
ARM
HelenOS

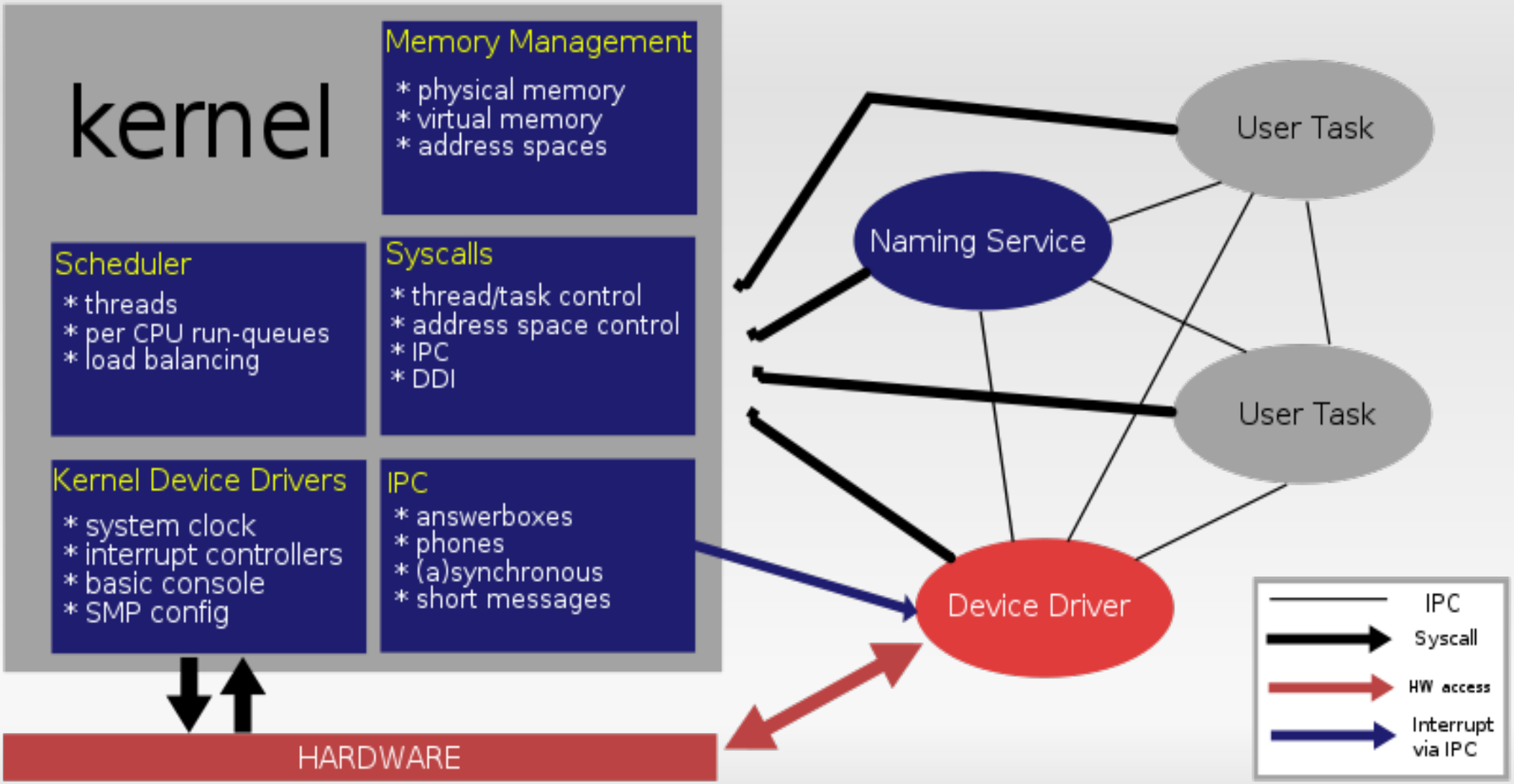
# HelenOS ARM port

Pavel Jančík
Michal Kebrt
Petr Štěpán

# HelenOS

- experimentální operační systém (MFF)
  - multiplatformní
    - amd64, ia32, ia32xen, ia64, mips32, ppc32, ppc64, sparc64
  - microkernel
    - plánování
    - správa paměti
    - IPC
  - userspace
    - device drivers
    - naming service
    - uživatelské aplikace (tetris)

# HelenOS architektura



# ARM

- 32-bit RISC
  - velká sada obecných registrů
  - load/store, operace s registry ne s pamětí
  - všechny instrukce 32 bitů (snazší dekódování)
- architektura navržená s ohledem na malá zařízení (jednoduchý návrh → malá spotřeba)
- 75% podíl na trhu embedded RISC procesorů
  - PDA, mobily, kalkulačky, počítačové periferie (disky, routery), ...

