EXERCISE 1 (H): TOMASULO

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 0.5 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
```

We assume the program be executed on a pipelined CPU with dynamic scheduling based on TOMASULO algorithm with:

- **2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE FU (LDU1) with latency 2 cycles**
- **2 RESERVATION STATIONS (RS3, RS4) + 1 ALU/BR FU (ALU1) with latency 1 cycle**
- **Check structural hazards for RS in ISSUE phase**
- **Check RAW hazards and Check structural hazards for FUs in START EXECUTE phase**
- **WRITE RESULT in RS and RF**
- **Static branch prediction for backward branches: branch always taken**

Please complete the Tomasulo table by assuming all cache hits:

ISTRUZIONE	ISSUE	START EXEC	WRITE RESULT	Hazards Type	RSi	UNIT	OP1	OP2	STORE BUFFER
`L1:lw $2,BASEA($4)`	1	2	4		RS1	LDU1	BASEA	$4	-
`addi $2, $2, INC1`	2	5	6	**RAW in $2**	RS3	ALU1	RS1	INC1	-
`lw $3,BASEB($4)`	3	5	7	**STRUCT LDU1**	RS2	LDU1	BASEB	$4	-
`addi $3,$3,INC2`	4	8	9	**RAW in $3**	RS4	ALU1	RS2	INC2	-
`add $5,$2,$3`	7	10	11	**Struct RS3+RAW $3+STRUCT ALU1**	RS3	ALU1	$2	RS4	-
`sw $5,BASEC($4)`	8	12	14	**RAW $5**	RS1	LDU1	BASEC	$4	RS3
`addi $4,$4,4`	10	12	13	**Struct RS4+ STRUCT ALU1**	RS4	ALU1	$4	4	-
`bne $4,$7, L1`	12	14	15	**Struct RS3+STRUCT ALU1+ RAW $4**	RS3	ALU1	RS4	$7	-

- Instruction Count per iteration **(IC)**: **8**
- **CPI = # cycles / IC = 15 / 8 = 1.875**

- **Calculate the speedup with respect to the first version of Scoreboard (EX 1.F):**

  Speedup = (Exec. Time Scoreboard 1.F) / (Exec. Time Tomasulo) = 18/15 = 1,2

EXERCISE 1 (I): TOMASULO

We assume the original program be executed on CPU with dynamic scheduling based on TOMASULO algorithm with:

- **2 RESERVATION STATIONS (RS1, RS2) + 2 LOAD/STORE FU (LDU1, LDU2) with latency 2 cycles**
- **2 RESERVATION STATIONS (RS3, RS4) + 2 ALU/BR FU (ALU1, ALU2) with latency 1 cycle**
- **Check structural hazards for RS in ISSUE phase**
- **Check RAW hazads and Check structural hazards for FUs in START EXECUTE phase**
- **WRITE RESULT in RS and RF**
- **Static branch prediction for backward branches: branch always taken**

Please complete the Tomasulo table by assuming all cache hits:

ISTRUZIONE	ISSUE	START EXEC	WRITE RESULT	Hazards Type	RSi	UNIT	OP1	OP2	STORE BUFFER
`L1:lw $2,BASEA($4)`	1	2	4		RS1	LDU1	BASEA	$4	-
`addi $2, $2, INC1`	2	5	6	**RAW in $2**	RS3	ALU1	RS1	INC1	-
`lw $3,BASEB($4)`	3	4	7	Struct CDB	RS2	LDU2	BASEB	$4	-
`addi $3,$3,INC2`	4	8	9	**RAW in $3**	RS4	ALU2	RS2	INC2	-
`add $5,$2,$3`	7	10	11	**Struct RS3**+ **RAW $3**	RS3	ALU1	$2	RS4	-
`sw $5,BASEC($4)`	8	12	14	**RAW $5**	RS1	LDU1	BASEC	$4	RS3
`addi $4,$4,4`	10	11	12	(WAR $4 ok)	RS4	ALU2	$4	4	-
`bne $4,$7, L1`	12	13	14	**Struct RS3**	RS3	ALU1	RS4	$7	-

- Instruction Count per iteration **(IC)**: **8**
- **CPI = # cycles / IC = 14 / 8 = 1.75**

