UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA

CARLOS AUGUSTO PEREIRA MAIA

RELATÓRIO - PROJETO DE MICROPROGRAMAÇÃO

PROFESSOR: DR. EWERTON MONTEIRO SALVADOR
DISCIPLINA: ARQUITETURA DE COMPUTADORES

João Pessoa - PB
2020

INTRODUÇÃO

Como avaliação dos conhecimentos adquiridos com o curso de arquitetura de computadores é cobrado um relatório da implementação de uma instrução no nível da microarquitetura de um microprocessador MIPS. A funcionalidade a ser implementada é substituir a instrução BEQ (*branch if equal*) pela instrução BNE (*branch if not equal*).

METODOLOGIA

Antes de realizar qualquer alteração, ocorreram estudos no caminho de dados da microarquitetura MIPS. A figura 1 representa o caminho de dados estudado disponibilizado nos slides do professor.

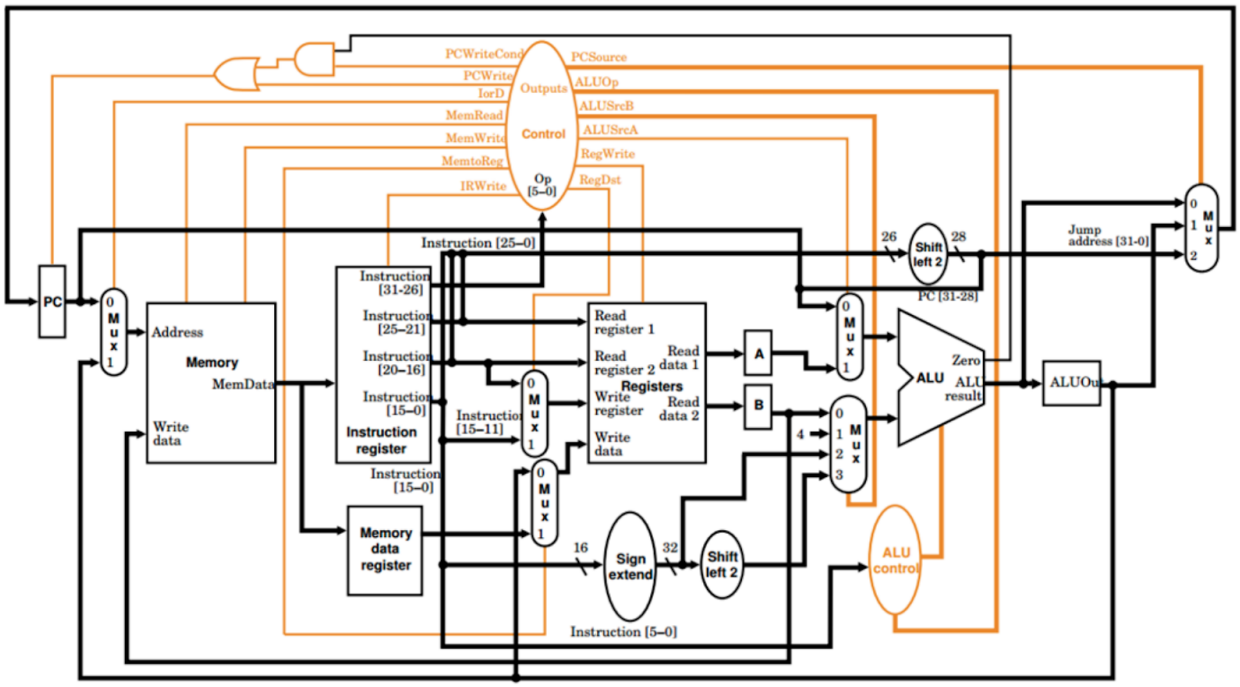

Figura 1 - Caminho de dados - MIPS.

Também foi disponibilizado pelo professor um diagrama de uma máquina de estados que representa o controle do caminho de dados. Esse controle implementa as instruções LW, SW, instruções R (Add, Sub, And, Or, Slt), BEQ e J. A figura 2 representa esse diagrama.

