Joseph George

CMSC 301

Program 1

**Design Document**

This program provides means to, when provided with a file containing the binary encoding of MIPS instructions, translate these binary encoding line by line to create their corresponding decoded MIPS instructions. The basic algorithm can be described through the following process: First, acquire a file containing the binary encoding of MIPS instructions, and after checking the file is the correct format, parse the binary encoding according to how MIPS instruction are stored. After this task is completed, the instructions are categorized into their name based on the opcode field and function field (i.e. R-Type, J-Type, I-Type). Then, based on this type and the way that the positions are stored in the OpcodeTable class, each instruction is created by knowledge the instruction names, and any associated registers and/or immediate fields and memory addresses used for jump type instructions. These are then appropriately printed out.

This program consists of the following classes, which when combined, yield a result using the general algorithm described above: BIN.cpp, BINParser.cpp, RegisterTable.cpp, Instruction.cpp, and Opcode.cpp. BIN.cpp is where the program starts out at – it is responsible for reading in an appropriate file and then passing this file into BINParser.cpp. BINParser.cpp performs the core actions to decode the binary encoding – it parses the file, storing individual components into a data structure, and then using these parsed components of the binary encoding, stores the fields into instructions given an opcode and the string representations of the operands which were previously parsed. When all of the operands are acquired, individual instructions are then printed out according to their opcodes and positions of RS, RT, RD registers, and shift amounts and immediate fields etc. Additionally, this class is responsible for appropriate conversion of two's complement binary representation, as well as decimal to hexadecimal representation.

The rest of the classes are essential to the functioning of BINParser.cpp. RegisterTable.cpp, which stores information about registers, gives the appropriate register when passed a 5bit binary encoding. Instruction.cpp is used to represent the actual MIPS instructions by setting the appropriate components of the instruction. This is used to set the values of the instructions inside of BINParser.cpp. Opcode.cpp includes information about all of the supported MIPS instructions, including information about the opcode, expected operands, and other fields. Additionally, this class contains methods which allow access to the data fields of the instructions. This is essential to the functioning of BINParser.cpp, which calls certain methods inside of Opcode.cpp to acquire all of the information about a specific instruction.

In order to add a new MIPS instruction, appropriate information about an instruction would have to be added to the Opcode.cpp class. Information would have to include knowledge of things like the RD, RS, and RT positions, the opcode, the function field, immediate label, etc. Once all of this information is added, the rest of the program should still function optimally, making the adding of a new instruction straightforward given that the user knows information about the instruction.