SP3_2 Implementing the control unit (FSM) and completing design phase #1

I. Specifications

The aim of this preparatory laboratory assignment is to analyse the given circuit for the programmable timer and complete hardware and software.

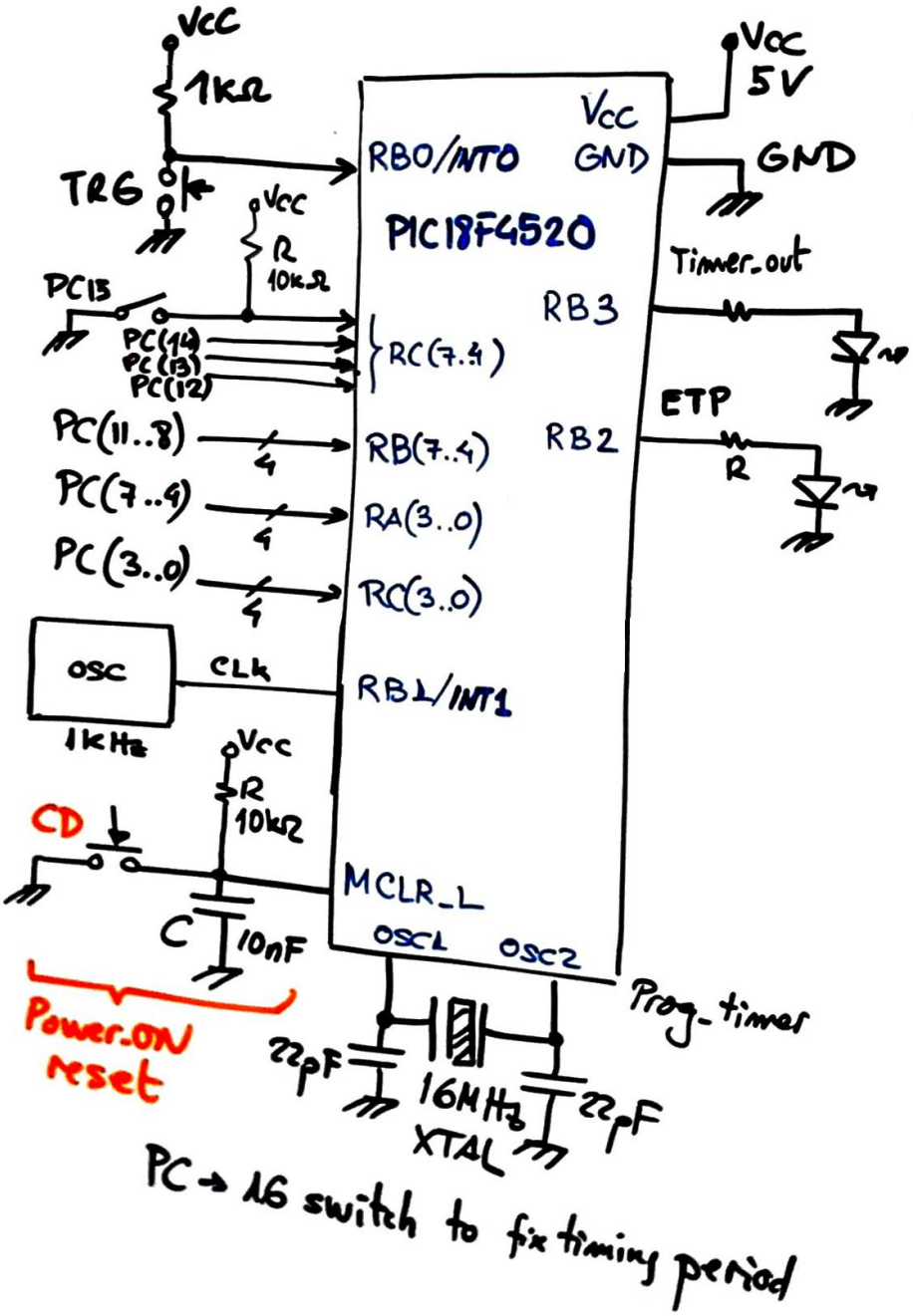


Fig. 1 Programmable timer.

