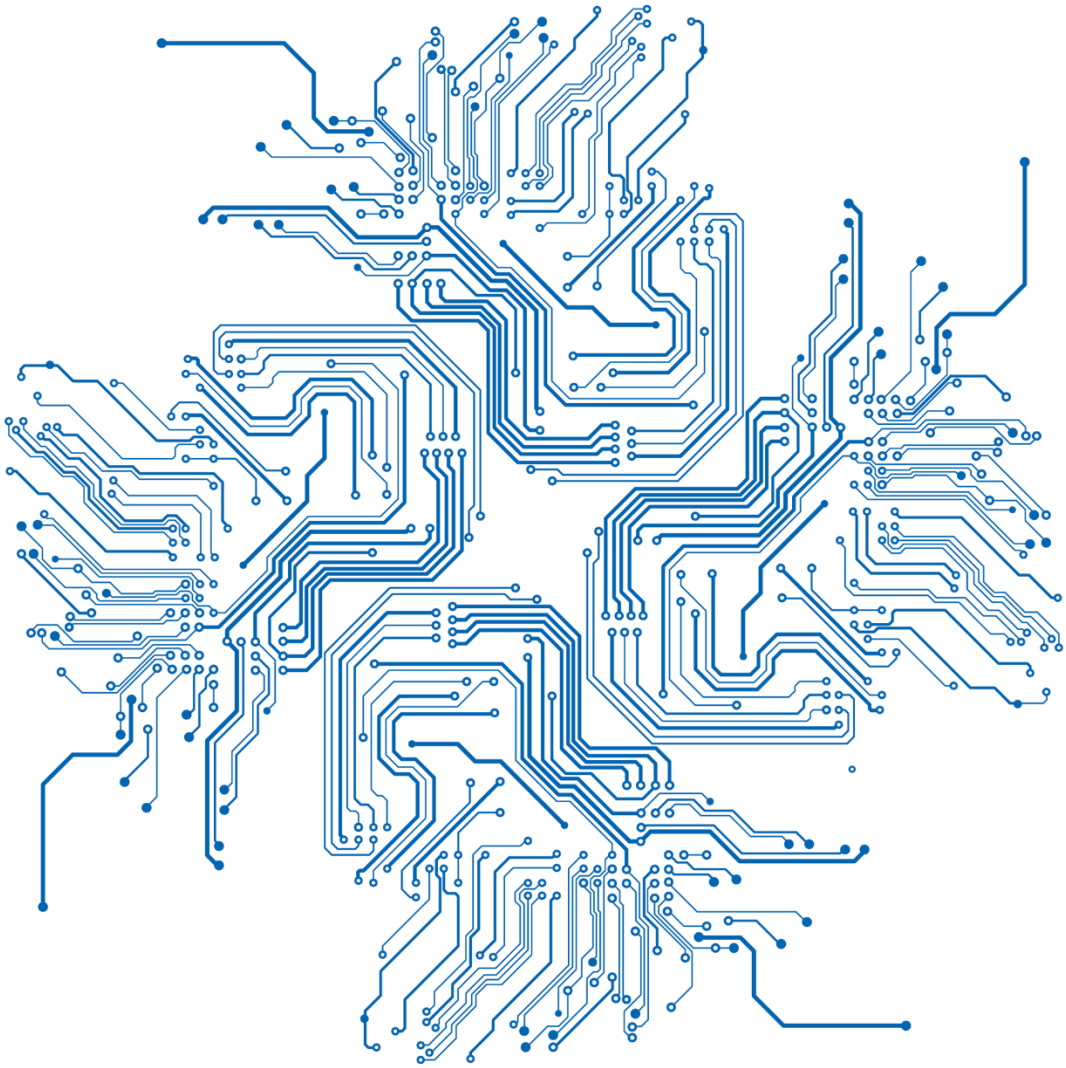
nRisc Processor

Aluno: Tarcísio Batista Prates



REESTRTURAÇÃO DO PROJETO

nRisc Instruction

000	000	00
OPERATION	**RESGISTER_0**	**REGISTER_1**

A instrução adotada para a execução de tarefas no processador ***nRisc***, possui 8 bits de largura, sendo os três primeiros dedicados à operação a ser executada, os três seguintes são dedicados ao primeiro registrador da operação e os dois últimos ao segundo registrador. Sendo assim, o usuário dispõe de oito tipos operações diferentes e oito registradores para serem usados nas operações. De modo geral a arquitetura das operações segue o modelo descrito acima, no entanto, a operação de imediato, utiliza duas instruções para executar a operação, mais detalhes na descrição completa abaixo. Além disso, como os registradores possuem três bits, o segundo registrador recebe um bit zero à esquerda, proporcionado por um extensor de sinal.

Code operations table

Nº	*CODE*			*OP*	*DESCRIPTION*
0	**0**	**0**	**0**	ADD	Soma dois registradores
1	**0**	**0**	**1**	SUB	Subtrai dois registradores
2	**0**	**1**	**0**	MUL	Multiplica dois registradores
3	**0**	**1**	**1**	LI	Carrega um imediato em um registrador
4	**1**	**0**	**0**	SW	Armazena o valor de um registrador na memória
5	**1**	**0**	**1**	LW	Carrega um valor da memória em um registrador
6	**1**	**1**	**0**	BEQ	Compara igualdade entre dois registradores
7	**1**	**1**	**1**	BNZ	Faz um jump com base no resultado de BEQ

Register code table

Nº	*CODE*			*REGISTER*
0	**0**	**0**	**0**	ZERO
