

## [PATCH] Kgdb dwarf2 for asm

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-01/6155.html>

---

**From:** George Anzinger ([george\\_at\\_mvista.com](mailto:george_at_mvista.com))

**Date:** 01/24/04

Date: Fri, 23 Jan 2004 19:47:11 -0800  
To: Tom Rini <[trini@kernel.crashing.org](mailto:trini@kernel.crashing.org)>

This is a second try at this with a different subject and the note on gdb version at the end.

Here is the dwarf2 patch. It assumes that mm-kgdb is already there, but there are really very few dependencies. With little effort it should work with Amit's version.

What it does:

This patch allows back trace and "up" commands to go smoothly back to the root of the stack through interrupt frames, trap frames, you name it. It also tells the asm compile engine to save the temp source as this is what gas builds debug records for. (This is not needed to get the back trace, its just "nice".)

I removed the debug info overlap in kconfig. Kgdb sets -gdwarf2 while that set -g. Not nice to have both.

There is only one code change and that is to push a "1" on the stack so that it looks like the return address for idle tasks other than the cpu 0 idle task. This is done so that a "bt" on that task will not go on forever (I had a screen with 1000+ levels on it...).

What it is:

The dwarf2.h file from the binutils tree is here as it is there. There is an awk script to build dwarf2-defs.h which is nothing but all the constants, from that file, in a form that asm AND c understand. The script works for this file, for other files you are on your own. This translation is forced in the main make file, I suppose this could be done more cleanly, but... I just did not want to figure out the make maze.

The dwarf2-lang.h file is a set of macros that make writing dwarf2 half way enjoyable and much more understandable.

All most all of the dwarf2 code is at the end of entry.S but for 2 lines in head.S.

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

Daniel keeps telling me that this should be done in asm, and I suppose one could do it there, but it does not support the expression code as yet and that is where most of the magic is.

Oh and by the way, you do need call frames (CONFIG option) as without them gdb will most likely get lost in the switch code in schedule(). This comes up in the "info thread" command so it is not uncommon.

What it needs:

For the dwarf2 expression code to be correctly understood by gdb you will need to get the CVS version of gdb as of at least 1/24/04. Otherwise, the bug in the gdb expression code will most likely stop the unwind at the interrupt/trap frame regardless of having interrupted the kernel and not user space code.

--

George Anzinger [george@mvista.com](mailto:george@mvista.com)  
High-res-timers: <http://sourceforge.net/projects/high-res-timers/>  
Preemption patch: <http://www.kernel.org/pub/linux/kernel/people/rml>

```
diff -urP -I '\$Id:.*Exp \$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/Makefile linux/Makefile
--- linux-2.6.0-akgdb/Makefile 2004-01-08 15:36:35.000000000 -0800
+++ linux/Makefile 2004-01-22 13:01:31.000000000 -0800
@@ -1025,5 +1025,7 @@
descend =$(Q)$(MAKE) -f $(if $(KBUILD_SRC),$(srctree)/)scripts/Makefile.build obj=$(1) $(2)
```

```
endif # skip-makefile
+include/linux/dwarf2-defs.h: include/linux/dwarf2.h scripts/dwarfh.awk
+ awk -f scripts/dwarfh.awk include/linux/dwarf2.h > include/linux/dwarf2-defs.h
```

-FORCE:

+FORCE: include/linux/dwarf2-defs.h

Binary files linux-2.6.0-akgdb/a.out and linux/a.out differ

```
diff -urP -I '\$Id:.*Exp \$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/arch/i386/Kconfig
linux/arch/i386/Kconfig
--- linux-2.6.0-akgdb/arch/i386/Kconfig 2004-01-08 15:39:58.000000000 -0800
+++ linux/arch/i386/Kconfig 2004-01-09 14:50:07.000000000 -0800
@@ -1196,7 +1196,7 @@
```

config DEBUG\_INFO

bool "Compile the kernel with debug info"

- depends on DEBUG\_KERNEL

+ depends on DEBUG\_KERNEL && !KGDB

help

If you say Y here the resulting kernel image will include debugging info resulting in a larger kernel image.

```
@@ -1227,7 +1227,7 @@
```

will then break as soon as it looks at the boot options. This option also installs a breakpoint in panic and sends any kernel faults to the debugger. For more information see the

- Documentation/i386/kgdb.txt file.

[PATCH] Kgdb dwarf2 for asm

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

+ Documentation/i386/kgdb/kgdb.txt file.