II. Planning.

Study all the project sections and draw the truth table and its equivalent flowchart for *state_logic()*

III. Development, debugging

IV. Test

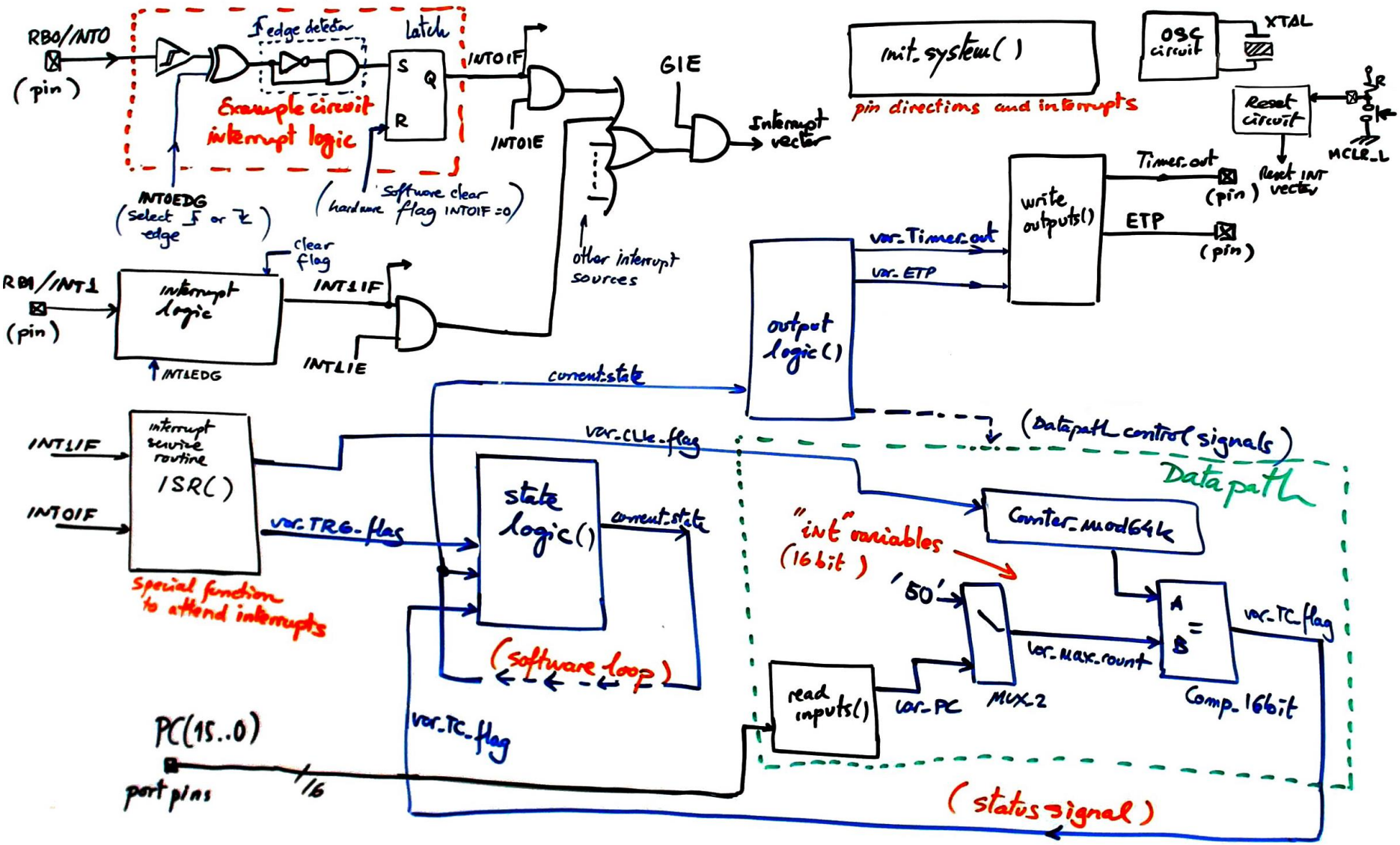
Write the *state_logic()* C code, start a new MPLABX project *Prog_Timer_prj*, compile and run and verify in Proteus that your circuit work as expected.

