Hoje

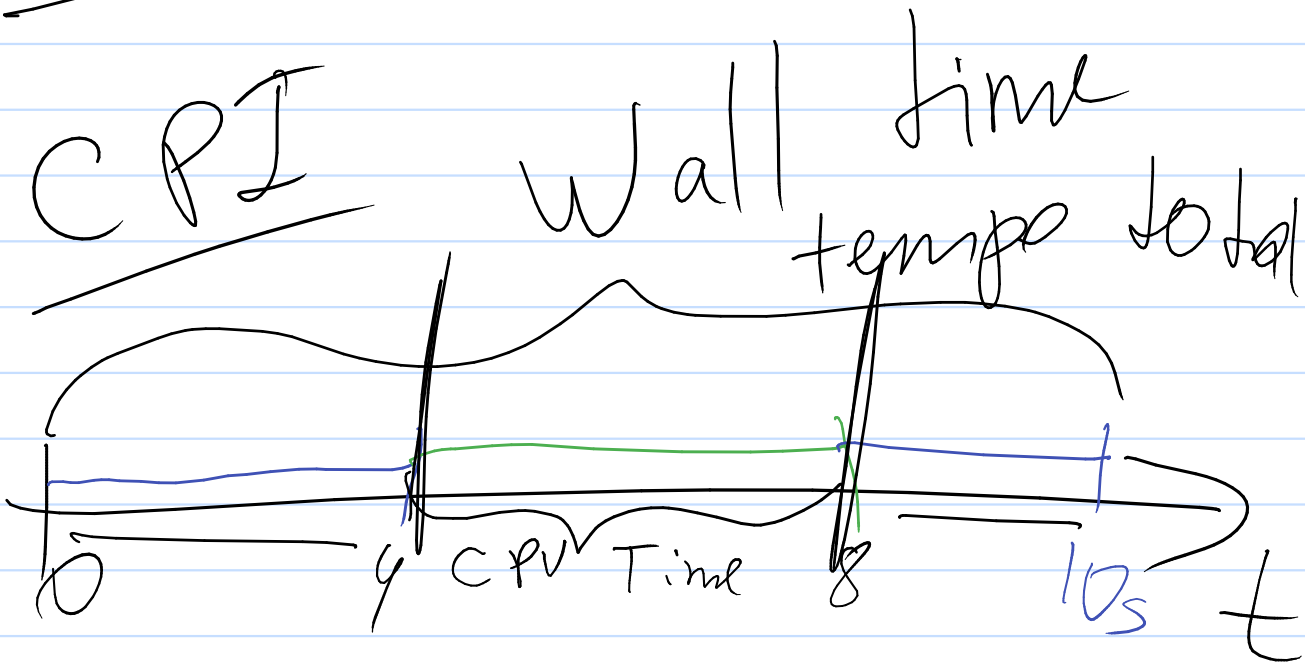
Medidas de desempenho

Sair do vi

tempo

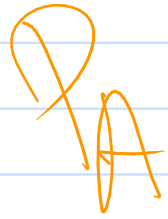
CPI

Wall time



ProgA mais rapide que ProgB

CPU time






CA

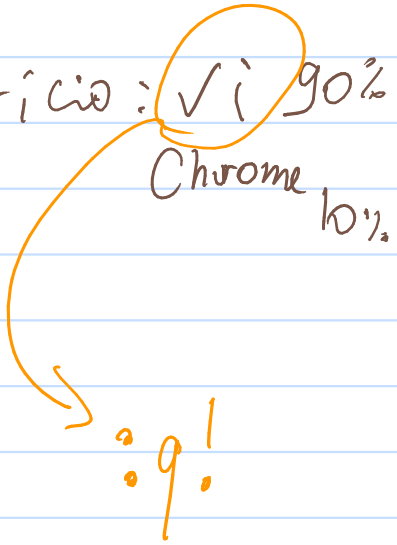
Gustavo: Sublime 50%
Chrome 50%

João Victor: Firefox 50%
XCode 50%

Francisco: Overwatch 50%
Opera 50%

CB

Mauricio: Vi 90%
Chrome 10%



Especificar uma tarefa que você faz no seu computador.

Não deixe de dar detalhes para reproduzir a execução.

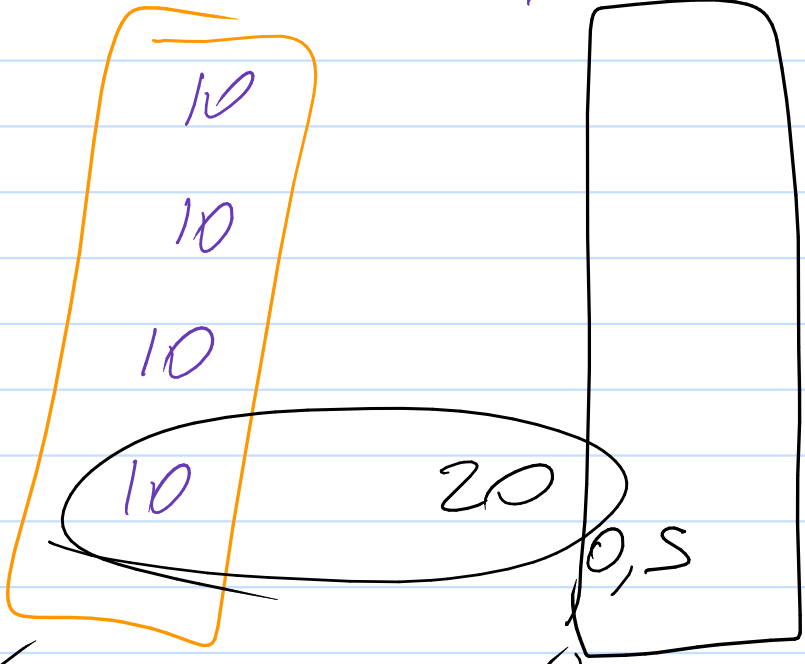
Benchmark mc732/mc722 2016s2:

A. Renderizar um vídeo específico
B. Rodar uma macro numa planilha gigante
C. Compilar o Kernel do Linux no gcc
D. Rodar 30 vídeos simultaneamente o mais rápido possível

...

Tempo em s

	Computador X	Y	Z
A	10		
B	10		
C	10		
D	10	20	0,5



média geométrica

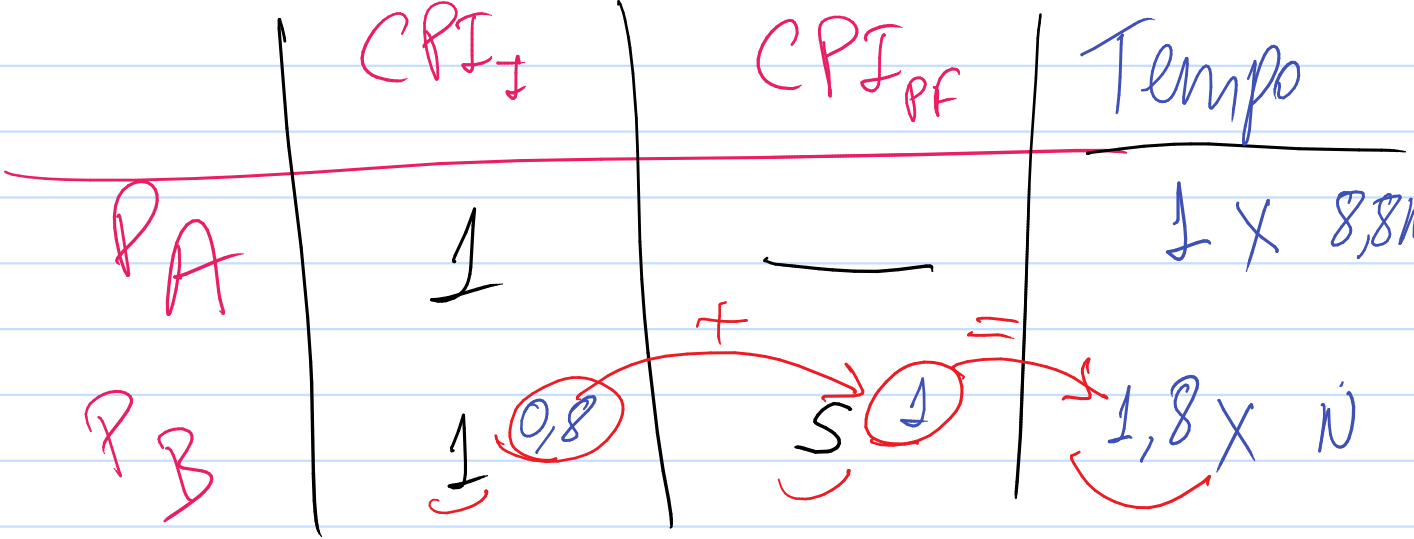
3 processadores

	20% CPI_X	30% CPI_Y	50% CPI_Z	Mesma frequência CPI	
P_A	1	1	1	1	
P_B	$1^{0,2}$	10^{3}	$5^{2,5}$	5,7	$\times p \times N$
P_C	$1^{0,2}$	$7^{2,1}$	6^{3}	5,3	$\times p \times N$