choice

depends on KGDB

```
diff -urP -I '$Id:.*Exp $' -X /usr/src/patch.exclude linux-2.6.0-akgdb/arch/i386/Makefile
linux/arch/i386/Makefile
```

```
--- linux-2.6.0-akgdb/arch/i386/Makefile 2004-01-08 15:39:58.000000000 -0800
```

```
+++ linux/arch/i386/Makefile 2004-01-13 16:04:02.000000000 -0800
```

```
@@ -100,7 +100,9 @@
```

```
drivers-$(CONFIG_PM) += arch/i386/power/
```

```
CFLAGS += $(mflags-y)
```

```
-AFLAGS += $(mflags-y)
```

```
+aflags-y = $(mflags-y)
```

```
+aflags-$(CONFIG_KGDB) += -save-temps
```

```
+AFLAGS += $(aflags-y)
```

```
boot := arch/i386/boot
```

```
diff -urP -I '$Id:.*Exp $' -X /usr/src/patch.exclude linux-2.6.0-akgdb/arch/i386/kernel/entry.S
linux/arch/i386/kernel/entry.S
```

```
--- linux-2.6.0-akgdb/arch/i386/kernel/entry.S 2004-01-13 15:53:06.000000000 -0800
```

```
+++ linux/arch/i386/kernel/entry.S 2004-01-23 16:03:00.000000000 -0800
```

```
@@ -910,3 +910,116 @@
```

```
.long sys_ni_syscall /* sys_vserver */
```

```
syscall_table_size=(.-sys_call_table)
```

```
+
```

```
+# Here we do call frames. We cheat a bit as we only really need
+# correct frames at locations we can actually look at from a
+# debugger. Since the break instruction trap actually goes thru
+# some of this code, we don't really need info on those areas, but
+# only after the fact. I.e. if we can not step or break in a
+# location or end up with a return address pointing at the
+# location, we don't need a correct call frame for it.
```

```
+
```

```
+#ifdef CONFIG_KGDB
```

```
+
```

```
+#include <linux/dwarf2-lang.h>
```

```
+/*
```

```
+ * The register numbers as known by gdb
```

```
+ */
```

```
+cfa_one: .long 1
```

```
+
```

```
+#define _EAX 0
```

```
+#define _ECX 1
```

```
+#define _EDX 2
```

```
+#define _EBX 3
```

```
+#define _ESP 4
```

```
+#define _EBP 5
```

```
+#define _ESI 6
```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+#define _EDI 7
+#define _PC 8
+#define _EIP 8
+#define _PS 9
+#define _EFLAGS 9
+#define _CS 10
+#define _SS 11
+#define _DS 12
+#define _ES 13
+#define _FS 14
+#define _GS 15
+
+ CFI_preamble(c1,_PC,1,1)
+ CFA_def_cfa_expression
+ CFA_exp_OP_breg(_ESP,OLDESP) /* push the stack & adjust */
+ CFA_exp_OP_dup /* save that */
+ CFA_exp_OP_consts(-8) /* offset to CS address */
+ CFA_exp_OP_plus /* should be CS address */
+ CFA_exp_OP_deref /* get the CS */
+ CFA_exp_OP_const4s(VM_MASK|3) /* prepare to mask it */
+ CFA_exp_OP_and /* mask it, zero means kernel */
+ CFA_exp_OP_bra(cfa_user_rtn) /* branch if user (TOS---) */
+ CFA_exp_OP_skip(cfa_end) /* done if kernel, skip out */
+cfa_user_rtn:
+ CFA_exp_OP_constu(0) /* user space, set exp to 0 */
+cfa_end:
+ CFA_expression_end
+ CFA_expression(_EIP)
+ CFA_exp_OP_dup /* copy old esp */
+ CFA_exp_OP_bra(eip_kernel_rtn) /* non-zero means kernel */
+ CFA_exp_OP_addr(cfa_one) /* for user, pass 1 */
+ CFA_exp_OP_skip(eip_end) /* done if kernel, skip out */
+eip_kernel_rtn:
+ CFA_exp_OP_const4s(EIP-OLDESP) /* offset to return address */
+ CFA_exp_OP_plus
+eip_end:
+ CFA_expression_end
+ CFA_define_offset(_EBX,EBX-OLDESP)
+ CFA_define_offset(_ECX,ECX-OLDESP)
+ CFA_define_offset(_EDX,EDX-OLDESP)
+ CFA_define_offset(_ESI,ESI-OLDESP)
+ CFA_define_offset(_EDI,EDI-OLDESP)
+ CFA_define_offset(_EBP,EBP-OLDESP)
+ CFA_define_offset(_EAX,EAX-OLDESP)
+ CFA_define_offset(_EFLAGS,EFLAGS-OLDESP)
+ CFI_postamble()
+
+/*
+ * This is an attempt to provide an unwind for location 0 which is
+ * identical to the prior frame. Should stop the unwind. Note that gdb
+ * uses the pc-1 to look for FDEs so the correct pc is 1 not 0 which,
```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ * because of addresses being unsigned, can not be used by man nor beast.
+ */
+ CFI_preamble(c2,_PC,1,1)
+ CFA_def_cfa_expression
+ CFA_exp_OP_constu(0) /* stack end, set exp to 0 */
+ CFA_expression_end
+ CFI_postamble()
+
+ FDE_preamble(c2,0,10)
+ FDE_postamble()
+ /*
+ * This is VERY sloppy. At this point all we want to do is get
+ * the frame right for back tracing. It will not be good if
+ * you try to single step. We use already defined labels.
+ * We want to cover all call outs.
+ * We could also recode this as just one FDE, but this works and
+ * I want to get it out.
+ */
+ FDE_preamble(c1,do_lcall,(ret_from_fork - do_lcall))
+ CFA_define_cfa_offset(OLDESP+8)
+ FDE_postamble()
+ FDE_preamble(c1,ret_from_fork,(ret_from_exception - ret_from_fork))
+ CFA_define_cfa_offset(OLDESP+4)
+ FDE_postamble()
+ FDE_preamble(c1,ret_from_exception,(divide_error - ret_from_exception))
+ FDE_postamble()
+ FDE_preamble(c1,error_code,(device_not_available_emulate - error_code))
+ CFA_define_cfa_offset(OLDESP+8)
+ FDE_postamble()
+ FDE_preamble(c1,device_not_available_emulate,(debug - device_not_available_emulate))
+ CFA_define_cfa_offset(OLDESP+4)
+ FDE_postamble()
+ FDE_preamble(c1,nmi,(int3 - nmi))
+ CFA_define_cfa_offset(OLDESP+8)
+ FDE_postamble()
+
+##
+##endif
diff -urP -I '$Id:.*Exp $' -X /usr/src/patch.exclude linux-2.6.0-akgdb/arch/i386/kernel/head.S
linux/arch/i386/kernel/head.S
--- linux-2.6.0-akgdb/arch/i386/kernel/head.S 2003-09-08 14:15:17.000000000 -0700
+++ linux/arch/i386/kernel/head.S 2004-01-23 16:19:21.000000000 -0800
@@ -278,6 +278,16 @@
        # just in case, we know what happens.

ready: .byte 0
#ifdef CONFIG_KGDB
#include <linux/dwarf2-lang.h>
+
+ /* This dwarf code tells gdb that this is the end of the unwind */
+ /* This uses the CFA set up for pc=1 located in entry.S */
```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+
+ FDE_preamble(c2,is386,(L6-is386))
+ FDE_postamble()
+
+ #endif

/*
 * We depend on ET to be correct. This checks for 287/387.
diff -urP -I '\$Id:.*Exp \$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/arch/i386/kernel/smpboot.c
linux/arch/i386/kernel/smpboot.c
--- linux-2.6.0-akgdb/arch/i386/kernel/smpboot.c 2003-11-10 15:59:40.000000000 -0800
+++ linux/arch/i386/kernel/smpboot.c 2004-01-18 00:33:32.000000000 -0800
@@ -476,6 +476,9 @@

