lee

Department of Computer Science
National Tsing Hua University

Agenda

- Lab 2 Outline
- Lab 2 Basic Questions
- Lab 2 Advanced Questions



Lab 2 Outline

- Basic questions (1.5%)
 - Individual assignment
 - Due on **9/27/2018. In class.**
 - Only demonstration is necessary. Nothing to submit.
 - Please **draw the circuits of basic question 1** in your report
- Advanced questions (5%)
 - Group assignment
 - ILMS submission due on **10/4/2018. 23:59:59.**
 - Demonstration on your FPGA board (**In class**)
 - Assignment submission (**Submit to ILMS**)
 - Source codes and testbenches
 - Lab report in PDF

Lab 2 Rules

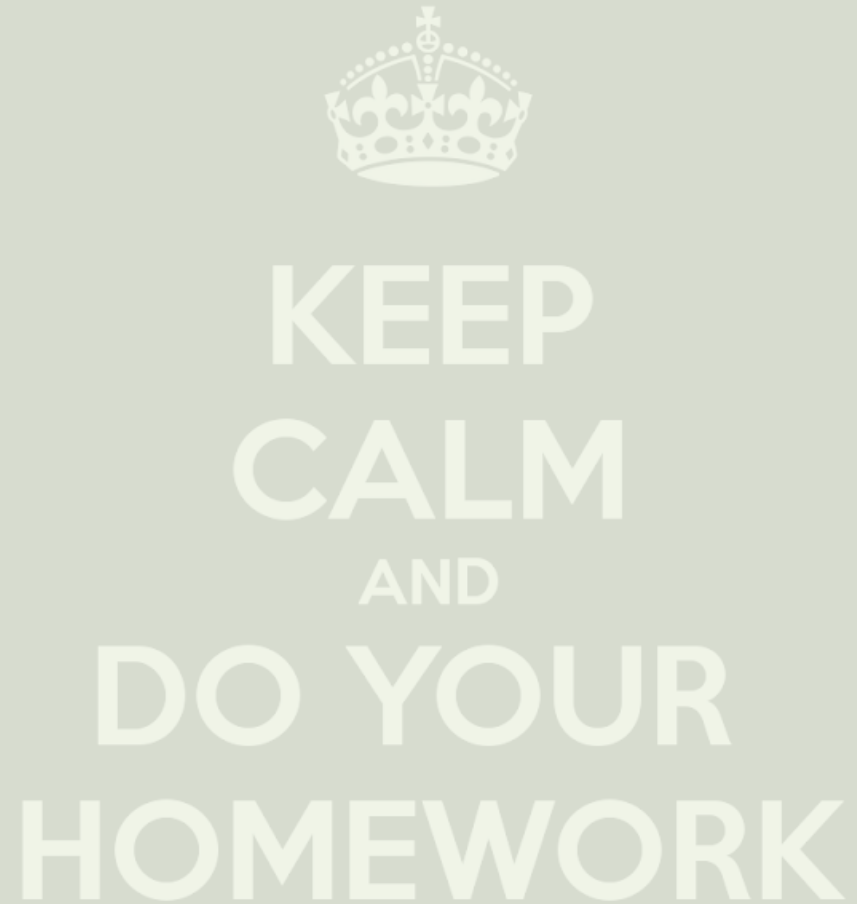
- Only gate-level description is permitted
 - Only basic logic gates are ALLOWED (AND, OR, NAND, NOR, NOT)
 - Sorry, no xor & xnor
- Please **AVOID** using
 - Continuous assignment (e.g., **assign** =, **wire** =) and conditional operators (e.g., **:** **?**)
 - Behavioral operators (e.g., **!**, **%**, **&**, *****, **+**, **/**, **<**, **>**, **^**, **|**, **~**)

Lab 2 Submission Requirements

- Source codes and testbenches
 - Please follow the templates **EXACTLY**
 - We will test your codes by TAs' testbenches
- Lab 2 report
 - Please submit your report in a single **PDF** file
 - Please **draw** the gate-level circuits of your designs
 - Please **explain** your designs in detail
 - Please **list** the contributions of each team member clearly
 - **Please explain how you test your design**
 - What you have **learned** from Lab 2

