	Laboratório de Arquitetura de Computadores	Data 29/09/19
	Relatório de Experimentos	Módulo: 1

Nome: Giovanne Prestes Dias

RA 171029

Título: Ambiente de simulação – Cpu Board.

Objetivos:

- Adquirir conhecimentos em arquitetura de computadores;
- Familiarização com o módulo CPU Board Studio e CPU Board Lab;
- Instruções básicas usando linguagem Assembly;
- Manipulação de alguns periféricos.

Material Utilizado:

- Softwares CPU Board Lab e Cpu Board Studio.

Relatório:

Projeto a ser desenvolvido no software Cpu Board:

Criar um relógio que conte os minutos "00 até 59" e os segundos "00 até 59", quando chegar no final 59 59, recomeçar;

- 1) Introdução com a apresentação dos assuntos abordados no experimento;

Criada na década de 50, o Assembly foi das primeiras linguagens de programação a aparecer. Ela usa uma sintaxe complicada e "exageradamente" difícil, isto porque, antes da década de 50 os programadores de máquinas tinham que escrever instruções em código binário, qualquer coisa como: 0110010110011011010110011010111010110101 ... Para escrever uma instrução. Na verdade, o Assembly foi criado para facilitar o uso dessa tarefa, mas é considerado uma linguagem de baixo nível, pois tudo o que o processador interpreta tem que ser descrito pelo programador. Assim o código acima seria "add EAX" em Assembly. Bastava apenas, depois de estar concluída a escrita do código, rodar o compilador e tínhamos o programa.

Cada arquitetura de computador tem seu próprio código de máquina, e cada montador gera códigos para uma arquitetura específica. Cada um desses montadores tem sua própria versão de código Assembly, que pode diferir ao uso de registradores, representação de números, ou até mesmo instruções mnemônicas. E isso pode dificultar um pouco na portabilidade do código, tendo em vista que o mesmo precisaria ser reescrito para poder ser montado para outra arquitetura.


Vantagens: programas extremamente rápidos e pequenos

Desvantagens: tempo de desenvolvimento lento e sujeito a erros; código preso a uma arquitetura.

- 2) Procedimento experimental executado (quais foram os passos executados para a criação do projeto);

Primeiramente, tivemos que codificar a lógica através do software Cpu Board Studio, o código em assembly e utilizando:

- Registrador de endereço (MAR) de 12 bits.
- Registrador de dados de (MBR) de 8 bits.
- Contador de programa (PC) de 12 bits.
- Registrador de instruções (IR) de 16 bits.
- Registrador acumulador (AC) de 8 bits.

	Laboratório de Arquitetura de Computadores	Data 29/09/19
	Relatório de Experimentos	Módulo: 1

- Registrador com flags de controle (SF, OF, ZF, GF e LF).

E com o seguinte conjunto de instruções:

Mnemônico	Descrição	Opcode	Operação
LDA	Load Accumulator	0000	$AC = M[\text{Endereço}]$
STR	Store	0001	$M[\text{Endereço}] = AC$
ADD	Add	0010	$AC = AC + M[\text{Endereço}]$
SUB	Subtract	0011	$AC = AC - M[\text{Endereço}]$
AND	Logical AND	0100	$AC = AC \& M[\text{Endereço}]$
OR	Logical OR	0101	$AC = AC M[\text{Endereço}]$
NOT	One's Complement	0110	$AC = \sim AC$
XOR	Exclusive OR	0111	$AC = AC \wedge M[\text{Endereço}]$
JMP	Indirect Jump	1000	Desvio indireto
JE	Jump if Equal	1001	Desvia se ZF = 1
JL	Jump if Less Than	1010	Desvia se LF = 1
JG	Jump if Greater Than	1011	Desvia se GF = 1
JLE	Jump if Less or Equal	1100	Desvia se ZF = 1 ou LF = 1
JGE	Jump if Greater or Equal	1101	Desvia se ZF = 1 ou GF = 1
HLT	Halt	1110	Para o processamento
NOP	No Operation	1111	Sem operação

Figura 1: Conjunto de instruções.

E o seguinte formato de instrução:

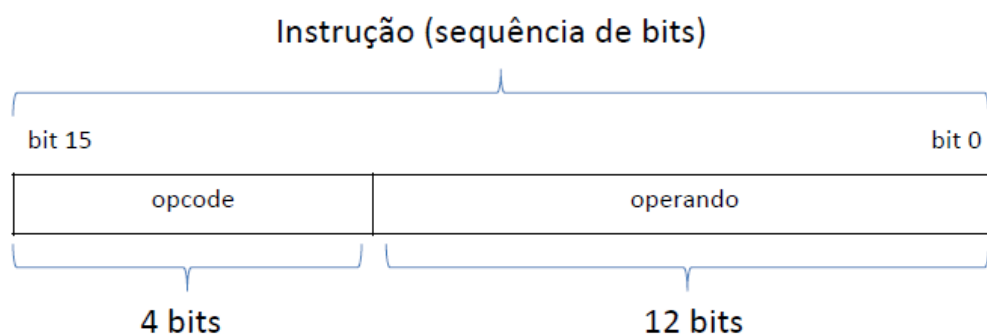


Figura 2: Formato de instrução.

3) Análise dos resultados (acrescentar comentários sobre o funcionamento do projeto);

Com as instruções informadas acima e o desafio proposto, cheguei em uma lógica como resultado descrito a seguir:

```
ram 100 59
ram 101 1
ram 103 0
cseg
org Inicio          //Inicio

lda ac, 103          //Carrega 0 no acumulador
str 102, ac          //Coloca no endereço 102, o que t no acumulador 102 -> 0

org minutos
lda ac, 102          //AC=0
str 4092, ac         //M[4092]=0
add ac, 101          //AC = AC + 1
str 102, ac          //M[102] = 1
lda ac, 103          //AC = 0
str 104, ac          //M[104] = 0

org segundos
lda ac, 104          //AC = 0
str 4093, ac         //M[4093] = 0
add ac, 101          //AC = AC + 1
str 104, ac          //M[104] = 1
sub ac, 100          //AC = AC - 59
jle segundos        //JUMP if LOWER or EQUALS

lda ac, 102          //AC = 1
sub ac, 100          //AC = AC - 59
jle minutos

jmp Inicio
```

4) Conclusão.

Por fim, concluí que para o desafio proposto este código alcançou o esperado e, portanto o relógio conta os minutos de “00 até 59” e os segundos de “00 até 59”, e quando chega no final “59 e 59” ele recomeça.

Ao final deste experimento, obtivemos um conhecimento básico sobre assembly, mais conhecimento em arquitetura de computadores, me familiarizei com os módulos CPU Board Studio e CPU Board Lab, conseguindo assim obter o conhecimento básico de manipulação de alguns periféricos conforme o proposto neste experimento.