1. Consider execu¸c~ to the following section of c' Odigo a processor com- ISA pat'ıvel with MIPS.

```
120h:  loop: LD     F0,0 (R1)      ; F0 ← M [R1 + 0]
124H:        DADD   R1, R1, # - 8  ; R1 ← R 1 - 8
128h:        LD     F2,0 (R2)      ; F2 ← M [R 2 + 0]
12CH:        DADD   R2 R2 # - 8    ; R2 ← R2 - 8
130h:        LD     F4,0 (R3)      ; F4 ← M [R3 + 0]
134h:        DADD   R3, R3, # - 8  ; R3 ← R3 - 8
138H:        MUL.D  F2, F0, F2     ; F2 ← F0 × F2
13CH:        MUL.D  F4, F4, F4     ; F4 ← F4 × F4
140h:        DSUB.D F0, F4, F2     ; F0 ← F4 - F2
144h:        SD     0 (R4), F0     ; M [R 4 + 0] ← F0
148h:        DADD   R4 R4 # - 8    ; R4 ← R4 - 8
14Ch:        BNE    R1, R0, loop   ; PRAÇA ← loop if R1 = R0
```

Solve the following al'ıneas doing all simpli fi ca¸c~ oes considers appro- ones (where the fi zer, write them down in the statement). (The)

2.0 val.

Consider the execution of a processor c'odigo *in-order* with 5 est' Agios (IF, ID, **EX, MEM, WB) without any mechanism *forwarding* data, predi¸c~** to **jumps or *delayed branch.* Identifies all dependencies of data and control that generate conflicts. Tell them** directly on oc' Odigo.

3.0 val.

(B) Rewrite oc' Odigo in order to resolve the dependencies shown in al'ınea previous.

4.0 val.

(W) Consider a super-scalar processor with:

- din schedule amico using Tomasulo algorithm;
- execu¸c~ the speculative (jump predictor with a 100% success rate);
- *issue* Simultaneous aneo two instru¸c~ oes;
- n'humerus suf fi ciently large esta¸c~ booking oes and entries in the ROB;
- 1 CBD and *commit* Simultaneous aneo 2 instru¸c~ oes;
- functional units with the following latencies:

1 × INT ALU / 1 cycle 1 BRANCH × LOAD / STORE	1 cycle c'alculo address + 1 cycle to access the `mem'oria
1 × FP ADD	3 cycles
1 × FP MULT	5 cycles

List the steps execu¸c~ to the section of c' Odigo for 2 iterates c~oes.
(fa ca all simpli fi ca¸c~oes as it considers advisable, indicating them with the answer.)

Instru¸c~ to		Cycle rel' ogen IF	issue	EX	CDB Commit		Comments
loop: LD	F0,0 (R1)						
DADD	R1, R1, # - 8						
LD	F2,0 (R2)						
DADD	R2 R2 # - 8						
LD	F4,0 (R3)						
DADD	R3, R3, # - 8						
MUL.D	F2, F0, F2						
MUL.D	F4, F4, F4						
DSUB.D	F0, F4, F2						
SD	0 (R4), F0						
DADD	R4 R4 # - BNE 8						
	R1, R0, loop						
loop: LD	F0,0 (R1)						
DADD	R1, R1, # - 8						
LD	F2,0 (R2)						
DADD	R2 R2 # - 8						
LD	F4,0 (R3)						
DADD	R3, R3, # - 8						
MUL.D	F2, F0, F2						
MUL.D	F4, F4, F4						
DSUB.D	F0, F4, F2						
SD	0 (R4), F0						
DADD	R4 R4 # - BNE 8						
	R1, R0, loop						

2. Consider a RISC processor data bus and 32-bit addresses and execu¸c~
the speculative and out of order. (The)

2.0 val. Outline state diagram corresponding to a dynamic predictor with 2-bit jumps.

2.0 val. (B) Outline structure of a *branch target bu ff er* BTB two bits associated with the est' goodwill of *Instruction Fetch.* Whereas the BTB is a cache predi¸c~ to jump with direct mapping, indicate dimens~ to each of BTB fields.

2.0 val. (W) Consider that the din schedule amico instru¸c~ oes uses the algorithm *Tomasulo.* Explain what is the mechanism that ensures that instru¸c~ oes *issued* ap' the one incorrectly predicted jump, n~ to alter the value of records or mem' oria. Justi fi succinctly.

3. Consider the following section of c' Odigo in *W:*

```
for (i = 50; i> 0; i--) {
        A [i + 1] = B [i] * D [i] + C [i-1];          / * * S1 /
        B [i-1] = D [i] - C [i];                      / * S2 * /
        C [i]     = 2 * cos (D [i] + alpha * F [i]);  / * * S3 /
        D [i]     = F [i];                            / * S4 * /
}
```

3.0 val.

