SMP and Networking support on NuttX / LC823450

Masayuki Ishikawa
Sony Corporation
3 PUBLICATIONS 0 CITATIONS

Embedded Linux Conference North America

OpenIoTSummit North America

SMP and Networking support on NuttX / LC823450

Masayuki.Ishikawa@sony.com

Koichi.Okamoto@sony.com

Sony Video & Sound Products Inc.

THE LINUX FOUNDATION

About us

SONY

Masayuki Ishikawa

Sony Video & Sound Products Inc.

Senior Software Engineer

Koichi Okamoto

Sony Video & Sound Products Inc.

Senior System Engineer

Technical background

- 3D graphics application development
- Home networking software development
- Internet-to-home service development
- Linux-based audio products development
- Android-based audio products development
- NuttX-based audio products development

Technical background

- DAB Base Band LSI development
- Non OS Car audio development
- Linux-based 1SEG mobile DTV development
- Android-based AVC/AVN development
- uITRON-based Car audio development
- Linux-based network platform development

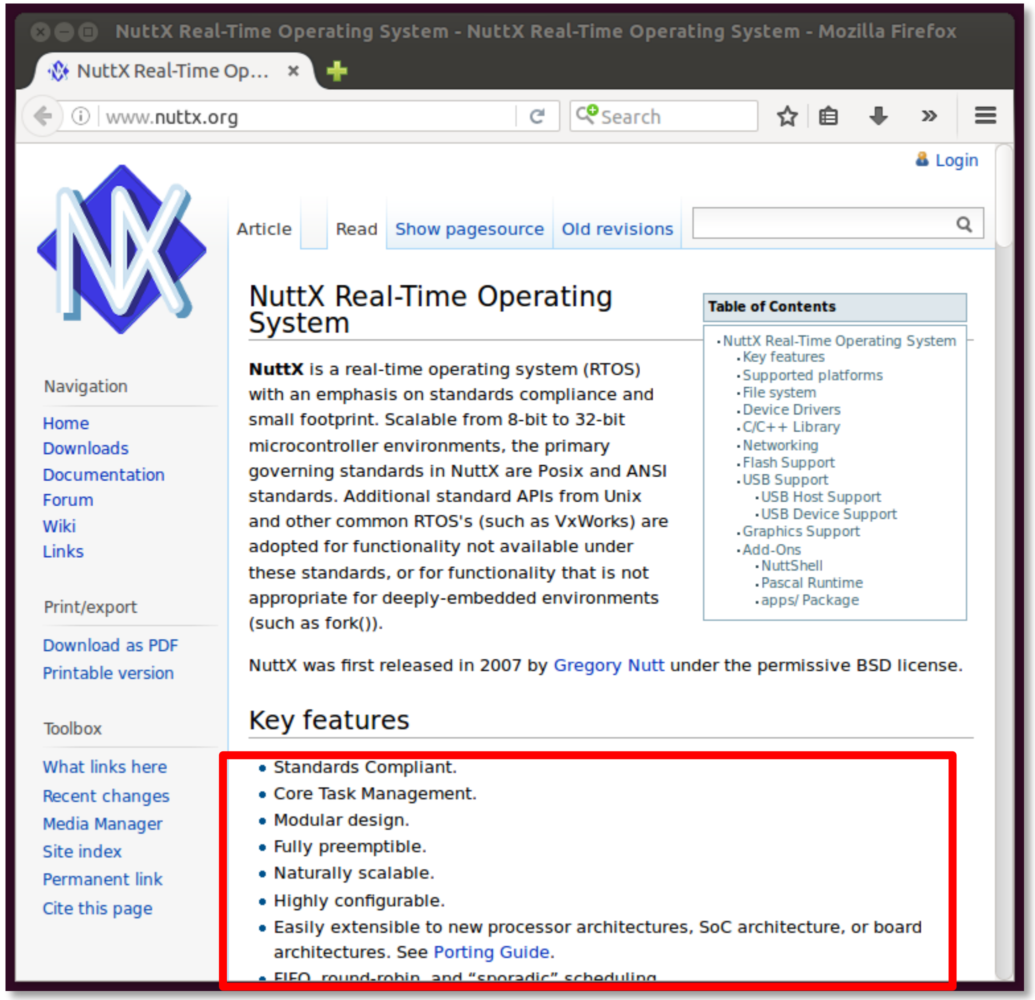
THE LINUX FOUNDATION

Agenda

- About NuttX and why we chose it
- Development history (NuttX-based products)
- New topics
  - The road to NuttX upstream
  - SMP (Symmetric Multiprocessing) related status
  - OpenOCD NuttX status
  - Networking related status
- Demo videos
- Future challenges

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


NuttX Real-Time Operating System - NuttX Real-Time Operating System - Mozilla Firefox
NuttX Real-Time Op...
www.nuttx.org
Search
Login
Article
Read
Show pagesource
Old revisions

NuttX Real-Time Operating System

NuttX is a real-time operating system (RTOS) with an emphasis on standards compliance and small footprint. Scalable from 8-bit to 32-bit microcontroller environments, the primary governing standards in NuttX are Posix and ANSI standards. Additional standard APIs from Unix and other common RTOS's (such as VxWorks) are adopted for functionality not available under these standards, or for functionality that is not appropriate for deeply-embedded environments (such as fork()).

NuttX was first released in 2007 by Gregory Nutt under the permissive BSD license.

Key features

- Standards Compliant.
- Core Task Management.
- Modular design.
- Fully preemptible.
- Naturally scalable.
- Highly configurable.
- Easily extensible to new processor architectures, SoC architecture, or board architectures. See Porting Guide.
- FIFO, round-robin, and “sporadic” scheduling

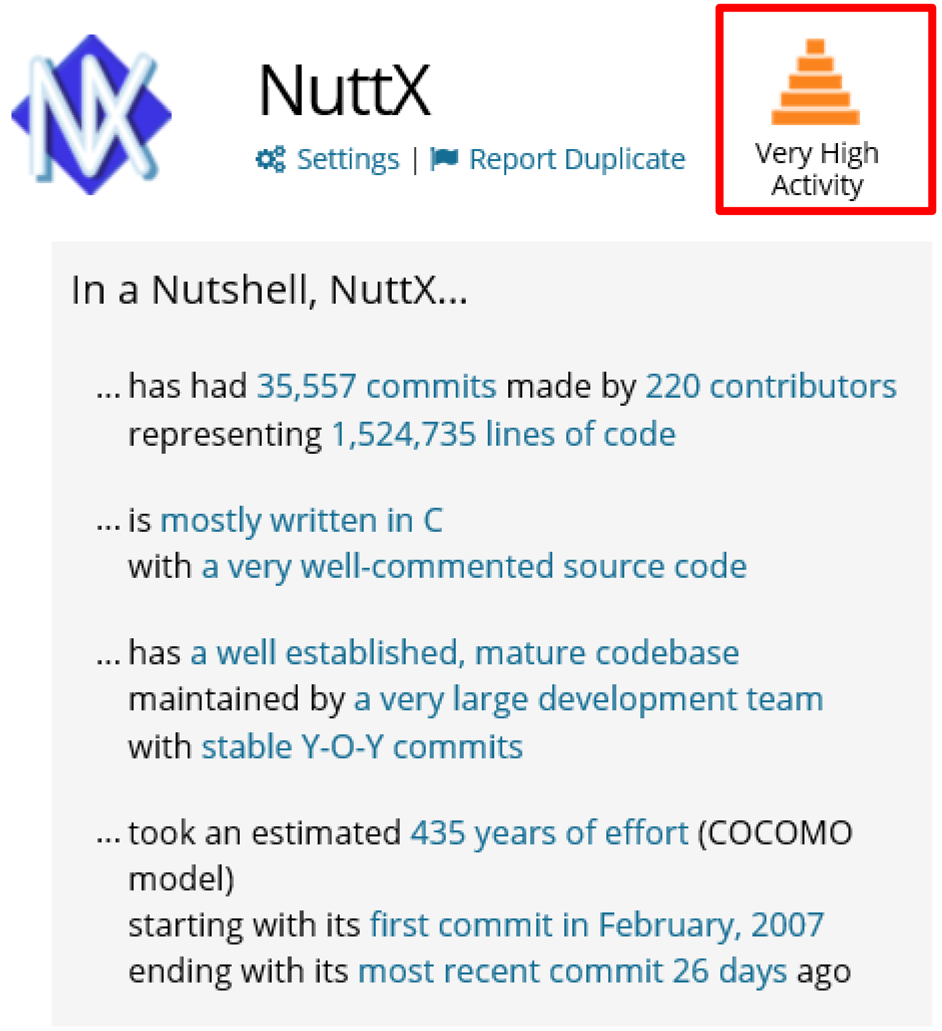
Table of Contents

From http://www.nuttx.org/

* ELF = Executable and Linking Format

Project report from OpenHub *

SONY



NuttX

Settings | Report Duplicate

Very High Activity

In a Nutshell, NuttX...

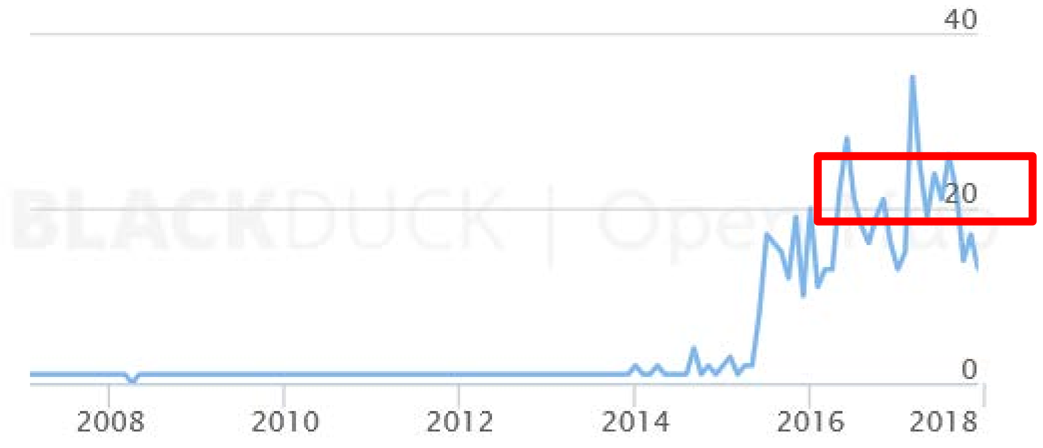
... has had 35,557 commits made by 220 contributors representing 1,524,735 lines of code

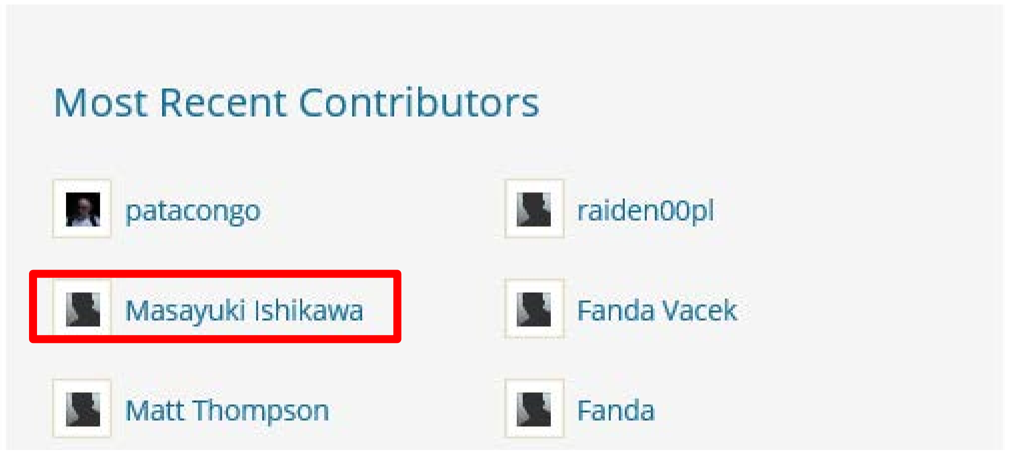
... is mostly written in C with a very well-commented source code

... has a well established, mature codebase maintained by a very large development team with stable Y-O-Y commits

... took an estimated 435 years of effort (COCOMO model) starting with its first commit in February, 2007 ending with its most recent commit 26 days ago

Contributors per Month





Most Recent Contributors

patacongo	raiden00pl
Masayuki Ishikawa	Fanda Vacek
Matt Thompson	Fanda

*https://www.openhub.net/p/nuttx (as of 23/Feb/2018)

THE LINUX FOUNDATION

Development history*(NuttX-based products)

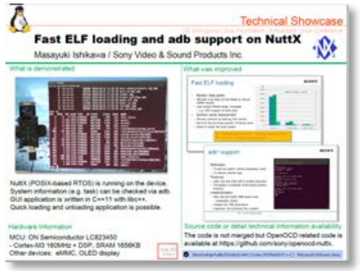
SONY

- 10/2013 -
  - Ported NuttX to LC823425 (ARM7)
- 04/2014 –
  - Ported bluetooth stack to NuttX + QEMU
- 07/2014 -
  - Ported NuttX to LC823450 (Cortex-M3) FPGA
- 01/2015 -
  - Migrated to LC823450-ES board
- 09/2015 -
  - Released NuttX-based audio products.
- 02/2017 -
  - Talked at ELC2017 North America **

SONY

Developing Audio Products with
Cortex-M3/NuttX/C++11

Masayuki.Ishikawa@sony.com
Senior Software Engineer
Sony Video & Sound Products Inc.



Technical Showcase

Fast ELF loading and adb support on NuttX

Masayuki Ishikawa / Sony Video & Sound Products Inc.

What is demonstrated

NuttX (POSIX-based RTOS) is running on the device.
System information (e.g. task) can be checked via adb.
GUI application is written in C++11 with libc++.
Quick loading and unloading application is possible.

Hardware information

MCU: ON Semiconductor LC823450
- Cortex-M3 160MHz + DSP, SRAM 1656KB
Other devices: eMMC, OLED display

What was improved

Fast ELF loading

adb* support

Source code or detail technical information availability

The code is not merged but OpenOCD related code is available at https://github.com/sony/openocd-nuttx.

