Western Digital®

32-bit RISC-V glibc port

Alistair Francis <alistair.francis@wdc.com>

Linux Plumbers

24th of August 2020

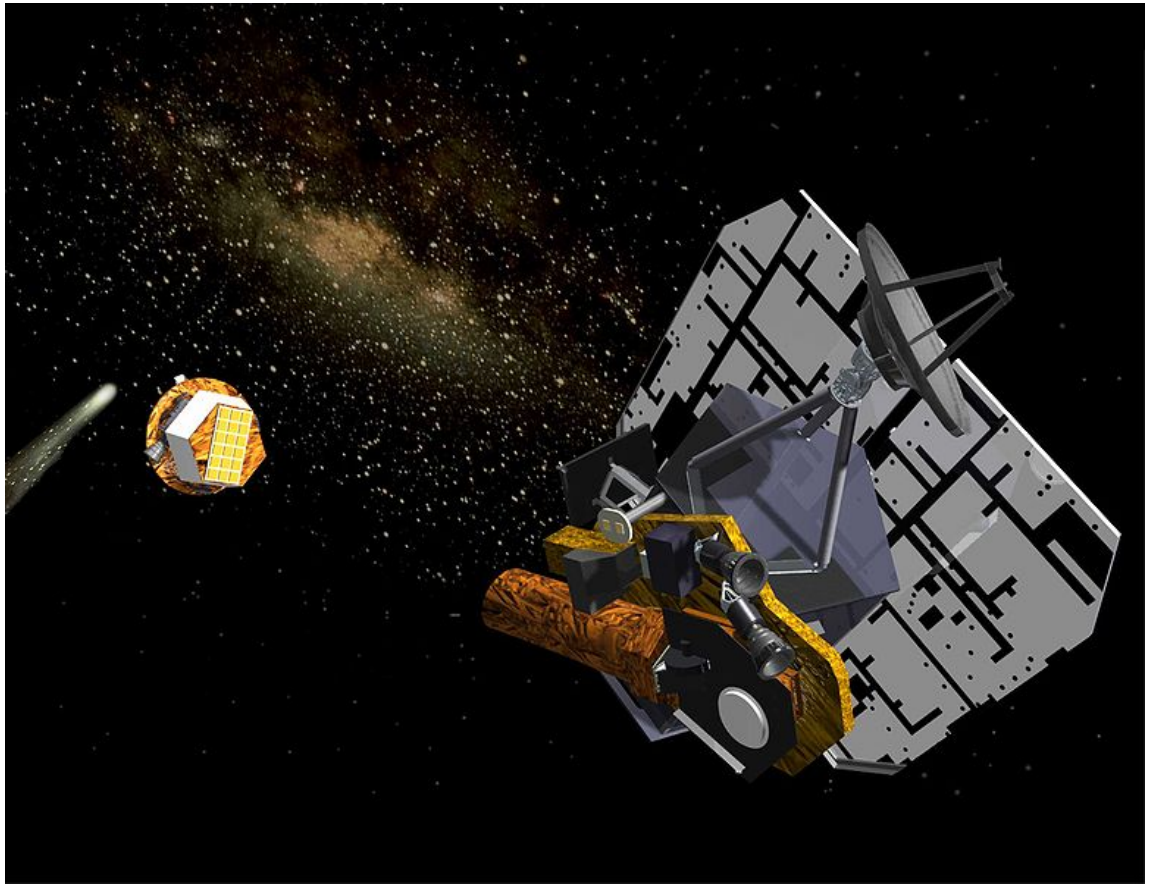
Context

Skipped in presentation, included for slide context

The y2038 problem

Y2K all over again

- In January 2038, the Unix Epoch will overflow a 32-bit signed integer
- One of the solutions is to use a 64-bit time_t
  - 64-bit Linux® archs already use a 64-bit time_t
  - Linux 5.1 added initial support for using a 64-bit userspace time_t for 32-bit archs



