

Arquitectura de Computadores I

Exercícios

Licenciatura em Engenharia Informática

Aula Prática #3 (Instruções aritméticas e lógicas RISC-V)

Nesta aula prática será apresentado o simulador RARS: código fonte, instruções assembly geradas, código máquina, endereços de memória e registos do processador. Irá fazer programas muito simples com instruções aritméticas e lógicas.

1. Escreva código em Assembly para calcular a seguinte expressão:

- $3 + 4 + 5$ (coloque o valor 3 no registo `t0` e depois some sucessivamente com 4 e com 5)
- Sabendo que a ordenação de bytes da arquitectura RISC-V é *little endian*, desenhe o mapa de memória do programa com o código máquina das instruções (veja o código máquina no simulador RARS).

Address	Byte
:	:
0x00400003	?
0x00400002	?
0x00400001	?
0x00400000	?

- Como seria o mapa de memória se a ordenação fosse *big endian*?

2. Implemente troços de código Assembly que implementem as seguintes instruções.

- `t1 = 2 * t0 + 1;`
- `t1 = 9 * t0 + 5;`
- `t1 = t0 - 1;` (haverá necessidade de uma instrução `subi`?)
- `t2 = 2 * t0 + 3 * (t0 - t1)`

3. Implemente um conjunto de instruções lógicas que inverta a ordem dos bytes do registo `t0`. Pode usar outros registos auxiliares.

4. Escreva um troço de código para calcular o simétrico de um número guardado no registo `t0`. O resultado deve ficar em `t1`.

5. Complete as instruções seguintes de modo a que cada uma delas coloque o registo `t0` a zero:

```

add t0, --, --
addi t0, --, --
sub t0, t0, --
or t0, --, --
and t0, t0, --
xor t0, t0, --
ori t0, --, --
sll t0, --, --

```

6. Sem usar o simulador, analise o seguinte troço de código e indique o valor final dos registos `t0`, `t1`, `t2`.

```
lui  t0, 0x12345  
addi t0, t0, 0x678  
ori  t1, zero, 0x123  
slli t1, t1, 20  
ori  t1, t1, 0x678  
sub  t2, t0, t1
```