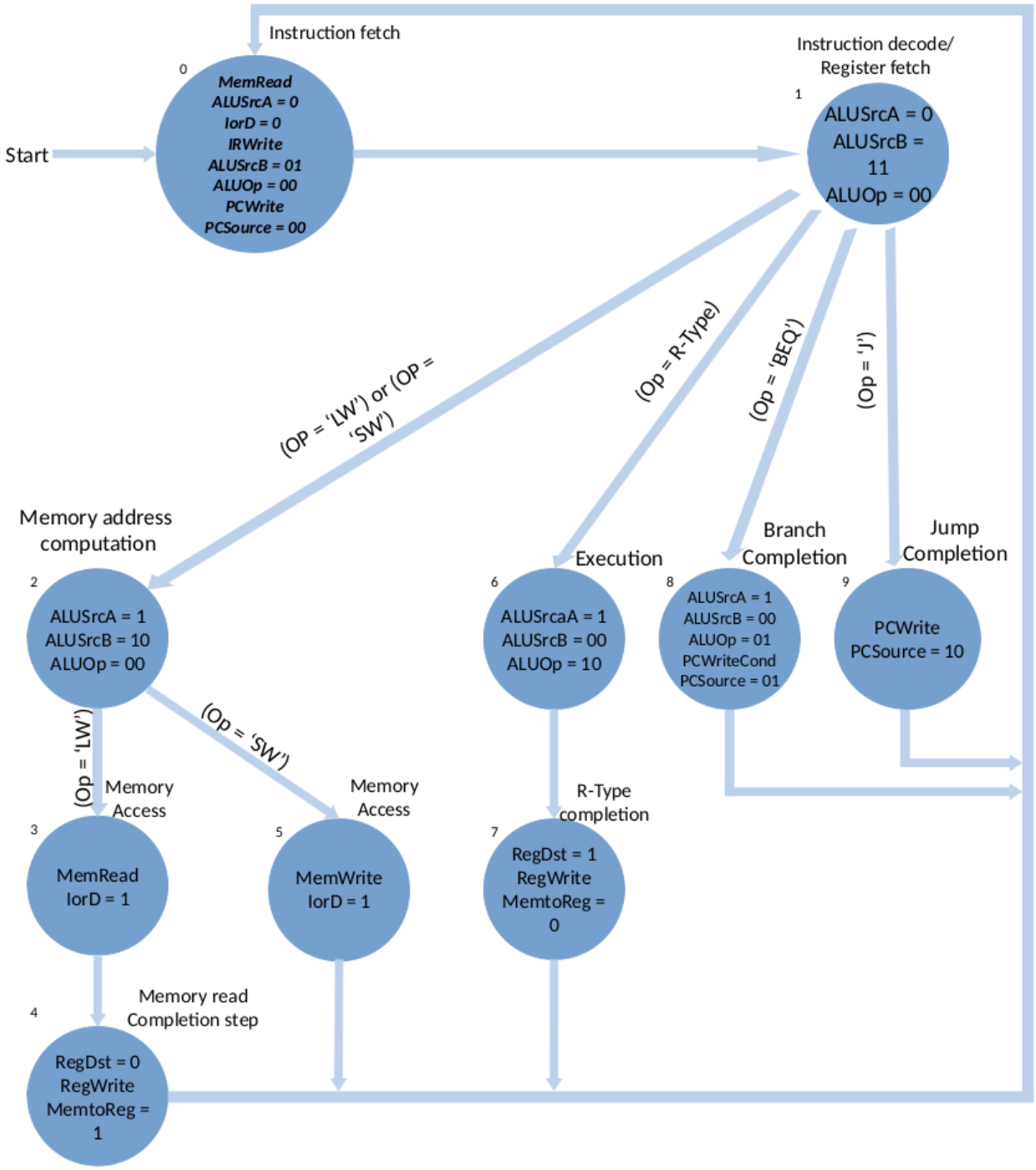

Figura 2 - Máquina de estados do controle do caminho de dados.