asm volatile(
    "movl %0,%esp\n\t"
+ #ifdef CONFIG_KGDB
+ "pushl $1\n\t"
+ #endif
    "jmp *%1"
    :
    : "r" (current->thread.esp), "r" (current->thread.eip));
diff -urP -I '\$Id:.*Exp \$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/include/linux/dwarf2-lang.h
linux/include/linux/dwarf2-lang.h
--- linux-2.6.0-akgdb/include/linux/dwarf2-lang.h 1969-12-31 16:00:00.000000000 -0800
+++ linux/include/linux/dwarf2-lang.h 2004-01-22 13:06:43.000000000 -0800
@@ -0,0 +1,286 @@
+ #ifndef DWARF2_LANG
+ #define DWARF2_LANG
+
+
+ /*
+ /*
+ * This is free software; you can redistribute it and/or modify it under
+ * the terms of the GNU General Public License as published by the Free
+ * Software Foundation; either version 2, or (at your option) any later
+ * version.
+ */
+ /*
+ /* This file defines macros that allow generation of DWARF debug records
+ * for asm files. This file is platform independent. Register numbers
+ * (which are about the only thing that is platform dependent) are to be
+ * supplied by a platform defined file.
+ */
+ /*
+ /* We need this to work for both asm and C. In asm we are using the
+ * old comment trick to concatenate while C uses the new ANSI thing.
+ * Here we have concat macro... The multi level thing is to allow and
+ * macros used in the names to be resolved prior to the cat (at which
+ * time they are no longer the same string).
+ */
+ #define CAT3(a,b,c) _CAT3(a,b,c)
+ #define _CAT3(a,b,c) __CAT3(a,b,c)
```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+#ifdef __ASSEMBLY__
+#define __CAT3(a,b,c) a/**/b/**/c
+#define IFC(a)
+#define IFN_C(a) a
+#define NL ;
+#define QUOTE_THIS(a) a
+#define DWARF_preamble .section .debug_frame,"",@progbits;
+#else
+#define __CAT3(a,b,c) a ## b ## c
+#define IFC(a) a
+#define IFN_C(a)
+#define NL \n\t
+#define QUOTE_THIS(a) _QUOTE_THIS(a)
+#define _QUOTE_THIS(a) #a
+/* Don't let CPP see the " and , \042=" \054=, */
+#define DWARF_preamble .section .debug_frame \054\042\042\054@progbits
+
+#endif
+#include <linux/dwarf2-defs.h>
+/*
+ * This macro starts a debug frame section. The debug_frame describes
+ * where to find the registers that the enclosing function saved on
+ * entry.
+ *
+ * ORD is use by the label generator and should be the same as what is
+ * passed to CFI_postamble.
+ *
+ * pc, pc register gdb ordinal.
+ *
+ * code_align this is the factor used to define locations or regions
+ * where the given definitions apply. If you use labels to define these
+ * this should be 1.
+ *
+ * data_align this is the factor used to define register offsets. If
+ * you use struct offset, this should be the size of the register in
+ * bytes or the negative of that. This is how it is used: you will
+ * define a register as the reference register, say the stack pointer,
+ * then you will say where a register is located relative to this
+ * reference registers value, say 40 for register 3 (the gdb register
+ * number). The <40> will be multiplied by <data_align> to define the
+ * byte offset of the given register (3, in this example). So if your
+ * <40> is the byte offset and the reference register points at the
+ * begining, you would want 1 for the data_offset. If <40> was the 40th
+ * 4-byte element in that structure you would want 4. And if your
+ * reference register points at the end of the structure you would want
+ * a negative data_align value(and you would have to do other math as
+ * well).
+ */
+
+#define CFI_preamble(ORD, pc, code_align, data_align) \
+DWARF_preamble NL \
```

```

+ .align 4 NL \
+ .globl CAT3(frame,_,ORD) NL \
+CAT3(frame,_,ORD): NL \
+ .long 7f-6f NL \
+6: \
+ .long DW_CIE_ID NL \
+ .byte DW_CIE_VERSION NL \
+ .byte 0 NL \
+ .uleb128 code_align NL \
+ .sleb128 data_align NL \
+ .byte pc NL
+
+/*
+ * After the above macro and prior to the CFI_postamble, you need to
+ * define the initial state. This starts with defining the reference
+ * register and, usually the pc. Here are some helper macros:
+ */
+
+#define CFA_define_reference(reg, offset) \
+ .byte DW_CFA_def_cfa NL \
+ .uleb128 reg NL \
+ .uleb128 (offset) NL
+
+#define CFA_define_offset(reg, offset) \
+ .byte (DW_CFA_offset + reg) NL \
+ .uleb128 (offset) NL
+
+#define CFA_restore(reg) \
+ .byte (DW_CFA_restore + reg) NL
+
+#define CFI_postamble() \
+ .align 4 NL \
+7: NL \
+.previous NL
+/*
+ * So now your code pushes stuff on the stack, you need a new location
+ * and the rules for what to do. This starts a running description of
+ * the call frame. You need to describe what changes with respect to
+ * the call registers as the location of the pc moves through the code.
+ * The following builds an FDE (fram descriptor entry?). Like the
+ * above, it has a preamble and a postamble. It also is tied to the CFI
+ * above.
+ * The preamble macro is tied to the CFI thru the first parameter. The
+ * second is the code start address and then the code length.
+ */
+// DWARF_preamble() NL
+#define FDE_preamble(ORD, initial_address, length) \
+ DWARF_preamble NL \
+ .align 4 NL \
+ .long 9f-8f NL \
+8: \

```

```

+ .long CAT3(frame,_,ORD) NL \
+ .long initial_address NL \
+ .long length NL
+
+ #define FDE_postamble() \
+ .align 4 NL \
+9: NL \
+.previous NL
+/*
+ * That done, you can now add registers, subtract registers, move the
+ * reference and even change the reference. You can also define a new
+ * area of code the info applies to. For discontinuous bits you should
+ * start a new FDE. You may have as many as you like.
+ */
+
+ /* To advance the stack address by <bytes> (0x3f max)
+ */
+
+ #define CFA_advance_loc(bytes) \
+ .byte DW_CFA_advance_loc+bytes NL
+
+ /* This one is good for 0xff or 255
+ */
+ #define CFA_advance_loc1(bytes) \
+ .byte DW_CFA_advance_loc1 NL \
+ .byte bytes NL
+
+ /* With the above you can define all the register locations. But
+ * suppose the reference register moves... Takes the new offset NOT an
+ * increment. This is how esp is tracked if it is not saved.
+ */
+
+ #define CFA_define_cfa_offset(offset) \
+ .byte DW_CFA_def_cfa_offset NL \
+ .uleb128 (offset) NL
+/*
+ * Or suppose you want to use a different reference register...
+ */
+ #define CFA_define_cfa_register(reg) \
+ .byte DW_CFA_def_cfa_register NL \
+ .uleb128 reg NL
+
+ /* If you want to mess with the stack pointer, here is the expression.
+ * The stack starts empty.
+ */
+ #define CFA_def_cfa_expression \

```

```

+ .byte DW_CFA_def_cfa_expression NL \
+ .uleb128 20f-10f NL \
+10: NL
+/*
+ * This expression is to be used for other regs. The stack starts with the
+ * stack address.
+ */
+
+#define CFA_expression(reg) \
+ .byte DW_CFA_expression NL \
+ .uleb128 reg NL \
+ .uleb128 20f-10f NL \
+10: NL
+/*
+ * Here we do the expression stuff. You should code the above followed
+ * by expression OPs followed by CFA_expression_end.
+ */
+
+
+#define CFA_expression_end \
+20: NL
+
+#define CFA_exp_OP_const4s(a) \
+ .byte DW_OP_const4s NL \
+ .long a NL
+
+#define CFA_exp_OP_swap .byte DW_OP_swap NL
+#define CFA_exp_OP_dup .byte DW_OP_dup NL
+#define CFA_exp_OP_drop .byte DW_OP_drop NL
+/*
+ * All these work on the top two elements on the stack, replacing them
+ * with the result. Top comes first where it matters. True is 1, false 0.
+ */
+#define CFA_exp_OP_deref .byte DW_OP_deref NL
+#define CFA_exp_OP_and .byte DW_OP_and NL
+#define CFA_exp_OP_div .byte DW_OP_div NL
+#define CFA_exp_OP_minus .byte DW_OP_minus NL
+#define CFA_exp_OP_mod .byte DW_OP_mod NL
+#define CFA_exp_OP_neg .byte DW_OP_neg NL
+#define CFA_exp_OP_plus .byte DW_OP_plus NL
+#define CFA_exp_OP_not .byte DW_OP_not NL
+#define CFA_exp_OP_or .byte DW_OP_or NL
+#define CFA_exp_OP_xor .byte DW_OP_xor NL
+#define CFA_exp_OP_le .byte DW_OP_le NL
+#define CFA_exp_OP_ge .byte DW_OP_ge NL
+#define CFA_exp_OP_eq .byte DW_OP_eq NL
+#define CFA_exp_OP_lt .byte DW_OP_lt NL
+#define CFA_exp_OP_gt .byte DW_OP_gt NL
+#define CFA_exp_OP_ne .byte DW_OP_ne NL
+/*
+ * These take a parameter as noted

