Assignment 2: The Making Of An IDIOT Implementor's Notes

Dylan Wright
dylan.wright@uky.edu
Casey O'Kane
casey.okane@uky.edu

***Abstract*—Goal of this assignment involved the implementation of the IDIOT instruction set using the AIK assembler, the Verilog Hardware Design Language and detailed test plan to exhaustively test the different components and logic of the design.**

I. Testing

A. Instruction Set Architecture

In order to test the IDIOT instruction set specification a test framework was implemented. This framework is in the `IDIOT/` directory. The framework consists of the folowing files:

1) `aik.py`: To automatically test files `aik.py` sends a PUSH request to the AIK cgi program. The returned html page is parsed and each section is output. The `.text` and `.data` sections are sent to stdout and the assembler messages are sent to `stderr`. This method is not ideal, an AIK executable would be prefferable. Sample run:

```
$ echo "file.idiot" | ./aik.py
```

2) `diss.py`: To make test results human readable, `diss.py` dissassembles a `.out` file (the `.text` and `.data` segment of the output of `aik.py`). The code is converted to binary and displayed in tabular format. Sample run:

```
$ echo "file.out" | ./diss.py
```

3) `test.sh`: This file can be used to test each `.idiot` file in the `progs/` directory. This script runs each file through AIK and compares the output to the expected output. `.text` and `.data` segment expected output should be placed in a file with the same name as the program and a `.expected.out` file extension. Expected assembler messages should be placed in a file with a `.expected.err` extension. The test script will report the number of passed, failed and possibly failed tests. This test framework was adapted from a script provided by Dr. Jaromczyk in the Fall 2015 CS 441G: Compilers course. Sample run:

```
$ ./test
```

B. Verilog Modules

II. General Approach

III. Issues