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

THE LINUX FOUNDATION

FY16-17 products*

NW-WS620

- Music player with bluetooth (A2DP, HFP/HSP)
- Ambient sound mode
- Up to 12h of battery life

ICD-TX800

- Small (38mm x 38mm) and light (22g) voice recorder
- REC Remote App support with bluetooth

SMR-10

- Personal sound amplifier
- Bluetooth (A2DP with Low latency SBC: 50ms)
- SPI Flash Boot

*ICD-PX470 is also available but not shown here

LC823450 Features

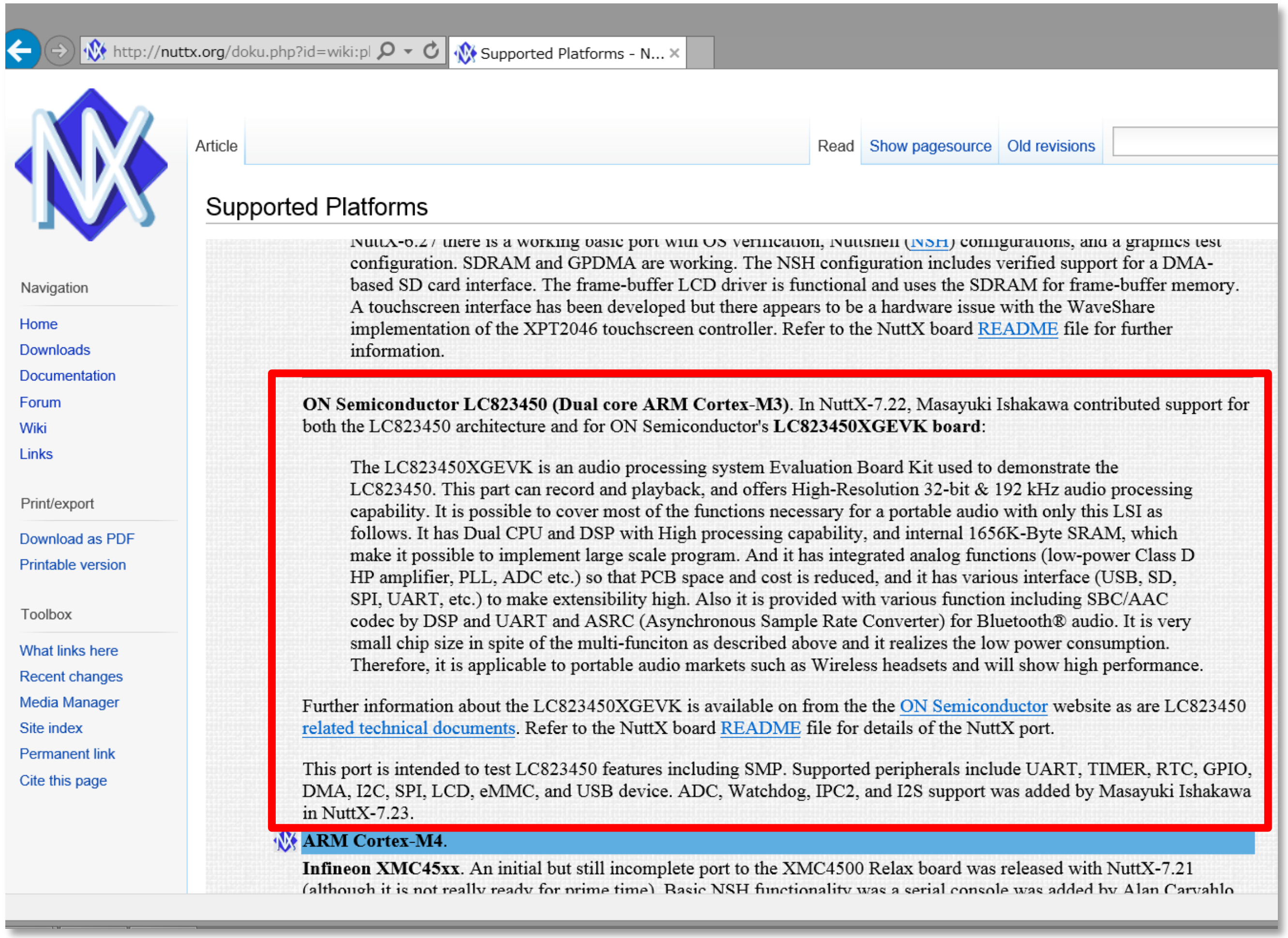
- ARM Cortex-M3 Dual Core
- 32bit fixed point, dual-MAC original DSP
- Internal SRAM (1656KB) for ARM and DSP
- I2S I/F with 16/24/32bit, MAX 192kHz (2chx2)
- Hard wired audio functions
  - MP3 encoder and decoder, EQ (6-band equalizer), etc.
- Integrated analog functions
  - Low-power Class D HP amplifier, 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

The road to NuttX upstream *

- Start discussion with ON Semiconductor
  - To disclose their technical documents
  - Because we developed the code based on their documents.
- Purchase LC823450XGEVK evaluation kit
  - Using an evaluation board is much better than a Sony's proprietary board.
- Port existing code to the latest upstream
  - Must comply with NuttX C Coding Standard
- Prepare an account on bitbucket
  - Sending a PR (Pull Request) is more useful than sending patches by e-mail.
- Finally send a Pull Request



Supported Platforms

NuttX-6.27 there is a working basic port with OS verification, Nuttshell (NSH) configurations, and a graphics test configuration. SDRAM and GPDMA are working. The NSH configuration includes verified support for a DMA-based SD card interface. The frame-buffer LCD driver is functional and uses the SDRAM for frame-buffer memory. A touchscreen interface has been developed but there appears to be a hardware issue with the WaveShare implementation of the XPT2046 touchscreen controller. Refer to the NuttX board README file for further information.

ON Semiconductor LC823450 (Dual core ARM Cortex-M3). In NuttX-7.22, Masayuki Ishakawa contributed support for both the LC823450 architecture and for ON Semiconductor's **LC823450XGEVK board**:

The LC823450XGEVK is an audio processing system Evaluation Board Kit used to demonstrate the LC823450. This part can record and playback, and offers High-Resolution 32-bit & 192 kHz audio processing capability. It is possible to cover most of the functions necessary for a portable audio with only this LSI as follows. It has Dual CPU and DSP with High processing capability, and internal 1656K-Byte SRAM, which make it possible to implement large scale program. And it has integrated analog functions (low-power Class D HP amplifier, PLL, ADC etc.) so that PCB space and cost is reduced, and it has various interface (USB, SD, SPI, UART, etc.) to make extensibility high. Also it is provided with various function including SBC/AAC codec by DSP and UART and ASRC (Asynchronous Sample Rate Converter) for Bluetooth® audio. It is very small chip size in spite of the multi-funciton as described above and it realizes the low power consumption. Therefore, it is applicable to portable audio markets such as Wireless headsets and will show high performance.

Further information about the LC823450XGEVK is available on from the the ON Semiconductor website as are LC823450 related technical documents. Refer to the NuttX board README file for details of the NuttX port.

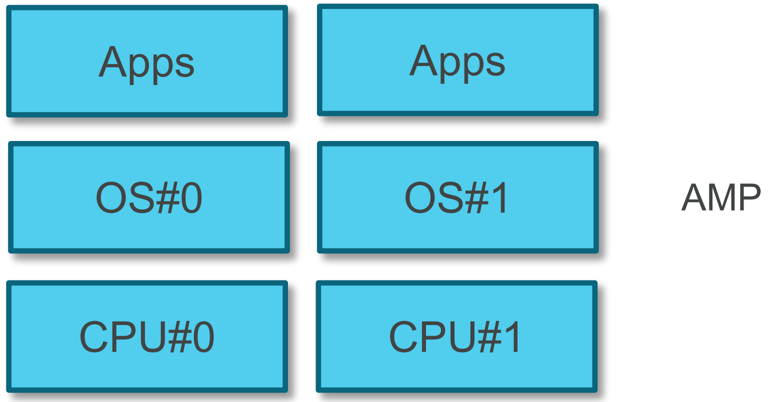
This port is intended to test LC823450 features including SMP. Supported peripherals include UART, TIMER, RTC, GPIO, DMA, I2C, SPI, LCD, eMMC, and USB device. ADC, Watchdog, IPC2, and I2S support was added by Masayuki Ishakawa in NuttX-7.23.

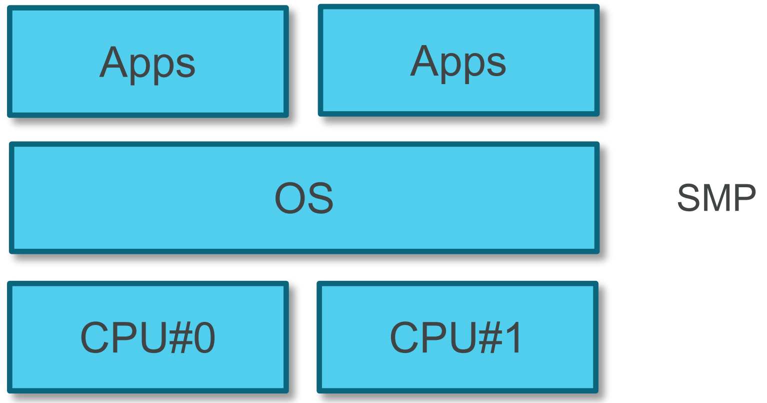
ARM Cortex-M4.

Infineon XMC45xx. An initial but still incomplete port to the XMC4500 Relax board was released with NuttX-7.21 (although it is not really ready for prime time). Basic NSH functionality was a serial console was added by Alan Carvahlo

AMP vs SMP *

- 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

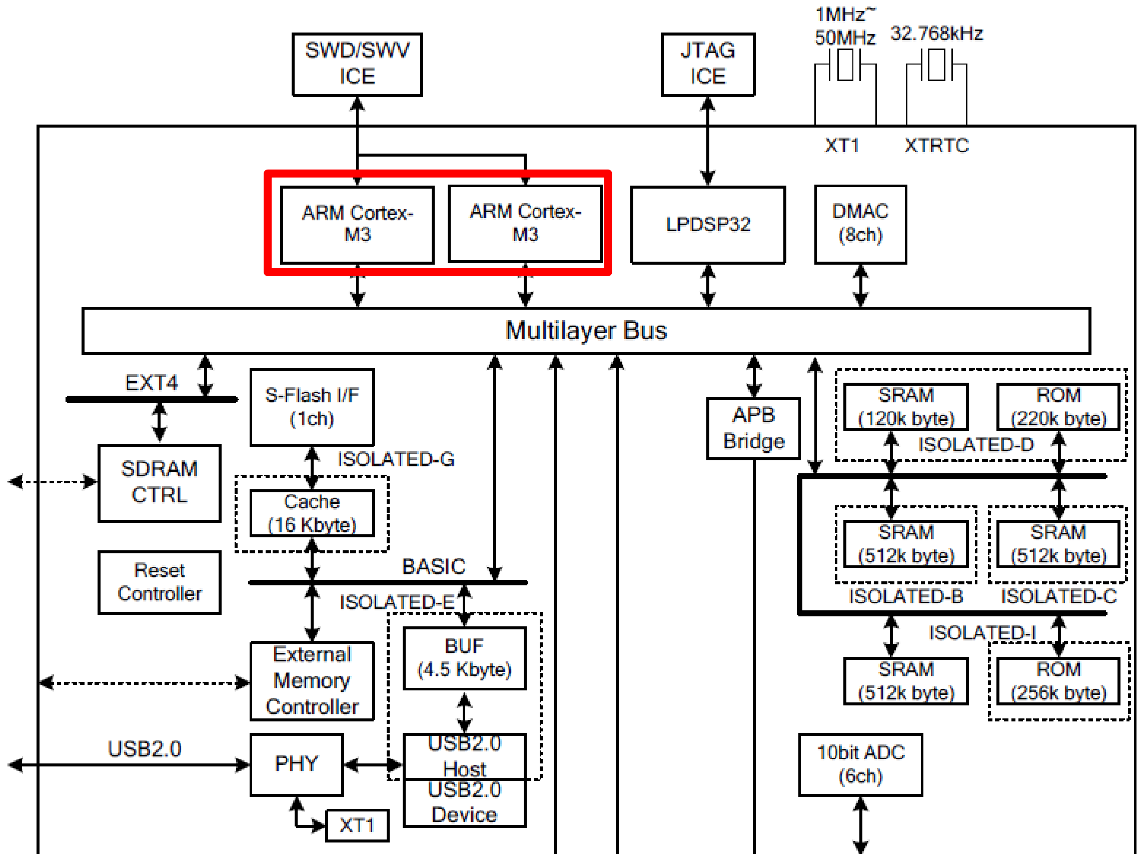




* http://www.embeddedintel.com/special_features.php?article=189

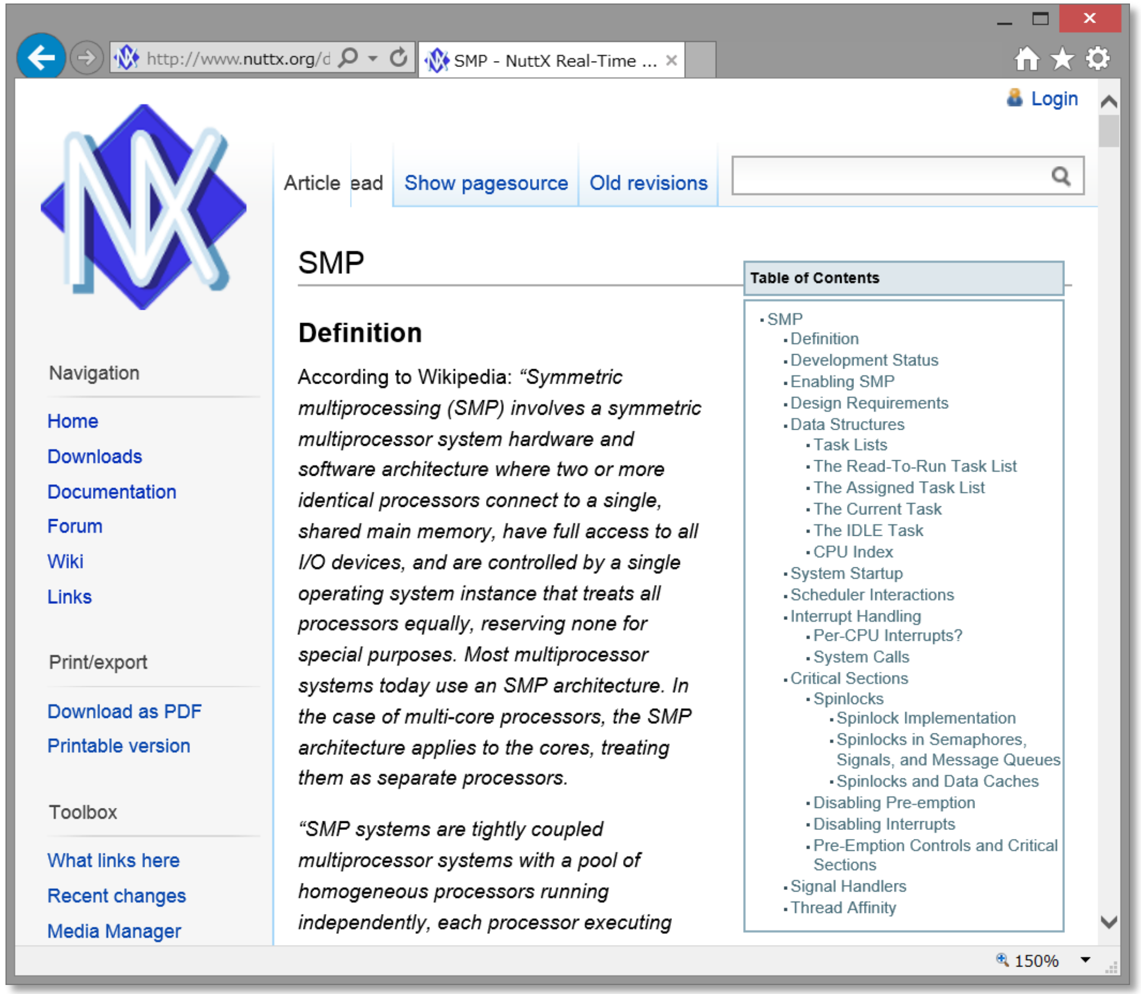
Why SMP with LC823450?

- Motivation
  - Achieve low power + high performance
  - Run existing applications in SMP mode
  - Confirm performance penalty
  - Establish knowledge on debugging
  - 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.



Introduction to the NuttX SMP kernel

- Minimum changes to non-SMP kernel
  - CONFIG_SMP is introduced.
  - Main changes are done in the scheduler
- Newly introduced
  - g_assignedtasks[cpu] to hold assigned tasks including currently running tasks for each CPU
  - Spinlock to protect shared resources
  - Critical section APIs to replace with local interrupt control APIs.
- CPU affinity
  - 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



SMP

Definition

According to Wikipedia: *"Symmetric multiprocessing (SMP) involves a symmetric multiprocessor system hardware and software architecture where two or more identical processors connect to a single, shared main memory, have full access to all I/O devices, and are controlled by a single operating system instance that treats all processors equally, reserving none for special purposes. Most multiprocessor systems today use an SMP architecture. In the case of multi-core processors, the SMP architecture applies to the cores, treating them as separate processors.*

"SMP systems are tightly coupled multiprocessor systems with a pool of homogeneous processors running independently, each processor executing

Table of Contents

NuttX SMP : available boards

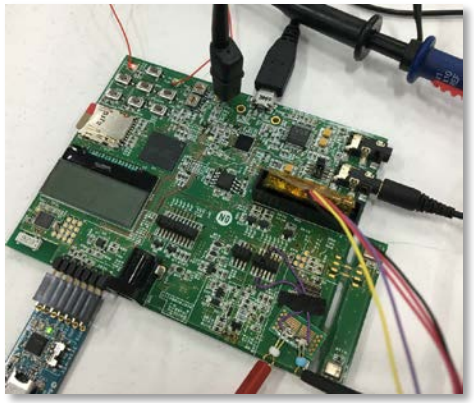
- NXP (Freescale) i.MX6 Quad Sabre
  - Quad Arm Cortex-A9
  - SMP kernel can run on QEMU *
- Espressif Systems ESP32
  - Dual Tensilica LX6

- 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.

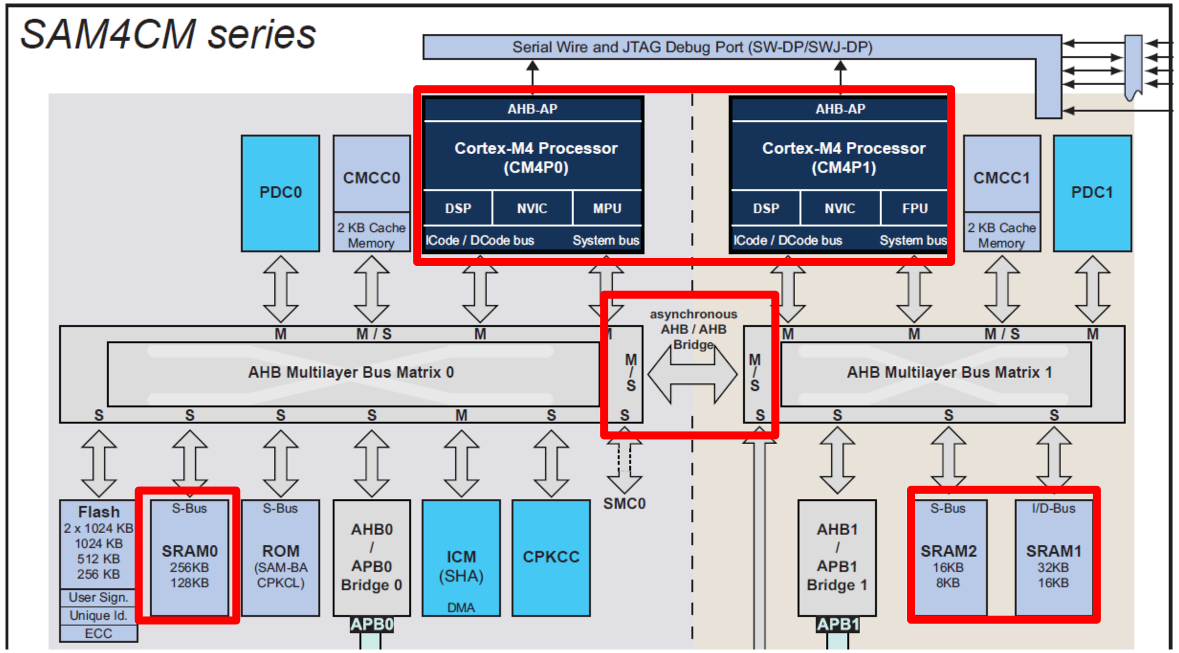
**http://www.components-center.com/product/ON-Semiconductor/LC823450XGEVK.html

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.



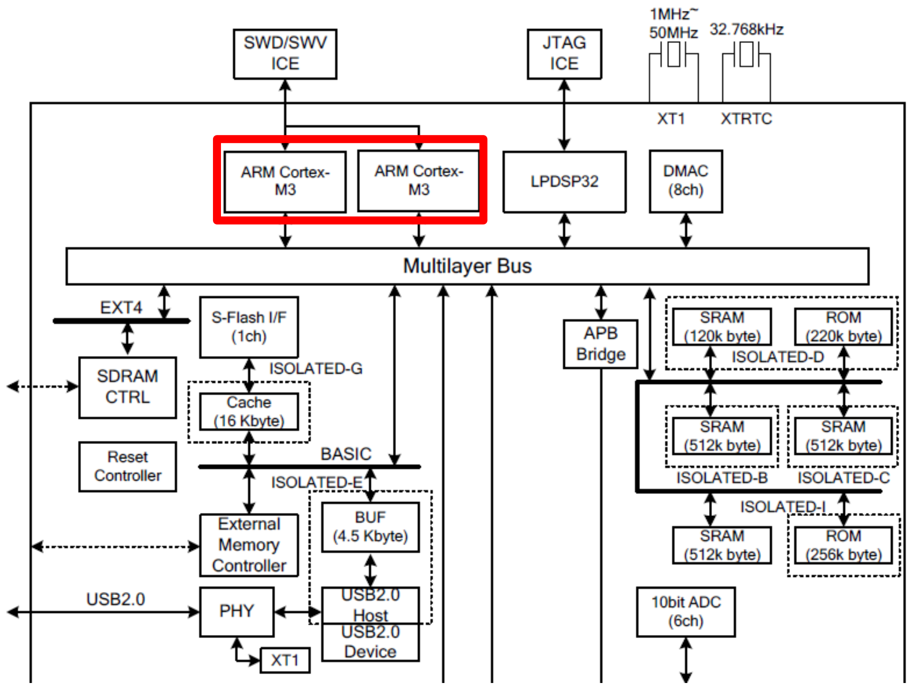
* we don’t think this board can perfectly work in SMP mode

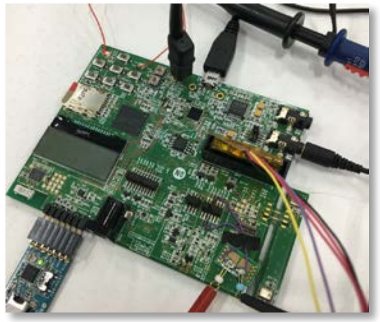
THE LINUX FOUNDATION

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 (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



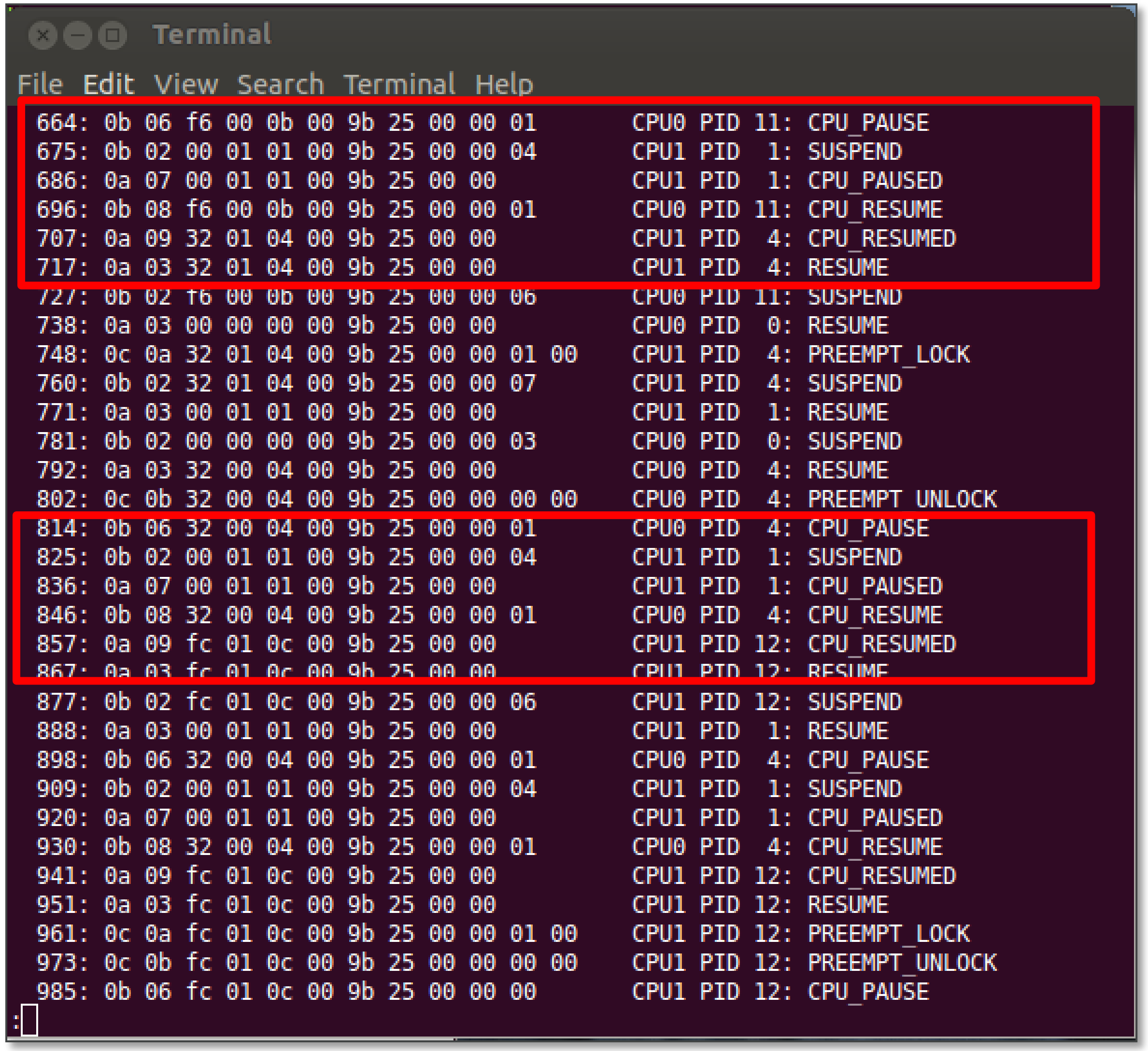


*Code is already merged into the upstream
*I2S and audio codec drivers were developed from scratch.

