An Incubator Project in the Apache Software Foundation

http://mynewt.apache.org/

11 April 2016
Apache Mynewt

Open Source OS for Constrained IoT
- MCU / Hardware independent
- ARM Cortex-M*, AVR, MIPS, more

http://mynewt.apache.org/
About Me: Sterling Hughes

• CTO / Co-Founder, Runtime

• 20+ Years Experience in Open Source

• 10+ Years Experience in Embedded & Wireless Networking
  – Designed, developed, deployed Industrial IoT mesh networks
  – More than 20M+ connected devices
First release of a successful IoT product...

...now make that repeatable, please.
An Embedded OS for MCUs

- Community-driven Open Source
- Build and Package Management integral to OS
- Direct and obvious solution for a problem
- Scalability driven by simplicity

<table>
<thead>
<tr>
<th>NewtMgr</th>
<th>HTTP</th>
<th>CoAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Security</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wi-Fi</td>
<td>Thread</td>
</tr>
<tr>
<td>Stats + Logging</td>
<td>Console + Shell</td>
<td>Config + Upgrade</td>
</tr>
<tr>
<td>Kernel / RTOS</td>
<td>HAL</td>
<td></td>
</tr>
<tr>
<td>Secure Bootloader, FFS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Moore’s Law Hit Microcontrollers
A Community Effort

Why the ASF?

- Liberal Apache 2.0 license
- Individuals, not Corporations
- Meritocracy
- Strong licensing and IP policies
- Long history of working with large organizations: IBM, Oracle, Pivotal/EMC
- Many years experience managing large, complex projects: Apache Web Server, Hadoop, Cassandra, Kafka, Subversion, etc.

Community-driven Open Source: Best Way to Maintain Healthy User Ecosystem
Apache Mynewt Users

(today)
• Bluetooth connected products
  • Medical: everything
  • Consumer/Enterprise: locks, lights
  • Industrial
• Makers
  • Home
  • Hardware Labs
  • Clothing

(tomorrow)
• Industrial Wireless Sensor Networks
• Wi-Fi Products
• Who Knows?

• Power Optimization
• Mesh networking
• Security
• Sensor Algorithms
• Control Systems

Scale Makes Problems Interesting
• 280,000 lines of code

• Initial support for Simulator, Nordic NRF51/52, STM32F3/4 and Arduino

• Active contributors, We Want You!
  ✓ http://mynewt.apache.org/
## Goals

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maintain and re-use packages across multiple products</strong></td>
</tr>
<tr>
<td><strong>Manage debug and production build setups</strong></td>
</tr>
<tr>
<td><strong>Make it easy to find, install 3rd party libraries</strong></td>
</tr>
<tr>
<td><strong>Efficiency: use only what you need</strong></td>
</tr>
</tbody>
</table>

## Description

- Everything is a package. Each package describes its dependencies to the rest of the world.
- A collection of packages is called an application.
- There are a few special packages: BSP and Project. Project contains `main()` and BSP defines linker script and hardware layout.
- Targets are used to combine projects and BSP.
- Packages can be distributed, upgraded and installed remotely.

Compose across multiple repos
Project Structure (coming soon)
“core” broken into appropriate sub-projects

- ASF governance structure (PMCs) corresponds with sub-package structure
- ASF repositories clean, Apache 2.0 license
- Users can assemble projects sourced from multiple repos

Composability across multiple repos provides flexibility
BSP and HAL

Goals

- Provide quality drivers for major MCU platforms
- Design for cross-platform: well-defined APIs for HAL, BSP and drivers
- Make it easy to add board specific definitions

Description

- BSP definition is provided in <app>/hw/bsp
- HAL definitions are in hw/hal and contain uniform, cross-platform APIs
- MCU definitions are in hw/mcu and provide implementations for various MCUs
  - Hierarchy allows code-reuse within MCU families
- BSPs for common dev kits are available as packages (e.g., Nordic nRF51/2DK)
- BSPs depend on MCUs
- BSP + MCU provides implementation for HAL APIs

Chip Vendors: We Want You!
Kernel

- Pre-emptive, multi-tasking RTOS
  - Strict, priority-based scheduling
  - Up to 253 different priority levels
- Tickless kernel
- Power management
- Resource utilization tracking
- Built-in tasks:
  - Idle
  - Sanity

```c
#include <os/os.h>
#include <assert.h>

/* Task 1 */
#define TASK1_PRIO (1)
#define TASK1_STACK_SIZE OS_STACK_ALIGN(1024)

struct os_task task1;
struct os_stack_stack1[TASK1_STACK_SIZE];
static volatile int g_task1_loops;

void
task1_handler(void *arg)
{
    while (1) {
        g_task1_loops++;
        /* Wait one second */
        os_time_delay(1000);
    }
}

int
main(int argc, char **argv)
{
    int rc;
    os_init();
    os_task_init(&task1, "task1", task1_handler, NULL,
                 TASK1_PRIO, OS_WAIT_FOREVER, stack1, TASK1_STACK_SIZE);
    os_start();
    assert(0);
    return (0);
}
```
Energy Efficient Event Model

