# Specifications and Simulations Of Digital Systems

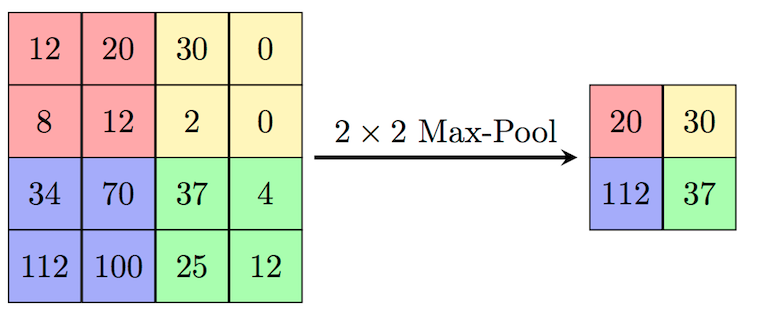
## Assignment Report: Dalmasso Luca s281316

## INTRODUCTION: Max Pooling algorithm.

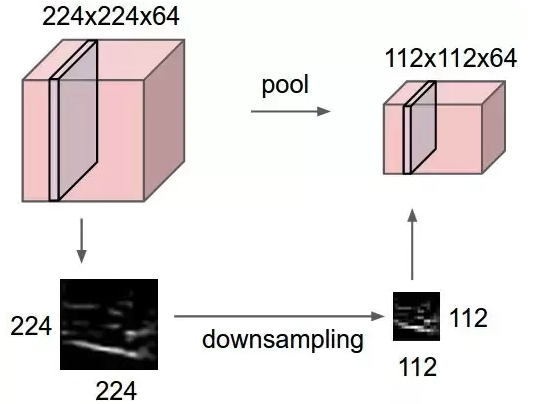
The goal of this assignment was to design a VHDL module capable of doing a Max-Pooling algorithm.

Max pooling is a sample-based discretization process. The objective is to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions.

Graphical representation of the algorithm on two dimensions:



Real life example: image



## Algorithm in C:

## This piece of code is executed with the purpose to generate a memory file used by the testbench in order to test the HLSM output stream.

