



CS5120 VLSI System Design, Spring 2025

# VLSI System Design

## 超大型積體電路系統設計

黃稚存

Chih-Tsun Huang

[cthuang@cs.nthu.edu.tw](mailto:cthuang@cs.nthu.edu.tw)



國立清華大學  
NATIONAL TSING HUA UNIVERSITY

資訊工程學系  
Computer Science

Lecture 00

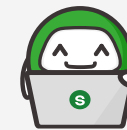
slido

Join at  
**slido.com**  
**#8279 305**



slido

Please download and install the  
Slido app on all computers you use



我希望從這堂課獲得...

① Start presenting to display the poll results on this slide.

slido

Please download and install the  
Slido app on all computers you use



## Audience Q&A

① Start presenting to display the audience questions on this slide.



# Course Resources

## ◎ EECLASS 數位學習平台:

<https://eeclass.nthu.edu.tw/course/24612>

- ◆ Announcements, lectures, assignments, forum and grading
- ◆ Video recoding on YouTube
  - As a free service
  - Not an online course
- ◆ Ask questions on the EECLASS forum
  - Course-related discussion ONLY on the forum, instead of private email
  - Personal questions to <nthucs5120@gmail.com>
- ◆ It is your responsibility to check the announcements

## ◎ Accounts for EDA tools and ADFP (Academic Design Foster Package)

- ◆ To be announced



# 聲明

- ◎ 本課程之內容 (包括但不限於教材、影片、圖片、檔案資料等)，僅供修課學生個人合理使用，非經授課教師同意，不得以任何形式轉載、重製、散布、公開播送、出版或發行本影片內容 (例如將課程內容放置公開平台上，如 Facebook, Instagram, YouTube, Twitter, Google Drive, Dropbox 等等)。如有侵權行為，需自負法律責任。



# Course Objectives

- ◎ This course covers modern perspectives on the digital VLSI system designs
  - ◆ The concept of system with hardware and software components, and their integration
  - ◆ Efficient hardware design and its methodology
  - ◆ Synthesis-based (cell-based) design flow
- ◎ Deep learning acceleration engine as a case study
  - ◆ Understanding essential deep learning operations
  - ◆ Exploiting design optimization of deep learning accelerators



# Course Outline

- ① Deep neural networks (DNNs)
  - ◆ Essential DNN components
  - ◆ Quantization
  - ◆ Dataflow
  - ◆ Tiling/mapping
  - ◆ Efficient hardware accelerators for DNNs
- ① Synthesis-based (cell-based) ASIC design flow
  - ◆ Verilog modeling
  - ◆ Synthesis flow
  - ◆ Debugging and design for testability
  - ◆ Automatic placement and routing
- ① System architectures and evaluations
  - ◆ Data communications and bandwidth
  - ◆ Memory subsystem
  - ◆ Architecture exploration and performance evaluation
  - ◆ Domain-specific architecture
  - ◆ System-on-Chip (SoC) platform





# Course Organization (1/2)

## ◎Lecture

- ◆ Class Hours: T7T8R7R8
- ◆ Classroom
  - Rm102, EECS Bldg.
  - \*323, EECS Bldg.  
(demo, hand-on labs) if necessary

## 黃稚存 (CT) Chih-Tsun Huang

- ◆ Email: [cthuang@cs.nthu.edu.tw](mailto:cthuang@cs.nthu.edu.tw)
- ◆ <http://nthucad.cs.nthu.edu.tw/~cthuang/>
- ◆ Office Hours: after classes





# Course Organization (2/2)

## ⊙ Homework assignments

- ◆ DNN training, quantization, inference modeling (Python/C)
- ◆ Accelerator engine design and synthesis (front-end design, Verilog)
- ◆ SoC platform + accelerator engine (Python/C/Verilog)
- ◆ Automatic placement and routing (back-end design)

## ⊙ Final project



# Prerequisites

- ⊙ EECS1010 Logic Design
- ⊙ EECS2070 Logic Design Lab or CS2104 Hardware Design Lab
- ⊙ CS4100/EE3450 Computer Architecture
- ⊙ Basic understanding of logic design and computer architecture
- ⊙ Basic Verilog, C/C++ and Python skills



