香港中文大學
The Chinese University of Hong Kong

CENG3430 Rapid Prototyping of Digital Systems

Lecture 00: Course Information

Ming-Chang YANG

mcyang@cse.cuhk.edu.hk

CENG3430 Course Information

- **CENG3430 Rapid Prototyping of Digital Systems**
- **Course Time and Place**
  - **Lecture** (*2)
    - MON 16:30~18:15 (@ ERB 404)
  - **Lab** (*2)
    - TUE 16:30~18:15 (@ SHB 102)
- **Course Website**
  - http://www.cse.cuhk.edu.hk/~mcyang/ceng3430/2019S/ceng3430.html
  - https://blackboard.cuhk.edu.hk/

Course Instructor & TAs

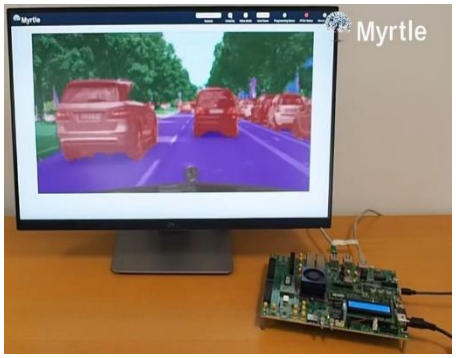
- **Course Instructor**
  - Prof. Ming-Chang YANG (楊明昌)
    - Office: SHB 906 (3943-8405)
    - Office Hours: TUE 14:00~16:00
    - Email: mcyang@cse.cuhk.edu.hk
- **Teaching Assistants**
  - Zhiliang ZENG (曾志良)
    - Office: SHB 902
    - Office Hours: FRI 15:30~17:30
    - Email: zlzeng@cse.cuhk.edu.hk
  - TBA (TBA)
    - Office:
    - Office Hours:
    - Email:

Overview: What you will learn

- **Software**: Hardware Description Languages (HDL)
  - VHDL and Verilog
- **Hardware**: Field Programmable Gate Array (FPGA)
  - The hardware can be reprogrammable.
- Techniques for Building a Digital System
  - Building blocks of a processor
  - Finite state machines
- ***Advanced Techniques for Rapid Prototyping of Digital Systems
  - Intellectual property (IP) block design
  - Operating system (e.g., Linux) on Zynq ZedBoard
  - Vivado high-level synthesis (HLS)

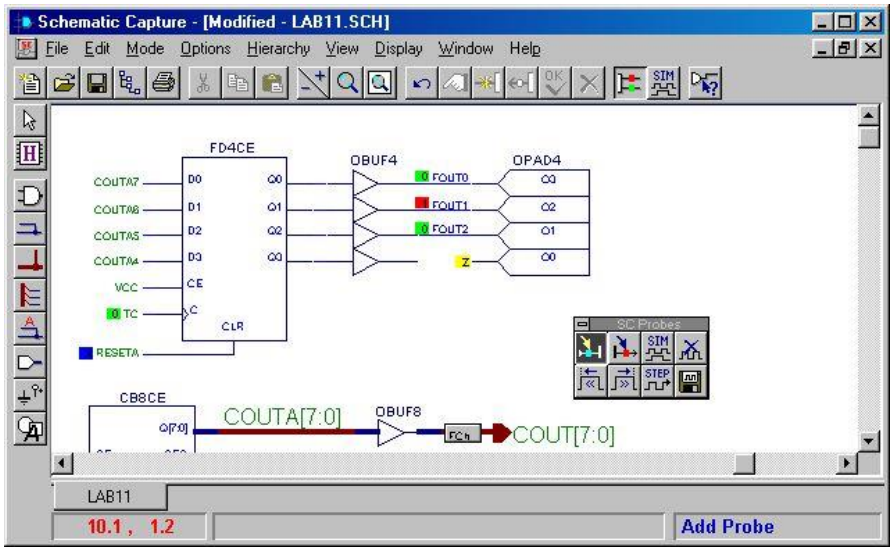
Examples of Digital System Designs

- Mass Products
  - Media players
  - Mobile phones
- Novel Products
  - Wearable devices
  - Robots
- Research
  - Real time edge detection
  - Deep learning acceleration



Methods for Digital System Designs

- **Schematic**
  - Complicated
  - Suitable for top level design to merge modules
    - Like data flow block diagram



- **Language**
  - VHDL (**V**ery-High-Speed-Integrated-Circuits **H**ardware **D**escription **L**anguage)
    - Each module in the schematic can be implemented by VHDL
  - Verilog

Ex: VHDL AND-Gate Program

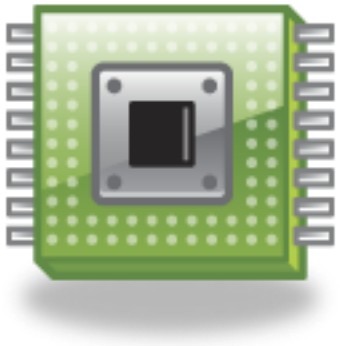
```
1 entity and2 is
2 port (a,b: in std_logic;
3         c: out std_logic);
4 end and2
5 architecture and2_arch of and2
6 begin
7    c <= a and b;
8 end and2_arch
```

What is HDL used for?

Ex: VHDL AND-Gate Program

```
1 entity and2 is
2 port (a,b: in std_logic;
3         c: out std_logic);
4 end and2
5 architecture and2_arch of and2
6 begin
7    c <= a and b;
8 end and2_arch
```

Write HDL code, then it will generate the hardware chip automatically

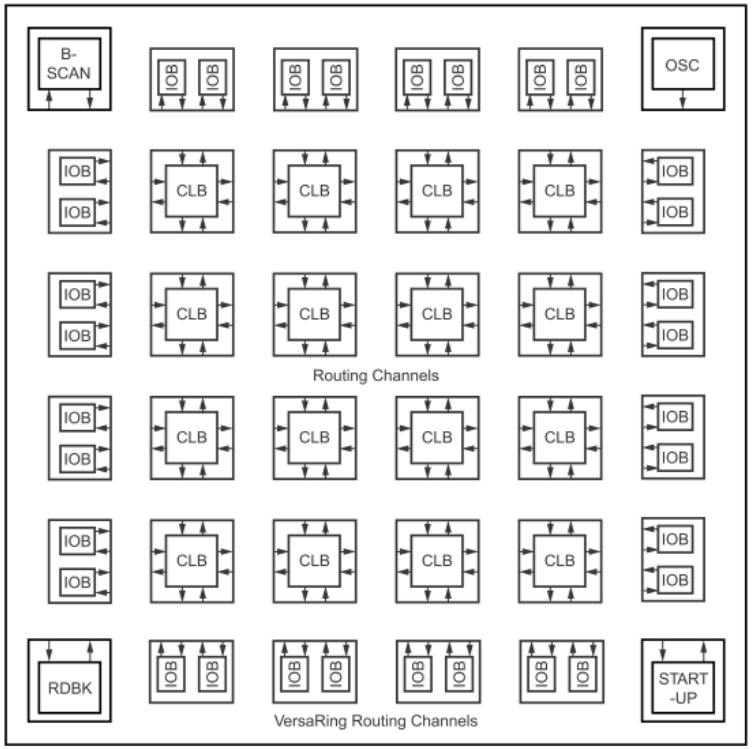


- Design Microprocessors (e.g., ARM processor)
- Design Products (an industrial standard)
  - Robots controllers, media players, games, mobile phones, image processing, computer vision, super computing, etc.
- Design Your NEW Ideas
  - Start your business!

What is FPGA?

