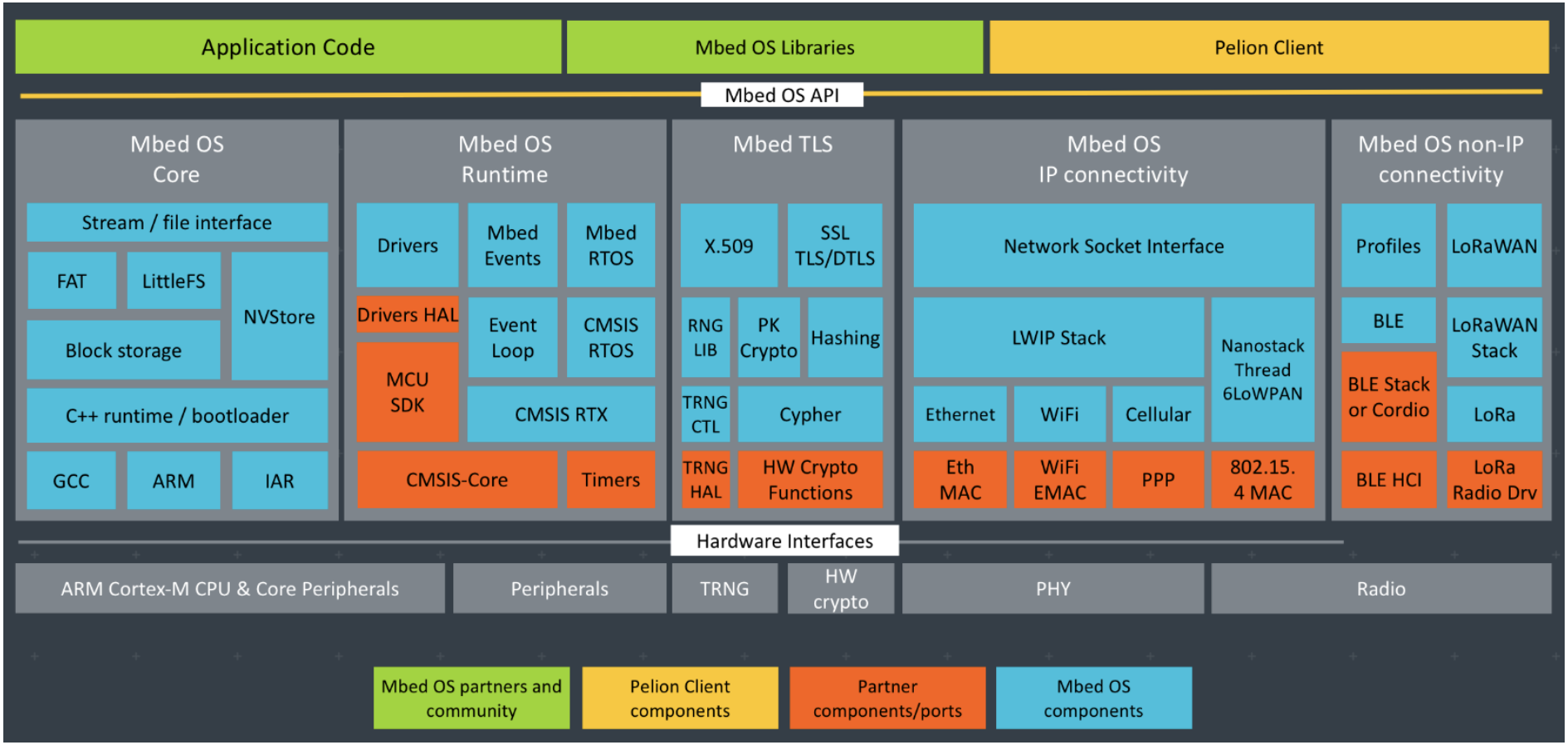
Embedded systems engineering
Distributed real-time systems

