Alexander Gaskins

Dynamic Energy/Power Effect

$$DE = \frac{1}{2}[Capacitive\ Load][Voltage]^2$$

- Capacitive load charges and releases energy

$$DP = \frac{[Capacitive\ Load][Voltage]^2[Frequency\ Switched]}{2}$$

- Reducing clock rate reduces power, not energy

Dependability

- Mean time to failure (MTTF) [*Measures reliability*]
- Mean time to repair (MTTR)
- Mean time between failures (MTBF)=MTTF+MTTR
- Availability = MTTF / MTBF
- Failures in time (FIT) = $\frac{1}{MTTF}$ [*The rate of failure*]
- Module availability = $\frac{MTTF}{(MTTF+MTTR)}$

Systems alternate between 2 states of service for an SLA

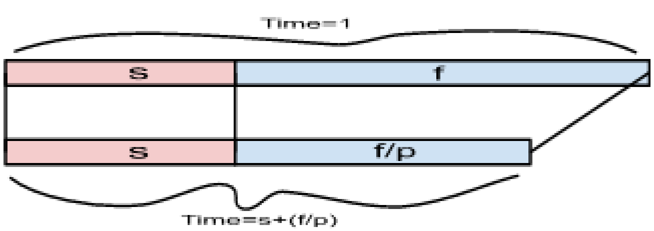
1.) Service Accomplishment: Service is delivered as specified

2.) Service Interruption: Delivered service is different from SLA

- Failure = transition from state 1 to state 2
- Restoration = transition from state 2 to state 1

Amdahl's Law (Speedup/Improvement)

- When the common case is sped up, the computer spends less time on that case so the case becomes less common



Trends in Cost

$$Die\ Yield = \frac{Wafer\ yield}{[1+Defects\ per\ unit\ area \bullet Die\ area]^N}$$

- N = process complexity factor

$$Dies\ per\ wafer = \frac{\pi[\frac{Wafer\ diameter}{2}]^2}{Die\ area} - \frac{\pi(Wafer\ diameter)}{\sqrt{2 \bullet Die\ Area}}$$

$$Cost\ of\ Integrated\ Circuit = \frac{Cost\ of\ die + Cost\ of\ testing\ die + Cost\ of\ packaging\ and\ final\ test}{Final\ test\ yield}$$

$$Cost\ of\ die = \frac{Cost\ of\ wafer}{Dies\ per\ wafer \bullet Die\ yield}$$

SPEC Power Benchmark

- Performance: ssj_ops/sec
- Power: Watts (Joules/sec)

Pwr consumption of server at dif workload lvls

$$Overall\ ssj_{ops}\ per\ Watt = \frac{\left(\sum_{i=0}^{10} ssj_{ops_i}\right)}{\left(\sum_{i=0}^{10} power_i\right)}$$

```
fadd.d   f2, f0, 3      ;f2 is initialized with 3
fld      f4, 0(x1)      ;f4 is array element
fmult.d  f4, f4, f2     ;multiply scalar
fsd      f4, 0(x1)      ;store result
subi     x1, x1, 8      ;decrement pointer 8 bytes
bne      x1, x2, -20    ;branch x1 != x2; If branch ta

lw       x1, 104(x0)    ;load B into x1
lw       x2, 108(x0)    :load C into x2
addw     x3, x1, x2     ;B+C
sw       x3, 100(x0)    ;store A from x3
addiw    x4, x0, 1      ;initialize x4 with 1
addw     x5, x3, x2     ;A+C
subw     x1, x5, x4     ;(A+C) -1
sw       x1, 104(x0)    ;store B
beq      x2, 0, 8       ; If C=0, branch taken; then P(
addiw    x5, x0, 2      ; if branch not taken, D=2
jal      x31, 4         ;jump post taken branch
addiw    x5, x0, 1      ; D is set to 1
sw       x5, 112(x0)    ;result for D is stored
```

Five Pipeline Stages of RISC-V

IF (Instruction Fetch) → Fetch instruction from mem & Update pc variable

ID (Instruction Decode) → Fetch register values, detect if it is branch or jump, compute target address

EX → (ALU Operation, or Compute memory Address) Test Branch Condition; change PC to target if needed

MEM (or DM) → Read or Write memory Operand

WB → Write ALU result or Memory Load Operand into Destination Register