EXERCISE 1 (A)

Given the following loop taken from a high level program:

```
do      {
         BASEC[i] = BASEA[i] + BASEB[i] + INC1 + INC2;
         i++;
        }
while (i != N)
```

The program has been compiled in MIPS assembly code assuming that registers $4 and $7 have been initialized with values 0 and 4N respectively. The symbols BASEA, BASEB and BASEC are 16-bit constant. The processor clock cycle is 2 ns.

```
L1:     lw   $2, BASEA ($4)
        addi $2, $2, INC1
        lw   $3, BASEB ($4)
        addi $3, $3, INC2
        add  $5, $2, $3
        sw   $5, BASEC ($4)
        addi $4, $4, 4
        bne  $4, $7, L1
```

- Let us consider a single iteration of the loop executed by **5-stage pipelined MIPS processor with optimized pipeline, where in the Register File, it is possible to read and write at the same address at the same clock cycle;**

- **Draw the pipeline scheme by marking in RED the RAW (Read After Write) data hazards and in BLUE the control hazards:**

Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	Type of Hazards
`L1: lw $2,BASEA($4)`	IF	ID	EX	M	WB								
`addi $2,$2,INC1`		IF	ID	EX	M	WB							RAW $2
`lw $3,BASEB($4)`			IF	ID	EX	M	WB						
`addi $3,$3,INC2`				IF	ID	EX	M	WB					RAW $3
`add $5,$2,$3`					IF	ID	EX	M	WB				RAW $3
`sw $5,BASEC($4)`						IF	ID	EX	M	WB			RAW $5
`addi $4, $4, 4`							IF	ID	EX	M	WB		
`bne $4,$7,L1`								IF	ID	EX	M	WB	RAW $4
									IF				CNTR

- **Insert in the following pipeline scheme the STALLS for each stage needed to solve the previous data and control hazards**
- **Indicate synthetically in the first column the NUMBER OF STALLS to be inserted in each instruction to solve data and control hazards**

Instruction	No stalls	C 1	C 2	C 3	C 4	C 5	C 6	C 7	C 8	C 9	C 10	C 11	C 12	C 13	C 14	C 15	C 16	C 17	C 18	C 19	C 20	C 21	C 22
`L1:lw $2,BASEA($4)`		IF	ID $4	EX	M	WB $2																	
`addi $2,$2,INC1`	*2*		IF	ID S	ID S	ID $2	EX	M	WB $2														
`lw $3,BASEB($4)`				IF S	IF S	IF	ID $4	EX	M	WB $3													
`addi $3,$3,INC2`	*2*						IF	ID S	ID S	ID $3	EX	M	WB $3										
`add $5,$2,$3`	*2*							IF S	IF S	IF	ID S	ID S	ID $2,$3	EX	M	WB $5							
`sw $5,BASEC($4)`	*2*										IF S	IF S	IF	ID S	ID S	ID $4,$5	EX	M	WB				
`addi $4, $4, 4`														IF S	IF S	IF	ID $4	EX	M	WB $4			
`bne $4,$7,L1`	*2*																IF	ID S	ID S	ID $4,$7	EX	M	WB
`Next instruction`	*3*																	IF S	IF S	IF S	IF S	IF S	IF

- **Express the formulas, then calculate the following metrics:**
  - Instruction Count per iteration **(IC)** = **8**
  - Number of stalls per iteration = **13**
  - **CPI** per iteration: **CPI = Number of clock cycles / IC = (IC+ # stalls + 4) /IC = 25 / 8 = 3.125**
  - **Throughput** (expressed in **MIPS**) per iteration: **MIPS =** f_{CLOCK} **/ (CPI * 10^6) = (500 * 10^6) / (3.125 * 10^6) = 160**
  - Asymptotic **CPI** (**N** cycles): **CPI** $_{AS}$**= (IC + # stalls) / IC = (8 + 13) / 8 = 2.625**
  - Asymptotic **Throughput** (expressed in **MIPS**) (**N** cycles): **MIPS**$_{AS}$ **=** f_{CLOCK} **/ (CPI**$_{AS}$ *** 10^6) = (500 * 10^6) / (2.625 * 10^6) = 190**

EXERCISE 1 (B)

Let us assume the following ***optimizations*** have been introduced in the pipeline:

- **Forwarding paths;**
- **Early evaluation of branch in the ID stage;**

- **Draw the pipeline scheme by marking in RED the RAW (Read After Write) data hazards and in BLUE the control hazards still remaining in the pipeline:**

Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14
`L1: lw$2,BASEA($4)`	IF	ID	EX	M	WB									
`addi $2,$2,INC1`		IF	ID	EX	M	WB								
`lw $3,BASEB($4)`			IF	ID	EX	M	WB							
`addi $3,$3,INC2`				IF	ID	EX	M	WB						
`add $5,$2,$3`					IF	ID	EX	M	WB					
`sw $5,BASEC($4)`						IF	ID	EX	M	WB				
`addi $4, $4, 4`							IF	ID	EX	M	WB			
`bne $4,$7,L1`								IF	ID	EX	M	WB		
									IF					

