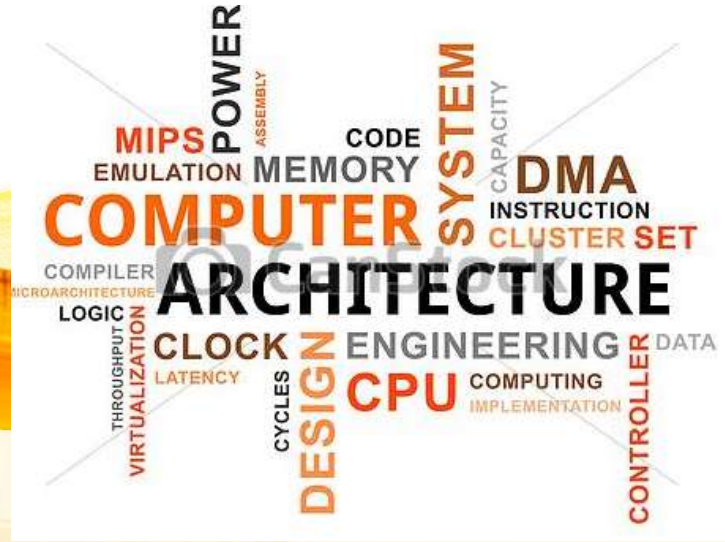
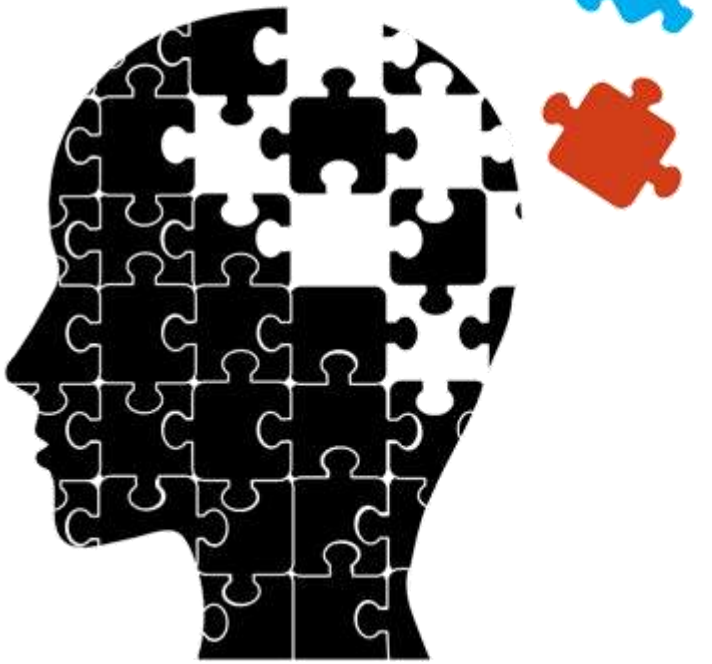
UNIT I
BASIC STRUCTURE OF COMPUTERS

