– Elettronica dei Sistemi Digitali –
Lab#6

ASM chart

This lab aims to develop your digital design skills. You will address a real problem starting from the functional specification and develop a circuit comprising several blocks, including memories, a control unit, and a data path. You will write and simulate the VHDL of the entire design. Unlike previous labs, **you are not required to download the code onto the FPGA circuit to test it**. Instead, you must design the circuit, document it, simulate it to validate your work, and then write a final report covering the following items:

- Description of the Design
- Complete scheme of the Datapath and the Control Unit so that anyone can implement a PCB (Printed Circuit Board) for your design starting solely from your schemes
- The logical procedure you used to validate your design using the TestBench
- Modelsim/Questa Simulation fragments to demonstrate your Work

In any case, at the end of these pages, you will be provided with all the materials you need to submit to the teachers for the evaluation of the lab.

The evaluation of your work will be based on your report. Submit your report before the deadline published on the "Portale della didattica" following these steps::

1. Produce a pdf file of your report. **Note** pdf is the **ONLY** accepted format.
2. The name of the pdf **MUST** be obtained by concatenating in alphabetical order the surnames of the people who worked on the preparation of the report. Special characters including spaces and apostrophe **MUST** be avoided. Surnames are separated by the underscore '_' character.
   Example: the group XYY (*e.g. C14*) with *Lennon*, *McCartney*, *Harrison* and *Starr* will produce a file named *sqC_tav14_lab06_harrison_lennon_mccartney_starr.pdf.*
3. Upload the pdf file on "Portale della didattica" in the section named "elaborati".

Note: the report can be part of the discussion during the exam.

Contents:

Abbreviations and acronyms:

ASM – Algorithmic State Machine
VHDL – Very high speed integrated circuits Hardware Description Language
CU – Control Unit

1 – Introduction

The digital system you will design serves as the controller for an automatic irrigation system. Its primary function is to process data collected from two soil sensors measuring water content. Initially, the system captures data from the soil sensors, available on a serial bus called DATA_IN, and stores it into memory (MEM_A). Subsequently, it conducts digital filtering and evaluation operations on the data stored in MEM_A. The filtered results are then stored in a second memory (MEM_B), with its output DATA_OUT connected to a transmission bus. Additionally, the designed digital system controls an output FLAG named IRRIGATIONALARM. Finally, the system signals the end of the protocol to the receiver by activating the signal DONE.

2 – Automatic irrigation system

Agriculture accounts for 70% of water usage worldwide, making water conservation fundamental in reducing its impact. Typically, irrigation is managed through timed systems or manually by farmers. However, modern advancements in smart agriculture, coupled with sensor technology, are increasingly being embraced and yielding promising outcomes. Figure 1 illustrates a potential automatic irrigation system. Below, you'll discover a fundamental description of an automatic irrigation system, along with the associated signals intended for analysis in this project.

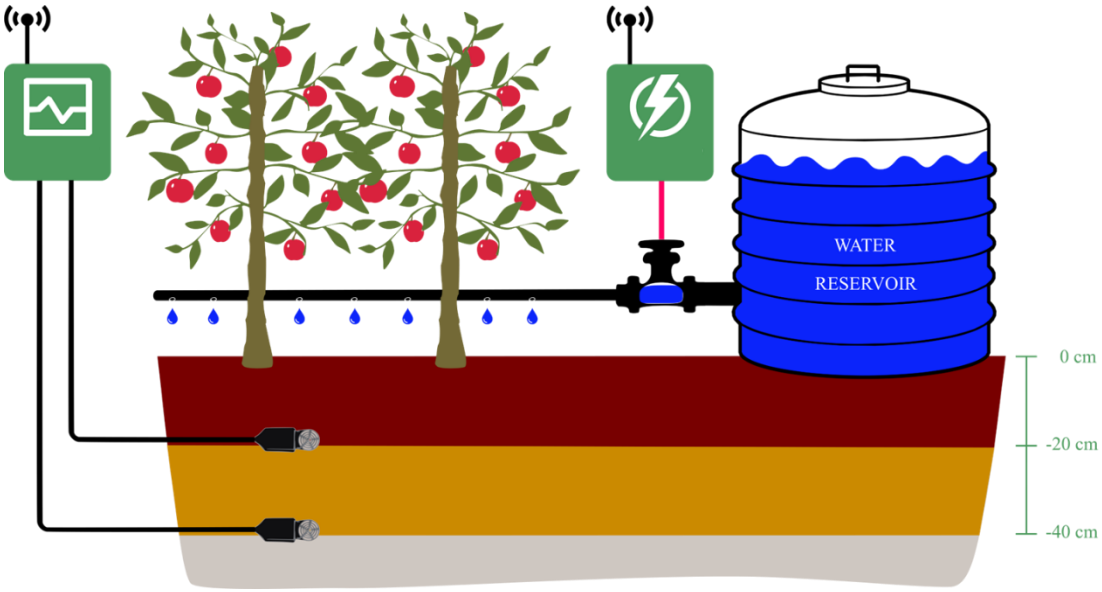
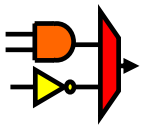


Figure 1 - Simple automatic irrigation system

The irrigation system comprises sensors embedded at varying soil depths, a reading unit, and an actuation unit directly linked to the valve controlling water flow in the drip irrigation setup. These sensors gauge soil water content, enabling the system to analyze data and determine irrigation necessity. Upon detecting potential water stress, the system triggers the irrigation process by sending a command to the valve board. Once adequate water is dispensed, the valve closes.