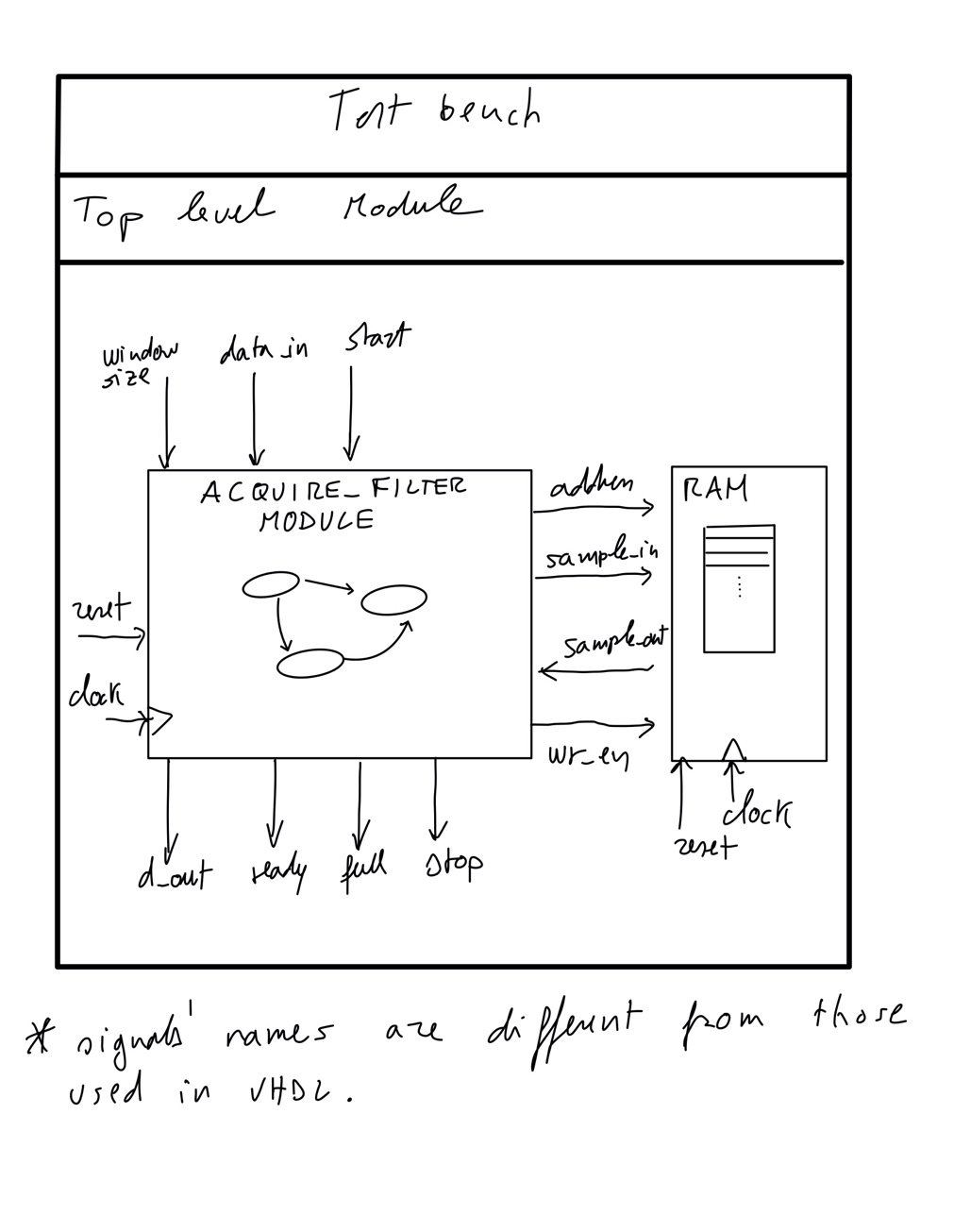
The Max-pooling algorithm is performed by 4 nested loops that basically are looking for the maximum value inside every MxM non overlapped submatrices of the source NxN matrix, where M is the pooling-window’s width.



## ARCHITECTURE

I designed 3 main components in order to compute the algorithm.

