Checklist

- Project I –Computers –

I have read and understood the directions uploaded on Classroom regarding the implementation of the project. For my doubts, I asked the teaching staff for help during the project meetings.	☐
I extracted the instructions from the manual and created the table with all the instructions (common and specific)	☐
For each instruction I identified the affected flags .	☐
I defined the complete list of control signals and completed the truth table for the Control Unit. In the table, one-bit signals can only have the values ‚0', ‚1' or ‚X'. Multi-bit signals will have values combinations between '0', '1' or 'X'	☐
I have implemented in VHDL the Control Unit fully respecting the previously defined table (the names of the signals used in the table are identical to the names of the signals used in the implementation, if in a case a signal is '0' in the table then it is also '0' in the implementation, if it is '1' in the table then it is also '1' in implementation and if it is 'X' in the table then it can be '0' or '1' in implementation)	☐
I have implemented the common instructions (it is recommended to adapt the MIPS implementation that you did in the laboratory of Computer Structure and Organization, in year II, semester II)	☐
I created the ROM file and ran a "Behavioural Simulation" for " sequence A (all)" without errors. ROM File Name:	☐
I have implemented the Z flag for common instructions	☐
I created the ROM file and ran a "Behavioural Simulation" for " sequence Z (zero)" without errors. ROM File Name:	☐
I implemented the N flag for common instructions	☐
I created the ROM file and ran a "Behavioural Simulation" for " sequence N (negative)" without errors. ROM File Name:	☐
I have implemented the C flag for common instructions	☐
I created the ROM file and ran a "Behavioural Simulation" for " sequence C (carry)" without errors. ROM File Name:	☐
I have implemented the OV flag for common instructions	☐
I created the ROM file and ran a "Behavioural Simulation" for " sequence OV(overflow)" without errors . ROM File Name:	☐
I implemented the instruction "BRA Z, Expr "	☐
I created the ROM file and validated the implementation by executing, without errors, a " Behavioural Simulation ” by which I checked the behavior of the instruction when the flag is ‚0' and when the flag is ‚1'. ROM File Name:	☐

I implemented the instruction "BRA N, Expr "	☐
I created the ROM file and validated the implementation by executing, without errors, a " Behavioural Simulation ” by which I checked the behavior of the instruction when the flag is ,0' and when the flag is ,1'. ROM File Name:	☐
I implemented the instruction "BRA C, Expr "	☐
I created the ROM file and validated the implementation by executing, without errors, a " Behavioural Simulation ” by which I checked the behavior of the instruction when the flag is ,0' and when the flag is ,1'. ROM File Name:	☐
I implemented the instruction "BRA OV, Expr "	☐
I created the ROM file and validated the implementation by executing, without errors, a " Behavioural Simulation ” by which I checked the behavior of the instruction when the flag is ,0' and when the flag is ,1'. ROM File Name:	☐
I have implemented one of the specific instructions of the project Instruction Name: .... File containing the implementation: …….....	☐
I created the ROM file(s) and validated the implementation by running, without errors, one or several "Behavioural Simulation” through which I verified the nominal behavior of the instruction and the updating of the flags affected by the instruction. ROM file name/s:	☐
I have created the documentation as per the requirements	☐

By checking all the boxes up to this point, the maximum grade is 5. If at least one of the boxes is not checked then the grade is between 1 and 4

I have implemented o second of the specific instructions of the project Instruction Name: .... File containing the implementation: …….....	☐
I created the ROM file(s) and validated the implementation by running, without errors, one or several "Behavioural Simulation” through which I verified the nominal behavior of the instruction and the updating of the flags affected by the instruction. ROM file name/s:	☐
I have implemented a third of the specific instructions of the project Instruction Name: .... File containing the implementation: …….....	☐

I created the ROM file(s) and validated the implementation by running, without errors, one or several "Behavioural Simulation" through which I verified the nominal behavior of the instruction and the updating of the flags affected by the instruction. ROM file name/s:	☐

By checking all the boxes up to this point, the maximum grade is 8. If at least one of the boxes is not checked then the grade is between 1 and 7

I have implemented a fourth of the specific instructions of the project Instruction Name: .... File containing the implementation: …….....	☐
I created the ROM file(s) and validated the implementation by running, without errors, one or several "Behavioural Simulation" through which I verified the nominal behavior of the instruction and the updating of the flags affected by the instruction. ROM file name/s:	☐
I executed and validated all simulations using the "Post-Route Simulation" mode	☐