Agenda

- Lab 2 Outline
- **Lab 2 Basic Questions**
- Lab 2 Advanced Questions



Basic Questions

- Individual assignment
- Verilog questions (due on 9/27/2018. In class.)
 - (Gate Level) NOR gates only
 - (Gate Level) Latch & Flip flop
- Demonstrate your work by waveforms

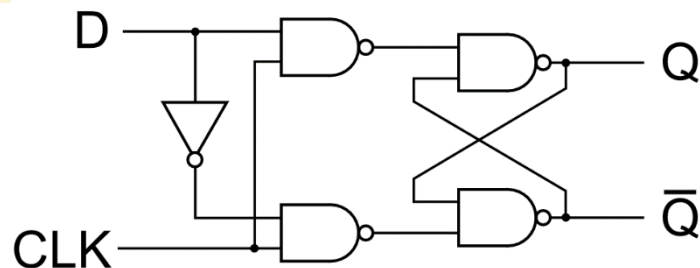
Verilog Question 1

- (Gate Level) NOR gates only
 - Use **NOR gates only** to realize the following functions
 - **NOT, NOR, AND, OR, XOR, XNOR, NAND**
 - Input/Output: A (1bit), B (1bit), Sel (3 bits), Out (1 bit)
 - Please **draw your circuits** in your report

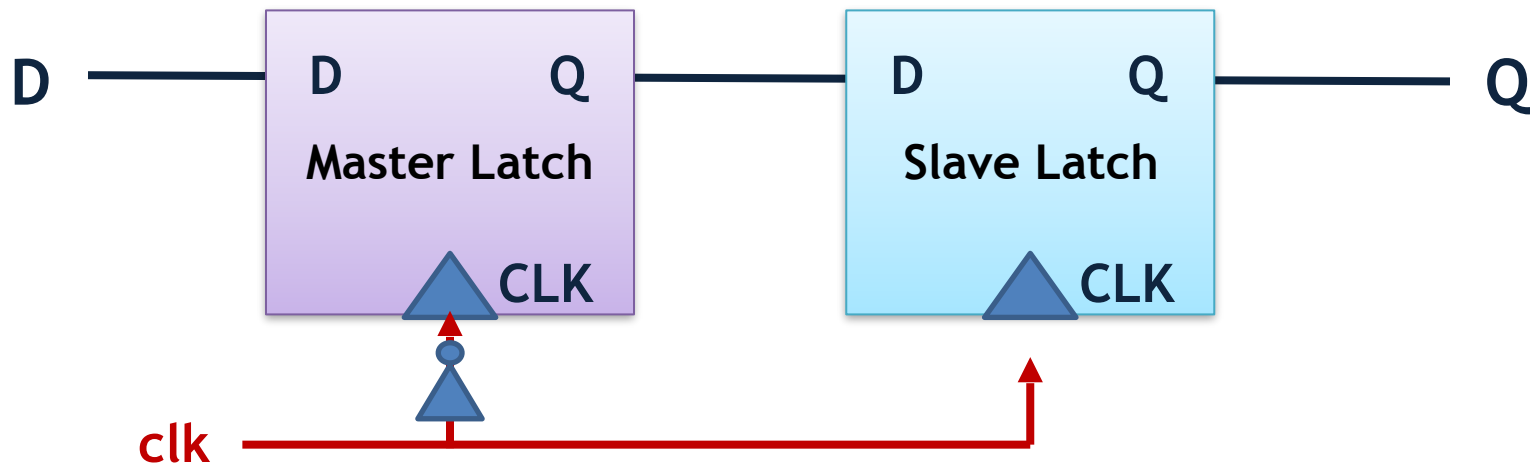
Sel [2:0]	Out
000	Out = !A
001	Out = A nor B
010	Out = A and B
011	Out = A or B
100	Out = A xor B
101	Out = A xnor B
110 & 111	Out = A nand B

Verilog Question 2

- (Gate Level) Latch & Flip flop
- Design a **latch** module as follows:



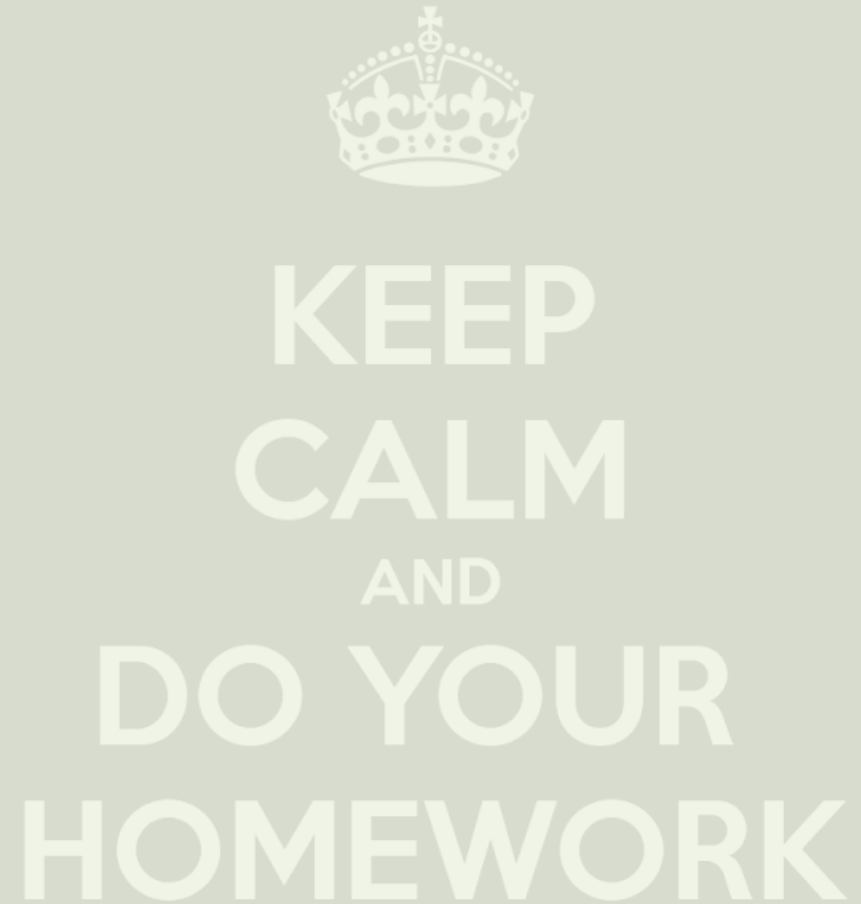
- Then design a **clk positive** edge trigger **flip-flop** module as:



- We will test your **latch** and **flip-flop** by TA's testbenches

Agenda

- Lab 2 Outline
- Lab 2 Basic Questions
- **Lab 2 Advanced Questions**



Advanced Questions

- Group assignment
- Verilog questions (due on 10/4/2018. 23:59:59.)
 - (Gate Level) Binary code to Grey code
 - (Gate Level) Multiplier
 - (Gate Level) 4-bit Carry-Lookahead (CLA) Adder
 - (Gate-level) Decode and Execute
- FPGA demonstration (due on 10/4/2018. In class.)
 - (Gate Level) 4-bit Carry-Lookahead (CLA) Adder