THE LINUX FOUNDATION

Tracing SMP kernel

- What 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

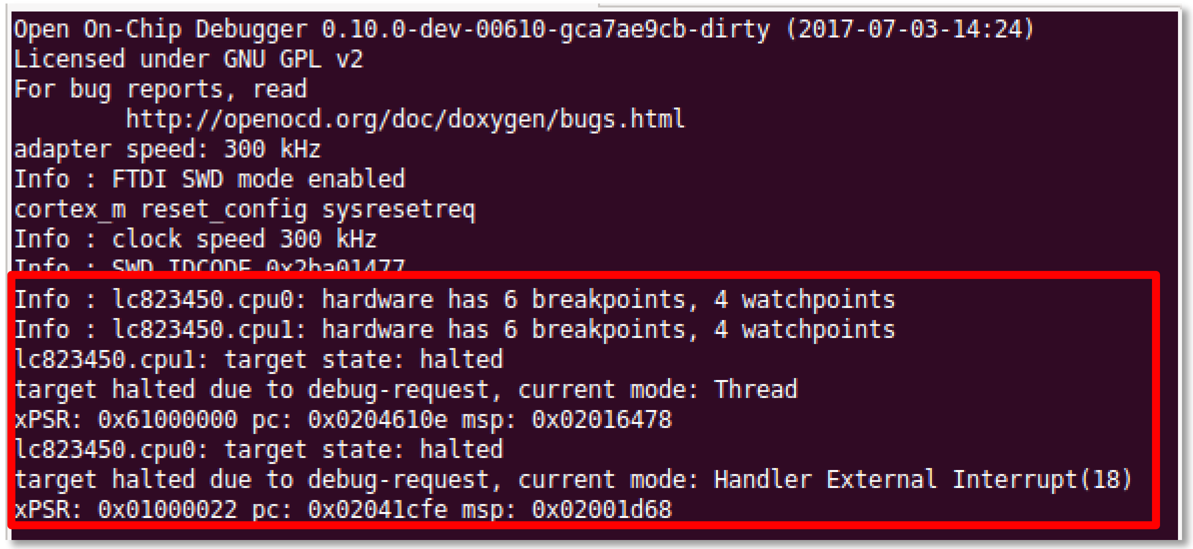


```
664: 0b 06 f6 00 0b 00 9b 25 00 00 01       CPU0 PID 11: CPU_PAUSE
675: 0b 02 00 01 01 00 9b 25 00 00 04       CPU1 PID  1: SUSPEND
686: 0a 07 00 01 01 00 9b 25 00 00          CPU1 PID  1: CPU_PAUSED
696: 0b 08 f6 00 0b 00 9b 25 00 00 01       CPU0 PID 11: CPU_RESUME
707: 0a 09 32 01 04 00 9b 25 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
:
```

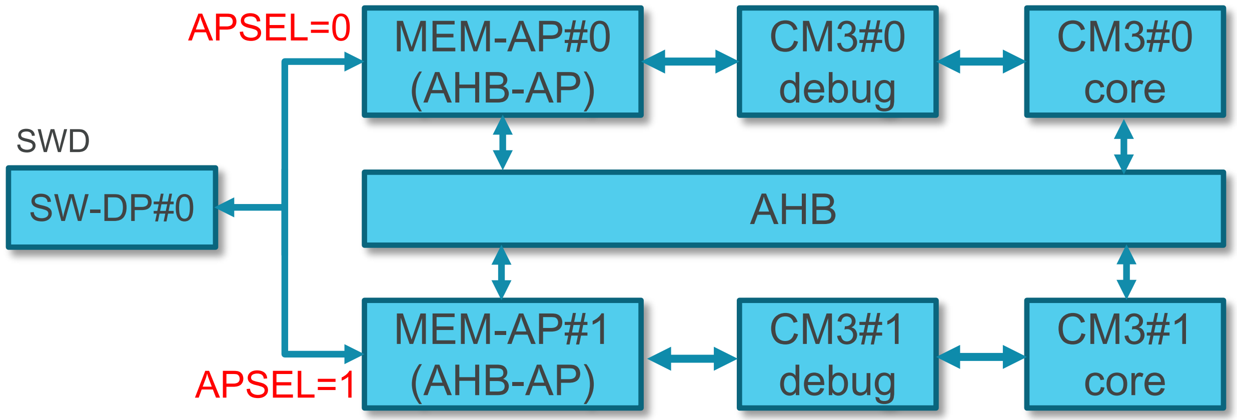
OpenOCD for lc823450-smp*

- 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
```

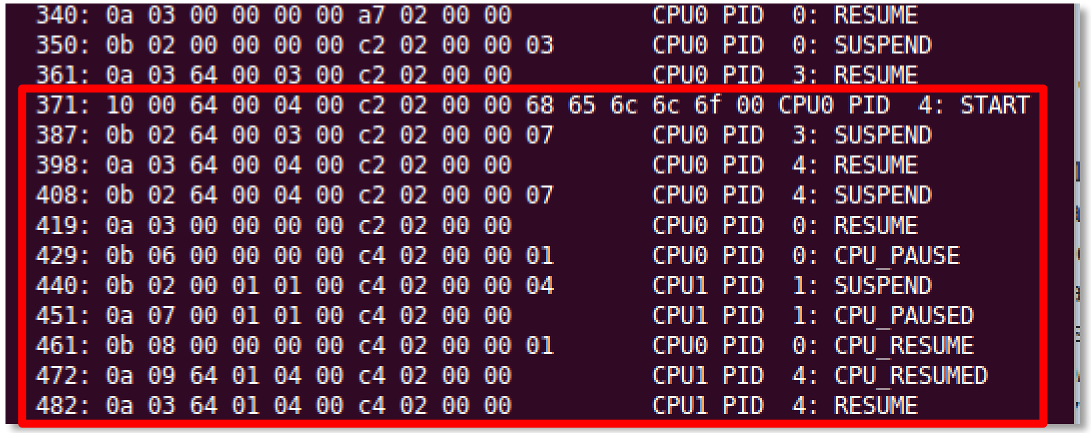


*Code is NOT merged yet.

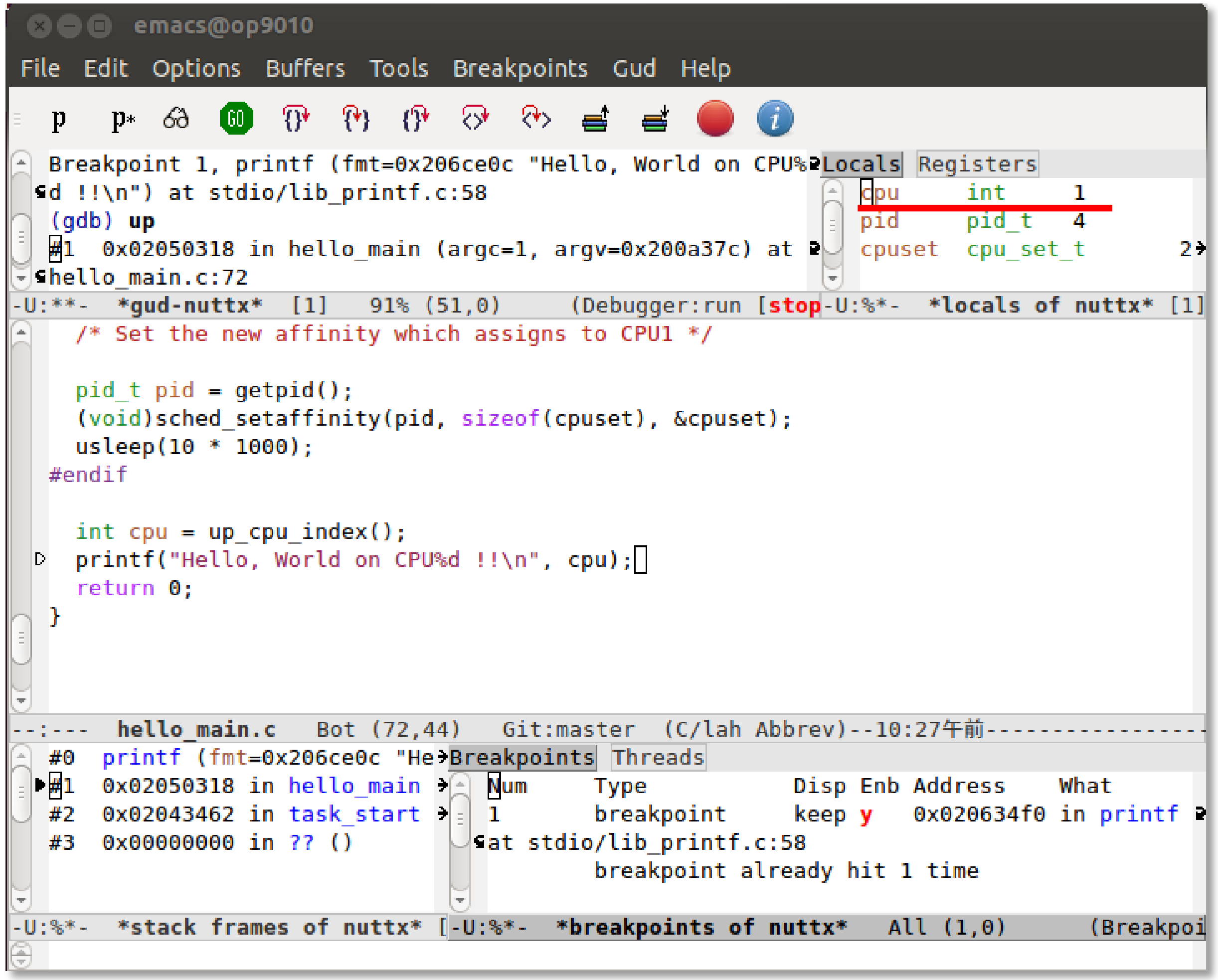
Debugging example

SONY

- Modify hello_main.c
  - Assign the current task to CPU1
  - Print CPU index.
- Add a break point at printf()
- Run "hello" on the nsh
- Break point hits on CPU1
- Check the trace log



```
340: 0a 03 00 00 00 00 a7 02 00 00          CPU0 PID  0: RESUME
350: 0b 02 00 00 00 00 c2 02 00 00 03       CPU0 PID  0: SUSPEND
361: 0a 03 64 00 03 00 c2 02 00 00          CPU0 PID  3: RESUME
371: 10 00 64 00 04 00 c2 02 00 00 68 65 6c 6c 6f 00 CPU0 PID  4: START
387: 0b 02 64 00 03 00 c2 02 00 00 07       CPU0 PID  3: SUSPEND
398: 0a 03 64 00 04 00 c2 02 00 00          CPU0 PID  4: RESUME
408: 0b 02 64 00 04 00 c2 02 00 00 07       CPU0 PID  4: SUSPEND
419: 0a 03 00 00 00 00 c2 02 00 00          CPU0 PID  0: RESUME
429: 0b 06 00 00 00 00 c4 02 00 00 01       CPU0 PID  0: CPU_PAUSE
440: 0b 02 00 01 01 00 c4 02 00 00 04       CPU1 PID  1: SUSPEND
451: 0a 07 00 01 01 00 c4 02 00 00          CPU1 PID  1: CPU_PAUSED
461: 0b 08 00 00 00 00 c4 02 00 00 01       CPU0 PID  0: CPU_RESUME
472: 0a 09 64 01 04 00 c4 02 00 00          CPU1 PID  4: CPU_RESUMED
482: 0a 03 64 01 04 00 c4 02 00 00          CPU1 PID  4: RESUME
```



```
Breakpoint 1, printf (fmt=0x206ce0c "Hello, World on CPU%
d !!\n") at stdio/lib_printf.c:58
(gdb) up
#1  0x02050318 in hello_main (argc=1, argv=0x200a37c) at
hello_main.c:72
-U:**-  *gud-nuttx*   [1]    91% (51,0)      (Debugger:run [stop
```

```
Locals Registers
cpu       int        1
pid       pid_t      4
cpuset    cpu_set_t          2
-U:%*-   *locals of nuttx* [1]
```

```
  /* Set the new affinity which assigns to CPU1 */

  pid_t pid = getpid();
  (void)sched_setaffinity(pid, sizeof(cpuset), &cpuset);
  usleep(10 * 1000);
#endif

  int cpu = up_cpu_index();
  printf("Hello, World on CPU%d !!\n", cpu);
  return 0;
}
--:---  hello_main.c   Bot (72,44)   Git:master  (C/lah Abbrev)--10:27午前----------------
```

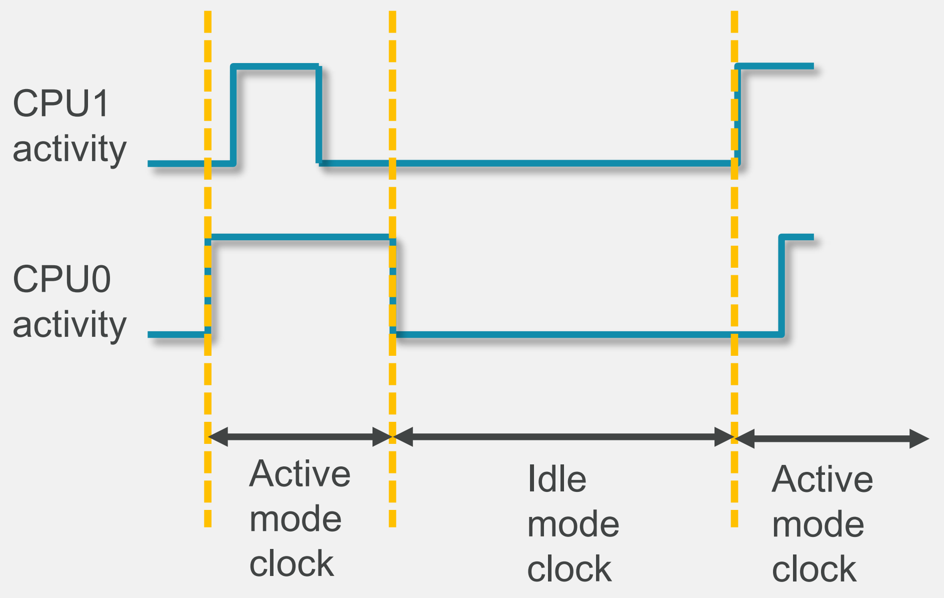
```
#0  printf (fmt=0x206ce0c "He
#1  0x02050318 in hello_main
#2  0x02043462 in task_start
#3  0x00000000 in ?? ()
-U:%*-  *stack frames of nuttx* [
```

