# WEEK 2 SCRIPT -

-- PIC 1 --

Welcome back to week 2 of the computer design and architecture course, in this week, we’re going to build a RAM Unit for our baby computer this is extremely important for the computer as the ALU we built previously needs a place to store and access information while it’s executing an instruction. Also, we will learn about a machine language that we ourselves are going to develop, this is essential as our computer is useless if we have no way to communicate with it.

Just as in week 1, this week too will be divided into 4 parts, 3 which we will do together and 1 as your weekly project.

-- PIC 2 --

In part 1, I’m going to introduce you to how memory works and why we need it. We’ll also explore these weird and interesting objects called flip-flops. Now there are many kinds of flip-flops, of we’re using just 1 to make the RAM of our computer.

Just as the Nand gate was the building block for our ALU, the flip-flop is going to be the building block for our RAM chip. As to what a flip-flop does I would like to illustrate with an example: Let’s say you have to go to the supermarket and buy an icecream, now you can write it down on a piece of paper and read that piece of paper when you reach there, or you can keep reminding yourself all the time while you’re going there, the flip-flop works like the latter case, it “remembers” the value of a single bit, by continuously running the signal within it in a loop until a new value is loaded.By combining a few of these flip-flops, we get a REGISTER, which is truly a remarkable object that’s the basis for all memory in the RAM of our computer.

-- PIC 3 --

With the idea and implementation of a Register, we take our first steps into part 2, where will combine them and using some basic chips we made in week 1, create our first RAM unit consisting of a grand total of 8 Registers, or 16 bytes called RAM8. It might seem a bit unimpressive, but by combining them with other 8 other chips, we get a RAM64 chip with 64 Register chips and so on we keep going and watch the snowball getting bigger and bigger until we’ve reached our desired RAM implementation.

-- PIC 4 --

Our RAM completed, we move on to part 3, where we learn about something called an assembly language which is the closest interface present between the physical chip we’ve built and a language that humans can arguably understand. We will also learn how to handle input and output devices to our computer such as a keyboard or a monitor.

-- PIC 5 --

In the weekly project, you’re going to build and simulate a 16 KiloByte RAM unit and write a program in assembly language to control the keyboard input and screen output to simulate the pressing of a key.

-- PIC 6 --

Our computer, Now armed with Memory, input and output devices and an assembly language to convert human instructions to the binary language it can understand is now almost ready to be used. Except, it’s missing a very crucial component - yes, you’ve guessed right, it’s the CPU.

Meet you next week, where we’ll build our very own CPU and try to understand how our assembler actually converts the assembly language to 0s and 1s that a computer can understand.