Verilog Question 1

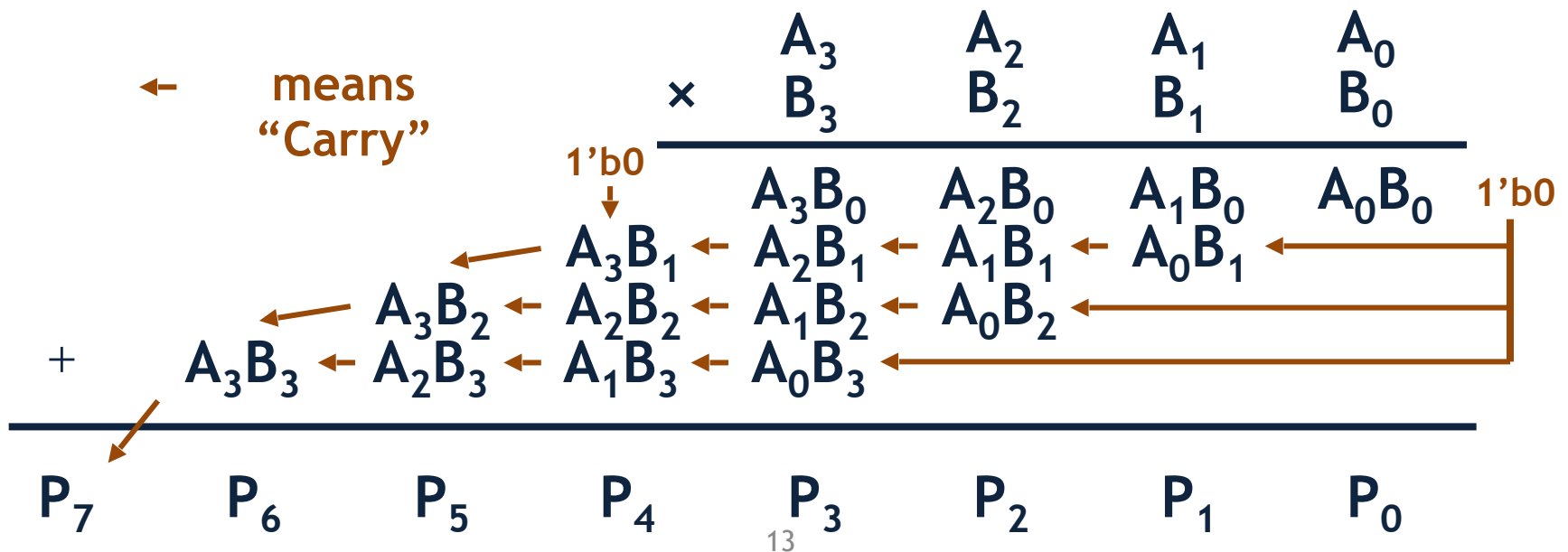
- (Gate Level) Binary code to Grey code
- $Din[3:0] = \text{Binary code}; Dout[3:0] = \text{Grey code}$

Decimal value	Binary code	Grey code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100

Decimal value	Binary code	Grey code
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

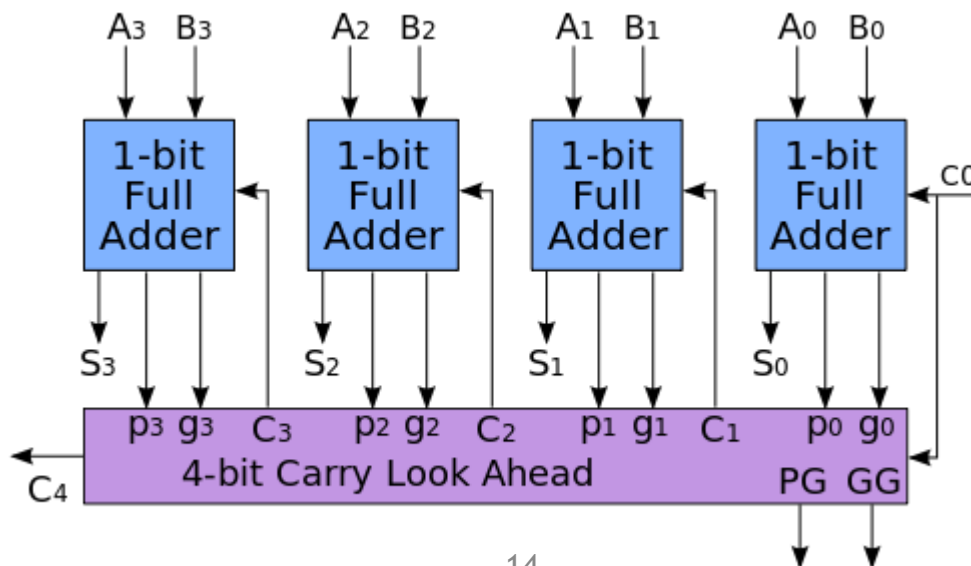
Verilog Question 2

- (Gate Level) Multiplier
 - Design a 4-bit unsigned Multiplier
 - Using your 1-bit Full Adder module in Lab 1 and logic gates
 - Please explain the how it works
 - Please draw your block diagram using **full adders** and **logic gates**
- **Inputs:** A[3:0] and B[3:0]; **Output:** P[7:0]



Verilog Question 3

- (Gate Level) 4-bit Carry-Lookahead (CLA) Adder
 - Using your 1-bit Full Adder module in Lab 1 and logic gates
 - Please explain the benefits of a Carry-Lookahead Adder
 - Please explain **the circuit of your CLA and how it works**
- Go to Wikipedia to check out the details of it
 - https://en.wikipedia.org/wiki/Carry-lookahead_adder



