The Odyssey of HWCAP on RISC-V platforms

September 14th, 2022 • RISC-V MC in Linux Plumbers"22 Dublin

cat /etc/os-release

Name : Ruinland
Place of Birth : Taiwan
Occupation : Jack of all trades (master of none) regarding Linux.
Employer : SiFive Taiwan
F/LOSS related activities :

- Track host for COSCUP (Conference for Open Source Coders, Users and Promoters)
- Hsinchu Code Serfs

©2022 SiFive

credit : [Arnold Böcklin - Odysseus und Polyphemus (1896) - Odysseus and Polyphemus (Böcklin) - Wikipedia]

©2022 SiFive

Poleis :

City states with their own sovereignty

credit: [Homeric Greece-en - File:Homeric Greece-en.svg - Wikipedia]

©2022 SiFive

Extensions, specs …

Specification Status - Home - RISC-V International (riscv.org)

Besides single -lettered ones, we have plenty others :

- Zcee
- Zjpm
- Zawrs
- Zmmul

  ……

©2022 SiFive

Vendors

Members - RISC-V International (riscv.org)

Many of us introduce our own flavors on their IPs. There are Linux build for them in quantities :

- Zero-bubble quick-prediction turnaround
- Andes : qemu/andes_custom.decode at qemu-ast-v5_1_0-branch · andestech/qemu · GitHub
- Syntacore : Syntacore introduction (riscv.or.jp)
- T-Head : target/riscv: Xuantie CPU support · T-head-Semi/qemu@fd25f40 · GitHub

Sometimes, we have errata :

- SiFive FU540-C000 Manual: v1p0
- [v3,0/5] riscv: introduce alternative mechanism to apply errata patches - Patchwork (kernel.org)

©2022 SiFive

What should be funnel through and how ?

A million dollar question.

credit:Woman in Green Shirt Thinking · Free Stock Photo (pexels.com)

©2022 SiFive

CSRs

The closest things we have like the cpuid.

- mvendorid
- marchid
- mimplid

RISC-V is not like ARM, X86 or any other monopoly/oligarchy ISA.
There are too many combinations! It’s not feasible to compose an exhausted list!

©2022 SiFive

The ISA string

Discussions :
Canonical form of ISA string: extensions set · Issue #670 · riscv/riscv-isa-manual · GitHub
The privileged specs regulating ISA string has changed and there are already binaries shipping with the attribute tag.

©2022 SiFive

Just not enough.

Even we have all these things included, different batches of the exact model of IC could vary in performance or even have different errata.
iPhone 6s: Samsung And TSMC A9 SoCs Tested | Tom's Hardware (tomshardware.com)

©2022 SiFive

Alright, let's pretend that we solved it.

By what we can pass it to user programs ?

©2022 SiFive

credit:Frustrated ethnic man in casual wear on blue background in studio · Free Stock Photo (pexels.com)

/proc/cpuinfo ?

Containers don’t like it for exposing procfs, it’s dangerous (reasonably.)
Even they do, sometimes they make their own facade : lxc/lxcfs: FUSE filesystem for LXC (github.com)

©2022 SiFive

HWCAP ?

Return value of getauxval() is “unsigned long”-typed, 32 bits long on RV32 platforms with 26 of it allocated already.

```
GETAUXVAL(3)                                Linux Programmer's Manual                               GETAUXVAL(3)

NAME
       getauxval - retrieve a value from the auxiliary vector

SYNOPSIS
       #include <sys/auxv.h>

       unsigned long getauxval(unsigned long type);
```

©2022 SiFive

HWCAP “2” ?

Another bit-vector, seriously ?

credit:Girl in Red Crew Neck Shirt Looking Confused · Free Stock Photo (pexels.com)

©2022 SiFive

vDSO data

where kernel pass information to userspace

©2022 SiFive

arch_vdso _data

Only S390 uses it now, though.

/ arch / s390 / include / asm / vdso / data.h

```
1   /* SPDX-License-Identifier: GPL-2.0 */
2   #ifndef __S390_ASM_VDSO_DATA_H
3   #define __S390_ASM_VDSO_DATA_H
4
5   #include <linux/types.h>
6   #include <vdso/datapage.h>
7
8   struct arch_vdso_data {
9           __s64 tod_steering_delta;
10          __u64 tod_steering_end;
11  };
12
13  #endif /* __S390_ASM_VDSO_DATA_H */
```