Inteiros

Memória

Ponto-flutuante



Prog | 80% | 20% | $N_1 + N_2$

N insth ~> 0,2 N . 40 + 0,8N

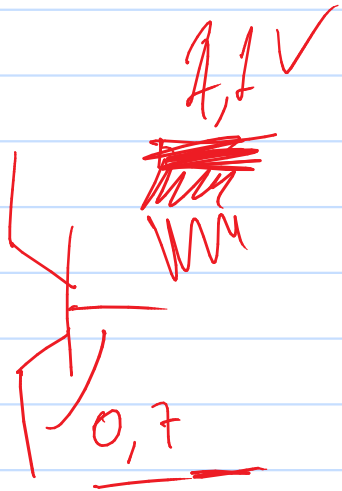
8,8N

Soma(N_1, N_2)

40 insth

$$\text{Power} = C \cdot V^2 \cdot f$$

$$\text{Power}_2 = \frac{C}{4} \cdot 4V^2 \cdot 2f$$



$$\text{Power} = 2Cf \quad V^2 \to 4 \quad 4f$$

Dennard scaling

Aula passada:

Velocidade dos processadores

- Qual é mais rápido?

Fechar Capítulo 1

- Casos de análise de desempenho

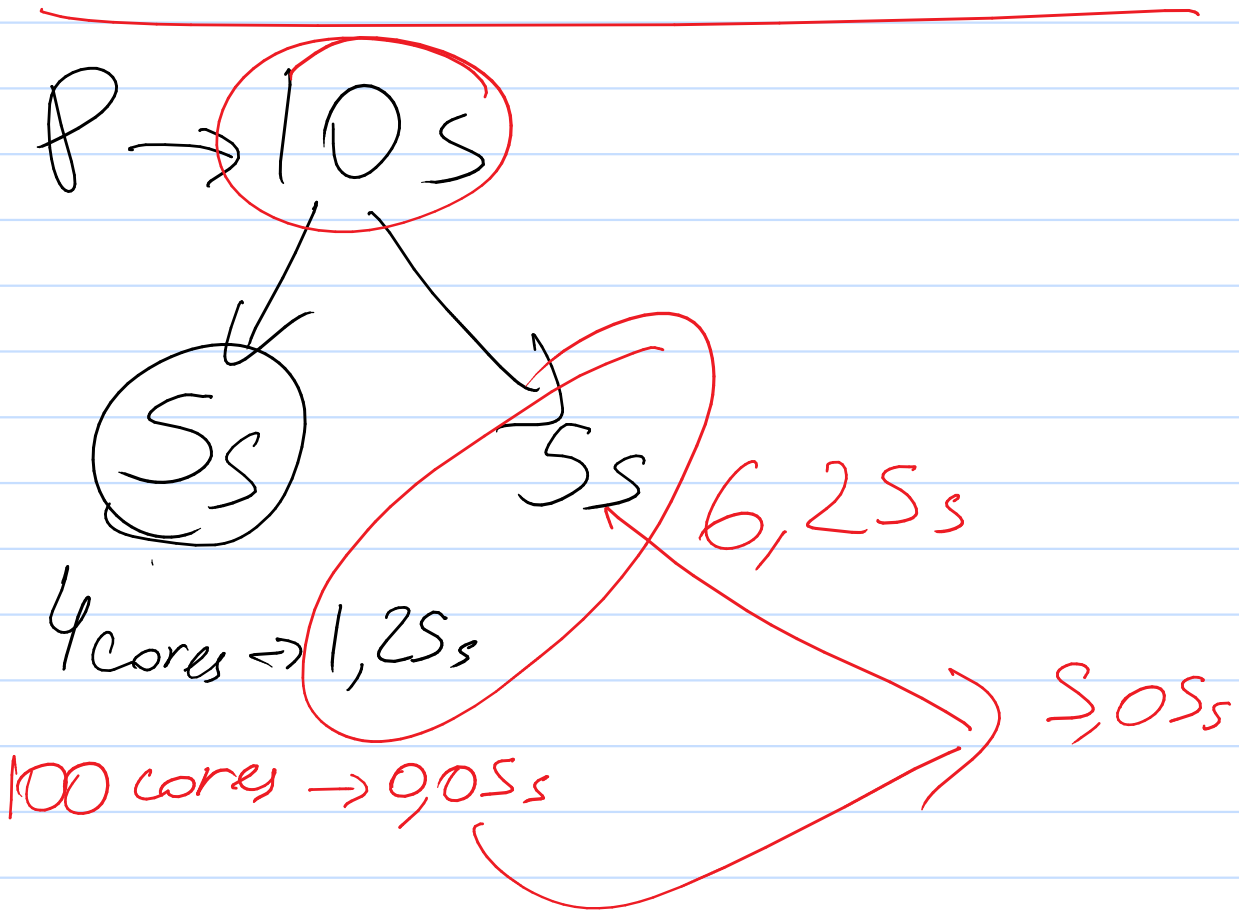
Lucas: Aula no CB até as 12:00
→ 12:00 – 13:00 RS
13:00 → Reunião do CaCo
Aula no IC as 14:00

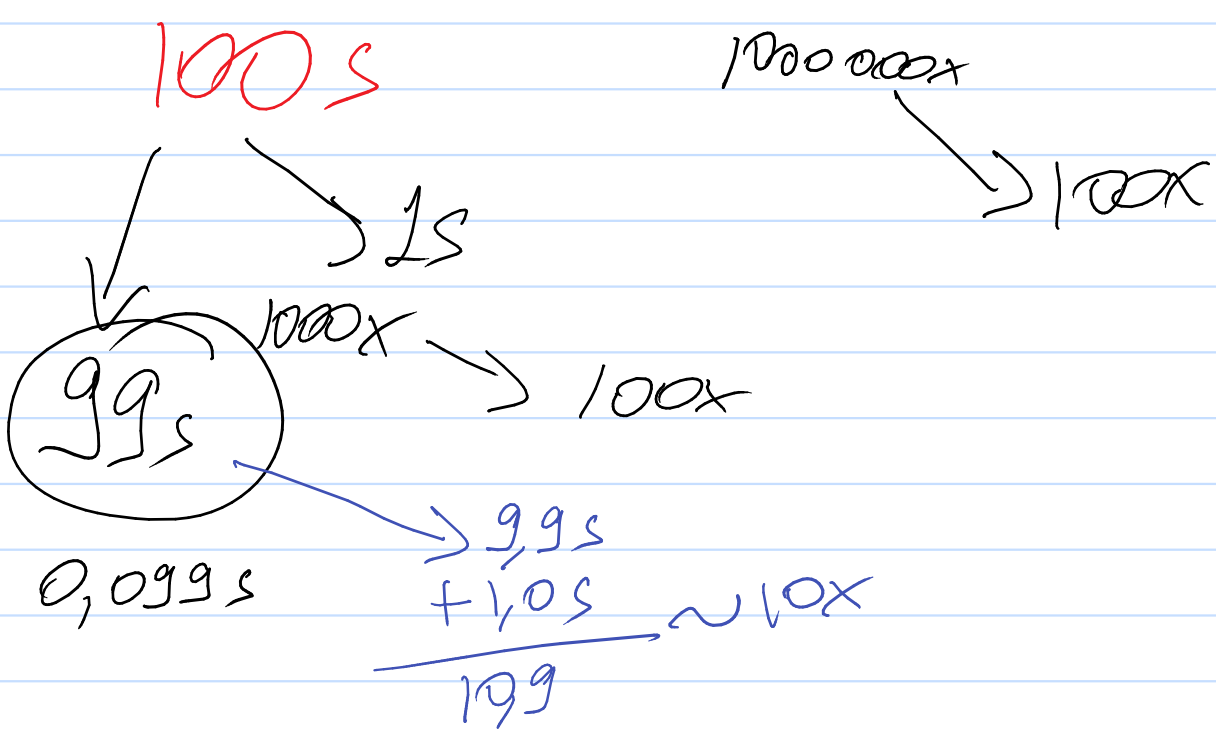
- Correndo
- Uber
- Circular interno e RU

