*Vitaly Wool, Softprise Consulting OÜ*

# Spreading the disease: Linux on microcontrollers

# This presentation

- **Linux on an MCU**
- Task
- Solution

# Modern MCUs

- AVR or Cortex-M
- tight integration
- limited RAM and flash
- low or ultra-low power consumption
- easy to design hardware

# Example usage

- Home automation
- Industrial automation
- low-power wireless accessories

# OS choice: RTOS or… Linux?

- RTOS
  - mostly non-free
  - often POSIX-incompliant
  - may require expensive development tools
- Linux
  - free and open-source
  - well-developed applications
  - stable and portable
  - free tools (compiler, IDE, debugger)

# Linux on an MCU: obstacles

- **Storage requirements**
- Performance considerations

# Linux storage requirements

- Requirements are vague and architecture-dependent
- Plain case storage estimations
  - RAM: 8+ MB (+something for userspace)
  - Persistent: 2+ MB (+some for filesystem)
- Estimations for XIP
  - RAM: 1+ MB (+something for userspace)
  - Persistent: 4+ MB (+some for filesystem)
- XIP is supported only on some platforms
  - ARM is one of these

# Linux on an MCU: obstacles

- Storage requirements
- **Performance considerations**

# Linux performance considerations

- NB: CPU frequencies are lower than 200 MHz
- Boot-up time
  - 3+ seconds
- Latencies
  - generally higher than for an RTOS
- Overhead
  - generally bigger than for an RTOS

# Linux on an MCU: so…?

- Vanilla Linux kernel will not run on a basic MCU
- Something needs tweaking
  - add external RAM/flash to the design?
  - optimize Linux kernel/userspace to run tight?
- Is it worth it?

# This presentation

- Linux on an MCU
- **Task**
- Solution

# Home automation device

- Requirements
  - Ultra low power
  - PoE capable
  - Monitors BLE (Bluetooth Low Energy) sensors
  - [Secure] updates using USB stick or uSD

# HW design considerations

- no DRAM
- MCU rather than CPU
  - ARM Cortex-M4 is a good match
  - included Ethernet support is preferable
- standalone BLE chip

# MCU considerations

- ARM Cortex-M4 based solutions
  - Freescale Kinetis
    - up to 128 KB RAM
    - up to 2MB flash
  - STMicro STM32F4XX
    - up to 256 KB RAM
    - up to 2MB flash

# Software considerations

- OS should be POSIX compliant
- Tools should be FOSS

# This presentation

- Linux on an MCU
- Task
- **Solution**

# Proposed hardware design

- Considerations
  - ARM Cortex-M4
  - only SRAM, no DRAM
    - only NOR flash (possibly uSD/eMMC)
- Selection
  - STMicro STM32F29 (more SRAM than on others)
    - 256 KB SRAM
    - 2 MB NOR flash

# ARM Linux evaluation

- Vanilla Linux kernel is too large
  - both in terms of RAM and flash usage
    - Search for projects that optimize kernel for size
- XIP kernel is a must
  - RAM is no more than 256K
  - but the flash image size is bigger

# Linux for Cortex-M 1.12.0

- 2.6.33 based
- supports Cortex-M0, M3, M4
- supports Thumb mode
- recommended by Emcraft

# Linux kernel

- All unnecessary stuff removed
- Thumb mode on
- vmlinux: 1042816 bytes
  - vmlinux.text: 878460 bytes
  - vmlinux.data: 61440 bytes
  - vmlinux.bss:  48624 bytes

# Linux root filesystem

- Root filesystem: XIP/squashfs
  - executables come uncompressed
- Size: 851968 bytes
- We need 64+kb for RW filesystem (JFFS2)
  - further optimization needed

# Linux kernel optimizations

- Compress .data
  - compressed size: 8220 bytes
- remove .bss
  - just need to zero the RAM area

# Resulting flash map

- Bootloader:  0x10000 bytes
- XIP Kernel: 0xE0000 bytes
- Root tilesystem (RO): 0xD0000 bytes
- Config filesystem (RW): 0x30000 bytes
- …and we even have 64k left :)

# Thanks!

- Questions?
  vitaly.wool@softprise.net