NuttX for Embedded Linux Developers

Masayuki.Ishikawa@sony.com
Senior Software Engineer
Sony Home Entertainment & Sound Products Inc.
Agenda

- Sony audio products based on NuttX
- Feasibility studies for NuttX enhancements
  - SMP
  - Networking
  - Porting the AVS device SDK
- Support for Sony Spresense board
- NuttX workshop
About Me

Senior Software Engineer at Sony Home Entertainment & Sound Products Inc.

Technical background
- 3D graphics, home networking, Internet-to-Home, Embedded Systems
- Portable Media Player (Linux/Android)
- Digital voice recorder, music player, head phone (NuttX)

Product development
- Portable Media Player (Linux/Android)
- Digital voice recorder, music player, head phone (NuttX)

Public talks
About NuttX (1/2)

- Gregory Nutt released in 2007 as an open source

- Mostly POSIX-compliant* real-time OS
  - Suitable for robotics ** including Drones.

- From 8bit to 32bit CPU are supported
  - Z80, x86, Arm7/9/11, Cortex-Mx/Ax/Rx, AVR, MIPS, RISC-V, Xtensa,...
  - Many evaluation boards (over 180) are supported

- Small footprint and runtime memory
  - Suitable for resource constraint devices

---

* Linux is also a mostly POSIX-compliant OS

** NuttX is the default RTOS for ROS2 embedded
About NuttX (2/2)

- Almost all code are written from scratch*
  - Straightforward and consistent without vendor HAL

- Many key features are implemented
  - Virtual file systems, loadable module, tickless, SMP
  - FAT12/16/32, SmartFS, romfs, procfs, NFS (client only)
  - Networking (IPv4, IPv6, UDP, TCP, ...)
  - USB (CDC/ACM, MSC, HID, RNDIS, ...),

- Many example applications are included
  - NuttX shell, webserver and client, telnet daemon,...

* Some drivers are ported from other open source
Activities

- Sony audio products based on NuttX
  - Development history
  - Audio products
  - Reasons for choosing NuttX

- Feasibility studies for NuttX enhancements
Development history*

- Oct 2013 -
  - Ported NuttX to LC823425 (ARM7)
- Apr 2014 -
  - Ported bluetooth stack to NuttX + QEMU
- Jul 2014 -
  - Ported NuttX to LC823450 (Cortex-M3) FPGA
- Jan 2015 -
  - Migrated to LC823450-ES board
- Sep 2015 -
  - Released the first NuttX-based audio products.
- Oct 2016 -
  - Talked at Arm TechCon 2016, ELC NA 2017*-2019, NuttX 2019

*https://www.youtube.com/watch?v=TjuzH6JthxQ  ** https://www.youtube.com/watch?v=T8fLjWyl5nI
NuttX based audio products

FY2015
- Voice Recorders
- Music Player

FY2016,17
- Voice Recorders
- Wireless Headphones

FY2018
- Wireless Headphones
- PCM Recorders
Why we chose NuttX for audio products

- POSIX and libc are supported
  - Can reuse existing software
  - Can reduce learning costs
- ELF* is supported
  - Can divide a big app into small apps
- Driver framework is supported
  - Helps us implement drivers
- Has Linux-like configuration system
  - Helps us develop multiple products
- Many MCUs and boards are supported
  - Helps us port NuttX to new MCU
- Provided with BSD license

* ELF = Executable and Linking Format

From http://www.nuttx.org/
LC823450 Features

- ARM dual Cortex-M3
- 32bit fixed point, dual-MAC original DSP
- Internal SRAM (1656KB) for ARM and DSP
- I2S I/F with 16/24/32bit, MAX 192kHz (2chx2)
- Hard wired audio functions
  - MP3 encoder and decoder, EQ (6-band equalizer), etc.
- Integrated analog functions
  - Low-power Class D HP amplifier, system PLL
  - Dedicated audio PLL, ADC
- Various interfaces
  - USB2.0 HS device / host (not OTG), eMMC, SD card, SPI, I2C, etc.
- ARM and DSP clock max frequency
  - 160MHz at 1.2V
  - 100MHz at 1.0V

From http://www.onsemi.com/PowerSolutions/product.do?id=LC823450
Why SMP with LC823450?