# Back to the Lower Level: A Simple Verilog Example

```
module example (  
    input wire clk,  
    input wire rst_n,  
    output reg [7:0] num  
);  
  
    wire [7:0] next_num;  
  
    always @(posedge clk, negedge rst_n) begin  
        if (rst_n == 0)  
            num <= 0;  
        else  
            num <= next_num;  
    end  
  
    assign next_num = num + 1;  
  
endmodule
```



# Finite-State Machine Example

```
parameter S0 = 2'b00;
parameter S1 = 2'b01;
parameter S2 = 2'b10;
reg [2:0] state, next_state;
always @(posedge clock, negedge rst_n)
begin
    if (rst_n == 1'b0)
        state <= S0;
    else
        state <= next_state;
end
always @* begin
    next_state = S0;
    case (state)
        S0: begin
            if (in == 1)
                next_state = S1;
            else
                next_state = S0;
        end
    end
```

State Update  
(Sequential)

State Transition  
(Combinational)

```
S1: begin
    if (in == 1)
        next_state = S1;
    else
        next_state = S2;
end
S2: begin
    if (in == 1)
        next_state = S1;
    else
        next_state = S0;
    end
endcase // case end
end //always end

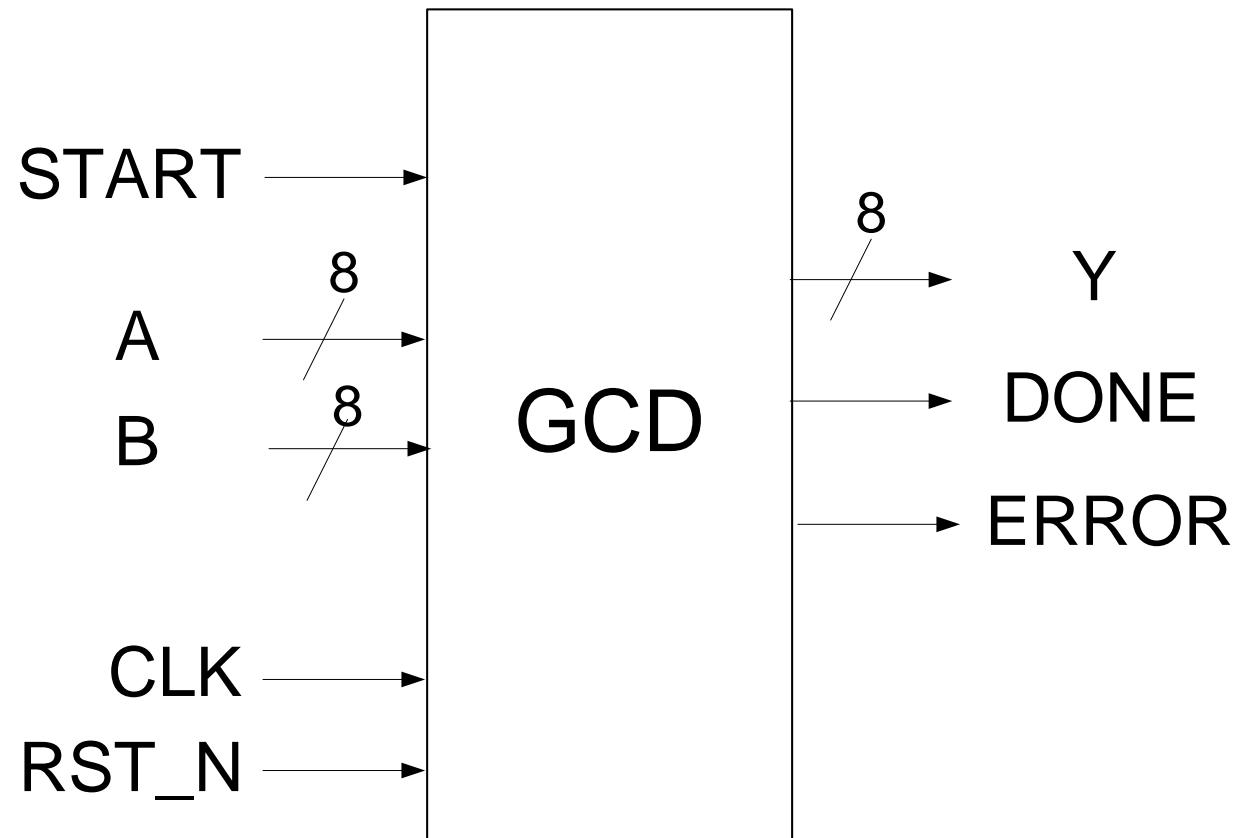
assign out = state == S2 ? 1 : 0;
```