V. Measurements and questions

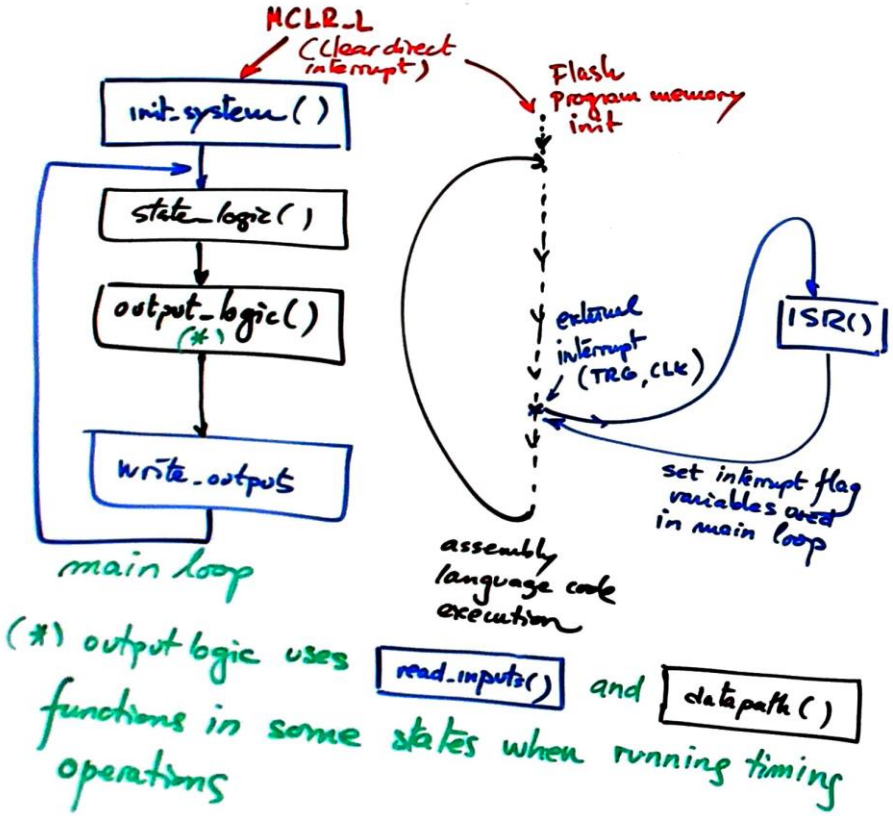
- Measure the circuit precision in timing periods when for instance PC = 3; PC = 3000
- Measure using breakpoints how long does it takes to execute the main loop.
- Measure how long does it takes to execute *ISR()*
- What happens if OSC crystal is replaced by 4 MHz?
- How to make TRG sensitive to rising edges?

Annex

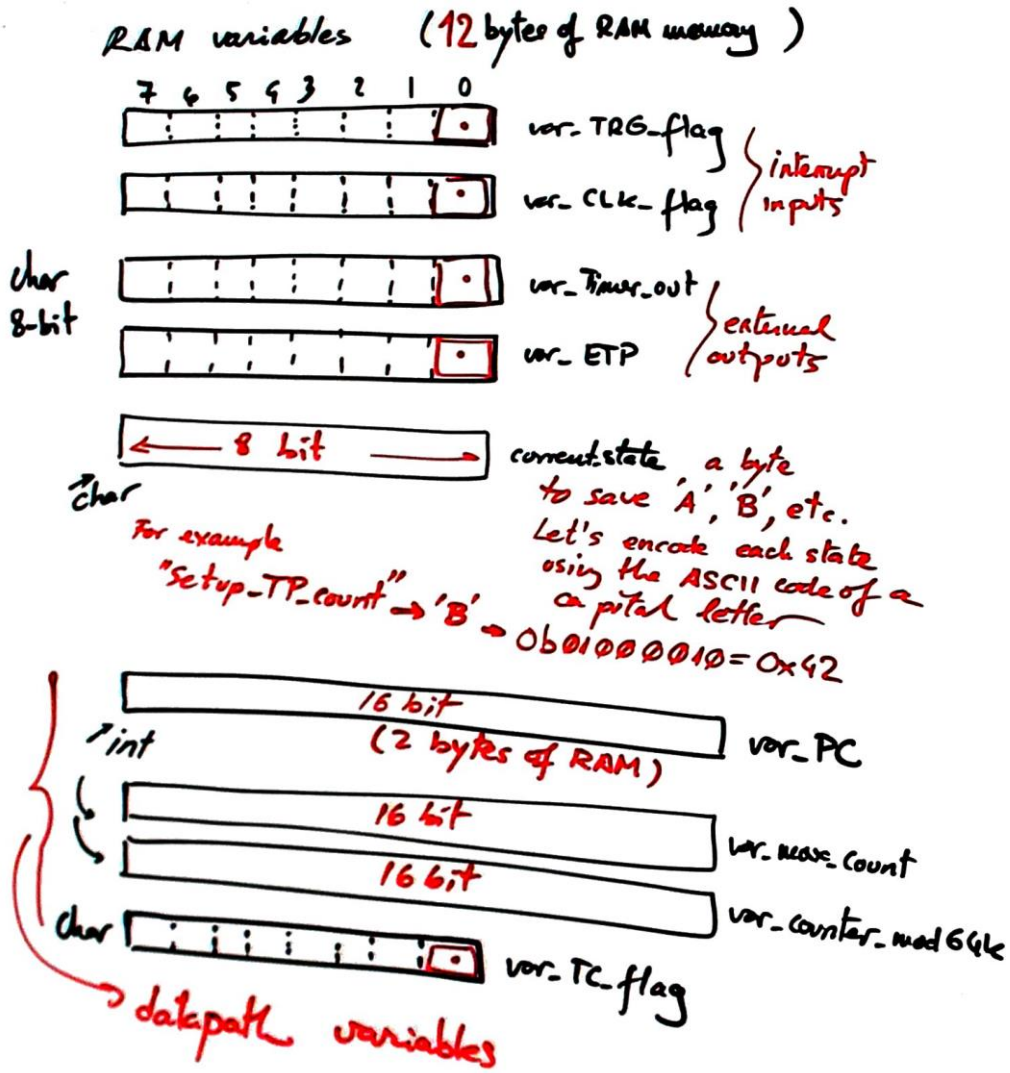
Paper work on diagrams, sketches, flowcharts and materials for studying the project in detail and completing it.
Example of hardware-software diagram:



