FreeRTOS Start Up - Preliminaries

Dr. Kuan-Hsun Chen

LS 12, TU Dortmund

October 24, 2019
Hardware Considerations

- Hardware-dependent layer
  - Talk to the chip architecture you choose
- FreeRTOS chips with all the hardware-independent
  - ARM7, ARM Cortex-M3, STM32, x86, etc.
Project Structure

- **Official website:** [http://www.freertos.org](http://www.freertos.org)
  - News, documents, downloads, demos, etc.
- **3 core source files**
  - tasks.c: subroutines for maintaining the tasks
  - queue.c: subroutines for maintaining the message passing queues
  - list.c: utilities for maintaining data structures for “lists”
Create A FreeRTOS Project (Optional)

- https://www.freertos.org/Eclipse_CDT.html
- Eclipse
  - C/C++ IDE
  - import a demo project for starting
- makefile + scripts or cmake
  - configure the makefile for compilations and executions
- Choose a correct port for the hardware/platform
## Data Types in FreeRTOS

<table>
<thead>
<tr>
<th>macro or typedef</th>
<th>actual type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TickType_t</td>
<td>to store the tick count value and to specify block times usually 16-bit or 32-bit unsigned integer</td>
</tr>
<tr>
<td>BaseType_t</td>
<td>represent the most efficient data type for the architecture usually align with the bits of the architecture</td>
</tr>
</tbody>
</table>

- Signed or unsigned `char` types should be always specified
- Plain `int` types should never be used
Variables and Functions Naming

- **Variables**: prefixes
  - 'c': for char
  - 's': for short
  - 'l': for long
  - 'x': for portBASE_TYPE and any others
  - 'u': for unsigned
  - 'p': pointer
  - combinations are possible

- **Functions**: prefixes
  - by the returning data type
  - 'v': for void

- **Common macros**:
  - pdTRUE is 1, pdFALSE is 0
  - pdPASS is 1, pdFAIL is 0
Configurations: FreeRTOSConfig.h

Some important fields/features:

- `configUSE_PREEMPTION`: This is set to 1 if the preemptive kernel is desired.
- `configUSE_IDLE_HOOK`: An idle task hook will execute a function during each cycle of the idle task.
- `configUSE_TICK_HOOK`: A tick hook function will execute on each RTOS tick interrupt if this value is set to 1.
- `configTICK_RATE_HZ`: This is the frequency at which the RTOS tick will operate.
- `configMAX_PRIORITIES`: The total number of priority levels that can be assigned when prioritizing a task.
- `configUSE_COUNTING_SEMAPHORES`: This is set to 1 if counting semaphores are required.
- `configUSE_MUTEXES`: This is set to 1 if mutexes are needed. Priority inheritance will then be enforced.
- etc...