CB → RS → 12 min

RS → 45 min

RS → IC → 3 min



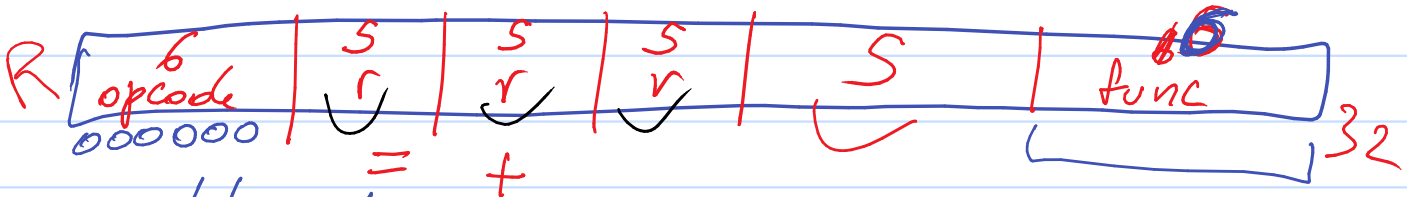


Conjunto de instruções - ISA

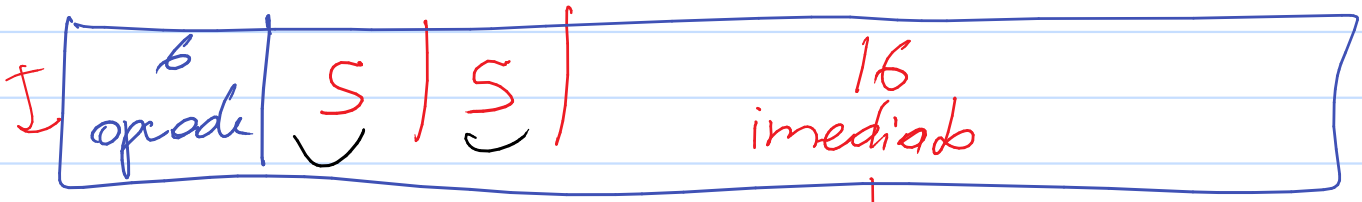
↳ Especificação do processador

add	beq
j	bne
jsub	slt
lw	
sw	
and	
or	

32



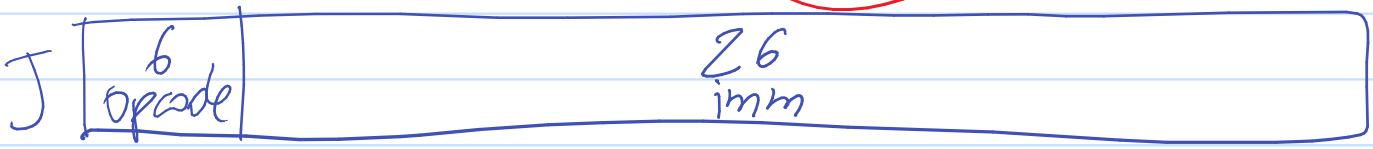
add r1, r2, r3



add r1, r2, 42

↓

resto

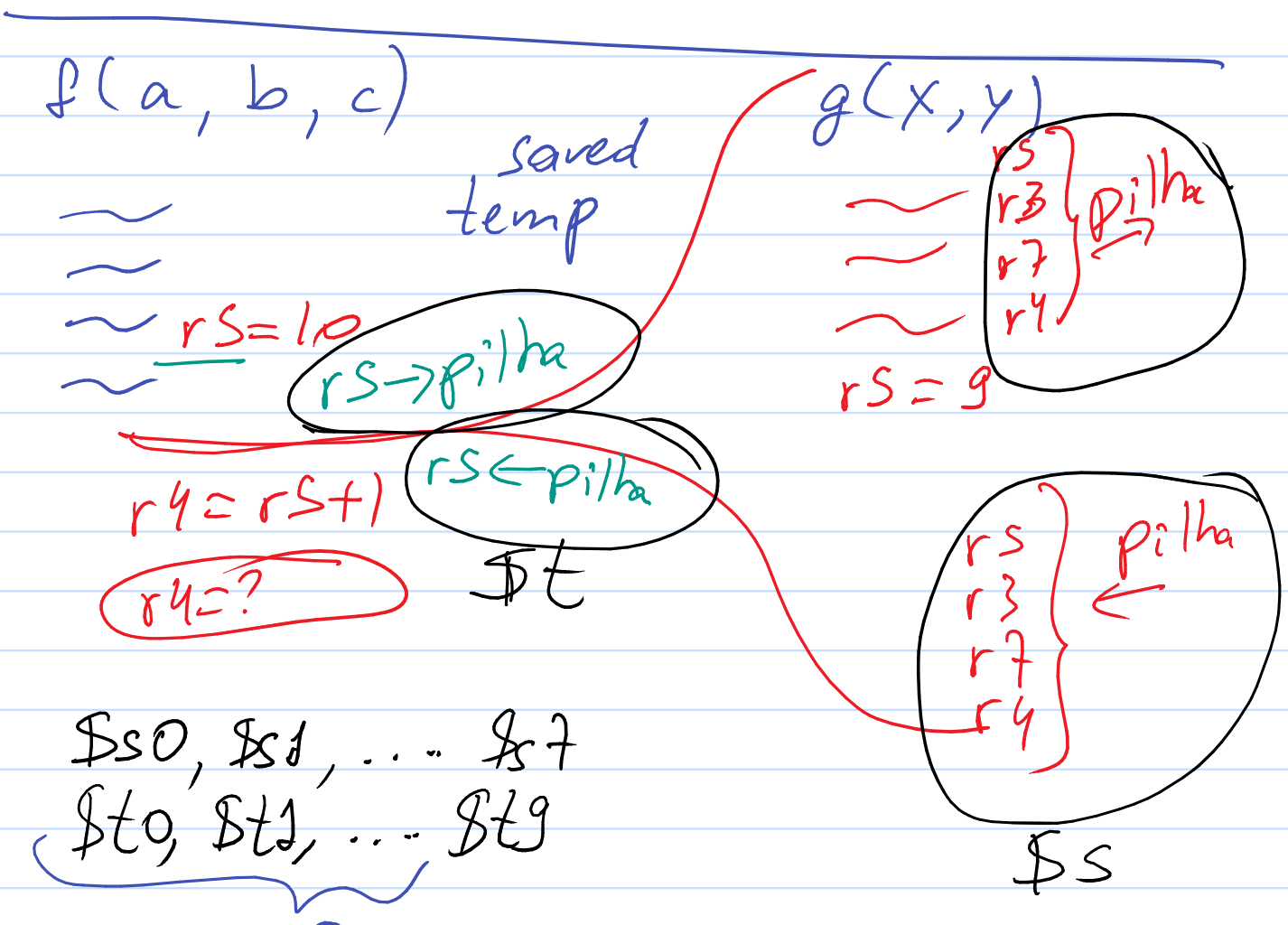


63 + 64 instruções

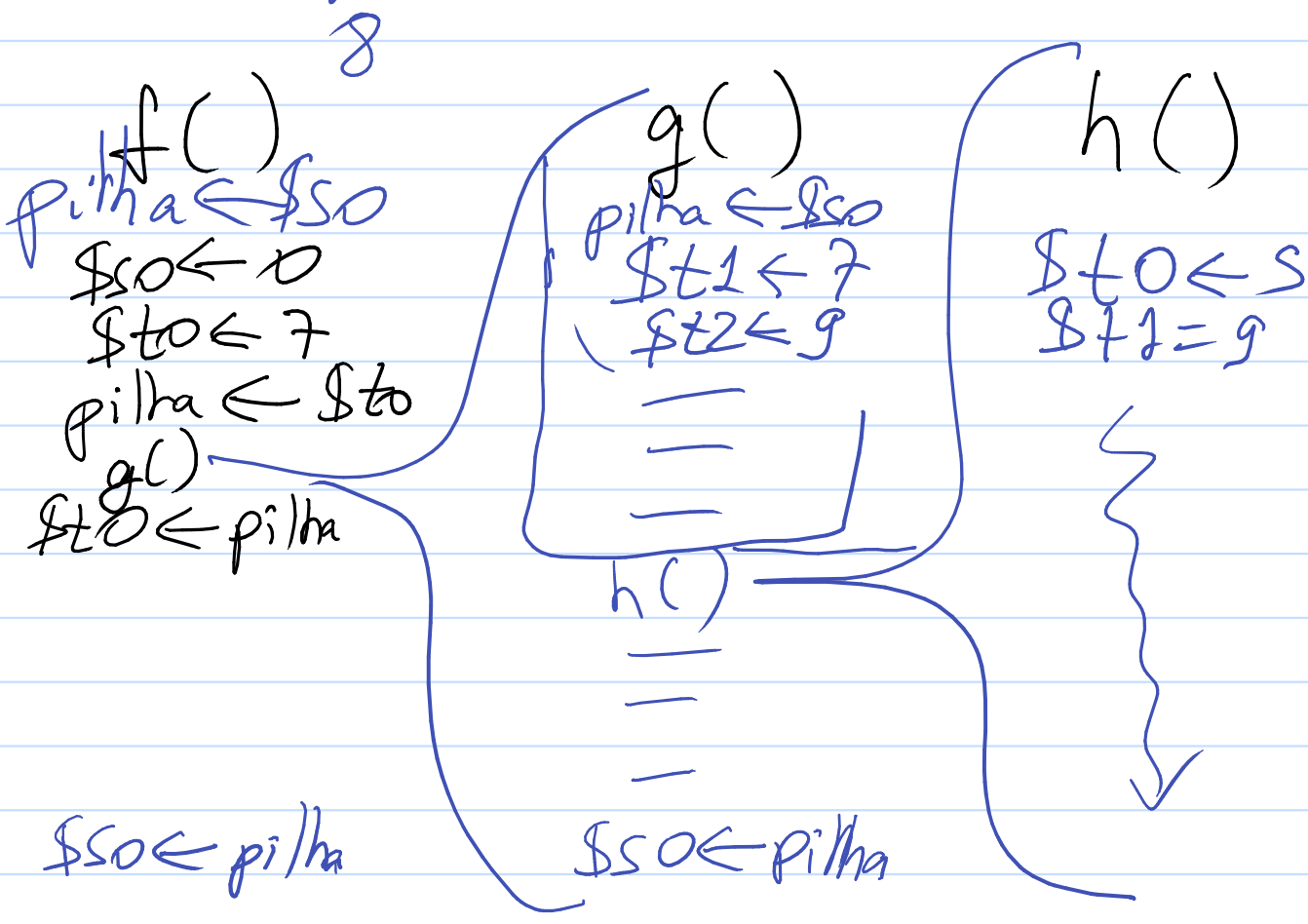