```
Breakpoints Threads
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0x020634f0 in printf
at stdio/lib_printf.c:58
        breakpoint already hit 1 time
-U:%*-  *breakpoints of nuttx*   All (1,0)      (Breakpoi
```

THE LINUX FOUNDATION

Enhance DVFS for SMP*

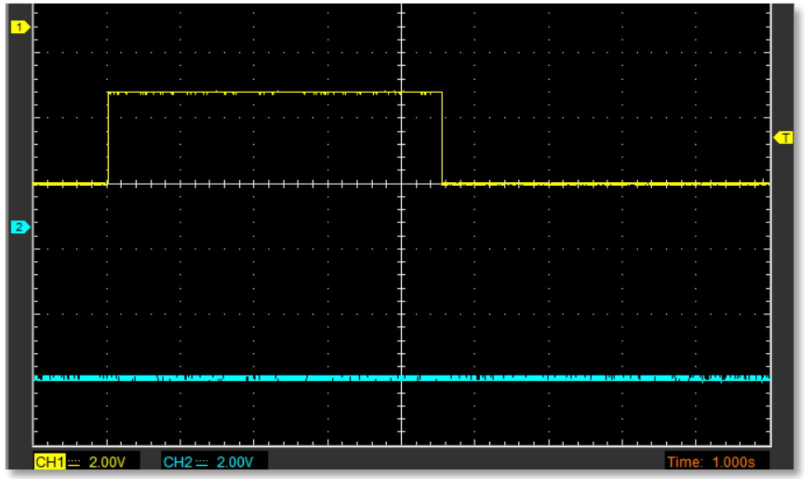
- Need to handle both CPUs
  - 1. If at least one CPU is active, the apply active mode clock.
  - 2. If both CPUs are idle (i.e. WFI), then apply idle mode clock
- Calculate CPU idle time on both CPUs
  - 3. If at least one CPU falls below lower threshold (e.g. 20% idle), then go to higher clock mode.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

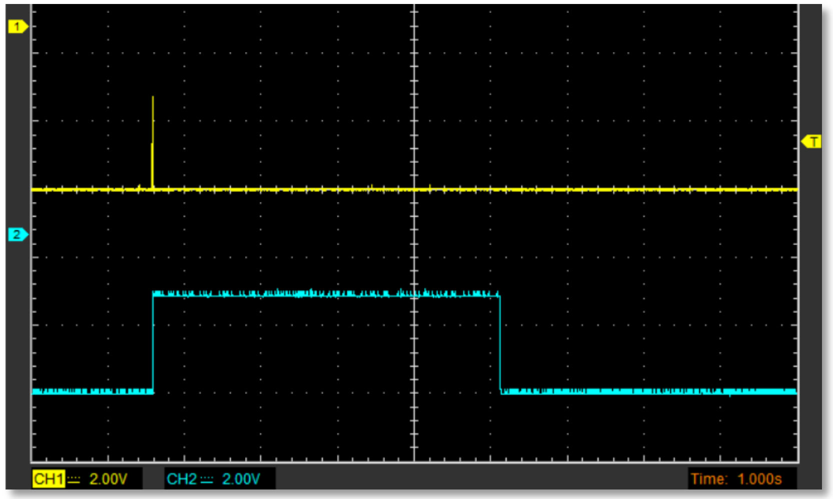
*LC823450 specific code is available at bitbucket.org/nuttx/nuttx

CPU activity examples* (1/2)



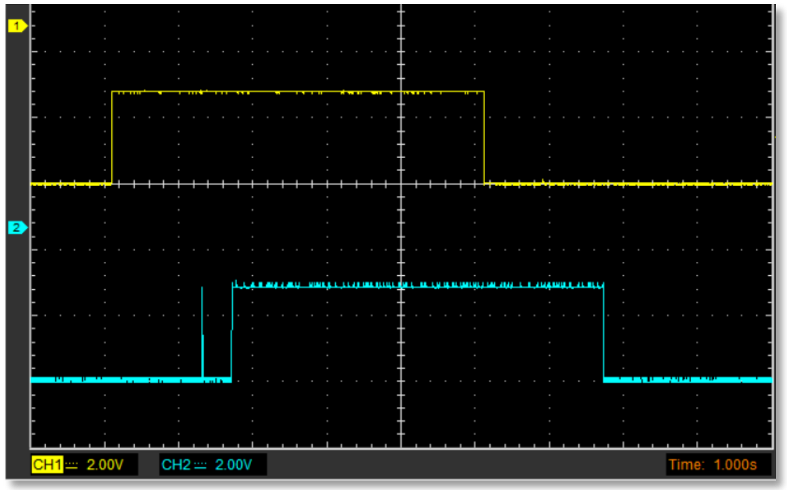
(1) busyloop x 1

nsh> taskset 3 busyloop



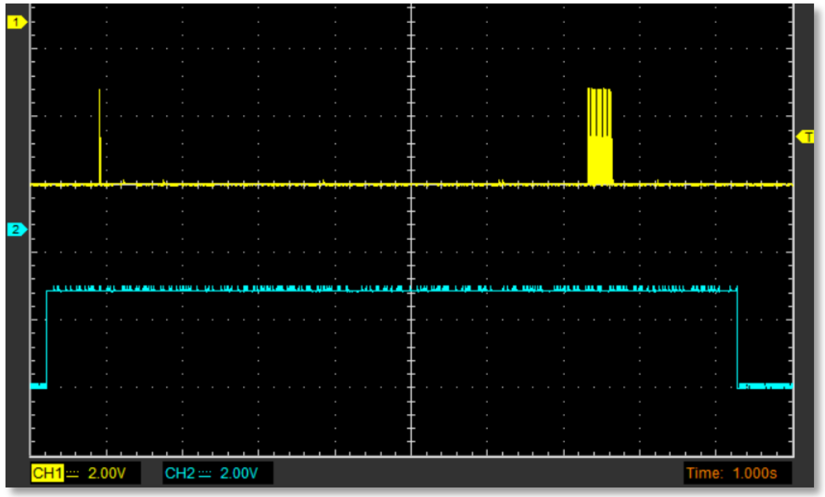
(3) busyloop x 1 bound to CPU1

nsh> taskset 2 busyloop &



(2) busyloop x 2

nsh> taskset 3 busyloop &
nsh> taskset 3 busyloop &



(4) busyloop x 2 bound to CPU1

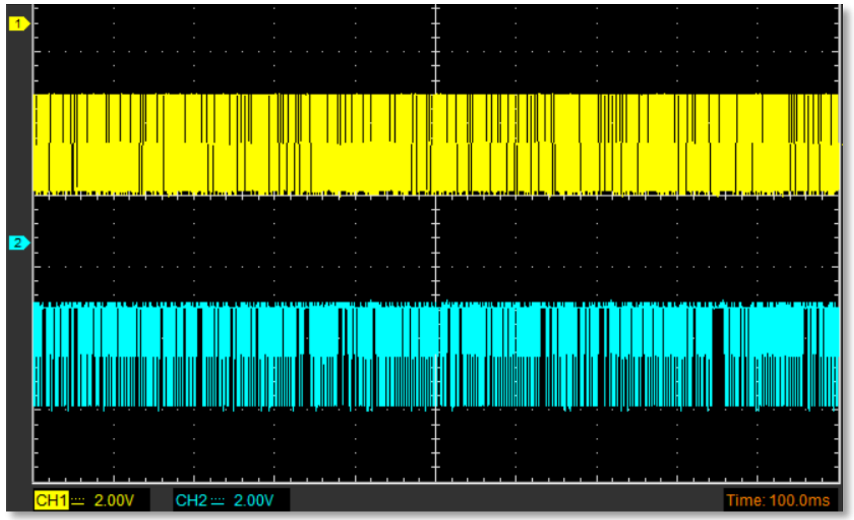
nsh> taskset 2 busyloop &
nsh> taskset 2 busyloop &

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

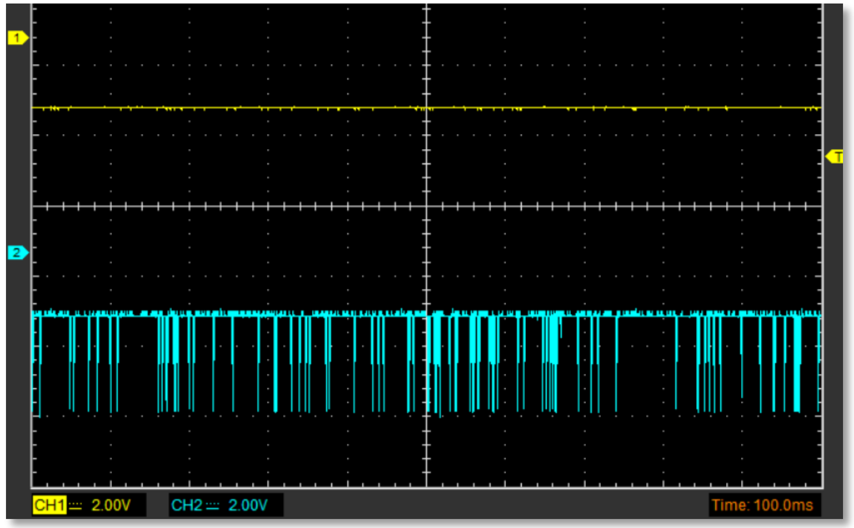
Usage: taskset mask command…
mask=1 means CPU0, mask=2 means CPU1, mask=3 means CPU0 or CPU1

CPU activity examples (2/2)

- Background
  - LC823450 has 3 SDIO controllers. eMMC is assigned to CH0 and uSD is assigned to CH1.
  - Accessing different channels will be faster than accessing the same channel.
- (1) Two md5 to the same file on eMMC
  - Concurrent access is impossible.
  - time 85.4sec (file size=44MB)
- (2) md5 to eMMC and md5 to uSD
  - Concurrent access is possible.
  - time 46.6sec & 53.0sec (file size=44MB)



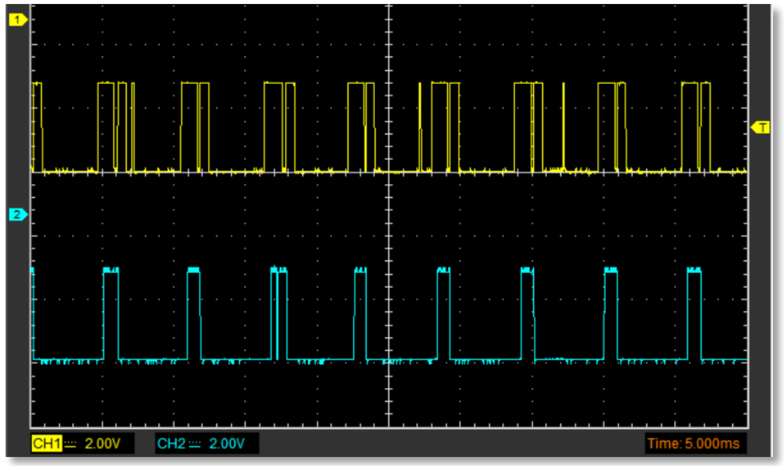
(1) Two md5 to the same file on eMMC



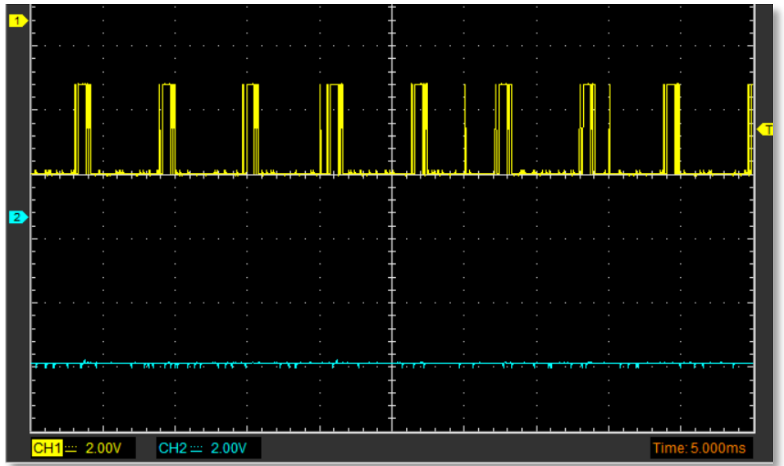
(2) md5 to eMMC and md5 to uSD

Power consumption comparison

- nxplayer with local playback
  - WAV file 44.1kHz/16bit/2ch
  - Vdd1=1.0V *
  - CPU clock = 40MHz (active), 6MHz(idle)
- Power consumption @Vdd1
  - SMP : 5.6mA (idle=3.5mA)
  - non-SMP : 4.2mA (idle=3.4mA)
  - SMP scheduling overhead is outstanding because CPU load is relatively low. However, more optimization would be possible.



SMP



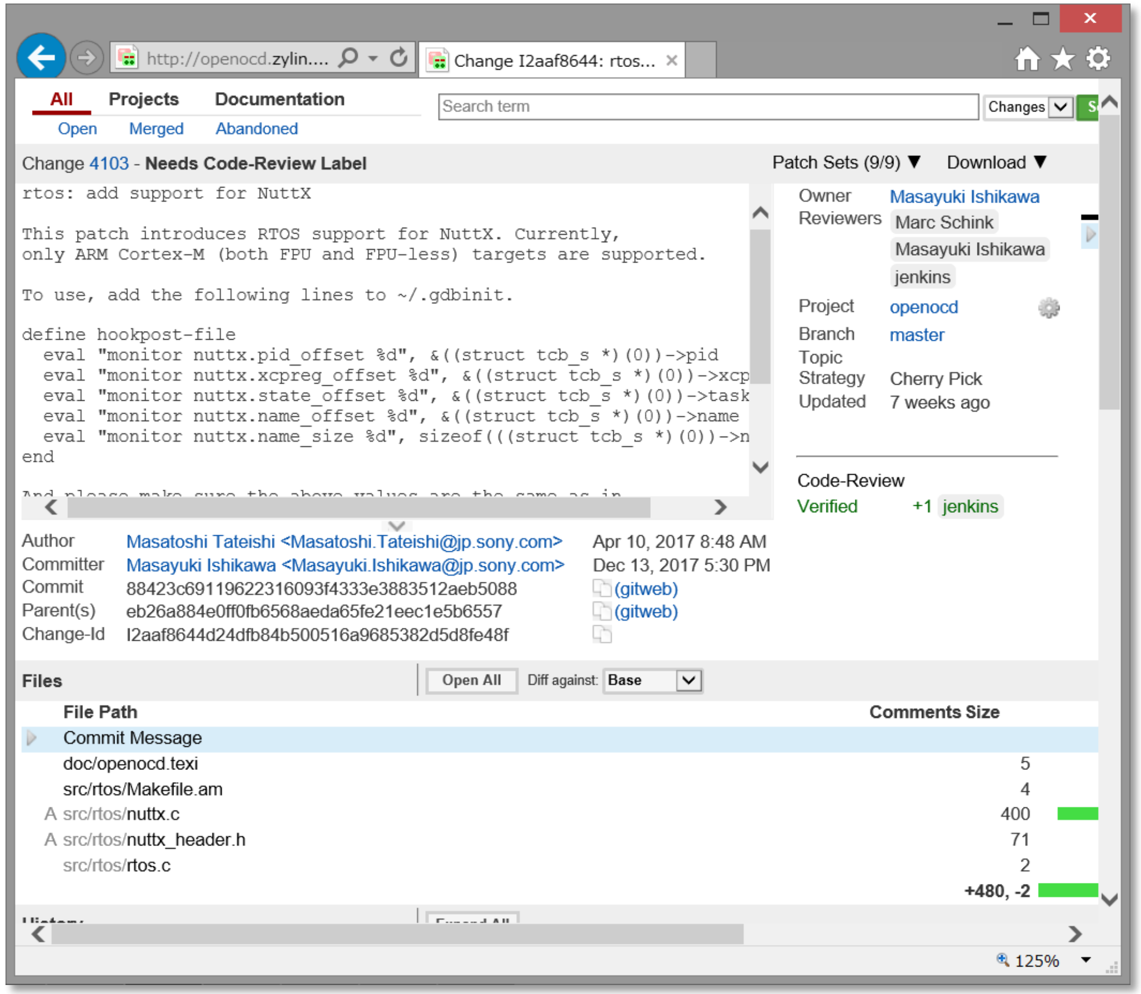
non-SMP

*Power consumption of the logic part (i.e. Cortex-M3, SRAM, DMA, I2S, …) inside the MCU

OpenOCD NuttX support status

- github.com/sony/openocd-nuttx
  - Initial release in Oct 2016
  - Merged 0.10.0 release
  - Merged Cortex-M4F support by Sony Semiconductor Solution group
  - Added LC823450 related scripts

- OpenOCD upstream *
  - Contribution started in Apr 2017
  - Review started in Dec 2017
  - Still open … (as of 26/Feb/2018)



Change 4103 - **Needs Code-Review Label**

```
rtos: add support for NuttX

This patch introduces RTOS support for NuttX. Currently,
only ARM Cortex-M (both FPU and FPU-less) targets are supported.