/ include / vdso / datapage.h

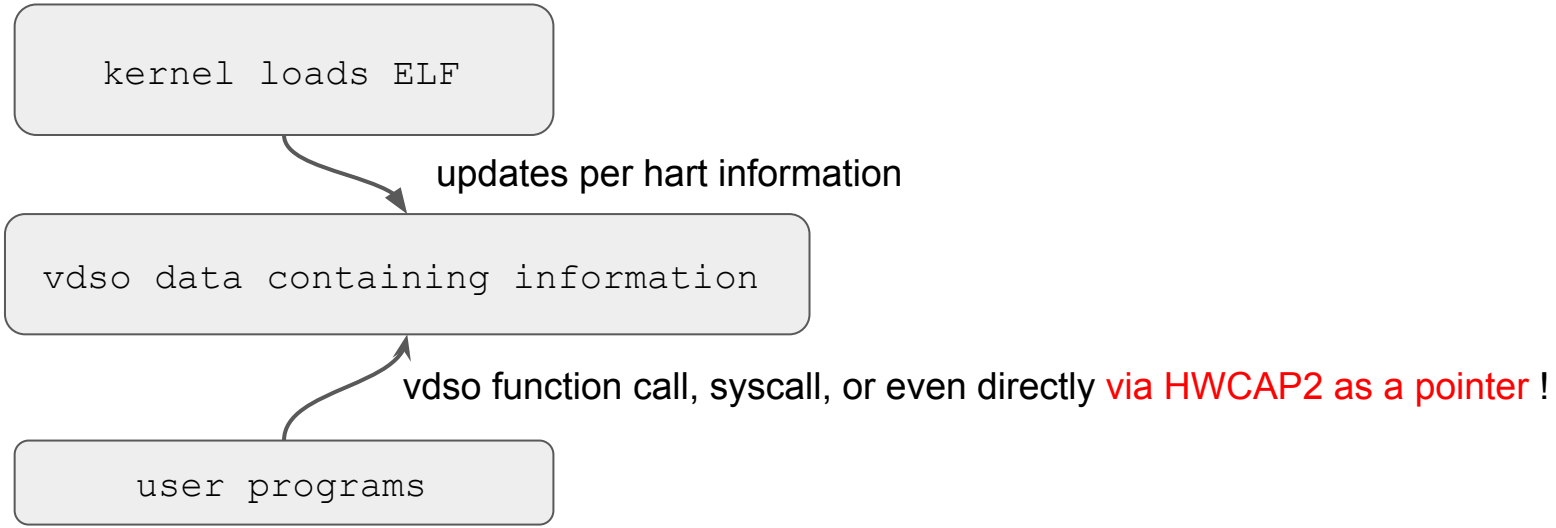
```
21
22  #ifdef CONFIG_ARCH_HAS_VDSO_DATA
23  #include <asm/vdso/data.h>
24  #else
25  struct arch_vdso_data {};
26  #endif
27
```

Existing RISC-V core (StarFive Dubhe) needs per-hart information :
https://youtu.be/m3R6ejtmXls?t=182

©2022 SiFive

Mechanism

Sugar, spice, and everything nice.



©2022 SiFive

Look and feel

The userspace programs could fetch HWCAP2 via `getauxval()` :

```
struct arch_vdso_data {
    uint64_t mvendor;
    uint64_t march;
    uint64_t mimpl;
    struct list_head extension_head;
    char buffer[];
} __attribute__((packed));

struct rvext {
    uint64_t  spec_maj;
    uint64_t  spec_min;
    char ext_name[100];
    struct list_head exts;
    };

#define container_of(ptr, type, member) ({                      \
        const typeof( ((type *)0)->member ) *__mptr = (ptr);    \
        (type *)( (char *)__mptr - offsetof(type,member) );})

int main(int ac, char *av) {
        unsigned long hwcap2_address = getauxval(AT_HWCAP2);
        struct arch_vdso_data* cur;
        struct list_head* iter;
        struct rvext* ext_ptr;

        printf("hwcap2 points to %lx \n", hwcap2_address);
        cur = (struct arch_vdso_data*) hwcap2_address;
        printf("HWCAP2 info :\nmvendor %x, march %x, mimpl %x\n", cur->mvendor, cur->march, cur->mimpl);
        iter = &(cur->extension_head);
        for(ext_ptr = (struct rvext*) cur->buffer; iter != NULL; ext_ptr++) {
                printf("ext name : %s, maj : %d, min : %d\n", ext_ptr->ext_name, ext_ptr->spec_maj, ext_ptr->spec_min);
                iter = ext_ptr->exts.next;
                }

        return 0;
        }
```

```
# cat /proc/device-tree/cpus/cpu@0/riscv,isa
rv64imafdcbsu_zba0p1_zbb2p3_zfh3p4#
# hwcap_traverse
hwcap2 points to ffffff904110f0
HWCAP2 info :
mvendor 0, march 0, mimpl 0
ext name : zba, maj : 0, min : 1
ext name : zbb, maj : 2, min : 3
ext name : zfh, maj : 3, min : 4
```