- Por que o opcode não é de 7 bits?
- Por que não tenho 64 registradores?
- Como ter registradores de 64 bits?

05/09 → • Convenções de uso dos registradores

• Exemplos de código



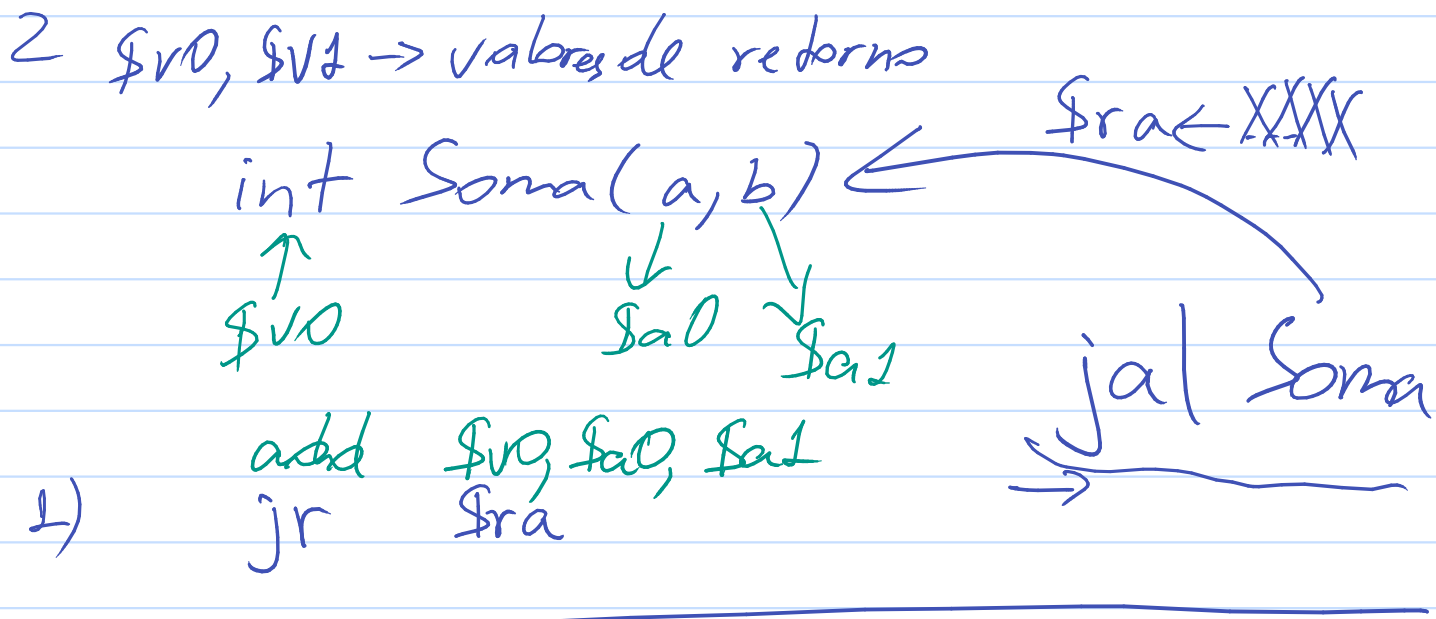
8

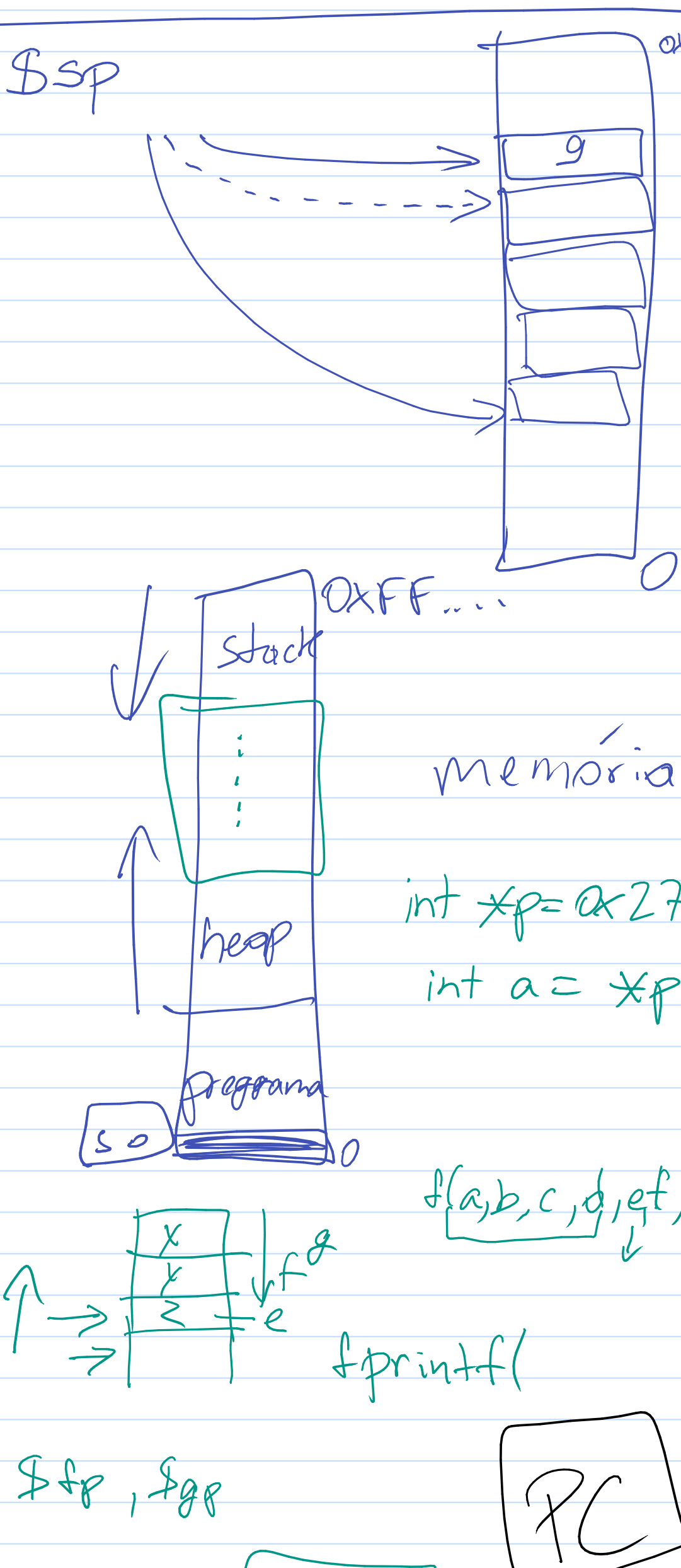


1g → $t, $s, $zero

4 $a0, $a1, $a2, $a3 → passagem de parâmetros

2 $v0, $v1 → valores de retorno





memória

```
int *p = 0x27ff1008;
int a = *p;
```

f(a, b, c, d, ef, g)

fprintf( );