Outputs (Combinational)



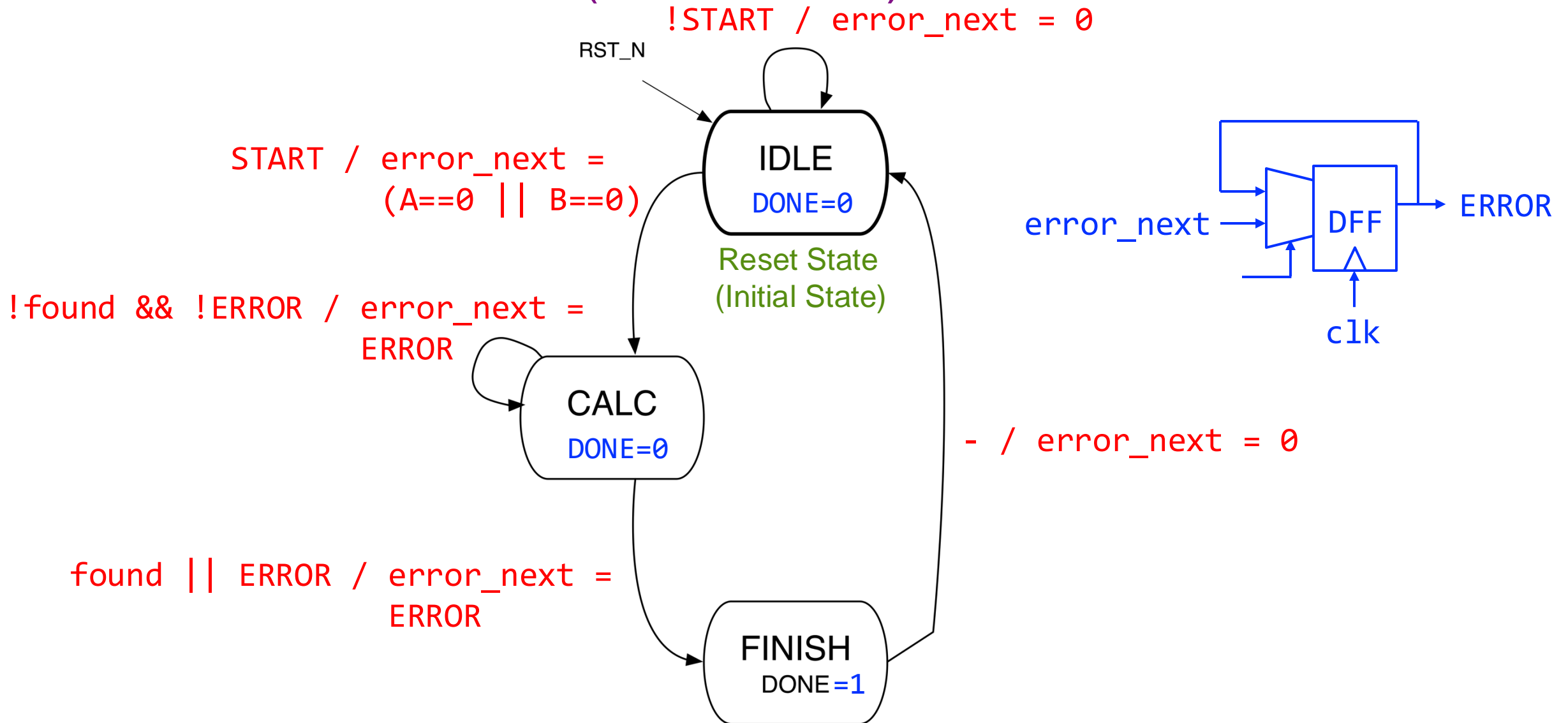
# Design Exercise:

## Top-level Block Diagram and Primary IOs



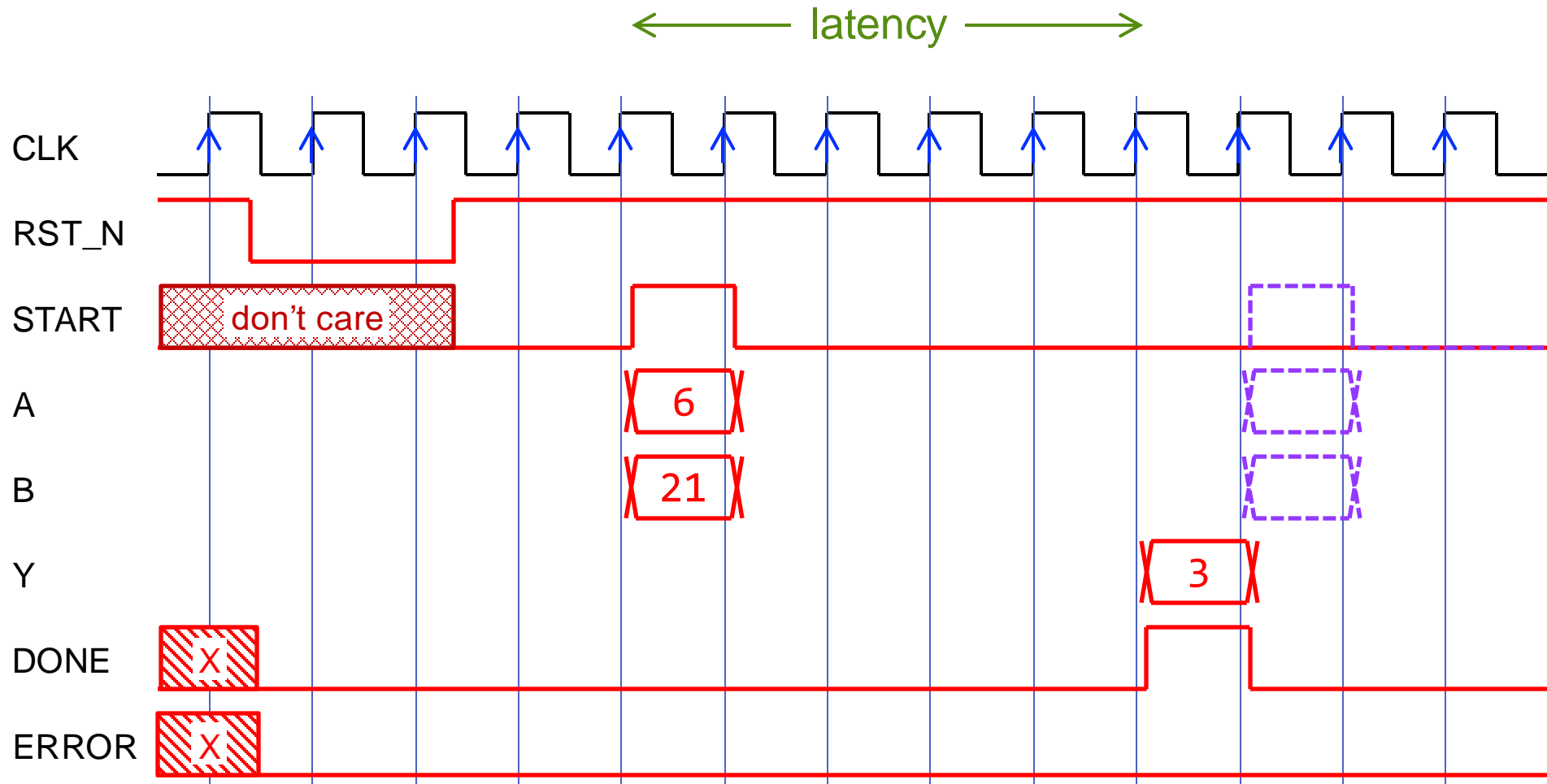


# Design Exercise: Finite-State Machine (Control Unit)





# Design Exercise: Timing Diagram







# Design Exercise: Example of Testbench

```
initial begin
    clk = 1;
    rst_n = 1;
    #(cyc);
    #(delay) rst_n = 0;
    #(cyc*4) rst_n = 1;
    #(cyc*2);
    #(cyc) nop;
    #(cyc) load; data_in(8'd6, 8'd21);
    #(cyc) nop;
    @(posedge done);
    // [HW] apply more patterns to cover
    // different conditions

    #(cyc) nop;
    #(cyc*8);
    $finish;
end
```

start = 0;

start = 1;

a = 8'd6;  
b = 8'd21;

```
// using tasks to improve the
// readability
task nop;
    begin
        start = 0;
    end
endtask
task load;
    begin
        start = 1;
    end
endtask
task data_in;
    input [7:0] data1, data2;
    begin
        a = data1;
        b = data2;
    end
endtask

endmodule
```