- Event Queues provide a mechanism for “mostly sleeping” asynchronous tasks
  - Wake-up on:
    - Message from another task
    - Timer
    - I/O state change
    - Incoming packet
    - Watchdog
  - Perform operations:
    - Send an alert
    - Respond to a request
    - Schedule a wakeup
  - Go back to sleep

```
struct os_eventq task1_evq;
struct os_eventq task2_evq;

#define OS_EVENT_T_PING (OS_EVENT_PERUSER)
#define OS_EVENT_T_PONG (OS_EVENT_PERUSER + 1)

void
task1_handler(void *arg)
{
    struct os_event *ev;
    struct os_event ping_ev;
    ping_ev.ev_type = OS_EVENT_T_PING;
    ping_ev.ev_arg = NULL;
    os_eventq_put(&task2_evq, &ping_ev);

    while (1) {
        ev = os_eventq_get(&task1_evq);
        assert(ev->ev_type == OS_EVENT_T_PONG);
        os_eventq_put(&task2_evq, &ping_ev);
        ++g_task1_loops;

        /* Wait one second */
        os_time_delay(1000);
    }
}

void
task2_handler(void *arg)
{
    struct os_event *ev;
    struct os_event pong_ev;
    pong_ev.ev_type = OS_EVENT_T_PONG;
    pong_ev.ev_arg = NULL;

    while (1) {
        ev = os_eventq_get(&task2_evq);
        assert(ev->ev_type == OS_EVENT_T_PING);
        os_eventq_put(&task1_evq, &pong_ev);
        ++g_task2_loops;
    }
}
```
Energy Efficient Event Model

- Bootloader can be located in ROM or Flash
  - Options for internal and external flashes
- Performs integrity check and swaps images
- Images contain SHA-256 hash and RSA signature
- NFFS optional
  - Provides a log-structured flash filesystem designed for small flashes
- CB (Circular Buffer) optional
  - Provides implementation of flash circular buffer
## System Security

<table>
<thead>
<tr>
<th>What We’re Protecting</th>
<th>How We Protect It</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootloader</td>
<td>• First stage bootloader can operate from ROM and verify signature of second stage bootloader</td>
</tr>
</tbody>
</table>
| Images                | • All images have SHA-256 of image contents  
                        | • Images support signing with ECC or RSA 2048 bit signatures  
                        | • Second stage bootloader can verify image signature |
| Network Interfaces    | • Full support for BLE 4.2 security at 1.0 release, including link-layer and app-layer |
Simulator and Test Framework

- In addition to various MCU ports: OS, HAL, FS and the majority of packages can run on Mac, Linux
- Develop your code on the host and then port to the real hardware
- Unit test framework is incorporated to all of the packages: ability to run unit tests on simulated environment and real hardware
- OS and libraries are fully regression tested to ensure API compatibility between releases

```c
/* Test some error cases */
TEST_ASSERT(os_mutex_init(NULL) == CS_INVALID_PARAM);
TEST_ASSERT(os_mutex_delete(NULL) == CS_INVALID_PARAM);
TEST_ASSERT(os_mutex_release(NULL) == CS_INVALID_PARAM);
TEST_ASSERT(os_mutex_pend(NULL, 0) == CS_INVALID_PARAM);

/* Get the mutex */
err = os_mutex_pend(mu, 0);
TEST_ASSERT(err == 0,
    "Did not get free mutex immediately (err=%d)", err);

/* Check mutex internals */
TEST_ASSERT(mu->mu_owner == t && mu->mu_level == 1 &&
    mu->mu_prio == t->t_prio && SLIST_EMPTY(&mu->mu_head),
    "Mutex internals not correct after getting mutex\n"
    "Mutex: owner=prio=level= head=p\n"
    "Task: task=prio=u",
    mu->mu_owner, mu->mu_prio, mu->mu_level,
    SLIST_FIRST(&mu->mu_head),
    t, t->t_prio);

/* Get the mutex again: should be level 2 */
err = os_mutex_pend(mu, 0);
TEST_ASSERT(err == 0, "Did not get my mutex immediately (err=%d)", err);
```
Apache Mynewt Roadmap

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>V0.8</td>
<td>V0.9</td>
<td>V0.10</td>
<td>V0.11</td>
<td>V1.0-b1</td>
<td>V1.0-GA</td>
</tr>
</tbody>
</table>

**Highlights**

v0.8: First release, BLE 4.2, FFS, Kernel, Console, Shell, Secure Boot  
v0.9: Expanded HW support and HAL  
v0.10: Wi-Fi & IP support  
v0.11: Full Bluetooth Support / Qualification  
v1.0 (GA) API compatibility, Full Regression Testing
THANK YOU

http://mynewt.apache.org/

Mailing List: dev@mynewt.incubator.apache.org

IRC: #mynewt on freenode