To use, add the following lines to ~/.gdbinit.

define hookpost-file
  eval "monitor nuttx.pid_offset %d", &((struct tcb_s *)(0))->pid
  eval "monitor nuttx.xcpreg_offset %d", &((struct tcb_s *)(0))->xcp
  eval "monitor nuttx.state_offset %d", &((struct tcb_s *)(0))->task
  eval "monitor nuttx.name_offset %d", &((struct tcb_s *)(0))->name
  eval "monitor nuttx.name_size %d", sizeof(((struct tcb_s *)(0))->n
end

And please make sure the above values are the same as in
```

Owner	Masayuki Ishikawa
Reviewers	Marc Schink, Masayuki Ishikawa, jenkins
Project	openocd
Branch	master
Topic	
Strategy	Cherry Pick
Updated	7 weeks ago

Code-Review
Verified +1 jenkins

Author	Masatoshi Tateishi <Masatoshi.Tateishi@jp.sony.com>	Apr 10, 2017 8:48 AM
Committer	Masayuki Ishikawa <Masayuki.Ishikawa@jp.sony.com>	Dec 13, 2017 5:30 PM
Commit	88423c69119622316093f4333e3883512aeb5088	(gitweb)
Parent(s)	eb26a884e0ff0fb6568aeda65fe21eec1e5b6557	(gitweb)
Change-Id	I2aaf8644d24dfb84b500516a9685382d5d8fe48f	

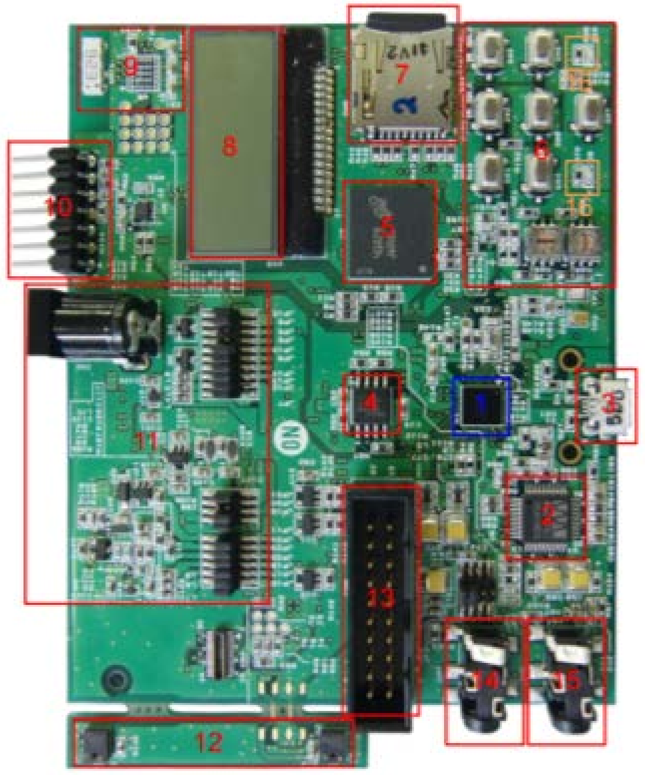
Files Open All Diff against: Base

	File Path	Comments	Size
	Commit Message		
	doc/openocd.texi		5
	src/rtos/Makefile.am		4
A	src/rtos/nuttx.c		400
A	src/rtos/nuttx_header.h		71
	src/rtos/rtos.c		2
			+480, -2

* http://openocd.zylin.com/#/c/4103/

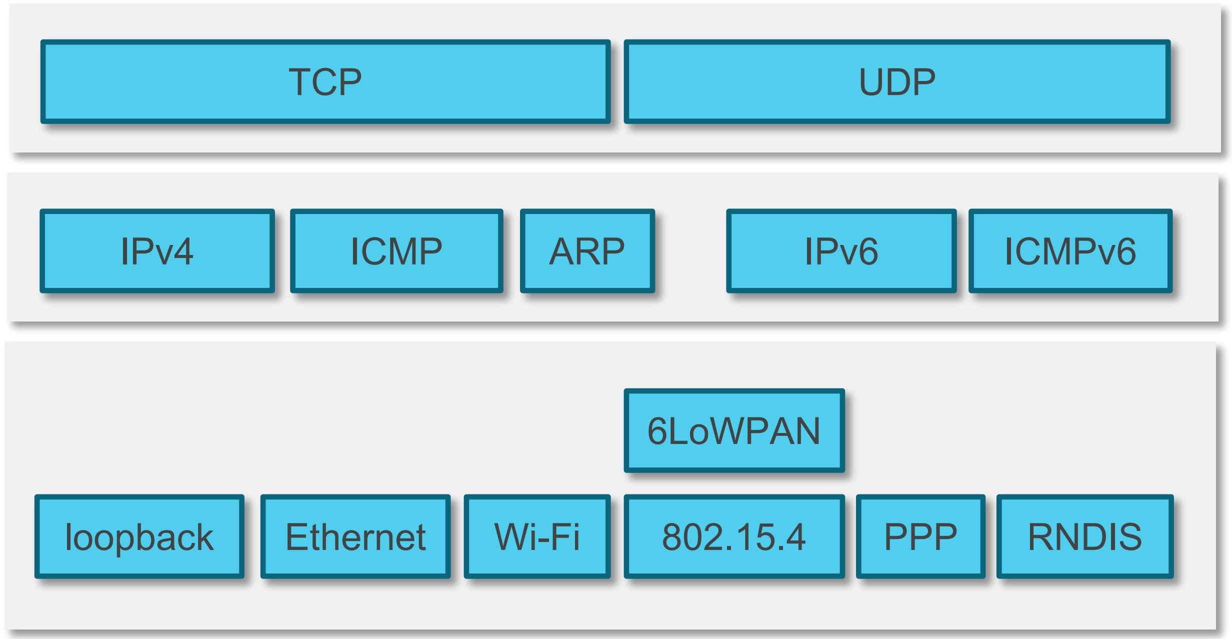
Networking with LC823450XGEVK

- 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
  - Run the network stack in SMP mode
  - Do various tests via telnet



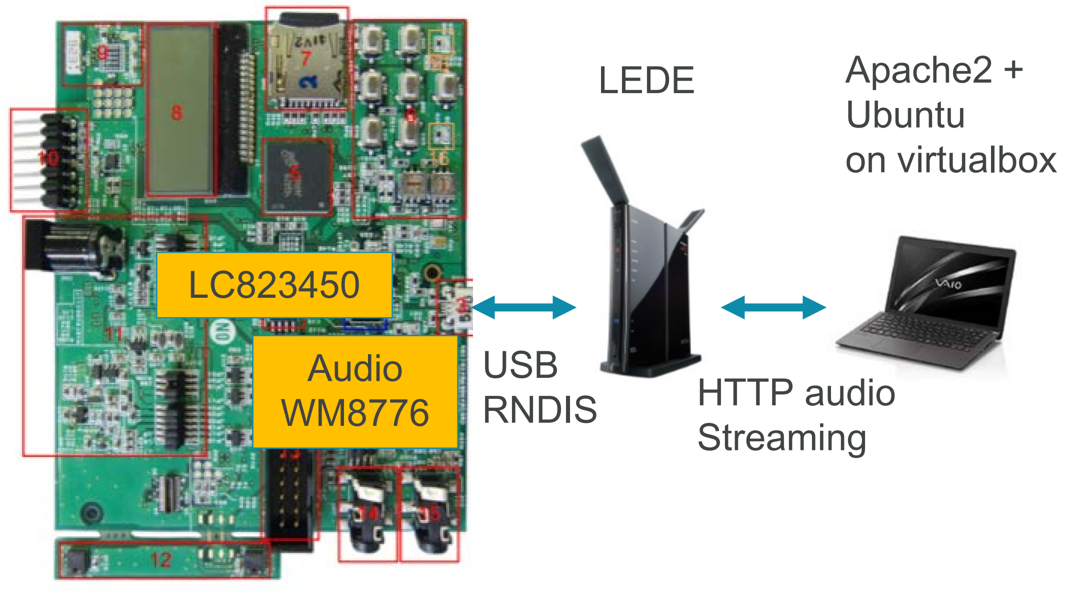
NuttX networking features

- Ethernet and IEEE 802.11 Full MAC
- 6LoWPAN for radio network drivers (IEEE 802.15.4 MAC)
- USB RNDIS **(Newly added in Sep 2017)**
- SLIP, TUN/PPP, local loopback devices
- IPv4, IPv6, TCP, UDP, ARP, ICMP, ICMPv6, IGMPv2
- ICMPv6 autonomous auto-configuration
- IP forwarding
- BSD compatible socket layer
- DNS name resolution / NetDB



HTTP audio streaming support *

- Fix RNDIS driver for NuttX
  - Fix data corruption
  - Add USB high speed mode support
- Modify tcp_send.c to support receive window control.
  - Still experimental
- Modify nxplayer to support HTTP connection.
  - Currently only WAV format is supported.
- Still testing with SMP kernel

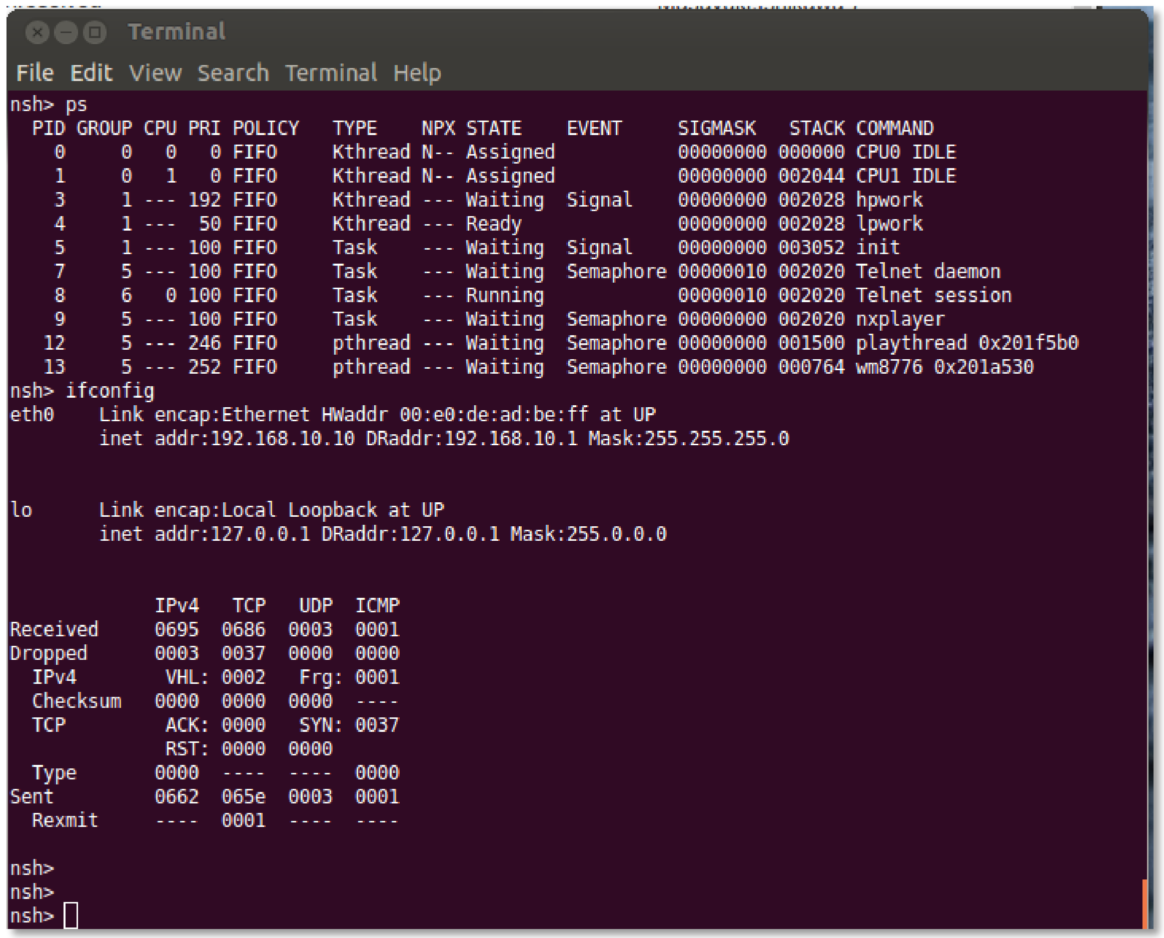


ON Semiconductor LC823450XGEVK evaluation kit with the latest NuttX

*Code is available at bitbucket.org/nuttx/nuttx

HTTP audio streaming example

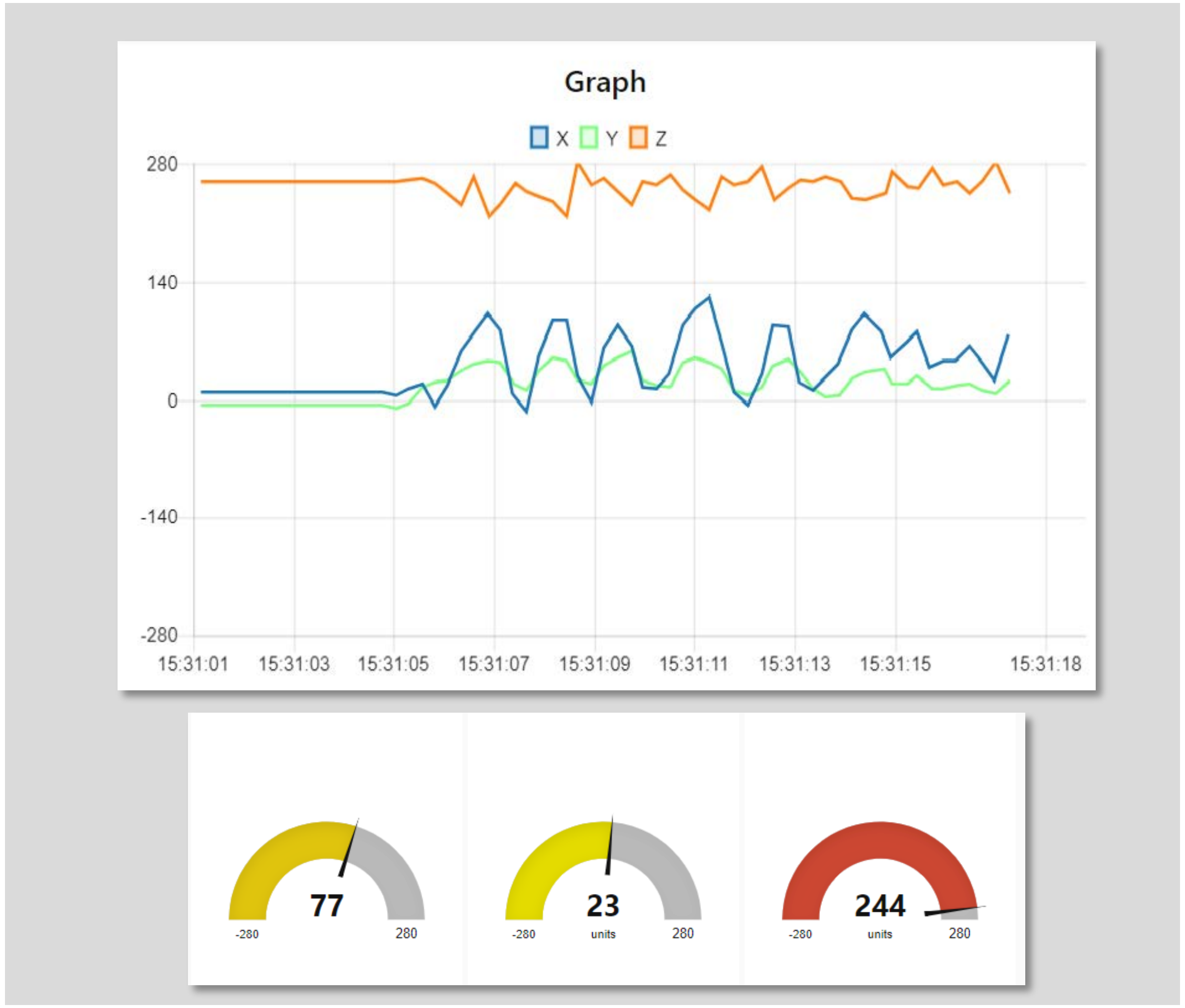
- ‘ps’ command results shows
  - Dual CPUs are running
  - telnet daemon is running
  - one telnet session is running
  - nxplayer is running
- ‘ifconfig’ command results shows
  - private address has been assigned via DHCP
  - TCP/UDP traffic