# CT Verilog Series

- ①00 Preface
- ①01 Fundamental Concepts for Verilog HDL
- ①02 My Very First Verilog Coding
- ①03 Verilog Overview Part 1 – Structural Modeling
- ①04 Verilog Overview Part 2 – Dataflow Modeling
- ①05 Verilog Overview Part 3 – Behavioral Modeling
- ①06 Verilog Overview Part 4 – Combinational Blocks
- ①07 Verilog Overview Part 5 – Sequential Blocks
- ①08 Finite-State Machines
- ①09 Design Example

# CT Verilog Series for Your Reference



## CT Verilog Series

錄製這系列的 Verilog 教材是為了讓同學能參考複習。

- 若對這系列內容有任何問題，可在課堂上提出，我們會視同學反應，針對特定主題再詳細討論。
- 對 Verilog 熟悉的同學也請很快複習過這系列，以便與我們對 Verilog 的認知同步，提高上課效果。

影片及教材之內容，限於同學個人使用，請勿轉載或散佈。

1.	CT Verilog Series 00 - Preface	-	閱讀 > 5 分鐘	70%	⋮
2.	CT Verilog Series 01 - Fundamental Concepts for Verilog HDL	-	閱讀 > 35 分鐘	74%	⋮
3.	CT Verilog Series 02 - My Very First Verilog Coding	-	閱讀 > 63 分鐘	76%	⋮
4.	CT Verilog Series 03 - Verilog Overview Part 1 - Structural Modeling	-	閱讀 > 35 分鐘	76%	⋮
5.	CT Verilog Series 04 - Verilog Overview Part 2 - Dataflow Modeling	-	閱讀 > 27 分鐘	74%	⋮
6.	CT Verilog Series 05 - Verilog Overview Part 3 - Behavioral Modeling	-	閱讀 > 45 分鐘	68%	⋮
7.	CT Verilog Series 06 - Verilog Overview Part 4 - Combinational Blocks	-	閱讀 > 30 分鐘	71%	⋮
8.	CT Verilog Series 07 - Verilog Overview Part 5 - Sequential Blocks	-	閱讀 > 31 分鐘	66%	⋮
9.	CT Verilog Series 08 - Finite-State Machines	-	閱讀 > 30 分鐘	77%	⋮



# Reference Materials

## ⦿ DNN hardware

- ◆ Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, Joel S. Emer, ***Efficient Processing of Deep Neural Networks – Synthesis Lectures on Computer Architecture***, Morgan & Claypool Publishers, 2020.
- ◆ Eyeriss tutorial: <http://eyeriss.mit.edu>

## ⦿ Verilog

- ◆ HDLBits Verilog Practice: [https://hdlbits.01xz.net/wiki/Main\\_Page](https://hdlbits.01xz.net/wiki/Main_Page)
- ◆ Dr. D J Greaves' (Cambridge University) guide to Verilog: <http://www.cl.cam.ac.uk/%7Edjg11/teaching/learners.pdf>

## ⦿ Deep learning

- ◆ Ian Goodfellow and Yoshua Bengio and Aaron Courville, ***Deep Learning***, MIT Press, 2016: <https://www.deeplearningbook.org>
- ◆ Dive into Deep Learning: <https://d2l.ai/>
- ◆ **PyTorch tutorial**: <https://pytorch.org/tutorials/>



# Important Dates (subject to change)

⦿ 16+1 class weeks

⦿ No class:

- ◆ April 3: Children's Day

⦿ Offline video lectures:

- ◆ April 29, May 1 (11<sup>th</sup> Week)

⦿ Final project due

- ◆ June 10 (17<sup>th</sup> Week)



# Evaluation (subject to change)

⦿ 0% Honor code agreement (essential to get course credits)

⦿ 60% Homework assignments

- ◆ Problem assignments with reports  
(may require face-to-face demo/QA for grading)
- ◆ Class participation

⦿ 40% Final project

⦿ It is a graduate-school-level course to learn the knowledge and skills for your future research studies



