# Formulas

Performance = 1 / Execution Time

Performance\_x / Performance\_y = Execution\_time\_y / execution\_time\_x = n

Clock Cycles = Instruction Count \* Cycles per Instruction

CPU Time = CPU Clock Cycles \* Clock Cycle Time

= CPU Clock Cycles / Clock Rate

= Instruction Count \* CPI \* Clock Cycle Time

= Instructions / Program \* Clock Cycles / Instruction \* Seconds / Clock Cycles

- Measuring Execution Time

- Elapsed time

- Total response time, including

- processing

- time the cpu actually spends running the instructions

- IO

- anything that comes from outside, would include sensors

- the slowest component

- OS Overhead

- management and scheduling

- idle time

- time spend waiting. Time waiting to be scheduled

- CPU Time

- This is what we will be focusing on today

- Time spent processing a given job

- There is no subtract immediate instruction

- just use negative numbers in the 'addi' instruction

# General Info

- There is no subtract immediate instruction

- just use negative numbers in the 'addi' instruction

- Basic Blocks

- a block is a sequence of instructions with

- no embedded branches (except an end)

- no branch targets

- there are different instructions for comparing signed and unsigned numbers

- Procedure Call

- Jump and link

`jal ProcedureLabel`

- jumps to a given label, when the program reaches the end, it jumps back to the

caller

- Procedure Return

- jump register

`jr $ra`

- what effects the program performance

- CPI / Clock speed / Time per cycle

- algorithm, programming language, compiler, architecture

- Instruction count is determined by program, ISA and compiler!

- Response Time and Throughput

- response time

- how long it takes to do a task

- throughput

- total work done per unit time

- for relative negative means up on the page, and positive means down

- think about how the instructions would be numbered from first to last

# Translating Between Assembly and Machine Code:

add $s1, $t1, $t2

op / funct rd rs rt

add $t0, $s1, $s2

| op | rs | rt | rd | shamt | funct |

|--------------|---------------|-------------|------------|--------------|---------------|

| special | $s1 | $s2 | $t0 | 0 | add |

| 0 | 17 | 18 | 8 | 0 | 32 |

| 000000 | 10001 | 10010 | 01000 | 00000 | 100000 |

Machine instruction = 00000010001100100100000000100000 = 0x02324020