SMP and Networking support on NuttX / LC823450

Masayuki.Ishikawa@sony.com
Senior Software Engineer

Sony Home Entertainment & Sound Products Inc.

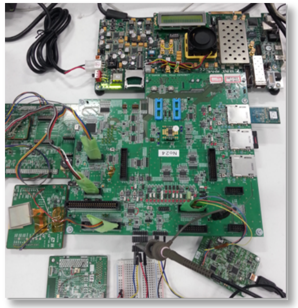
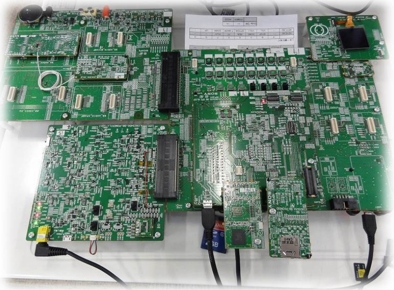
Agenda

- Development history (NuttX-based products)
- SMP (Symmetric Multiprocessing) related status
- Networking related status
- Demo videos

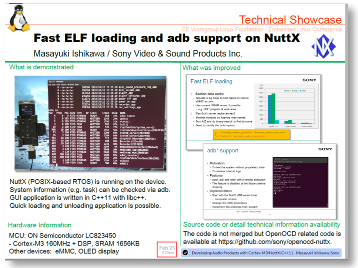
Development history*(NuttX-based products)

SONY

- 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 ** and OpenIoT NA 2018

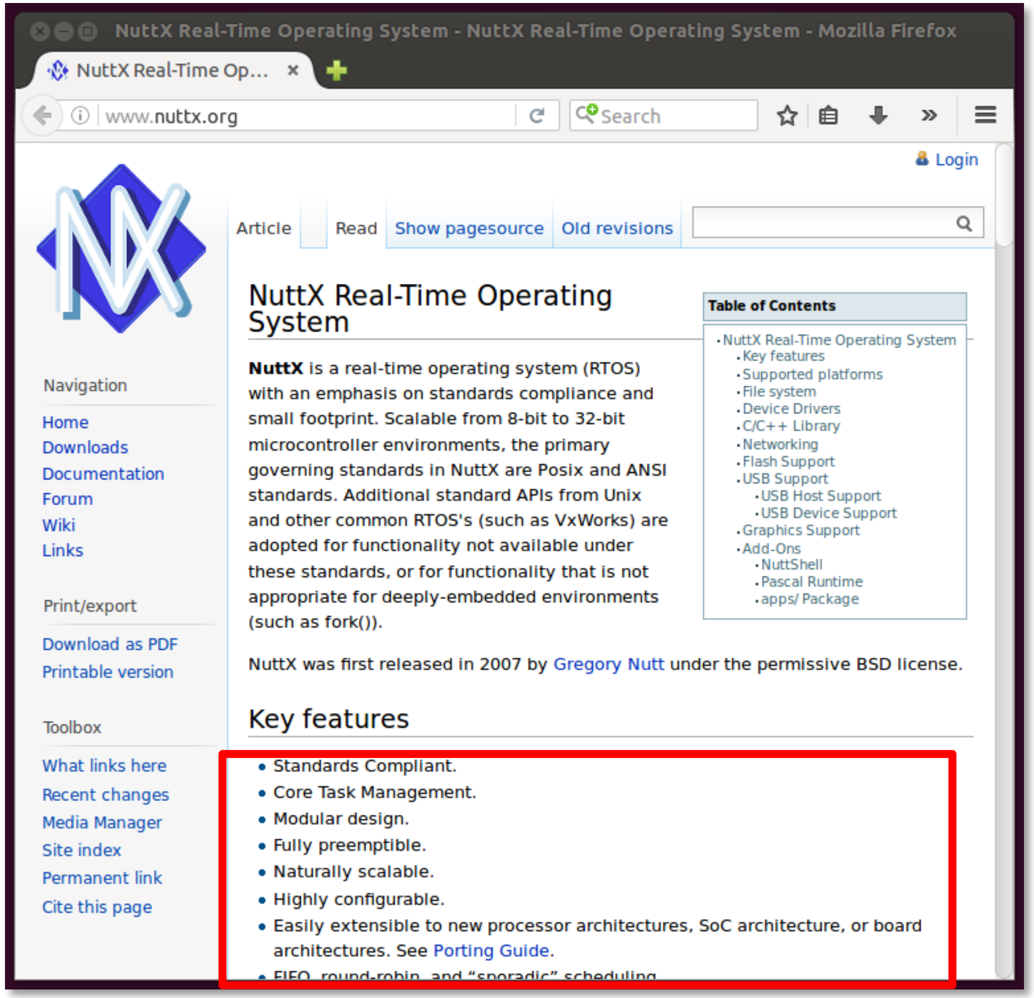




*https://www.youtube.com/watch?v=TjuzH6JthxQ ** https://www.youtube.com/watch?v=T8fLjWyI5nI

About NuttX and why we chose it

- POSIX and libc are supported
  - Can reuse existing software
  - Can reduce training costs
- ELF* is supported
  - Can divide 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



From http://www.nuttx.org/

* ELF = Executable and Linking Format

LC823450 Features

SONY

- 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

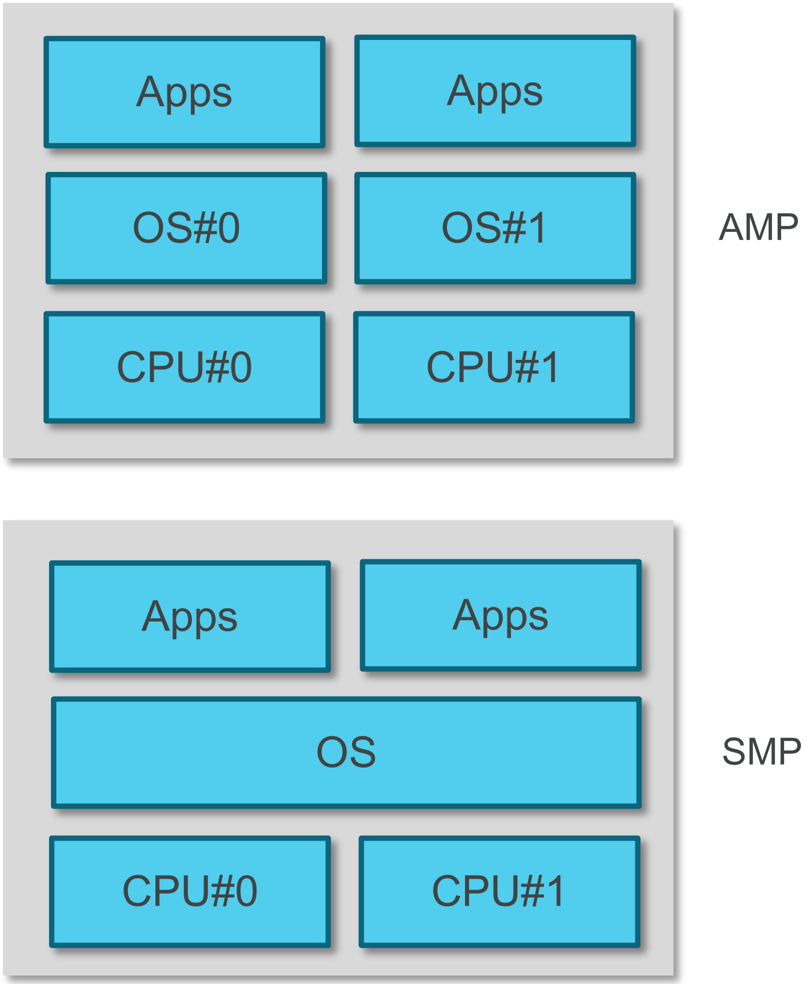
ON Semiconductor LC823450

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

AMP vs SMP in general *

SONY

- Asymmetric multiprocessing (AMP)
  - A separate OS, or a separate copy of the same OS, manages each core.
  - Provides an execution environment similar to that of uniprocessor system, allowing simple migration of legacy code. Also allows developers to manage each core independently.
- Symmetric multiprocessing (SMP)
  - A single OS manages all processor cores simultaneously. The OS can dynamically schedule any process on any core.
  - Provides greater scalability and parallelism than AMP, along with simpler shared resource management

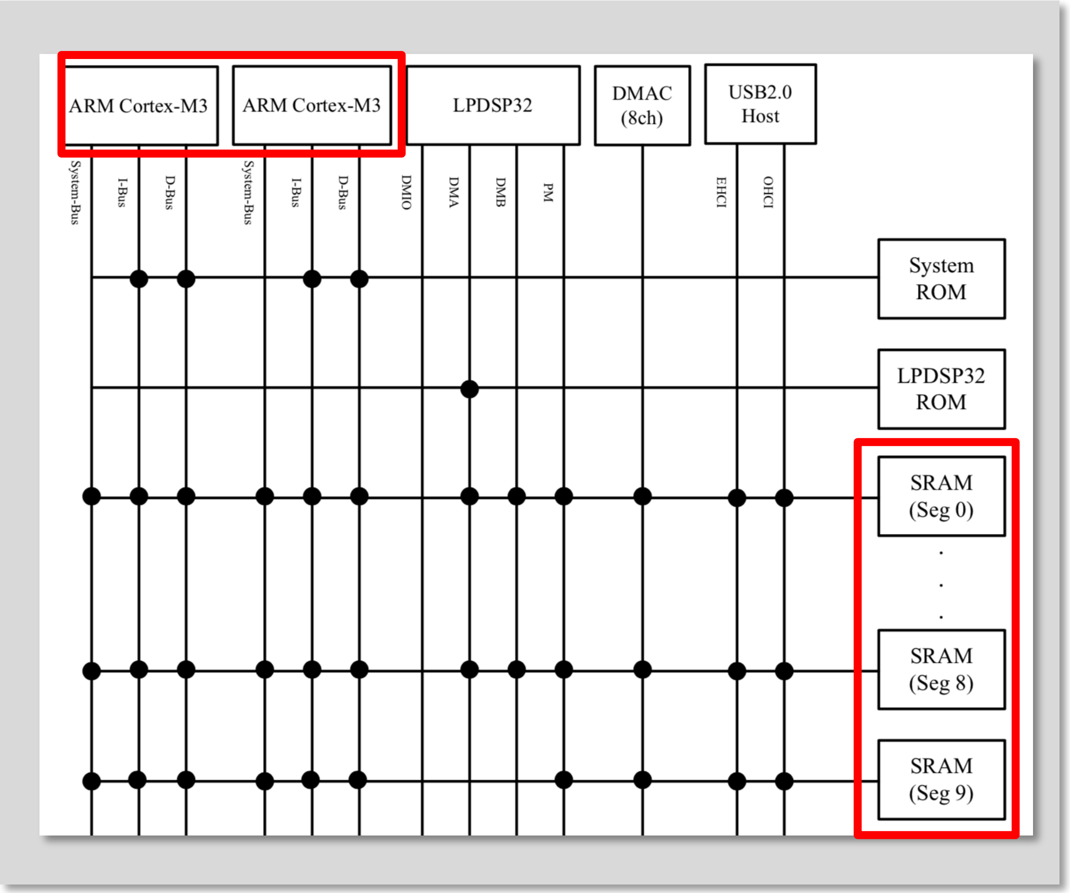


* http://www.embeddedintel.com/special_features.php?article=189

Why SMP with LC823450?