Acquisition system, Memory, Top-Level module



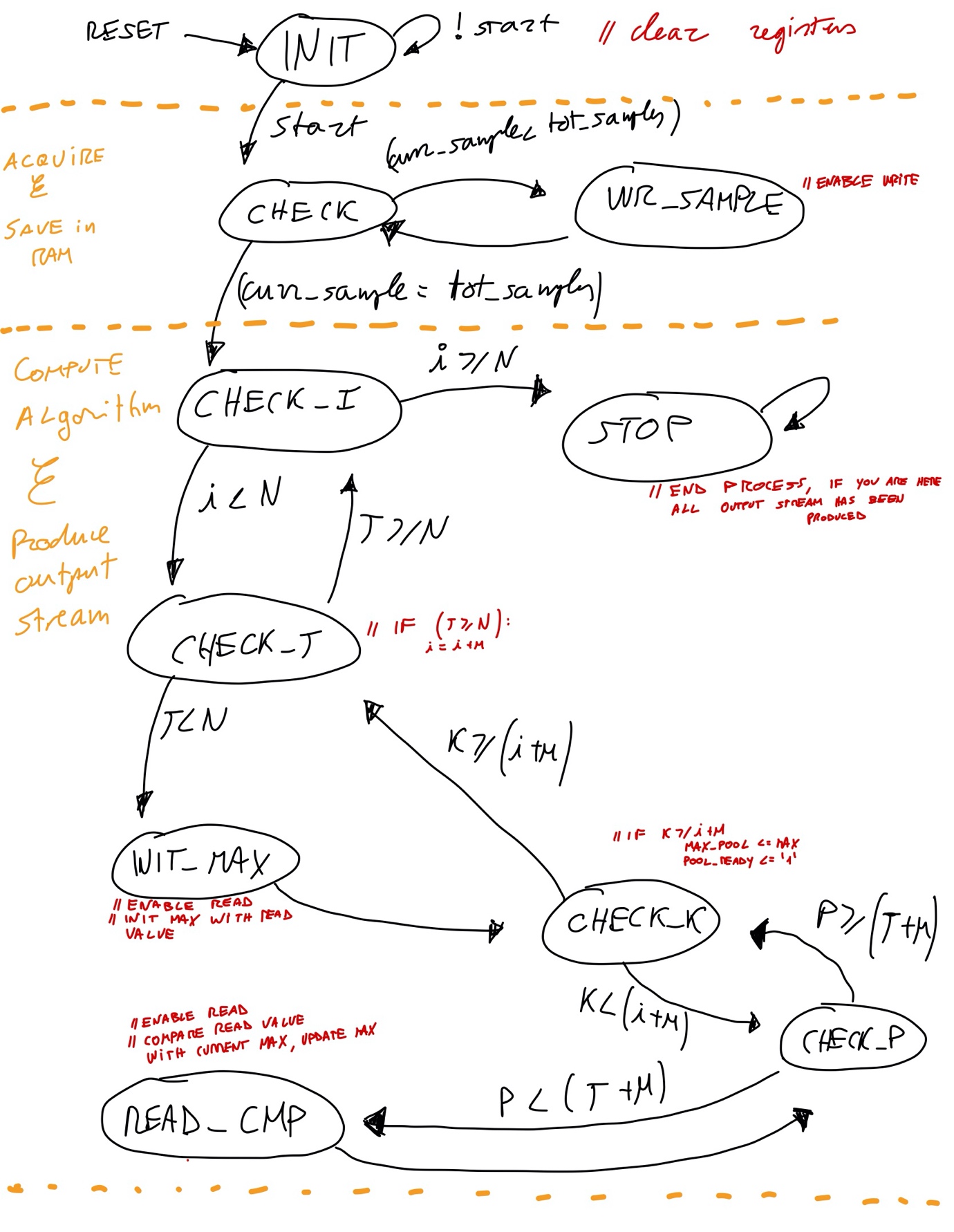
1. **Acquisition system**:

This module is able to capture a stream of input samples, coming from a Top-Level module, and send them to external memory (RAM type).

Once all samples have been saved, the Max Pooling algorithm is performed, and a new output data stream, containing sub-region max, is generated.

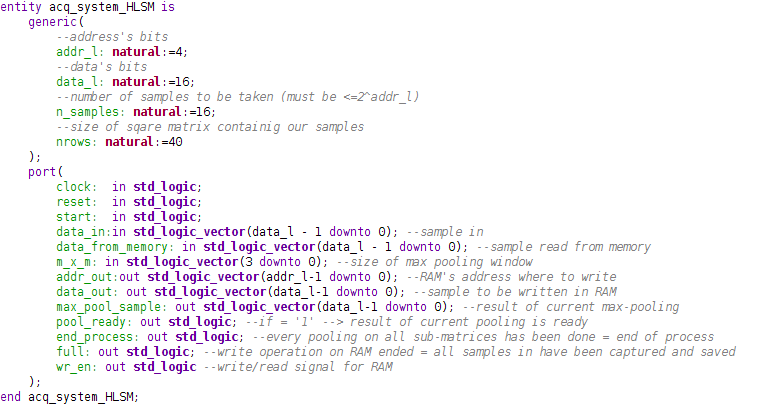
Once every sub-region max has been computed and streamed, this module goes in an idle state where waits for an external reset;

This module is designed to be a HLSM, the following schema is showing the HLSM states that I designed:

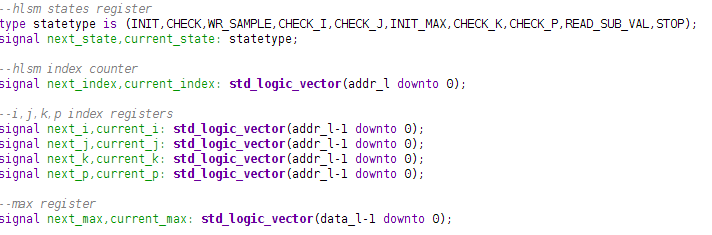


The following is the VHDL code describing the entity acquisition system and the HLSM:

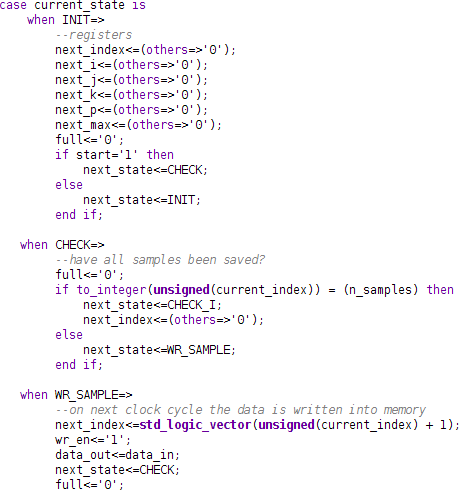
Entity:



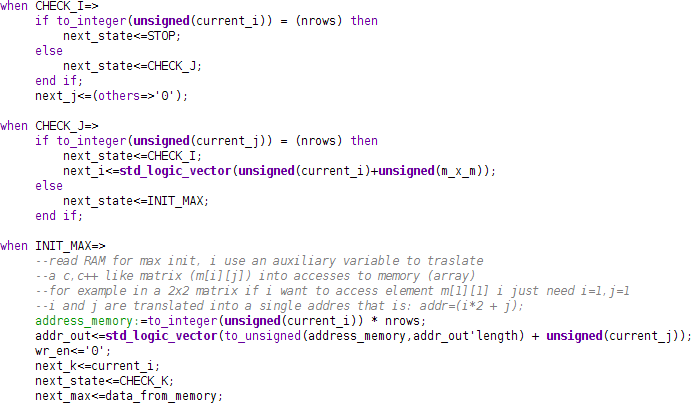
HLSM internal registers and signals:



Sample and acquire stages:



Max-Pooling algorithm: subregions, max initialization.



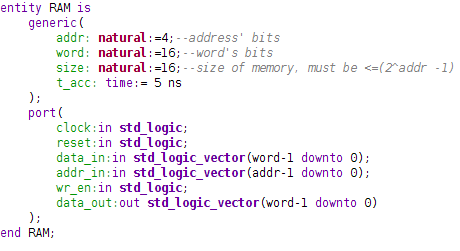
Max-Pooling algorithm: max research.



1. **RAM storage data type**:

Due to the fact that I have at first a ‘sample’ phase, and then a ‘computation’ stage, I need to memorize all the input stream in a storage.

Entity:



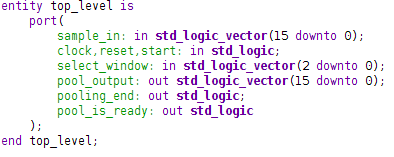
Read & Write processes:

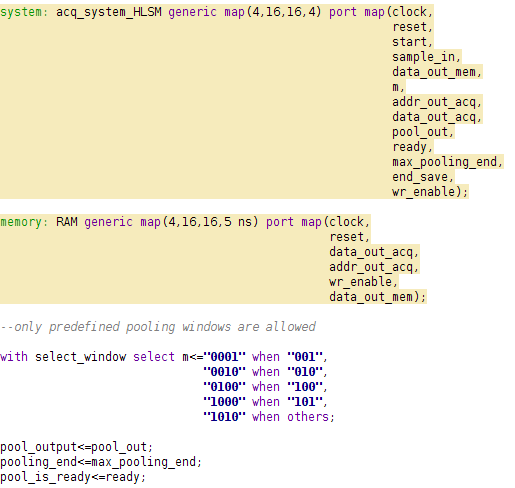


1. **Top-Level:**

VHDL module that includes in a structural way the two previous modules.