$fp, $gp

RC

$k0, $k1

```
int SomaVetor (int *v, int n)
{                  →$s0   ↳$a0  ↳$a1
   int total = 0;
   int i; →$s1
   for (i=0; i<n; i++)
      total += v[i]; //total += *v++;
   return total;
}
```

```
      sw   $s0, -4($sp)
      sw   $s1, -8($sp)
      addi $sp, $sp, -8
      add  $s0, $zero, $zero
      add  $s1, $zero, $zero
laco: lw   $t0, 0($a0)
      add  $s0, $s0, $t0
      addi $a0, $a0, 4
      addi $s1, $s1, 1
      bne  $s1, $a1, laco
      add  $v0, $s0, $zero
      addi $sp, $sp, 8
      lw   $s1, -8($sp)
      lw   $s0, -4($sp)
      jr   $ra
```

sw
addi
add →
lw →
bne →
jr

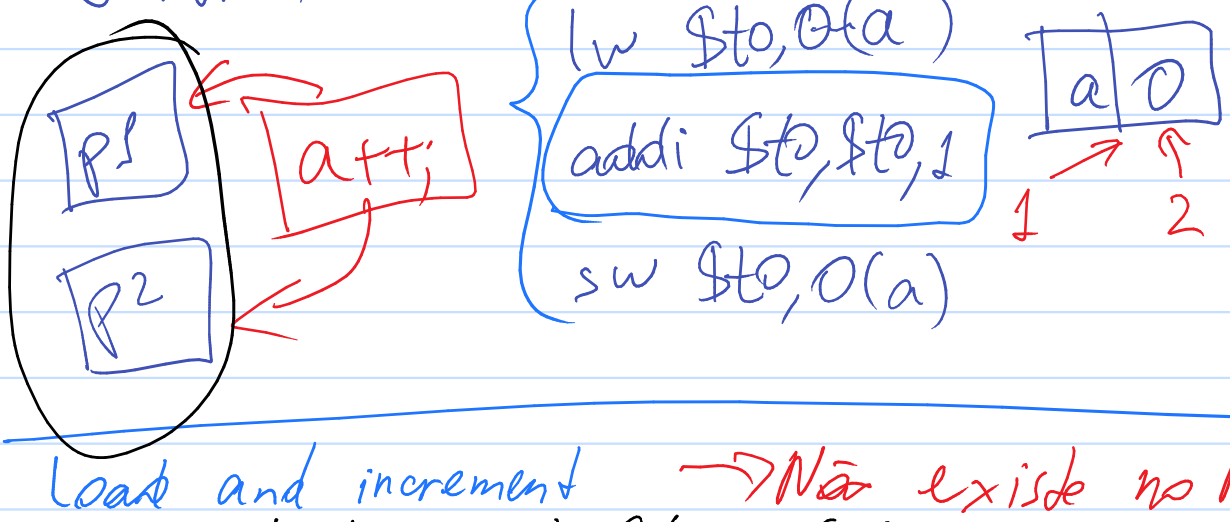
MC102 → Hello world

```
#include <stdio.h>
main()
{
   int i;
```

OpenMP

```
#pragma omp parallel for
for (i=0; i<1000; i++)
   puts("Hello World");
}
```

Instruções atômicas



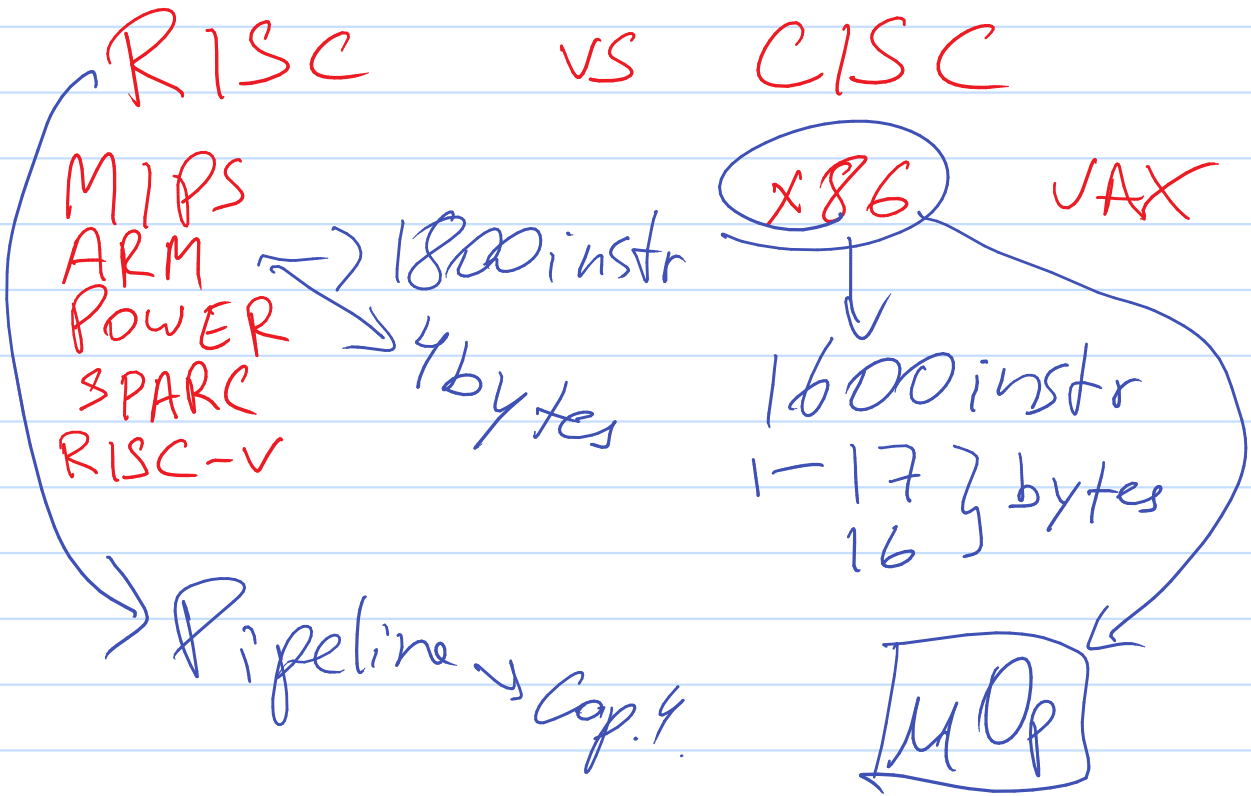
Load and increment → Não existe no MIPS

```
loadincrement $t0, 0(a)
[ operação exclusiva
sw $t0, 0(a)
```