SONY

- Motivation
  - Run existing applications in SMP mode
  - Establish knowledge on debugging
  - Confirm performance penalty *
  - 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.

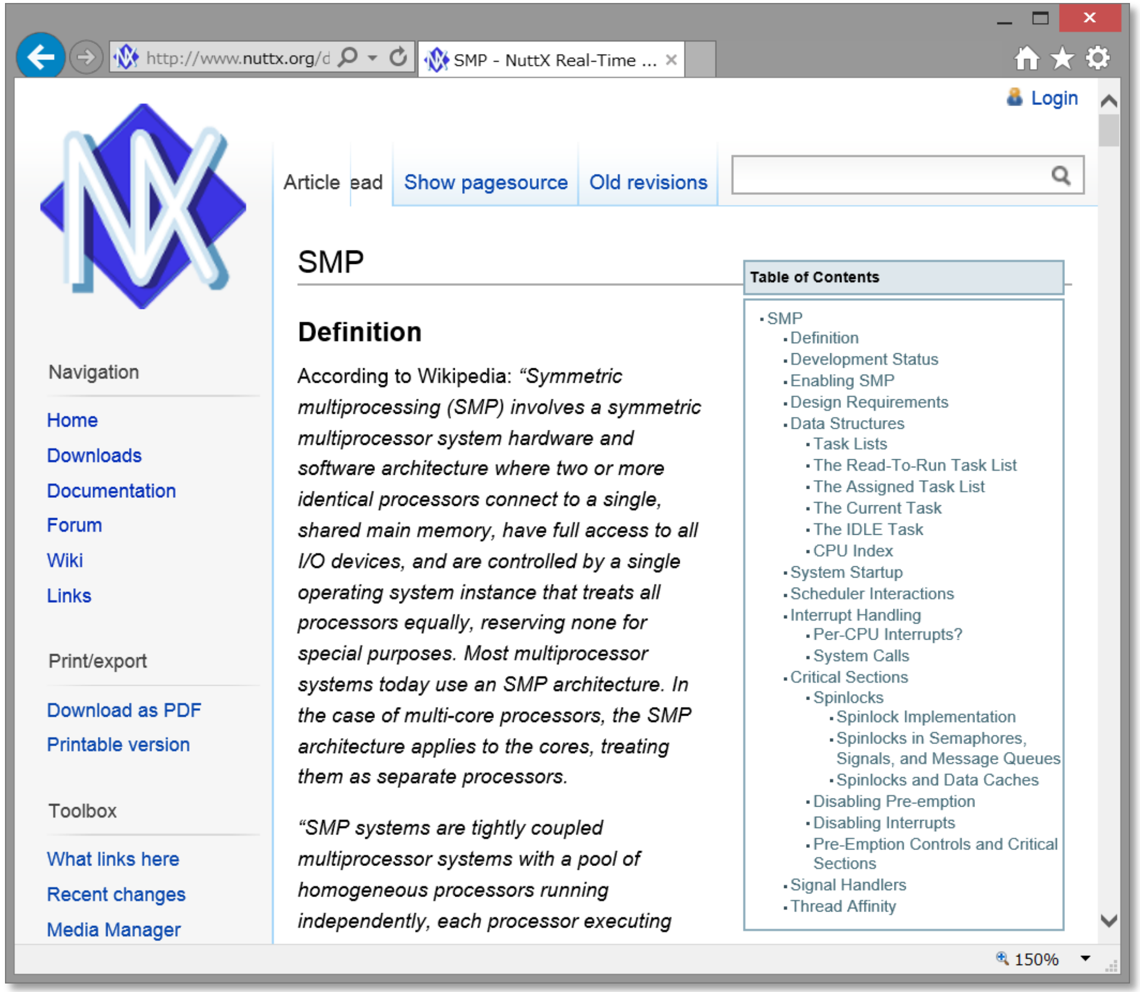


* Note that LC823450 does not have CPU cache but has multiple SRAM segments

Introduction to the NuttX SMP kernel

SONY

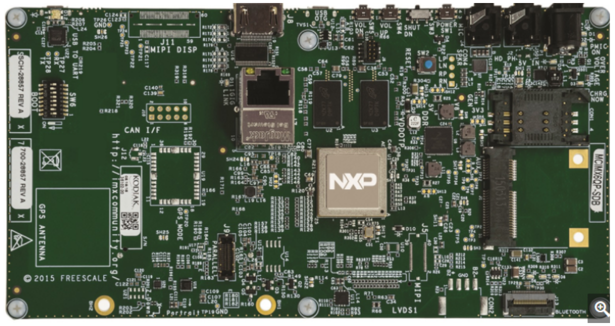
- Minimum changes to non-SMP kernel
  - CONFIG_SMP is introduced.
  - Main changes are done in the scheduler
- Newly introduced
  - Spinlock to protect shared resources
  - Critical section APIs to replace with local interrupt control APIs.
  - pthread_setaffinity_np(), sched_setaffinity() are supported
- H/W interrupts except for inter-CPU interrupts are assumed to be handled at CPU0
  - To prevent deadlocks



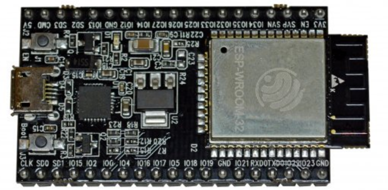
NuttX SMP : available boards

SONY

- NXP (Freescale) i.MX6 Quad Sabre
  - Quad Arm Cortex-A9
  - SMP kernel can run on QEMU *
- Espressif Systems ESP32
  - Dual Tensilica LX6 *
- Microchip (Atmel) SAM4CMP-DB
  - Arm Cortex-M4 w/MPU + Cortex-M4F *
- ON Semiconductor LC823450XGEVK
  - Dual Arm Cortex-M3
  - Approx. $46 **

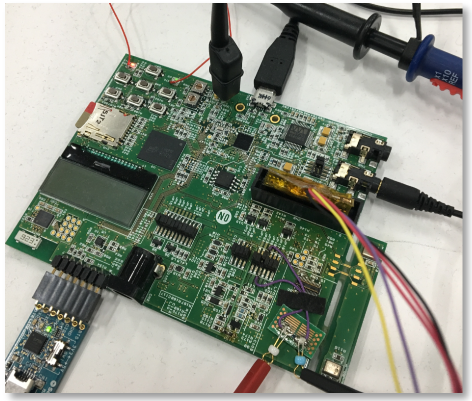


i.MX6 Quad Sabre



ESP32

SAM4CMP-DB



LC823450XGEVK

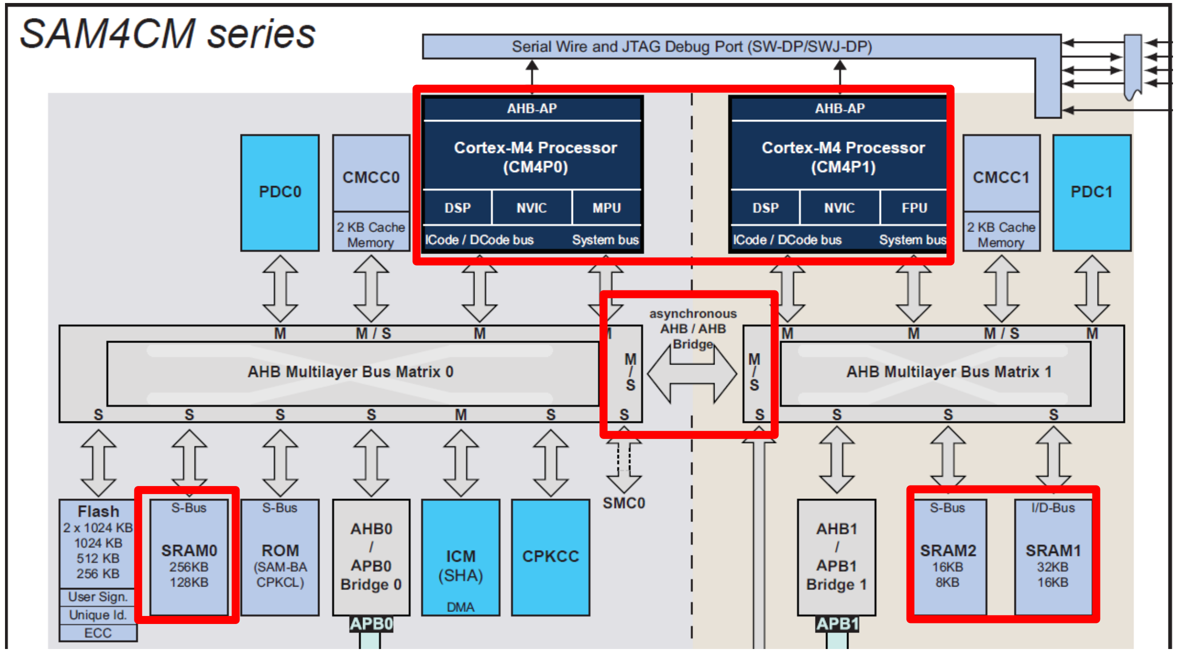
*ostest still has some issues.

