What is Going on with Debugging Info? (2026)

DWARF6 and Support for GPUs and Vectorized Code

Ben Woodard

Red Hat

Disclaimer

I'm presenting this but this is **certainly** not exclusively my work. This is the work of the DWARF for GPU's group. Participants include:

- Started by AMD and they did the original design
  (highly evolved since then)
- TotalView (John DelSignore) instrumental in the design
- Led by Cary Coutant the DWARF Committee chair
- Intel has made important contributions
- Nvidia is actively participating
- Red Hat

Source:
https://llvm.org/docs/AMDGPUDwarfExtensionsForHeterogeneousDebugging.html#introduction
https://github.com/ccoutant/dwarf-locations/tree/main
https://snapshots.sourceware.org/dwarfstd/dwarf-spec/2025-05-17_20-15_1747512902/dwarf6-20250517-2014.pdf

Overall status of DWARF

After several years of hiatus post DWARF5. The DWARF committee is meeting regularly being led by Cary Coutant (ELF standard maintainer).

- I'm now on the committee & driving the agenda.
- People are continuing to work constructively
- Draft standard is being published in the open.
- DWARF6 is probably about 9 months out (guess)
- GPU support is the marquee feature.

Source:
https://lists.dwarfstd.org/mailman/listinfo/dwarf-discuss
https://dwarfstd.org/issues.html
https://snapshots.sourceware.org/dwarfstd/dwarf-spec/

Simple changes

New since last year

- Allocator call sites
- Vtable improvement
- C++ packs
- Operation headings
- Address Spaces and Address Classes

Source:
https://dwarfstd.org/issues/250407.1.html
https://dwarfstd.org/issues/250506.1.html https://dwarfstd.org/issues/250506.2.html https://dwarfstd.org/issues/250506.3.html
https://dwarfstd.org/issues/250925.1.html
https://dwarfstd.org/issues/260127.1.html
https://dwarfstd.org/issues/260617.1.html

Allocator sites

Status: approved in the current draft standard

Add type to allocation site when known.

- Already exists in MS and LLVM IR
- DW_AT_alloc_type <type DIE>
- Applied to DW_TAG_call_site or DW_TAG_inlined_subroutine
- Uses:
  - Heap profiling
  - Associating PEBS/IBS with data cache access
  - Associating memory access with type info.

Source:
https://dwarfstd.org/issues/250407.1.html

VTable improvements

Status: approved in the current draft standard

Long overdue. Help consumers find vtable rather than depending on ABI.

- DW_AT_vtable_location applied to a variable.
- DW_TAG_vtable with DW_AT_vtable_for_type
  - Quick to index rather than looking through all DIEs
- DW_AT_vtable_elem_location (DEPRECATED) - improperly implemented
- DW_AT_vtable_elem_index - what GCC/LLVM have been doing just with a new attribute

Source:
https://dwarfstd.org/issues/250506.1.html https://dwarfstd.org/issues/250506.2.html https://dwarfstd.org/issues/250506.3.html

VTable example

```
class MyClass {
 public:
     virtual void doSomething();
     int myData; // 4 bytes
 };
```

```
<1><100>: Abbrev Number: 2 (DW_TAG_class_type)
    <101>   DW_AT_name        : "MyClass"
    <109>   DW_AT_byte_size   : 16   // 8 bytes for vptr, 4 for int, 4 padding
    <10a>   DW_AT_vtable_location: 1 byte block: 6 (DW_OP_deref)
 <2><110>: Abbrev Number: 3 (DW_TAG_member)
    <111>   DW_AT_name        : "myData"
    <115>   DW_AT_type        : <Reference to 'int' type>
    <119>   DW_AT_data_member_location: 8 // Data starts after the 8-byte vptr
<2><120>: Abbrev Number: 4 (DW_TAG_subprogram)
    <121>   DW_AT_name        : "doSomething"
    <125>   DW_AT_virtuality  : 1        // (DW_VIRTUALITY_virtual)
    <126>   DW_AT_vtable_elem_index: 0   // The zero-based slot index!
    <127>   DW_AT_declaration : 1
 <2><130>: Abbrev Number: 0 (end of children of MyClass)
 <1><200>: Abbrev Number: 6 (DW_TAG_vtable)
    <201>   DW_AT_artificial  : 1
    <202>   DW_AT_location    : <Address of the vtable>
    <210>   DW_AT_vtable_for_type: <0x100> // Points directly back to MyClass
```


Source:
https://dwarfstd.org/issues/250506.1.html https://dwarfstd.org/issues/250506.2.html https://dwarfstd.org/issues/250506.3.html

C++ packs

Status: accepted

Like varargs but for modern C++

- Template parameter packs
- Formal parameter packs
- Lambda capture packs
- Possibly other packs.

Source:
lhttps://github.com/ccoutant/dwarf-locations/blob/main/001a-context.md
https://dwarfstd.org/issues/241011.1.html

Operation Headers

Status: accepted

Largely editorial. Like man pages for DWARF operators

- Standardized how operators are presented.
- Removes confusion between inline operands and stack operands.
- First step in larger plan
  - Reorganize chapter 3 DWARF expressions (submitted)
  - Rewrite descriptions. (planned but not yet written)
  - Formal description of DWARF expressions

Source:
https://dwarfstd.org/issues/211206.1.html

Address Classes & Address Spaces

Status: Accepted

- GPUs certainly require Address Spaces
- Address Classes were almost deprecated but we stepped back
- We may still deprecate xderef operators. (undecided)
- Really made people much more aware of DWARF expression evaluation context.
    - Thread and Lane may be needed to select the correct instance of that memory address.

