Sequential logic

How to write this chapter

`Where we are?`

We know how computers do computing. So compute what? How to remember data?This is what we are going to deal with in this chapter.

`Ideas delivery`

By writing this chapter, we will focus on deliverying the follow ideas:

- There is two essential concepts in memory: **time and space**. The states of data must be changed and maintained overtime, and to memory different data at one time we must store them in different places, and access them by their locations.
- We will **abstract** the chips built in chapter 1 to help us **make chioce**, what to do and where to locate.

`Project developing`

3 parts:

- Register : Basical R&W
- Memory : Location
- Counter : Control

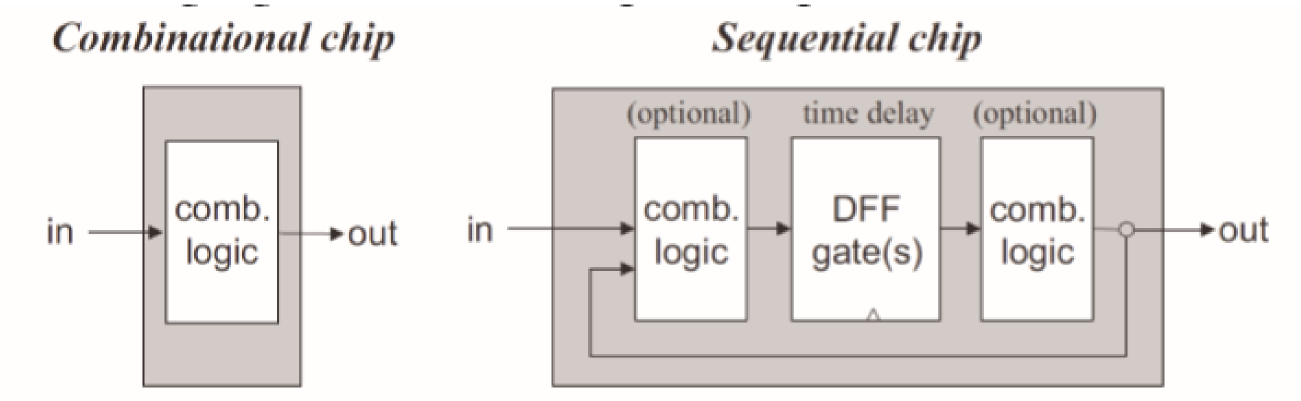
(Same requirement.)

Introduction: Why Time?

`why sequential?`

Combinational logic means that the output of a combinational chip **depends solely on the current input**, regardless of what the previous output the chip has. So here is a question: Can we build a computer with only combinational logic ?

Not actually. If there is only combinational logic, if you give the computer an input, the computer will produce one output regardless of the previous state, and after that **the state of the computer will not be changed**. But in most cases we must do the same operation many times, to **reuses the limted computing resouces**, so the **state must be changed over time.** As a result, we must have components which can maintain and change states overtime.



`DFF`

So we should find a component which is **time-dependent.** And the most basic time-dependent expression can be: **out(t) = in(t-1)**. We can change state at t by define in(t-1), and by input the output we can maintain state.

$$out(t) = in(t-1)$$

Fig 3.3 DFF

Project development

- "register"
  - Read and write.
  - Hint: **Choose what operation to do**.Which chip can you use?
- "memory"
  - Add Location
  - Hint: **Choose which one to get**. Which chip cna you use?
- "Control"
  - Intro: the picture:
  - Hint: More choices. Consider carefully about **the level**.

Futher reading

- Comparision: **Disk**
  - Remained when power off?
- Expansion: **Web**
  - More memory across space.
  - Different location, but the structure is the same.
- Comparision with **human mind** .

Connection with latter chapter

- ROM and RAM. Pay attention to the location:
  - A instruction
  - Jump instruction.
- VM memory mapping: virtual register, stack, heap.
-

RAM addresses	*Usage*
0–15	Sixteen virtual registers, usage described below
16–255	Static variables (of all the VM functions in the VM program)
256–2047	Stack
2048–16483	Heap (used to store objects and arrays)
16384–24575	Memory mapped I/O

- Types: Float, object and array.