©2022 SiFive

Look and feel

Syscall or vdso function call :-)

```
root@qemuriscv64:~# ./test_syscall_with_version
Found xsfvfhbfmin v7p8 !
Found zfh v5p6 !
Found zba v1p2 !

root@qemuriscv64:~# ./test_vdso_with_version
Found xsfvfhbfmin v7p8 !
Found zfh v5p6 !
Found zba v1p2 !
```

```
int main(int ac, char *av) {
        unsigned long maj = 0, min = 0;
        void *handle;
        int (*getext)(char *, unsigned long *, unsigned long *);
        char *error;

        handle = dlopen("linux-vdso.so.1", RTLD_LAZY);
        if (!handle) {
        fprintf(stderr, "%s\n", dlerror());
        exit(EXIT_FAILURE);
        }
        *(void **) (&getext) = dlsym(handle, "__vdso_riscv_check_extension");

           if ((error = dlerror()) != NULL)  {
                fprintf(stderr, "%s\n", error);
                exit(EXIT_FAILURE);
            }

        char find[100] = "xsfvfhbfmin";
        if((*getext)(find, &maj, &min) == 0) {
                printf("Found xsfvfhbfmin v%dp%d !\n", maj, min);
                }
        char find1[] = "zfh";
        if((*getext)(find1, &maj, &min) == 0) {
                printf("Found zfh v%dp%d !\n", maj, min);
                }
        char find2[] = "zba";
        if((*getext)(find2, &maj, &min) == 0) {
                printf("Found zba v%dp%d !\n", maj, min);
                }
        getchar();
        return 0;
        }
```

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/syscall.h>
#include <unistd.h>

int main(int ac, char *av) {
        long getext = 248;
        unsigned long maj = 0, min = 0;
        char find[100] = "xsfvfhbfmin";
        if(syscall(248, find, &maj, &min) == 0) {
                printf("Found xsfvfhbfmin v%dp%d !\n
                }
        char find1[] = "zfh";
        if(syscall(248, find1, &maj, &min) == 0) {
                printf("Found zfh v%dp%d !\n", maj,
                }
        char find2[] = "zba";
        if(syscall(248, find2, &maj, &min) == 0) {
                printf("Found zba v%dp%d !\n", maj,
                }
        getchar();
        return 0;
}
```

©2022 SiFive

Technical details

Add a new header containing the definition of `arch_vdso_data` and update `Kconfig`.

```
8 struct arch_vdso_data {
9     u64 mvendor;
0     u64 march;
1     u64 mimpl;
2     struct list_head extension_head;
3     char buffer[8192];
4 } __packed;
5
6 struct rvext {
7     u64  spec_maj;
8     u64  spec_min;
9     char ext_name[100];
0     struct list_head exts;
1     };
2 #endif
```

```
--- a/arch/riscv/Kconfig
+++ b/arch/riscv/Kconfig
@@ -29,6 +29,7 @@ config RISCV
        select ARCH_HAS_SET_DIRECT_MAP
        select ARCH_HAS_SET_MEMORY
        select ARCH_HAS_STRICT_KERNEL_RWX if MMU
+       select ARCH_HAS_VDSO_DATA
        select ARCH_OPTIONAL_KERNEL_RWX if ARCH_HAS_STRICT_KERNEL_RWX
        select ARCH_OPTIONAL_KERNEL_RWX_DEFAULT
        select ARCH_WANT_DEFAULT_TOPDOWN_MMAP_LAYOUT if MMU
```

This is just proof-of-concept data structure. Mr. Palmer Dabbelt and Mr. Paul Walmsley come up with a much well-defined data structure/interface proposal.

©2022 SiFive

Technical details (cont.)

Define `ELF_HWCAP2` to `vdso_data` + the offset of `arch_data`, which will be updated in `riscv_fill_hwcap()` which will be called during RISC-V's `parse_dtb()` init sequence.

```
20 extern unsigned long vdso_data_location;
21 #define ELF_HWCAP2      (vdso_data_location + offsetof(struct vdso_data, arch_data))
```