```

```

+ */
+/*
+ * Unconditional skip to loc. loc is a label (loc:)
+ */
+#define CFA_exp_OP_skip(loc) \
+ .byte DW_OP_skip NL \
+ .hword loc-.-2 NL
+/*
+ * Conditional skip to loc (TOS != 0, TOS--) (loc is a label)
+ */
+#define CFA_exp_OP_bra(loc) \
+ .byte DW_OP_bra NL \
+ .hword loc-.-2 NL
+
+/*
+ * TOS += no (an unsigned number)
+ */
+#define CFA_exp_OP_plus_uconst(no) \
+ .byte DW_OP_plus_uconst NL \
+ .uleb128 no NL
+
+/*
+ * ++TOS = no (a unsigned number)
+ */
+#define CFA_exp_OP_constu(no) \
+ .byte DW_OP_constu NL \
+ .uleb128 no NL
+/*
+ * ++TOS = no (a signed number)
+ */
+#define CFA_exp_OP_consts(no) \
+ .byte DW_OP_consts NL \
+ .sleb128 no NL
+/*
+ * ++TOS = no (an unsigned byte)
+ */
+#define CFA_exp_OP_const1u(no) \
+ .byte DW_OP_const1u NL \
+ .byte no NL
+
+
+/*
+ * ++TOS = no (a address)
+ */
+#define CFA_exp_OP_addr(no) \
+ .byte DW_OP_addr NL \
+ .long no NL
+
+/*
+ * Push current frames value for "reg" + offset
+ * We take advantage of the opcode assignments to make this a litteral reg

```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ * rather than use the DW_OP_bregx opcode.
+ */
+
+#define CFA_exp_OP_breg(reg,offset) \
+ .byte DW_OP_breg0+reg NL \
+ .sleb128 offset NL
+#endif
diff -urP -I '\$Id:.*Exp \$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/include/linux/dwarf2.h
linux/include/linux/dwarf2.h
--- linux-2.6.0-akgdb/include/linux/dwarf2.h 1969-12-31 16:00:00.000000000 -0800
+++ linux/include/linux/dwarf2.h 2004-01-15 00:26:12.000000000 -0800
@@ -0,0 +1,775 @@
+/* Declarations and definitions of codes relating to the DWARF2 symbolic
+ debugging information format.
+ Copyright (C) 1992, 1993, 1995, 1996, 1997, 1999, 2000, 2001, 2002,
+ 2003 Free Software Foundation, Inc.
+
+ Written by Gary Funck (gary@intrepid.com) The Ada Joint Program
+ Office (AJPO), Florida State University and Silicon Graphics Inc.
+ provided support for this effort -- June 21, 1995.
+
+ Derived from the DWARF 1 implementation written by Ron Guilmette
+ (rfg@netcom.com), November 1990.
+
+ This file is part of GCC.
+
+ GCC is free software; you can redistribute it and/or modify it under
+ the terms of the GNU General Public License as published by the Free
+ Software Foundation; either version 2, or (at your option) any later
+ version.
+
+ GCC is distributed in the hope that it will be useful, but WITHOUT
+ ANY WARRANTY; without even the implied warranty of MERCHANTABILITY
+ or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
+ License for more details.
+
+ You should have received a copy of the GNU General Public License
+ along with GCC; see the file COPYING. If not, write to the Free
+ Software Foundation, 59 Temple Place - Suite 330, Boston, MA
+ 02111-1307, USA. */
+
+/* This file is derived from the DWARF specification (a public document)
+ Revision 2.0.0 (July 27, 1993) developed by the UNIX International
+ Programming Languages Special Interest Group (UI/PLSIG) and distributed
+ by UNIX International. Copies of this specification are available from
+ UNIX International, 20 Waterview Boulevard, Parsippany, NJ, 07054.
+
+ This file also now contains definitions from the DWARF 3 specification. */
+
+/* This file is shared between GCC and GDB, and should not contain
+ prototypes. */
```

```

+
+ifndef _ELF_DWARF2_H
+#define _ELF_DWARF2_H
+
+/* Structure found in the .debug_line section. */
+typedef struct
+{
+ unsigned char li_length [4];
+ unsigned char li_version [2];
+ unsigned char li_prologue_length [4];
+ unsigned char li_min_insn_length [1];
+ unsigned char li_default_is_stmt [1];
+ unsigned char li_line_base [1];
+ unsigned char li_line_range [1];
+ unsigned char li_opcode_base [1];
+}
+DWARF2_External_LineInfo;
+
+typedef struct
+{
+ unsigned long li_length;
+ unsigned short li_version;
+ unsigned int li_prologue_length;
+ unsigned char li_min_insn_length;
+ unsigned char li_default_is_stmt;
+ int li_line_base;
+ unsigned char li_line_range;
+ unsigned char li_opcode_base;
+}
+DWARF2_Internal_LineInfo;
+
+/* Structure found in .debug_pubnames section. */
+typedef struct
+{
+ unsigned char pn_length [4];
+ unsigned char pn_version [2];
+ unsigned char pn_offset [4];
+ unsigned char pn_size [4];
+}
+DWARF2_External_PubNames;
+
+typedef struct
+{
+ unsigned long pn_length;
+ unsigned short pn_version;
+ unsigned long pn_offset;
+ unsigned long pn_size;
+}
+DWARF2_Internal_PubNames;
+
+/* Structure found in .debug_info section. */

```