1	**0**	**0**	**1**	R_BEQ
2	**0**	**1**	**0**	S0
3	**0**	**1**	**1**	S1
4	**1**	**0**	**0**	S2
5	**1**	**0**	**1**	S3
6	**1**	**1**	**0**	S4
7	**1**	**1**	**1**	S5

Instructions examples

Operação de soma [ADD]

```
add s3, s1   //Estrutura
s3 = s3 + s1 //Interpretação no processador
000 101 11   //Representação em binário
```

Operação de subtração [SUB]

```
sub s3, s1   //Estrutura
s3 = s3 - s1 //Interpretação no processador
001 101 11   //Representação em binário
```

Operação de multiplicação [MUL]

```
mul s3, s1   //Estrutura
s3 = s3 * s1 //Interpretação no processador
010 101 11   //Representação em binário
```

Operação carregamento de imediato [LI]

```
li s3          //Primeiro o reg. de destino
10             //Valor a ser gravado
MEMORY[s3] = 10      //Interpretação no
processador
011 101 00       //Representação em binário
00001010      //valor
//A instrução é composta de duas partes, assim
//é possível gravar valores mais altos no reg.
```

Operação de store word [SW]

```
sw s3, s1          //Estrutura
MEMORY[s1] = s3 //Interpretação no processador
100 101 11       //Representação em binário
//Grava o valor de um registrador na memória
```

Operação de load word [LW]

```
lw s3, s1          //Estrutura
S3 = MEMORY[s1] //Interpretação no processador
101 101 11       //Representação em binário
//Carrega um valor da memória no registrador
//informado
```

Operação de comparação de registradores [BEQ]

```
beq s3, s1           //Estrutura
R_BEQ = (s3 == s1) //Interpretação
110 101 11           //Representação em binário
```