```
#define SET_ISA_EXT_MAP(name, bit)                                          \
                    do {                                                    \
                            if ((ext_end - ext == sizeof(name) - 1) &&      \
                                 !memcmp(ext, name, sizeof(name) - 1))      \
                                 !memcmp(ext, name, sizeof(name) - 1)) {    \
                                    set_bit(bit, this_isa);                 \
                                    craft = (struct rvext *) \
                                    &(vdata[riscv_of_processor_hartid(node)].arch_data.buffer[\
                                    sizeof(struct rvext) * ext_offset_n[riscv_of_processor_hartid(node)]]); \
                                    craft->spec_maj = ext_major; \
                                    craft->spec_min = ext_minor; \
                                    strncpy(craft->ext_name, tmp, ext_end-ext); \
                                    iter = &(vdata[riscv_of_processor_hartid(node)].arch_data.extension_head);\
                                    while (iter->next != NULL) { \
                                            iter = iter->next; \
                                    } \
                                    craft->exts.prev = iter; \
                                    craft->exts.next = NULL; \
                                    iter->next = &(craft->exts); \
                                    ++ext_offset_n[riscv_of_processor_hartid(node)]; \
                                    } \
                    } while (false)                                         \
```

©2022 SiFive

The good (just my 2 cents)

1. The information could be free-formed, even if you don't like what comes up with Palmer & Paul. You are free to defined your own and use this "plumbing" mechanism.

2. The information is per hart and thus heterogeneous CPU cluster could deal with it correctly.

3. If you really really want to use it with MMU-less processor, I guess you can put information into a kernel space and access it via system call.

©2022 SiFive

The bad

Well, it's processed during dtb parsing logic. So if you enable/disable CPU features, such as VPU during "wrong"-time. The information would be mismatched.

credit : The General (1927) a film is now in public domain.

©2022 SiFive

And the ugly

Doing things like string comparing (for checking extension string) in VDSO function is chaotic. There's _no_ any kind of libc for you. I have to copy-paste the code :

```
static inline int strcmp(const char *cs, const char *ct)
{
    unsigned char c1, c2;

    while (1) {
    c1 = *cs++;
    c2 = *ct++;
    if (c1 != c2)
        return c1 < c2 ? -1 : 1;
    if (!c1)
        break;
    }
    return 0;
}

extern
int __vdso_riscv_check_extension(const char *, unsigned long *, unsigned long *);
int __vdso_riscv_check_extension(const char *query, unsigned long *maj, unsigned long *min)
{
```


©2022 SiFive

Rabbit hole :

Atish reminds me that Plumber is more of a discussion, I'll leave the detail here :
https://github.com/Ruinland-Tsai/br2_external_rvext

©2022 SiFive

Thank you

SIFIVE.COM

©2022 SiFive, Inc. All rights reserved. All trademarks referenced herein belong to their respective companies.
This presentation is intended for informational purposes only and does not form any type of warranty.

Certain information in this presentation may outline SiFive's general product direction. The presentation shall not serve to amend or affect the rights or obligations of SiFive or its licensees under any license or service agreement or documentation relating to any SiFive product. The development, release, and timing of any products, features, and functionality remains at SiFive's sole discretion.

Backups

GLIBC improvements & what to expect in future Linux distributions | Blog | Linaro
We need to setup some kinds of profiles for RISC-V.

©2022 SiFive

Look and feel : from the glibc side

In `csu/libc-start.c` :

```
LIBC_START_MAIN() =>
     _dl_aux_init(); // The hwcaps will be fetched from memory here.
     ARCH_INIT_CPU_FEATURES () ; // x86 and Aarch64 want this instead of plain hwcaps.
     ARCH_SETUP_IREL() ; // Where the GNU IFUNC is resolved.
     ……
     ARCH_APPLY_IREL() ; // If GNU IFUNC needs something more to be resolved …
```

©2022 SiFive

Palmer's Proposal

Excerpt of this Google Doc.

Proposal

```
struct riscv_feature {
     u32 key;
     uintptr_t val;
};

/*
 * Fills out up to the input number of entries of features, returning
 * the number of bytes copied.
 */
ssize_t sys_riscv_hwcaps(struct riscv_feature *features, size_t nent,
                         struct cpuset_t *cpus, size_t offset);
```

Every key and bit in val are reserved, reserved bits are 0. When new bits are defined 0 will mean unknown, unless there's some value that's harmless to alias with unknown. I'm not sure if the copying is a problem here, if it is we can sort out a different syscall. Having some stuff in the VDSO/HWCAP makes sense too, we just need to define it as something like "the subset that's supported on all CPUs".

Keys and Values

0: End of list

N: ISA STring for User 2.2

Pointer to a string

1: marchid

2: mvendorid

3: mimplid

4: base system behavior

A bit field that specifies the base behaviors the system exhibits. This is subtly different than just saying a specification, as it may contain behavior that is not defined or allowed by a specification.

0: Version 2.2 of the user, version 1.10 of priv, rv32/64ima

Also all the implicit uABI stuff, like "don't use fence.i, but it doesn't trap", RVWMO, support for mis-aligned accesses (even if they're slow). We can't enumerate everything here.

5: RVI-Defined Extensions to 4/0, part 1

Also a bit field, though maybe we merge some together. These are all defined as a specific version of the spec, if there's a future version of the spec then we'll allocate another bit if there are any differences (and maybe HW is compatible with both).

©2022 SiFive