Debug Info

Status and Directions

Eric Christopher
echrisko@google.com
Introduction

What works today?

What doesn't work?

Where are we going?
Debugging a few years ago?
std::cout << "My variable is: " << MyVar << "\n";

assert(false && "Why are we here?");
Debugging today!
So what really works?

Debugging

C, C++, Objective-C(++)

> 97% of the gdb testsuite, 100% of the lldb testsuite
C++ 11 Status

These things are done:

- rvalue references
- enum classes
- enums with fixed types
- enum forward declarations
- unions with special member functions
- inline namespaces
- nullptr_t
- lambdas*
So what doesn't work?

<table>
<thead>
<tr>
<th>GDB Testsuite</th>
<th>Stabs?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unused types</td>
</tr>
<tr>
<td></td>
<td>C++ Template Edge Cases</td>
</tr>
<tr>
<td></td>
<td>Labels</td>
</tr>
<tr>
<td></td>
<td>Line info for constants</td>
</tr>
<tr>
<td></td>
<td>Unused struct parameters</td>
</tr>
<tr>
<td></td>
<td>TLS variables</td>
</tr>
</tbody>
</table>

Line break interpretations

[PR14330]
Unused Struct Parameters

```c
struct foo { long a, b, c, d; };  
void func(foo f, int i) {
}

int main() {
    foo f;  
    func(f, 3);
}
```

ptype func

type = void (foo, int)

type = void (int)
Referenced Constants

```c
int main() {
    FILE *f = stderr;
}
```
So what doesn't work?

C++ 11 debugging support isn't complete

DWARF4 Missing Features

Optimized and LTO Debugging
Where are we going from here?
C++ 11 Status

These things are not done:

- atomic types
- template aliases
- user defined literals
- capture 'this' in a lambda
Immediate

DWARF4: Finish off full support

DWARF5: Fission

DWARF5: Accelerated Access
DWARF4

Shrink all the things!
DWARF4 - Size Optimizations

Type Units

Encoding changes

Compression techniques
DWARF5 - Fission

Splitting debug information

Complete implementation

Submitted to committee
DWARF5 - Fission

Fewer Relocations

Parallelizable Linking

Faster initial link step
DWARF5 - Accelerated Access

Speeding up debugger access

Complete implementation

Submitted to committee
DWARF5 - Accelerated Access

Compact tables

Fast access

Extensible

Strictly specified contents
Near Term - LTO and Optimized Code
LTO and Optimized Code

Variable tracking

Type merging
A.h:

class A {
    int a;
};

A a;
LTO - Type Merging

!4 = metadata !{i32 786484, i32 0, null, metadata !"a", metadata !"a", metadata !"", metadata !5, i32 5, metadata !6, i32 0, i32 1, %class.A* @a, null} ; [ DW_TAG_variable ] [a] [line 5] [def]

!6 = metadata !{i32 786434, metadata !1, null, metadata !"A", i32 1, i64 32, i64 32, i32 0, i32 0, null, metadata !7, i32 0, null, null} ; [ DW_TAG_class_type ] [A] [line 1, size 32, align 32, offset 0] [from ]
LTO - Type Merging

Foo.cpp:

#include "A.h"
#include "B.h"

Bar.cpp:

#include "B.h"
#include "A.h"
LTO - Type Merging

!4 = metadata !{i32 786484, i32 0, null, metadata !"a", metadata !"a", metadata !"", metadata !5, i32 5, metadata !6, i32 0, i32 1, %class.A* @a, null} ; [ DW_TAG_variable ] [a] [line 5] [def]

!6 = metadata !{i32 786434, metadata !1, null, metadata !"A", i32 1, i64 32, i64 32, i32 0, i32 0, null, metadata !7, i32 0, null, null} ; [ DW_TAG_class_type ] [A] [line 1, size 32, align 32, offset 0] [from ]
LTO - Line Tables Only

Line tables... -gline-tables-only

and some minimal DIEs.
C++ Modules

Plenty of ideas... ASTs for types

... no concrete plans with DWARF for line tables

and more DWARF for archival purposes?
How Can I Help?

C++ 11 Features

PR14330

Identify and report bugs

Discover size optimizations

Donuts?
Questions?
Image Credits

hyperboleandahalf.blogspot.com
retiredindelaware.blogspot.com
icanhazcheeseburger.com
chemiphysic.blogfa.com