Alun Moon

MBED OS


Application Code
Mbed OS Libraries
Pelion Client
Mbed OS API
Mbed OS Core
Stream / file interface
FAT
LittleFS
NVStore
Block storage
C++ runtime / bootloader
GCC
ARM
IAR
Mbed OS Runtime
Drivers
Mbed Events
Mbed RTOS
Drivers HAL
Event Loop
CMSIS RTOS
MCU SDK
CMSIS RTX
CMSIS-Core
Timers
Mbed TLS
X.509
SSL TLS/DTLS
RNG LIB
PK Crypto
Hashing
TRNG CTL
Cypher
TRNG HAL
HW Crypto Functions
Mbed OS IP connectivity
Network Socket Interface
LWIP Stack
Nanostack Thread 6LoWPAN
Ethernet
WiFi
Cellular
Eth MAC
WiFi EMAC
PPP
802.15.4 MAC
Mbed OS non-IP connectivity
Profiles
LoRaWAN
BLE
LoRaWAN Stack
BLE Stack or Cordio
LoRa
BLE HCI
LoRa Radio Drv
Hardware Interfaces
ARM Cortex-M CPU & Core Peripherals
Peripherals
TRNG
HW crypto
PHY
Radio
Mbed OS partners and community
Pelion Client components
Partner components/ports
Mbed OS components

Documentation

MBED has a fairly full set of documentation.

```
https://os.mbed.com/docs/v5.10/apis/index.html
```

Each class has the documentation for the functions it provides, and some examples. (Some need a little interpretation/translation to our setup)

The RTOS has an overview at

```
https://os.mbed.com/docs/v5.10/apis/rtos.html
```

MBED RTOS

Characteristics

- deterministic
- multithreaded
- real-time
- pre-emptive

The RTOS primitives are always available

- threads
- semaphores
- mutexes
- events and eventqueue

Threads

Threads can be created and started in much the same way as in other OSs

```
Thread thread;
thread.start(taskfn);
```

where the parameter to `start` is the thread task function

thread task

```
void taskfn(void)
{
    while(true) {
        /* ... do something ...*/
    }
}
```

Threads

Priorities

Threads can be created with a given priority

```
Thread thread( osPriority );
```

The values that can be used for priorities range over 48 values

lowest	`osPriorityLow`, `osPriorityLow1`...`osPriority7`
	`osPriorityBelowNormal`
	`osPriorityNormal`
	`osPriorityAboveNormal`
	`osPriorityHigh`
Highest	`osPriorityRealtime`, `osPriorityRealtime1`...`osPriorityRealtime7`

Semaphores

Semaphores are provided for synchronisation and resource access.

creation

```
Semaphore sync;
```

wait, pend,

```
sync.wait();
```

release, post,

```
sync.release();
```

Semaphores

initial state

Semaphores can be created with an initial state,

```
Semaphore resource(2);
```

Recall the behaviour of wait and release

wait depends on semaphore state s

- $s > 0$ decrement state by 1
- $s = 0$ wait until state is $\neq 0$

release increment state by 1

Semaphores

uses

The LCD takes a long time to update. The Thread handling the display can be made to wait until data is available for it.

Semaphore

```
Semaphore available;
```

LCD thread

```
while(true) {
    available.wait();
    lcd.printf("etc"... );
}
```

Data acquisition

```
available.release();
```

Events

Events are provided to trigger and handle actions in the code.

EventQueue is provided to manage and dispatch events

The EventQueue needs to run in a thread, I've found the simplest way is to put it in `main`

```
EventQueue queue;
int main(void)
{

    queue.dispatch();
}
```

the `dispatch` function does not return if called with no parameters.

Events

simple events

Event handlers are usually functions that take no parameters and return no values, similar to ISRs.

simple events

```
// immediate
queue.call(red_on);

// after delay in ms
queue.call_in(500, green_on);

// repeats every period in ms
queue.call_every(1000, blink);
```

Events

Event objects and ISR

The event queue can create an event object that is suitable for use as an ISR handler to trigger the event.

falling edge ISR

```
sw2.fall(  queue.event(display) );
```

- The ISR triggers the event.
- The event handler runs in the context of the `EventQueue` thread.
- The event handler can be a long process or one not allowed in an ISR.
- Be aware of effects on the event-queue if using a *very long* or *blocking* handler.

Examples and exercises

1. https://github.com/dr-alun-moon/blinky-mbed-rtos
2. https://github.com/dr-alun-moon/blinky-mbed-semaphore
3. https://github.com/dr-alun-moon/blinky-mbed-semaphore-alt
4. https://github.com/dr-alun-moon/blinky-mbed-events