# Rules of Grading

- No overdues allowed. No absences from exams and demos. The rule is strictly applied.  
Lab/作業/期末專題不能遲交；考試、Demo 不能缺席
- Do your own homework, but discussing it with classmates or the TA is highly encouraging.  
鼓勵同學提問討論，但自己的作業自己寫
- No grade will be given when failing to meet homework requirements.  
不符合規定的作業不予計分 (繳交格式、檔名不符、未依規定Demo等等)
- Any **academic misconduct** (dishonesty, cheating, or plagiarism) is unacceptable.  
不容許任何學術不當行為 (不誠實、作弊、抄襲、協助或默許上述行為等等)
  - ◆ You will get **NEGATIVE 100% points for academic misconduct**  
若有不當行為，該次成績負100分
  - ◆ Apply to any turn-in materials (e.g., homework/lab assignment, reading review, quiz, exam, project, proposal, roll call, etc.)  
適用任何課堂繳交或評分的項目
  - ◆ Any misconduct in an exam or project will definitely fail the course and be reported to the University Office  
涉及考試、專題或情節嚴重者學期成績會不及格，並且送交學校調查懲處
  - ◆ Never show/share your code to others privately or publicly (e.g., on the forum or public repositories, e.g., GitHub)  
禁止分享、公開原始碼
  - ◆ We will apply plagiarism checking. Source codes with high similarity will be identified as plagiarism  
經軟體及人工比對，高相似度的原始碼若無法合理解釋，會被視為抄襲
- **No right to claim bonus if any violation of the above conditions.**  
違反上述規定，不予考慮期末調分或加分





# CT Course Honor Code (1/2)

- To maintain the integrity of our courses, and protect the value of course credits, all students should comply with our honor code and copyright:
  - ◆ Any academic misconduct (dishonesty, cheating, or plagiarism) is unacceptable.
  - ◆ You must submit your own original work for any turn-in materials (i.e., homework assignments, reading reviews, quizzes, exams, labs, projects, proposals, roll call, etc.) in the course.
    - Any work you submit authored by another individual or (AI) tool is considered plagiarism.
  - ◆ You must not share your solutions to any turn-in materials with anyone other than the instructor and teaching assistants.
    - This includes posting solutions publicly on the course forum, GitHub, or any other code repository.
    - The instructor and teaching assistants will take action if you are found doing this.
  - ◆ You must not re-distribute slides, videos, or lecture materials.
  - ◆ You must report suspected violations.

Ref: DeepLearning.AI Honor Code





# CT Course Honor Code (2/2)

- ⦿ Any academic misconduct (dishonesty, cheating, or plagiarism) is unacceptable.
  - ◆ Also, if you knowingly aid in academic misconduct, you are guilty.
  - ◆ We will definitely take action against any academic misconduct.
- ⦿ Some specific examples are listed as follows:
  - ◆ Take someone else's work or ideas and pass them off as your own without giving credit to the original owner.
    - Turn in someone else's work as if it were your own.
    - Read others' code and then reproduce it.
    - Ask others (humans or machines) to do the (partial or entire) assignment or project for you.
  - ◆ Falsifying data or deceitfully manipulating experiment/simulation results.
  - ◆ Other dishonesty or cheating for any turn-in materials in the course.

Ref: DeepLearning.AI Honor Code



# 推薦同意書

## 2025「超大型積體電路系統設計」修課推薦書

本課程目前採全加簽方式修課，為確保修課人數適當、維持課程品質，並避免同學產生修課期望的落差，請同學務必確認以下事項：

1. 本課程需要投入心力與時間，同學應能自行平衡修課與研究工作的安排。
2. 授課著重關鍵設計觀念的介紹，期望同學能活用；作業的設計則旨在讓同學主動應用觀念並探索實作細節。課程不會手把手指導作業完成方式，同學應在老師與助教適度的提示協助下，培養獨立解決問題的能力。
3. 任何抄襲或學術不誠實行為（包括作業與專題），將導致學期成績不及格，並通報指導教授，同時將依校規處分。