```

+typedef struct
+{
+ unsigned char cu_length [4];
+ unsigned char cu_version [2];
+ unsigned char cu_abbrev_offset [4];
+ unsigned char cu_pointer_size [1];
+}
+DWARF2_External_CompUnit;
+
+typedef struct
+{
+ unsigned long cu_length;
+ unsigned short cu_version;
+ unsigned long cu_abbrev_offset;
+ unsigned char cu_pointer_size;
+}
+DWARF2_Internal_CompUnit;
+
+typedef struct
+{
+ unsigned char ar_length [4];
+ unsigned char ar_version [2];
+ unsigned char ar_info_offset [4];
+ unsigned char ar_pointer_size [1];
+ unsigned char ar_segment_size [1];
+}
+DWARF2_External_ARange;
+
+typedef struct
+{
+ unsigned long ar_length;
+ unsigned short ar_version;
+ unsigned long ar_info_offset;
+ unsigned char ar_pointer_size;
+ unsigned char ar_segment_size;
+}
+DWARF2_Internal_ARange;
+
+
+/* Tag names and codes. */
+enum dwarf_tag
+ {
+ DW_TAG_padding = 0x00,
+ DW_TAG_array_type = 0x01,
+ DW_TAG_class_type = 0x02,
+ DW_TAG_entry_point = 0x03,
+ DW_TAG_enumeration_type = 0x04,
+ DW_TAG_formal_parameter = 0x05,
+ DW_TAG_imported_declaration = 0x08,
+ DW_TAG_label = 0x0a,
+ DW_TAG_lexical_block = 0x0b,

```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ DW_TAG_member = 0x0d,  
+ DW_TAG_pointer_type = 0x0f,  
+ DW_TAG_reference_type = 0x10,  
+ DW_TAG_compile_unit = 0x11,  
+ DW_TAG_string_type = 0x12,  
+ DW_TAG_structure_type = 0x13,  
+ DW_TAG_subroutine_type = 0x15,  
+ DW_TAG_typedef = 0x16,  
+ DW_TAG_union_type = 0x17,  
+ DW_TAG_unspecified_parameters = 0x18,  
+ DW_TAG_variant = 0x19,  
+ DW_TAG_common_block = 0x1a,  
+ DW_TAG_common_inclusion = 0x1b,  
+ DW_TAG_inheritance = 0x1c,  
+ DW_TAG_inlined_subroutine = 0x1d,  
+ DW_TAG_module = 0x1e,  
+ DW_TAG_ptr_to_member_type = 0x1f,  
+ DW_TAG_set_type = 0x20,  
+ DW_TAG_subrange_type = 0x21,  
+ DW_TAG_with_stmt = 0x22,  
+ DW_TAG_access_declaration = 0x23,  
+ DW_TAG_base_type = 0x24,  
+ DW_TAG_catch_block = 0x25,  
+ DW_TAG_const_type = 0x26,  
+ DW_TAG_constant = 0x27,  
+ DW_TAG_enumerator = 0x28,  
+ DW_TAG_file_type = 0x29,  
+ DW_TAG_friend = 0x2a,  
+ DW_TAG_namelist = 0x2b,  
+ DW_TAG_namelist_item = 0x2c,  
+ DW_TAG_packed_type = 0x2d,  
+ DW_TAG_subprogram = 0x2e,  
+ DW_TAG_template_type_param = 0x2f,  
+ DW_TAG_template_value_param = 0x30,  
+ DW_TAG_thrown_type = 0x31,  
+ DW_TAG_try_block = 0x32,  
+ DW_TAG_variant_part = 0x33,  
+ DW_TAG_variable = 0x34,  
+ DW_TAG_volatile_type = 0x35,  
+ /* DWARF 3. */  
+ DW_TAG_dwarf_procedure = 0x36,  
+ DW_TAG_restrict_type = 0x37,  
+ DW_TAG_interface_type = 0x38,  
+ DW_TAG_namespace = 0x39,  
+ DW_TAG_imported_module = 0x3a,  
+ DW_TAG_unspecified_type = 0x3b,  
+ DW_TAG_partial_unit = 0x3c,  
+ DW_TAG_imported_unit = 0x3d,  
+ /* SGI/MIPS Extensions. */  
+ DW_TAG_MIPS_loop = 0x4081,  
+ /* HP extensions. See: ftp://ftp.hp.com/pub/lang/tools/WDB/wdb-4.0.tar.gz . */
```

Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ DW_TAG_HP_array_descriptor = 0x4090,
+ /* GNU extensions. */
+ DW_TAG_format_label = 0x4101, /* For FORTRAN 77 and Fortran 90. */
+ DW_TAG_function_template = 0x4102, /* For C++. */
+ DW_TAG_class_template = 0x4103, /* For C++. */
+ DW_TAG_GNU_BINCL = 0x4104,
+ DW_TAG_GNU_EINCL = 0x4105,
+ /* Extensions for UPC. See: http://upc.gwu.edu/~upc. */
+ DW_TAG_upc_shared_type = 0x8765,
+ DW_TAG_upc_strict_type = 0x8766,
+ DW_TAG_upc_relaxed_type = 0x8767,
+ /* PGI (STMicroelectronics) extensions. No documentation available. */
+ DW_TAG_PGI_kanji_type = 0xA000,
+ DW_TAG_PGI_interface_block = 0xA020
+ };
+
+#define DW_TAG_lo_user 0x4080
+#define DW_TAG_hi_user 0xffff
+
+/* Flag that tells whether entry has a child or not. */
+#define DW_children_no 0
+#define DW_children_yes 1
+
+/* Form names and codes. */
+enum dwarf_form
+ {
+ DW_FORM_addr = 0x01,
+ DW_FORM_block2 = 0x03,
+ DW_FORM_block4 = 0x04,
+ DW_FORM_data2 = 0x05,
+ DW_FORM_data4 = 0x06,
+ DW_FORM_data8 = 0x07,
+ DW_FORM_string = 0x08,
+ DW_FORM_block = 0x09,
+ DW_FORM_block1 = 0x0a,
+ DW_FORM_data1 = 0x0b,
+ DW_FORM_flag = 0x0c,
+ DW_FORM_sdata = 0x0d,
+ DW_FORM_strp = 0x0e,
+ DW_FORM_udata = 0x0f,
+ DW_FORM_ref_addr = 0x10,
+ DW_FORM_ref1 = 0x11,
+ DW_FORM_ref2 = 0x12,
+ DW_FORM_ref4 = 0x13,
+ DW_FORM_ref8 = 0x14,
+ DW_FORM_ref_udata = 0x15,
+ DW_FORM_indirect = 0x16
+ };
+
+/* Attribute names and codes. */
+enum dwarf_attribute
```

```
+ {  
+ DW_AT_sibling = 0x01,  
+ DW_AT_location = 0x02,  
+ DW_AT_name = 0x03,  
+ DW_AT_ordering = 0x09,  
+ DW_AT_subscr_data = 0x0a,  
+ DW_AT_byte_size = 0x0b,  
+ DW_AT_bit_offset = 0x0c,  
+ DW_AT_bit_size = 0x0d,  
+ DW_AT_element_list = 0x0f,  
+ DW_AT_stmt_list = 0x10,  
+ DW_AT_low_pc = 0x11,  
+ DW_AT_high_pc = 0x12,  
+ DW_AT_language = 0x13,  
+ DW_AT_member = 0x14,  
+ DW_AT_discr = 0x15,  
+ DW_AT_discr_value = 0x16,  
+ DW_AT_visibility = 0x17,  
+ DW_AT_import = 0x18,  
+ DW_AT_string_length = 0x19,  
+ DW_AT_common_reference = 0x1a,  
+ DW_AT_comp_dir = 0x1b,  
+ DW_AT_const_value = 0x1c,  
+ DW_AT_containing_type = 0x1d,  
+ DW_AT_default_value = 0x1e,  
+ DW_AT_inline = 0x20,  
+ DW_AT_is_optional = 0x21,  
+ DW_AT_lower_bound = 0x22,  
+ DW_AT_producer = 0x25,  
+ DW_AT_prototyped = 0x27,  
+ DW_AT_return_addr = 0x2a,  
+ DW_AT_start_scope = 0x2c,  
+ DW_AT_stride_size = 0x2e,  
+ DW_AT_upper_bound = 0x2f,  
+ DW_AT_abstract_origin = 0x31,  
+ DW_AT_accessibility = 0x32,  
+ DW_AT_address_class = 0x33,  
+ DW_AT_artificial = 0x34,  
+ DW_AT_base_types = 0x35,  
+ DW_AT_calling_convention = 0x36,  
+ DW_AT_count = 0x37,  
+ DW_AT_data_member_location = 0x38,  
+ DW_AT_decl_column = 0x39,  
+ DW_AT_decl_file = 0x3a,  
+ DW_AT_decl_line = 0x3b,  
+ DW_AT_declaration = 0x3c,  
+ DW_AT_discr_list = 0x3d,  
+ DW_AT_encoding = 0x3e,  
+ DW_AT_external = 0x3f,  
+ DW_AT_frame_base = 0x40,  
+ DW_AT_friend = 0x41,
```