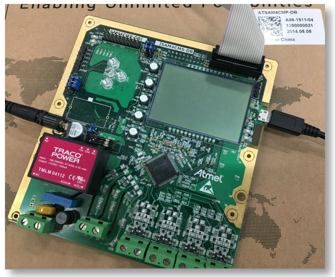
**http://www.components-center.com/product/ON-Semiconductor/LC823450XGEVK.html

Running SMP kernel : SAM4CMP-DB

SONY

- Cortex-M4 /w MPU + Cortex-M4F
  - Not symmetric, but if both CPU does not use MPU nor FPU, it should be OK.
  - Each CPU has local SRAM which can be accessed via bus bridge from another CPU.
- Bus bridge issue *
  - "ostest" crashes due to CPU lockup or hardfault
  - It's difficult to assure memory access just by memory barrier operations.
  - Dummy memory read/write might resolve this issue, but we still can not find the correct way.
  - We asked this issues to Atmel before, but no response received yet.





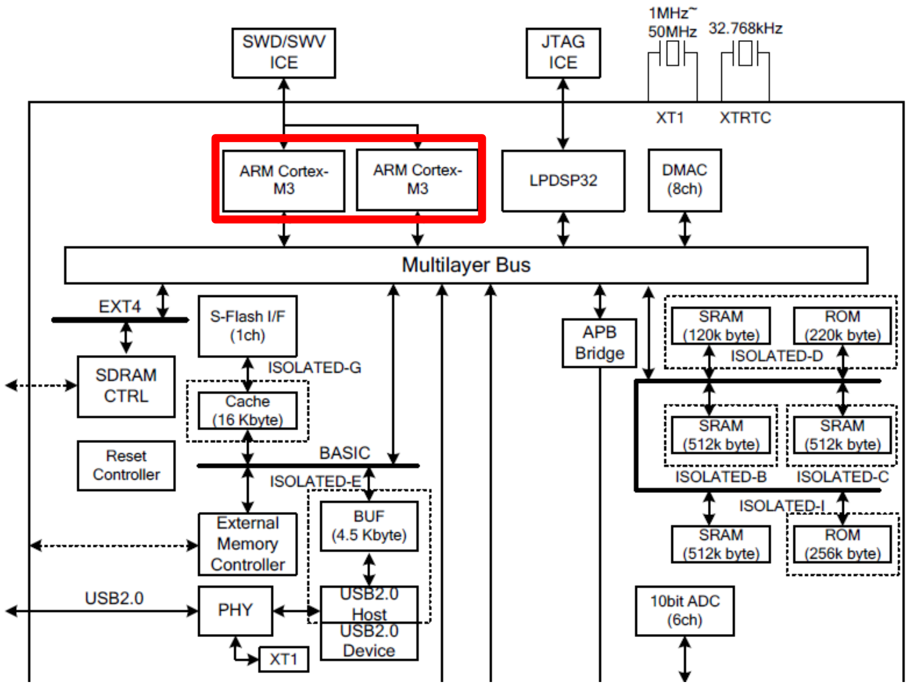
* I don't think this board can perfectly work in SMP mode

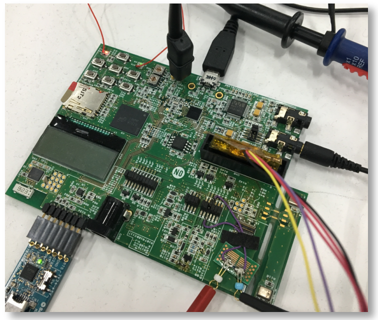
NuttX 2019 International workshop

Running SMP kernel : LC823450XGEVK

SONY

- 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_cpuidlestack.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





NuttX 2019 International workshop

Tracing SMP kernel

SONY

- What events can be traced
  - SMP specific (inter-CPU communication)
    - CPU_PAUSE, CPU_PAUSED, CPU_RESUMED
  - SMP/non-SMP common
    - SUSPEND, RESUME (context switch)
    - PREEMPT_LOCK, PREEMPT_UNLOCK
- Tools
  - Use gdb macro to dump the trace buffer
  - Use “noteinfo” to analyze the dump file

Terminal

File Edit View Search Terminal Help

```
664: 0b 06 f6 00 0b 00 9b 25 00 00 01       CPU0 PID 11: CPU_PAUSE
675: 0b 02 00 01 01 00 9b 25 00 00 04       CPU1 PID  1: SUSPEND
686: 0a 07 00 01 01 00 9b 25 00 00          CPU1 PID  1: CPU_PAUSED
696: 0b 08 f6 00 0b 00 9b 25 00 00 01       CPU0 PID 11: CPU_RESUME
707: 0a 09 32 01 04 00 9b 25 00 00          CPU1 PID  4: CPU_RESUMED
717: 0a 03 32 01 04 00 9b 25 00 00          CPU1 PID  4: RESUME
727: 0b 02 f6 00 0b 00 9b 25 00 00 06       CPU0 PID 11: SUSPEND
738: 0a 03 00 00 00 00 9b 25 00 00          CPU0 PID  0: RESUME
748: 0c 0a 32 01 04 00 9b 25 00 00 01 00    CPU1 PID  4: PREEMPT_LOCK
760: 0b 02 32 01 04 00 9b 25 00 00 07       CPU1 PID  4: SUSPEND
771: 0a 03 00 01 01 00 9b 25 00 00          CPU1 PID  1: RESUME
781: 0b 02 00 00 00 00 9b 25 00 00 03       CPU0 PID  0: SUSPEND
792: 0a 03 32 00 04 00 9b 25 00 00          CPU0 PID  4: RESUME
802: 0c 0b 32 00 04 00 9b 25 00 00 00 00    CPU0 PID  4: PREEMPT_UNLOCK
814: 0b 06 32 00 04 00 9b 25 00 00 01       CPU0 PID  4: CPU_PAUSE
825: 0b 02 00 01 01 00 9b 25 00 00 04       CPU1 PID  1: SUSPEND
836: 0a 07 00 01 01 00 9b 25 00 00          CPU1 PID  1: CPU_PAUSED
846: 0b 08 32 00 04 00 9b 25 00 00 01       CPU0 PID  4: CPU_RESUME
857: 0a 09 fc 01 0c 00 9b 25 00 00          CPU1 PID 12: CPU_RESUMED
867: 0a 03 fc 01 0c 00 9b 25 00 00          CPU1 PID 12: RESUME
877: 0b 02 fc 01 0c 00 9b 25 00 00 06       CPU1 PID 12: SUSPEND
888: 0a 03 00 01 01 00 9b 25 00 00          CPU1 PID  1: RESUME
898: 0b 06 32 00 04 00 9b 25 00 00 01       CPU0 PID  4: CPU_PAUSE
909: 0b 02 00 01 01 00 9b 25 00 00 04       CPU1 PID  1: SUSPEND
920: 0a 07 00 01 01 00 9b 25 00 00          CPU1 PID  1: CPU_PAUSED
930: 0b 08 32 00 04 00 9b 25 00 00 01       CPU0 PID  4: CPU_RESUME
941: 0a 09 fc 01 0c 00 9b 25 00 00          CPU1 PID 12: CPU_RESUMED
951: 0a 03 fc 01 0c 00 9b 25 00 00          CPU1 PID 12: RESUME
961: 0c 0a fc 01 0c 00 9b 25 00 00 01 00    CPU1 PID 12: PREEMPT_LOCK
973: 0c 0b fc 01 0c 00 9b 25 00 00 00 00    CPU1 PID 12: PREEMPT_UNLOCK
985: 0b 06 fc 01 0c 00 9b 25 00 00 00       CPU1 PID 12: CPU_PAUSE
:
```

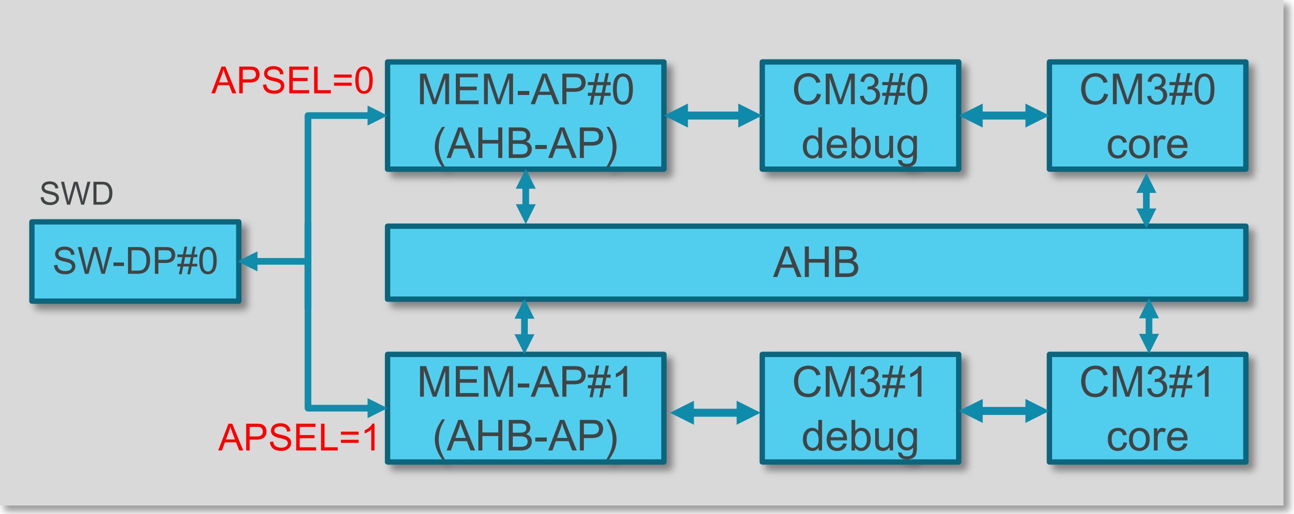

NuttX 2019 International workshop

OpenOCD for lc823450-smp*

SONY

- Implementation
  - Understand how Cortex-A SMP support works in OpenOCD
  - Modify several files (target/cortex_m.c …) to support Cortex-M in SMP mode
  - Specify APSEL (Access Port Selection) when accessing to each core in LC823450
  - Modify tcl/target/lc823450.cfg to support multiple debug access ports and targets.
  - Modify rtos/nuttx.c to show SMP related tasklists

```
Open On-Chip Debugger 0.10.0-dev-00610-gca7ae9cb-dirty (2017-07-03-14:24)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 300 kHz
Info : FTDI SWD mode enabled
cortex_m reset_config sysresetreq
Info : clock speed 300 kHz
Info : SWD IDCODE 0x2ba01477
Info : lc823450.cpu0: hardware has 6 breakpoints, 4 watchpoints
Info : lc823450.cpu1: hardware has 6 breakpoints, 4 watchpoints
lc823450.cpu1: target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x0204610e msp: 0x02016478
lc823450.cpu0: target state: halted
target halted due to debug-request, current mode: Handler External Interrupt(18)
xPSR: 0x01000022 pc: 0x02041cfe msp: 0x02001d68
```