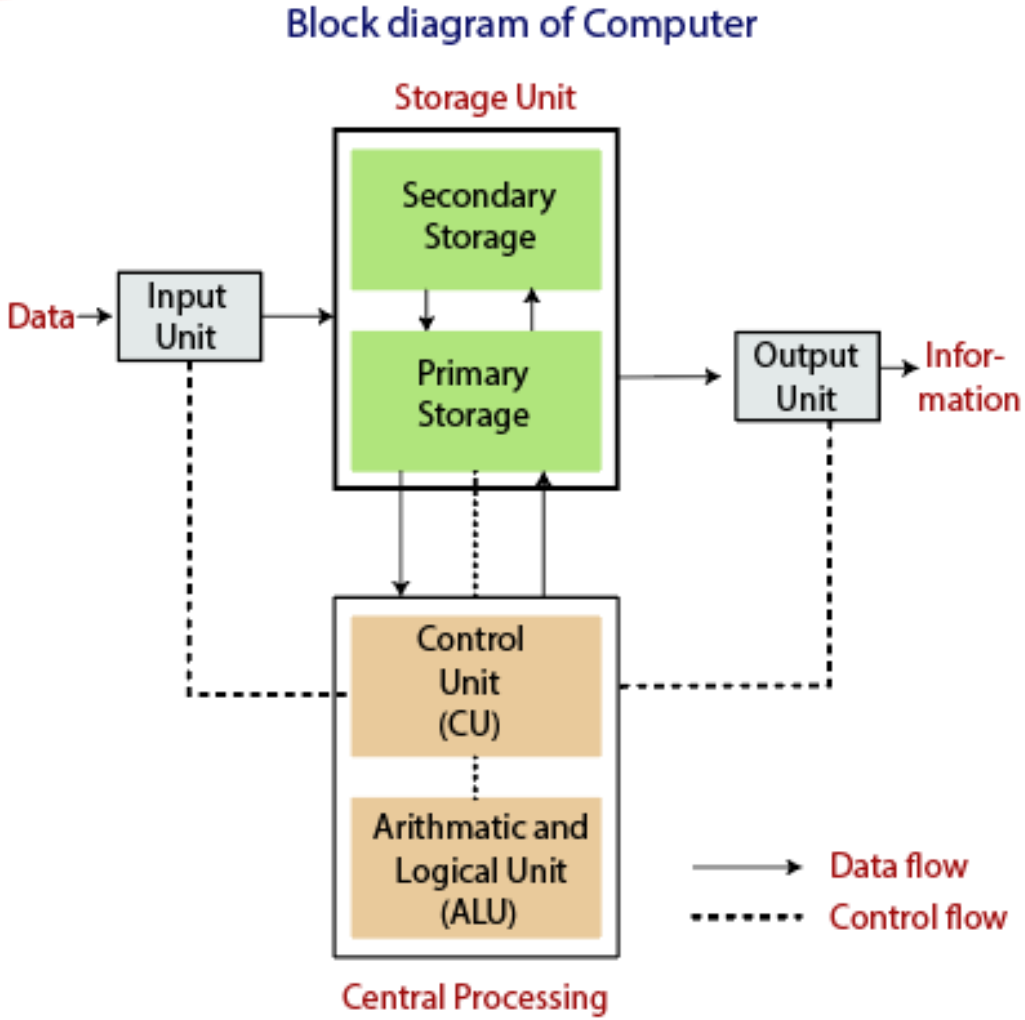
Functional units – Basic operational concepts – Bus Structures – Performance – Memory locations and addresses – Memory operations – Instruction and Instruction sequencing –– Addressing modes – Assembly language – Case study : RISC and CISC Architecture.