```
nsh> ps
  PID GROUP CPU PRI POLICY   TYPE    NPX STATE    EVENT     SIGMASK  STACK COMMAND
    0     0   0   0 FIFO     Kthread N-- Assigned           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 ---  50 FIFO     Kthread --- Ready              00000000 002028 lpwork
    5     1 --- 100 FIFO     Task    --- Waiting  Signal    00000000 003052 init
    7     5 --- 100 FIFO     Task    --- Waiting  Semaphore 00000010 002020 Telnet daemon
    8     6   0 100 FIFO     Task    --- Running            00000010 002020 Telnet session
    9     5 --- 100 FIFO     Task    --- Waiting  Semaphore 00000000 002020 nxplayer
   12     5 --- 246 FIFO     pthread --- Waiting  Semaphore 00000000 001500 playthread 0x201f5b0
   13     5 --- 252 FIFO     pthread --- Waiting  Semaphore 00000000 000764 wm8776 0x201a530
nsh> ifconfig
eth0    Link encap:Ethernet HWaddr 00:e0:de:ad:be:ff at UP
        inet addr:192.168.10.10 DRaddr:192.168.10.1 Mask:255.255.255.0


lo      Link encap:Local Loopback at UP
        inet addr:127.0.0.1 DRaddr:127.0.0.1 Mask:255.0.0.0



             IPv4  TCP   UDP  ICMP
Received     0695  0686  0003  0001
Dropped      0003  0037  0000  0000
  IPv4        VHL: 0002   Frg: 0001
  Checksum   0000  0000  0000  ----
  TCP         ACK: 0000   SYN: 0037
              RST: 0000  0000
  Type       0000  ----  ----  0000
Sent         0662  065e  0003  0001
  Rexmit     ----  0001  ----  ----

nsh>
nsh>
nsh> 
```

MQTT example with Bluemix *

- What is MQTT?
  - MQ Telemetry Transport
  - Useful to send telemetry data such as accelerometer.
- What is Bluemix?
  - A cloud platform as a service developed by IBM
  - You can create IoT applications with Node-Red on Bluemix
- MQTT library
  - Eclipse Paho MQTT C/C++ client library for Embedded platforms
  - https://github.com/eclipse/paho.mqtt.embedded-c

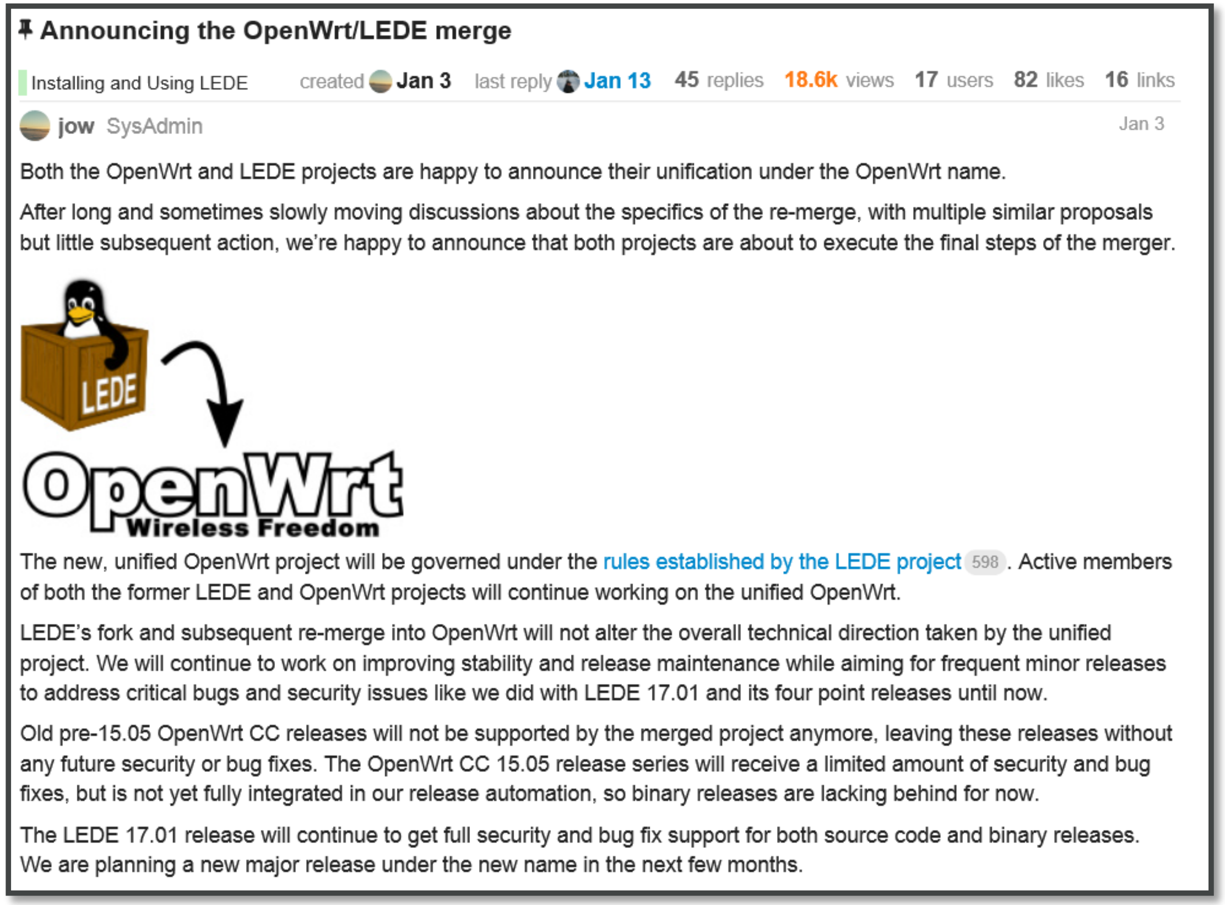


*Code is NOT merged yet

Introduction to LEDE

- Motivation
  - Build a shareable network testing environment for NuttX
- Software
  - LEDE project as of ELC2017 session
  - The project was forked from OpenWRT that is famous OSS for the router world as a turn key solution but they became one again (at the beginning of 2018)
- Hardware
  - WZR-HP-G300NH (buffalo) Wi-Fi router with USB 2.0 port

WZR-HP-G300NH


Announcing the OpenWrt/LEDE merge

Installing and Using LEDE   created Jan 3   last reply Jan 13   45 replies   18.6k views   17 users   82 likes   16 links

jow SysAdmin   Jan 3

Both the OpenWrt and LEDE projects are happy to announce their unification under the OpenWrt name.

After long and sometimes slowly moving discussions about the specifics of the re-merge, with multiple similar proposals but little subsequent action, we're happy to announce that both projects are about to execute the final steps of the merger.

The new, unified OpenWrt project will be governed under the rules established by the LEDE project 598. Active members of both the former LEDE and OpenWrt projects will continue working on the unified OpenWrt.

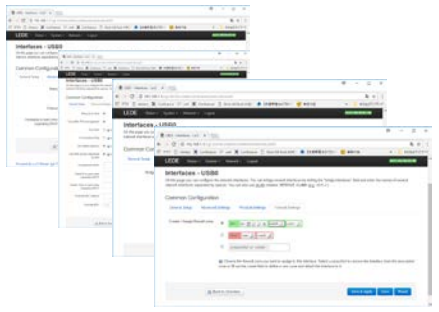
LEDE's fork and subsequent re-merge into OpenWrt will not alter the overall technical direction taken by the unified project. We will continue to work on improving stability and release maintenance while aiming for frequent minor releases to address critical bugs and security issues like we did with LEDE 17.01 and its four point releases until now.

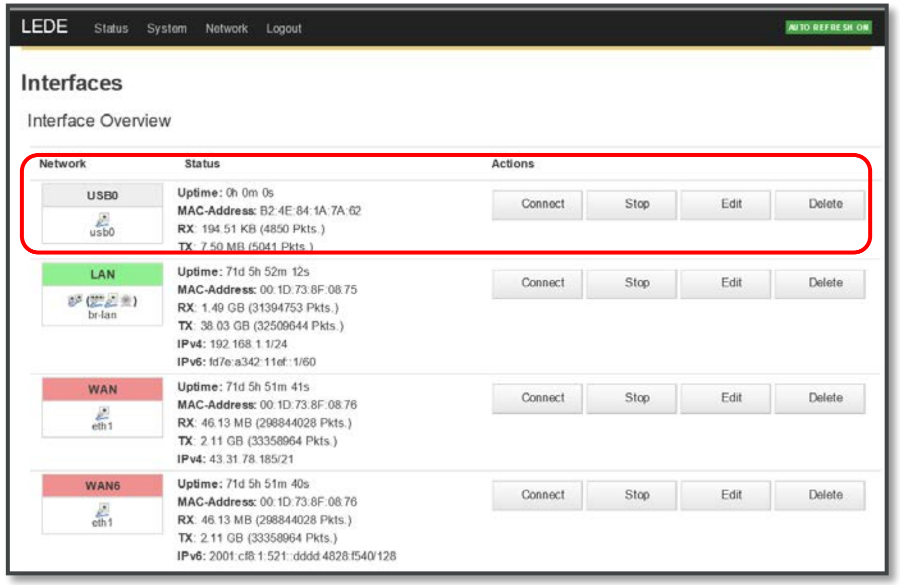
Old pre-15.05 OpenWrt CC releases will not be supported by the merged project anymore, leaving these releases without any future security or bug fixes. The OpenWrt CC 15.05 release series will receive a limited amount of security and bug fixes, but is not yet fully integrated in our release automation, so binary releases are lacking behind for now.

The LEDE 17.01 release will continue to get full security and bug fix support for both source code and binary releases. We are planning a new major release under the new name in the next few months.

Support RNDIS on LEDE

- How to setup
  - Modify configuration
  - Add network USB0 (RNDIS) via LuCI
  - Change the network setting of USB0

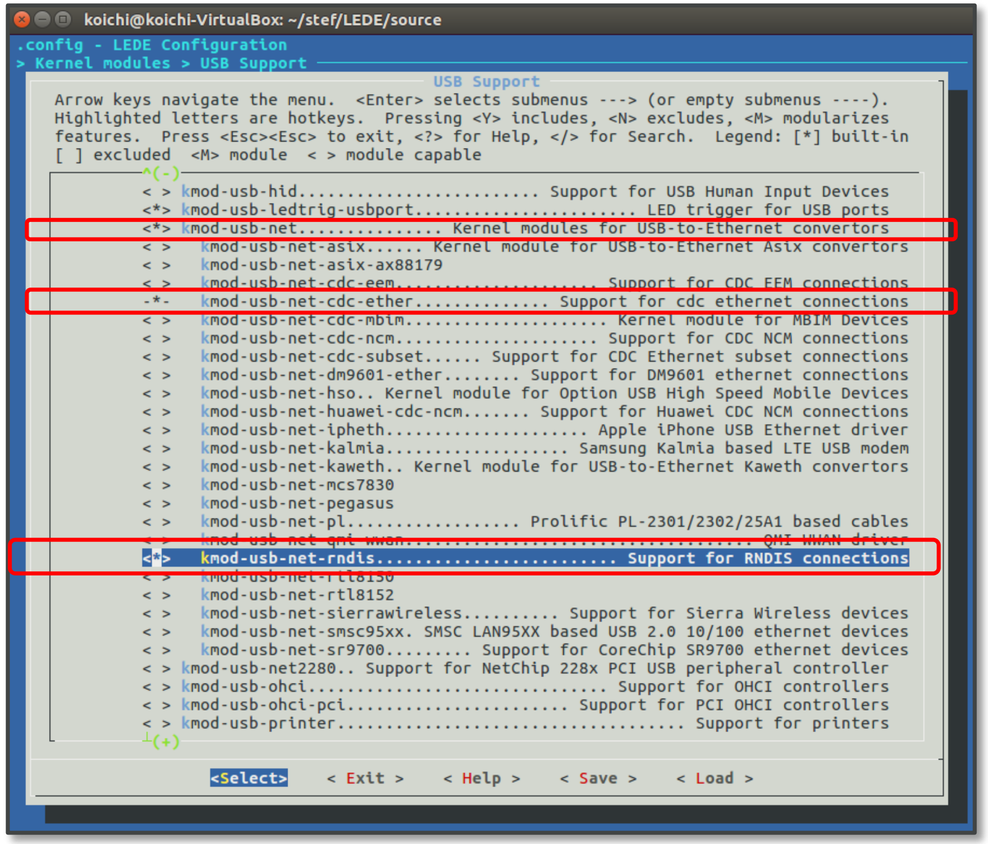




Interfaces

Interface Overview

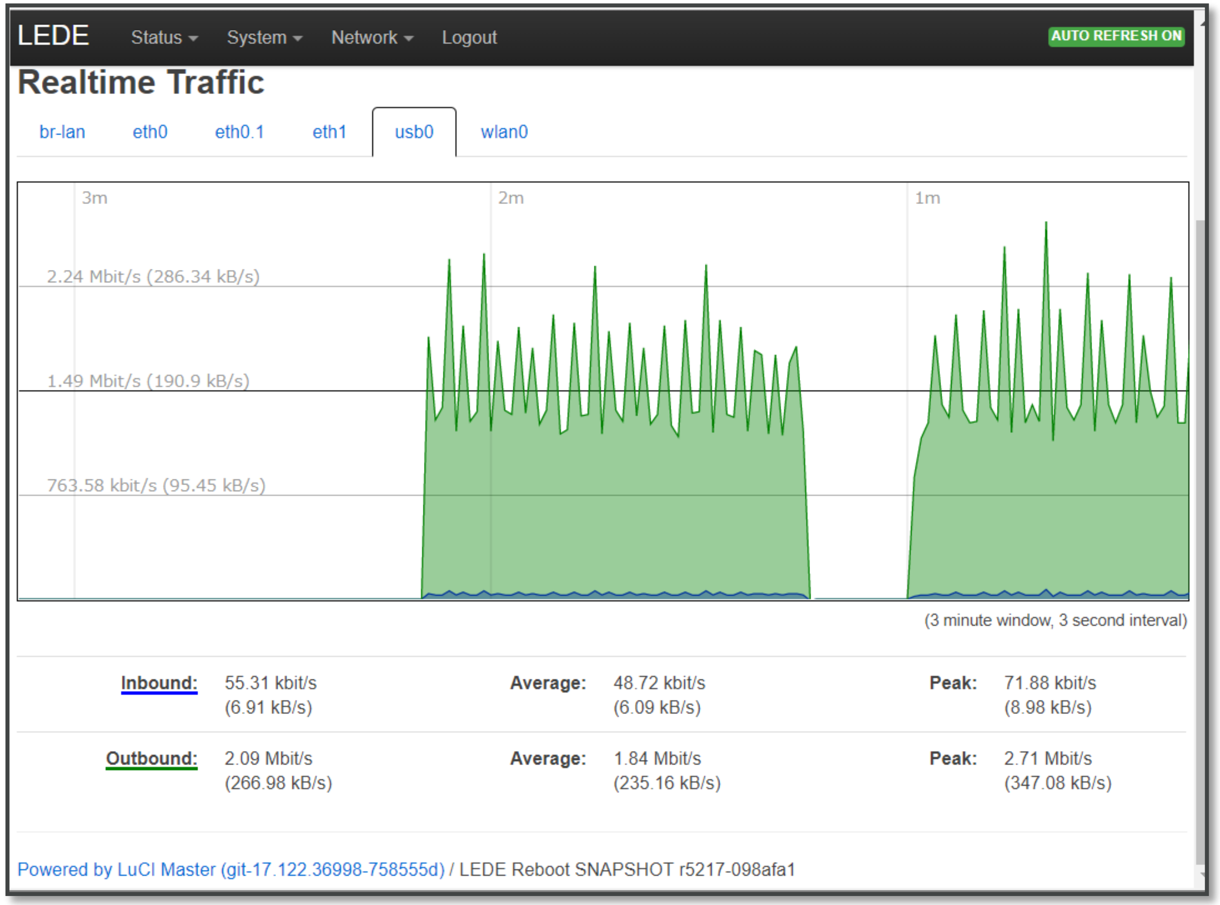
Network	Status	Actions
USB0 usb0	**Uptime:** 0h 0m 0s **MAC-Address:** B2:4E:84:1A:7A:62 **RX:** 194.51 KB (4850 Pkts.) **TX:** 7.50 MB (5041 Pkts.)	Connect Stop Edit Delete
LAN br-lan	**Uptime:** 71d 5h 52m 12s **MAC-Address:** 00:1D:73:8F:08:75 **RX:** 1.49 GB (31394753 Pkts.) **TX:** 38.03 GB (32509644 Pkts.) **IPv4:** 192.168.1.1/24 **IPv6:** fd7e:a342:11ef::1/60	Connect Stop Edit Delete
WAN eth1	**Uptime:** 71d 5h 51m 41s **MAC-Address:** 00:1D:73:8F:08:76 **RX:** 46.13 MB (298844028 Pkts.) **TX:** 2.11 GB (33358964 Pkts.) **IPv4:** 43.31.78.185/21	Connect Stop Edit Delete
WAN6 eth1	**Uptime:** 71d 5h 51m 40s **MAC-Address:** 00:1D:73:8F:08:76 **RX:** 46.13 MB (298844028 Pkts.) **TX:** 2.11 GB (33358964 Pkts.) **IPv6:** 2001:cf8:1:521::dddd:4828:f540/128	Connect Stop Edit Delete