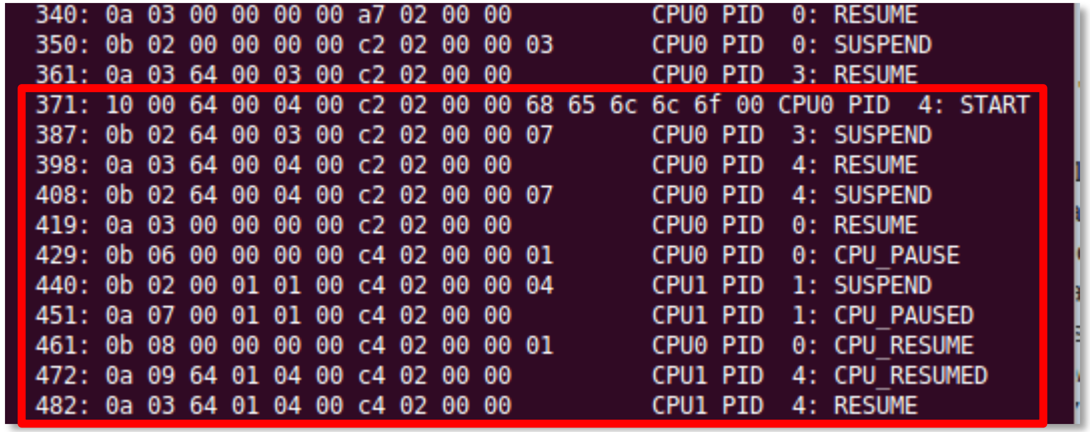
*Code is NOT merged yet.

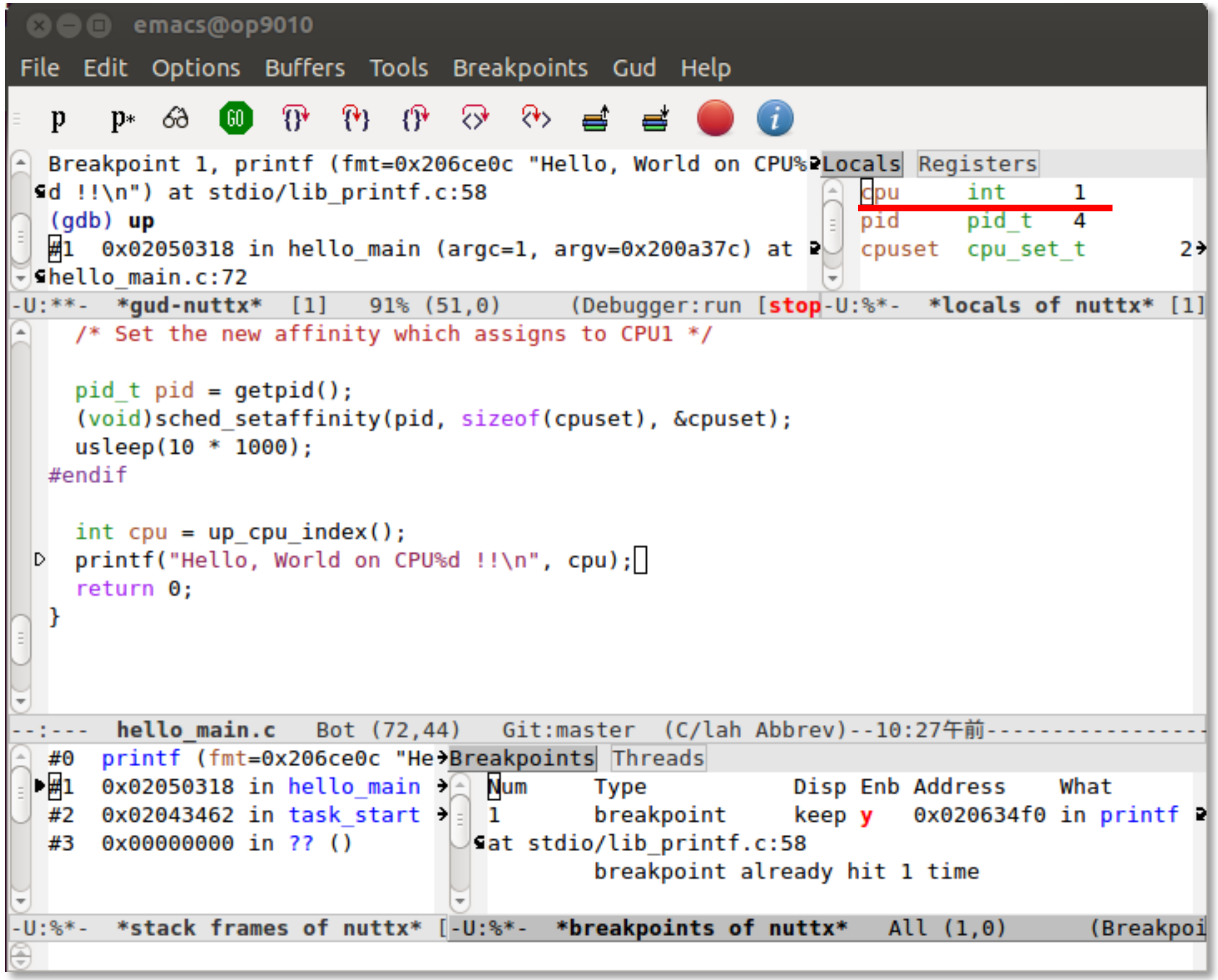
NuttX 2019 International workshop

Debugging an SMP application

SONY

- Modify hello_main.c
  - Assign the current task to CPU1 (not CPU0)
  - Print CPU index.
- Add a break point at printf()
- Run "hello" on the nsh
- Break point hits on CPU1
- Check the trace log



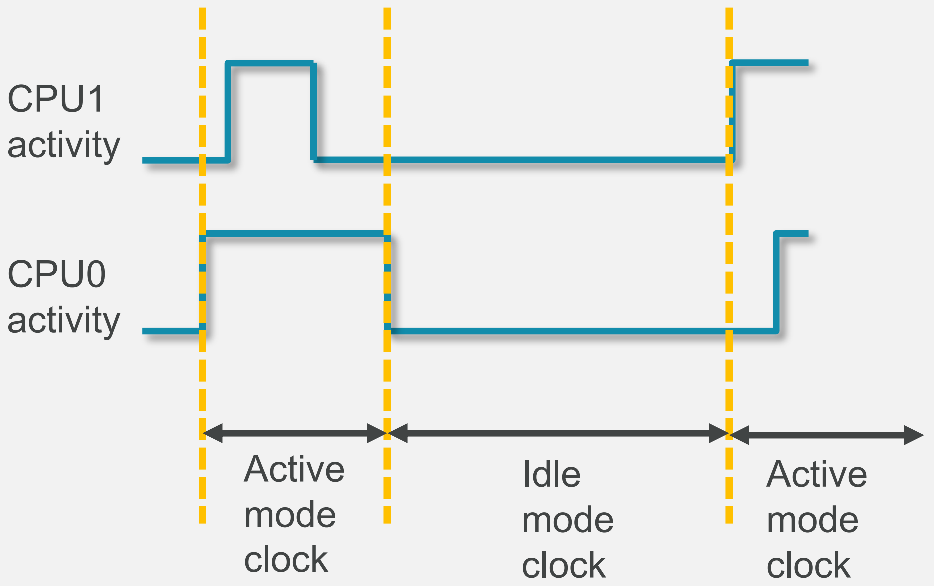


NuttX 2019 International workshop

Enhance DVFS* for SMP

SONY

- Need to handle both CPUs
  - 1. If at least one CPU is active, the apply active mode clock.
  - 2. If both CPUs are idle (i.e. WFI), then apply idle mode clock
- Calculate CPU idle time on both CPUs
  - 3. If at least one CPU falls below lower threshold (e.g. 20% idle), then go to higher clock mode.
  - 4. If both CPUs exceed higher threshold (e.g. 70% idle), then go to lower clock mode



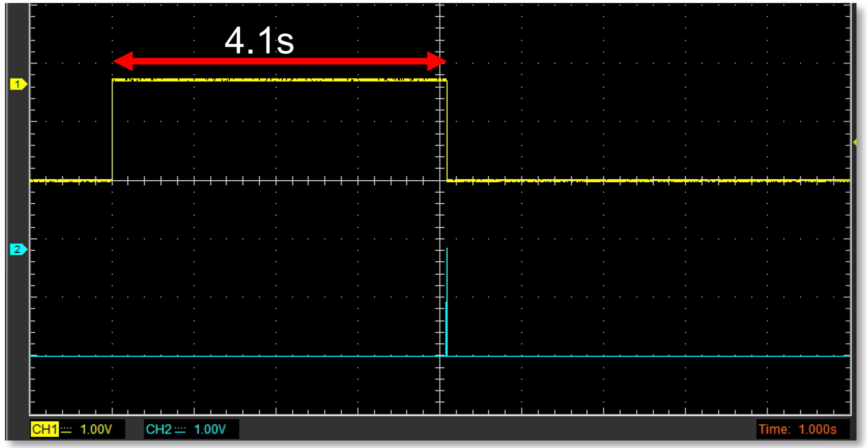
CPU0 idle	CPU1 idle	Next clock state
10%	80%	Go to higher clock mode
80%	10%	Go to higher clock mode
80%	80%	Go to lower clock mode
50%	50%	Keep the current clock mode