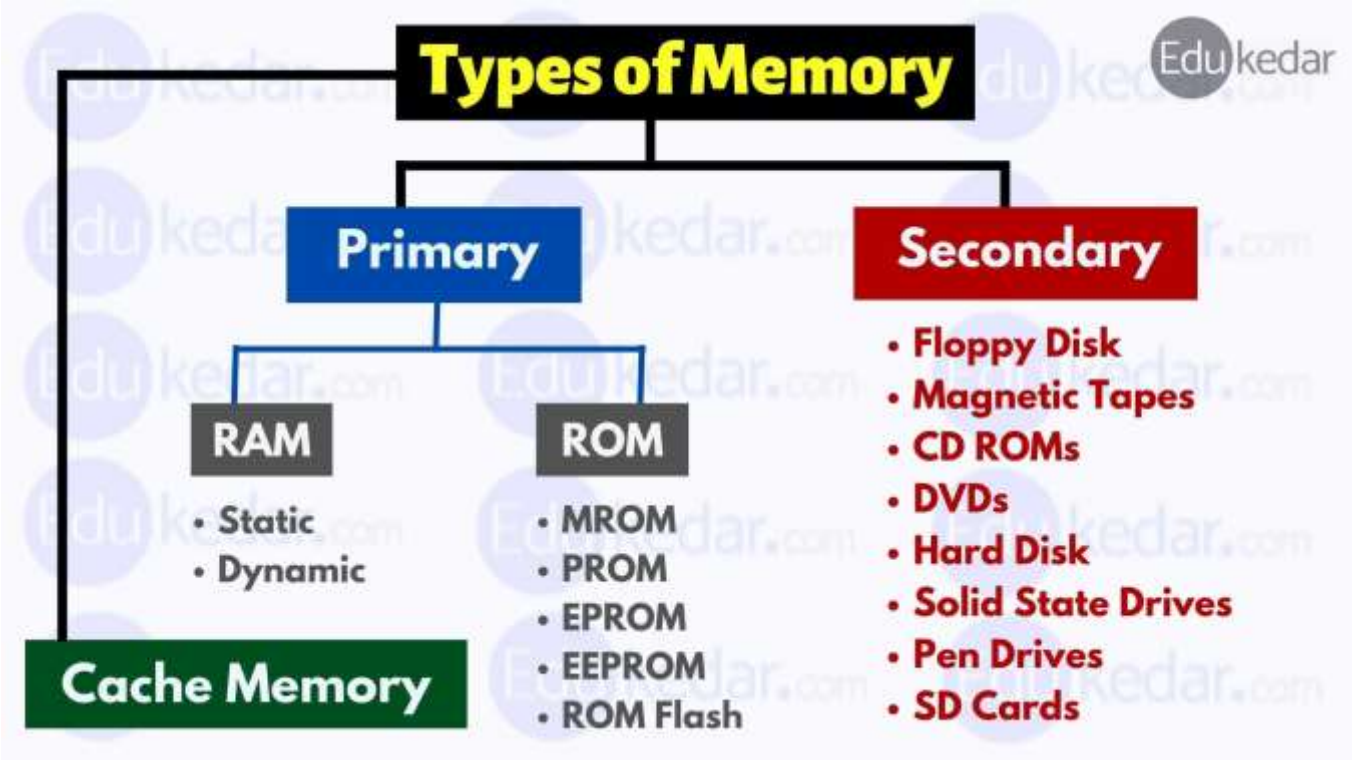


Recall the previous class concepts



Functional Unit



