Saifil Ali - 323008228

Chris Folleras - 422007002

Jonathan Wang - 223009591

Dr. Mahapatra, TA: Richa Surbhi

CSCE 312

**Instruction Set Architecture and Minimal Processor Design**

1. The instruction set architecture for our 8 bit load-store processor included the following commands (basic syntax included to the right of each command):

**HALT** - *Immediately stops the program*

**LOAD** [first register] [address] – *Writes value of the memory at [first register] to the corresponding [address]*

**STORE** [first register] [address] – *Writes value of [first register] to [address]*

**ADD** [first register] [second register] – *Adds the values of [first register] and [second register]; store result in [second register]*

Further, we decided to encode each operation where **HALT** is represented by 11, **LOAD** by 00, **STORE** by 01, and **ADD** by 10. Therefore, each 8-bit instruction follows the same structure: *[command] [1stRegister] [2ndRegister] [Address] 🡪 2bit 1bit 1bit 4bit.* This implementation allowed for a simpler circuit design, as the address does not immediately follow the first register, which results in the value for the second register to be arbitrary (null).

1. We decided to implement a number of modules to facilitate the production of memory address space suitable to run our planned program. First, in order to handle writing to and reading form RAM, our group created a *Bidirectional Bus* – off (01) when writing and on when reading. Next, we designed a register file which could support the basic functionalities of each register: namely reading and writing. Lastly, we used full adders in the design of the 5-bit incrementor which stepped through instructions and, obviously, the 8-bit adder that actually added each inputted number together.
2. We would procure a standard P-ROM (programmable read-only memory) chip in order to program the code we wrote into a physical ROM module. This type of chip is programmed by physically burning fuses – where an open fuse is interpreted as a 1 and a burned fuse, 0. Also, it’s important to note that this process is irreversible, meaning this chip is only able to be programmed one time. The binary sequence that results from this precise burning of fuses in this chip would be our program.