* See also: https://www.youtube.com/watch?v=T8fLjWyI5nI

CPU activity examples* (1/2)

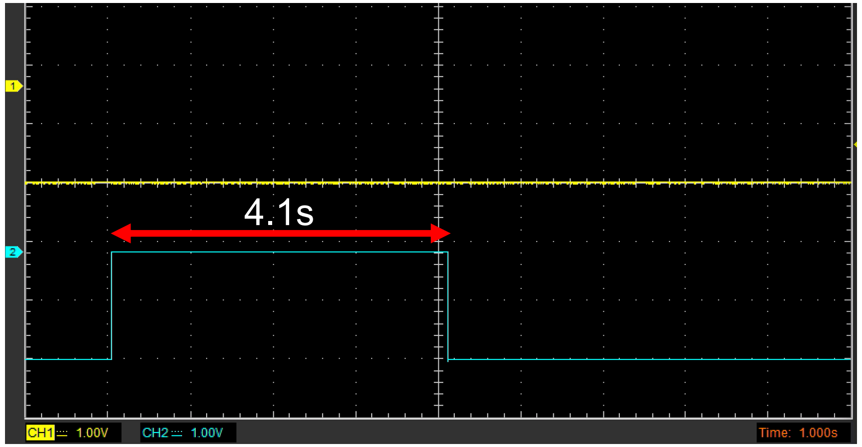
SONY

(1)



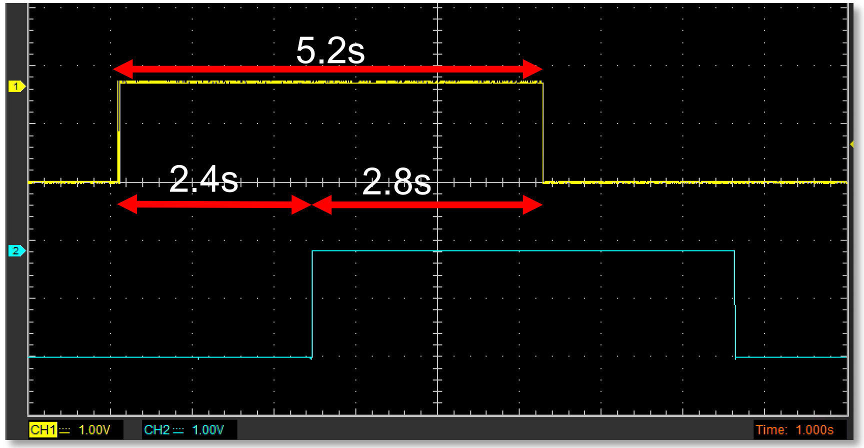

```
nsh> taskset 3 busyloop 4
```

(3)



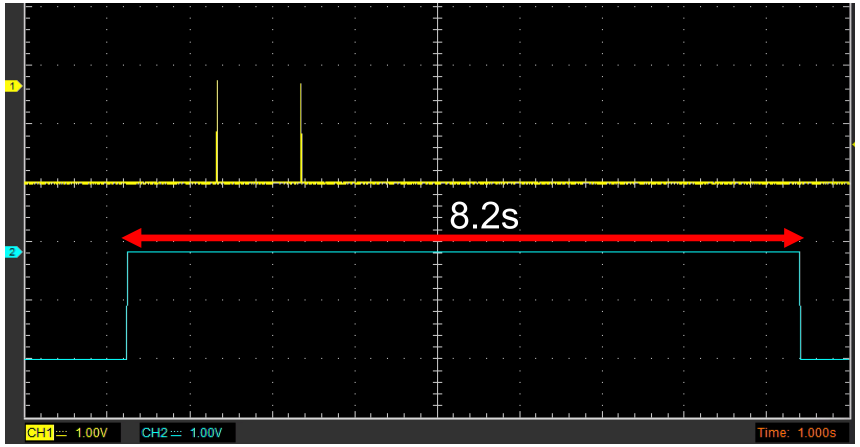

```
nsh> taskset 2 busyloop
```

(2)




```
nsh> taskset 3 busyloop 4 &
nsh> taskset 3 busyloop 4 &
```

(4)




```
nsh> taskset 2 busyloop 4 &
nsh> taskset 2 busyloop 4 &
```

* CH1=Cortex-M3 #0, CH2=Cortex-M3 #1

```
Usage: taskset mask command [args]
```

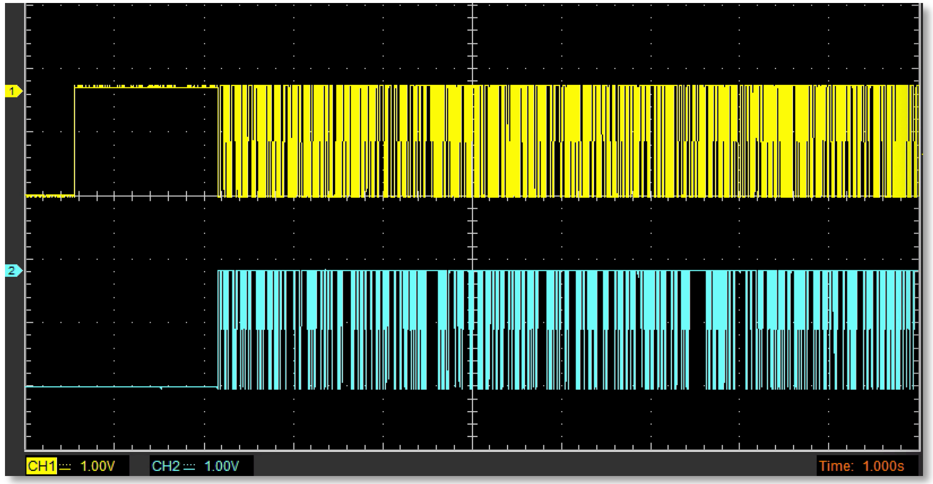
mask=1 assigns CPU0, mask=2 assigns CPU1, mask=3 assigns CPU0 or CPU1

NuttX 2019 International workshop

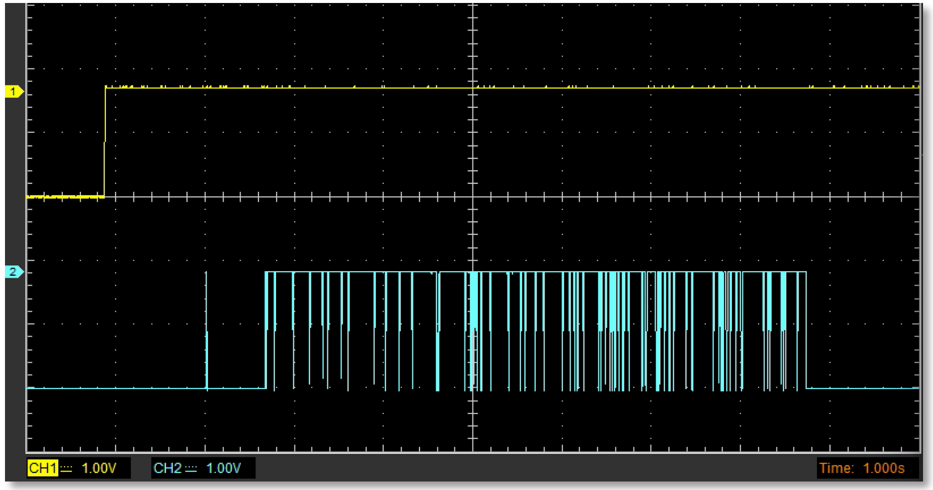
CPU activity examples (2/2)

SONY

- Background
  - LC823450 has 3 SDIO controllers.
  - eMMC uses CH0, uSD uses CH1.
  - Accessing different channels will be faster than accessing the same channel.
- (1) Two md5 for the same channel
  - Concurrent access is impossible.
  - Results: 11.0 sec & 11.0 sec (file size=6.6MB)
  - NOTE: 5.9 sec (eMMC single access)
- (2) Two md5 for different channels
  - Concurrent access is possible.
  - Results: 7.8 sec & 7.9 sec (file size=6.6MB)
  - NOTE: 6.2sec (uSD single access)



(1) Two md5 for the same channel (eMMC)



(2) Two md5 for different channels (eMMC and uSD)

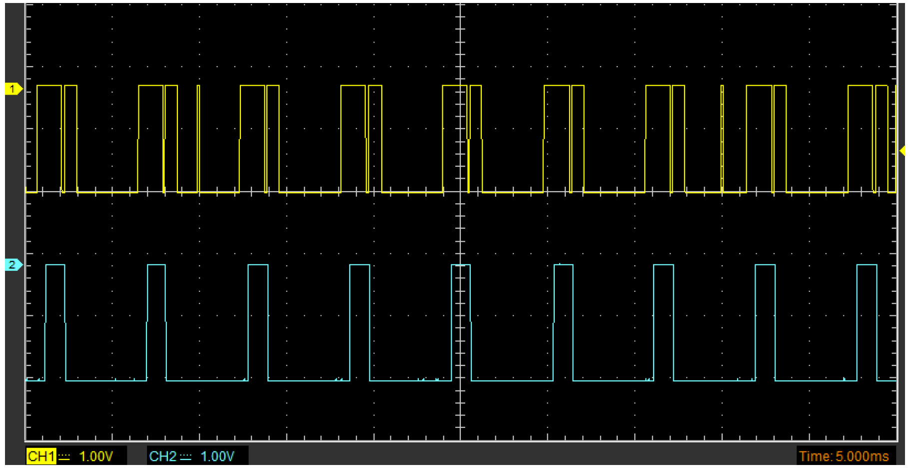
* uSD: SanDisk 16GB (SDSDQUP-016G-J35A)

Power consumption comparison

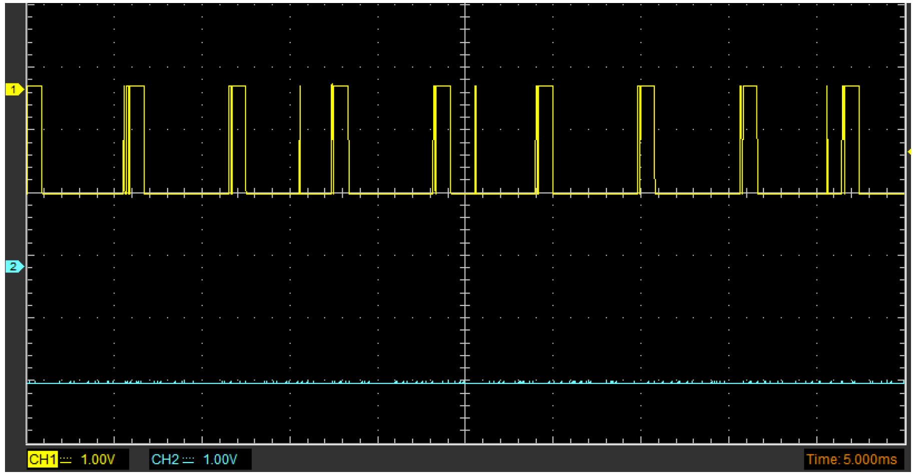
SONY

- nxplayer with local playback
  - WAV file 44.1kHz/16bit/2ch on eMMC
  - Vdd1=1.0V *
  - CPU clock = 40MHz (active), 6MHz(idle)
- Power consumption** @Vdd1
  - SMP : 6.0mA (idle=3.6mA)
  - non-SMP : 4.4mA (idle=3.5mA)

Performance penalty in SMP mode is outstanding (i.e. bus conflicts and scheduling overhead) . However, more optimization would be possible.



