**ELEC6234 – SystemVerilog Design of an Embedded Processor**

Max Fuller

mf5g14

MEng Electronic Engineering

Dr Alex Weddell

**ABSTRACT:** *Su*

1. **Introduction**

The objective of this piece of work was to design, model and test a synthesisable implementation of a picoMIPS-style processor. The design must be capable of performing an affine transform on a manually-entered set of co-ordinates and produce the correct output. It was required to design it in such a way as to use as few hardware resources as possible whilst still performing as a processor, as opposed to a block of dedicated hardware. This meant that there should be two separate discernible sections – a control path and a data path, with the affine transform implemented using generic instructions stored in program memory.

An initial overall system design was created before any HDL coding took place. This made it clear what components were required, how they would fit together and where there might be opportunities for space optimisation later in development. The design was started at the top-level (namely *picoMIPS*), with all the lower-level modules then being added before deciding on the data and control wire connections.

Once the design was complete, development of each module began using SystemVerilog. This was done from the bottom up - starting with the simplest of modules like the program counter. A testbench was created for each module so they could be simulated using ModelSim to test their functionality before moving on to the next one.

Unit testing meant that, once wired up, the overall system worked on the first attempt. Similarly, synthesising the design and running it on a DE1-SoC board also worked the first time around.

Several optimisations were done to the original design to reduce the hardware resources further, as was suggested in the specification:

You may design your own instruction set and modify the instruction format in any way you wish. You may also modify the architecture if it helps to reduce the cost figure.

These included a minimisation of the instruction format and removal of branching in the program counter. The final design correctly runs the affine transform and is still a processor.

1. **Instruction format, decoder design, program memory and program counter**

*Pr*

1. **General Purpose Register file design, simulation and synthesis**

*As*

1. **Arithmetic Logic Unit and Multiplier design**

*Ex*

1. **Altera DE0 implementation**

*Ex*

1. **Conclusion**

*St*

1. **References**

*Q*