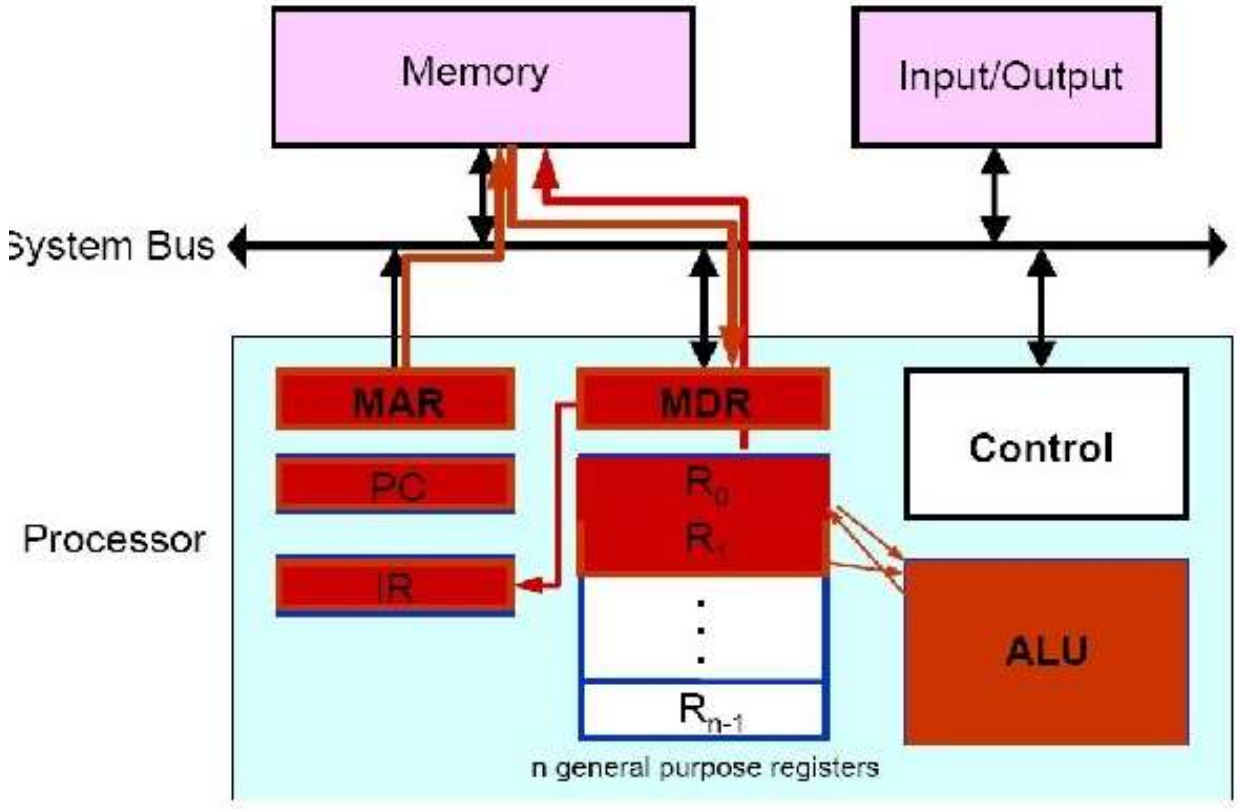


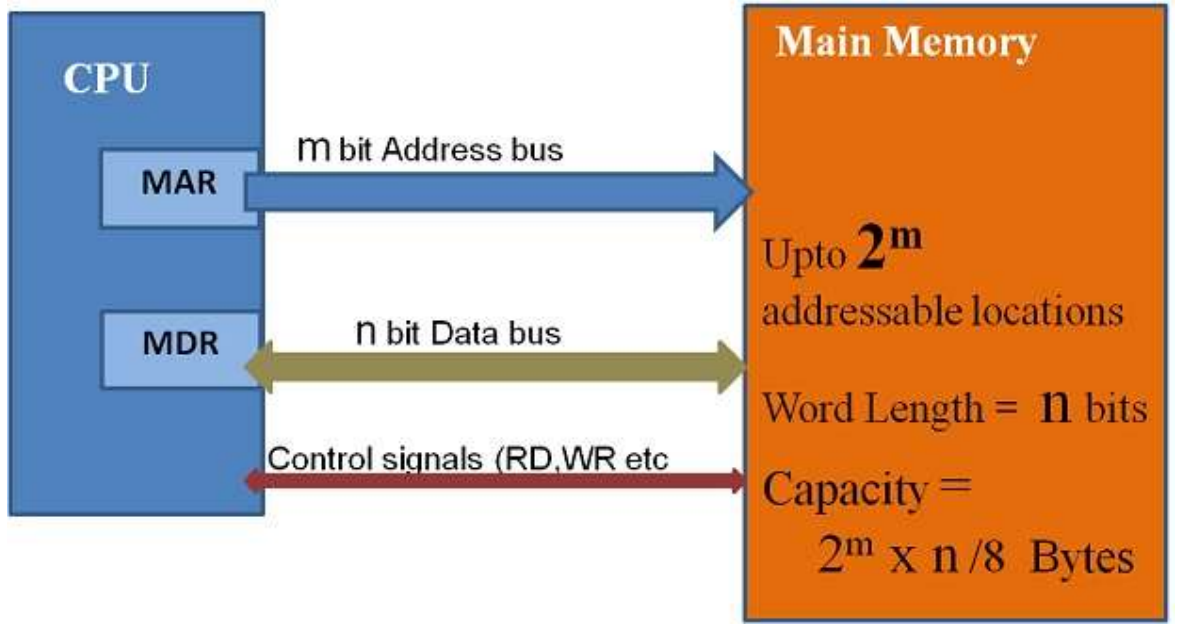
Analysing how processor and memory are connected