SMP



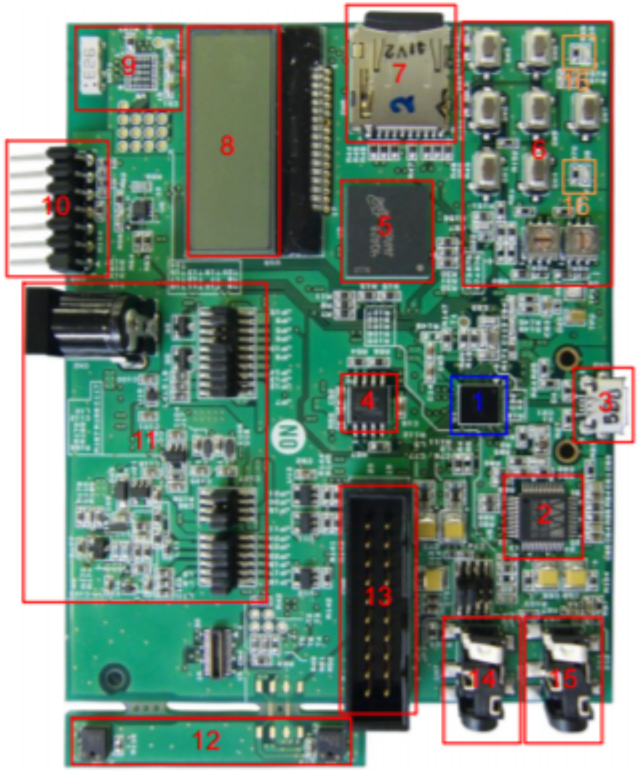
non-SMP

*Power consumption of the logic part (i.e. Cortex-M3, SRAM, DMA, I2S, …) inside the MCU
**Audio paths are need to be changed as of OpenIoT NA 2018

Networking with LC823450XGEVK

SONY

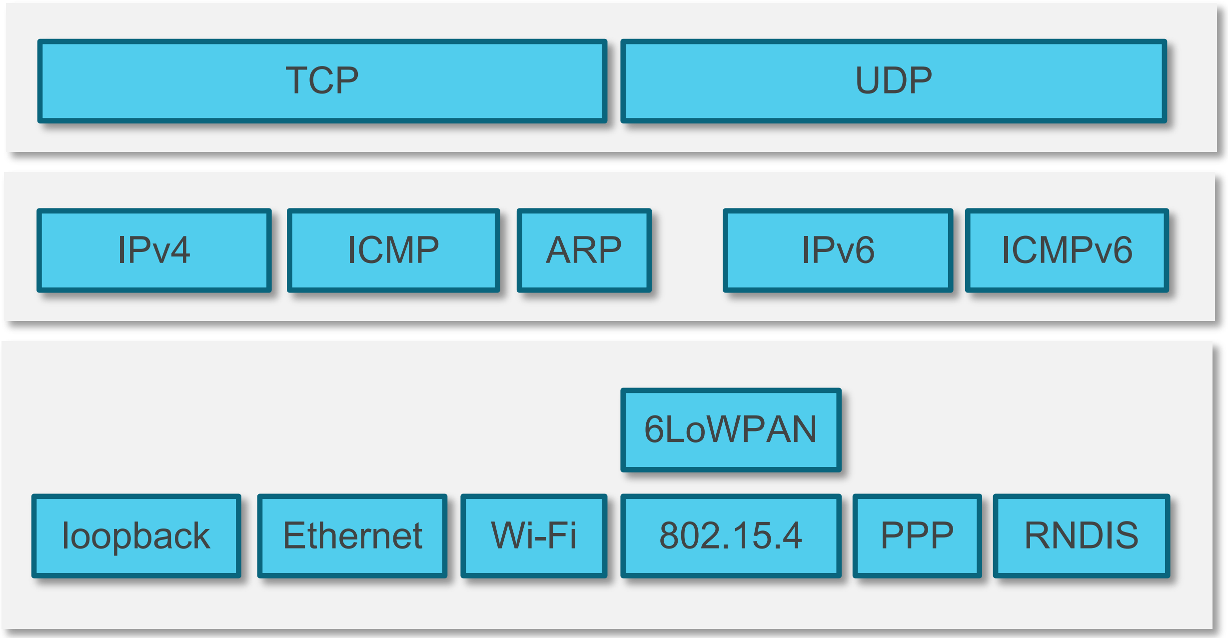
- Motivation
  - Confirm NuttX network stack feasibility
    - IPv4, IPv6, ICMP, UDP, TCP, …
  - Run the network stack with minimum efforts. (We already have an USB driver for LC823450)
  - Audio streaming (PCM and MP3)
  - Run the network stack in SMP mode
  - Do various tests via telnet



NuttX networking features

SONY

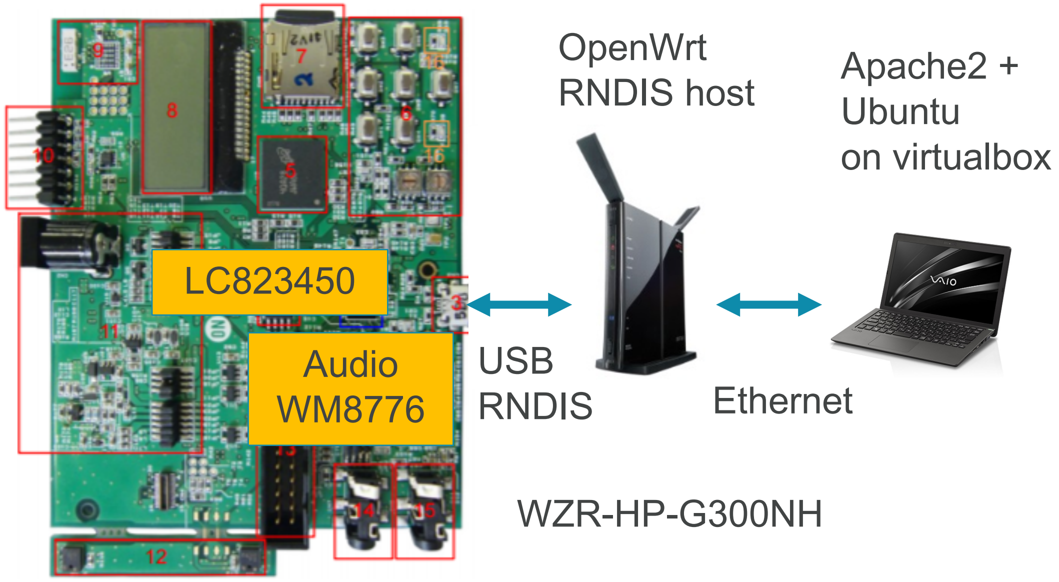
- Ethernet and IEEE 802.11 Full MAC
- 6LoWPAN for radio network drivers (IEEE 802.15.4 MAC)
- USB RNDIS (since 7.23), CDC-ECM (since 7.26)
- SLIP, TUN/PPP, local loopback devices
- IPv4, IPv6, TCP, UDP, ARP, ICMP, ICMPv6, IGMPv2
- IP forwarding
- BSD compatible socket layer
- DNS name resolution / NetDB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



PCM audio streaming via RNDIS

SONY

- Fix RNDIS driver for NuttX
  - Fix data corruption
  - Add USB high speed mode support
- Receive window control has been added
  - Need more improvement due to packet drop
- 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)