- **Motivation**
  - Run *existing applications* in SMP mode
  - Establish knowledge on debugging
  - Confirm effects on *memory contention* because the processor *does not have* CPU cache.
  - Confirm power consumption
  - Very challenging theme (because NuttX is not just a scheduler)

- **Other reasons…**
  - The architecture is much simpler than quad Cortex-A9.
  - Suitable system to understand SMP kernel.
Supporting SMP on the processor

- Port existing drivers to the latest NuttX
  - UART, Timer, GPIO, DMA, I2C, SPI, LCD
  - eMMC (including boot), SD, USB, ADC, …
- Implement SMP related code
  - lc823450_cpubusidlestack.c, lc823450_cpuindex.c
  - lc823450_cpupause.c, lc823450_cpustart.c, lc823450_testset.c (NOTE: H/W Mutex is used instead of ldex, strex)
- Performance improvement
  - Introduced spin_lock_irqsave(), spin_unlock_irqrstore()
  - Applied APIs inside the driver code.
  - Up to 20% performance improvement achieved
Feasibility studies for networking

- Adding Networking
- NuttX networking features
- USB RNDIS and Bluetooth PAN
Adding Networking

- **Motivation**
  - Confirm NuttX network stack feasibility
    - IPv4, IPv6, ICMP, UDP, TCP, …
  - Run the network stack with minimum efforts. (We already have a USB driver for LC823450)
  - Audio streaming (PCM and MP3)
  - Run the network stack in SMP mode
  - Do various tests via telnet

LC823450XGEVK board
NuttX networking features

- Ethernet and IEEE 802.11 Full MAC
- 6LoWPAN for radio network drivers (IEEE 802.15.4 MAC)
- USB RNDIS (since 7.23), CDC-ECM (since 7.26)
- SLIP, TUN/PPP, local loopback devices
- IPv4, IPv6, TCP, UDP, ARP, ICMP, ICMPv6, IGMPv2
- IP forwarding
- BSD compatible socket layer
- DNS name resolution / NetDB
- User socket (listen/accept are supported in 7.26)
- Bluetooth socket
PCM audio streaming via RNDIS

- Fix RNDIS driver for NuttX
  - Fix data corruption
  - Add USB high speed mode support
- Receive window control has been added
  - Needs more improvement
- Modify nxplayer to support HTTP streaming
  - Currently only WAV format is supported.
- Still testing with SMP kernel
  - In various conditions (clock speed, network traffic, etc)
MP3 audio streaming via Bluetooth

- Port the BTstack* by Bluekitchen to NuttX
  - Based on posix-h4** with H/W flow control
  - UART speed: 921600 baud
  - Tested with iOS/Android/macOS/OpenWrt
  - Free for non-commercial use

- Add TAP mode to the NuttX tun driver
  - TAP mode is used for network bridge
  - NOTE: TUN mode is used for network routing

- Add H/W MP3 decoder to lc823450_i2s.c

- HCI_RESET issue in SMP mode
  - CSR’s mode change with HCI_RESET is tricky
  - Still unstable in SMP mode

---

* https://bluekitchen-gmbh.com/
** We can use posix-h5 (3-wire protocol) as well. However, it has performance drawbacks.
Running the BTstack on NuttX

Profiles:
- GATT
- SPP
- HSP
- HFP
- MFI
- SDAP
- PAN
- GAP

Protocols:
- SMP
- ATT
- RFCOMM
- SDP
- BNEP

Transport:
- HCI
- L2CAP LE
- L2CAP

Profiles:
- tinyplay
- renew (dhcp client)

Protocols:
- TCP
- UDP
- IPv4
- ICMP
- ARP

Transport:
- tun (bnep0)
- UART

OpenWrt
- PAN-NAP* role
- DHCP server

WZR-HP-G300NH

BT PAN profile

*PAN: Personal Area Network
*BNEP: Bluetooth Network Encapsulation Protocol
*NAP: Network Access Point
Porting the AVS* device SDK to NuttX