Por fim, também utilizou-se para estudo uma tabela que mapeia as microinstruções que podemos utilizar para escrever o microprograma. Essa tabela pode ser visualizada na figura 3 e é chamada de tabelão.

Nome do campo	Valores para o campo	Função do campo com valor específico
Label	Qualquer string	Usado para especificar rótulos para controlar a seqüenciação de microcódigo. Os rótulos que terminam em 1 ou 2 são usados para despachar com uma tabela de desvios indexada com base no opcode. Outros rótulos são usados como destinos diretos na seqüenciação de microinstruções. Os rótulos não geram sinais de controle diretamente mas são usados para definir o conteúdo das tabelas de despacho e gerar controle para o campo Sequencing.
ALU Control	Add	Faz com que a ALU realize uma soma.
	Subt	Faz com que a ALU realize uma subtração; isso implementa a comparação para desvios.
	Func Code	Usa o campo funct da instrução para determinar o controle da ALU.
SRC1	PC	Usa o PC como a primeira entrada da ALU.
	A	O registrador A é a primeira entrada da ALU.
SRC2	B	O registrador B é a segunda entrada da ALU.
	4	Usa 4 para a segunda entrada da ALU.
	Extend	Usa a saída da unidade de extensão de sinal como a segunda entrada da ALU.
	Extshft	Usa a saída da unidade de deslocamento como a segunda entrada da ALU.
Register Control	Read	Lê dois registradores usando os campos rs e rt do IR como os números de registrador, colocando os dados nos registradores A e B.
	Write ALU	Escreve no banco de registradores usando o campo rd do IR como o número de registrador e o conteúdo de SaídaALU como os dados.
	Write MDR	Escreve no banco de registradores usando o campo rt do IR como o número de registrador e o conteúdo de MDR como os dados.
Memory	Read PC	Lê a memória usando o PC como o endereço; escreve o resultado em IR (e no MDR).
	Read ALU	Lê a memória usando SaídaALU como o endereço; escreve o resultado em MDR.
	Write ALU	Escreve na memória usando SaídaALU como o endereço e o conteúdo de B como os dados.
PCWrite Control	ALU	Escreve a saída da ALU no PC.
	ALUOut-cond	Se a saída Zero da ALU estiver ativa, escreve em PC o conteúdo do registrador SaídaALU.
	Jump Address	Escreve no PC o endereço de desvio da instrução.
Sequencing	Seq	Escolhe a próxima microinstrução seqüencialmente.
	Fetch	Vai para a primeira microinstrução para começar uma nova instrução.
	Dispatch i	Despacha usando a ROM especificada por i (1 ou 2).

Figura 3 - Tabelão.

Com o estudo desses diagramas e tabelas foi possível entender como as instruções são implementadas, em especial a que queremos substituir. A estratégia utilizada para realizar a substituição da instrução BEQ por BNE foi adicionar uma porta NOT na saída da flag zero da ALU.

O novo caminho de dados está representado na figura 4 abaixo.

