

ELB816 Development Environment - Interim Report

James Bowden - 110104485

Project Supervisor: Christopher Harte

Contents

- **Introduction**
- **Assembler**
- **Emulator**
- **Debugger**
- **References**

Introduction

Due to a personal injury and its impact on this project and other academic obligations, not as much progress has been made by this point as originally expected, and development is lagging behind the time-line outlined in the original specification.

However there have been some advances.

- A preliminary version of the assembler has been developed. Although it is unfinished in terms of supporting the entire language described in the ELB816 specification, it can produce machine code for the majority of operations in the instruction set and will be useful for assembling small programs used for testing during the development of the emulator.
- The emulator is in very early stages of development
- The debugger is being designed along side development of the emulator.

Assembler

Project Files:

- assembler/assembler.py - first and second pass functions
- assembler/language.py - language definition

A two pass assembler for the ELB816 assembly language is being written in Python 2.7 and tested on the latest version Debian Linux,

It's designed to support v0.9 of the instruction set ^[1] and subsequently produces machine code for every mnemonic and argument format in the referenced file.

Support for the following data types have not get been added:

- addr11 - 11bit integer used for paged addressing
- port_addr - port addressing
- vect8 - interrupt addressing

The design and structure of language.py makes adding these, or other extensions to the language only slightly more than trivial.

The preprocessor currently only supports the 'label:' directive. Extending the language with other pre-processes such as 'org', '#include', 'EQU' etc will require further design and development of assembler.py's first_pass() function

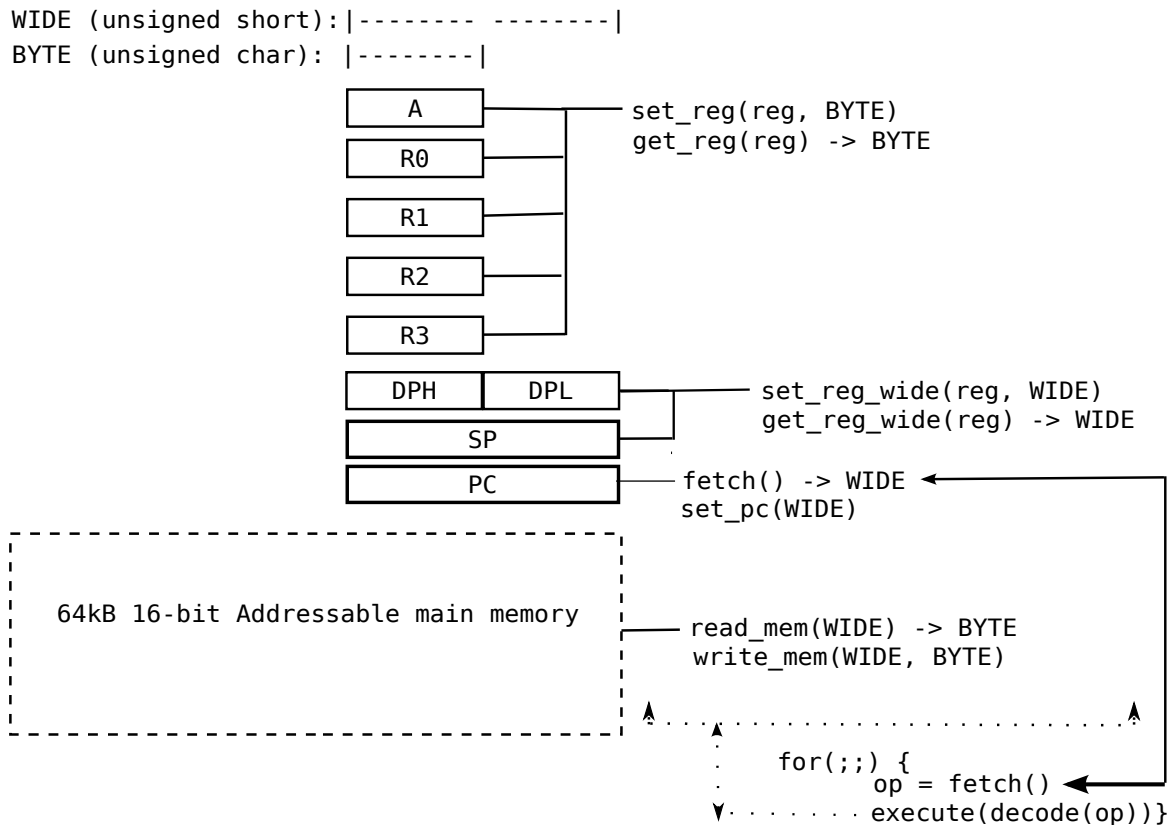
Emulator

Project Files:

- emu.c – fetch, decode, execute cycle
- emu.h – data types, register, memory and function definitions
- mem.c – memory access functions

The emulator is being designed, developed and tested using GCC on the latest Debian Linux.

The immediate goal is to create a system like the one below:



The design of the `decode()` and `execute()` functions is still in never early stages.

Debugger

At this stage, the debugging interface is nothing more that using STDIN/OUT to control the process.

References

1. https://code.google.com/p/elb816/source/browse/trunk/doc/general/elb816_opcodes.ods