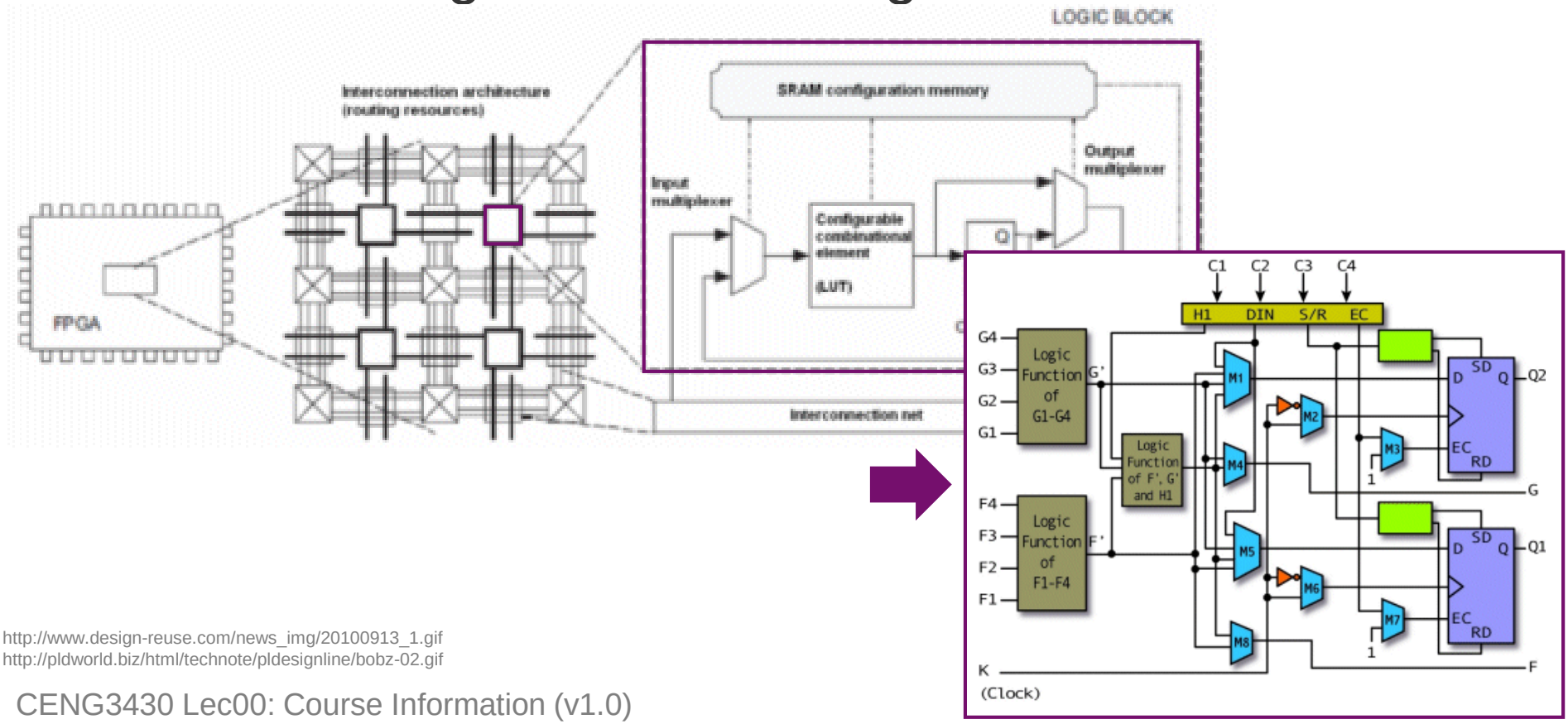
- FPGA (Field Programmable Gate Array)
  - The hardware can be reprogrammable.
  - Designs can be changed rapidly and easily.
  - No additional hardware manufacturing cost is needed.

- What is inside an FPGA?
  - **Input/Output Block (IOB)**
    - Input/output of FPGA
  - **Configurable Logic Block (CLB)**
    - Static RAM based
    - Changed for required functions
  - **Programmable Interconnects**
    - Interconnect IOBs and CLBs



Inside a CLB of FPGA

- The Configurable Logic Block (CLB) is a fixed design, but you can change the logic function inside it.
  - By reprogramming the bits in the logic function lookup table.
- This will change the overall logic function of the CLB.



http://www.design-reuse.com/news_img/20100913_1.gif
http://pldworld.biz/html/technote/pldesignline/bobz-02.gif

Work Flow of Digital System Designs

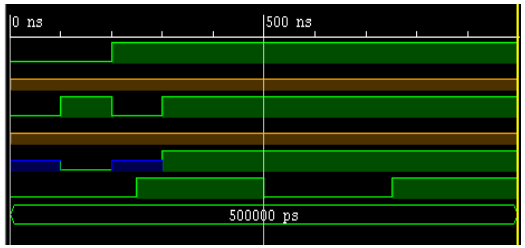
Idea Generation

Drafting on Paper

Designing Chip (VHDL)

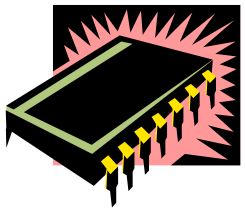
Ex: VHDL AND-Gate Program

```
1 entity and2 is
2 port (a,b: in std_logic;
3         c: out std_logic);
4 end and2
5 architecture arch of and2
6 begin
7    c <= a and b;
8 end and2_arch
```



Testing (FPGA)

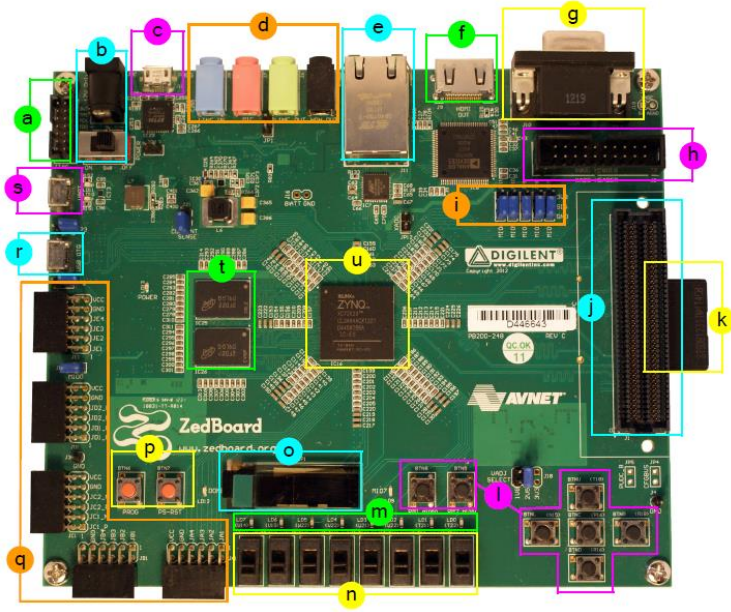
Manufacturing
Production Line Design



Quality Control

What we are gonna use in our lab

- **Software**: VHDL
  - **V**ery-High-Speed-Integrated-Circuits (VHSIC) **H**ardware **D**escription **L**anguage
- **Hardware**: Zynq ZedBoard
  - Dual-core ARM Cortex-A9, with
  - Traditional Field Programmable Gate Array (FPGA)



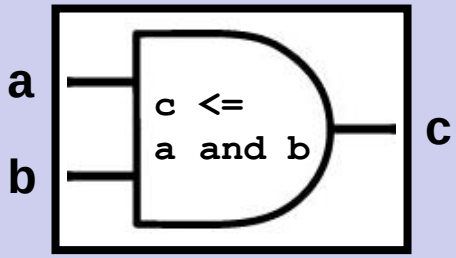
a Xilinx JTAG connector
b Power input and switch
c USB-JTAG (programming)
d Audio ports
e Ethernet port
f HDMI port (output)
g VGA port
h XADC header port
i Configuration jumpers
j FMC connector
k SD card (underside)
l User push buttons
m LEDs
n Switches
o OLED display
p Prog & reset push buttons
q 5 x Pmod connector ports
r USB-OTG peripheral port
s USB-UART port
t DDR3 memory
u Zynq device (+ heatsink)

Software: VHDL

- **An Example: AND-Gate in VHDL**

Entity Declaration: Define I/Os

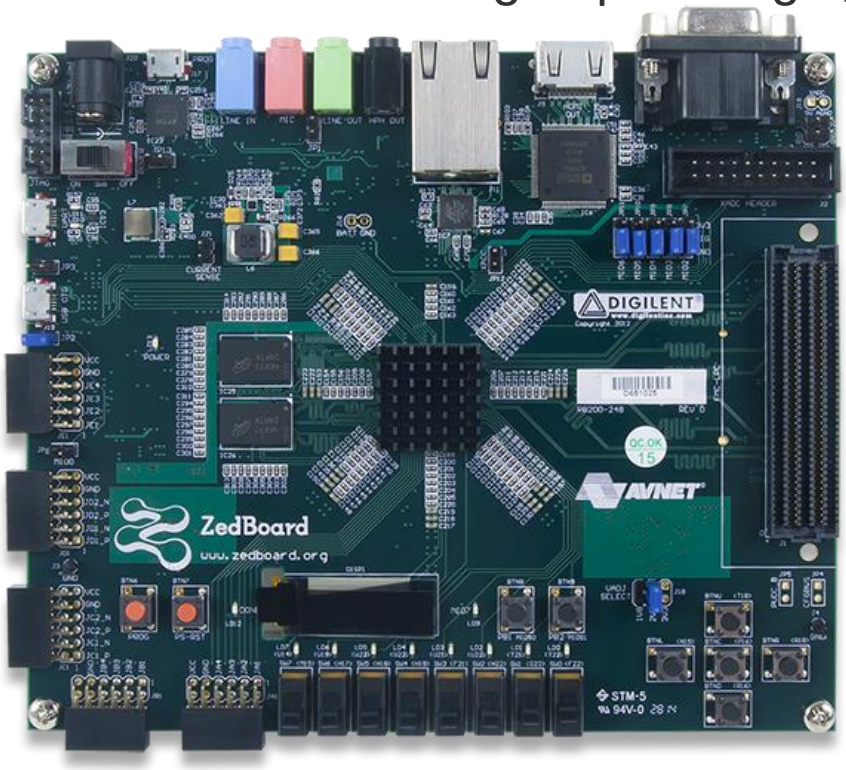
```
1 entity and2 is
2 port (a,b: in std_logic;
3           c: out std_logic);
4 end and2
5 architecture and2_arch of and2
6 begin
7     c <= a and b;
8 end and2_arch
```

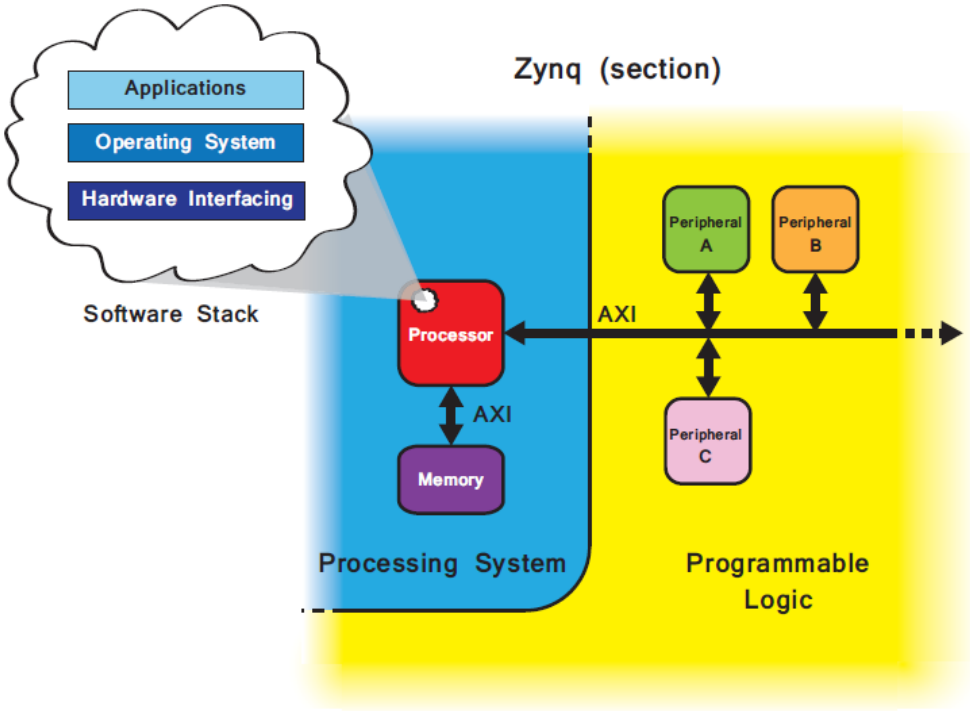


Architecture Body: Define functions

Hardware: Zynq ZedBoard

- Zynq ZedBoard combines
  - Processing System (PS): Dual-core ARM Cortex-A9 CPU
    - Supports software routines and/or operating systems
  - Programmable Logical (PL): Equivalent to trad. FPGA
    - Ideal for high-speed logic, arithmetic and data flow subsystems





