



$d_0 \ d_1 \ / \ d_2 \ d_3 \ d_4 \ d_5 \ d_6 \ d_7 \ d_8$

Nome: GABARITO

Matrícula: 13/0123456

Prova 1

(6.0) 1) Dado o programa em C ao lado.

(2.5)a) Respeitando a convenção do uso de registradores, compile eficientemente para Assembly RISC-V ISA RV32IMF as rotinas `main()` e `RAIZ()`.

Dica: Use os serviços do sistema (`ecall`) para implementar `scanf` e `printf`.

(1.5)b) Para um processador RISC-V, onde as instruções de acesso à memória (incluindo `flw` e `fsw`) possuem $CPI=2$, instruções de ponto flutuante $CPI=5$ e todas as outras instruções possuem $CPI=1$, com frequência de *clock* de 1GHz, se o valor lido do teclado for $a=1.1$:

```
float RAIZ(float n, float p){
    float m;
    m=(p+n/p)/2.0;
    if (fabs(m-p) < 1e-5*m)
        return m;
    else
        return RAIZ(n,m);
}

void main(){
    float a;
    scanf("%f",&a);
    printf("sqrt=%f",RAIZ(a,1.0));
}
```

(1.0)b.1) Qual a CPI média obtida para o seu procedimento `RAIZ`?

(0.5)b.2) Qual o tempo necessário à execução do seu procedimento `RAIZ`?

(1.0)c) Se fizermos $a=-1.0$ o que ocorrerá no seu programa?

(1.0)d) Se o compilador gcc usasse a diretiva `-O0`, qual seria a CPI média ótima do procedimento `RAIZ`?

(3.0) 2) Considere o procedimento PROC abaixo sendo chamado com o argumento $a0=0xFFFFFFFF$.

PROC: <code>li t0, 18</code>	<u>0x01200293</u>	<u>0,4</u>
<code>add a0, t0, a0</code>	<u>0x00A28533</u>	<u>0,4</u>
<code>bne a0, zero, FIM</code>	<u>0x00051463</u>	<u>0,4</u>
<code>ori a0, a0, 99</code>	<u>0x06356513</u>	<u>0,4</u>
FIM: <code>ret</code>	<u>0x00008067</u>	<u>0,4</u>

(2.0)a) Escreva o código em linguagem de máquina, em hexadecimal, ao lado de cada instrução acima.

(1.0)b) Se o valor retornado no registrador `a0` for tratado como uma instrução da ISA RISC-V, qual o mnemônico completo desta instrução? L1: bcs zero, zero, L1

(2.0) 3) As pseudo-instruções são uma poderosa ferramenta fornecida pelo montador que facilita a vida do programador em linguagem Assembly. Supondo que as instruções abaixo não existissem na ISA RV32I, e fossem consideradas pseudo-instruções, implemente-as utilizando instruções reais.

(1.0) a) `ecall` # `PC=UTVECT` e `EPC=PC+4`

`jalr gp, tp, 0`

(1.0) b) `uret` # `PC=EPC`

`jalr zero, gp, 0`

Considere que `UTVECT` e `EPC` são os registradores `tp` e `gp` respectivamente.

(0.5) 4) Responda:

(0.5) a) Por que você acha que OAC é difícil? Dê sugestões sobre como melhorar a disciplina. Se você é aluno do BCC, escreva sobre a influência de ISC no seu caminho até aqui (ao longo do curso e em outras disciplinas) (ISC é válida? Não adianta em nada? Tempo perdido? Qual a sua percepção? Sugestões?).

Muita hora nessa calma!
Boa Sorte!

1ª PROVA GABARITO

1) a) Folha em Anexo

A compilação eficiente identifica que o algoritmo não é realmente recursivo.

```

Float RAIZ (Float n, Float p) {
    Float m;
    L1: m = (p+n/p)/2;
    IF (Abs(m-p) < 1e-5 * m)
        return m
    else { p = m;
           GOTO L1;
         }
}

```

1	2
1,1 1,0	1,1 1,05
m = 1,05	m = 1,0488
2,05	3,0011
Falso	Falso
p = 1,05	p = 1,0488
3	
1,1 1,0488	
m = 1,0488	
6×10^{-7}	
verdadeiro	
FM	

b) depende do código do aluno.

Para o Gabarito

$q = 1,1 \rightarrow 3$ iterações

$N =$ instruções executadas : 44

$M =$ de ciclos : 159

b.1)

$$\text{Logo } CPI = \frac{159}{44} = 3,61$$

1,0

0,5

b.2) $t_{\text{exec}} = N \text{ ciclos} \times T = 159 \times 1 \text{ ns} = 159 \text{ ns}$

c) $a = -1,0$

no algoritmo $h = -1$ $p = 1$ $n = -1$ $d = 0$

$m = 0$

$\text{fabs}(0 - 1) < 1E-5$

false

$p = m$

$$m = (0 - \frac{1}{0}) / 2$$

$m = -\infty$ infinito

MAS NFO GERA EXCEÇÃO!
NO RISC-V

$h = -1$ $p = -\infty$

$$m = (-\infty - \frac{1}{-\infty}) / 2 = -\infty$$

e m fica em $-\infty$ / sempre
Logo fica em loop ∞ .

$$\text{fabs}(m \cdot p) = \text{fabs}(-\infty - (-\infty))$$

indeterminação! NaN

RESPOSTAS VÁLIDAS:

→ Ocorre divisão por zero e trava (Acito) ex

→ Fica em loop infinito ✓

d) COMPILADOR SUPER ESPERTO identifica que
o procedimento calcula a RAIZ QUADRADA
e compila como

RAIZ: fsqrt fadd , fadd
pr ra

$\text{CPI} = 5$

$\text{CPI} = 1$

$$\text{Logo } \text{CPI} = \frac{5+1}{2} = 3$$

(1,0)

Questão 1

.data

NUM: .float 1.0e-5

STRING: .string "sqrt="

.text

```
MAIN: li a7, 6          # le teclado      0,1
      ecall            # a = fa0
```

```
      la a0,STRING      # imprime "sqrt="  0,1
      li a7,4
      ecall
```

```
      li t0,1
      fcvt.s.w fa1,t0    # fa1=1.0      } 0,2
      jal RAIZ           # retorno em fa0
```

```
      li a7,2           # printf float    0,1
      ecall
```

```
      li a7,10          # exit          0,1
      ecall
```

```
RAIZ:  fdiv.s ft0,fa0,fa1    # ft0 = n/p
      fadd.s ft0,ft0,fa1    # ft0 = p+n/p
      li t1,2
      fcvt.s.w ft3,t1       # ft3=2.0
      fdiv.s ft0,ft0,ft3    # ft0 = m=(p+n/p)/2.0 } 0,5

      fsub.s ft1,ft0,fa1    # ft1 = m-p
      fabs.s ft1,ft1        # ft1 = fabs(m-p) } 0,2

      la t0,NUM
      flw ft2,0(t0)         # ft2 = 1e-5
      fmul.s ft5,ft0,ft2    # ft5 = 1e-5*m } 0,2

      flt.s t1,ft1,ft5     # fabs(m-p) < 1e-5*m ? 0,2

      fmv.s fa1,ft0        # fa1 = m          0,1

      beq t1,zero,RAIZ     # não: volta para RAIZ 0,1

      fmv.s fa0,ft0        # fa0 = m
      ret                  # retorna          } 0,2
```