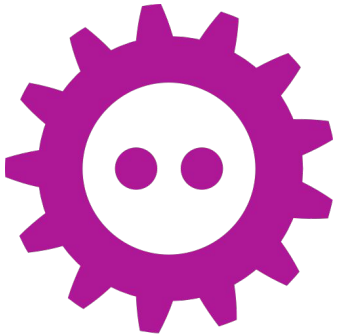
Backtraces for embedded Linux C and C++ programs



Landis+Gyr

```
root@qemuarm:~# /tmp/plain_crash
Segmentation fault
```

```
root@qemuarm:~# /tmp/unwind_crash
Backtrace (newest to oldest):
  #0 0x4317e7: segv_handler+0xd
  #1 0xb6dfa181: unknown+0x0
  #2 0x4317f6: f3+0x9
  #3 0x431805: f2+0x5
  #4 0x43180f: f1+0x5
  #5 0x431857: main+0x43
  #6 0xb6dea2cd: unknown+0x0
  #7 0xb6dea37d: __libc_start_main+0x6f
  #8 0x431469: unknown+0x0
```

Work environment

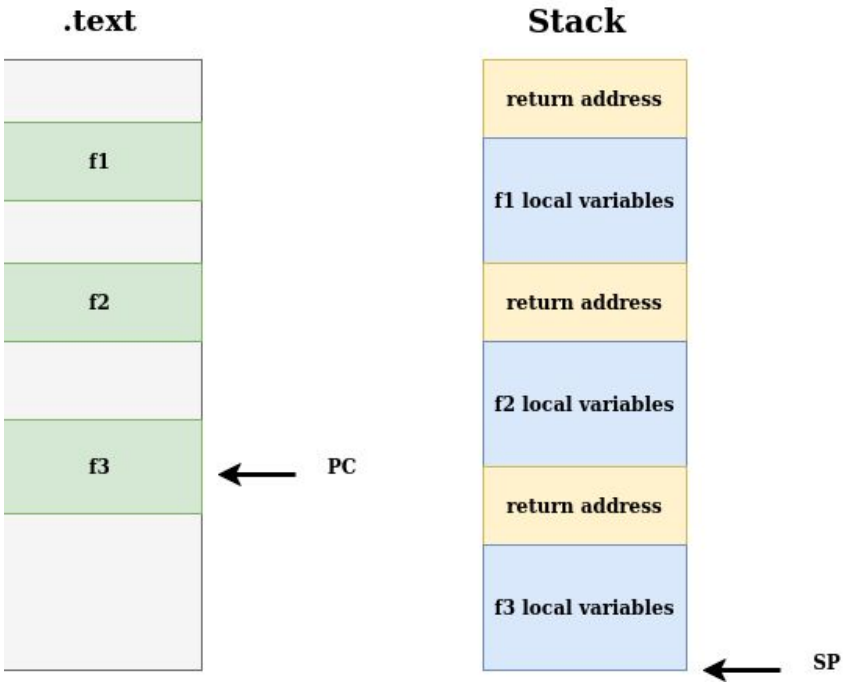
- ARMv7 architecture (Cortex-A8)
- MACHINE=qemuarm bitbake core-image-full-cmdline
- MACHINE=qemuarm runqemu core-image-full-cmdline nographic

```c
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

static void segv_handler(int sig, siginfo_t *si, void *uc)
{
        exit(1);
}

void f3(void) {
    int *t = NULL;
    *t = 3;
}

void f2(void) {
    f3();
}

void f1(void) {
    f2();
}

int main() {
        struct sigaction sa = {
                .sa_sigaction = segv_handler,
                .sa_flags = SA_SIGINFO
        };
        sigemptyset(&sa.sa_mask);
        sigaction(SIGSEGV, &sa, NULL);

        f1();

        return 0;
}
```

ARMv7 registers:

- **r0** to **r12**
- **r13** : sp
- **r14** : lr
- **r15** : pc

```
000017fe <f2>:
    17fe:       b508            push    {r3, lr}
    1800:       f7ff fff4       bl      17ec <f3>
    1804:       bf00            nop
    1806:       bd08            pop     {r3, pc}

00001808 <f1>:
    1808:       b508            push    {r3, lr}
    180a:       f7ff fff8       bl      17fe <f2>
    180e:       bf00            nop
    1810:       bd08            pop     {r3, pc}
```

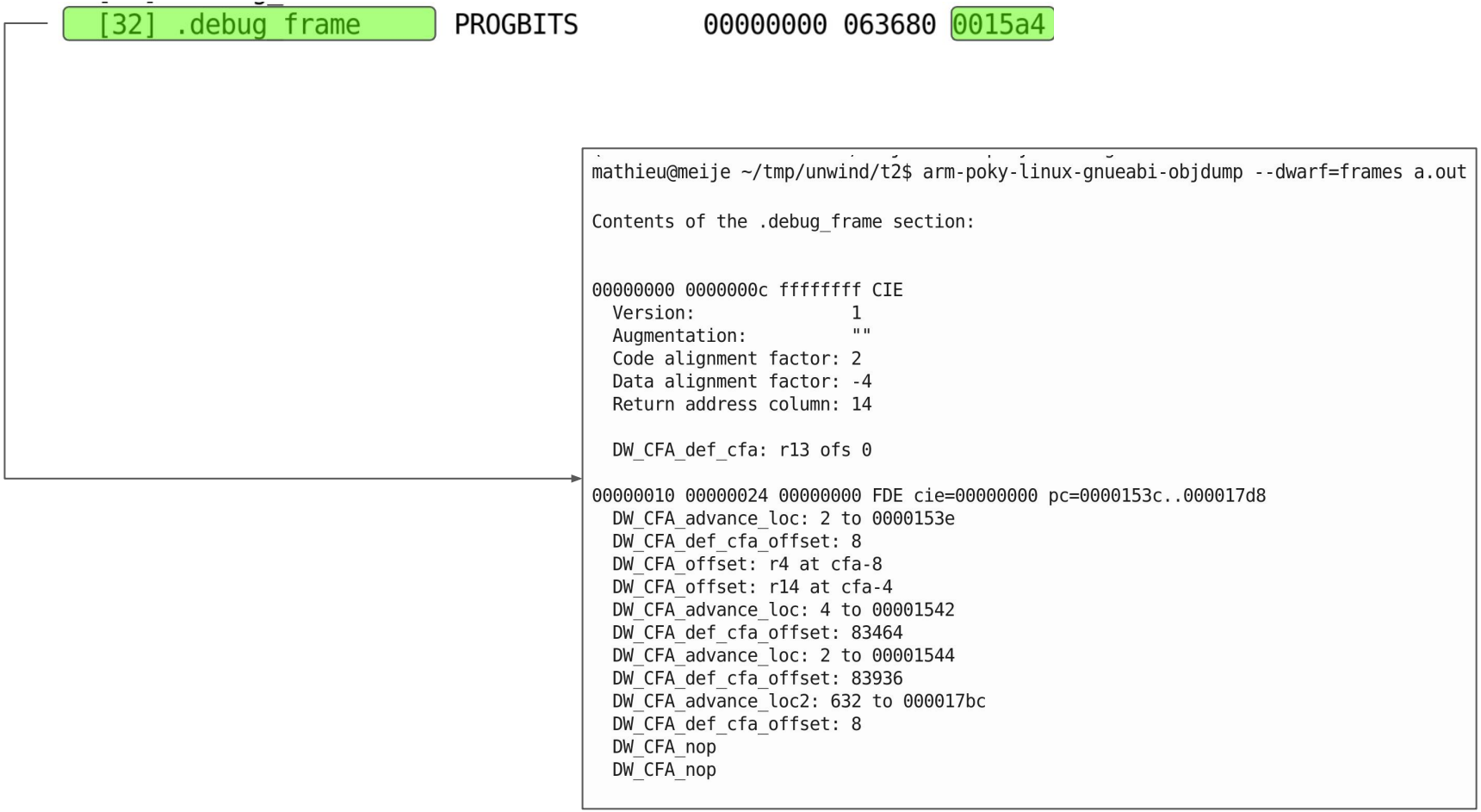



```
0x407fc8d0:     0x00412190      0x408a6ba7      0x00000003      0x0000000a
0x407fc8e0:     0x004007cc      0x40960c44      0x4096115c      0x408a7f7b
0x407fc8f0:     0x40960ba0      0x408a8245      0x00000002      0x40960ba0
0x407fc900:     0x004007cc      0x408a0f0b      0x4095fe5c      0x00000001
0x407fc910:     0xe3a070ad      0xef9000ad      0x004006b9      0x00400677
0x407fc920:     0x00410f04      0x00000000      0x004006b9      0x00400699
0x407fc930:     0x00000000      0x004006b1      0x00000000      0x004006fd
0x407fc940:     0x00400655      0x00000000      0x00000000      0x00000000
0x407fc950:     0x00000000      0x00000000      0x00000000      0x00000000
0x407fc960:     0x00000000      0x00000000      0x00000000      0x00000000
0x407fc970:     0x00000000      0x00000000      0x00000000      0x00000000
0x407fc980:     0x00000000      0x00000000      0x00000000      0x00000000
0x407fc990:     0x00000000      0x00000000      0x00000000      0x00000000
```

```
mathieu@meije ~/tmp/unwind/t2$ arm-poky-linux-gnueabi-readelf -S a.out
There are 40 section headers, starting at offset 0x85f34:

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .interp           PROGBITS        00000154 000154 000019 00   A  0   0  1
  [ 2] .note.gnu.bu[...] NOTE            00000170 000170 000024 00   A  0   0  4
  [ 3] .note.ABI-tag     NOTE            00000194 000194 000020 00   A  0   0  4

  [16] .ARM.extab        PROGBITS        0000f044 00f044 000120 00   A  0   0  4
  [17] .ARM.exidx        ARM_EXIDX       0000f164 00f164 000318 00  AL 13   0  4
  [18] .eh_frame         PROGBITS        0000f47c 00f47c 000004 00   A  0   0  4

  [28] .debug_aranges    PROGBITS        00000000 0101c8 0004e8 00      0   0  8
  [29] .debug_info       PROGBITS        00000000 0106b0 036d5d 00      0   0  1
  [30] .debug_abbrev     PROGBITS        00000000 04740d 008464 00      0   0  1
  [31] .debug_line       PROGBITS        00000000 04f871 013e0c 00      0   0  1
  [32] .debug_frame      PROGBITS        00000000 063680 0015a4 00      0   0  4
  [33] .debug_str        PROGBITS        00000000 064c24 00585c 01  MS  0   0  1
  [34] .debug_line_str   PROGBITS        00000000 06a480 0000cc 01  MS  0   0  1
  [35] .debug_loclists   PROGBITS        00000000 06a54c 015241 00      0   0  1
  [36] .debug_rnglists   PROGBITS        00000000 07f78d 002368 00      0   0  1
```

```
  [16] .ARM.extab          PROGBITS        0000f044 00f044 000120
  [17] .ARM.exidx          ARM_EXIDX       0000f164 00f164 000318
```

```
mathieu@meije ~/tmp/unwind/t2$ arm-poky-linux-gnueabi-readelf --unwind a.out

Unwind section '.ARM.exidx' at offset 0xf164 contains 99 entries:

0x1440 <_start>: 0x1 [cantunwind]

0x1888 <_Uarm_get_proc_name_by_ip>: 0x8012afb0
  Compact model index: 0
  0x12      vsp = vsp + 76
  0xaf      pop {r4, r5, r6, r7, r8, r9, r10, r11, r14}
  0xb0      finish

0x19d8 <_Uarm_get_proc_name>: 0x8002a9b0
  Compact model index: 0
  0x02      vsp = vsp + 12
  0xa9      pop {r4, r5, r14}
  0xb0      finish

0x1a28 <_Uarm_get_reg>: 0x1 [cantunwind]

0x1a78 <get_dyn_info_list_addr>: 0x80a8b0b0
  Compact model index: 0
  0xa8      pop {r4, r14}
  0xb0      finish
  0xb0      finish
```

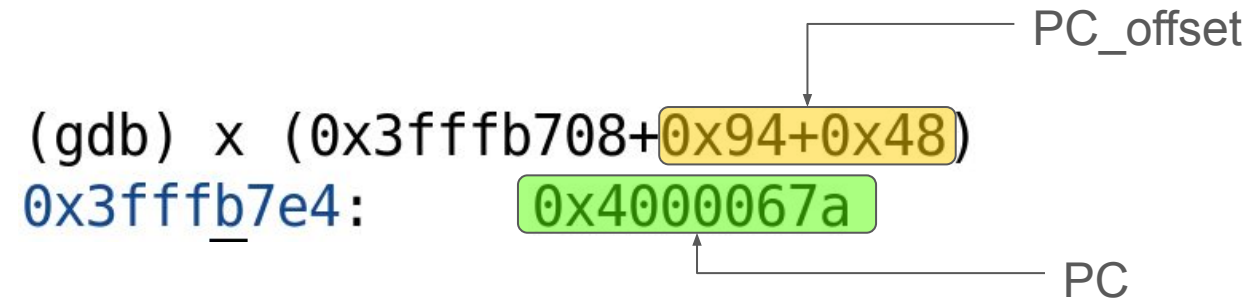

[32] .debug_frame PROGBITS 00000000 063680 0015a4
mathieu@meije ~/tmp/unwind/t2$ arm-poky-linux-gnueabi-objdump --dwarf=frames a.out
Contents of the .debug_frame section:
00000000 0000000c ffffffff CIE
Version: 1
Augmentation: ""
Code alignment factor: 2
Data alignment factor: -4
Return address column: 14
DW_CFA_def_cfa: r13 ofs 0
00000010 00000024 00000000 FDE cie=00000000 pc=0000153c..000017d8
DW_CFA_advance_loc: 2 to 0000153e
DW_CFA_def_cfa_offset: 8
DW_CFA_offset: r4 at cfa-8
DW_CFA_offset: r14 at cfa-4
DW_CFA_advance_loc: 4 to 00001542
DW_CFA_def_cfa_offset: 83464
DW_CFA_advance_loc: 2 to 00001544
DW_CFA_def_cfa_offset: 83936
DW_CFA_advance_loc2: 632 to 000017bc
DW_CFA_def_cfa_offset: 8
DW_CFA_nop
DW_CFA_nop

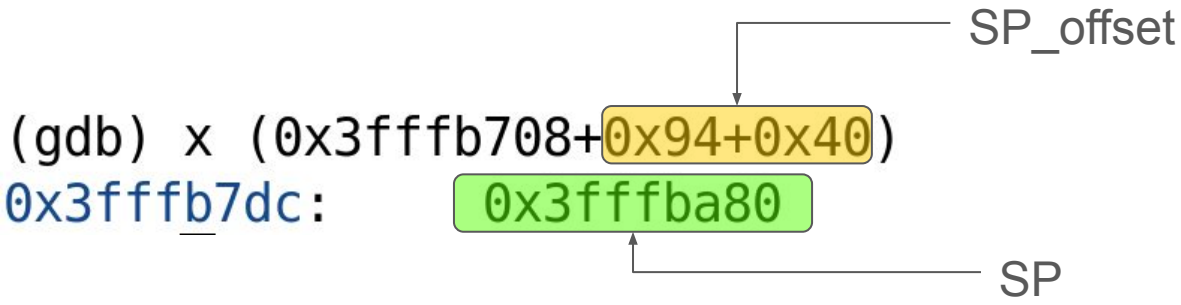
```
(gdb) info registers pc
pc             0x4000065e          0x4000065e <segv_handler+10>
```

```
00000654 <segv_handler>:
 654:   b500            push    {lr}
 656:   b085            sub     sp, #20
 658:   9003            str     r0, [sp, #12]
 65a:   9102            str     r1, [sp, #8]
 65c:   9201            str     r2, [sp, #4]
 65e:   2001            movs    r0, #1
 660:   f7ff ef62       blx     528 <exit@plt>
```

```
00000010 00000014 00000000 FDE cie=00000000 pc=00000654..00000664
  DW_CFA_advance_loc: 2 to 00000656
  DW_CFA_def_cfa_offset: 4
  DW_CFA_offset: r14 at cfa-4
  DW_CFA_advance_loc: 2 to 00000658
  DW_CFA_def_cfa_offset: 24
```

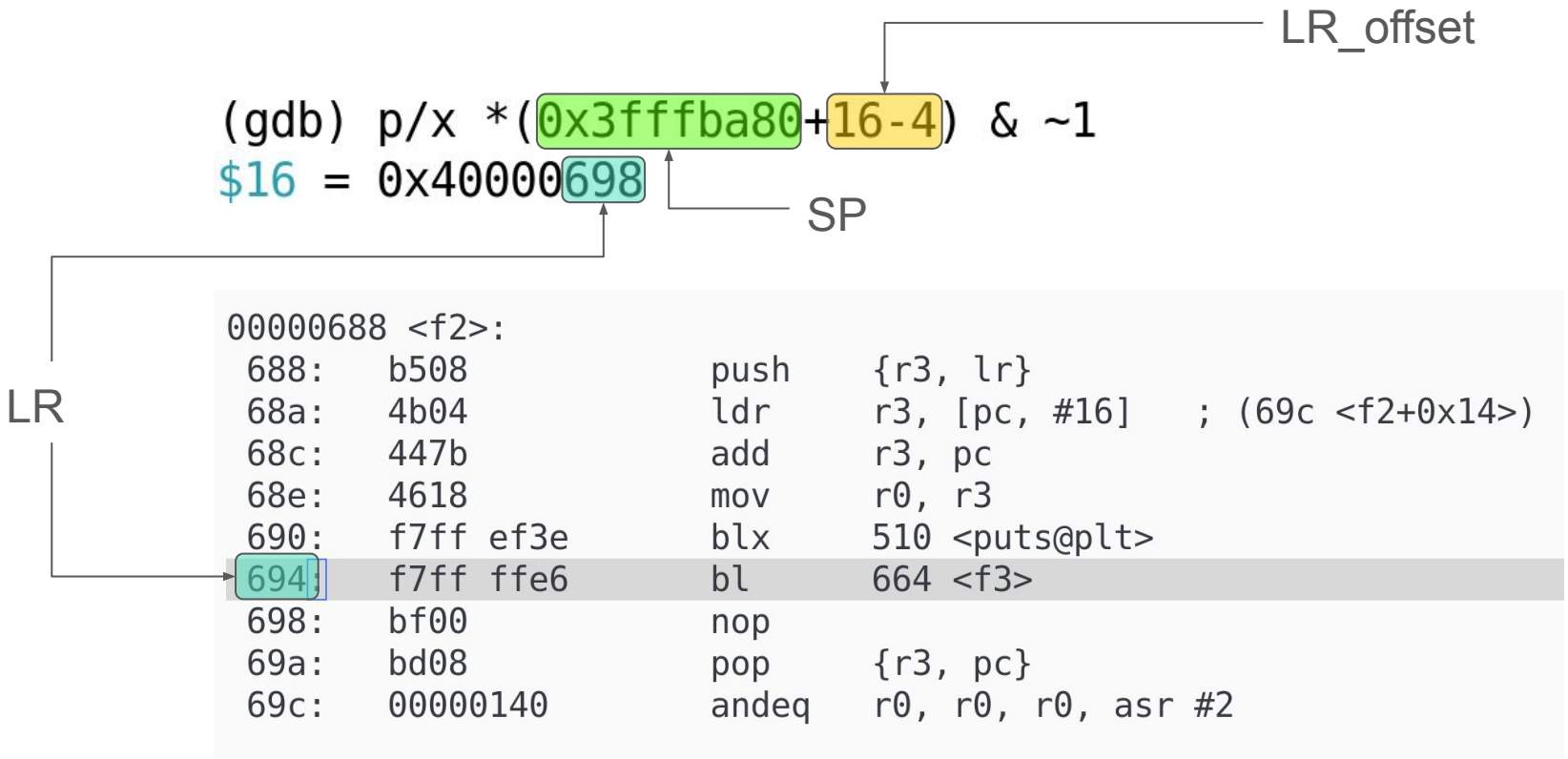

R14/LR
(gdb) x/32wx $sp
0x3fffb6f0: 0x0963cf85 0x3fffb788 0x3fffb708 0x0000000b
0x3fffb700: 0x3f7ca000 0x3f6d06a1 0x0000000b 0x00000000
0x3fffb710: 0x00000001 0x00000000 0x00000001 0x3f6aad28
0x3fffb720: 0x0000023b 0x3f6b49b8 0x3f7ca000 0x3fffb77c
0x3fffb730: 0x3fffb778 0x00000000 0x00000000 0x00000000
0x3fffb740: 0x3f6b49b8 0x3f6a8978 0x677f9a5f 0x033bfcd2
0x3fffb750: 0x3fffb778 0x3fffb77c 0x3f7ce9c6 0x3f7ce6d4
0x3fffb760: 0x3fffb804 0x3f7fdd00 0xaaaaaaab 0x3f7ca2c8
(gdb) x 0x3f6d06a0
0x3f6d06a0: 0x07adf04f
rt_sigreturn

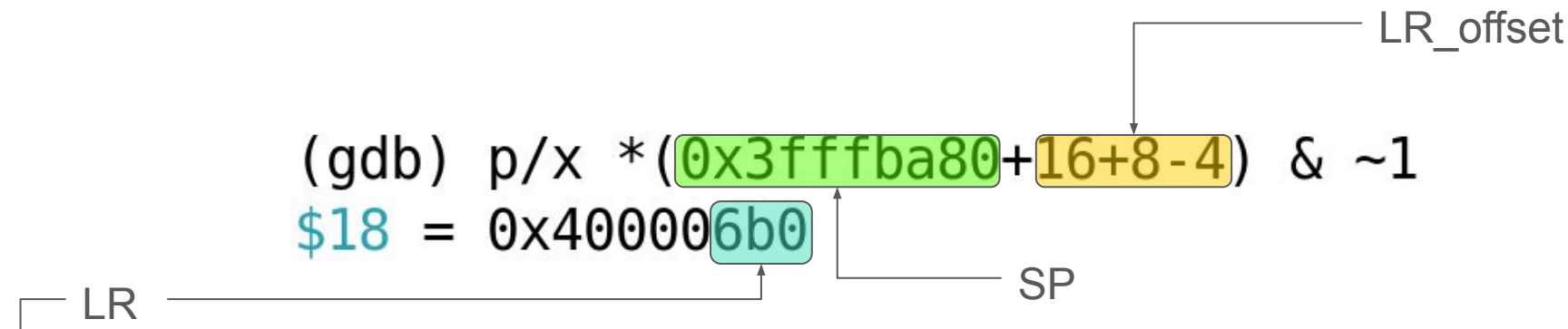

PC_offset
(gdb) x (0x3fffb708+0x94+0x48)
0x3fffb7e4: 0x4000067a
PC


SP_offset
(gdb) x (0x3fffb708+0x94+0x40)
0x3fffb7dc: 0x3fffba80
SP

```
00000664 <f3>:
 664:   b500            push    {lr}
 666:   b083            sub     sp, #12
 668:   2300            movs    r3, #0
 66a:   9301            str     r3, [sp, #4]
 66c:   4b05            ldr     r3, [pc, #20]   ; (684 <f3+0x20>)
 66e:   447b            add     r3, pc
 670:   4618            mov     r0, r3
 672:   f7ff ef4e       blx     510 <puts@plt>
 676:   9b01            ldr     r3, [sp, #4]
 678:   2203            movs    r2, #3
 67a:   601a            str     r2, [r3, #0]
 67c:   bf00            nop
 67e:   b003            add     sp, #12
 680:   f85d fb04       ldr.w   pc, [sp], #4
 684:   0000015a        andeq   r0, r0, sl, asr r1
```

```
00000028 00000018 00000000 FDE cie=00000000 pc=00000664..00000688
  DW_CFA_advance_loc: 2 to 00000666
  DW_CFA_def_cfa_offset: 4
  DW_CFA_offset: r14 at cfa-4
  DW_CFA_advance_loc: 2 to 00000668
  DW_CFA_def_cfa_offset: 16
  DW_CFA_advance_loc: 24 to 00000680
  DW_CFA_def_cfa_offset: 4
  DW_CFA_nop
```


LR_offset
(gdb) p/x *(0x3fffba80+16-4) & ~1
$16 = 0x40000698
SP
LR
00000688 <f2>:
688: b508 push {r3, lr}
68a: 4b04 ldr r3, [pc, #16] ; (69c <f2+0x14>)
68c: 447b add r3, pc
68e: 4618 mov r0, r3
690: f7ff ef3e blx 510 <puts@plt>
694: f7ff ffe6 bl 664 <f3>
698: bf00 nop
69a: bd08 pop {r3, pc}
69c: 00000140 andeq r0, r0, r0, asr #2




```
000006a0 <f1>:
 6a0:   b508            push    {r3, lr}
 6a2:   4b04            ldr     r3, [pc, #16]   ; (6b4 <f1+0x14>)
 6a4:   447b            add     r3, pc
 6a6:   4618            mov     r0, r3
 6a8:   f7ff ef32       blx     510 <puts@plt>
 6ac:   f7ff ffec       bl      688 <f2>
 6b0:   bf00            nop
 6b2:   bd08            pop     {r3, pc}
 6b4:   0000012c        andeq   r0, r0, ip, lsr #2
```

```
00000044 00000014 00000000 FDE cie=00000000 pc=00000688..000006a0
  DW_CFA_advance_loc: 2 to 0000068a
  DW_CFA_def_cfa_offset: 8
  DW_CFA_offset: r3 at cfa-8
  DW_CFA_offset: r14 at cfa-4
  DW_CFA_nop
```

```
(gdb) p/x *(0x3fffba80+16+8+8-4) & ~1
$19 = 0x400006fc
```

SP

LR_offset

LR

```
000006b8 <main>:
 6b8:   b500            push    {lr}
 6ba:   b0a5            sub     sp, #148        ; 0x94
 6bc:   4a18            ldr     r2, [pc, #96]   ; (720 <main+0x68>)
 6be:   447a            add     r2, pc
 6c0:   4b18            ldr     r3, [pc, #96]   ; (724 <main+0x6c>)
 6c2:   58d3            ldr     r3, [r2, r3]
 6c4:   681b            ldr     r3, [r3, #0]
 6c6:   9323            str     r3, [sp, #140]  ; 0x8c
 6c8:   f04f 0300       mov.w   r3, #0
 6cc:   466b            mov     r3, sp
 6ce:   228c            movs    r2, #140        ; 0x8c
 6d0:   2100            movs    r1, #0
 6d2:   4618            mov     r0, r3
 6d4:   f7ff ef2e       blx     534 <memset@plt>
 6d8:   4b13            ldr     r3, [pc, #76]   ; (728 <main+0x70>)
 6da:   447b            add     r3, pc
 6dc:   9300            str     r3, [sp, #0]
 6de:   2304            movs    r3, #4
 6e0:   9321            str     r3, [sp, #132]  ; 0x84
 6e2:   466b            mov     r3, sp
 6e4:   3304            adds    r3, #4
 6e6:   4618            mov     r0, r3
 6e8:   f7ff ef2a       blx     540 <sigemptyset@plt>
 6ec:   466b            mov     r3, sp
 6ee:   2200            movs    r2, #0
 6f0:   4619            mov     r1, r3
 6f2:   200b            movs    r0, #11
 6f4:   f7ff ef06       blx     504 <sigaction@plt>
 6f8:   f7ff ffd2       bl      6a0 <f1>
 6fc:   2300            movs    r3, #0
```

```
0000005c 00000014 00000000 FDE cie=00000000 pc=000006a0..000006b8
  DW_CFA_advance_loc: 2 to 000006a2
  DW_CFA_def_cfa_offset: 8
  DW_CFA_offset: r3 at cfa-8
  DW_CFA_offset: r14 at cfa-4
  DW_CFA_nop
```

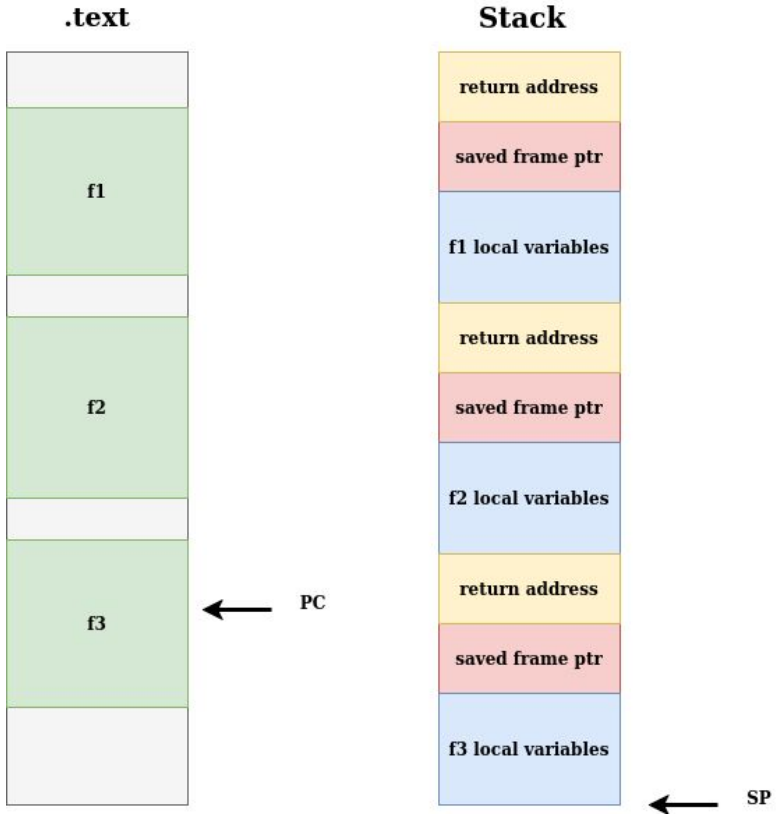
Custom backtrace

#0 **0x4000065e** in segv_handler
#1 <signal trampoline>
#2 **0x4000067a** in f3
#3 **0x40000698** in f2
#4 **0x400006b0** in f1
#5 **0x400006fc** in main

GDB backtrace

```
(gdb) bt
#0  segv_handler (sig=11, si=0x3fffb708, uc=0x3fffb788) at t1.c:8
#1  <signal handler called>
#2  0x4000067a in f3 () at t1.c:14
#3  0x40000698 in f2 () at t1.c:19
#4  0x400006b0 in f1 () at t1.c:24
#5  0x400006fc in main () at t1.c:35
```

Fast unwind with frame pointers



fp = r7 (thumb) or r11

```
00001830 <f2>:
    1830:       e52db008        str     fp, [sp, #-8]!
    1834:       e58de004        str     lr, [sp, #4]
    1838:       e28db004        add     fp, sp, #4
    183c:       ebffffef        bl      1800 <f3>
```

Fast unwind with frame pointers

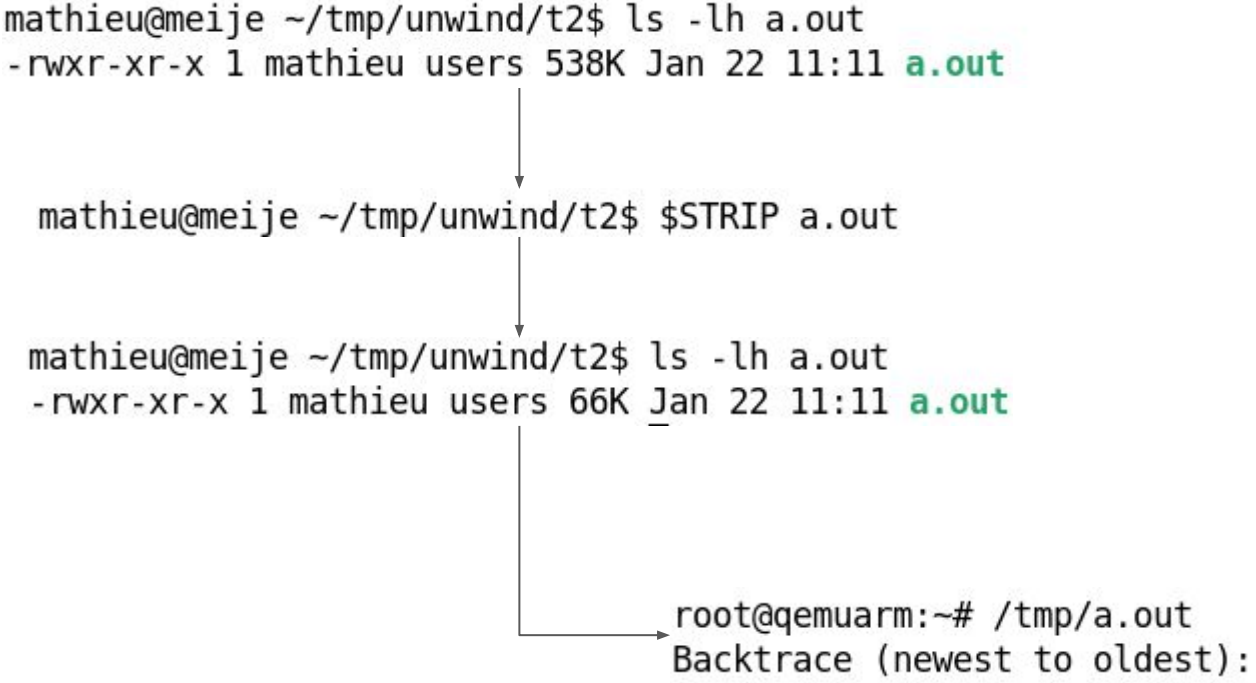
gcc-arm -mthumb -O[123] : No frame pointer
gcc-arm -mthumb -O0 : Unusable frame pointer in r7 (see gcc bugzilla)
gcc-arm -O[123] -fno-omit-frame-pointer : Frame pointer in r11
gcc-arm -O0 : Frame pointer in r11

```c
void print_backtrace(void)
{
    unw_cursor_t cursor;
    unw_context_t context;
    struct frame frames[MAX_FRAMES];
    int count = 0;

    unw_getcontext(&context);
    unw_init_local(&cursor, &context);

    while (unw_step(&cursor) > 0 && count < MAX_FRAMES) {
        unw_get_reg(&cursor, UNW_REG_IP, &frames[count].pc);
        if (unw_get_proc_name(&cursor,
                              frames[count].func_name,
                              sizeof(frames[count].func_name),
                              &frames[count].offset) != 0) {
            snprintf(frames[count].func_name,
                     sizeof(frames[count].func_name),
                     "unknown");
            frames[count].offset = 0;
        }
        count++;
    }

    printf("Backtrace (newest to oldest):\n");

    for (int i = 0; i < count; i++) {
        printf("  #%d 0x%lx: %s+0x%lx\n",
               i,
               (long)frames[i].pc,
               frames[i].func_name,
               (long)frames[i].offset);
    }
}

static void segv_handler(int sig, siginfo_t *si, void *uc)
{
        print_backtrace();
        exit(1);
}
```

```c
/* unwinding method selection support */
#define UNW_ARM_METHOD_ALL          0xFF
#define UNW_ARM_METHOD_DWARF        0x01
#define UNW_ARM_METHOD_FRAME        0x02
#define UNW_ARM_METHOD_EXIDX        0x04
```

Yocto and libunwind

```
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
```


mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 538K Jan 22 11:11 a.out
mathieu@meije ~/tmp/unwind/t2$ $STRIP a.out
mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 66K Jan 22 11:11 a.out
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):


mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 538K Jan 22 11:11 a.out
mathieu@meije ~/tmp/unwind/t2$ $STRIP --keep-section=.debug_frame a.out
mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 72K Jan 22 11:14 a.out
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
#0 0x4c17e7: unknown+0x0
#1 0xb6e07181: unknown+0x0
#2 0x4c17f6: unknown+0x0
#3 0x4c1805: unknown+0x0
#4 0x4c180f: unknown+0x0
#5 0x4c1857: unknown+0x0
#6 0xb6df72cd: unknown+0x0
#7 0xb6df737d: __libc_start_main+0x6f
#8 0x4c1469: unknown+0x0

```
mathieu@meije ~/tmp/unwind/t2$ $READELF -S a.out
There are 40 section headers, starting at offset 0x87718:

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .interp           PROGBITS        00000154 000154 000019 00   A  0   0  1
  [ 2] .note.gnu.bu[...] NOTE            00000170 000170 000024 00   A  0   0  4
  [ 3] .note.ABI-tag     NOTE            00000194 000194 000020 00   A  0   0  4
  [ 4] .gnu.hash         GNU_HASH        000001b4 0001b4 000018 04   A  5   0  4
  [ 5] .dynsym           DYNSYM          000001cc 0001cc 000490 10   A  6   3  4
  [ 6] .dynstr           STRTAB          0000065c 00065c 0003b4 00   A  0   0  1
  [ 7] .gnu.version      VERSYM          00000a10 000a10 000092 02   A  5   0  2
  [ 8] .gnu.version_r    VERNEED         00000aa4 000aa4 0000c0 00   A  6   4  4
  [ 9] .rel.dyn          REL             00000b64 000b64 000580 08   A  5   0  4

  [25] .bss              NOBITS          00011198 010194 019184 00  WA  0   0  8
  [26] .comment          PROGBITS        00000000 010194 000024 01  MS  0   0  1
  [27] .ARM.attributes   ARM_ATTRIBUTES  00000000 0101b8 00003b 00      0   0  1
  [28] .debug_aranges    PROGBITS        00000000 0101f8 0004e8 00      0   0  8
  [29] .debug_info       PROGBITS        00000000 0106e0 037711 00      0   0  1
  [30] .debug_abbrev     PROGBITS        00000000 047df1 008460 00      0   0  1
  [31] .debug_line       PROGBITS        00000000 050251 01432c 00      0   0  1
  [32] .debug_frame      PROGBITS        00000000 064580 001638 00      0   0  4
  [33] .debug_str        PROGBITS        00000000 065bb8 005a42 01  MS  0   0  1
  [34] .debug_line_str   PROGBITS        00000000 06b5fa 0000a6 01  MS  0   0  1
  [35] .debug_loclists   PROGBITS        00000000 06b6a0 0156b9 00      0   0  1
  [36] .debug_rnglists   PROGBITS        00000000 080d59 002436 00      0   0  1
  [37] .symtab           SYMTAB          00000000 083190 002f70 10     38 628  4
  [38] .strtab           STRTAB          00000000 086100 00147c 00      0   0  1
  [39] .shstrtab         STRTAB          00000000 08757c 00019a 00      0   0  1
```

```
mathieu@meije ~/tmp/unwind/t2$ $READELF -S a.out
There are 29 section headers, starting at offset 0x10300:

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .interp           PROGBITS        00000154 000154 000019 00   A  0   0  1
  [ 2] .note.gnu.bu[...] NOTE            00000170 000170 000024 00   A  0   0  4
  [ 3] .note.ABI-tag     NOTE            00000194 000194 000020 00   A  0   0  4
  [ 4] .gnu.hash         GNU_HASH        000001b4 0001b4 000018 04   A  5   0  4
  [ 5] .dynsym           DYNSYM          000001cc 0001cc 000490 10   A  6   3  4
  [ 6] .dynstr           STRTAB          0000065c 00065c 0003b4 00   A  0   0  1
  [ 7] .gnu.version      VERSYM          00000a10 000a10 000092 02   A  5   0  2
  [ 8] .gnu.version_r    VERNEED         00000aa4 000aa4 0000c0 00   A  6   4  4
  [ 9] .rel.dyn          REL             00000b64 000b64 000580 08   A  5   0  4
  [10] .rel.plt          REL             000010e4 0010e4 0001f0 08  AI  5  23  4
  [11] .init             PROGBITS        000012d4 0012d4 00000c 00  AX  0   0  4
  [12] .plt              PROGBITS        000012e0 0012e0 000300 04  AX  0   0  4
  [13] .text             PROGBITS        000015e0 0015e0 00b944 00  AX  0   0  4
  [14] .fini             PROGBITS        0000cf24 00cf24 000008 00  AX  0   0  4
  [15] .rodata           PROGBITS        0000cf2c 00cf2c 002308 00   A  0   0  4
  [16] .ARM.extab        PROGBITS        0000f234 00f234 00012c 00   A  0   0  4
  [17] .ARM.exidx        ARM_EXIDX       0000f360 00f360 000318 00  AL 13   0  4
  [18] .eh_frame         PROGBITS        0000f678 00f678 000004 00   A  0   0  4
  [19] .init_array       INIT_ARRAY      00010a90 00fa90 000004 04  WA  0   0  4
  [20] .fini_array       FINI_ARRAY      00010a94 00fa94 000004 04  WA  0   0  4
  [21] .data.rel.ro      PROGBITS        00010a98 00fa98 000460 00  WA  0   0  4
  [22] .dynamic          DYNAMIC         00010ef8 00fef8 000108 08  WA  6   0  4
  [23] .got              PROGBITS        00011000 010000 000164 04  WA  0   0  4
  [24] .data             PROGBITS        00011164 010164 000030 00  WA  0   0  4
  [25] .bss              NOBITS          00011198 010194 019184 00  WA  0   0  8
  [26] .comment          PROGBITS        00000000 010194 000024 01  MS  0   0  1
  [27] .ARM.attributes   ARM_ATTRIBUTES  00000000 0101b8 00003b 00      0   0  1
  [28] .shstrtab         STRTAB          00000000 0101f3 00010d 00      0   0  1
```

```
mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 538K Jan 22 11:11 a.out
```



```
mathieu@meije ~/tmp/unwind/t2$ $STRIP --keep-section=.debug_frame --strip-debug a.out
```



```
mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 88K Jan 24 18:49 a.out
```

```
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
  #0 0x42f9d9: segv_handler+0xf
  #1 0xb6e8c181: unknown+0x0
  #2 0x42f9ee: f3+0xd
  #3 0x42f9ff: f2+0x7
  #4 0x42fa0b: f1+0x7
  #5 0x42fa61: main+0x51
  #6 0xb6e7c2cd: unknown+0x0
  #7 0xb6e7c37d: __libc_start_main+0x6f
  #8 0x42f609: unknown+0x0
```

```
# this function is slight modification of the one used in RPM
# found at https://bugzilla.redhat.com/show_bug.cgi?id=834073
# written by Alexander Larsson <alexl@redhat.com>
add_minidebug()
{
  debuginfo="$1" ## we don't have separate debuginfo file
  binary="$1"

  dynsyms=`mktemp`
  funcsyms=`mktemp`
  keep_symbols=`mktemp`
  mini_debuginfo=`mktemp`

  # Extract the dynamic symbols from the main binary, there is no need to also have these
  # in the normal symbol table
  nm -D "$binary" --format=posix --defined-only | awk '{ print $1 }' | sort > "$dynsyms"
  # Extract all the text (i.e. function) symbols from the debuginfo
  nm "$debuginfo" --format=posix --defined-only | awk '{ if ($2 == "T" || $2 == "t") print $1 }' | sort > "$funcsyms"
  # Keep all the function symbols not already in the dynamic symbol table
  comm -13 "$dynsyms" "$funcsyms" > "$keep_symbols"
  # Copy the full debuginfo, keeping only a minimal set of symbols and removing some unnecessary sections
  objcopy -S --remove-section .gdb_index --remove-section .comment --keep-symbols="$keep_symbols" "$debuginfo" "$mini_debuginfo"
 &> /dev/null
  wait
  #Inject the compressed data into the .gnu_debugdata section of the original binary
  xz "$mini_debuginfo"
  mini_debuginfo="${mini_debuginfo}.xz"
  objcopy --add-section .gnu_debugdata="$mini_debuginfo" "$binary"
  rm -f "$dynsyms" "$funcsyms" "$keep_symbols" "$mini_debuginfo"

  strip "$binary" ## throw away the symbol table
}
```

```
$NM a.out --format=posix --defined-only | awk '{ if ($2 == "T" || $2 == "t") print $1 }' | sort > funcsyms
$OBJCOPY -S --remove-section .gdb_index --remove-section .comment --keep-symbols=funcsyms a.out a.out.mdi
$STRIP --keep-section=.debug_frame a.out
xz a.out.mdi
$OBJCOPY   --add-section .gnu_debugdata=a.out.mdi.xz a.out
```

```
mathieu@meije ~/tmp/unwind/t2$ ls -lh a.out
-rwxr-xr-x 1 mathieu users 106K Jan 22 12:40 a.out
```

```
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
  #0 0x4f1987: segv_handler+0xd
  #1 0xb6e98181: unknown+0x0
  #2 0x4f1996: f3+0x9
  #3 0x4f19a5: f2+0x5
  #4 0x4f19af: f1+0x5
  #5 0x4f19f7: main+0x43
  #6 0xb6e882cd: unknown+0x0
  #7 0xb6e8837d: __libc_start_main+0x6f
  #8 0x4f1609: unknown+0x0
```

Size increase for backtrace support

Full	Stripped	Stripped + .debug_frame	Stripped + .debug_frame + .symtab	Stripped + .debug_frame + .gnu_debugdata
538K	66K	72K	88K	106K

```c
int main() {
        const char data[] = "boom";

        struct sigaction sa = {
                .sa_sigaction = segv_handler,
                .sa_flags = SA_SIGINFO
        };
        sigemptyset(&sa.sa_mask);
        sigaction(SIGSEGV, &sa, NULL);

        EVP_MD_CTX *ctx = EVP_MD_CTX_new();
        if (!ctx) {
            fprintf(stderr, "EVP_MD_CTX_new failed\n");
            return 1;
        }

        if (EVP_DigestInit_ex(ctx, EVP_sha256(), NULL) != 1) {
            fprintf(stderr, "EVP_DigestInit_ex failed\n");
            return 1;
        }

        EVP_MD_CTX_free(ctx);
        EVP_DigestUpdate(ctx, data, strlen(data));

        return 0;
}
```

Partial backtrace on Yocto

```
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
  #0 0x4a2b41: segv_handler+0xf
  #1 0xb6bc2181: unknown+0x0
  #2 0xb6dcf0ec: EVP_DigestUpdate+0x73
```

```
mathieu@FRMONPC1176:~/unwind/build/tmp/deploy/images/qemuarm/root$ $READELF -S ./usr/lib/libcrypto.so.3
There are 27 section headers, starting at offset 0x2a9278:

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .note.gnu.bu[...] NOTE            00000114 000114 000024 00   A  0   0  4
  [ 2] .gnu.hash         GNU_HASH        00000138 000138 008574 04   A  3   0  4
  [ 3] .dynsym           DYNSYM          000086ac 0086ac 0161b0 10   A  4   3  4
  [ 4] .dynstr           STRTAB          0001e85c 01e85c 01c41b 00   A  0   0  1
  [ 5] .gnu.version      VERSYM          0003ac78 03ac78 002c36 02   A  3   0  2
  [ 6] .gnu.version_d    VERDEF          0003d8b0 03d8b0 0000ec 00   A  4   7  4
  [ 7] .gnu.version_r    VERNEED         0003d99c 03d99c 0000b0 00   A  4   2  4
  [ 8] .rel.dyn          REL             0003da4c 03da4c 024b40 08   A  3   0  4
  [ 9] .rel.plt          REL             0006258c 06258c 0004f0 08  AI  3  21  4
  [10] .init             PROGBITS        00062a7c 062a7c 00000c 00  AX  0   0  4
  [11] .plt              PROGBITS        00062a88 062a88 0007e0 04  AX  0   0  4
  [12] .text             PROGBITS        00063280 063280 1927e4 00  AX  0   0 64
  [13] .fini             PROGBITS        001f5a64 1f5a64 000008 00  AX  0   0  4
  [14] .rodata           PROGBITS        001f6000 1f6000 082be8 00   A  0   0 4096
  [15] .ARM.exidx        ARM_EXIDX       00278be8 278be8 000008 00  AL 12   0  4
  [16] .eh_frame         PROGBITS        00278bf0 278bf0 000004 00   A  0   0  4
  [17] .init_array       INIT_ARRAY      00279474 279474 000008 04  WA  0   0  4
  [18] .fini_array       FINI_ARRAY      0027947c 27947c 000004 04  WA  0   0  4
  [19] .data.rel.ro      PROGBITS        00279480 279480 02e2e8 00  WA  0   0  8
  [20] .dynamic          DYNAMIC         002a7768 2a7768 000118 08  WA  4   0  4
  [21] .got              PROGBITS        002a7880 2a7880 00077c 04  WA  0   0  4
  [22] .data             PROGBITS        002a8000 2a8000 001124 00  WA  0   0  8
  [23] .bss              NOBITS          002a9128 2a9124 002420 00  WA  0   0  8
  [24] .ARM.attributes   ARM_ATTRIBUTES  00000000 2a9124 000039 00      0   0  1
  [25] .gnu_debuglink    PROGBITS        00000000 2a9160 000014 00      0   0  4
  [26] .shstrtab         STRTAB          00000000 2a9174 000101 00      0   0  1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings), I (info),
  L (link order), O (extra OS processing required), G (group), T (TLS),
  C (compressed), x (unknown), o (OS specific), E (exclude),
  D (mbind), y (purecode), p (processor specific)
```

```
$NM -D ./usr/lib/libcrypto.so.3|grep DigestUpdate
 00102078 T EVP_DigestUpdate@@OPENSSL_3.0.0
```

Mathieu Othacehe

Add a 'PACKAGE_KEEP_SECTIONS' variable to keep some specific ELF sections while stripping binaries and libraries.

That one can then be used to keep the .debug_frame section around for example, this way:

PACKAGE_KEEP_SECTIONS = ".debug_frame"

35.16 Enabling Minidebuginfo

Enabling the DISTRO_FEATURES minidebuginfo adds a compressed ELF section `.gnu_debugdata` to all binary files, containing only function names, and thus increasing

Default (stripped)	.symtab + .debug_frame	.gnu_debugdata (minidebuginfo) + .debug_frame	No stripping
48M	55M	57M	194M

INHIBIT_PACKAGE_STRIP

If set to “1”, causes the build to not strip binaries in resulting packages and prevents the `-dbg` package from containing the source files.

Full backtrace on Yocto

```
root@qemuarm:~# /tmp/a.out
Backtrace (newest to oldest):
  #0 0x4bbb41: segv_handler+0xf
  #1 0xb6b81181: unknown+0x0
  #2 0xb6d8e0ec: EVP_DigestUpdate+0x73
  #3 0x4bbc1d: main+0xd3
  #4 0xb6b712cd: __libc_start_call_main+0x47
  #5 0xb6b7137d: __libc_start_main+0x6f
  #6 0x4bb771: unknown+0x0
```

Conclusion

- Unwind info (DWARF or architecture specific)
- Symbols (symbols table, DWARF or minidebuginfo)
- Unwider (libunwind, custom or else)
- Light Yocto patching might be necessary
- Backtraces in the syslog are quite handy