Esta ficha foi elaborada para acompanhar o estudo da disciplina de Arquitetura de Computadores Avançada (a4s1) ou para complementar a preparação para os momentos de avaliação finais da mesma. Num segundo ficheiro poderás encontrar um conjunto de propostas de solução aos exercícios que estão nesta ficha. É conveniente relembrar que algum conteúdo destes documentos pode conter erros, aos quais se pede que sejam notificados pelas vias indicadas na página web, e que serão prontamente corrigidos, com indicações de novas versões.

1. Considera a seguinte sequência de código abaixo (Assembly para MIPS64):

```
loop:   l.d     f2, 0(rx)
i0:     div.d   f8, f2, f0
i1:     mult.d  f2, f6, f2
i2:     l.d     f4, 0(ry)
i3:     add.d   f4, f0, f4
i4:     add.d   f10, f8, f2
i5:     addi    rx, rx, 8
i6:     addi    ry, ry, 8
i7:     s.d     f4, 0(ry)
i8:     sub     r20, r4, rx
i9:     bnz     r20, loop
```

Consideremos também, e para o efeito, que as instruções de load têm uma latência de 4 ciclos, as instruções de store, branches e somas em vírgula flutuante têm uma latência de 1 ciclo, a instrução de multiplicação em vírgula flutuante têm uma latência de 5 ciclos, as instruções de divisão em vírgula flutuante têm uma latência de 12 ciclos e as operações aritméticas e lógicas inteiras têm uma latência de 0 ciclos. Neste cenário:

a) Qual é o desempenho base, em número de ciclos (por iteração do loop), da sequência de código acima se nenhuma instrução puder ser iniciada sem que uma anterior seja completada? Ignorando os tempos de fetch e de descodificação (decode), consideremos ainda que a execução das instruções não para à conta de uma instrução seguinte, mas que as instruções são lançadas uma por ciclo e que o branch é taken, isto é, existe um delay slot de um ciclo de branch.

b) Quantos ciclos de execução o corpo do "loop" se a sequência de código necessitaria se o pipeline detetasse dependências de dados verdadeiras e só parasse a execução (stall) nesses instantes, ao invés de parar tudo cegamente só porque uma determinada unidade funcional se encontra ocupada? Mostra o código com a indicação de stalls onde necessário de forma a acomodar as tais latências.

2. O seguinte código é denominado de DAXPY e a sua operação fundamental é a redução de uma matriz por eliminação com o método de Gauss, neste caso, para um vetor de tamanho 100. Considere-se que r1 contém o endereço-base do array X e que r2 contém o endereço-base do array Y:

```
        daddiu  r4, r1, 800     ; endereço-base de X
for:    l.d     f2, 0(r1)       ; (f2) = X[i]
        mul.d   f4, f2, f0      ; (f4) = a*X[i]
        l.d     f6, 0(r2)       ; (f6) = Y[i]
        add.d   f6, f4, f6      ; (f6) = a*X[i] + Y[i]
        s.d     f6, 0(r2)       ; Y[i] = a*X[i] + Y[i]
        daddiu  r1, r1, 8       ; incremento de índice de X
        daddiu  r2, r2, 8       ; incremento de índice de Y
        dsltu   r3, r1, r4      ; teste: prosseguir no ciclo?
        bnez    r3, for         ; ciclo em for se necessário
```

Assumam-se, também, que temos as seguintes latências, por par (unidade funcional, instrução): (multiplicador FP, operação ALU FP) = 6 ciclos de relógio; (somador FP, operação ALU FP) = 4 ciclos de relógio; (multiplicador FP, operação de store) = 5 ciclos de relógio; (somador FP, operação de store) = 4 ciclos de relógio; (unidade inteira, operações inteiras e loads) = 2 ciclos de relógio.

a) Considera um pipeline comum de cinco andares e mostra a que é que o "loop" se assemelharia tanto antes como depois do agendamento para ambas operações de vírgula flutuante e branch delays, incluindo quaisquer stalls ou ciclos de inatividade (idle cycles). Qual é o tempo de execução (em ciclos) por elemento do vetor de resultado Y, tanto na sua versão não-agendada, como na agendada? Quão mais rápido deverá ser o relógio para que o processador, por si, consiga atingir o desempenho da versão agendada pelo compilador? (Ignora quaisquer efeitos de aumento do ciclo de relógio no desempenho da memória do sistema)

3. Descreve uma forma de exploração de cada um dos dois tipos de paralelismo juntamente com um desafio em fazê-la.

a) Paralelismo de Instruções;

b) Paralelismo de Dados;

4. Consideremos uma máquina com um pipeline clássico de cinco andares MIPS que usa predição de branch sem delay slots e que possui uma taxa de mispredict penalty de 3 ciclos de execução. Considerando também que em uma de cada cinco instruções encontra-se uma instrução de branch para um dado programa, para os quais 80% são bem previstos pelo nosso preditor:

a) Quantos ciclos demora a executar N instruções?

b) Assumindo que temos um processador Intel Pentium 4 (que possui um pipeline de 20 andares), agora temos uma inacreditável taxa de mispredict penalty de 19 ciclos. Qual deverá ser o rácio de predição de branch de forma a que tenhamos o mesmo desempenho que no pipeline de cinco andares clássico do MIPS, tal como vimos em a)?

5. Consideremos um pipeline MIPS clássico de cinco andares e o código abaixo a ser executado neste:

```
linha
1.      lw      r5, 20(r0)
2.      add     r4, r0, r0
3.      sw      r5, 100(r4)
4.      add     r5, r5, r5
5.      sw      r5, 104(r4)
6.      add     r5, r5, r5
7.      sw      r5, 106(r4)
8.      add     r5, r5, r5
9.      sw      r5, 108(r4)
```

a) Identifica potenciais hazards no código acima. Para cada tipo de hazard (estrutural, de dados e de controlo), apresenta uma ocorrência. Detalha a tua resposta com a linha(s) envolvidas no hazard, e sugere uma correção. Mesmo que todas as linhas tenham um hazard, identifica apenas um de cada tipo, assumindo que tais hazards estão presentes no código.

b) Calcula o número de ciclos por instrução (CPI) do código acima, assumindo que não há unidade de forwarding (considera também que a última instrução é sw r5, 112(r4)).

c) Corrige as dependências de dados com forwarding e identifica, caso existam dependências não passíveis de serem resolvidas com técnicas de forwarding ou interlocking. Considera, novamente, que a última instrução é sw r5, 112(r4). Calcula novamente o CPI.

6. Poderá acontecer algum hazard do pipeline envolvendo o registo R0 (zero)?