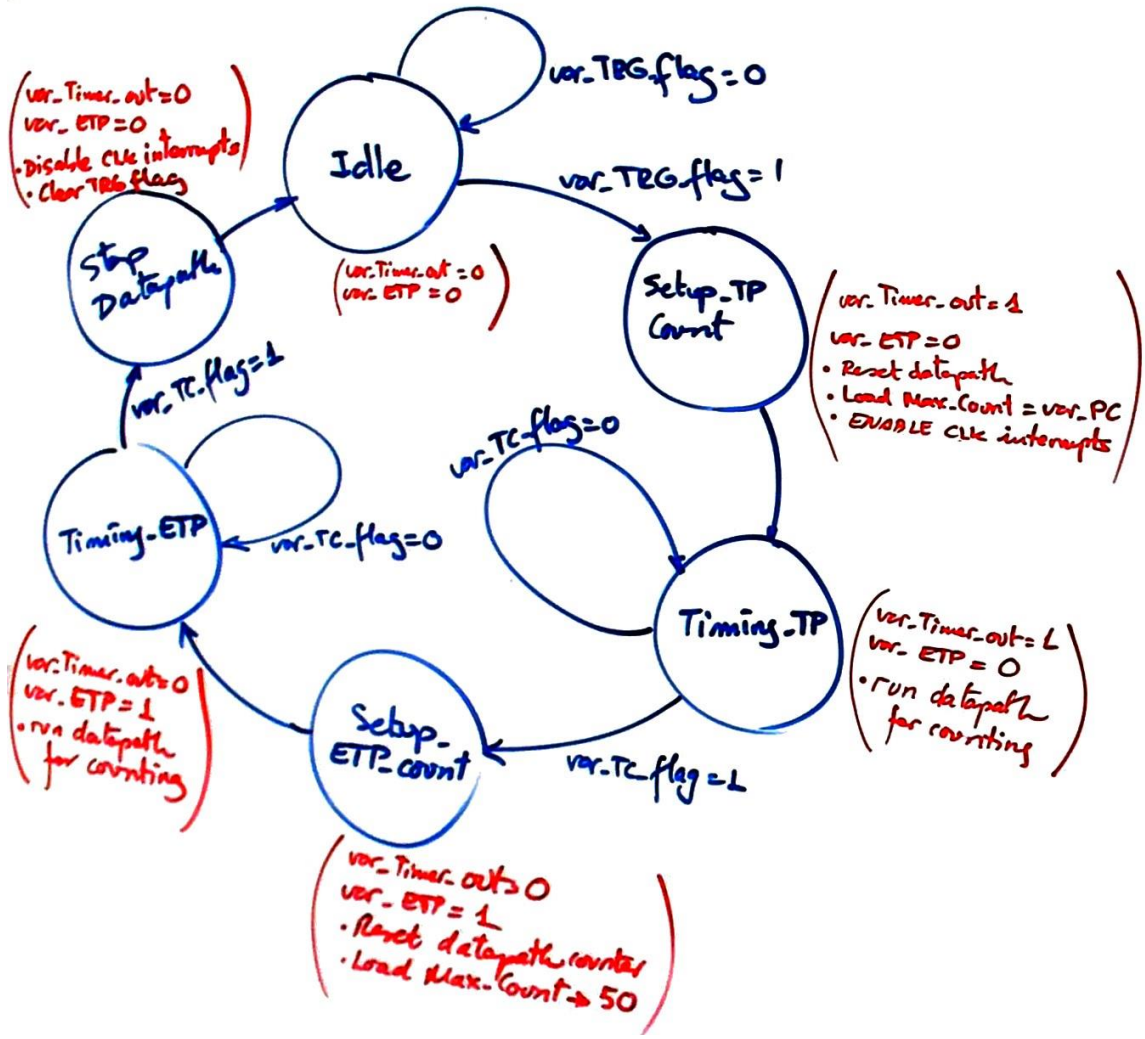
General software organisation



RAM variables

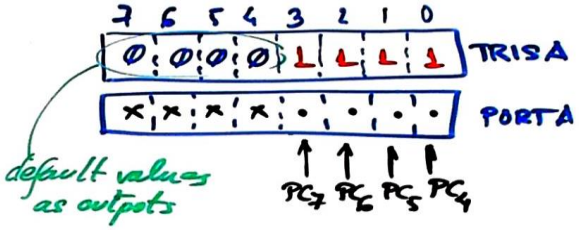


Proposed state diagram



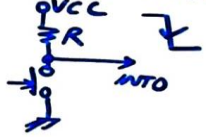
Initialising the system

✓ init_system() discussion (Complete the discussion)