- **Calculate the speedup with respect to the previous version of Tomasulo (EX 1.H):**

  Speedup = (Exec. Time Tomasulo 1.H) / (Exec. Time Tomasulo) = 15/14 = 1,07

Given the following loop expressed in a high level language:

EXERCISE 2 (F) - TOMASULO

```
for (i =0; i < N; i ++)
      vectC[i] = vectA[i] + vectB[i];
```

The program has been compiled in MIPS assembly code assuming that registers `$t6` and `$t7`have been initialized with values 0 and N respectively. The symbols VECTA, VECTB and VECTC are 16-bit constant. The processor clock frequency is **2 GHz.**

INSTRUCTION	Comment
FOR1:beq $t6,$t7, END	# if ($t6 == $t7) goto END
lw $t2,VECTA($t6)	# $t2 <- VECTA [$t6];
lw $t3,VECTB($t6)	# $t3 <- VECTB [$t6];
add $t2,$t2,$t3	# $t2 <- $t2 + $t3;
sw $t2,VECTC($t6)	# VECTC[$t6] <- $t2;
addi $t6,$t6,4	# $t6 <- $t6 + 4;
j FOR1	# goto FOR1;
END:	

We assume the program be executed on a CPU with dynamic scheduling based on TOMASULO algorithm with:

- 2 RESERVATION STATIONS (RS1, RS2) + 1 LOAD/STORE FU (LDU1) with latency 4 cycles
- 2 RESERVATION STATIONS (RS3, RS4) + 1 ALU/BR FU (ALU1) with latency 2 cycles
- Check structural hazards for RS in ISSUE phase
- Check RAW hazards and Check structural hazards for FUs in START EXECUTE phase
- WRITE RESULT in RESERVATION STATIONS and RF
- Static Branch Prediction BTFNT (BACKWARD TAKEN FORWARD NOT TAKEN) with Branch Target Buffer

Please complete the Tomasulo table by assuming all cache hits:

ISTRUZIONE	PRED. T / NT	ISSUE	START EXEC	WRITE RESULTS	HAZARDS TYPE	RSi	UNIT	OP1	OP2	STORE BUFFER
`FOR1:beq $t6,$t7, FOR2`	NT	1	2	4		RS3	ALU1	$t6	$t7	-
`lw $t2,VECTA($t6)`	-	2	3	7		RS1	LDU1	VECTA	$t6	-
`lw $t3,VECTB($t6)`	-	3	8	12	STRUCT LDU1	RS2	LDU1	VECTB	$t6	-
`add $t2,$t2,$t3`	-	4	13	15	RAW $t2 ,RAW $t3	RS4	ALU1	RS1	RS2	-
`sw $t2,VECTC($t6)`	-	8	16	20	STRUCT RS1 +(STRUCT LDU1)+ RAW $t2	RS1	LDU1	VECTA	$t6	RS4
`addi $t6,$t6,4`	-	9	16	18	STRUCT ALU1	RS3	ALU1	$t6	4	-
`j FOR1`	T	16	19	**21**	STRUCT RS4 +STRUCT ALU1	RS4	ALU1	FOR1		-

- Given a clock frequency of 2 GHz, express the FORMULA then calculate the following metrics:
  - CPI = # cycles / IC = 21 / 7 = 3
  - IPC = 1 / CPI = 1 / 3 = 0.33
  - **Throughput** (expressed in **MIPS**): MIPS = f_{CLOCK} / (CPI * 10^6) = (2 * 10^9) / (3 * 10^6) = 666.67

EXERCISE 2 (G) - TOMASULO

We assume the original program be executed on CPU with dynamic scheduling based on TOMASULO algorithm with:

- **2 RESERVATION STATIONS (RS1, RS2) + 2 LOAD/STORE FU (LDU1, LDU2) with latency 4 cycles**
- **2 RESERVATION STATIONS (RS3, RS4) + 2 ALU/BR FU (ALU1, ALU2) with latency 2 cycles**
- **Check structural hazards for RS in ISSUE phase**
- **Check RAW hazards and Check structural hazards for FUs in START EXECUTE phase**
- **WRITE RESULT in RESERVATION STATIONS and RF**
- **Static Branch Prediction BTFNT (BACKWARD TAKEN FORWARD NOT TAKEN) with Branch Target Buffer**