- **Motivation**
  - Confirm OSS portability with large software systems

- **Approaches**
  - Build the AVS device SDK on Linux
  - Run the AVS device SDK on Linux
  - Port OSS components such as mbedTLS to NuttX
  - Implement missing components such as MediaPlayer with minimal efforts
  - Reduce runtime memory on NuttX

* AVS stands for Alexa Voice Service
Inside the AVS device SDK

**PCM:** 16kHz, 16bit, mono

**MPEG Audio:** 24kHz, mono, 48kbps

**HTTP2 over TLS**

ECDHE-RSA-AES256-GCM-SHA384

*Push-to-talk was used to test avs device sdk instead of Key Word Detection*

**Source code for GStreamer and AndroidSLES are available**
Software stack

*NxMediaPlayer and NxMicWrapper were newly implemented for LC823450
On Linux, GStreamer and PortAudio are used to playback and capture audio

Total code size is about 3.6MB
Network topology

avd-device-sdk 1.7.1
NuttX 7.26 (CONFIG_SMP=n)

HTTPS Proxy : squid
Guest OS : ubuntu 18.04

* VAIO S11 + VirtualBox
Host OS : Windows 8.1

* MacBook Pro + VMWare Fusion can be used as well
Activities

- Spresense overview
- Spresense SDK
- Upstreaming status
- Working with Wi-Fi
Spresense overview

- **Application Domain**
  - Cortex-M4F x6 + SRAM 1.5MB
  - eMMC/USB/SDIO

- **System and IOP Domain**
  - System management
  - Power domain control

- **Sensor Domain**
  - Sensor Engine + FIFO (40KB)

- **GNSS Domain**
  - Independent GNSS positioning

Spresense (cxd5602) is here (size 6.5x6.5mm)
Spresense SDK*

* code is available at https://github.com/sonydevworld/spresense
<table>
<thead>
<tr>
<th>CORE</th>
<th>COMPLETED (July Workshop)</th>
<th>COMPLETED (October Meetup)</th>
<th>PLANNED (January Meetup)</th>
</tr>
</thead>
<tbody>
<tr>
<td>adc</td>
<td>pwm</td>
<td>GNSS</td>
<td>MODEM</td>
</tr>
<tr>
<td>allocateheap</td>
<td>rtc</td>
<td>cpu1signal</td>
<td>altdmd</td>
</tr>
<tr>
<td>clock</td>
<td>scu</td>
<td>geofence</td>
<td>altdmd_spi</td>
</tr>
<tr>
<td>composite</td>
<td>sdcard</td>
<td>gnss</td>
<td>audio_baseband</td>
</tr>
<tr>
<td>cpuif0</td>
<td>sdhci</td>
<td></td>
<td></td>
</tr>
<tr>
<td>delay</td>
<td>serial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dmac</td>
<td>spi</td>
<td></td>
<td></td>
</tr>
<tr>
<td>farapi</td>
<td>spisd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>flash</td>
<td>syscall</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gpio</td>
<td>timer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gpiooint</td>
<td>timerisr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>i2c</td>
<td>uart</td>
<td></td>
<td></td>
</tr>
<tr>
<td>i2cdev</td>
<td>udmac</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gpioif</td>
<td>uid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>icc</td>
<td>usbddev</td>
<td></td>
<td></td>
</tr>
<tr>
<td>idle</td>
<td>usb msmc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>leds</td>
<td>userleds</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pinconfig</td>
<td>Wdt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pmic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>powermgr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WIFI</td>
<td>gs2200m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENSORS (I2C)</td>
<td>ak09912</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bmi160</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bmp280</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISPLAY</td>
<td>ili9340</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>lpm013m091a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAMERA</td>
<td>video</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cisif</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>isx012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra HW</td>
<td>buttons</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>charger</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>emmc</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>gauge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>ge2d</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Working with Wi-Fi

- **Wi-Fi module**: Telit GS2200M
  - Radio protocols: **802.11b/g/n (2.4GHz)**
  - Interface: **SPI 10MHz with DMA**

- **Implement GS2200M driver from scratch**
  - Based on the NuttX usrsock
  - Both STA and AP modes are supported
  - Fix cxd56_gpioint.c for interrupt handling
  - TCP and UDP are supported

- **Modify the uIP webserver app for NuttX**
  - Add a directory listing feature

---

* The code is available at https://bitbucket.org/nuttx/nuttx
What is the usrssock?