請同學確認並簽署後，取得指導教授簽名同意並推薦其修課。謝謝。

學生姓名：\_\_\_\_\_ (簽名) 學號：\_\_\_\_\_

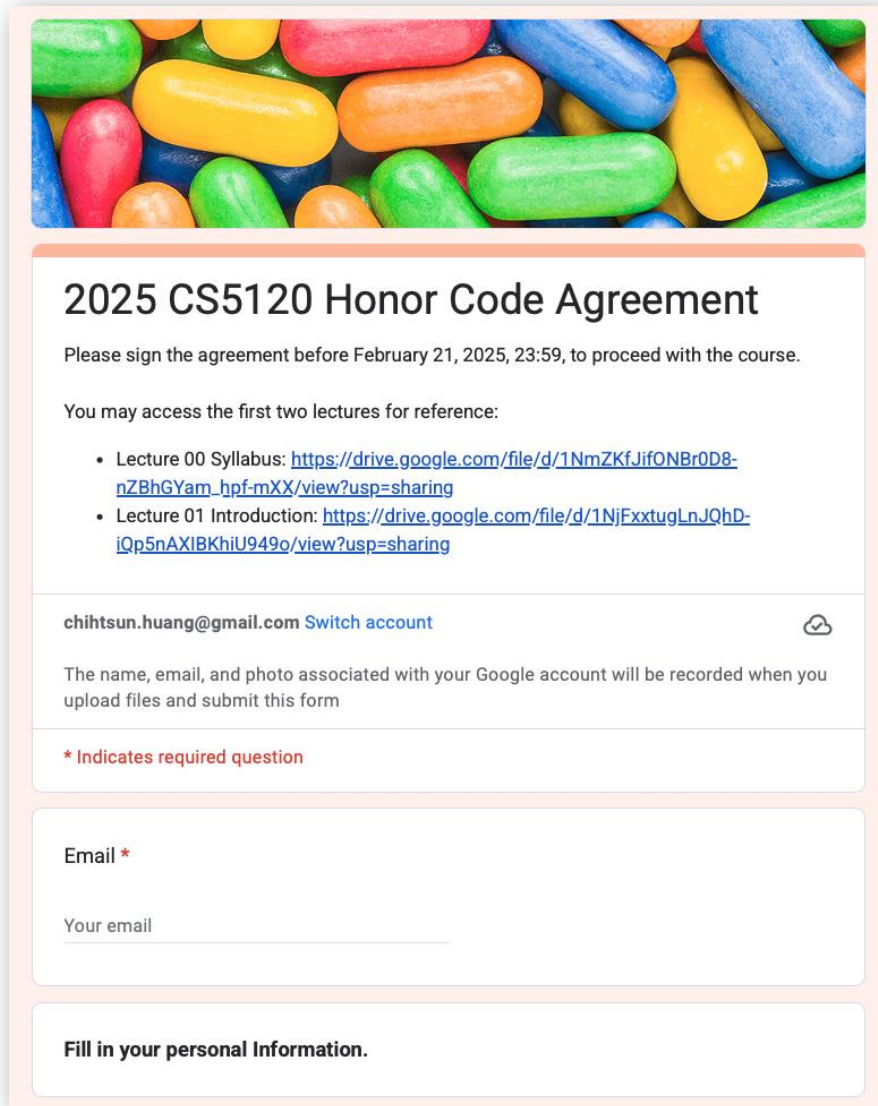
指導教授：\_\_\_\_\_ (簽名)



# Honor Code Agreement

**Due on 2025/2/21 23:59**

<https://forms.gle/UEfqux9hfHedrcNr6>



**2025 CS5120 Honor Code Agreement**

Please sign the agreement before February 21, 2025, 23:59, to proceed with the course.

You may access the first two lectures for reference:

- Lecture 00 Syllabus: [https://drive.google.com/file/d/1NmZKfJifONBr0D8-nZBhGYam\\_hpf-mXX/view?usp=sharing](https://drive.google.com/file/d/1NmZKfJifONBr0D8-nZBhGYam_hpf-mXX/view?usp=sharing)
- Lecture 01 Introduction: <https://drive.google.com/file/d/1NjFxtugLnJQhD-iQp5nAXiBKhiU949o/view?usp=sharing>

chihtsun.huang@gmail.com [Switch account](#)

The name, email, and photo associated with your Google account will be recorded when you upload files and submit this form

\* Indicates required question

Email \*

Your email

Fill in your personal information.





So, that's it!



- » Enjoy VLSI designing!
- » Questions or Comments?