These sensors measure soil matric potential, indicative of the negative pressure roots exert to draw water from the soil. Each sensor yields a negative value, expressed in kPa, where zero denotes saturated soil, and more negative values indicate water scarcity. The data are represented as 16-bit signed values. Two sensors are deployed at depths of -20 and -40 cm, ensuring comprehensive monitoring of the plant root zone. The system alternates between readings from the -20cm and -40cm sensors.



Traditionally, irrigation systems function as Decision Support Systems (DSS), offering predictions to farmers and suggesting irrigation strategies. In this instance, a closed-loop automatic system directly regulates the irrigation valve. Identifying the stress condition prompting valve activation is crucial. Here, the average value of the two sensors is used as the stress detection parameter. If this average remains below the stress threshold for 10 consecutive samples, irrigation is initiated, and the IRRIGATIONALARM signal is set to "high" to commence the process.

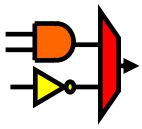
The controller system developed in this laboratory represents the detection component of the automatic irrigation system. It retrieves data from sensors stored in MEM_A, which consists of 1024 8-bit memory locations. As the data are expressed in 16 bits, they need to be stored in two successive memory addresses using the LITTLE ENDIAN order (with the most significant byte placed second in memory). Additionally, data from the same time sample are stored in consecutive memory addresses, with readings from the -20cm sensor taking precedence, as depicted in the table below:

Address 0	Sample 0 @ -20cm
Address 1	
Address 2	Sample 0 @ -40cm
Address 3	
Address 4	Sample 1 @ -20cm
Address 5	
Address 6	Sample 1 @ -40cm
Address 7	
…….	…….
Address 1022	Sample 255 @ -40cm
Address 1023	

After retrieving data from the memory, the system must calculate the AVERAGE of the data from the two sensors of the same sample, as well as determine the HIGHEST value between them expressed as an absolute value (keeping in mind that the data are always equal to or lower than 0, therefore the **most negative one**). These two values are then written into another memory, MEM_B, which shares the same dimensions as MEM_A. Specifically, MEM_B follows the same bit order (LITTLE ENDIAN), so the first average value starts from location 0, and the first highest value starts from location 2. The structure of MEM_B is outlined in the table below:

Address 0	Sample 0 AVERAGE
Address 1	
Address 2	Sample 0 HIGHEST
Address 3	
Address 4	Sample 1 AVERAGE
Address 5	
Address 6	Sample 1 HIGHEST
Address 7	
…….	…….
Address 1022	Sample 255 HIGHEST
Address 1023	

Additionally, the water stress threshold, stored in a dedicated read-only register before commencing all operations, must be compared with the average of each sample. If 10 successive samples are consistently below the threshold, the system must set the IRRIGATIONALARM output to ‘1’. This signal must remain high until a new acquisition is initiated, regardless of whether the sample average exceeds the threshold before the procedure concludes.



The circuit features an input signal, denoted as START. When the circuit detects START at '1', processing initiates, beginning from the subsequent clock cycle. During this phase, the circuit stores matric potential samples from input DATA_IN into MEM_A. It's important to note that DATA_IN is transmitted in 8 bits, with the least significant bits transmitted first.

The writing operation is synchronized with the positive edge of the clock signal CLK, with one datum per clock cycle being written to the memory. Samples are stored sequentially, starting from location 0 and ending at location 1023.

The memory is implemented as a *register file* with synchronous writing and asynchronous reading operation:

- if WR='0' on the rising edge of CLK, and the memory is selected, i.e. CS='1', then the datum available at DATA_IN is written at the location pointed by ADDRESS
- if CS='1' and RD='1' the DATA_OUT output is combinational, and it is the value stored at the location pointer by ADDRESS.

Once all the data are stored in MEM_A, the circuit must start the analysis filling MEM_B and comparing the average with the stress threshold.

The algorithm must rely on a **multiplierless datapath made of one adder** for the computation of the average and the comparisons. It means that the multiplications must be performed only using add/sub and shift operations (not too complex, since all the constants are combinations of powers of 2) and the same adder must be employed for computing the sum/subtraction of the contributions. The parallelism of the datapath has to be sized to compute the values **without overflow**.