- Processors have various registers to perform various functions
- **Program Counter** - It contains the memory address of next instruction to be fetched.
- **Instruction Register** - It holds the instruction which is currently being executed
- **MDR** - It facilities communication with memory. It contains the data to be written into or read out of the addressed location.
- **MAR** - It holds the address of the location that is to be accessed

n general purpose registers that is R0 to Rn-1

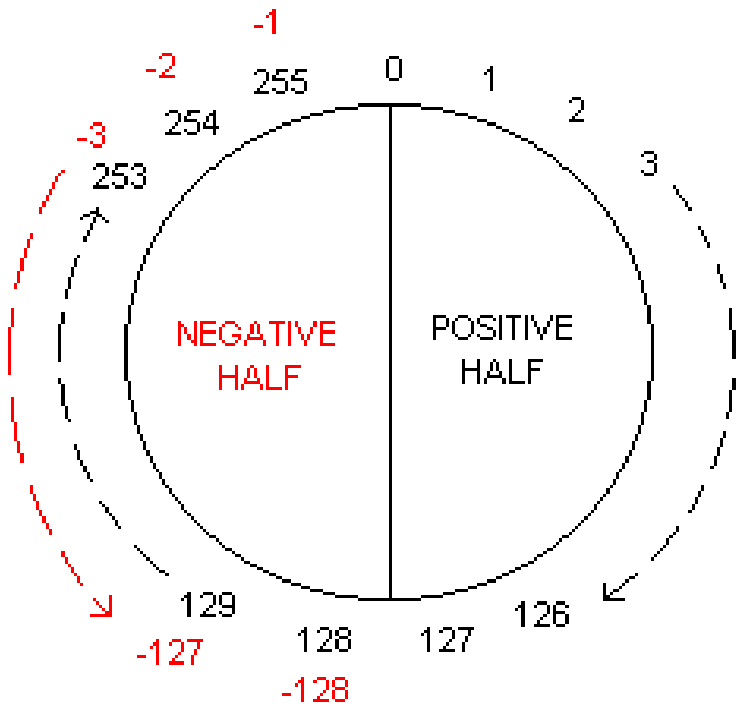
Connection between Processor & Memory





Operations & Operands -

UNSIGNED NUMBERS: 0 to 255
SIGNED NUMBERS: -1 to -128 and 0 to +127

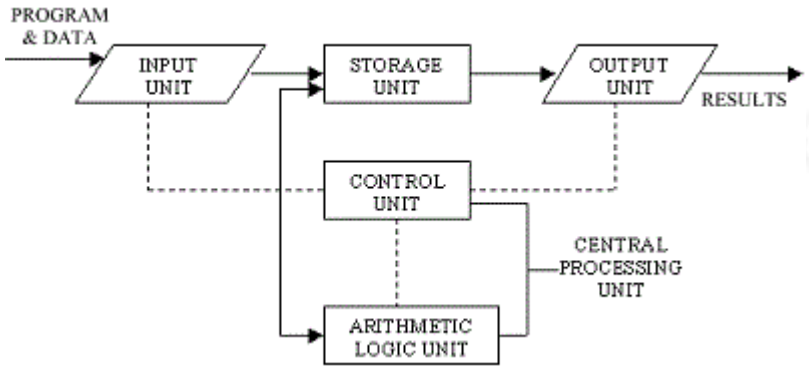


8-BIT NUMBER SYSTEM

Operations of Computer Hardware

add a, b, c # a gets b + c

Adds 4 Variables



ADD a,b,c
ADD a,a,d
ADD a,a,e

Basic Operational Concepts

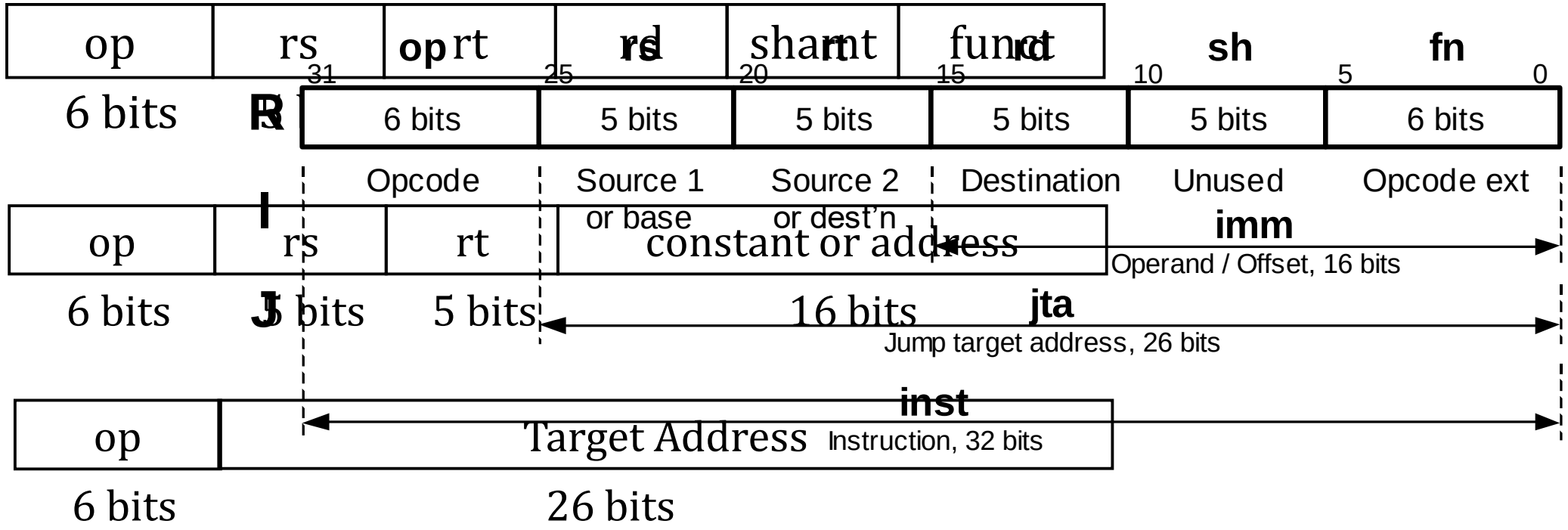
- Instruction consists of 2 parts

- Example

```
ADD   LOCA, R0
```

```
Load LOCA, R1
Add R1, R0
```

Instructions Format



Translating Arm Assembly Instructions into Machine Instructions

op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