Computer rendering of the Deep Impact space probe after separation of the impactor (credit NASA/JPL) https://en.wikipedia.org/wiki/File:Deep_Impact.jpg

64-bit time_t on 32-bit systems

- Linux kernel 5.1 added support for 64-bit time_t syscalls
  - All existing 32-bit archs can continue using the current syscalls unchanged
  - RISC-V 32-bit does not support the backwards compatible 32-bit time_t syscalls
- Examples of the new syscalls (for all 32-bit archs only)
  - clock_gettime64
  - clock_settime64
  - clock_getres64
  - clock_nanosleep_time64
  - waitid
  - fsblkcnt64_t and fsfilcnt64_t padding

Discussion Points

RISC-V 32-bit (RV32) glibc port

- RISC-V 32-bit support is hopefully going to be included in glibc 2.33
- RV32 will require kernel/kernel headers 5.4+
  - 5.4 adds support for the P_PGID idtype, required for the waitid() function

- Benefits of a 32-bit RISC-V port
  - Great for Linux capable FPGAs
    - VexRISCV can boot 32-bit Linux on FPGAs
  - Useful for testing QEMU, Linux, GCC, OpenEmbedded and other software

memset Procedure Linkage Table (PLT)

- libc_pic.a contains a PLT memset call
  - https://patchwork.ozlabs.org/project/glibc/patch/20200622211034.659739-1-alistair.francis@wdc.com
  - https://www.mail-archive.com/gcc@gcc.gnu.org/msg92537.html
    - 1bc: 009aa023 sw s1,0(s5)
    - memset (result->__data, '\0',
    - 1c0: 865e mv a2,s7
    - 1c2: 4581 li a1,0
    - 1c4: 008a8513 addi a0,s5,8
    - 1c8: 00000097 auipc ra,0x0
    - 1c8: R_RISCV_CALL_PLT __GI_memset
    - 1c8: R_RISCV_RELAX *ABS*
    - 1cc: 000080e7 jalr ra # 1c8 <.LVL39+0x14>
- What is the plan with R_RISCV_CALL and R_RISCV_CALL_PLT in GCC?

__pthread_rwlock_arch_t

RV64 version of struct __pthread_rwlock_arch_t isn't generic

- https://patchwork.ozlabs.org/project/glibc/patch/ba4cbdd944753b55845ccb3a48a73ba6c6e09cd7.1591201405.git.alistair.francis@wdc.com/

- RV64 uses (same as AArch64):
  - int __cur_writer;
  - int __shared;
  - unsigned long int __pad1;
  - unsigned long int __pad2;
  - unsigned int __flags;

- Generic/RV32 uses:
  - # if __BYTE_ORDER == __BIG_ENDIAN
  - unsigned char __pad1;
  - unsigned char __pad2;
  - unsigned char __shared;
  - unsigned char __flags;
  - # else
  - unsigned char __flags;
  - unsigned char __shared;
  - unsigned char __pad1;
  - unsigned char __pad2;
  - # endif
  - int __cur_writer;

__WORDSIZE_TIME64_COMPAT32

- 64-bit RISC-V defined __WORDSIZE_TIME64_COMPAT32 as 1
  - https://patchwork.ozlabs.org/project/glibc/patch/ba4cbdd944753b55845ccb3a48a73ba6c6e09cd7.1591201405.git.alistair.francis@wdc.com
  - Defines time size for utmp struct
    - Used by login, getutline and others
  - Being set to 1 for RV64 tells glibc to use a 32-bit time for logs
    - This was done with the assumption RV32 will use a 32-bit time_t
    - Both RV32 and RV64 need to be the same to allow mixed length userspace
  - Currently RV32 also sets this to true (32-bit time for logs) because of this
- How should we handle this?

64-bit time_t syscalls

- libc is not the only software doing syscalls
- All software needs to be checked and updated to use new syscalls
- OpenSSL and BusyBox have been updated already
- Futex syscalls are a common example

RV32 Strace port

- syslog test hangs at the SYSLOG_ACTION_READ_ALL
- Any ideas?

Western Digital®