Please complete the Tomasulo table by assuming all cache hits:

ISTRUZIONE	PRED. T / NT	ISSUE	START EXEC	WRITE RESULTS	HAZARDS TYPE	RSi	UNIT	OP1	OP2	STORE BUFFER
`FOR1:beq $t6,$t7, FOR2`	NT	1	2	4		RS3	ALU1	$t6	$t7	-
`lw $t2,VECTA($t6)`	-	2	3	7		RS1	LDU1	VECTA	$t6	-
`lw $t3,VECTB($t6)`	-	3	4	8		RS2	LDU2	VECTB	$t6	-
`add $t2,$t2,$t3`	-	4	9	11	RAW $t2, RAW $t3	RS4	ALU2	RS1	RS2	-
`sw $t2,VECTC($t6)`	-	8	12	**16**	STRUCT RS1 + RAW $t2	RS1	LDU1	VECTA	$t6	RS4
`addi $t6,$t6,4`	-	9	10	12		RS3	ALU1	$t6	4	-
`j FOR1`	T	12	13	15	STRUCT RS4	RS4	ALU2	FOR1		-

- **Given a clock frequency of 2 GHz, express the FORMULA then calculate the following metrics:**
  - **CPI = # cycles / IC = 16 / 7 = 2.29**
  - **IPC = 1 / CPI = 0.44**
  - **Throughput** (expressed in **MIPS**): **MIPS =** f_{CLOCK} **/ (CPI * 10^6) = (2 * 10^9) / (2.29 * 10^6) = 873.36**
- **Calculate the Speedup obtained with respect to the previous case (EX1.A):**
  - **Speedup = CPI1A / CPI1B = 3/ 2.29 = 1.31 Tomasulo 1B is better than Tomasulo 1A by 31%**

EXERCISE 2 (H) - TOMASULO

We assume the original program be executed on CPU with dynamic scheduling based on TOMASULO algorithm with:

- **2 RESERVATION STATIONS (RS1, RS2) + 2 LOAD/STORE FU (LDU1, LDU2) with latency 4 cycles**
- **2 RESERVATION STATIONS (RS3, RS4) + 2 ALU/BR FU (ALU1, ALU2) with latency 2 cycles**
- **Check structural hazards for RS in ISSUE phase**
- **Check RAW hazards and Check structural hazards for FUs in START EXECUTE phase**
- **WRITE RESULT in RESERVATION STATIONS and RF**
- **Static Branch Prediction BTFNT (BACKWARD TAKEN FORWARD NOT TAKEN) with Branch Target Buffer**

Please complete the Tomasulo table by assuming all instruction cache hits and all data cache write hits but all data cache read misses introducing 4 stalls cycles to WRITE RESULTS:

ISTRUZIONE	PRED. T / NT	ISSUE	START EXEC	WRITE RESULTS	HAZARDS TYPE	RSi	UNIT	OP1	OP2	STORE BUFFER
`FOR1:beq $t6,$t7, FOR2`	NT	1	2	4		RS3	ALU1	$t6	$t7	-
`lw $t2,VECTA($t6)`	-	2	3	11	DATA CACHE READ MISS	RS1	LDU1	VECTA	$t6	-
`lw $t3,VECTB($t6)`	-	3	4	12	DATA CACHE READ MISS	RS2	LDU2	VECTB	$t6	-
`add $t2,$t2,$t3`	-	4	13	15	**RAW $t2, RAW $t3**	RS4	ALU2	RS1	RS2	-
`sw $t2,VECTC($t6)`	-	12	16	**20**	**STRUCT RS1 + RAW $t2**	RS1	LDU1	VECTA	$t6	RS4
`addi $t6,$t6,4`	-	13	14	16		RS3	ALU1	$t6	4	-
`j FOR1`	T	16	17	19	**STRUCT RS4**	RS4	ALU2	FOR1		-

- **Given a clock frequency of 2 GHz, express the FORMULA then calculate the following metrics:**
  - **CPI = # cycles / IC = 20 / 7 = 2.86**
  - **IPC = 1 / CPI = 0.35**
  - **Throughput** (expressed in **MIPS**): **MIPS =** f_{CLOCK} **/ (CPI * 10^6) = (2 * 10^9) / (2.86 * 10^6) = 699.3**
- **Calculate the performance lost with respect to the previous case (EX1.B):**
  - **PL = CPI1C / CPI1B = 2.86/ 2.29 = 1.25 Tomasulo 1C is worst than Tomasulo 1B by 25%**