✓ allow INT0 interrupts (TRG) → INT0IE=1;

✓ select INT0 falling edge

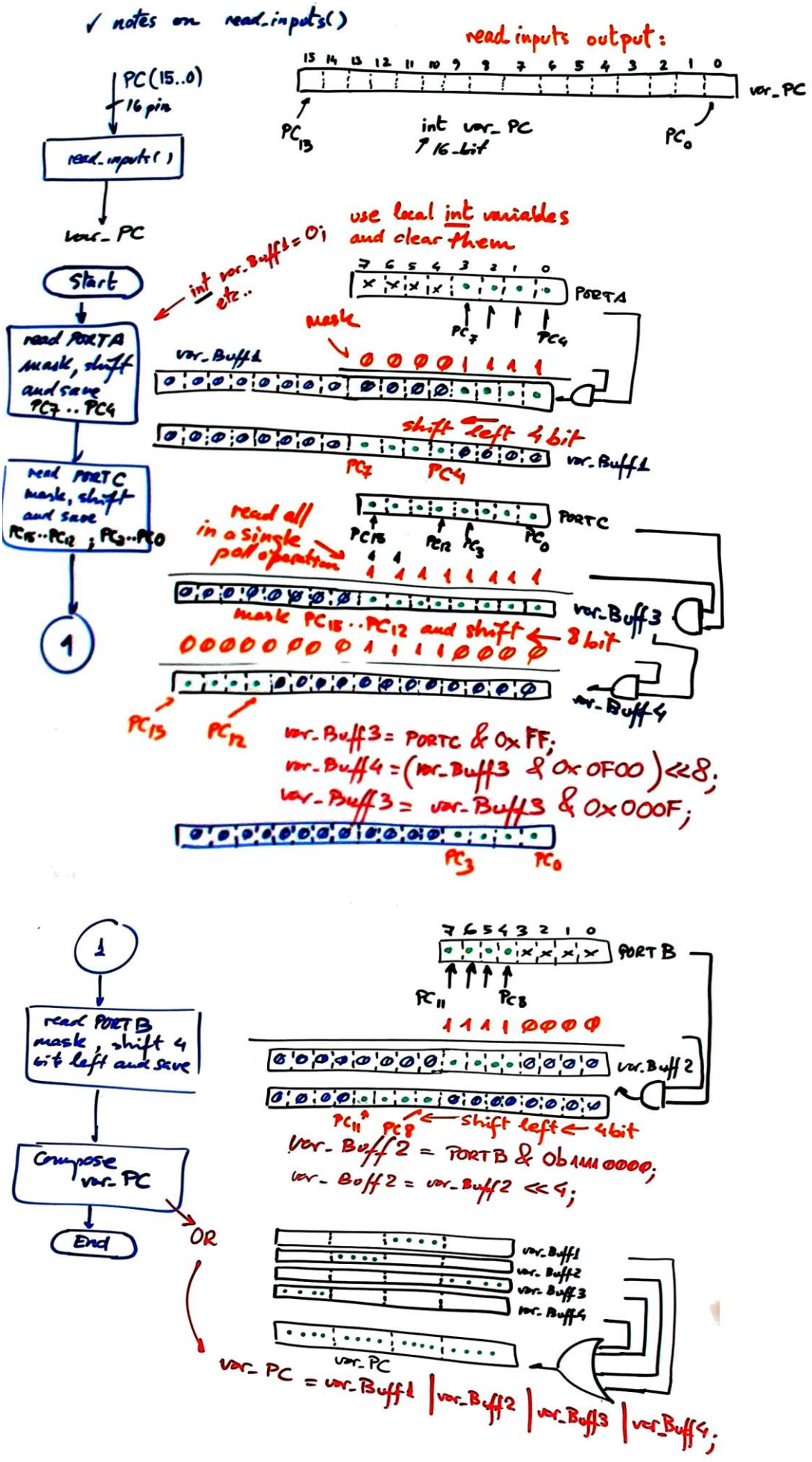


INT0EDG=0;