PCM audio streaming demo environment

PCM audio streaming example

SONY

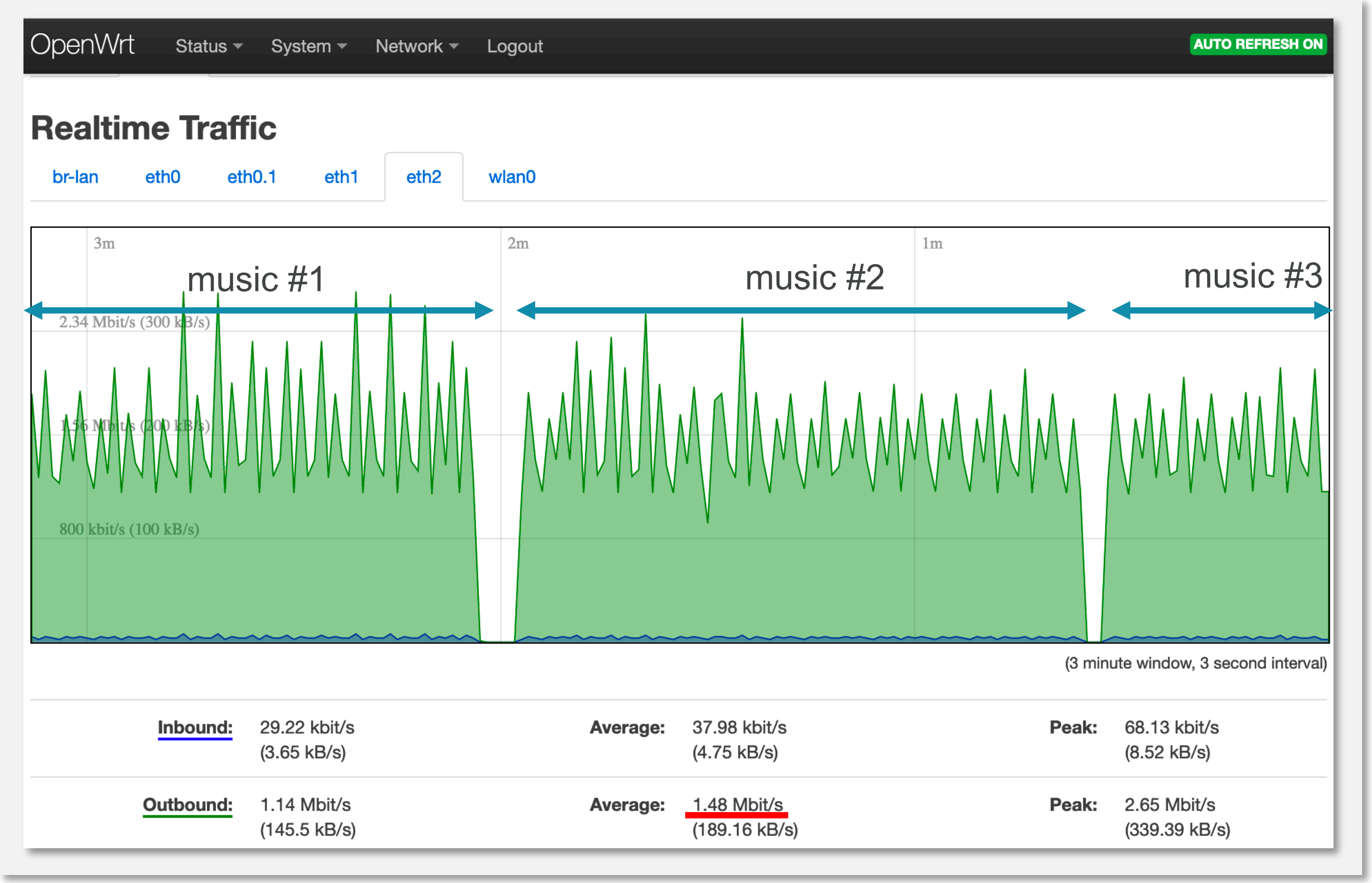
- 'ps' command results shows
  - Dual CPUs are running
  - telnet daemon is running
  - one telnet session is running
  - nxplayer is running

- 'ifconfig' command results shows
  - private address has been assigned via DHCP
  - TCP/UDP traffic (NOTE: some TCP packets are dropped due to iob starvation, so TCP flow control should be improved)

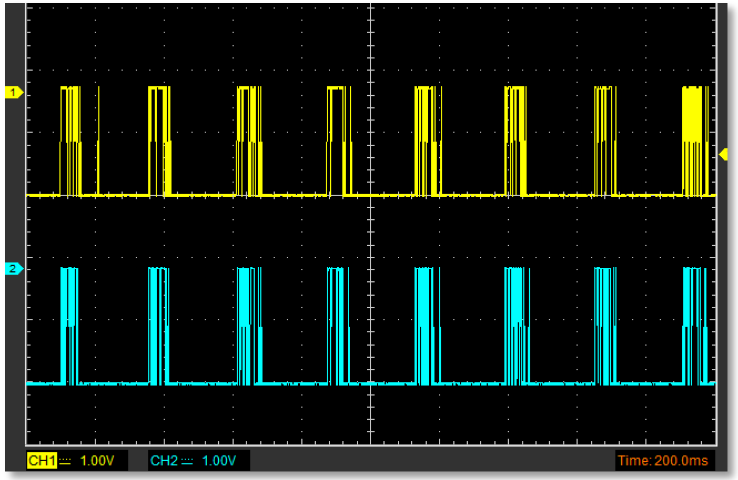
```
File  Edit  View  Search  Terminal  Help
nsh> ps
  PID GROUP CPU PRI POLICY   TYPE    NPX STATE    EVENT     SIGMASK   STACK COMMAND
    0     0   0   0 FIFO     Kthread N-- Running            00000000 000000 CPU0 IDLE
    1     0   1   0 FIFO     Kthread N-- Assigned           00000000 002044 CPU1 IDLE
    3     1 --- 192 FIFO     Kthread --- Waiting  Signal    00000000 002028 hpwork
    4     1 ---  60 FIFO     Kthread --- Waiting  Signal    00000000 002028 lpwork
    5     1 --- 100 FIFO     Task    --- Waiting  Signal    00000000 003052 init
    7     5 --- 100 FIFO     Task    --- Waiting  Semaphore 00000010 002020 Telnet daem
  114     6   1 100 FIFO     Task    --- Running            00000010 003044 Telnet sess
  115     5 --- 100 FIFO     Task    --- Waiting  Semaphore 00000000 003044 nxplayer
  116     5 --- 246 FIFO     pthread --- Waiting  Semaphore 00000000 001500 playthread
  117     5 --- 252 FIFO     pthread --- Waiting  MQ empty  00000000 000764 wm8776 0x0x
nsh> ifconfig
lo      Link encap:Local Loopback at UP
        inet addr:127.0.0.1 DRaddr:127.0.0.1 Mask:255.0.0.0

eth0    Link encap:Ethernet HWaddr 00:e0:de:ad:be:ff at UP
        inet addr:192.168.1.245 DRaddr:192.168.1.1 Mask:255.255.255.0

          IPv4  TCP   UDP   ICMP
Received  401d  2f9a  0210  0014
Dropped   0e5f  1fff  0000  0000
  IPv4     VHL: 003a   Frg: 0259
  Checksum 0000  0000  0000  ----
  TCP      ACK: 0000   SYN: 0000
           RST: 001d  001d
  Type    0000  ----  ----  0000
Sent      100d  0ff5  0004  0014
  Rexmit  ----  002d  ----  ----
nsh>
```


NuttX 2019 International workshop

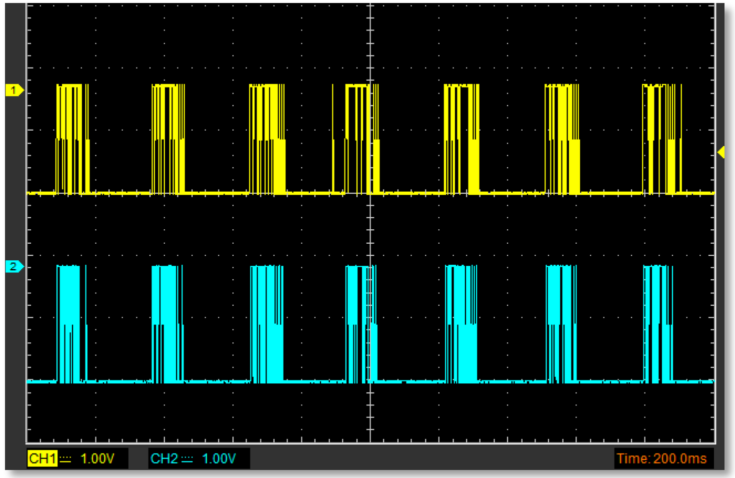
Network traffic and CPU activity examples



Network traffic when PCM audio (44.1k/16bit/2ch) streaming is working



(1) CPU clock : 160MHz (fixed)



(2) CPU clock : 160/80/40MHz (auto)