- **Insert in the following pipeline scheme the stalls needed to solve the hazards by marking in GREEN the forwarding paths used;**
- **Indicate synthetically in the last column the number of stalls to be inserted in each instruction to solve the hazards:**

Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	No. Stalls + Forwarding path
`L1: lw $2,BASEA($4)`	IF	ID $4	EX	M $2	WB $2													
`addi $2,$2,INC1`		IF	IDS	ID $2	EX $2	M $2	WB $2											1 Stall + **MEM-EX $2**
`lw $3,BASEB($4)`			IFS	IF	ID $4	EX	M $3	WB $3										
`addi $3,$3,INC2`					IF	IDS	ID $3	EX $3	M $3	WB $3								1 Stall + MEM-EX **$3**
`add $5,$2,$3`						IFS	IF	ID $2,$3	EX $5	M $5	WB $5							**EX-EX $3**
`sw $5,BASEC($4)`								IF	ID $4,$5	EX	M	WB						**EX-EX $5**
`addi $4, $4, 4`									IF	ID $4	EX $4	M $4	WB $4					
`bne $4,$7, L1`										IF	IDS	ID $4,$7	EX	M	WB			1 Stall + **EX-ID $4**
`Next instr.`											IFS	IFS	IF	ID	EX	M	WB	1 stall

Express the formulas, then calculate the following metrics:

- Instruction Count per iteration **(IC)** = **8**
- Number of stalls per iteration = **4**
- **CPI** per iteration: **CPI = # cycles / IC = (IC+ # stalls + 4) /IC = 16 / 8 = 2**
- **Throughput** (expressed in **MIPS**) per iteration: **MIPS =** f_{CLOCK} **/ (CPI * 10^6) = (500 * 10^6) / (2 * 10^6) = 250**
- Asymptotic **CPI** (**N** cycles) : **CPI** $_{AS}$**= (IC+ # stalls) / IC = (8 + 4) / 8 = 1,5**
- Asymptotic **Throughput** (expressed in **MIPS**) (**N** cycles): **MIPS**$_{AS}$ **=** f_{CLOCK} **/ (CPI**$_{AS}$ *** 10^6) = (500 * 10^6) / (1,5 * 10^6) = 333,33**

EXERCISE 1 (C)

Besides the previous optimizations, we assume to introduce **Static branch prediction for backward branches: branch always taken**

- **Draw the pipeline scheme by marking in RED the RAW (Read After Write) data hazards and in BLUE the control hazards still remaining in the program:**

Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14
L1: lw$2,BASEA($4)	IF	ID	EX	**M**	WB									
addi $2,$2,INC1		IF	ID	**EX**	M	WB								
lw $3,BASEB($4)			IF	ID	EX	**M**	WB							
addi $3,$3,INC2				IF	ID	**EX**	M	WB						
add $5,$2,$3					IF	ID	EX	M	WB					
sw $5,BASEC($4)						IF	ID	EX	M	WB				
addi $4, $4, 4							IF	ID	**EX**	M	WB			
bne $4,$7,L1								IF	**ID**	EX	M	WB		

- **Insert in the following pipeline scheme the stalls needed to solve the hazards by marking in GREEN the forwarding paths used;**
- **Indicate synthetically in the last column the number of stalls to be inserted in each instruction to solve the hazards:**

Instruction	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	No. Stalls + Forwarding path
`L1: lw $2,BASEA($4)`	IF	ID $4	EX	M $2	WB $2												
`addi $2,$2,INC1`		IF	IDS	ID $2	EX $2	M $2	WB $2										1 Stall + **MEM-EX $2**
`lw $3,BASEB($4)`			IFS	IF	ID $4	EX	M $3	WB $3									
`addi $3,$3,INC2`					IF	IDS	ID $3	EX $3	M $3	WB $3							1 Stall + **MEM-EX $3**
`add $5,$2,$3`						IFS	IF	ID $2,$3	EX $5	M $5	WB $5						**EX-EX $3**
`sw $5,BASEC($4)`								IF	ID $4,$5	EX	M	WB					**EX-EX $5**
`addi $4, $4, 4`									IF	ID $4	EX $4	M $4	WB $4				
`bne $4,$7, L1`										IF	IDS	ID $4,$7	EX	M	WB		1 Stall + **EX-ID $4**
`L1: lw $2,BASEA($4)` `(Next iteration)`											IFS	IF	ID $4	EX	M $2	WB $2	

- **Express the formulas, then calculate the following metrics:**

  - Instruction Count per iteration **(IC)** = **8**
  - Number of stalls per iteration = **3**
  - Asymptotic **CPI** (**N** cycles) : **CPI** $_{AS}$**= (IC + # stalls) / IC = (8 + 3) / 8 = 1,375**
  - Asymptotic **Throughput** (expressed in **MIPS**) (**N** cycles): **MIPS**$_{AS}$ **=** f_{CLOCK} **/ (CPI**$_{AS}$ *** 10**^{6}**) = (500 * 10**^{6}**) / (1,375 * 10**^{6}**) = 363,6**

Please notice that at the end of the loop iterations, the static branch prediction will fail and there will be needed 1 stall before fetching the next instruction.

