Trabalho 2 – Arquitetura de Computadores

DATA DE ENTREGA: 15/07/2024
TRABALHO DEVERÁ SER FEITO EM DUPLAS

Construa um simulador de um processador simples, que tenha suporte à unidade de controle e memória principal. O processador deverá suportar as instruções listadas na tabela abaixo. O simulador deve ser escrito em linguagem C, C++ ou Python, a escolha da linguagem fica a critério da dupla.

Funcionamento do simulador: O usuário deverá listar a sequência de instruções em um arquivo de entrada (no máximo 32 linhas). O arquivo de entrada deve ter o nome de `entrada.txt`

Como resultado, após a execução do seu simulador, seu simulador deverá gerar três arquivos de saída:

- `unidade_controle.txt` (com os valores de `PC` e `IR` em cada linha, respectivamente)
- `banco_registradores.txt` (com os valores de `R0`, `R1`, `R2` e `R3` em cada linha, respectivamente)
- `memória_ram.txt` (com até 32 linhas, cada linha referente à posição da memória – de 0 até 31)

O arquivo de saída deverá representar o estado **FINAL** da execução, ou seja, o conteúdo dos três arquivos de saída deve estar preenchido com os valores FINAIS, após a execução de todas as instruções contidas no arquivo `entrada.txt`. Não há necessidade do programa mostrar na tela a execução das instruções passo-a-passo. Seu código deve estar endentado e comentado.

Tabela de instruções a serem suportadas:

Data Movement Instructions:			
Assembly Language Instruction:	Example:	Meaning:	Machine Language Instruction:

`LOAD [REG] [MEM]`
`STORE [MEM] [REG]`
`MOVE [REG1] [REG2]`
	`LOAD R2 13`
	`STORE 8 R3`
	`MOVE R2 R0`
		`R2 = M[13]`
		`M[8] = R3`
		`R2 = R0`
			`1 000 0001 0 RR MMMMM`
			`1 000 0010 0 RR MMMMM`
			`1 001 0001 0000 RR RR`

Arithmetic and Logic Instructions:			
Instruction:	Example:	Meaning:	Machine Language Instruction:

`ADD [REG1] [REG2] [REG3]`
`SUB [REG1] [REG2] [REG3]`
`AND [REG1] [REG2] [REG3]`
`OR [REG1] [REG2] [REG3]`
	`ADD R3 R2 R1`
	`SUB R3 R1 R0`
	`AND R0 R3 R1`
	`OR R2 R2 R3`
		`R3 = R2 + R1`
		`R3 = R1 - R0`
		`R0 = R3 & R1`
		`R2 = R2 | R3`
			`1 010 0001 00 RR RR RR`
			`1 010 0010 00 RR RR RR`
			`1 010 0011 00 RR RR RR`
			`1 010 0100 00 RR RR RR`

Branching Instructions:			
Instruction:	Example:	Meaning:	Machine Language Instruction:

`BRANCH [MEM]`
`BZERO [MEM]`
`BNEG [MEM]`
	`BRANCH 10`
	`BZERO 2`
	`BNEG 7`
		`PC = 10`
		`PC = 2 IF ALU RESULT IS ZERO`
		`PC = 7 IF ALU RESULT IS NEGATIVE`
			`0 000 0001 000 MMMMM`
			`0 000 0010 000 MMMMM`
			`0 000 0011 000 MMMMM`

Other Instructions:			
Instruction:	Example:	Meaning:	Machine Language Instruction:
`NOP` `HALT`	`NOP` `HALT`	`Do nothing.` `Halt the machine.`	`0000 0000 0000 0000` `1111 1111 1111 1111`