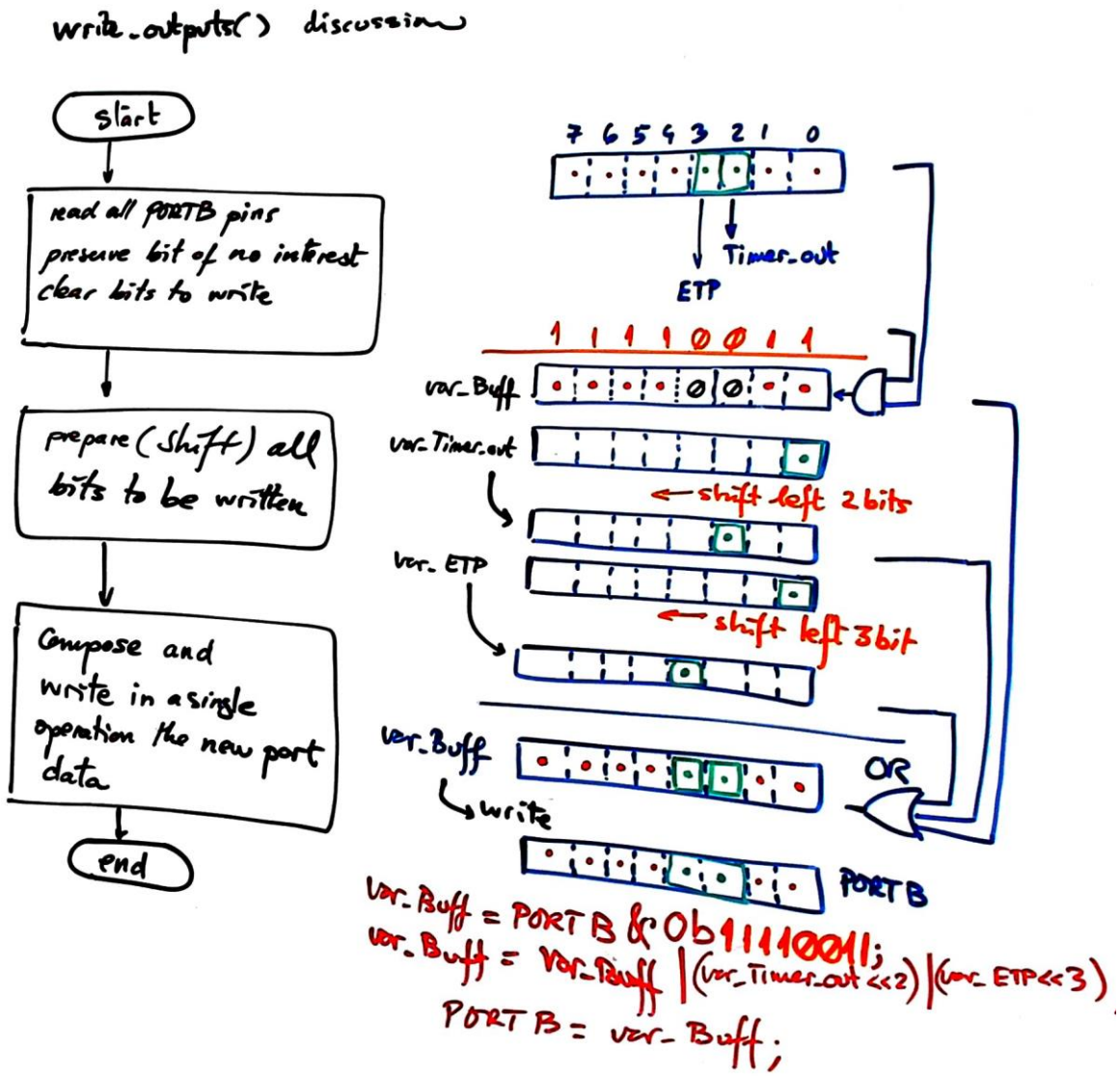
✓ disable INT1 interrupts (CLK) because the datapath is not required at Idle state → INT1IE=0;

✓ enable general interrupts → GIE=1; (to allow TRG interrupts when Idle)