- User-space networking stack API defined in NuttX
- User-space daemon and HAL provide NuttX networking
- This allows **seamless integration of HW-provided TCP/IP stacks to NuttX**
Use-case for Webserver via Wi-Fi

Network applications

- DHCP client
- DNS client
- telnetd
- webserver
- gs220m daemon
- /dev/mmcasd0
- /dev/ussrsock
- /dev/gs2200m

The latest NuttX upstream + Spresense

Web browsing with Firefox

<table>
<thead>
<tr>
<th>Name</th>
<th>Last modified</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>audio/</td>
<td>2019-07-05 16:05</td>
<td>-</td>
</tr>
<tr>
<td>ps.txt</td>
<td>2019-07-05 17:06</td>
<td>814</td>
</tr>
<tr>
<td>.sevented/</td>
<td>2019-07-05 17:48</td>
<td>-</td>
</tr>
<tr>
<td>Spotlight-V100/</td>
<td>2019-07-03 09:24</td>
<td>-</td>
</tr>
<tr>
<td>spresense_header_code.jpg</td>
<td>2019-07-05 17:48 107550</td>
<td>-</td>
</tr>
<tr>
<td>.trashcan/</td>
<td>2019-07-05 16:04</td>
<td>-</td>
</tr>
<tr>
<td>hello.txt</td>
<td>2019-07-05 16:14</td>
<td>8</td>
</tr>
</tbody>
</table>

OpenWrt

WZR-HP-G300NH

VirtualBox + Ubuntu

Code size:
- wifi: .text=157,234, bss=7,956
- rndis*: .text=166,924, bss=15,384

* USB MSC was removed to compare, because it is not used in Wi-Fi configuration
NuttX workshop

- About NuttX2019
- Applications Introduced at NuttX2019
- Why they chose NuttX
About NuttX2019

- The 1st international workshop for NuttX
- Held in Gouda, the Netherlands (Jul/16,17)
- Over 40 people joined

* Slides are available at https://nuttx2019.org/slides-and-videos/
Applications* introduced at NuttX2019

Drone by Verge Aero
Pixhawk (flight controller)
NXP drone reference design
micro-ROS Kobuki by Bosch

Personal Alert system by Hexagon
CubeSat by STARA

TagMaster by TagMaster AB

*We also introduced audio products and Spresense.
Why they chose NuttX

Why to use NuttX for IIoT:

- Linux compatibility;
- Broad range of features;
- POSIX compliance (easy to port applications);
- Easy to move to high-end OS in the future;
- Flexibility to move among supported MCUs;

Why NuttX?

- Focus on standards
  - Posix
  - Linux compatibility (where possible)
- RTOS
- Vendor Neutral - many MCUs are supported
- Most applications can be written and tested entirely in Linux first
- C++ support
- Full network stack

Why PX4 Chose NuttX

- The BSD Licensing - "BSD licenses are a family of permissive free software licenses, imposing minimal restrictions on the use and distribution of covered software. This is in contrast to copyleft licenses, which have share-alike requirements. The original BSD license was used for its namesake, the Berkeley Software Distribution (BSD)"
- "The Portable Operating System Interface (POSIX) is an IEEE standard that helps compatibility and portability between operating systems. Theoretically, POSIX compliant source code should be seamlessly portable. In the real world, application transition often runs into system specific issues"
- Real Time OS
- The scalability and degree of freedom to which it can be modified to suit application specific needs, from small footprint to large
- Code Quality and conformity

Leveraging POSIX compliance

- Application can be run under NuttX on embedded target and under Linux
- Implementing, debugging and testing application code under Linux
  - huge development speed-up
- Plenty of documentation for POSIX APIs available online
- Unit testing under Linux is straightforward
- Application code can be integrated in a Linux-based simulation framework
Demo videos

- LC823450 + AVS Device SDK
- Spresense + Wi-Fi
- LC823450 + HTTP Audio streaming in SMP
Any Questions?