Assembly Relocation Table

Asm – Workgroup: 0x10c Standards Committee
RFC X1001

James Rhodes

April 11, 2012

This draft provides a formal structure for providing an assembly relocation table from within DCPU-16 programs.

1 Introduction

As it stands, code generated by assemblers is either not relocatable, or the relocation format is not standardized. Thus this document suggests a standard mechanism for providing a table of addresses that need relocating.

1.1 Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119

2 Role of Assemblers

For the purposes of this draft, the role of assemblers is to generate code from a defined syntax to DCPU-16 bytecode.

In this case, assemblers SHOULD provide an option to generate relocatable code, but MUST NOT generate relocatable code unless the user indicates that they wish to do so.

3 Relocation Table Format

For purposes of future versioning, this document specifies version 1 of the relocation table format.

The format of the relocation table is as follows:

Contents of single Word
Magic number
Version number
Size of table
Entry 1
...
Entry N

Magic number The magic number consists of the string "RT" (stands for "Relocation Table") as a packed 7-bit ASCII literal, thus resulting in the value 0x5254.

Version number The current version number is 0x0001.

Size of table This field must be set to the total size of the table, in words. This includes the magic number, the version number and this field, too.

Entries Each entry is a absolute address pointing to a single word in the file the relocation table is contained in, which should be relocated relatively to the position the code is loaded to.

4 Relocation Table Positioning

The assembly relocation table must be positioned inside the generated code, but have no effect on the program execution.

When an assembler generates relocatable code, the first instruction MUST be a jump to the start of the actual program code. This results in the first two words being:

Contents of single Word
SET PC, <next word literally>
Location of first program instruction

It is important to note that assemblers will have to offset all label addresses by the size of the relocation table, plus the two words at the start.

5 Security Considerations

It is potentially possible for a malicious user to generate code which determines the offset of the resulting relocatable program when it is loaded into memory and executed.

This is possible by creating a label with a predetermined address if the program was running at 0x0, and calculating the difference between the actual address that the program would jump to and the original value.