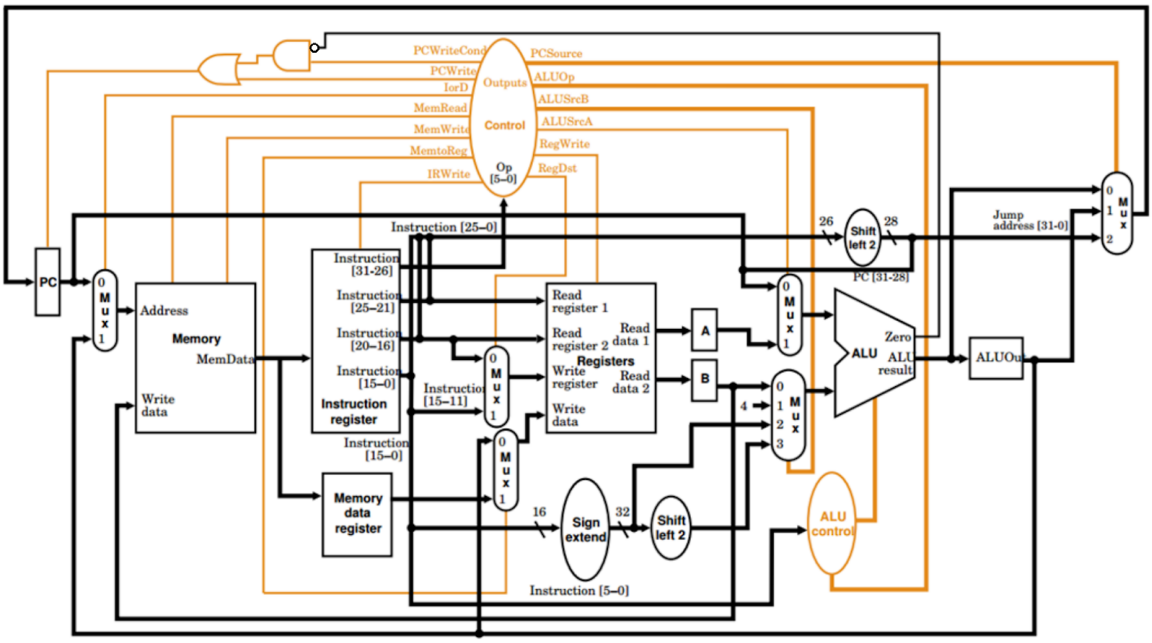

Figura 4 - Caminho de dados alterado.

Seguindo essa estratégia, são necessárias poucas alterações no diagrama que representa a máquina de estados do controle do caminho de dados e no tabelão. A figura 5 mostra o novo diagrama da máquina de estados.