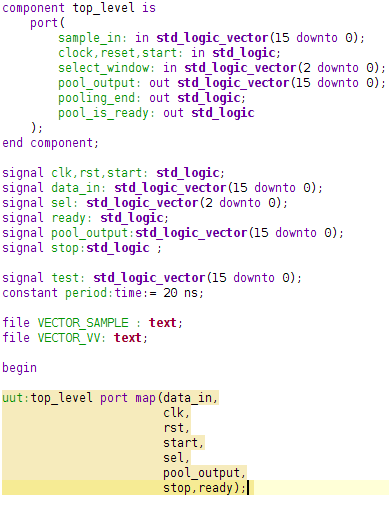
This module is only one accessed externally, in fact is the Unit Under Test in the TestBench.





## TESTBENCH, test of the proposed solution, waveforms analisys

Unit Under Test: Top-Level module:

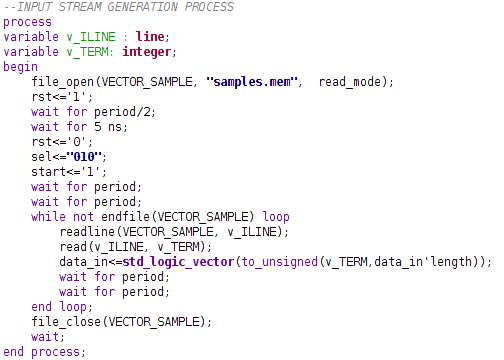


Action Performed:

* Generation of the input data stream, selection of the pooling window:

The input data stream is generated by reading a memory file called ‘samples.m’ , previously created by a C piece of code that randomly generates NxN values and writes them the file.

This is the testbench process that perform this operation:



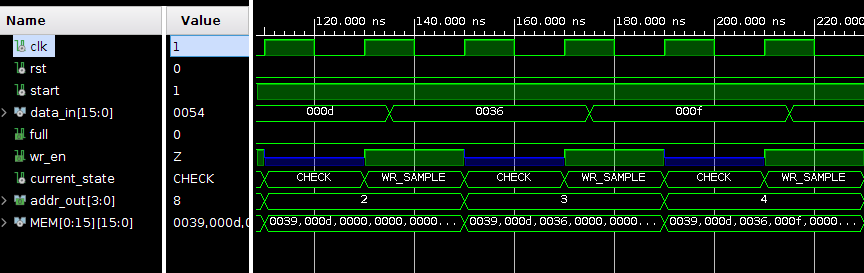
Note:

During the input stage the HLSM is locked in CHECK and WR\_SAMPLE stages so we need to wait for 2 clock cycles between an input and another.

Folliwing image shows a waveforms capture of some important signals, such as:

Data\_in: input sample coming from testbench

Wr\_enable: HLSM output control signal for the RAM



* Test of the results

Verification and validation of correctness of the results is performed by a process that save output data stream generated by HLSM (every data is the current max of a subregion) and compare it with a correct result saved into another memory file.

Memory file is written by the C function discussed previously.

Testbench’s test process that perform this operation:



Waveforms capturing pool\_output coming from HLSM:

Important signals:

Stop: when = ‘1’, the Max-Pooling on all matrix is computed, the HLSM is idle (goes in STOP state, after going through CHECK\_I for the last time) until a reset signal.

The TestBench’s test process now compares all results, saved in an array variable, with pooling.mem files containing correct results.

Ready: when=’1’ the HLSM has the current max-pool of a subregion