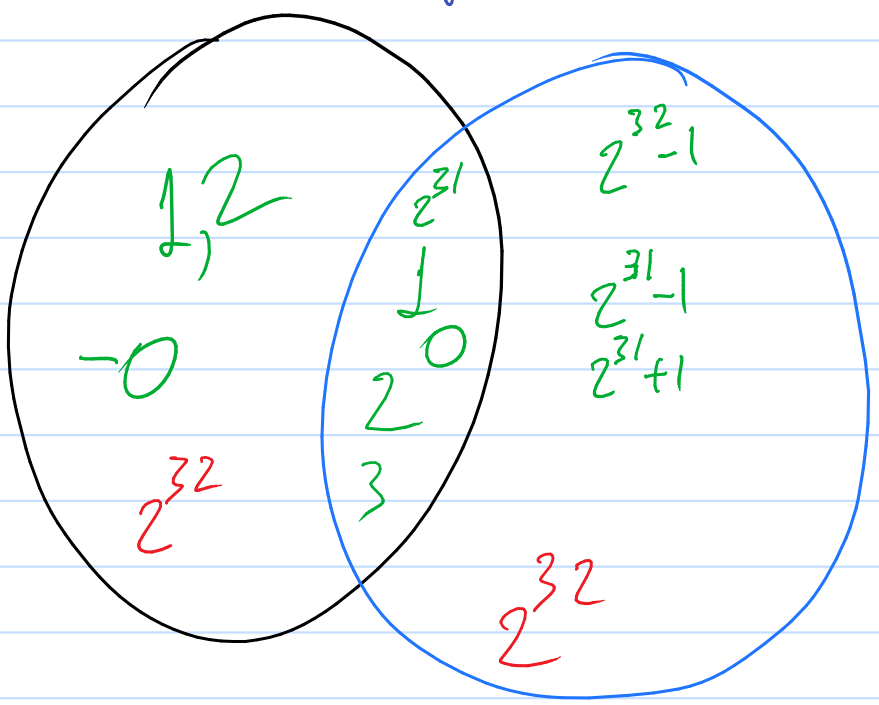
compare and swap

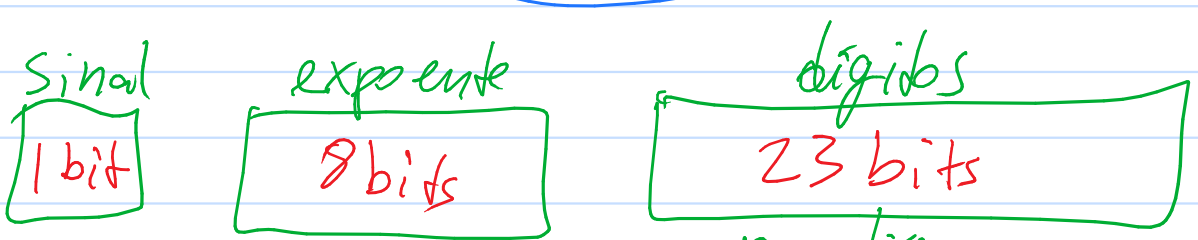
test and set

arquivo com várias seções – ELF
~ COFF


RISC
vs
CISC
MIPS
ARM
POWER
SPARC
RISC-V
1800 instr
4 bytes
x86
JAX
1600 instr
1-17 } bytes
16
Pipeline
Cap. 4
μOp

Números de ponto flutuante





± infinito

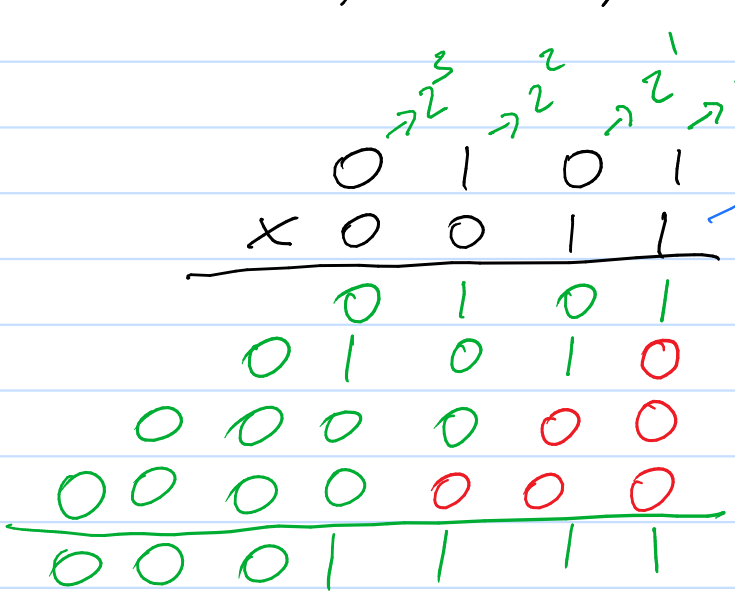
+0

-0

0,1

NaN

```
int i;
float f;
 :
i = f;    f = i;
```



total = 0

→ Para cada dígito
se dígito == 1
total += multiplicando << posição

Dados 3 vetores A, B, C de tamanho $n \geq 1$

escreva 1 programa que preencha A[i] com o menor valor entre B[i] e C[i]
Siga as convenções do MIPS

Protótipo: void MenorVetor(int A[], int B[], int C[], int n)

Implemente e use também int Menor(int a, int b)

Instruções mips: add, sub, mul, or, and, beq, bne, lw, sw, jr, jal, slt
além das versões com imediatos (addi, ...)

Regs: $t0, $t1 ... $a0, $a1 ... $ra
$s0, $s1 ... $v0, $v1 ... $sp

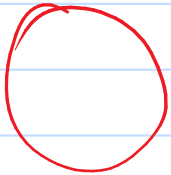
- Números binários.
- Soma / Subtração
- Multiplicação / Divisão
- Ponto flutuante

15 11 12 10

8 14 13 9

4 14 15 7

13 5 6 12



3 15 7 14

11 2 6 10

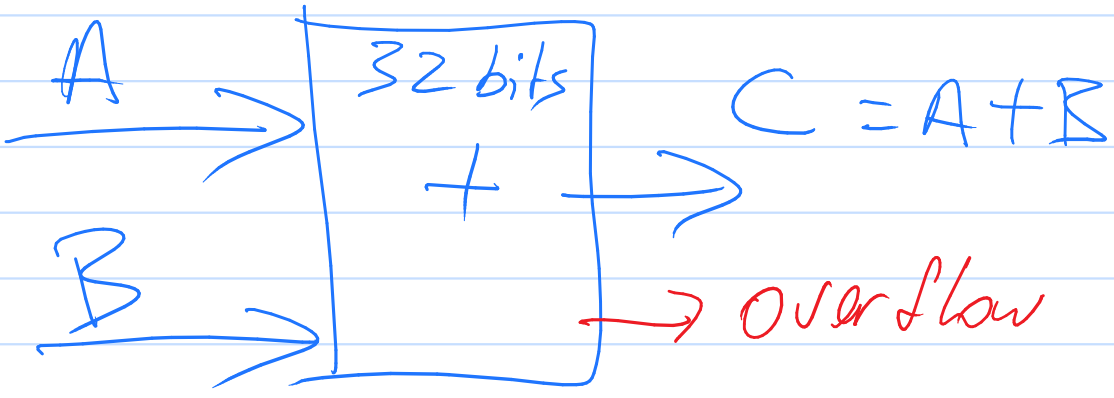
7 9 3 13

1 11 15 5

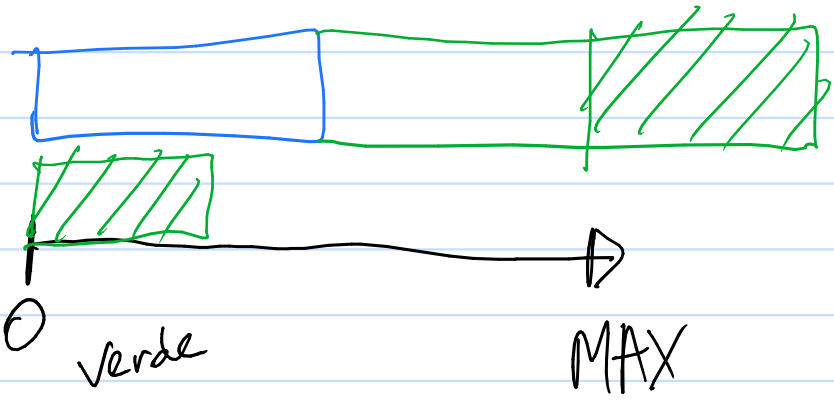
1

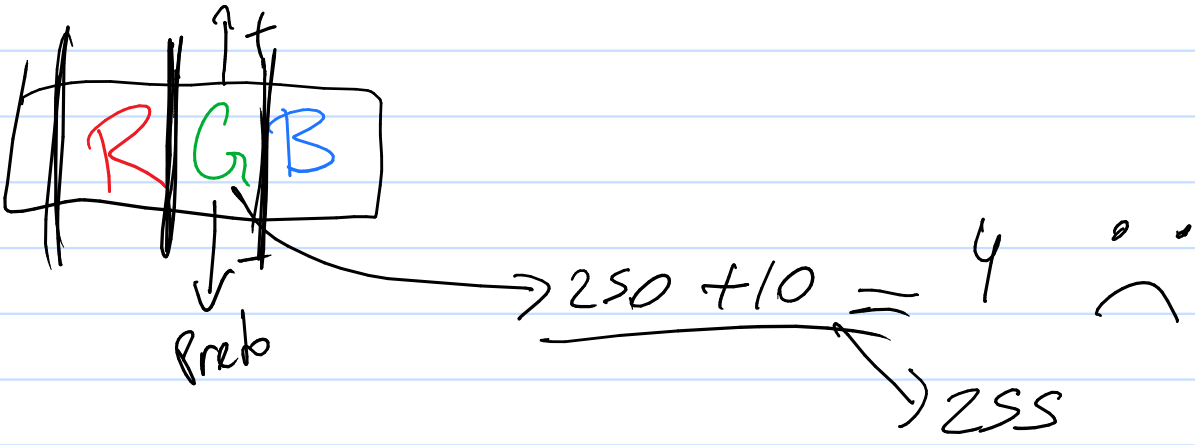
Overflow → { Felipe gostaria de saber que acontecen overflow.