Course Schedule *(tentative)*

W	Date	Lecture	Lab
1	Jan. 7, 8	Lec00: Course Information	No Lab
2	Jan. 14, 15	Lec01: Introduction to VHDL	Lab01: Intro. to Vivado & SW Simulation
3	Jan. 21, 22	Lec02: Introduction to ZedBoard	Lab02: First VHDL Program on ZedBoard
4	Jan. 28, 29	Lec03: Architecture Body	Lab03-1: Tri-state Logic (Data Flow)
5	Feb. 4, 5	Lunar New Year Vacation (No Lecture)	Lunar New Year Vacation (No Lab)
6	Feb. 11, 12	Lec03: Architecture Body	Lab03-2: 4-to-1 Multiplexer (Structural)
7	Feb. 18, 19	Lec04: Building Blocks of a Processor	Lab04: Serial-in-parallel-out Shift Register
8	Feb. 25, 26	Lec05: Finite State Machine	Lab05-1: Driving Seven Segment Display
9	Mar. 4, 5	Lec05: Finite State Machine	Lab05-2: Countdown Counter
10	Mar. 11, 12	Lec06*: IP Block Design	Lab06: Software Stopwatch
11	Mar. 18, 19	Lec07*: Linux on Zynq	Lab07: Linux GPIO + LED
12	Mar. 25, 26	Lec08*: High-level Synthesis	Lab08: HLS Exercise
13	Apr. 1, 2	Reading Week (No Lecture)	Reading Week (No Lab)
14	Apr. 8, 9	Lec09: Introduction to Verilog	Lab09: Verilog Exercise
15	Apr. 15, 16	Final Project Demo (Peer Assessment)	Final Project Demo (Peer Assessment)

Course Assessment

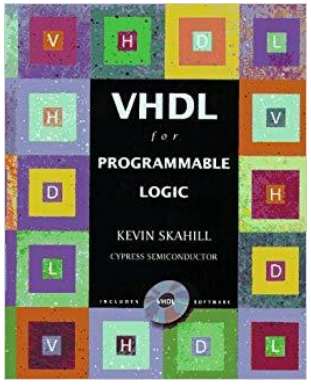
- Grading *(subject to changes)*
  - Class Exercises 10%
  - Laboratory Exercises 25%
  - Final Project 40% (New Policy: Peer Assessment)
  - Final Exam 25%
  - *Bonus* *5%* (How to get? Q&A, Best Project)
- Notes
  - Lab. exercises and final project: two students in a group.
  - Late submission per day is subject to 10% of penalty.
  - A student must attend at least 80% of lectures in order to gain all class attendance/exercise credits.

Final Project Examples (2017 & 2018)

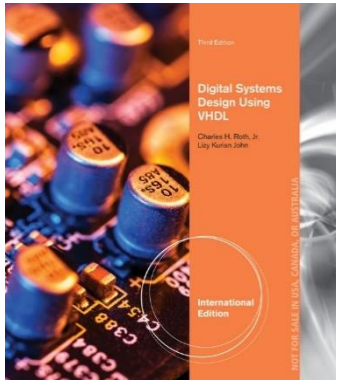
- Piano: https://youtu.be/_VH3fUazEEI?t=87
- Music Player: https://youtu.be/dEdnp1Tni9c?t=27
- Bullhorn: https://youtu.be/dtQ88yL0FUM?t=26
- Wash Machine: https://youtu.be/z7C8dXn9EQ0?t=10
- Thermometer: https://youtu.be/i0swDnATRt4?t=41
- Space Invader: https://youtu.be/2wEG-U8DNak?t=72
- Tetris: https://youtu.be/JyEU1YbYMrc?t=11
- Snake: https://youtu.be/dFdr0KqXw7Q?t=66
- Car: https://youtu.be/FDbSyYKHYes?t=28
- Sonar: https://youtu.be/DiLjDbkbejs?t=180

References

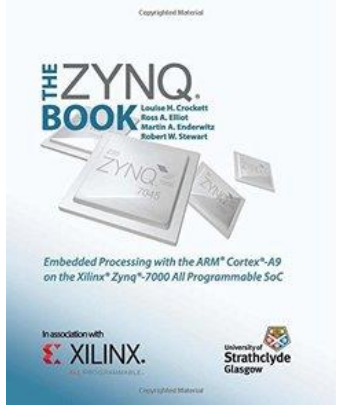
- **VHDL for Programmable Logic**
  - Kevin Skahill
  - Addison-Wesley



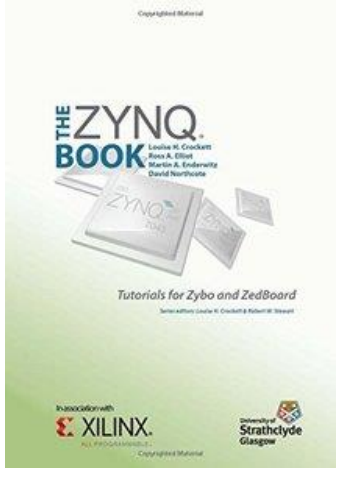
- **Digital Systems Design Using VHDL**
  - Charles H. Roth Jr., Lizy Kurian John
  - Cengage Learning