```

+ DW_AT_identifier_case = 0x42,
+ DW_AT_macro_info = 0x43,
+ DW_AT_namelist_items = 0x44,
+ DW_AT_priority = 0x45,
+ DW_AT_segment = 0x46,
+ DW_AT_specification = 0x47,
+ DW_AT_static_link = 0x48,
+ DW_AT_type = 0x49,
+ DW_AT_use_location = 0x4a,
+ DW_AT_variable_parameter = 0x4b,
+ DW_AT_virtuality = 0x4c,
+ DW_AT_vtable_elem_location = 0x4d,
+ /* DWARF 3 values. */
+ DW_AT_allocated = 0x4e,
+ DW_AT_associated = 0x4f,
+ DW_AT_data_location = 0x50,
+ DW_AT_stride = 0x51,
+ DW_AT_entry_pc = 0x52,
+ DW_AT_use_UTF8 = 0x53,
+ DW_AT_extension = 0x54,
+ DW_AT_ranges = 0x55,
+ DW_AT_trampoline = 0x56,
+ DW_AT_call_column = 0x57,
+ DW_AT_call_file = 0x58,
+ DW_AT_call_line = 0x59,
+ /* SGI/MIPS extensions. */
+ DW_AT_MIPS_fde = 0x2001,
+ DW_AT_MIPS_loop_begin = 0x2002,
+ DW_AT_MIPS_tail_loop_begin = 0x2003,
+ DW_AT_MIPS_epilog_begin = 0x2004,
+ DW_AT_MIPS_loop_unroll_factor = 0x2005,
+ DW_AT_MIPS_software_pipeline_depth = 0x2006,
+ DW_AT_MIPS_linkage_name = 0x2007,
+ DW_AT_MIPS_stride = 0x2008,
+ DW_AT_MIPS_abstract_name = 0x2009,
+ DW_AT_MIPS_clone_origin = 0x200a,
+ DW_AT_MIPS_has_inlines = 0x200b,
+ /* HP extensions. */
+ DW_AT_HP_block_index = 0x2000,
+ DW_AT_HP_unmodifiable = 0x2001, /* Same as DW_AT_MIPS_fde. */
+ DW_AT_HP_actuals_stmt_list = 0x2010,
+ DW_AT_HP_proc_per_section = 0x2011,
+ DW_AT_HP_raw_data_ptr = 0x2012,
+ DW_AT_HP_pass_by_reference = 0x2013,
+ DW_AT_HP_opt_level = 0x2014,
+ DW_AT_HP_prof_version_id = 0x2015,
+ DW_AT_HP_opt_flags = 0x2016,
+ DW_AT_HP_cold_region_low_pc = 0x2017,
+ DW_AT_HP_cold_region_high_pc = 0x2018,
+ DW_AT_HP_all_variables_modifiable = 0x2019,
+ DW_AT_HP_linkage_name = 0x201a,

```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ DW_AT_HP_prof_flags = 0x201b, /* In comp unit of procs_info for -g. */
+ /* GNU extensions. */
+ DW_AT_sf_names = 0x2101,
+ DW_AT_src_info = 0x2102,
+ DW_AT_mac_info = 0x2103,
+ DW_AT_src_coords = 0x2104,
+ DW_AT_body_begin = 0x2105,
+ DW_AT_body_end = 0x2106,
+ DW_AT_GNU_vector = 0x2107,
+ /* VMS extensions. */
+ DW_AT_VMS_rtnbeg_pd_address = 0x2201,
+ /* UPC extension. */
+ DW_AT_upc_threads_scaled = 0x3210,
+ /* PGI (STMicroelectronics) extensions. */
+ DW_AT_PGI_lbase = 0x3a00,
+ DW_AT_PGI_soffset = 0x3a01,
+ DW_AT_PGI_lstride = 0x3a02
+ };
+
+#define DW_AT_lo_user 0x2000 /* Implementation-defined range start. */
+#define DW_AT_hi_user 0x3ff0 /* Implementation-defined range end. */
+
+/* Location atom names and codes. */
+enum dwarf_location_atom
+ {
+ DW_OP_addr = 0x03,
+ DW_OP_deref = 0x06,
+ DW_OP_const1u = 0x08,
+ DW_OP_const1s = 0x09,
+ DW_OP_const2u = 0x0a,
+ DW_OP_const2s = 0x0b,
+ DW_OP_const4u = 0x0c,
+ DW_OP_const4s = 0x0d,
+ DW_OP_const8u = 0x0e,
+ DW_OP_const8s = 0x0f,
+ DW_OP_constu = 0x10,
+ DW_OP_consts = 0x11,
+ DW_OP_dup = 0x12,
+ DW_OP_drop = 0x13,
+ DW_OP_over = 0x14,
+ DW_OP_pick = 0x15,
+ DW_OP_swap = 0x16,
+ DW_OP_rot = 0x17,
+ DW_OP_xderef = 0x18,
+ DW_OP_abs = 0x19,
+ DW_OP_and = 0x1a,
+ DW_OP_div = 0x1b,
+ DW_OP_minus = 0x1c,
+ DW_OP_mod = 0x1d,
+ DW_OP_mul = 0x1e,
+ DW_OP_neg = 0x1f,
```

```
+ DW_OP_not = 0x20,  
+ DW_OP_or = 0x21,  
+ DW_OP_plus = 0x22,  
+ DW_OP_plus_uconst = 0x23,  
+ DW_OP_shl = 0x24,  
+ DW_OP_shr = 0x25,  
+ DW_OP_shra = 0x26,  
+ DW_OP_xor = 0x27,  
+ DW_OP_bra = 0x28,  
+ DW_OP_eq = 0x29,  
+ DW_OP_ge = 0x2a,  
+ DW_OP_gt = 0x2b,  
+ DW_OP_le = 0x2c,  
+ DW_OP_lt = 0x2d,  
+ DW_OP_ne = 0x2e,  
+ DW_OP_skip = 0x2f,  
+ DW_OP_lit0 = 0x30,  
+ DW_OP_lit1 = 0x31,  
+ DW_OP_lit2 = 0x32,  
+ DW_OP_lit3 = 0x33,  
+ DW_OP_lit4 = 0x34,  
+ DW_OP_lit5 = 0x35,  
+ DW_OP_lit6 = 0x36,  
+ DW_OP_lit7 = 0x37,  
+ DW_OP_lit8 = 0x38,  
+ DW_OP_lit9 = 0x39,  
+ DW_OP_lit10 = 0x3a,  
+ DW_OP_lit11 = 0x3b,  
+ DW_OP_lit12 = 0x3c,  
+ DW_OP_lit13 = 0x3d,  
+ DW_OP_lit14 = 0x3e,  
+ DW_OP_lit15 = 0x3f,  
+ DW_OP_lit16 = 0x40,  
+ DW_OP_lit17 = 0x41,  
+ DW_OP_lit18 = 0x42,  
+ DW_OP_lit19 = 0x43,  
+ DW_OP_lit20 = 0x44,  
+ DW_OP_lit21 = 0x45,  
+ DW_OP_lit22 = 0x46,  
+ DW_OP_lit23 = 0x47,  
+ DW_OP_lit24 = 0x48,  
+ DW_OP_lit25 = 0x49,  
+ DW_OP_lit26 = 0x4a,  
+ DW_OP_lit27 = 0x4b,  
+ DW_OP_lit28 = 0x4c,  
+ DW_OP_lit29 = 0x4d,  
+ DW_OP_lit30 = 0x4e,  
+ DW_OP_lit31 = 0x4f,  
+ DW_OP_reg0 = 0x50,  
+ DW_OP_reg1 = 0x51,  
+ DW_OP_reg2 = 0x52,
```

```
+ DW_OP_reg3 = 0x53,  
+ DW_OP_reg4 = 0x54,  
+ DW_OP_reg5 = 0x55,  
+ DW_OP_reg6 = 0x56,  
+ DW_OP_reg7 = 0x57,  
+ DW_OP_reg8 = 0x58,  
+ DW_OP_reg9 = 0x59,  
+ DW_OP_reg10 = 0x5a,  
+ DW_OP_reg11 = 0x5b,  
+ DW_OP_reg12 = 0x5c,  
+ DW_OP_reg13 = 0x5d,  
+ DW_OP_reg14 = 0x5e,  
+ DW_OP_reg15 = 0x5f,  
+ DW_OP_reg16 = 0x60,  
+ DW_OP_reg17 = 0x61,  
+ DW_OP_reg18 = 0x62,  
+ DW_OP_reg19 = 0x63,  
+ DW_OP_reg20 = 0x64,  
+ DW_OP_reg21 = 0x65,  
+ DW_OP_reg22 = 0x66,  
+ DW_OP_reg23 = 0x67,  
+ DW_OP_reg24 = 0x68,  
+ DW_OP_reg25 = 0x69,  
+ DW_OP_reg26 = 0x6a,  
+ DW_OP_reg27 = 0x6b,  
+ DW_OP_reg28 = 0x6c,  
+ DW_OP_reg29 = 0x6d,  
+ DW_OP_reg30 = 0x6e,  
+ DW_OP_reg31 = 0x6f,  
+ DW_OP_breg0 = 0x70,  
+ DW_OP_breg1 = 0x71,  
+ DW_OP_breg2 = 0x72,  
+ DW_OP_breg3 = 0x73,  
+ DW_OP_breg4 = 0x74,  
+ DW_OP_breg5 = 0x75,  
+ DW_OP_breg6 = 0x76,  
+ DW_OP_breg7 = 0x77,  
+ DW_OP_breg8 = 0x78,  
+ DW_OP_breg9 = 0x79,  
+ DW_OP_breg10 = 0x7a,  
+ DW_OP_breg11 = 0x7b,  
+ DW_OP_breg12 = 0x7c,  
+ DW_OP_breg13 = 0x7d,  
+ DW_OP_breg14 = 0x7e,  
+ DW_OP_breg15 = 0x7f,  
+ DW_OP_breg16 = 0x80,  
+ DW_OP_breg17 = 0x81,  
+ DW_OP_breg18 = 0x82,  
+ DW_OP_breg19 = 0x83,  
+ DW_OP_breg20 = 0x84,  
+ DW_OP_breg21 = 0x85,
```

```

+ DW_OP_breg22 = 0x86,
+ DW_OP_breg23 = 0x87,
+ DW_OP_breg24 = 0x88,
+ DW_OP_breg25 = 0x89,
+ DW_OP_breg26 = 0x8a,
+ DW_OP_breg27 = 0x8b,
+ DW_OP_breg28 = 0x8c,
+ DW_OP_breg29 = 0x8d,
+ DW_OP_breg30 = 0x8e,
+ DW_OP_breg31 = 0x8f,
+ DW_OP_regx = 0x90,
+ DW_OP_fbreg = 0x91,
+ DW_OP_bregx = 0x92,
+ DW_OP_piece = 0x93,
+ DW_OP_deref_size = 0x94,
+ DW_OP_xderef_size = 0x95,
+ DW_OP_nop = 0x96,
+ /* DWARF 3 extensions. */
+ DW_OP_push_object_address = 0x97,
+ DW_OP_call2 = 0x98,
+ DW_OP_call4 = 0x99,
+ DW_OP_call_ref = 0x9a,
+ /* GNU extensions. */
+ DW_OP_GNU_push_tls_address = 0xe0,
+ /* HP extensions. */
+ DW_OP_HP_unknown = 0xe0, /* Ouch, the same as GNU_push_tls_address. */
+ DW_OP_HP_is_value = 0xe1,
+ DW_OP_HPfltconst4 = 0xe2,
+ DW_OP_HPfltconst8 = 0xe3,
+ DW_OP_HP_mod_range = 0xe4,
+ DW_OP_HP_unmod_range = 0xe5,
+ DW_OP_HP_tls = 0xe6
+ };
+
+#define DW_OP_lo_user 0xe0 /* Implementation-defined range start. */
+#define DW_OP_hi_user 0xff /* Implementation-defined range end. */
+
+ /* Type encodings. */
+enum dwarf_type
+ {
+ DW_ATE_void = 0x0,
+ DW_ATE_address = 0x1,
+ DW_ATE_boolean = 0x2,
+ DW_ATE_complex_float = 0x3,
+ DW_ATE_float = 0x4,
+ DW_ATE_signed = 0x5,
+ DW_ATE_signed_char = 0x6,
+ DW_ATE_unsigned = 0x7,
+ DW_ATE_unsigned_char = 0x8,
+ /* DWARF 3. */
+ DW_ATE_imaginary_float = 0x9,