add $t0, $s1, $s2

special	$s1	$s2	$t0	0	add
0	17	18	8	0	32
000000	10001	10010	01000	00000	100000

$$00000010001100100100000000100000_2 = 02324020_{16}$$

ARM Assembly Language

Logical	and	and $s1,$s2,$s3	$s1 = $s2 & $s3
	or	or $s1,$s2,$s3	$s1 = $s2 | $s3
	nor	nor $s1,$s2,$s3	$s1 = ~ ($s2 | $s3)
	and immediate	andi $s1,$s2,20	$s1 = $s2 & 20
	or immediate	ori $s1,$s2,20	$s1 = $s2 | 20
	shift left logical	sll $s1,$s2,10	$s1 = $s2 << 10
	shift right logical	srl $s1,$s2,10	$s1 = $s2 >> 10
Conditional branch	branch on equal	beq $s1,$s2,25	if ($s1 == $s2) go to PC + 4 + 100
	branch on not equal	bne $s1,$s2,25	if ($s1!= $s2) go to PC + 4 + 100
	set on less than	slt $s1,$s2,$s3	if ($s2 < $s3) $s1 = 1; else $s1 = 0
	set on less than unsigned	sltu $s1,$s2,$s3	if ($s2 < $s3) $s1 = 1; else $s1 = 0
	set less than immediate	slti $s1,$s2,20	if ($s2 < 20) $s1 = 1; else $s1 = 0
	set less than immediate unsigned	sltiu $s1,$s2,20	if ($s2 < 20) $s1 = 1; else $s1 = 0
Unconditional jump	jump	j 2500	go to 10000
	jump register	jr $ra	go to $ra
	jump and link	jal 2500	$ra = PC + 4; go to 10000

Operating System


Thank You