# ARM – specifika

- podmíněné spuštění u většiny instrukcí (podmínka součástí kódu instrukce)
- 2-úrovňový systém přerušení (FIQs)
- Load and Store Multiple
- banked registers
- Thumb instrukce
- Jazelle
- ...
- široká škála verzí ARM 1-11

# ARM – registry

- 31 obecných registrů (32-bit) + CPSR, SPSR
- v 1 okamžik je jich použitelných 16
  - r0-12 obecné registry
  - r13 obvykle stack pointer (není vynucováno)
  - r14 link register
  - r15 program counter
- zbylé registry jsou dostupné z různých režimů procesoru
  - některé registry (např. r14) mají svou verzi pro více režimů procesoru (banked registers)
    - při změně režimu procesor přepne na správnou verzi registru

# ARM – režimy

- User
  - určený pro běh uživatelských aplikací
  - nemůže přepnout do ostatních módů, omezení při práci s koprocesory a paměťovým systémem
- System
- Supervisor
- IRQ
- FIQ
- Abort
- Undefined

| | Privileged modes | | | | | |
|---|---|---|---|---|---|---|
| | | Exception modes | | | | |
| **User** | **System** | **Supervisor** | **Abort** | **Undefined** | **Interrupt** | **Fast interrupt** |
| R0 | R0 | R0 | R0 | R0 | R0 | R0 |
| R1 | R1 | R1 | R1 | R1 | R1 | R1 |
| R2 | R2 | R2 | R2 | R2 | R2 | R2 |
| R3 | R3 | R3 | R3 | R3 | R3 | R3 |
| R4 | R4 | R4 | R4 | R4 | R4 | R4 |
| R5 | R5 | R5 | R5 | R5 | R5 | R5 |
| R6 | R6 | R6 | R6 | R6 | R6 | R6 |
| R7 | R7 | R7 | R7 | R7 | R7 | R7 |
| R8 | R8 | R8 | R8 | R8 | R8 | R8_fiq |
| R9 | R9 | R9 | R9 | R9 | R9 | R9_fiq |
| R10 | R10 | R10 | R10 | R10 | R10 | R10_fiq |
| R11 | R11 | R11 | R11 | R11 | R11 | R11_fiq |
| R12 | R12 | R12 | R12 | R12 | R12 | R12_fiq |
| R13 | R13 | R13_svc | R13_abt | R13_und | R13_irq | R13_fiq |
| R14 | R14 | R14_svc | R14_abt | R14_und | R14_irq | R14_fiq |
| PC | PC | PC | PC | PC | PC | PC |
| CPSR | CPSR | CPSR | CPSR | CPSR | CPSR | CPSR |
| | | SPSR_svc | SPSR_abt | SPSR_und | SPSR_irq | SPSR_fiq |

*indicates that the normal register used by User or System mode has been replaced by an alternative register specific to the exception mode*

**Figure 2-1 Register organization**

# ARM emulátor

- GXemul
  - Gavare's eXperimental Emulator
  - opensource
  - emuluje ARM, MIPS, PowerPC, SuperH
- TestARM machine
  - ARM procesor verze 4
  - jednoduchá (paměťově mapovaná) zařízení
  - odstiňuje od složitosti reálného HW, zachovává vlastnosti architektury
  - není dokonalý
    - např. neimplementuje FIQ

# Portování HelenOSu

- multiplatformní OS
  - odráží se v členění zdrojového kódu
    - generic, genarch, arch
  - nyní již 9 architektur
- port
  - loader
  - implementace arch specific funkcí
    - kernel
      - paměťový subsystém, drivery, výjimky, kontexty
    - userspace
      - TLS, drivery

# Loader

- `pack` skript
  - před spuštěním se kernel a všechny tasky zabalí do jednoho souboru (image.boot)
- emulátor načte soubor image.boot
- start od adresy 0x0
  - zapne dočasné stránkování (jednoúrovňové, stránky 1MB)
  - zkopíruje kernel a tasky do kernel space
  - skočí na vstupní bod kernelu (0x80200000)

# Výjimky

- 7 typů
  - Reset, Undefined instructions, SWI, Prefetch Abort, Data Abort, IRQ, FIQ
- průběh
  - výjimka → přepnutí módu procesoru (+přepnutí banked registrů) → skok na exception vector (pevná adresa odpovídající vyvolané výjimce)
- problém – banked registry (sp)
  - HelenOS: 1 kernelovský zásobník
    - přepnutí na sp z předchozího módu nebo z proměnné (přišla-li výjimka z user módu)
  - ARM4: komplikovaný přístup k registrům z jiných módu (exception stack)

