Project Report

CSCE 2303

Dr. Mohamed Shalan

Raafat El Saeed - 900191080
Omar Ibrahim - 900192095
Kirolous Fouty - 900212444
Yehia Ragab -
Ziyad Amin - 900201190

Program Guide (How to use)

1. Compile the program
2. Run the program through the cmd line while providing the binary files you would like to dis-assemble and, optionally, a data file you'd like to include.

   E.g. enter "rvism.exe t0.bin t0-d.bin" in the proper directory in a command line window.
3. The program will start to dis-assemble the given binary files.

Simulator Design

1. First, the program examines the given instructions to determine whether it is in a compressed format or an uncompressed format format.
2. If the instruction is compressed, the program proceeds to decompress it using a specialized decompressor. The decompressor transforms the compressed instruction into its uncompressed counterpart.
3. After the decompression step, the program passes the uncompressed instruction to a translator. The translator is responsible for producing the appropriate assembly code representation of the instruction.
4. If the instruction is not compressed, the program skips the decompression step and immediately proceeds to the translator to generate the assembly code representation.
5. Once the translator has produced the assembly code representation of the instruction, the program executes the resulting code.

Simulator Limitations

- The simulator does not support floating-point integers, limiting its usefulness for applications that require high-precision calculations.
- Multiplication instructions are not supported by the simulator, which may restrict its effectiveness for programs that rely heavily on multiplication operations.
- The simulator does not support all compressed functions, which may limit its ability to simulate certain programs effectively.

Challenges Faced

The team encountered several challenges while working on the program.

- Extracting immediate values from instruction code proved to be difficult, particularly for instructions that were not properly indexed from left to right as MSB to LSB. This led to additional efforts and time being expended in order to extract the necessary information for the program.
- Debugging the program was a major challenge, as tracing the assembly code required manually writing the instruction codes on paper and translating the instructions that were not working properly while tracing. This process was time-consuming and required a high level of attention to detail, which made debugging a slow and arduous process.
- Re-writing and translating compressed instructions into their decompressed alternatives was a difficult task, particularly for instructions that did not have similar 32 and 16 bit formats. This required significant effort in order to ensure that the instructions were properly translated and that the program could run effectively.
- Lack of resources online regarding compressed instructions, which made it difficult to find solutions to certain issues that arose during development.
- Creating branches on GitHub to allow multiple team members to work simultaneously was not very applicable due to the complexity of the program and the need for constant communication and collaboration. This made it difficult to manage the codebase effectively and led to some confusion and delays in development.