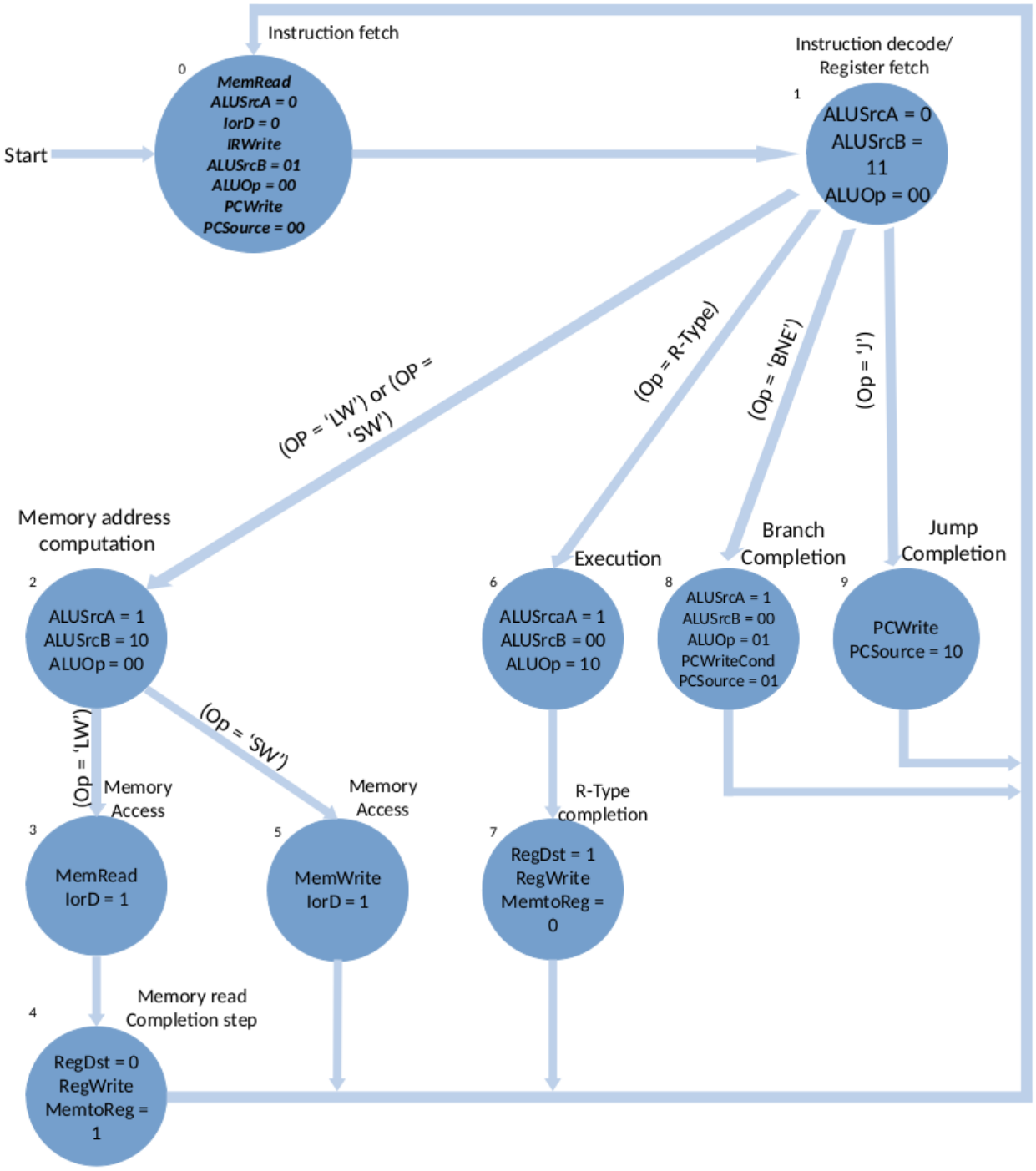

Figura 5 - Máquina de estados do controle do caminho de dados alterada.

A única alteração feita no diagrama da figura 5 foi na descrição da seta que representa a mudança de estado para a instrução BEQ. Como essa instrução foi substituída, onde estava escrito: “(Op = ‘BEQ’)” agora se encontra “(Op = ‘BNE’)”.

A tabela 1 mostra a alteração feita no tabelão.

PCWrite Control	ALU	Escreve a saída da ALU no PC.
	ALUOut-cond	Se a saída da ALU não estiver ativa, escreve em PC o conteúdo do registrador SaídaALU.
	Jump Address	Escreve em PC o endereço do desvio da instrução.

Tabela 1 - Alteração feita no tabelão.

A única alteração necessária no tabelão foi na descrição da microinstrução ALUOut-cond representada em vermelho na tabela 1. Com a alteração do caminho de dados mencionada anteriormente para realizar a substituição das instruções, percebe-se que se faz necessária essa atualização no tabelão para continuar com a coerência da descrição. No mais, todas as outras linhas permanecem inalteradas.

Com a estratégia mencionada implementada e as alterações mencionadas, o microprograma completo é descrito na tabela 2 abaixo.

Label	ALU Control	SRC1	SRC2	Register Control	Memory	PCWrite Control	Sequencing
Fetch	Add	PC	4		Read PC	ALU	Seq
	Add	PC	Extshft	Read			Dispatch 1
Mem1	Add	A	Extend				Dispatch 2
LW2					Read ALU		Seq
				Write MDR			Fetch
SW2					Write ALU		Fetch
Rformat 1	Func Code	A	B				Seq
				Write ALU			Fetch

BNE1	Subt	A	B			ALUOut-cond	Fetch
JUMP1						Jump Address	Fetch

Tabela 2 - Microprograma.

CONCLUSÃO

Foi possível realizar a substituição da instrução BEQ pela instrução BNE na microarquitetura MIPS. Poucas alterações foram necessárias no caminho de dados, na máquina de estados e no tabelão. Nos dois últimos apenas modificações textuais para permanecer a coerência com a nova instrução. E a implementação no caminho de dados se deu pela adição de uma porta lógica NOT ao final da flag zero da ALU.

REFERÊNCIAS

PATTERSON, D.; HENESSY, J. Arquitetura de Computadores. Uma Abordagem Quantitativa. 3. ed.