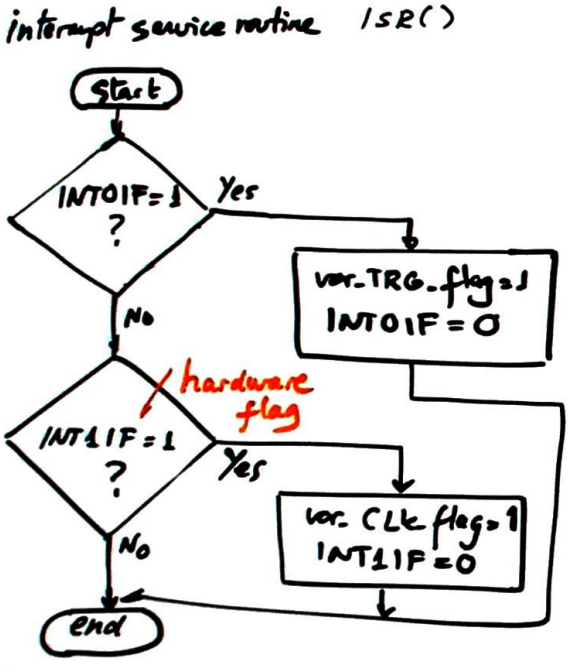
Reading inputs: PC(15..0)



Writing outputs

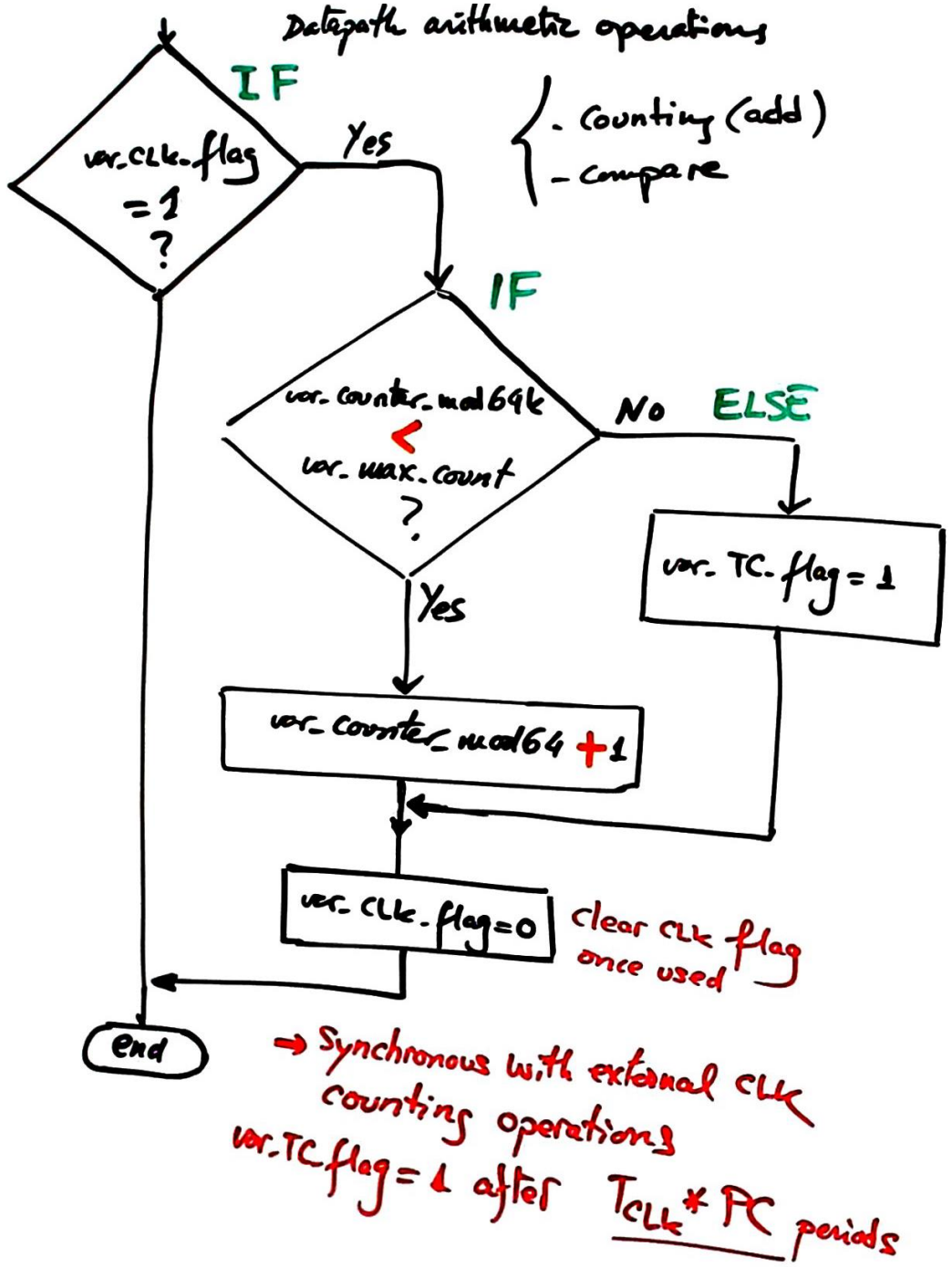


Interrupt service routine



- Clear hardware flag means that the system can be interrupted again by the same mechanism
- Software flag variables will be used in main programme and cleared once used

Counter structure in the datapath, a software/algorithmic/behavioural translation of the datapath operating resources in the hardware-software diagram.