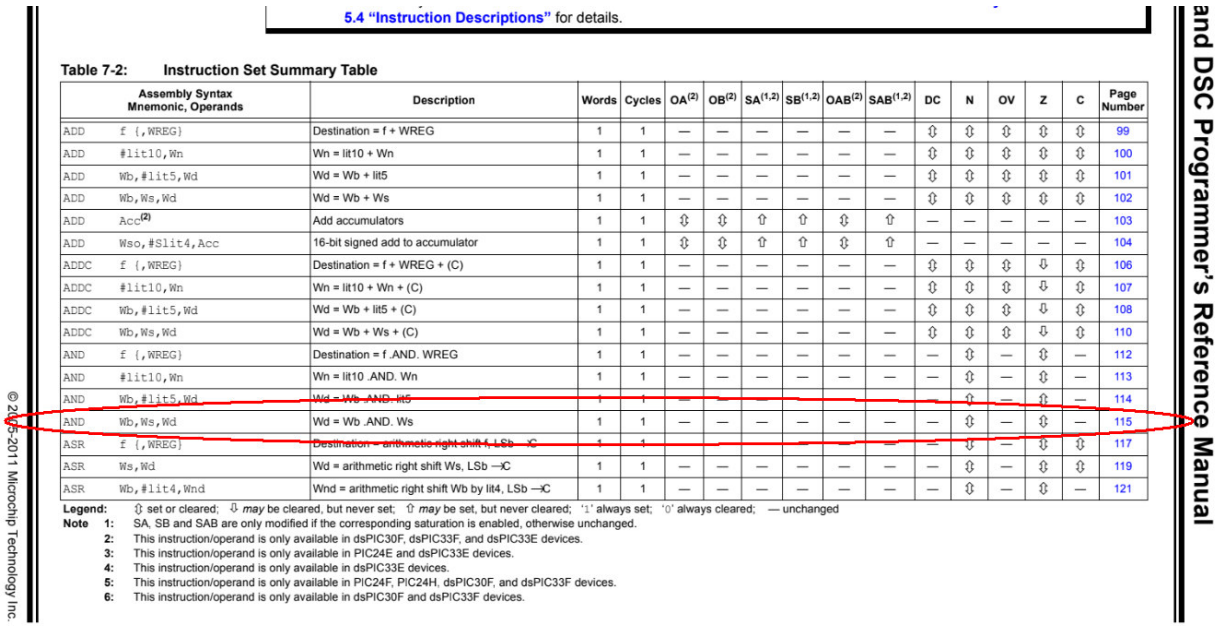
By checking all the boxes up to this point, the maximum grade is 10. If at least one of the boxes is not checked then the grade is between 1 and 9

Important Notes:

- It is recommended to validate together with the teaching staff, during the project sessions, the table with the instructions and the list of flags affected by each one before starting the implementation.
- Error-free execution of a simulation involves verifying that the registers, flags , instruction execution order, and memory locations modified by the simulation execution are 100% identical to those modified by the test sequence execution in the MPLab X IDE
- To test a conditional jump statement, you must force the value of the flag to '0', call the statement and check that no jump is made, then force the value of the flag to '1', call the statement and check that a jump is made. To force a flag to '0' or '1' you can take inspiration from existing check sequences.
    - Example of check sequence for BRA Z, Expr

```
LOOP: mov 0x1020, w1 ;INW0=ffff
      mov 0x1022, w2 ;INW1=0001
      add w2,w2,w7 ;0002, Z=0
      BRA Z, STOP ; no jump
      add w1,w2,w3 ;0000, Z=1
      bra Z, END ; jump to END
STOP: bra STOP   ; infinite loop
END:  bra LOOP   ;return to start
```

- Testing a project-specific instruction involves checking the instruction's nominal behavior and, as appropriate, generating the values 0 and 1 for the flags affected by the instruction. You can do all the checks in one program or make separate programs for each check. Program (s) may also use common statements or conditional jump statements.
    - Example test sequence for AND Wb, Ws , Wd (just as a model, you don't need to do tests like this for the common statements)
      The easiest is to identify the instruction in table 7.2 of the manual. ATTENTION: there are several variations of the AND statement. You have to identify the correct one according to the parameter list.
      In this case, the instruction you are looking for is in the manual on page 115.



5.4 "Instruction Descriptions" for details.

