**INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA**

Licenciatura em Engenharia Informática e de Computadores

Logo

Description automatically generated

Arquitetura de Computadores

**Relatório Trabalho 2**

Trabalho realizado por:

|  |  |
| --- | --- |
| Nome: André Monteiro | Nº 43842 |
| Nome: Constança Costa | Nº 50541 |
| Turma: LEIC24D  Docente: Rui Duarte |  |

2022 / 2023 Semestre Verão

15 de abril de 2023

**Índice**

1. **Introdução**
2. **Desenvolvimento do trabalho**
   1. Análise da microarquitectura
   2. Codificação das instruções
   3. Projeto do descodificador de instruções
   4. Codificação de programas em linguagem máquina
3. **Conclusão**

# **Introdução**

Este relatório remete para o acompanhamento da construção de uma computer network proposta pelo professor da cadeira. Neste momento será descrito a execução da 1º fase deste projeto, que se foca na criação de um webserver.

Temos como objetivo analisar o funcionamento de uma conexão HTTP com base no protocolo TCP, protocolo este que é usado para realizar uma conexão estável e sem perda de informação entre um webserver, disponibilizado pelo enunciado e o cliente (nós), cada um operando em locais distintos.

# **Desenvolvimento do trabalho**

***2.1.*** Análise da microarquitectura

1)

A afirmação é verdadeira, pois como indicado no enunciado o processador é de ciclo único e como se pode observar na imagem a memória para dados e programa estão separadas.

Diagram, schematic

Description automatically generated

2)

-O bloco EXT serve para a extensão(aumentar) do número de bits, ou seja, 6-8

-Sinal SE indica o valor que os bits estendidos será 0 ou 1

-bne

Sendo a memória de 256 endereços (2^8, ou seja, cada endereço de 8 bits), como a label é de 6 bits e são necessários 8 para aceder à memória é preciso extender o valor da label de 6 a 8 bits.

-mov

8 registos cada com 8 bits, valor imediato ao ser de 3bits para se poder guardá-lo num registo este irá ser estendido com o valor 0, por se estar a tratar de um número natural, para o valor pretendido (8bits).

3)

Vantagens:

bne

- Condicional por isso só executa caso a condição se verifique

-Tendo em conta a fig 1 podemos usar labels não estando restritos a colocar o valor da label num registo antes de executar o salto

b

- Mais rápido

Desvantagens:

bne

-Demora mais ciclos, devido à subtração, o incremento do PC e a verificação do valor da flag Z (0)

b

-Executa sempre

***2.2*** Codificação das instruturas

1)

Graphical user interface, text, application

Description automatically generated



2)

|  |  |
| --- | --- |
| Instrução | opcode |
| B | 000 |
| Bne | 001 |
| Ldr | 010 |
| Str | 011 |
| Add | 100 |
| Cmp | 101 |
| Push | 110 |
| Mov | 111 |

***2.3*** Projeto do descodificador de instruçõe

1)

Pela microarquitetura do processador ser de ciclo único, o valor da label que é lido é o offset calculado pela expressão Offset=label-PC. ??

Graphical user interface, application, table, Excel

Description automatically generated

2)

A ROM vai conter como inputs 16 linhas de 0000 a 1111 (2^4) onde:­­­

* input 0000 => output 0000011111;
* input 0001 => output 0000011111;
* input 0010 => output 0000110111;
* input 0011 => output 0000110111;
* input 0100 => output 0000001011;
* input 0101 => output 0000001011;
* input 0110 => output 1000101001;
* input 0111 => output 1000101001;
* input 1000 => output 1010101010;
* input 1001 => output 1010101010;
* input 1010 => output 0000100010;
* input 1011 => output 0000100010;
* input 1100 => output 0100000011;
* input 1101 => output 0100000011;
* input 1110 => output 1000000011;
* input 1111 => output 0000000011;

O input é composto pelos sinais de entrada do Instruction Decoder (opcode e flag Z), onde

input (3) = opcode (2),

input (2) = opcode (1),

input (1) = opcode (0),

input (0) = Z,

e o output é composto pelos sinais de saída (SO, SI, SS, SE, SD, ER, EP, nRD, nWR), onde

output (9) = SO,

output (8) = SI,

output (7) = SS,

output (6) = SE,

output (5) = SD (1),

output (4) = SD (0),

output (3) = ER,

output (2) = EP,

output (1) = nRD

e output (0) = nWR.

3)

Número de endereços = 16 = 2^4 endereços

Número de bits em cada palavra 10 bits

Como Capacidade = número de endereços \* número de bit de cada palavra,

Temos,

Capacidade = 16 x 10 = 160 bits / 2 = 20 bytes

***2.4*** Codificação de programas em linguagem máquina

1)

O troço de código apresentado adiciona os valores dos primeiros quatro endereços da memória, guardando depois o resultado desta soma na quarta posição da memória.

2)

Usando a tabela do 2.2.2 como base para preencher a tabela do código de máquina.Graphical user interface, application, table, Excel

Description automatically generatedGraphical user interface, application, table, Excel

Description automatically generated

# Conclusão

Com a realização deste trabalho pudemos observar as operações entre servidor e cliente. Neste momento somos capazes de entender o código de resposta retornado pela conexão servidor ao cliente. Pudemos apreender que existem vários cabeçalhos, os principais são os cabeçalhos de resposta (enviados pelo servidor) e os cabeçalhos de requisição (enviados pelo cliente).