A CPU is a large, complex and powerful device that is used to process digital data. It is a central part of any computer system and its job is to carry out instructions that a user enters into it. All the digital data that is used by the computer is first processed by the CPU. This is done by the CPU Memory Machine Language.

The CPU consists of a number of registers that are used to store data. The registers are divided into three categories: the instruction register, the data register, and the stack pointer. The instruction register stores the immediate instructions that the CPU is currently processing. The data register stores the data that the CPU is currently working with. The stack pointer stores the address of the next push or pop operation that the CPU will carry out.

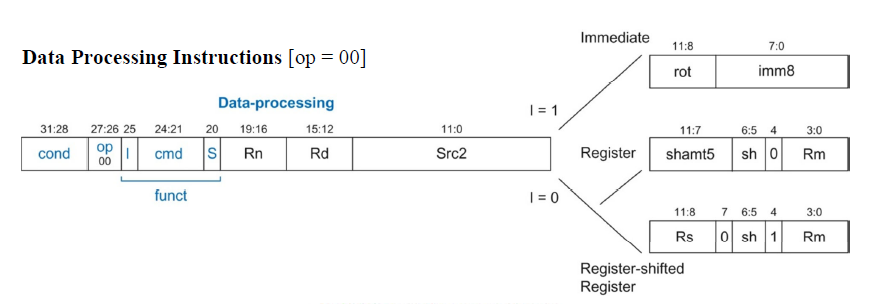
The CPU register process binary hexadecimal opcode memory introduction is a writing assignment which will introduce you to the different ways in which computers store and use information. It is essential that anyone who wants to understand how a computer works understands this basic building block of computing. By understanding these basics, you can start to make sense of more complex topics like machine language, software development, and processor designs.

The CPU is able to carry out different operations using a set of opcodes. These opcodes are written in a particular language, known as machine language. Machine language is written in a special code that is specific to a particular type of CPU. This code is processed by a computer's processor, and the results of the operations are displayed on the screen.

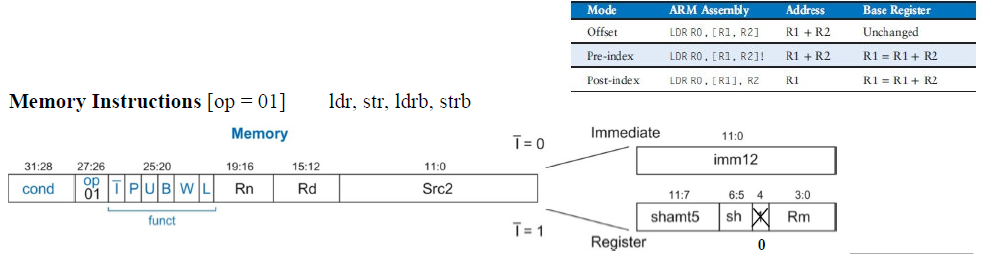
Memory is a lot more than just a place where data is stored. It is also responsible for containing the instructions that the CPU is currently processing. This is why the CPU needs access to memory in order to carry out its tasks. In order for the CPU to access memory, it needs to know its location on the computer. This is done by the Memory Machine Language.

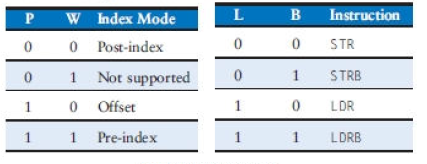
CPU Register Process Binary Hexadecimal Opcode Memory Introduction Memory Machine Language is a process that is used by the CPU to carry out its tasks. It is made up of a number of registers that are used to store data, and it is able to carry out different operations using a set of opcodes.

* **Breaking down the Instructions**



* Here the first 4 bits are conditional codes (if any).
* After that 2 bits are op codes to determine type of the instruction (Data Processing / Memory Management / Branching).
* Then 1st bit is representing I, then 4 bits for the actual command (Add/Sub/Mov/etc), and next 1 bit is for S.
* Next 4 bits are the value of Rn (if needed).
* Then the remaining bits represent src2. If I is set then last 4 bits will contain register no that need to be used or if I is clear then last 8 bits will represent the no to be used.



* Here the first 4 bits are conditional codes (if any).
* After that 2 bits are op codes to determine

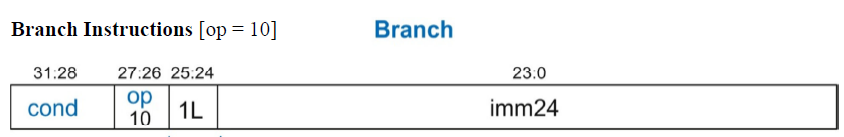
type of the instruction (Data Processing /

Memory Management / Branching).

* Then consecutive 1 bits are representing I, P,

U, B, W, L.

* Next 4 bits are the value of Rn (if needed).
* Next 4 bits are the value of Rd.
* Then the remaining bits represent src2. If I is set then last 4 bits will contain register no that need to be used or if I is clear then last 12 bits will represent the no to be used.



* Here the first 4 bits are conditional codes (if any).
* After that 2 bits are op codes to determine type of the instruction (Data Processing / Memory Management / Branching).
* Then 2 bits are representing L value.
* Then the remaining 24 bits represent src2.
* **Op Code Table**

|  |  |  |
| --- | --- | --- |
| CMD | NAME | OPERATION |
| 0100 | ADD Rd, Rn, Src |  |
| 0010 | SUB Rd, Rn, Src |  |
| 0101 | ADC Rd, Rn, Src |  |
| 0110 | SBC Rd, Rn, Src |  |
| 1010 | CMP Rn, Src |  |
| 1101 | MOV Rd, Src |  |