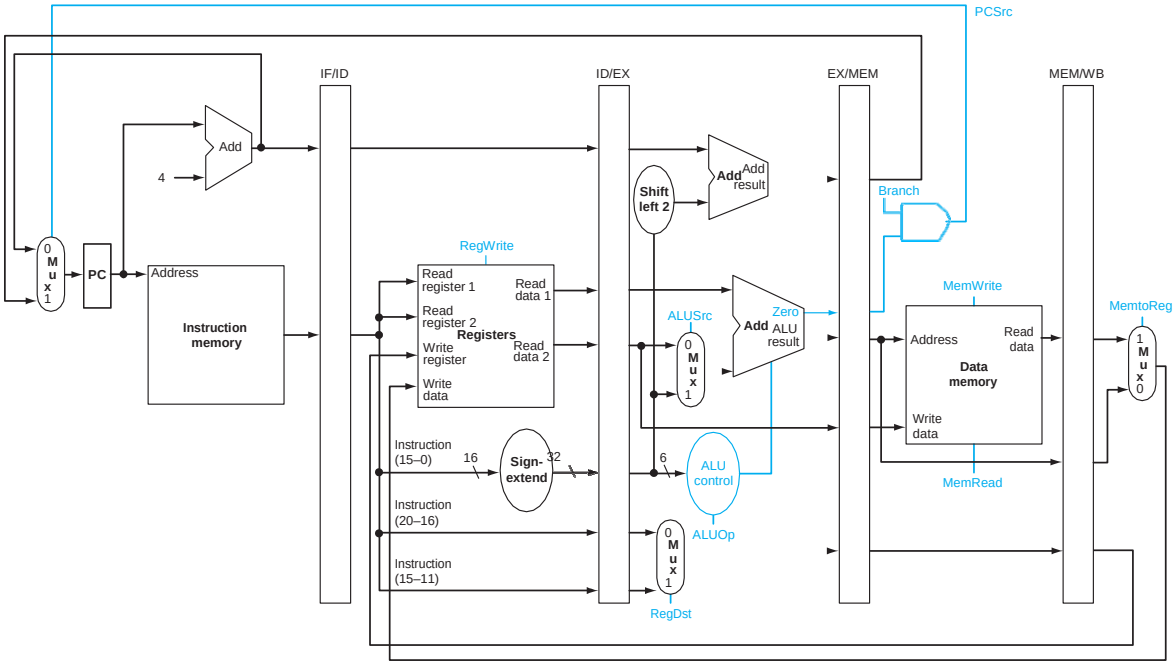
UNIVERSIDADE FEDERAL FLUMINENSE
INSTITUTO DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Arquiteturas de Computadores – Turma :A1 – Lista 2
Profa.: Simone Martins

1. Indique como modificar o código abaixo de modo que não se tenha atraso na execução do pipeline por dependência de dados. A modificação deve incluir mudança na ordem das instruções e modificação da instrução lw.

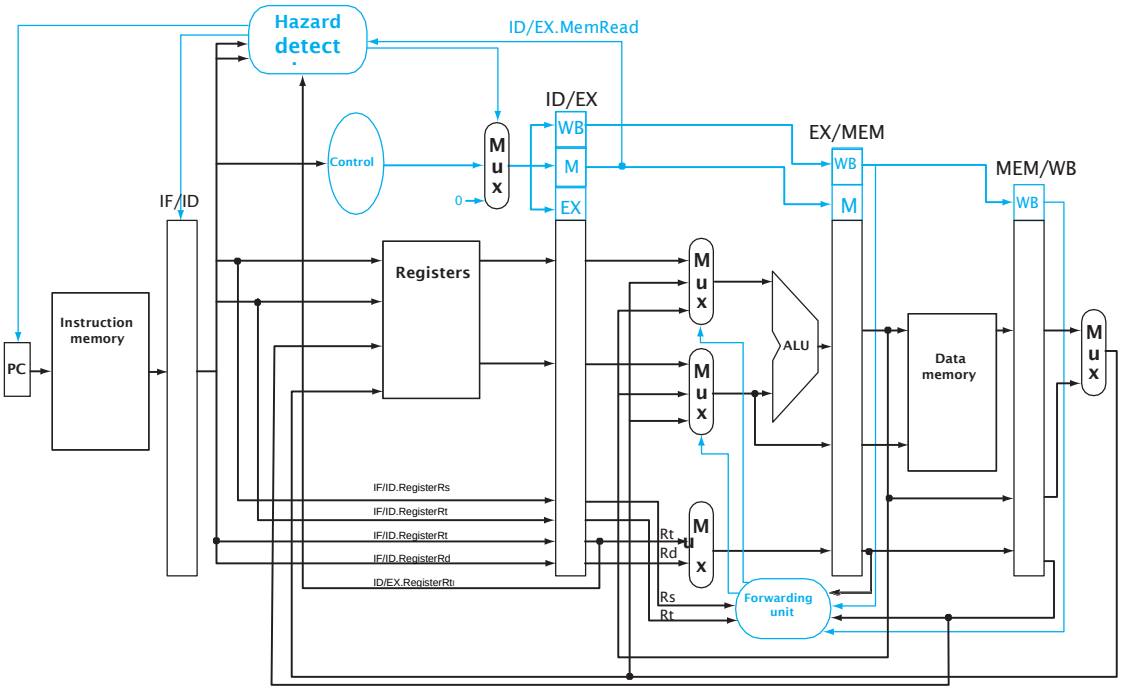
```
Loop: lw $2, 100($3)
      addi $3, 4
      beq $3, $4, Loop
```