Verilog Question 4

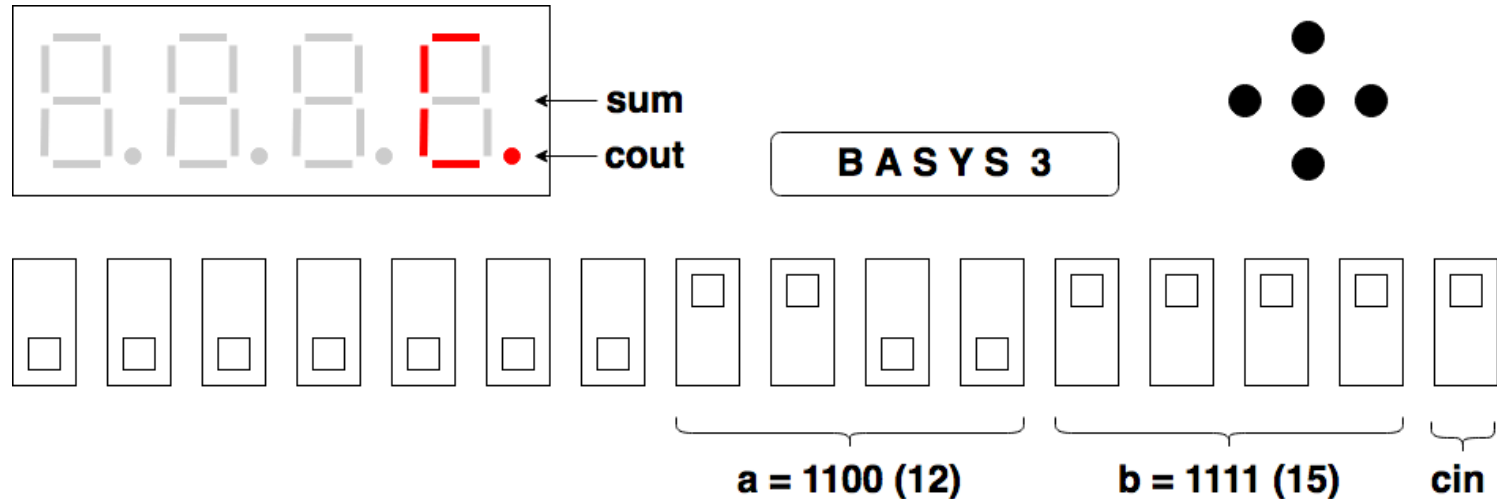
- (Gate-level) Instruction Decoding and Execution (**for unsigned numbers**)
- **Inputs:** Op_Code (3 bits), Rs (4 bits), and Rt (4 bits)
- **Output:** Rd (4 bits)
- No need to worry about overflow for **ADD, SUB, INC, RS MUL 2, MUL**

Instruction	OP_Code	Function
ADD	000	$Rd = Rs + Rt$
SUB	001	$Rd = Rs - Rt$ (hint: two's complement)
INC	010	$Rd = RS + 1'b1$
BITWISE NOR	011	$Rd = Rs$ (bitwise NOR) Rt
BITWISE NAND	100	$Rd = Rs$ (bitwise NAND) Rt
RS DIV 4	101	$Rd = Rs \gg 2$
RS MUL 2	110	$Rd = Rs \ll 1$
MUL	111	$Rd = Rs * Rt$

