Improving Debugging For Optimized Rust Code

Master Thesis

Niklas Lundberg

Department of Computer Science, Electrical and Space Engineering
Luleå University of Technology

September 14, 2021

Overview

1. Introduction
2. DWARF
3. Implementation
4. Evaluation and Disscussion
5. Conclution

What is debugging

- The process of finding an resolving errors, flaws, or faults.
- Debugging techniques:
  - Back tracking
  - Testing
  - Control flow analysis
  - And many more
- Debugging is very useful for embedded systems.

What is a Debugger

- A Debugger is a debugging tool.
- Control over the debugged computer program.
- Some of the most common control features:
  - Continue/Start/Run
  - Stop/Halt
  - Restart
  - Step
  - Set and remove breakpoint.

What is a Debugger

- Visualisation of the debugged target state.
- Some of the most common visualisation features:
  - Evaluate variables
  - Stacktrace, unwinding call stack
  - Show machine and Assembly code
  - Show relevant source code.

Unoptimized Vs Optimized code

- Unoptimized:
  - All variables stored in memory.
  - Very simmlilar to source code.
  - Slow to exectue.
  - Easy to debug.
- Optimized:
  - Faster to execute.
  - Some Variables temporarly stored in registery.
  - Some functions are inlined.
  - Difficult to debug.

Motivation

- Unoptimized Rust code is to slow.
- Debugging embedded systems.
- GDB and LLDB do not work very well.
- Write a debugger in Rust.

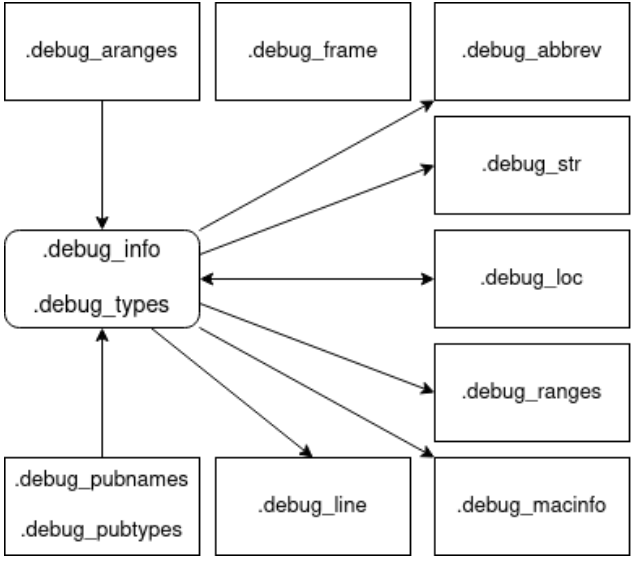
DWARF

DWARF

DWARF

- Debugging with Attributed Record Formats(DWARF)
- Debug information format
- Rust uses DWARF version 4
- DWARF is devided into 12 sections
- Executable and Linkable Format(ELF)

DWARF Sections



Debug Information Entry(DIE)

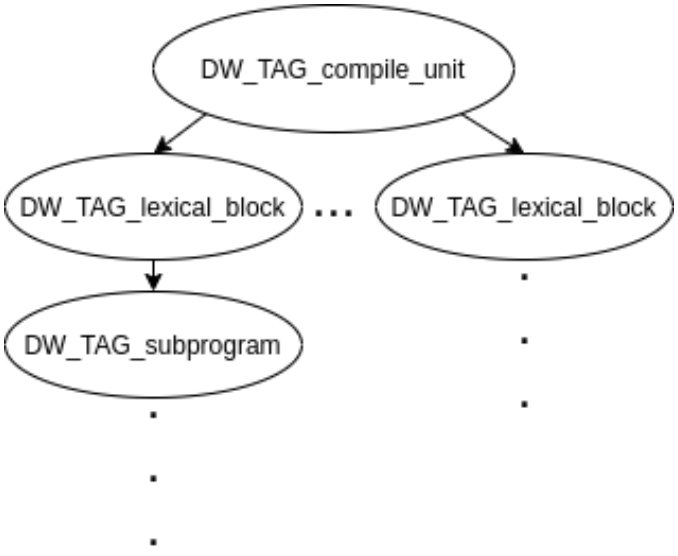
- DebugInformation Entry(DIE).
- DWARF Attributes.
- DWARF DIE example from the .debug_info section.

```
<8><241>: Abbrev Number: 9 (DW_TAG_variable)
   <242>   DW_AT_location    : 2 byte block: 7d 3c     (DW_OP_breg13 (r13): 60)
   <245>   DW_AT_name        : (indirect string, offset: 0x40466): ptr
   <249>   DW_AT_decl_file   : 1
   <24a>   DW_AT_decl_line   : 591
   <24c>   DW_AT_type        : <0x1069>
```

Compilation unit

- Computer program is devieded into compilation units.
- Each compilation unit contains a DIE tree.