```
koichi@koichi-VirtualBox: ~/stef/LEDE/source
.config - LEDE Configuration
> Kernel modules > USB Support ──────────────────────────────
                               USB Support
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
  features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable
          ^(-)
          < > kmod-usb-hid........................ Support for USB Human Input Devices
          <*> kmod-usb-ledtrig-usbport......................... LED trigger for USB ports
          <*> kmod-usb-net.............. Kernel modules for USB-to-Ethernet convertors
          < >   kmod-usb-net-asix...... Kernel module for USB-to-Ethernet Asix convertors
          < >   kmod-usb-net-asix-ax88179
          < >   kmod-usb-net-cdc-eem..................... Support for CDC EEM connections
          -*-   kmod-usb-net-cdc-ether............. Support for cdc ethernet connections
          < >   kmod-usb-net-cdc-mbim..................... Kernel module for MBIM Devices
          < >   kmod-usb-net-cdc-ncm..................... Support for CDC NCM connections
          < >   kmod-usb-net-cdc-subset...... Support for CDC Ethernet subset connections
          < >   kmod-usb-net-dm9601-ether........ Support for DM9601 ethernet connections
          < >   kmod-usb-net-hso.. Kernel module for Option USB High Speed Mobile Devices
          < >   kmod-usb-net-huawei-cdc-ncm....... Support for Huawei CDC NCM connections
          < >   kmod-usb-net-ipheth..................... Apple iPhone USB Ethernet driver
          < >   kmod-usb-net-kalmia.................. Samsung Kalmia based LTE USB modem
          < >   kmod-usb-net-kaweth.. Kernel module for USB-to-Ethernet Kaweth convertors
          < >   kmod-usb-net-mcs7830
          < >   kmod-usb-net-pegasus
          < >   kmod-usb-net-pl................. Prolific PL-2301/2302/25A1 based cables
          < >   kmod-usb-net-qmi-wwan.................................... QMI WWAN driver
          <*>   kmod-usb-net-rndis........................ Support for RNDIS connections
          < >   kmod-usb-net-rtl8150
          < >   kmod-usb-net-rtl8152
          < >   kmod-usb-net-sierrawireless.......... Support for Sierra Wireless devices
          < >   kmod-usb-net-smsc95xx. SMSC LAN95XX based USB 2.0 10/100 ethernet devices
          < >   kmod-usb-net-sr9700......... Support for CoreChip SR9700 ethernet devices
          < > kmod-usb-net2280.. Support for NetChip 228x PCI USB peripheral controller
          < > kmod-usb-ohci............................. Support for OHCI controllers
          < > kmod-usb-ohci-pci....................... Support for PCI OHCI controllers
          < > kmod-usb-printer................................. Support for printers
          └(+)

            <Select>    < Exit >    < Help >    < Save >    < Load >
```

Network traffic and CPU activity

SONY



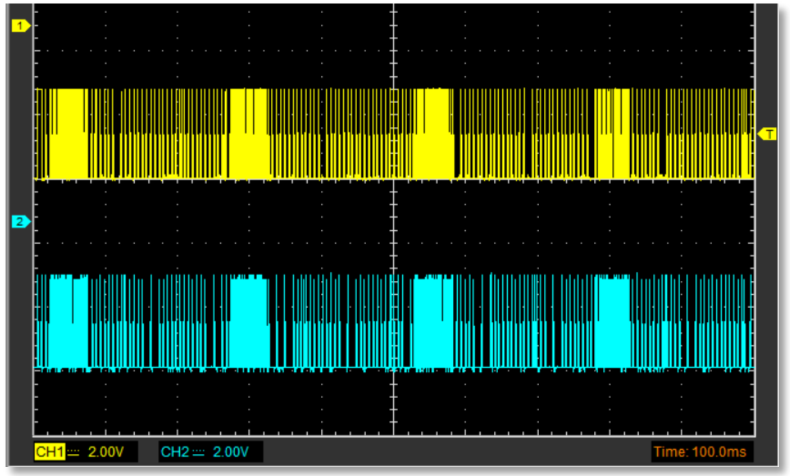
Realtime Traffic

(3 minute window, 3 second interval)

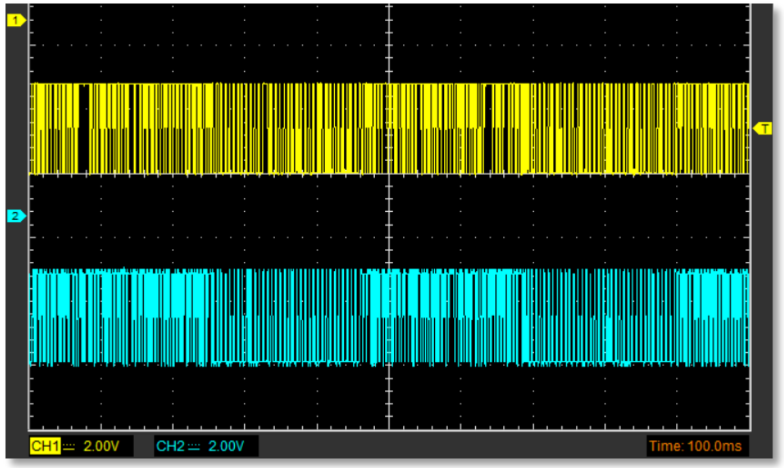
Inbound:	55.31 kbit/s (6.91 kB/s)	**Average:**	48.72 kbit/s (6.09 kB/s)	**Peak:**	71.88 kbit/s (8.98 kB/s)
Outbound:	2.09 Mbit/s (266.98 kB/s)	**Average:**	1.84 Mbit/s (235.16 kB/s)	**Peak:**	2.71 Mbit/s (347.08 kB/s)

Powered by LuCI Master (git-17.122.36998-758555d) / LEDE Reboot SNAPSHOT r5217-098afa1

Network traffic when HTTP audio streaming is working



Active clock = 160MHz



Active clock = 40MHz

THE LINUX FOUNDATION

Demo videos

- CPU activity examples (busyloop, md5)
- ‘smp’ app & ‘ostest’ app
- MQTT + Bluemix
- HTTP audio streaming + other tasks

Future challenges

- SMP related
  - Improve stability and performance
  - Contribute OpenOCD LC823450-SMP support
  - Real-time trace via OpenOCD
  - CPU hotplug and dynamic scheduler switching
  - Per-CPU interrupt handling
- Networking related
  - Improve TCP flow control
  - Bluetooth IP network

Acknowledgement

- We specially thank Mr. Gregory Nutt who is the author of NuttX. He discussed SMP related issues with us and helped us merge our code to the upstream.

- Also, we appreciate ON Semiconductor disclosed their technical documents.

Any Questions?