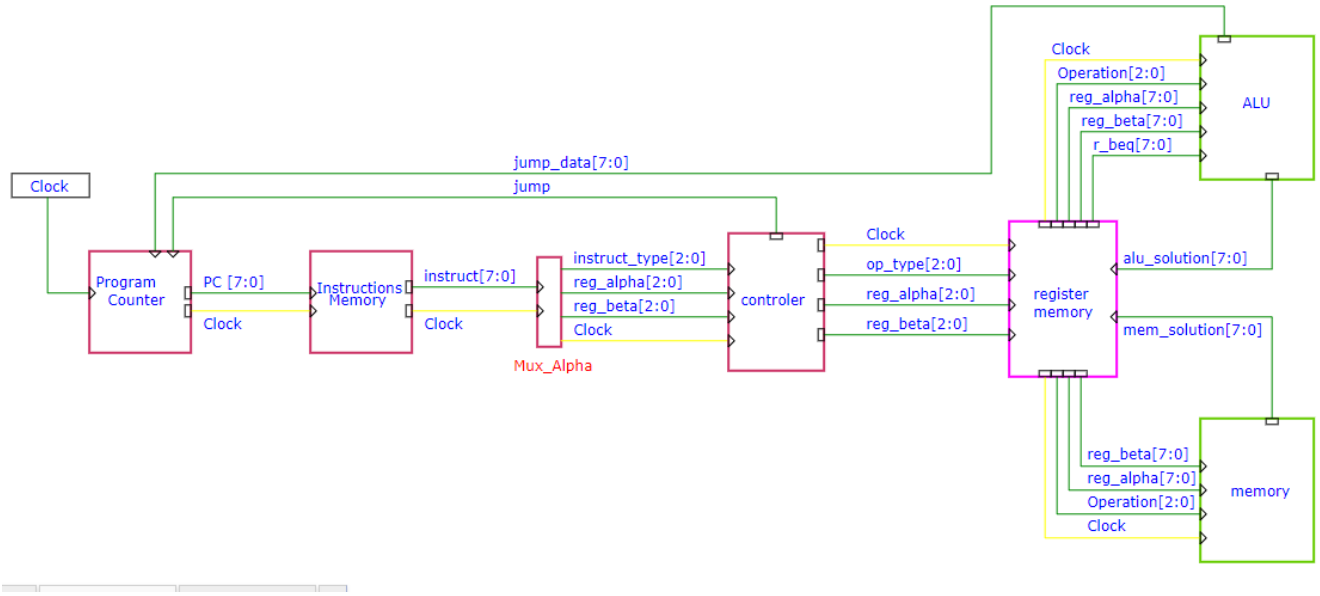
```
//Compara os dois registradores informados,
//sendo o valor gravado no registrador R_BEQ,
//onde 1 para verdadeiro e 0 para falso.
```

Operação de comparação de registradores [BNZ]

```
bnz s3, s1                          //Estrutura
pc = (R_BEQ == 0)? S3 : s1 //Interpretação
111 101 11          //Representação em binário
//Compara o resultado da instrução BEQ e com
//base nele faz um jump
```

Datapath nRisc

Arquivo com detalhes na pasta raiz



Control signals:

O novo projeto de processador nRisc foi pensado para seguir uma estrutura linear, de modo que quase todos os sinais de controle fossem desnecessários, já que tornou-se possível reaproveitar os bits da instrução que se referem a operação a ser realizada como sinal de controle, segue a nova tabela abaixo:

N°	SIGNAL	REG NAME	FUNCTION
0	1	Li_instruction	Habilita operação multiciclo para receber um imediato e posteriormente gravá-lo no registrador
1	1	await_solutions	Sempre que uma operação for executada e que o resultado precisa ser gravado no banco de registradores, o sinal é habilitado, colocado o módulo dos registradores em função de gravação, pois assim que o resultado, seja aritmético da ULA ou de leitura de memória retornar o dado, é imediatamente gravado na posição anteriormente definida pela instrução.
2	0 ou 1	clock	Para aproveitar ao máximo o periodo de clock, os módulos de **PC**, i**nstrução de memória** e **controle** funcionam detectando a borda de descida do clock, a ULA e a memória são sensíveis à borda de subida, já o banco de registradores é sensível aos dois. Assim toda instrução é realizada apenas em um único período de clock.
3	1	jump	Habilita a inserção de um valor arbitrário na contagem de PC
4	[2:0]	operation	Recebe a instrução da word