2. Para cada registrador de pipeline na figura abaixo, indique o nome do campo que é armazenado no registrador. Determine o tamanho de cada campo em bits. Por exemplo, o registrador de pipeline IF/ID contém dois campos, um dos quais é o campo da instrução que possui 32 bits.



3. Considere o seguinte código executado no processador pipeline da figura abaixo.

```
add $1,  $2, $3
add $4,  $5, $6
add $7,  $8, $9
add $10, $11,$12
add $13, $14,$15
```



Responda:

a) No final do quinto ciclo, quais registradores estão sendo lidos e qual está sendo escrito?
b) Explique o que a unidade de encaminhamento (forwarding unit) está fazendo durante o quinto ciclo de execução
c) Explique o que a unidade de conflito (hazard unit) está fazendo durante o quinto ciclo de execução

4. Considere o seguinte código que será executado no processador da questão anterior:

```
add $5,  $6, $7
lw  $6,  100($7)
sub $7,  $6, $8
```

Quantos ciclos de relógio serão necessários para executar este código? Desenhe um diagrama deste programa mostrando as dependências que deverão ser resolvidas, considerando que não existe unidade de encaminhamento e de conflito. Desenhe um outro diagrama para mostrar como o código realmente será executado (incorporando as unidades de encaminhamento e conflito) para resolver os problemas identificados e mostrados no diagrama anterior.

5. Liste todas as entradas e saídas da unidade de encaminhamento da figura da questão 3, indicando os nomes e quantidade de bits de cada entrada e saída.

6. Considere o seguinte código:

```
Loop: lw   $t0, 0($s1)
      add  $t0, $t0,$s2
      sw   $t0, 0($s1)
      addi $s1, $s1, –8
      bne  $s1, $zero, Loop
```

Calcule o número de ciclos de execução deste código considerando que serão executadas X iterações. Considere que a dependência de dados entre as instruções lw e outras instruções causa paradas e que o endereço a ser utilizado nas instruções de desvio é conhecido ao final do ciclo de execução.

Suponha que o número X de interações é múltiplo de 2, e considere que o código foi desenrolado como abaixo:

```
Loop: lw $t0, 0($s1)
      lw $t1, –4($s1)
      addu $t0, $t0, $s2
      addu $t1, $t1, $s2
      sw $t0, 0($s1)
      sw $t1, –4($s1)
      addi $s1, $s1, –8
      bne $s1, $zero, Loop
```

Reordene as instruções para que ele seja executado de forma mais rápida e calcule o número ciclos que serão necessários para execução deste programa. Compare o número de ciclos necessários para execução dos dois códigos.

7. Considere uma arquitetura de processador em que cada instrução está disponível para ser executada em um ciclo de relógio e cada instrução necessita dos seguintes números de ciclo de relógio para finalizar sua instrução (latência da instrução) :

```
LD - 4
DIVD – 12
MULTD - 5
ADDD – 1
ADDI - 0
SD - 1
SUB  - 0
BNZ  - 1
```

Considere também que uma instrução só pode começar a ser executada quando a outra acaba. Calcule o número de ciclos de relógio por iteração que serão utilizados para executar o código abaixo:

```
Loop:  LD     F2,0(Rx)
       DIVD   F8,F2,F0
       MULTD  F2,F6,F2
       LD     F4,0(Ry)
       ADDD   F4,F0,F4
       ADDD   F10,F8,F2
       ADDI   Rx,Rx,#8
       ADDI   Ry,Ry,#8
       SD     F4,0(Ry)
       SUB    R20,R4,Rx
       BNZ    R20,Loop
```

8. Considere uma nova arquitetura que detecta dependência de dados entre as instruções e não inicia a execução de uma nova instrução somente se ela depende do término da execução de uma instrução anterior. Mostre o código da questão anterior com as instruções de parada inseridas de modo que cada dependência de dados seja resolvida. Caso não haja dependência de dados, não é necessário inserir instruções de parada. Calcule o número de ciclos necessário para cada iteração nesta nova arquitetura.

9. Considere um processador projetado com múltiplas emissões, que possui dois pipelines de execução, cada um capaz de iniciar a execução de uma instrução por ciclo. Os resultados podem ser encaminhados imediatamente de um pipeline para outro ou para si mesmo. Um pipeline de execução somente é bloqueado se houver uma verdadeira dependência de dados. Calcule o número de ciclos que cada iteração do programa da questão 7 irá necessitar neste processador.

10. Reordene as instruções do código da questão 6 para melhorar o desempenho da execução do programa. Calcule quantos ciclos por iteração serão necessários para executar o código reordenado no processador da questão anterior, levando em consideração as dependências de dados do novo código e as latências das unidades funcionais.

11. Considere o código reordenado da questão anterior.
    a) Indique a fração da quantidade total de ciclos não utilizados para iniciar uma instrução considerando os dois pipelines de execução
    b) Reescreva o código, desenrolando cada iteração em 2 iterações
    c) Indique a aceleração obtida dividindo o tempo do código reordenado pelo código reordenado desenrolado

12. Uma maneira de fazer renomeação de registradores é designar o registrador **destino** de cada instrução a um registrador temporário diferente e depois verificar as dependências de dados verdadeiras e trocar os nomes dos registradores **fonte** das instruções para manter as dependências. Utilize esta técnica para o código abaixo, considerando que podem ser utilizados os registradores T9 até T63.

```
Loop:   LD    F4, 0(Rx)
IO:     MULTD F2,F0,F2
I1:     DIVD  F8,F4,F2
I2:     LD    F4,0(Ry)
I3:     ADDD  F6,F0,F4
I4:     SUBD  F8,F8,F6
I5:     SD    F8,0(Ry)
```

    Dica de como ficariam as duas primeiras instruções:

```
Loop:   LD    T9, 0(Rx)
IO:     MULTD T10,F0,F2
```

13. Considere o código abaixo que implementa a operação DAXPY=aX+Y, onde a é um escalar e X e Y são vetores de 100 posições:

```
      DADDIU    R4,R1,#800
foo:  L.D       F2,0(R1)
      MUL.D     F4,F2,F0
      L.D       F6,0(R2)
      ADD.D     F6,F4,F6
      S.D       F6,0(R2)
      DADDIU    R1,R1,#8
      DADDIU    R2,R2,#8
      DSLTU     R3,R1,R4
      BNEZ      R3,foo
```

    Considere que as unidades funcionais possuem as latências em ciclos de relógio como mostrado na tabela abaixo. A latência indica quantos ciclos de relógio são necessários para uma instrução gerar o valor correto em um registrador para poder ser usado por outra instrução. Por exemplo, se uma instrução que realiza uma instrução de PF na ALU depende de uma instrução para gerar o conteúdo de um registrador que é uma multiplicação PF, a instrução só poderá ser carregada 5 ciclos de relógio após a execução da instrução de multiplicação. Assuma também que as instruções de desvio introduzem 1 ciclo de latência.

Instrução que produz o resultado	**Instrução que usa o resultado**	**Latência em ciclos de relógio**
Multiplicação PF	Operação de PF na ALU	6
Soma PF	Operação de PF na ALU	4
Multiplicação PF	Armazenamento PF	5

Soma PF	Armazenamento PF	4
Todas as operações com inteiros e carregamento da memória	Qualquer	2

a) Considere que o código irá ser executado em um processador com um único pipeline. Calcule o número de ciclos de relógio necessários para calcular um elemento de Y considerando dois casos: (i) o código é executado na forma original; (ii) o código é compilado e gerado de forma reordenada para diminuir a dependência dos dados. Calcule quanto mais rápido o relógio deve ser para atingir o desempenho obtido pelo compilador
b) Para o mesmo processador acima, encontre o número de vezes necessário para desenrolar o laço para que não ocorram paradas. Mostre como ficaria o código desenrolado e quantidade de ciclos para gerar cada elemento.
c) Considere um processador VLIW que pode emitir, em cada ciclo de relógio, uma instrução com 5 operações: 2 referências à memória, 2 operações PF e uma operação com inteiros ou desvio. Desenrole o código em 6 vezes, e mostre o diagrama de execução neste processador VLIW. O diagrama deve conter em cada linha a indicação do ciclo de relógio, cada instrução de referência de memória, cada instrução PF e cada instrução inteira ou de desvio executadas em cada ciclo. Desenhe o mesmo diagrama para o código desenrolado 10 vezes.