EE480 Assignment 1: KySMet Encoding And Assembler

Implementor's Notes

Hank Dietz
Department of Electrical and Computer Engineering
University of Kentucky, Lexington, KY USA
hankd@engr.uky.edu

ABSTRACT

This project involved the design of an encoding scheme for the KySMet instruction set. The encoding was then to be embodied in an AIK specification.

1. GENERAL APPROACH

The first issue is how to encode 25 instruction types in a 16-bit instruction words where 12 bits are needed for the operands of some instructions. The remaining 4 bits can only distinguish 16 instruction types. Complicating matters a bit, I also wanted the encoding of similar operations to have more instruction bits in common. I further imposed the constraint that I wanted the `trap` instruction to encode as 0, so that any instruction sequence wandering into empty memory will cause a trap to the operating system.

I decomposed the instruction set into groups. The first group contains instructions that have no arguments. The second group is control flow instructions that take a 16-bit address. The next group take two register arguments. The three-register instructions are the group that uses most of the 16 possible opcode field values. The 8-bit load immediates form the last group. It's all very straightforward.

Although this instruction set is intended for a SIMD-parallel implementation, that really doesn't impact the instruction encoding. The only extra concern was that KySMet is a Harvard architecture, so there are separate `.text` and `.data` segments, both appearing to start at location 0.

2. THE LI MACRO

As required, there is an `li` macro operation implemented to use a `li8` alone when possible and to follow it with a `lu8` only when the immediate value made it necessary. This is implemented by having two patterns for `li $.d,.i16`. The first includes a condition that will fail if `.i16` doesn't match the sign-extension of the bottom 8 bit of `.i16` (i.e., `.i16 & 255`). The second pattern simply uses the pair `li8`, `lu8`.

3. CONSTANTS

Not much to report here; all constants were given the values that correspond to the order they were listed in the assignment. However, the HTML document specifying KySMet was actually self-inconsistent in that it swapped `IPROC` and `NPROC` in the table describing the values. My design assumes `IPROC` is 1 and `NPROC` is 2.

4. ISSUES

Really nothing terrible. Note that I was careful to avoid potential name conflicts by using names that started with "." in my specification.

The hardest thing to do is to test this. It really isn't testable in any obvious way... so this was tested only by manual inspection with a variety of simple test cases, the largest of which is included with my assignment. There were no errors flagged by AIK.

ACM ISBN N.A..
DOI: N.A.