# ELEC6025: VLSI Design Project
# Part 1: Microprocessor Research
# Topic: Instruction Set

Martin Wearn
Team: R4
Course Tutor: Mr B. Iain McNally

February 12, 2014

# Contents

# 1 Introduction

In programming a particular microprocessor, the size and formatting of assembly language instructions used are defined by the Instruction Set Architecture. This is a description of what operations can be performed, and how to carry them out. When choosing the encoding of this architecture, a number of factors need to be considered. Including, but not limited to, instruction size, completeness, regularity and streamlining. This report focusses on the principles of orthogonality and code density.

# 2 Instruction Set Considerations

## 2.1 Orthogonality

Each instruction within an architecture is defined as a series of binary bits defining the operation to be performed, the data to perform it on, and a number of instruction-specific parameters. The concept of orthogonality is based on separating these bits into multiple fields with each defining difference aspects of the instruction. The value of each field is then able to vary independently of each other part of the instruction. [**?** ]

Using the formatting defined as in table 1, any instruction can be entered into the 'Opcode' field with any choice of addressing mode, source or destination. If there is no dependency between different fields the set is completely orthogonal. Example architectures include: ARM11 and VAX-11. [**?** ] If there is some dependency, such as not having all addressing modes available for all available instructions, the instruction set can be perceived as near-orthogonal, such as in the DEC PDP-11 or Motorola 68k. [**?** ]

| Opcode | Addressing Mode | Source 1 | Source 2 | Destination |
|---|---|---|---|---|

Table 1: Example Instruction Set Format

## 2.2 Code Density

The principle of code density relates to how much can be performed using a fixed amount of instructions or available memory to store the instructions. In performing a given task, using one line of assembly code which translates to a small number of bytes in machine code will have a high density. Comparatively, the same task may require multiple lines of assembly which

translate to the same or more bytes of machine code per line on a different architecture. This will result in a much lower code density.

Instruction set architectures designed without emphasis on simple operations are known as Complex Instruction Set Computers (CISC). These typically have a very large set of instructions and can range from a simple arithmetic ADD to a REPE instruction as found on an Intel 8086. [**?** ] This repeats the following instruction while the ZF flag equals 1, up to a given maximum number of times.

#### 2.2.1 Kolmogorov Complexity

# 3 Case Studies

information

## 3.1 VAX-11

## 3.2 DEC Alpha

## 3.3 Intel 8086

# 4 Conclusion

Here I would like you to discuss how the issues raised in your report will affect your processor design. Where you have seen different processors opting for different solutions to the same problem you should discuss their relative merits in the context of your design.

Word Count:

# References

[1] Calcolatori Elettronici. Complete 8086 instruction set. http://www.gabrielececchetti.it/Teaching/CalcolatoriElettronici/Docs/i8086$_i nstruction_s et.p$

[2] Wikipedia. Orthogonal instruction set. http://en.wikipedia.org/wiki/Orthogonal$_i nstruction_s et.lastviewedon$11/01/2014.

# Bibliography

[1] Leslie Lamport, *LaTeX: A Document Preparation System*. Addison Wesley, Massachusetts, 2nd Edition, 1994.