EXERCISE 1 (D)

Besides the previous optimizations, we assume the **scheduling** of the assembly program has been optimized as follows:

```
L1:     lw   $2, BASEA ($4)
        lw   $3, BASEB ($4)
        addi $4, $4, 4
        addi $2, $2, INC1
        addi $3, $3, INC2
        add  $5, $2, $3
        bne  $4, $7, L1
        sw   $5, (BASEC-4) ($4)    # branch delay slot
```

- **Draw the pipeline scheme by marking in RED the RAW (Read After Write) data hazards and in BLUE the control hazards still remaining in the program (if any) and adding by GREEN ARROWs the number of stalls and the forwarding paths used to solved the hazards:**

Istruzione	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	No. stalls+ Forwarding Path
`L1: lw $2,BASEA($4)`	IF	ID $4	EX	M $2	WB $2								
`lw $3,BASEB($4)`		IF	ID $4	EX	M $3	WB $3							
`addi $4, $4, 4`			IF	ID $4	EX $4	M $4	WB $4						
`addi $2,$2,INC1`				IF	ID $2	EX $2	M $2	WB $2					
`addi $3,$3,INC2`					IF	ID $3	EX $3	M $3	WB $3				
`add $5,$2,$3`						IF	ID $2,$3	EX $5	M $5	WB $5			**MEM-EX$2 + EX-EX$3**
`bne $4,$7,L1`							IF	ID $4,$7	EX	M	WB		
`sw $5,BASEC-4($4)`								IF	ID $4,$5	EX	M	WB	**EX-ID $5**

- **Express the formulas, then calculate the following metrics:**
  - Instruction Count per iteration **(IC)** = **8**
  - Number of stalls per iteration = **0**
  - Asymptotic **CPI** (**N** cycles) : **CPI** $_{AS}$ **= (IC + # stalls) / IC = 1 = CPI** $_{ideal}$
  - Asymptotic **Throughput** (expressed in **MIPS**) (**N** cycles): **MIPS**$_{AS}$ **=** MIPS $_{ideal}$ $= f_{CLOCK} / (CPI_{ideal} * 10^6) = (500 * 10^6) / (1 * 10^6) = 500$

EXERCISE 1 (E)

We assume that, in the previously scheduled and optimized program, each READ access in the **MEM** phase to the data cache generates a **DATA CACHE MISS** requiring **2** stalls to access the memory:

```
L1:     lw   $2, BASEA ($4)
        lw   $3, BASEB ($4)
        addi $4, $4, 4
        addi $2, $2, INC1
        addi $3, $3, INC2
        add  $5, $2, $3
        bne  $4, $7, L1
        sw   $5, (BASEC-4) ($4)    # branch delay slot
```

- **Draw the pipeline scheme by marking in RED the RAW (Read After Write) data hazards and in BLUE the control hazards still remaining in the program (if any) and adding by GREEN ARROWs the number of stalls and the forwarding paths used to solved the hazards:**

Istruzione	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	No. stalls+ Forwarding Path
`L1: lw $2,BASEA($4)`	IF	ID $4	EX	M S	M S	M $2	WB $2										2 stalls
`lw $3,BASEB($4)`		IF	ID $4	EX S	EX S	EX	M S	M S	M $3	WB $3							2 stalls
`addi $4, $4, 4`			IF	ID S	ID S	ID $4	EX S	EX S	EX $4	M $4	WB $4						
`addi $2,$2,INC1`						IF	ID S	ID S	ID $2	EX $2	M $2	WB $2					
`addi $3,$3,INC2`									IF	ID $3	EX $3	M $3	WB $3				
`add $5,$2,$3`										IF	ID $2,$3	EX $5	M $5	WB $5			**MEM-EX$2 + EX-EX$3**
`bne $4,$7,L1`											IF	ID $4,$7	EX	M	WB		
`sw $5,BASEC-4($4)`												IF	ID $4,$5	EX	M	WB	**EX-ID $5**

- **Calculate the following metrics:**
  - Instruction Count per iteration **(IC)** = **8**
  - Number of stalls per iteration = **4 for the memory accesses (2 data cache misses)**
  - Asymptotic **CPI** (**N** cycles) : **CPI** $_{AS}$ **= (IC + # stalls) / IC = (8 + 4) / 8 = 1,5**
  - Asymptotic **Throughput** (expressed in **MIPS**) (**N** cycles): **MIPS**$_{AS}$ **=** f_{CLOCK} **/ (CPI**$_{AS}$ *** 10**^{6}**) = (500 * 10**^{6}**) / (1,5 * 10**^{6}**) = 333,33**