```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ /* HP extensions. */
+ DW_ATE_HP_float80 = 0x80, /* Floating-point (80 bit). */
+ DW_ATE_HP_complex_float80 = 0x81, /* Complex floating-point (80 bit). */
+ DW_ATE_HP_float128 = 0x82, /* Floating-point (128 bit). */
+ DW_ATE_HP_complex_float128 = 0x83, /* Complex floating-point (128 bit). */
+ DW_ATE_HP_floatahpintel = 0x84, /* Floating-point (82 bit IA64). */
+ DW_ATE_HP_imaginary_float80 = 0x85,
+ DW_ATE_HP_imaginary_float128 = 0x86
+ };
+
+ #define DW_ATE_lo_user 0x80
+ #define DW_ATE_hi_user 0xff
+
+ /* Array ordering names and codes. */
+ enum dwarf_array_dim_ordering
+ {
+ DW_ORD_row_major = 0,
+ DW_ORD_col_major = 1
+ };
+
+ /* Access attribute. */
+ enum dwarf_access_attribute
+ {
+ DW_ACCESS_public = 1,
+ DW_ACCESS_protected = 2,
+ DW_ACCESS_private = 3
+ };
+
+ /* Visibility. */
+ enum dwarf_visibility_attribute
+ {
+ DW_VIS_local = 1,
+ DW_VIS_exported = 2,
+ DW_VIS_qualified = 3
+ };
+
+ /* Virtuality. */
+ enum dwarf_virtuality_attribute
+ {
+ DW_VIRTUALITY_none = 0,
+ DW_VIRTUALITY_virtual = 1,
+ DW_VIRTUALITY_pure_virtual = 2
+ };
+
+ /* Case sensitivity. */
+ enum dwarf_id_case
+ {
+ DW_ID_case_sensitive = 0,
+ DW_ID_up_case = 1,
+ DW_ID_down_case = 2,
+ DW_ID_case_insensitive = 3
```