- **The Zynq Book**: Embedded Processing with the Arm Cortex-A9 on the Xilinx Zynq-7000 All Programmable Soc
  - Louise H Crockett, Ross A Elliot, Martin A Enderwitz, Robert W Stewart
  - Strathclyde Academic Media



- **The Zynq Book**: Tutorials for Zybo and ZedBoard
  - Louise H Crockett, Ross A Elliot, Martin A Enderwitz
  - Strathclyde Academic Media



Important Notes

- Visit our course website regularly!

- Plagiarism will **NOT** be tolerated!
  - Don't copy!
  - Don't let other(s) copy!
  - Can discuss but write up the solutions by yourself!

- Honesty in Academic Work:
  - http://www.cuhk.edu.hk/policy/academichonesty/

The best way to learn is through practice!

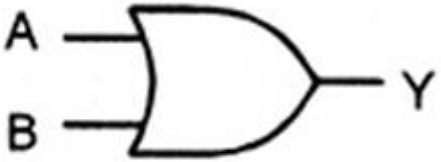
Review: Basic Gates in Logic Circuits

2-input AND gate

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

$Y = A \bullet B$

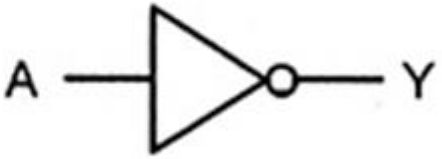
2-input OR gate



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

$Y = A + B$

Inverter (NOT gate)



A	Y
0	1
1	0

$Y = \overline{A}$

2-input EX-OR gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

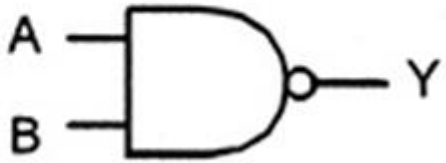
$Y = A \oplus B$

http://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_1

Review: NAND and NOR Gates

- In many technologies, implementation of NAND gates or NOR gates is easier than that of AND or OR gates.
  - NAND Gate:

2-input NAND gate

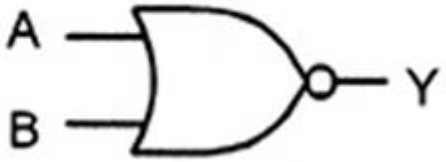


A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

$Y = \overline{A \bullet B}$

  - NOR Gate:

2-input NOR gate



A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

$Y = \overline{A + B}$

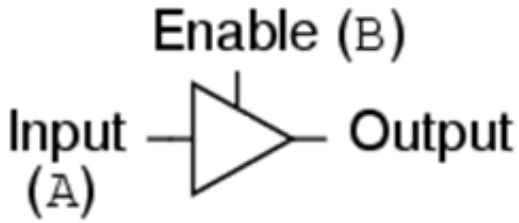
- Any logic function can be realized using only NAND gates or only NOR gates.

http://www.nutsvolts.com/magazine/article/understanding_digital_buffer_gate_and_ic_circuits_part_1

Review: Tristate Logic

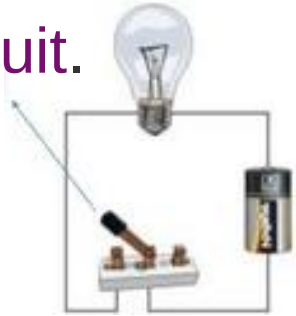
- The concept of tristate logic is also essential in digital system designs.
  - Directly connecting outputs of two gates together might not operate properly, and might cause damage to the circuit.
  - One ways is to use tristate buffers.
- Tristate buffers are gates with a high impedance state (High-Z or Z) in addition to high and low logic states.
  - High impedance state is equivalent to an open circuit.

Tristate buffer symbol



Truth table

A	B	Output
0	0	High-Z
0	1	0
1	0	High-Z
1	1	1



(analogy)

Class Exercise 0.1

Student ID: _____ Date:
Name: _____ _____

- Please use tristate buffers to implement a signal selector of two input signals using one enable signal.