Compilation unit



Evaluating a variable

- Find the current compilation unit.
- Find the current subprogram die.
- Find the searched variable die.
- Two parts to evaluating a variable:
  - Finding the location of the variable
  - Parsing the value into the correct type

Evaluating the location of a variable

```
<2><4321>: Abbrev Number: 16 (DW_TAG_subprogram)
   <4322>   DW_AT_low_pc      : 0x8000fca
   <4326>   DW_AT_high_pc     : 0x2c
   <432a>   DW_AT_frame_base  : 1 byte block: 57       (DW_OP_reg7 (r7))
   <432c>   DW_AT_linkage_name: (indirect string, offset: 0x473b8): _ZN24nucleo_r
   <4330>   DW_AT_name        : (indirect string, offset: 0x64a52): my_function
   <4334>   DW_AT_decl_file   : 1
   <4335>   DW_AT_decl_line   : 194
   <4336>   DW_AT_type        : <0x6233>
<3><433a>: Abbrev Number: 17 (DW_TAG_formal_parameter)
   <433b>   DW_AT_location    : 2 byte block: 91 7e    (DW_OP_fbreg: -2)
   <433e>   DW_AT_name        : (indirect string, offset: 0x11d94): val
   <4342>   DW_AT_decl_file   : 1
   <4343>   DW_AT_decl_line   : 194
   <4344>   DW_AT_type        : <0x6233>
```

Parsing the type of a variable

```
<1><6233>: Abbrev Number: 34 (DW_TAG_base_type)
   <6234>   DW_AT_name        : (indirect string, offset: 0x2a125): i16
   <6238>   DW_AT_encoding    : 5      (signed)
   <6239>   DW_AT_byte_size   : 2
```

Virtually Unwinding Call Stack

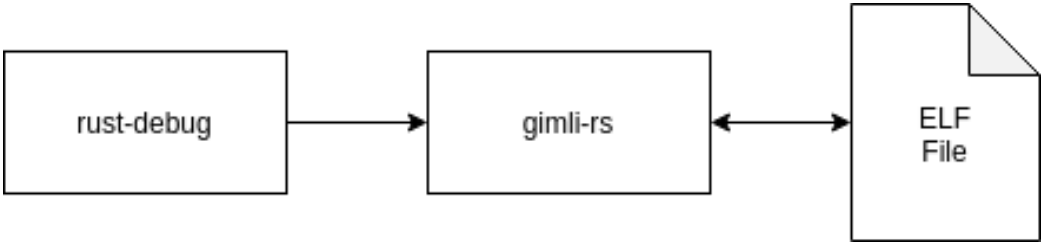
- Stack of subroutine activations.
- A subroutine activation consists of:
  - Code location were the subroutine stopped
  - Preserved register values
  - Canonical Frame Address (CFA)
- The needed infomarion is in section .debug_frame

Virtually Unwinding Subroutine Activations

1. Find the Common Information Entry (CIE)
2. Find the Frame Description Entry (FDE)
3. Unwind CFA and register values.
4. Repeat for all activations.

```
LOC CFA R0 R1 ... RN
L0
L1
...
LN
```

Debugging library rust-debug

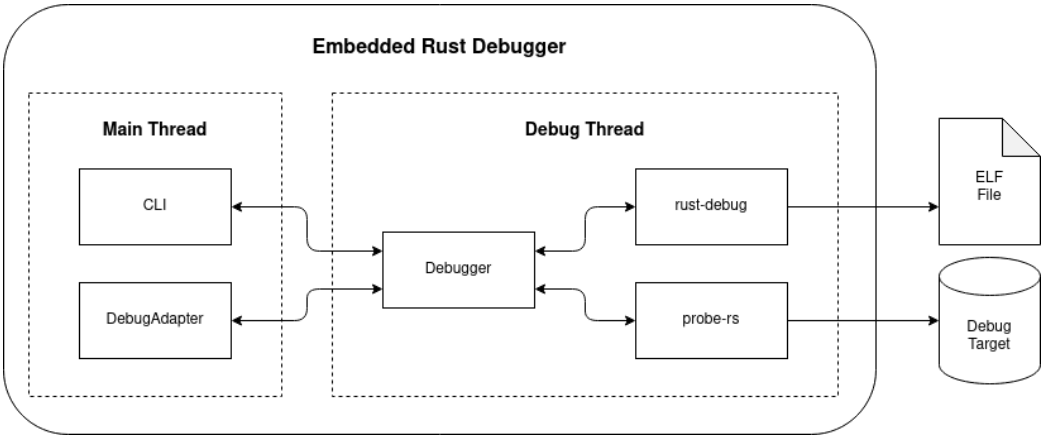


Debugging library rust-debug

Features

- Virtualy Unwinding Stack and Stacktrace
- Evaluating Variables
- Finding breakpoint location
- Retriving source location information from a DIE
- And more

Embedded Rust Debugger(ERD)



Comparing Evalutation Of Rust Enums

Rust Source Code

```
let mut test_enum3 = TestEnum::Struct(TestStruct { flag: true, num: 123 });
```

ERD

```
test_enum3 = TestEnum { < OptimizedOut >}
```

GDB

```
(gdb) p test_enum3
$ 1 = nucleo_rtic_blinking_led::TestEnum::ITest(<optimized out>)
```

Comparing Evalutation Of Rust Enums

Rust Source Code

```
let mut test_enum3 = TestEnum::Struct(TestStruct { flag: true, num: 123 });
```

LLDB Version 13

```
(nucleo_rtic_blinking_led::TestEnum) test_enum3 = {
  ITest = (0 = 0)
  UTest = (0 = 0)
  Struct = {
    0 = (flag = false, num = 0)
  }

  Non = {}
}
```

Evaluating Rust enums

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida

Conclution

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida

Demo

References


John Smith (2012)
Title of the publication
Journal Name 12(3), 45 – 678.

Thank you for listening

Bullet Points

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida

Blocks of Highlighted Text

In this slide, some important text will be highlighted because it's important. Please, don't abuse it.

Block

Sample text

Alertblock

Sample text in red box

Examples

Sample text in green box. The title of the block is "Examples".

Multiple Columns

Heading

1. Statement
2. Explanation
3. Example

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Table

Treatments	Response 1	Response 2
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table: Table caption

Theorem

Theorem (Mass–energy equivalence)

$E = mc^2$

Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.

Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References


John Smith (2012)
Title of the publication
Journal Name 12(3), 45 – 678.

The End