MP3 audio streaming via Bluetooth

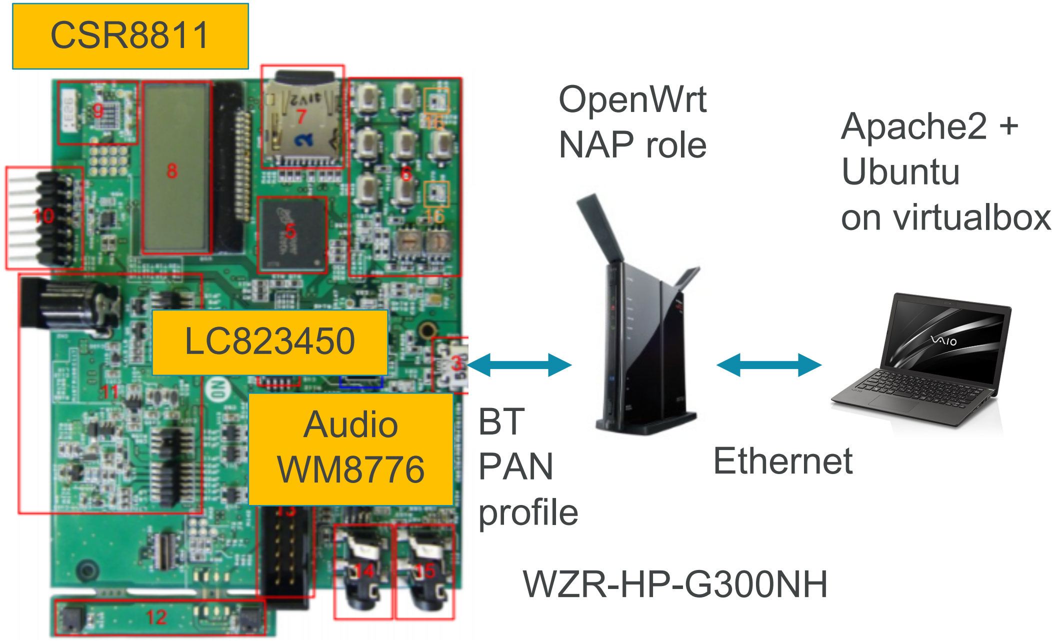
SONY

- 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

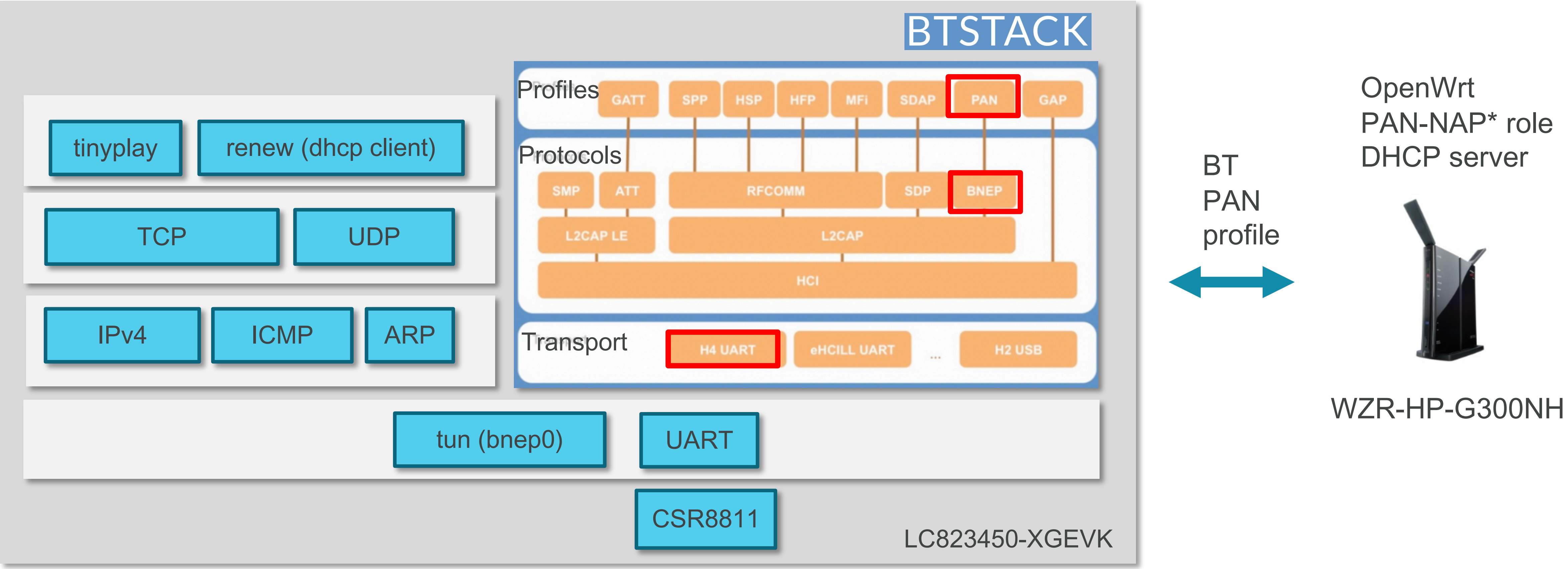


MP3 audio streaming demo environment

* https://bluekitchen-gmbh.com/

** We can use posix-h5 (3-wire protocol) as well. However, it has performance drawbacks.

Running the BTstack on NuttX



*PAN: Personal Area Network
*BNEP: Bluetooth Network Encapsulation Protocol

*NAP: Network Access Point

BTstack log example

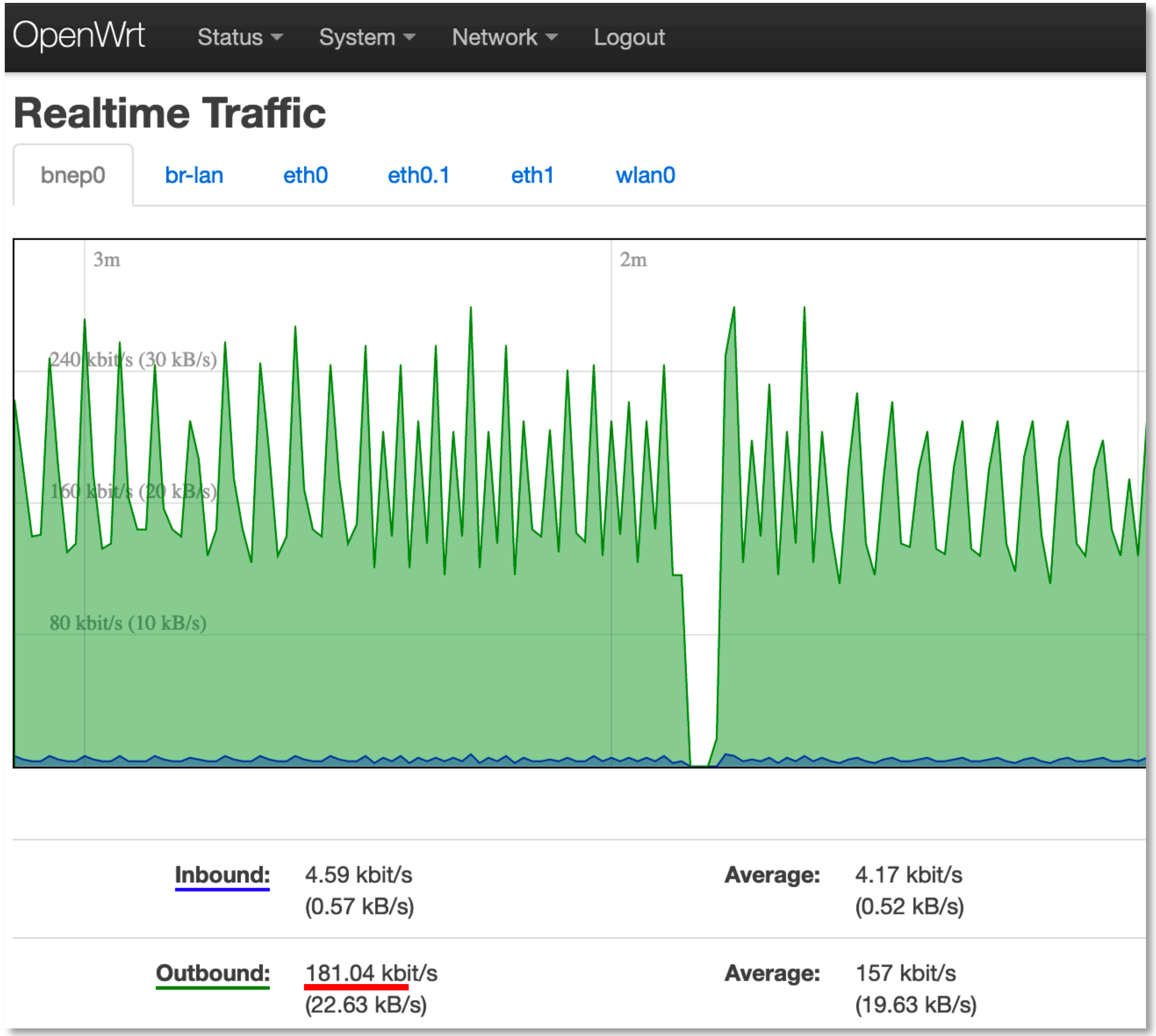
SONY

```
H4 device: /dev/ttyS1
[2019-06-27 12:12:41.950] LOG -- bnep.c.1582: BNEP_REGISTER_SERVICE mtu 1691
[2019-06-27 12:12:41.950] LOG -- l2cap.c.3387: L2CAP_REGISTER_SERVICE psm 0xf mtu 65535
[2019-06-27 12:12:41.950] LOG -- hci.c.2750: hci_power_control: 1, current mode 0
[2019-06-27 12:12:42.170] LOG -- btstack_uart_block_posix.c.189: h4_set_baudrate 115200
[2019-06-27 12:12:42.280] LOG -- hci.c.3797: BTSTACK_EVENT_STATE 1
[2019-06-27 12:12:42.490] LOG -- hci.c.1077: Resend HCI Reset
[2019-06-27 12:12:42.700] LOG -- hci.c.1077: Resend HCI Reset
[2019-06-27 12:12:42.810] LOG -- hci.c.1878: Manufacturer: 0x000a
Local version information:
- HCI Version    0x0006
- HCI Revision   0x2031
```

```
[2019-06-27 12:12:56.990] LOG -- bnep.c.1235: L2CAP_EVENT_CHANNEL_OPENED for BLUETOOTH_PRO
[2019-06-27 12:12:57.000] LOG -- bnep.c.1259: L2CAP_EVENT_CHANNEL_OPENED: outgoing connect
[2019-06-27 12:12:57.010] LOG -- bnep.c.694: bnep_max_frame_size_for_l2cap_mtu:  1691 -> 1
[2019-06-27 12:12:57.070] LOG -- bnep.c.1110: BNEP_CONTROL: Type: 2, size: 3, is_extension
[2019-06-27 12:12:57.070] LOG -- bnep.c.879: BNEP_CONNECTION_RESPONSE: Channel established
[2019-06-27 12:12:57.070] LOG -- bnep.c.79: BNEP_EVENT_CHANNEL_OPENED status 0x00 bd_addr:
BNEP connection open succeeded to 00:1B:DC:06:86:59 source UUID 0x1115 dest UUID: 0x1116,
[2019-06-27 12:12:57.070] LOG -- btstack_network.c.264: BNEP device "bnep0" allocated
Network Interface bnep0 activated
```

MP3 streaming via Bluetooth

SONY




```
56 bytes from 192.168.1.220: icmp_seq=3 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=4 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=5 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=6 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=7 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=8 time=20 ms
56 bytes from 192.168.1.220: icmp_seq=9 time=10 ms
10 packets transmitted, 10 received, 0% packet loss, time 1010
nsh> ifconfig
lo      Link encap:Local Loopback at UP
        inet addr:127.0.0.1 DRaddr:127.0.0.1 Mask:255.0.0.0

bnep0   Link encap:Ethernet HWaddr 00:02:5b:00:a5:a5 at UP
        inet addr:192.168.1.156 DRaddr:192.168.1.1 Mask:255.255

             IPv4   TCP   UDP   ICMP
Received     2131   202b  003e  0014
Dropped      00b3   061b  0000  0000
  IPv4        VHL:  0002   Frg: 0000
  Checksum   0000   0000  0000  ----
  TCP         ACK:  0000   SYN: 0000
              RST:  0002  0002
  Type       0000   ----  ----  0000
Sent         1a2a   1a14  0002  0014
  Rexmit     ----   0005  ----  ----
nsh> tinyplay http://192.168.1.220/~ishikawa/audio/sample2.mp3
tinyplay [14:140]
nsh> fmt=mp3 ch=2 freq=44100
```

Demo videos

- CPU activity examples (busyloop, md5)
- HTTP PCM audio streaming via RNDIS

Any Questions?