Assembler-Linker-Loader

IITG CS 244—System Programming Lab
Instructor: Prof.Santosh Biswas, Prof.Arnab Sarkar

By:
Dheeraj Khatri: 120101021
Dhruv Kohli: 120123054
Rahul Kadyan: 120122028

Content

Introduction

- An assembly language is a low-level programming language for a computer, or other programmable device, in which there is a very strong (generally one-to-one) correspondence between the language and the architecture's machine code instructions.
- Assembly language is converted into executable machine code by a utility program referred to as an *ASSEMBLER*; the conversion process is referred to as *assembly*, or *assembling* the code.
- One or more object files generated by a compiler are combined into a single executable program by a LINKER.
- LOADER places programs into memory and prepares them for execution.

Assemble Pass 1

- Any symbol used before it is defined will require "errata" at the end of the object code telling the linker or the loader to "go back" and overwrite a placeholder which had been left where the as yet undefined symbol was used.
- Tasks Performed:
  - Separate the symbol, mnemonic opcode and operand fields.
  - Build the symbol table.
  - Perform LC processing.
  - Construct intermediate representation.

Point to the Corresponding Code.

Assemble Pass 2

- Second pass creates a table with all symbols and their values in the first passes, then use the table in later passes to generate code.

- Tasks Performed:
    - Synthesize the target program.

Point to the Corresponding Code.

Link

- Typically, an object file can contain three kinds of symbols:
    - defined symbols, which allow it to be called by other modules,
    - undefined symbols, which call the other modules,
    - local symbols, used internally within the object file to facilitate relocation.
- For most compilers, each object file is the result of compiling one input source code file.
- When a program comprises multiple object files, the linker combines these files into a unified executable program, resolving the symbols as it goes along.

Point to the Corresponding Code.

Load

- The loader's tasks include:
  - validation (permissions, memory requirements etc.);
  - copying the program image from the disk into main memory;
  - initializing registers (e.g., the stack pointer);
  - jumping to the program entry point (_start).

Point to the Corresponding Code.

Output Interpretation

- Characteristics of final output generated:
  - Executable on GNUSIM 8085.
  - Program loaded at the user defined location.
  - Corresponding location need to be used at the time of execution.

Point to the Corresponding Code.

References

- Assemblers Handout provided.
- http://en.wikipedia.org/wiki/Assembly_language
- http://en.wikipedia.org/wiki/Assembly_language#Assembler
- http://en.wikipedia.org/wiki/Loader_(computing)
- http://en.wikipedia.org/wiki/Linker_(computing)