netbsd/aarch64 in the past year

ryo@nerv.org
ryo@netbsd.org
log

- 2018-04-01  Add support for aarch64
- 2018-07-09  SMP!
- 2018-08-15  MODULAR
- 2018-08-26  big.LITTLE
- 2018-08-26  SLJIT (by rjs@)
- 2018-10-03  ThunderX! (by skrll@)
- 2018-10-12  COMPAT_NETBSD32
- 2018-10-28  EFI (by jmcneill@)
- 2018-11-01  KASAN aarch64 (by maxv@)
- 2018-11-24  ARM Server Base System Architecture (SBSA), and ACPI (by jmcneill@)
- 2018-12-13  PT_TRACE
- 2018-12-27  kernel crash dump (by mrg@)
- 2019-02-06  improve pmap_remove
2018-04-01
Add support for aarch64

• on April fool day, but it is true
• https://twitter.com/rsh/statuses/980303726668759046
• https://twitter.com/h_kenken/statuses/980316789505540096
• https://www.reddit.com/r/NetBSD/comments/88pfue/aarch64_support_added/
• …but maximum 4 cores. restricted on the same cpu cluster.
big.LITTLE (multi cpu clusters)

- https://twitter.com/rsh/statuses/1031876876422144007
- http://www.nerv.org/netbsd/?q=id:20180826T181550Z.5d9e0b329abfa7442f0cf374297bdef47e91d162

- but maximum 32 cores. some variables (cpu_mbox, and so on) is 32bit… 😞
• arm32 userland can use whole 4GB memory on aarch64 kernel
• thumb mode is not supported yet.
2018-11-01
KASAN on aarch64 (by maxv@)

• kernel address sanitizer supported on aarch64
• excellent!
2018-10-28〜11-24
EFI, and ARM Server Base System Architecture (SBSA) (by jmcneill@)

• running on SCALEWAY ARM64 VPS!
• running on AWS ARM A1 instances!
http://blog.onodera.asia/2018/12/amazon-web-service-ec2a1netbsdaarch64.html
(how-to by ryouon@)

• Wonderful!
• add support hardware assist CPU single stepping for userland (gdb)
2019-02-06
improve pmap_remove

• toooooo slow when exiting process
• it takes 300 seconds for _exit(2) after mmap(2) large(128GB) file.
• why?
UVM call pmap_remove for all vm spaces when exiting a process. pmap_remove used lock/unlock per pages.

```c
pmap_remove(p, start_va, end_va, ...) {
    for (va = start_va; va < end_va; va += PAGE_SIZE) {
        _pmap_remove(p, va, ...);
    }
}

_pmap_remove(p, va, ...) {
    mutex_enter();
    ~remove a page table entry~
    ~remove and free vm_page~
    ...
    mutex_exit();
}

Oh....
I rewrote...

```c
pmap_remove(p, start_va, end_va, ...)
{
    mutex_enter();
    _pmap_remove(p, start_va, end_va, ...);
    mutex_exit();
    ~free vm_pages~
}

_pmap_remove(p, start_va, end_va...)
{
    for (va = start_va; va < end_va; va += PAGE_SIZE) {
        ~remove a page table entry~
        ~remove vm_page~
        ...
    }
}
```

300 seconds → 1 second 😊
future plan? (TODO?)

- kernel preemption
- COMPAT_LINUX
- meltdown/spectr? I doesn’t survey/catch up yet...
- more stability (fix bugs, fix bugs, fix bugs...)