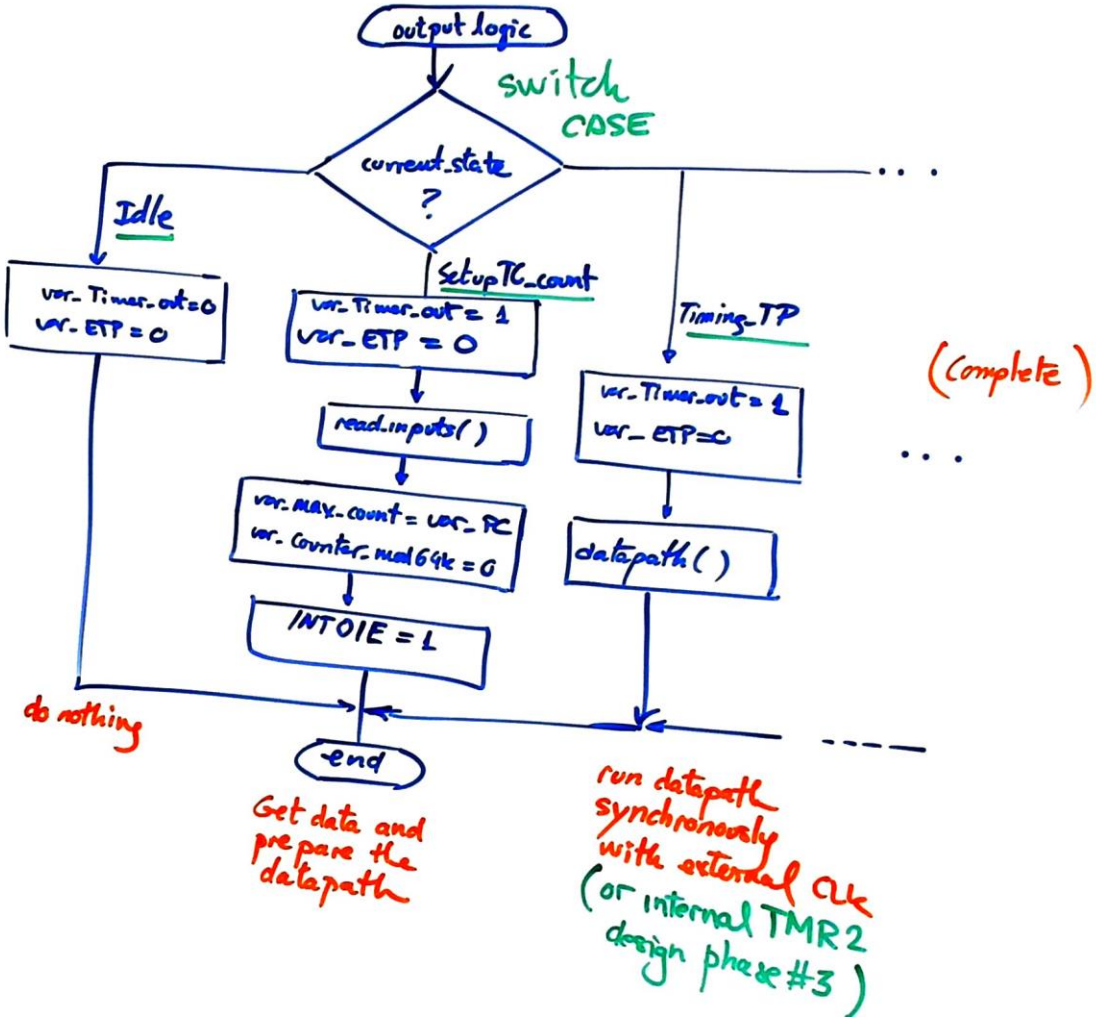


Output logic truth table and flowchart

output_logic truth table

state logic	var_ETP, var_Timer_out	Datapath variables	
Idle	0 0	x	
Setup_TP_count	0 1	read PC	← read_inputs();
		max_count = PC	← select
		Counter_mod64k = 0	← reset count
		enable CLK int	
Timing_TP	0 1	run datapath	← datapath();
Setup_ETP_count	1 0	max_count = 50 (50ms)	
		Counter_mod64k = 0	← reset count
Timing_ETP	1 0	run datapath	
Stop_datapath	0 0	disable CLK interrupts	

output logic drives outputs and controls datapath operations



State logic truth table and flowchart

Oscilloscope measurements and watch window for debug.