Table 7-2: Instruction Set Summary Table

Assembly Syntax Mnemonic, Operands		Description	Words	Cycles	OA(2)	OB(2)	SA(1,2)	SB(1,2)	OAB(2)	SAB(1,2)	DC	N	OV	Z	C	Page Number
ADD	f {,WREG}	Destination = f + WREG	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇕	⇕	99
ADD	#lit10,Wn	Wn = lit10 + Wn	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇕	⇕	100
ADD	Wb,#lit5,Wd	Wd = Wb + lit5	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇕	⇕	101
ADD	Wb,Ws,Wd	Wd = Wb + Ws	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇕	⇕	102
ADD	Acc(2)	Add accumulators	1	1	⇕	⇕	⇑	⇑	⇕	⇑	—	—	—	—	—	103
ADD	Wso,#Slit4,Acc	16-bit signed add to accumulator	1	1	⇕	⇕	⇑	⇑	⇕	⇑	—	—	—	—	—	104
ADDC	f {,WREG}	Destination = f + WREG + (C)	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇩	⇕	106
ADDC	#lit10,Wn	Wn = lit10 + Wn + (C)	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇩	⇕	107
ADDC	Wb,#lit5,Wd	Wd = Wb + lit5 + (C)	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇩	⇕	108
ADDC	Wb,Ws,Wd	Wd = Wb + Ws + (C)	1	1	—	—	—	—	—	—	⇕	⇕	⇕	⇩	⇕	110
AND	f {,WREG}	Destination = f .AND. WREG	1	1	—	—	—	—	—	—	—	⇕	—	⇕	—	112
AND	#lit10,Wn	Wn = lit10 .AND. Wn	1	1	—	—	—	—	—	—	—	⇕	—	⇕	—	113
AND	Wb,#lit5,Wd	Wd = Wb .AND. lit5	1	1	—	—	—	—	—	—	—	⇕	—	⇕	—	114
AND	Wb,Ws,Wd	Wd = Wb .AND. Ws	1	1	—	—	—	—	—	—	—	⇕	—	⇕	—	115
ASR	f {,WREG}	Destination = arithmetic right shift f, LSb →C	1	1	—	—	—	—	—	—	—	⇕	—	⇕	⇕	117
ASR	Ws,Wd	Wd = arithmetic right shift Ws, LSb →C	1	1	—	—	—	—	—	—	—	⇕	—	⇕	⇕	119
ASR	Wb,#lit4,Wnd	Wnd = arithmetic right shift Wb by lit4, LSb →C	1	1	—	—	—	—	—	—	—	⇕	—	⇕	—	121

Legend: ⇕ set or cleared; ⇩ *may* be cleared, but never set; ⇑ *may* be set, but never cleared; '1' always set; '0' always cleared; — unchanged

Note 1: SA, SB and SAB are only modified if the corresponding saturation is enabled, otherwise unchanged.
2: This instruction/operand is only available in dsPIC30F, dsPIC33F, and dsPIC33E devices.
3: This instruction/operand is only available in PIC24E and dsPIC33E devices.
4: This instruction/operand is only available in dsPIC33E devices.
5: This instruction/operand is only available in PIC24F, PIC24H, dsPIC30F, and dsPIC33F devices.
6: This instruction/operand is only available in dsPIC30F and dsPIC33F devices.

You can also see in Table 7.2 the list of affected flags. In this case only N and Z are affected. The double-headed arrow means that the flag can also be set (it receives the value 1) and reset (it receives the value 0). A down arrow would mean the flag is just reset and an up arrow would mean the flag is just set.
An example of a test program would be the following:

```
LOOP: mov 0x1020, w1 ;INW0=ffff
      mov 0x1022, w2 ;INW1=0001
      and w1,w2,w7 ;w7=0001, nominal case
      sub w1,w1,w3 ; w3=0000, prepare for flag test
      and w1,w2,w4 ; w4=0001, Z=0
      bra Z, STOP ; no jump
      and w1,w3,w5 ; w5=0000, Z=1
      bra Z, NEXT ; jump to NEXT
      bra STOP ; jump to stop.
NEXT: and w1,w2,w4 ; w4=0001, N=0
      bra N, STOP ; no jump
      and w1,w1,w5 ; w5=FFFF, N=1
      bra N, NEXT ; jump to NEXT
STOP: bra STOP   ; infinite loop
END:  bra LOOP   ;return to start
```

- Checking all the boxes in a section does not guarantee the maximum grade for that section. You must be able to answer the teacher's questions and if you fail to do so, your grade will drop. Incorrect ticking of a box automatically results in the grade dropping below the threshold for the section.