Advanced Questions

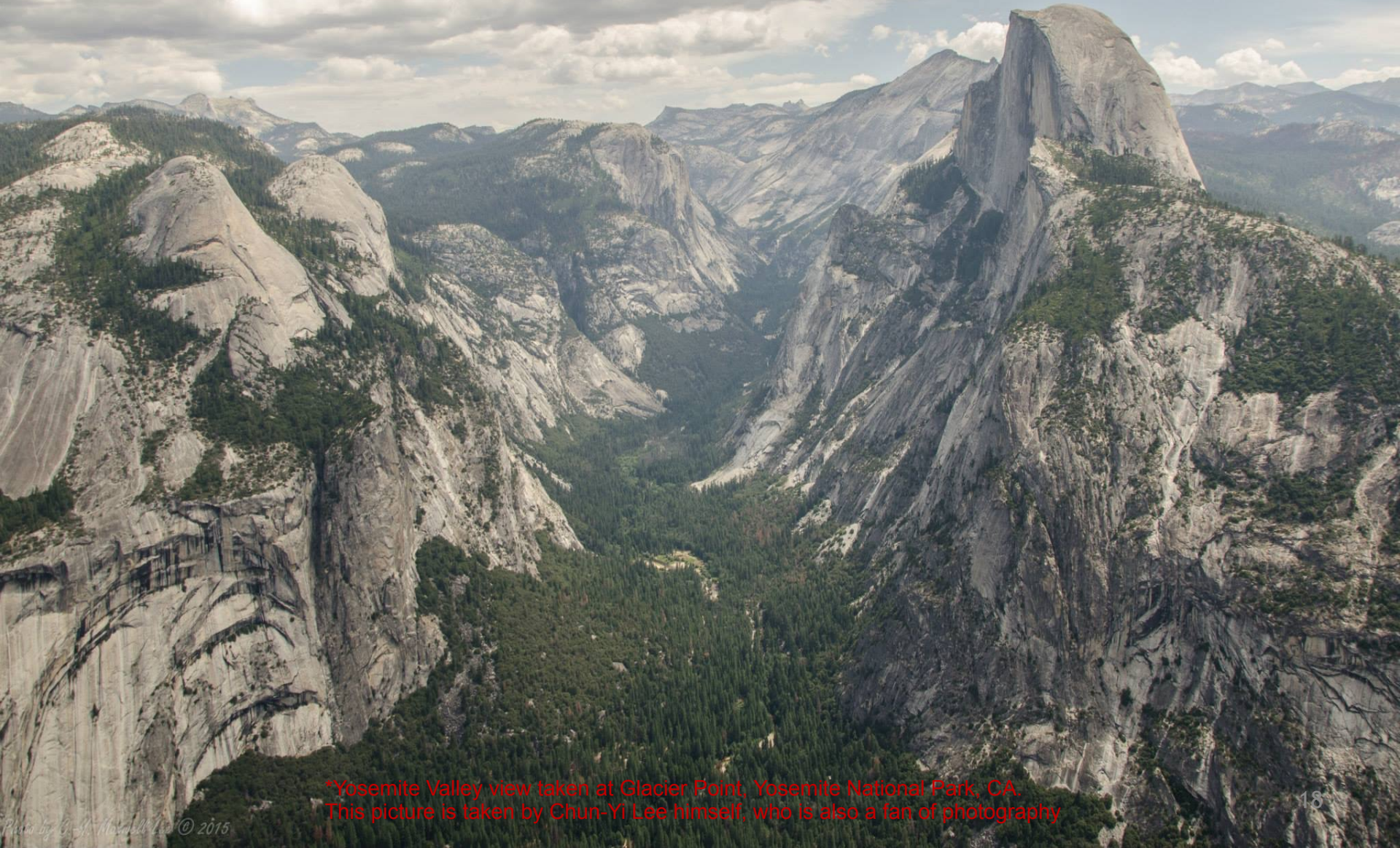
- Group assignment
- Verilog questions (due on 10/4/2018. 23:59:59.)
 - (Gate Level) Binary code to Grey code
 - (Gate Level) Multiplier
 - (Gate Level) 4-bit Carry-Lookahead (CLA) Adder
 - (Gate-level) Instruction Decoding and Execution
- FPGA demonstration (due on 10/4/2018. In class.)
 - (Gate Level) 4-bit Carry-Lookahead (CLA) Adder

FPGA Demonstration 1



- (Gate Level) 4-bit Carry-Lookahead (CLA) Adder
- Implement a carry-lookahead adder to compute **(a + b)** and represent the sum in a **single hexadecimal number**
 - Please assign your inputs/outputs as:
 - **SW[0]** stands for '**cin**', **SW[8:5]** stands for '**a**', **SW[4:1]** stands for '**b**'
 - Use the **rightmost 7-segment display** to show your **sum**
 - Use the **rightmost dot** to show your **cout**

Thank you for your attention!



*Yosemite Valley view taken at Glacier Point, Yosemite National Park, CA.
This picture is taken by Chun-Yi Lee himself, who is also a fan of photography