ignorar overflow



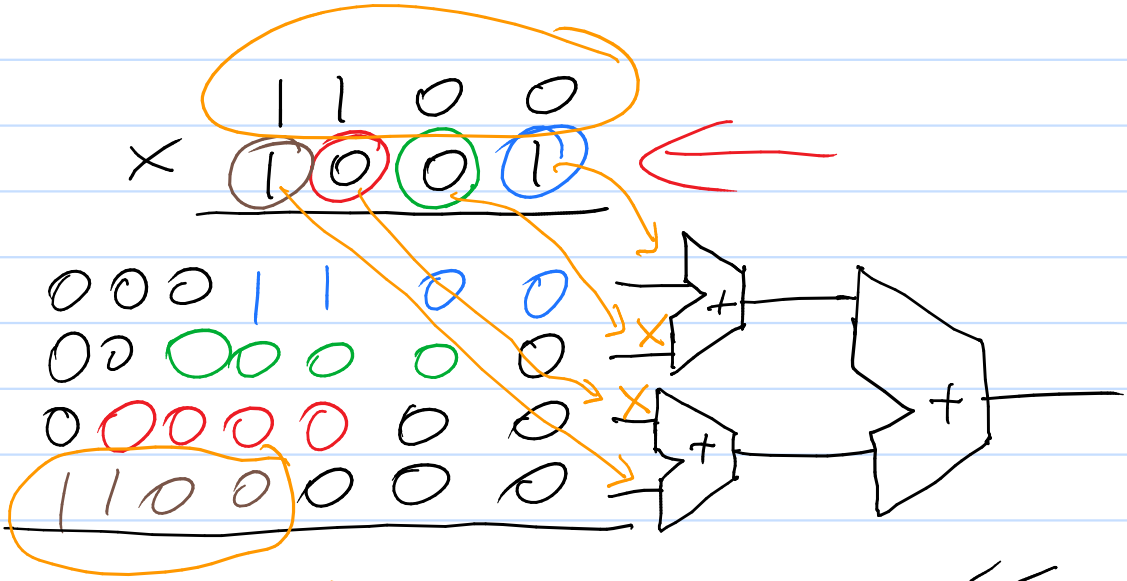
$C = A + B$

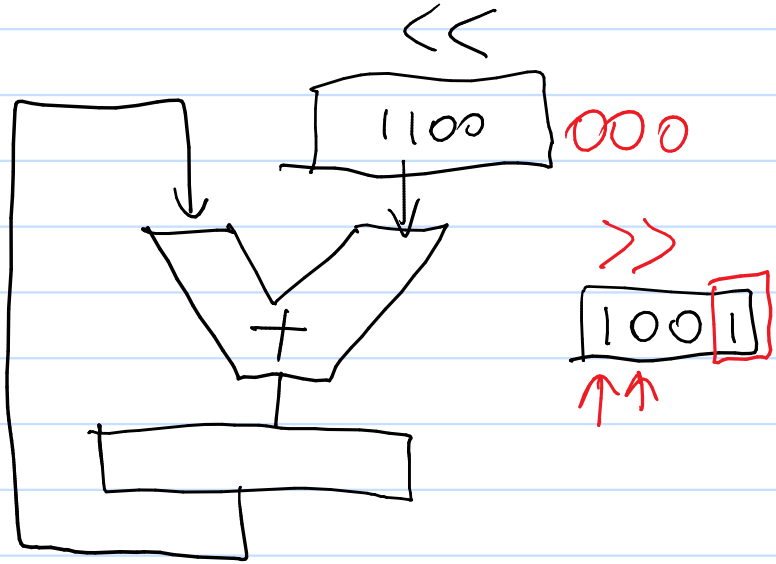




SSE AVX

MMX 3DNow Neon





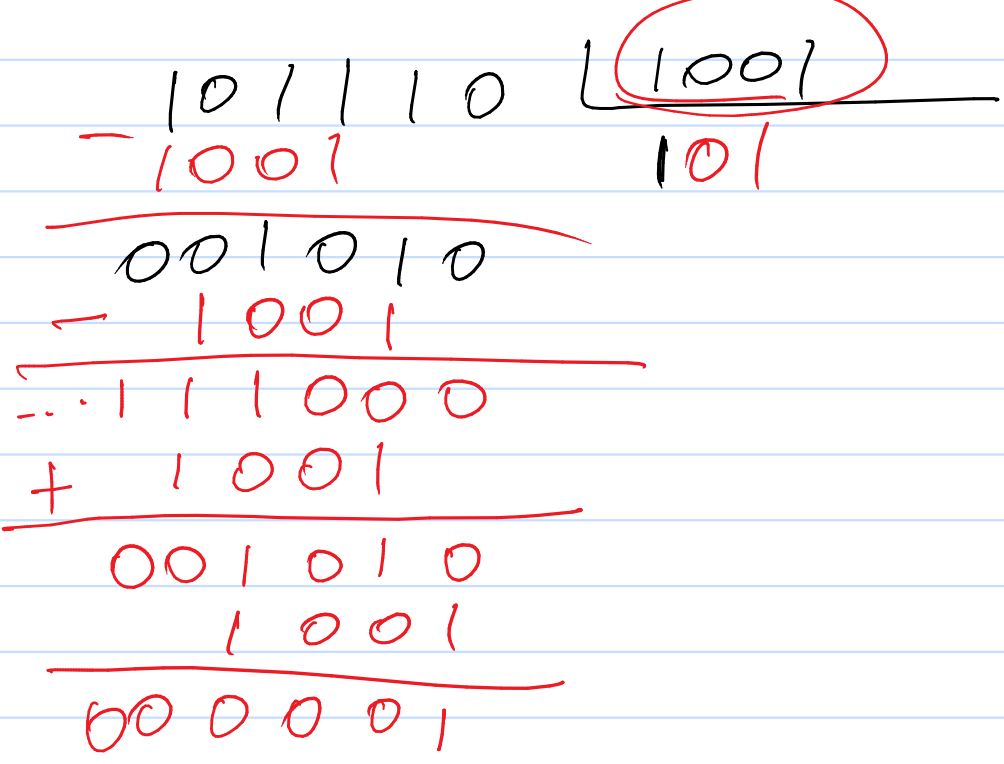
$X \rightarrow X0 \quad 2x$

$X00 \quad 4x$

$$\begin{array}{r} 111 \\ 1000X \\ \underline{-\ \ X\ \ } \end{array}$$

$X+$

$11000(1)$


101110
1001
101
001010
1001
111000
1001
001010
1001
000001

```
for (i=0; i<n; i++)
  A[i] = Menor(B[i], C[i]);
```

```
$v0                $a0      $a1
int   Menor (int a,  int b)

    slt  $t0, $a0, $a1
    bne  $t0, $zero, a_menor
    add  $v0, $a1, $zero
    jr   $ra
a_menor: add  $v0, $a0, $zero
    jr   $ra
```

```
                          $a0      $a1      $a2      $a3
void   MenorVetor(int A[], int B[], int C[], int n)
  sw   $s0, -4($sp)
  sw   $s1, -8($sp)
  sw   $s2, -12($sp)
  sw   $s3, -16($sp)
  sw   $ra, -20($sp)
  addi $sp, $sp, -20
  add  $s0, $a0, $zero
  add  $s1, $a1, $zero
  add  $s2, $a2, $zero
  add  $s3, $a3, $zero
loop: lw   $a0, 0($s1)      // B[i]
  lw   $a1, 0($s2)          // C[i]
  jal  Menor
  sw   $v0, 0($s0)          // A[i]
  addi $s0, $s0, 4
  addi $s1, $s1, 4
  addi $s2, $s2, 4
  addi $s3, $s3, -1
  bne  $s3, $zero, loop
  addi $sp, $sp, 20
  lw   $ra, -20($sp)
  lw   $s3, -16($sp)
  lw   $s2, -12($sp)
  lw   $s1, -8($sp)
  lw   $s0, -4($sp)
  jr   $ra
```

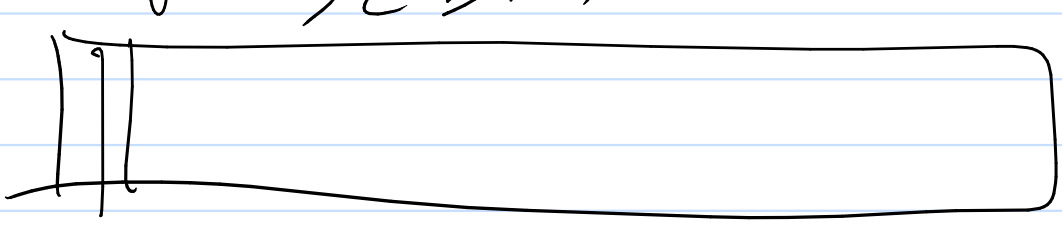
Quanto de memória você pode colocar num computador/processador?