# Přerušení na TestARMu

- ARM sice definuje IRQ exception, ale nedefinuje, jak se zjistí zdroj přerušení
  - v reálném HW: interrupt controller
- v TestARMu interrupt controller nebyl
  - nepříjemné zjištění
- open-source + průhledná architektura GXemulu
  - vlastní jednoduchý paměťově mapovaný interrupt controller
    - maska aktivních přerušení
    - maskování přerušení

# Správa paměti

- vlastnosti ARM CPU
  - MMU – HW stránkovací tabulky, více formátů
  - TLB – bez podpory ASID
- HelenOS
  - zóny – správa stránek, nejnižší generická vrstva
  - buddy systém – součást zóny, pro frame_alloc
  - slab alokátor – obecný kernelový alokátor paměti
  - výpadky stránek

# Stránkovací tabulky

- v registru řídícího koprocesoru je uložena **fyzická adresa** stránkovací tabulky 1. úrovně
- 1MB, 64KB, 4KB, 1KB stránky
  - 1. úroveň - 4096 položek * 4B záznam = 16KB
    - coarse page table (64 KB, 4KB stránky)
      - 2. úroveň: 256 položek * 4B = 1KB
    - fine page table (1KB stránky)
      - 2. úroveň: 1024 položek * 4B = 4KB
    - sections (1MB)
      - pouze jedna úroveň
  - 2. úroveň – 3 formáty položek
    - large, small, tiny ... dle velikosti stránek

# Stránkovací tabulky

- bootstrap, inicializace kernelu
  - mapování 1:1, horní 2GB kopie dolních
  - použity 1MB stránky
    - jen jedna tabulka – jednoduché na nastavení
    - jedno pevné místo ve výsledné binárce
- po inicializaci
  - 4KB stránky, rozumný kompromis velikosti
  - coarse page tables, level 2 small descriptors

# Práva a domény

- domény
  - ovlivňují testování přístupových práv
  - 16 nezávislých
  - pouze v položkách první úrovně → po 1MB blocích
  - nepoužíváme
- práva
  - nastavitelná s granularitou na ¼ stránky
  - nejsou všechny kombinace přístupů, není Execute
  - kernel RW & (user No | user Read | user RW)

# Paměťové výjimky

- ARM má 2 typy výjimek
  - data abort – snaha o R/W na neplatné adrese
  - prefetch abort – vykonání instrukce na nepl. adrese
  - při výpadku stránky v registrech koprocesoru
    - adresa (kam se přistupovalo)
    - proč nelze přistoupit – práva, neexist. mapování, doména
- HelenOS má “high level page fault handler”
  - hlavním úkolem low level handleru, je zjistit co se stalo a zavolat high level handler

# Přechod do user režimu

- pouze pro “user” vlákna
- je třeba nastavit
  - stack
  - neprivilegovaný mód procesoru
  - program counter
  - vynulovat zbylé registry
    - mohou obsahovat “citlivé věci”

# TLS – Thread Local Storage

- všechna vlákna 1 procesu sdílejí 1 paměťový prostor, tedy i (globální) proměnné
- co když je potřeba jiná hodnota pro různá vlákna → použiji TLS
- `__thread int var; // GCC syntax`
- s těmito proměnnými zachází překladač zvláštním způsobem
- pro správnou funkci je potřeba podpora runtimu

# TLS – Thread pointer

- při přístupu k `__thread` proměnné se "dohodnutým" způsobem zjistí adresa TLS
  - na ARMech – zavolá se funkce **`__aeabi_read_tp`**
  - na novějších ARMech speciální registr koprocesoru
- kde může být schována adresa TLS?
  - segmentové registry – ia32, amd64
  - **obecný registr** – RISCové CPU (MIPS - k1)
  - spodek zásobníku

# ARM TLS

- v přeloženém souboru sekce .tdata, .tbss
- při vytvoření vlákna
  - alokace místa (dle velikosti sekcí .tdata + .tbss)
  - okopírování .tdata sekce, zero fill .tbss sekce
  - uložení adresy TLS do registru r9
- při přístupu k `__thread` proměnné
  - zavolá se funkce **`__aeabi_read_tp`**, vrátí adresu začátku TLS (u nás vrací obsah registru r9)
  - přičte offset v rámci .t* sekce a přistoupí k datům

# “Drobnosti”

- atomické operace (instrukce SWP)
- identifikace procesoru (výrobce, typ)

# Stav

- na GXemul TestARMu funguje
  - kernel testy + userspace (tetris ☺)
- připraveno rozhraní pro podporu dalších simulátorů či reálných strojů