Once the analysis is completed, the circuit drives to '1' the output DONE. Pay attention to the fact that this signal is asserted when MEM_B is filled, so it is not required to transmit the results on the DATA_OUT bus connected to MEM_B.
Finally, the circuit waits for a new START (0 → 1) before starting again the algorithm.
For the circuit described above you must perform the following steps (that must be inserted in the final report you have to give the teachers!):

1. **Write** the pseudocode of the algorithm.
2. **Draw** the datapath of the algorithm with all the controls, status, connections, parallelism of the busses, etc...
3. **Prepare** the ASM chart of the algorithm.
4. **Detail** the ASM chart of the control unit.
5. **Write** the timing of the circuit both during the load of the samples and data analysis. **Note:** it is important to show only a meaningful fragment of the timing showing the evolution of the status register, the data and the control signals. This written by-hand timing will be compared with the simulated circuit timing to be sure that it works according to your requirements.
6. **Write the VHDL** of the whole system including the memories.
7. **Prepare a testbench and simulate** the behavior of your circuit.
8. **Write a final report** including all the materials requested in the previous items, plus your considerations about the design choices you made, the testing algorithm you used, the by-hand timing you prepared during the design, some Modelsim simulations fragments to demonstrate that it works.

Remember that the evaluation of this lab will be made uniquely on the report you write, **so write a good report!!!!!**

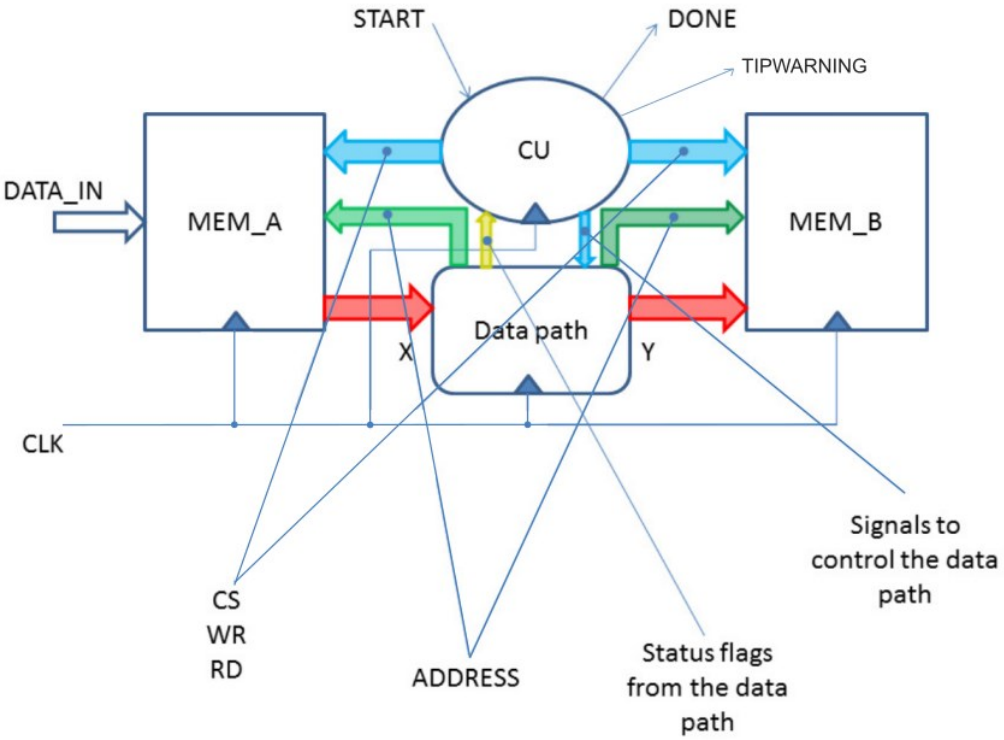


Figure 2 - simple filter architecture

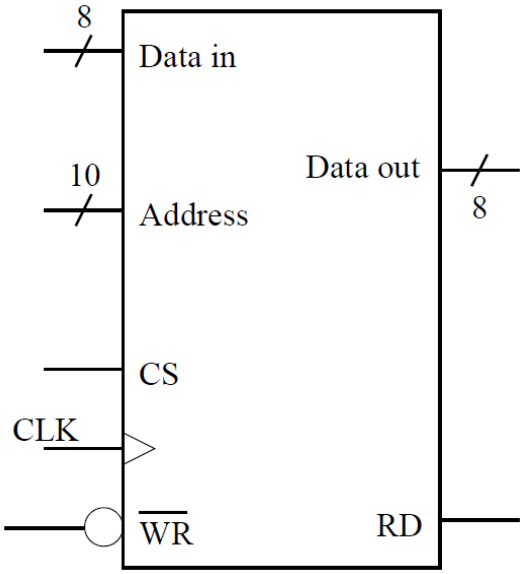


Figure 3 - Memory pinout