```

+ DW_LNE_end_sequence = 1,
+ DW_LNE_set_address = 2,
+ DW_LNE_define_file = 3,
+ /* HP extensions. */
+ DW_LNE_HP_negate_is_UV_update = 0x11,
+ DW_LNE_HP_push_context = 0x12,
+ DW_LNE_HP_pop_context = 0x13,
+ DW_LNE_HP_set_file_line_column = 0x14,
+ DW_LNE_HP_set_routine_name = 0x15,
+ DW_LNE_HP_set_sequence = 0x16,
+ DW_LNE_HP_negate_post_semantics = 0x17,
+ DW_LNE_HP_negate_function_exit = 0x18,
+ DW_LNE_HP_negate_front_end_logical = 0x19,
+ DW_LNE_HP_define_proc = 0x20
+ };
+
+ /* Call frame information. */
+enum dwarf_call_frame_info
+ {
+ DW_CFA_advance_loc = 0x40,
+ DW_CFA_offset = 0x80,
+ DW_CFA_restore = 0xc0,
+ DW_CFA_nop = 0x00,
+ DW_CFA_set_loc = 0x01,
+ DW_CFA_advance_loc1 = 0x02,
+ DW_CFA_advance_loc2 = 0x03,
+ DW_CFA_advance_loc4 = 0x04,
+ DW_CFA_offset_extended = 0x05,
+ DW_CFA_restore_extended = 0x06,
+ DW_CFA_undefined = 0x07,
+ DW_CFA_same_value = 0x08,
+ DW_CFA_register = 0x09,
+ DW_CFA_remember_state = 0x0a,
+ DW_CFA_restore_state = 0x0b,
+ DW_CFA_def_cfa = 0x0c,
+ DW_CFA_def_cfa_register = 0x0d,
+ DW_CFA_def_cfa_offset = 0x0e,
+ /* DWARF 3. */
+ DW_CFA_def_cfa_expression = 0x0f,
+ DW_CFA_expression = 0x10,
+ DW_CFA_offset_extended_sf = 0x11,
+ DW_CFA_def_cfa_sf = 0x12,
+ DW_CFA_def_cfa_offset_sf = 0x13,
+ /* SGI/MIPS specific. */
+ DW_CFA_MIPS_advance_loc8 = 0x1d,
+ /* GNU extensions. */
+ DW_CFA_GNU_window_save = 0x2d,
+ DW_CFA_GNU_args_size = 0x2e,
+ DW_CFA_GNU_negative_offset_extended = 0x2f
+ };
+

```

## Linux–Kernel: [PATCH] Kgdb dwarf2 for asm

```
+ #define DW_CIE_ID 0xffffffff
+ #define DW_CIE_VERSION 1
+
+ #define DW_CFA_extended 0
+ #define DW_CFA_lo_user 0x1c
+ #define DW_CFA_hi_user 0x3f
+
+ #define DW_CHILDREN_no 0x00
+ #define DW_CHILDREN_yes 0x01
+
+ #define DW_ADDR_none 0
+
+ /* Source language names and codes. */
+ enum dwarf_source_language
+ {
+   DW_LANG_C89 = 0x0001,
+   DW_LANG_C = 0x0002,
+   DW_LANG_Ada83 = 0x0003,
+   DW_LANG_C_plus_plus = 0x0004,
+   DW_LANG_Cobol74 = 0x0005,
+   DW_LANG_Cobol85 = 0x0006,
+   DW_LANG_Fortran77 = 0x0007,
+   DW_LANG_Fortran90 = 0x0008,
+   DW_LANG_Pascal83 = 0x0009,
+   DW_LANG_Modula2 = 0x000a,
+   DW_LANG_Java = 0x000b,
+   /* DWARF 3. */
+   DW_LANG_C99 = 0x000c,
+   DW_LANG_Ada95 = 0x000d,
+   DW_LANG_Fortran95 = 0x000e,
+   /* MIPS. */
+   DW_LANG_Mips_Assembler = 0x8001,
+   /* UPC. */
+   DW_LANG_Upc = 0x8765
+ };
+
+ #define DW_LANG_lo_user 0x8000 /* Implementation–defined range start. */
+ #define DW_LANG_hi_user 0xffff /* Implementation–defined range start. */
+
+ /* Names and codes for macro information. */
+ enum dwarf_macinfo_record_type
+ {
+   DW_MACINFO_define = 1,
+   DW_MACINFO_undef = 2,
+   DW_MACINFO_start_file = 3,
+   DW_MACINFO_end_file = 4,
+   DW_MACINFO_vendor_ext = 255
+ };
+
+ /* @@@ For use with GNU frame unwind information. */
```

## Linux-Kernel: [PATCH] Kgdb dwarf2 for asm

```
+#define DW_EH_PE_ahsplt 0x00
+#define DW_EH_PE_omit 0xff
+
+#define DW_EH_PE_uleb128 0x01
+#define DW_EH_PE_udata2 0x02
+#define DW_EH_PE_udata4 0x03
+#define DW_EH_PE_udata8 0x04
+#define DW_EH_PE_sleb128 0x09
+#define DW_EH_PE_sdata2 0x0A
+#define DW_EH_PE_sdata4 0x0B
+#define DW_EH_PE_sdata8 0x0C
+#define DW_EH_PE_signed 0x08
+
+#define DW_EH_PE_pcrel 0x10
+#define DW_EH_PE_textrel 0x20
+#define DW_EH_PE_datarel 0x30
+#define DW_EH_PE_funcrel 0x40
+#define DW_EH_PE_aligned 0x50
+
+#define DW_EH_PE_indirect 0x80
+
+#endif /* _ELF_DWARF2_H */
Binary files linux-2.6.0-akgdb/scripts/bin2c and linux/scripts/bin2c differ
diff -urP -I '$Id:.*Exp \$$' -X /usr/src/patch.exclude linux-2.6.0-akgdb/scripts/dwarfh.awk
linux/scripts/dwarfh.awk
--- linux-2.6.0-akgdb/scripts/dwarfh.awk 1969-12-31 16:00:00.000000000 -0800
+++ linux/scripts/dwarfh.awk 2004-01-22 12:58:07.000000000 -0800
@@ -0,0 +1,19 @@
+BEGIN {
+ print "#ifndef _ELF_DWARF_H"
+ print "/* Machine generated from dwarf2.h by scripts/dwarfh.awk */"
+}
+$2 == "=" {
+ gsub(/./, "", $3)
+ print "#define " $1 "\t" $3
+}
+$1 == "#define" {
+ print $0
+ while( index($0,"\\") == length($0)){
+ getline
+ print $0
+ }
+}
+./.*/ {}
+END {
+ print "#endif"
+}
+
-
```

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

Linux–Kernel: [PATCH] Kgdb dwarf2 for asm

More majordomo info at <http://vger.kernel.org/majordomo–info.html>

Please read the FAQ at <http://www.tux.org/lkml/>