(The) Assuming that the vectors s~ the distinct en~ the overlapping presents a graph with all existing inter-dependencies.

2.0 val.

(B) Say, justi fi cating if oc' Odigo shown'and paraleliz'avel n'ıvel the cycle, ie 'and if poss'ıvel run each iteration c~ao this cycle independently and para lelo. If the rmativo fi, rewrite it so that your execu¸c~ may be the performed in parallel.

4. Consider a processor data bus and 32-bit address cache with three n'ıveis:

[L1] separate Data Cache cache of instru¸c~ oes (L1-L1-I and D), each with ability to 32KB (divided into 4 ways);

[L2] Cache uni fi ed with a capacity of 64KB in each of the eight channels; [L3] Cache uni fi ed with a capacity of 256KB in each of the 16 channels.

Consider that in all n'ıveis lines of cache are of 32B. (The)

2.0 val. **Recital 32 bit addresses, decompose the address word label (*tag)* index (*index)* and displacement (*ff set)* for** L1 and L2 caches-D. Justi fi succinctly.

1.0 val. (B) Tell on' humerus and the width (n' number of bits) required of comparators for the L2 cache. Justi fi succinctly.

3.0 val. (W) Introduce the expression of m'edio time access to data. Indicate the meaning of each term of the expression.

2.5 val.

(D) Determine the failure rate in the L1 D-cache in the execution c~ao the next section of c'Odigo:

```
static double X [1024] Y [1024];
. . .
Y [0] = a0 * X [0] / 2; for (i = 1; i <1024; i
++) {
        Y [i] = a0 * X [i] + A1 * X [i-1] + b1 * Y [i-1]; }
```

Take a writing pol'ıtica *writeback* and a pol'ıtica of aloca¸c~ to *write-allocate.* Assume that X = Y = 00016000h and 00018000h, the vari' ables i N, a0, to 1 and b1 s stored in the registers, and the compiler n~ to perform any op-miza¸c~to the c' Odigo indicated, making *load* the vari' ables X and Y in order (ie, the left to right) and making *store* the vari' Hazel Y [i] in each itera¸c~ to the cycle.

2.5 val.

(and) Repeat the previous al'ınea considering a written pol'ıtica *write-through* and to allocate pol'ıtica dog *write-not-allocate.*

5. Consider a 64-bit processor supporting an area of f'ısico 16PB address space and virtual address 8 TB, endere¸c'avel byte by byte. Consider that the TRANS system c~ao virtual mem'oria'e multi-n'ıvel with PPAGES (and p'tables PAGES) 8KB and entries in the p'aginas 8-byte tables.

2.0 val. (The) Determine on' upper arm of n'ıveis necess' Aryans' the Tradu c~ao virtual address physical address.

2.0 val. (B) Represent tradu¸c~ scheme the address of a process with segments **program (instru¸c~oes) and data *heap* organized from the address 000h and 00 ... *stack* located from the address F ... FFh (the *stack* It grows in the direction of decreasing addresses).**

2.0 val. (W) Determine the m'ınimo space mem' oria necess' Aryan `the Tradu c~ao of address cos **a process and program *heap* 10MB, and *stack* of 15KB.**

6. Consider a system heterog'eneo constitu'ıdo for 1 processor *host* 1 and core an accelerator with 10 colors SMT, each capable of simultaneously at'e 20 threads, but at a slower pace than 10x processor *host.*

3.0 val.

(The) Assume that you want to run a program in heterog'eneo system which'and constitu'ıdo of 4 processing steps:

A. c' Odigo with purely sequential execution time in *host* 2s.

B. cycle *is* with 2000 itera¸c~ oes without dependencies between itera¸c~ oes and time of execu¸c~ to the *host* 1s.

B. cycle *is* to 2000 × 2000 ITERA c~oes without dependencies between itera¸c~ oes and time execu¸c~ to the *host* 50s.

D. c' Odigo with purely sequential execution time in *host* 1s. Admit that execu¸c~ to each of the stages it depends only on data generated the previous stage and the time of communication of data'e:

- $T_{host \to The\ accel \to B} T_{accel \to host}$ $_{THE \to B}$ = 10 *ms,*
- $T_{host \to B\ accel \to W} = T_{accel \to host}$ $_{B \to W}$ = 100 *ms,*
- $T_{host \to accel\ C \to D} = T_{accel \to host}$ $_{W \to D}$ = 1 *s.*

Calculate *speed-up* m' aximo that'e poss'ıvel get with heterog'eneo system, compara¸c~ with the execu¸c~ only the processor *host.* To do so, determine wherein the processor (host or accelerator) more advantageous to run each program phase.