Sistema de 32 bits → 2^{31} 2^{32}

32 bits → 64 bits → 2^{48}

00
01
10
11 → per. 32 bits

256 TB



lw $t0, 0($s0) → $s0 ← 0x80000000

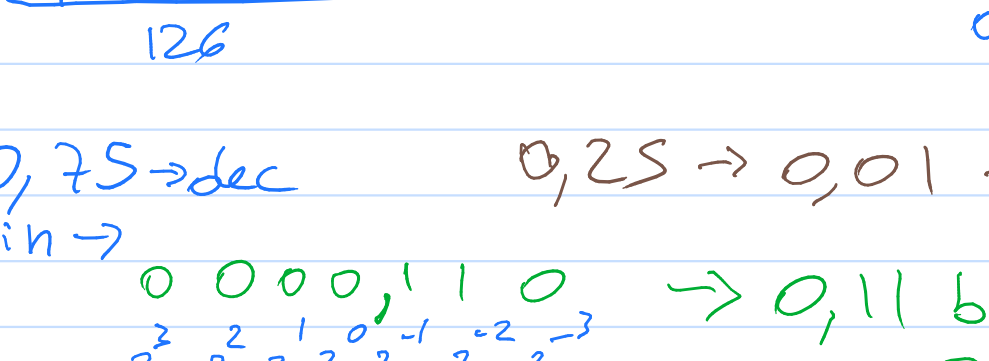
0xC

0,5 dec = $\frac{1}{2}$ = 2^{-1}

bin → 0,1

$+1,0 \times 2^{-1}$

-1 + 127



126 → 01111110

0,75 → dec

0,25 → 0,01 → $1,0 \times 2^{-2}$

bin →

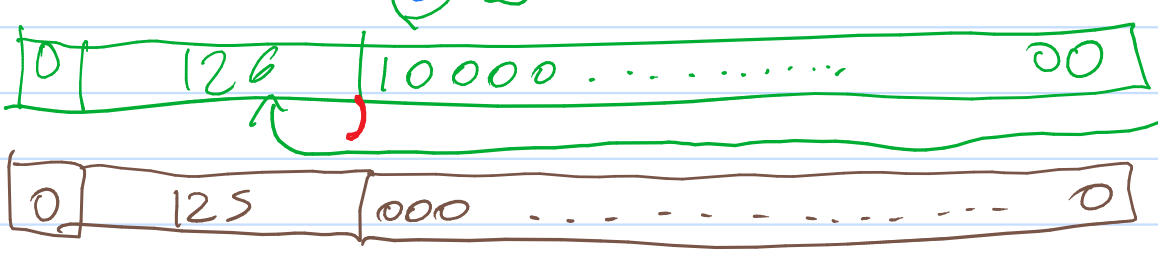
0 0 0 0 , 1 1 0

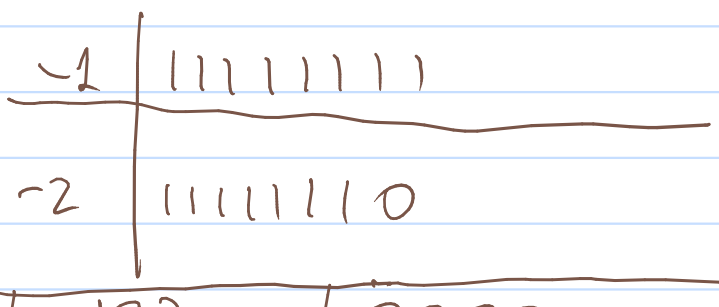
2^3 2^2 2^1 2^0 2^{-1} 2^{-2} 2^{-3}

8 4 2 1 $\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$

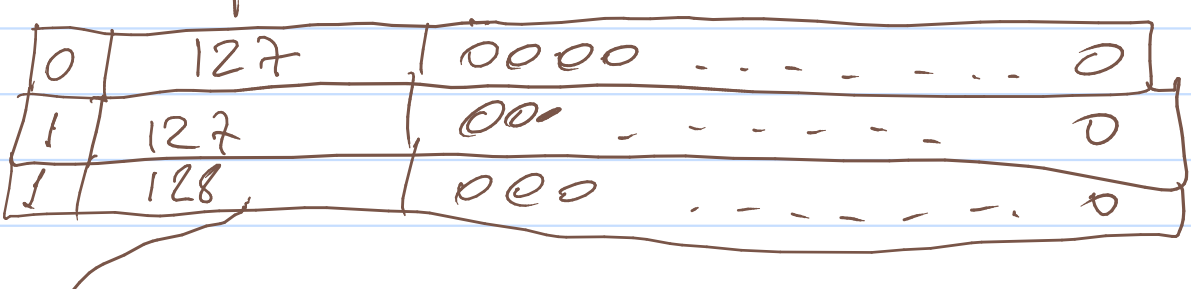
→ 0,11 binário

$1,1 \times 2^{-1}$





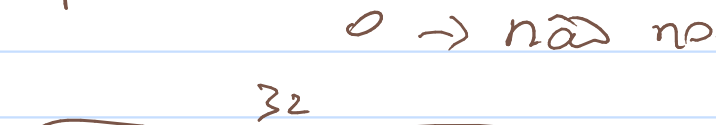
$1 = 1,0 \times 2^0$

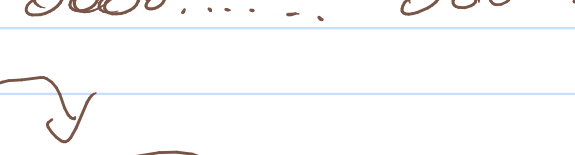


0,1 dec → dízima

Expentes 255 → $\pm\infty$ NaN

0 → não normalizados





$1,0 \times 2^{-127}$

$1,0 \times 2^{-128}$

2^{-129}

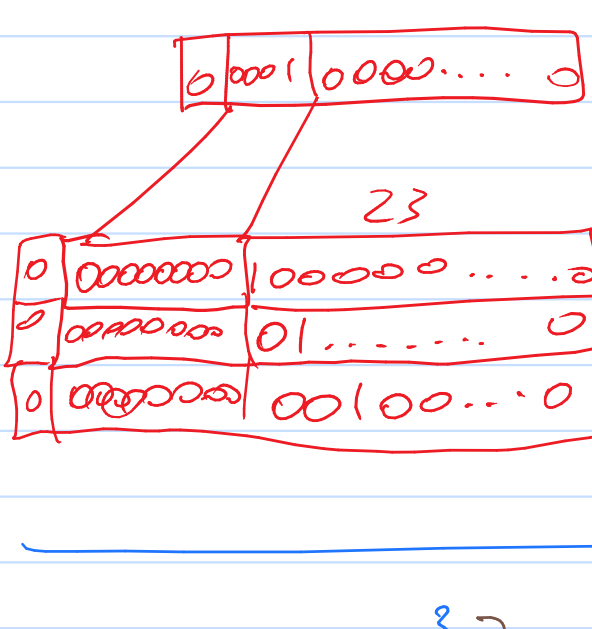
0,4

2^{-1} 2^{-2} 2^{-3} 2^{-4}

$\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$

0,25 0,125

$1,0 \times 2^{-126}$ ↓



+127 = 1

$\leftarrow 1,0 \times 2^{-127}$

$1,0 \times 2^{-128}$

$1,0 \times 2^{-129}$

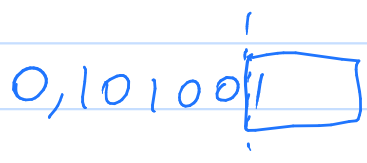
$2,37 \times 10^3$ } 237,00 x 10^1

$+ 5,02 \times 10^1$ 5,02 x 10^1

$2,37 \times 10^3$

$+ 0,05 \times 10^3$

$2,42 \times 10^3$



$X + 1 - X == 0$?