Source:
https://dwarfstd.org/issues/260211.1.html
https://dwarfstd.org/issues/260617.1.html

Bonus: C++ nested template parameters

Status: posted not yet discussed

Preserves source level template parameter relationship

- Needed for template meta programming (e.g. RAJA and Kokkos)
- GCC & LLVM currently agree but are not following the current DWARF5 spec
- Conceptually extremely simple and no new DWARF
- Current concern is challenges with implementability.

Source::
https://dwarfstd.org/issues/260703.1.html

Nested template example


```
template<typename U> struct t1 {
  U n1;
};

template<typename T> struct t2 {
  T m1;
  t1<T> m2;
};

int main() {
  t1<int> v0;
  t2<int> v1;
  return 0;
}
```

```
10$: DW_TAG_structure_type
       DW_AT_name("t1")
11$:   DW_TAG_template_type_parameter
         DW_AT_name("U")
         DW_AT_type(reference to int)
12$:   DW_TAG_member
         DW_AT_name("n1")
         DW_AT_type(reference to 11$)    ! points to parameter "U"
20$: DW_TAG_structure_type
       DW_AT_name("t2")
21$:   DW_TAG_template_type_parameter
         DW_AT_name("T")
         DW_AT_type(reference to int)     ! points to "int"
22$:   DW_TAG_member
         DW_AT_name("m1")
         DW_AT_type(reference to 21$)    ! points to parameter "T"
23$:   DW_TAG_template_alias
         DW_AT_name("t1")
         DW_AT_type(reference to 10$)    ! points to the top-level t1<int>
         DW_AT_artificial
24$:     DW_TAG_template_type_parameter
           DW_AT_name("U")
           DW_AT_type(reference to 21$)  ! points to enclosing parameter "T"
25$:   DW_TAG_member
         DW_AT_name("m2")
         DW_AT_type(reference to 23$)    ! points to the template alias
```

Source:
https://dwarfstd.org/issues/260703.1.html

Supporting GPUs

Last year vs. this year

Feature	**2025 status**	**2026 status**
Expression Context	Accepted	Refined by other proposals
Locations on the stack	Final review	Accepted
Overlays	Nearly ready for submission	Final review
Address spaces	On deck from GPU group	Accepted
Refined types	Not yet written	Written - deferred
CFI/CFA	Not complete	On deck

Overlays

Status: final review

A better way to make composites & basis for future work

- Most controversial topic this year.
- Some disagreement about removing piece operators. Still might be deprecated in DWARF6.
- Basis for Incremental location lists - replaces overlapping ranges
  - Introduces transparent vs. opaque overlays
  - Alternative would be multiloc
- Autovectorization & Scalar Replacement of Aggregates the big driver
  - Consumers need to plan on data being in multiple places
  - RO vs RW copies

Source:
https://dwarfstd.org/issues/251120.1.html
https://dwarfstd.org/issues/251017.2.html https://dwarfstd.org/issues/260227.1.html

CFI/CFA -> CFI/CFL

Status: in development

- On GPUs registers can be spilled to locations other than main memory
- Canonical Frame Address or Call Frame Address becomes Call Frame Location
- Yet another update to DWARF Expression Context
- Several minor errors in spec addressed during development.
  - Which operators can be used double checked.
- How to make it work for callee saved vector registers is an open question
- Locations on the stack make register rules theoretically obsolete
- Concerns about breaking with ELF
  - Skipping .debug_frame and only emitting .eh_frame is a useful optimzation

Source:
https://github.com/woodard/dwarf-locations/blob/cfi-update/003-cfi-update.md

Refined types

Status: written but deferred

Consumers need to know when the changes the type

- Size of a pointer may change based on address space
- Size of a register is different from the memory size
- Compiler proves that a smaller type can be used.
- Very controversial even within the DWARF for GPUs group.
  - May be other ways to solve the problem
  - No implementation experience

Other things on deck

Status: not complete

Still many changes to come

- Call frame entry registers - like DW_OP_entry_value but uses CFI
- Need more operations to selective spill registers based upon exec mask DW_OP_extend DW_OP_select_bit_piece
  - Kind of like overlay but with bitmasks
  - Needed for divergent flow control
  - Needed for logical PCs in SIMT sections with fused threads
- Memory spaces - type modifier for source language e.g. “private”
- Data in multiple locations
- Lots of clarifications and details

Source:
https://github.com/ccoutant/dwarf-locations/blob/main/003-clarifications-cfi.txt https://github.com/ccoutant/dwarf-locations/blob/main/017-call-frame-entry-registers.txt
https://github.com/ccoutant/dwarf-locations/blob/main/015-vector-composite-location-descriptions.txt
https://github.com/ccoutant/dwarf-locations/blob/main/022-memory-spaces.txt

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat

Crazy Ideas

Status: not yet written

Let's see what we can do with these

- Make split-dwarf work for packages like RPM and packagers like spack. I have an idea.
- Extend CFI table to refer to variables and expressions. It might be more compact than location lists.
- Remove CFI register rules it is all expressions
- Retire antiquated DWARF logo and replace it with: