Week 7

* Exercise #1: The Basics of Real-Time Linux
  + <http://www.cs.ru.nl/lab/xenomai/exercises_xenomai2.4/ex01/Exercise-1.html>
* What are RTOSs?
  + They offer real time scheduling guarantees
  + Have a limited set of tasks and processes running
  + Some of the features included
    - Priority based scheduler
      * Highest priority task runs until
        + It finishes
        + The task is preempted

A higher priority task is made ready to run

Original task pre-empted by newer priority task

* + - System clock interrupt routine
    - Deterministic behavior
* Mercury provides real-time capability in to ways

|  |  |
| --- | --- |
| Cobalt  Co-ernel variant of Xenomai 3 | Mercury  Native Linux variant of Xenomai 3 |
| * Uses co-kernel alongside linux kernel   + This co-kernel is deemed real-time capable     - Aka these APIs are real time capable * Better performance than Mercury | * Real-time APIs are emulated over native linux POSIX   + Uses PREEMPT-RT * Performance Limitations of this method   + PREEMPT\_RT can have unwanted impact on co-located non RT workloads |

* What is POSIX?
  + Portable Operating System Interface
    - IEEE standards to ensure operating system compatibility
* RT preempt vs RTAI vs Xenomai for real-time linux
  + <https://stackoverflow.com/questions/31109364/rt-preempt-vs-rtai-vs-xenomai-for-real-time-linux>
  + Cobalt has better performance than Mercury (using PREEMPT\_RT from native linux kernel)
* Overview of Xenomai (Real time Framework for linux)
  + <https://elinux.org/images/7/76/Kiszka.pdf>
* Definitions
  + Cobalt–Co-kernel variant of Xenomai 3
  + Mercury–Native Linux variant of Xenomai 3
  + Alchemy–Xenomai-own real-time API
  + Copperplate–Library layer for building RTOS APIs
  + Boilerplate–Internal utility Library
  + Trank–Library to support porting from Xenomai 2 to 3
  + RTDM–Real-Time Driver Model, kernel API that enablesRT drivers, specifically for Cobalt
  + Analogy–RTDM drivers for digital/analogue converters
  + Adeos–Original interrupt pipeline for Linux,used by early Xenomai 2 versions
  + I-pipe–Evolution and simplification of Adeos
  + Dovetail–New architecture of Linux extensions to hook Xenomai 3 into Linux
* RTOS and threading
  + RTOSs boosts functionality and efficiency
  + But it may not be the best system for multithreading because of its poor threading priority
* Callbacks
  + <https://en.wikipedia.org/wiki/Callback_(computer_programming)>
  + Callbacks (also known as a “call-after” function)
  + Illustration
    - CodeA (executable\_callback\_code1 , executable\_callback\_code2, executable\_callback\_code3)
    - Arguments to CodeA
      * executable\_callback\_code1
      * executable\_callback\_code2
      * executable\_callback\_code3
  + CodeA is expected to call back argument (executable\_callback\_code1) at a given time
  + Synchronous vs asynchronous callback
    - Synchronous callback
      * (also called Blocking callback)
      * In this case
        + executable\_callback\_code1 is invoked before CodeA returns
    - Asynchronous callback
      * (also called Deferred callback)
      * In this case
        + executable\_callback\_code1 is invoked before CodeA returns
      * Often used in context of I/O operations or event handling
      * Asynchronous callbacks are called by either
        + Interrupts
        + Different thread (in case of multiple threads)
  + How are callbacks implemented?
  + The form of a callback varies among programming languages
    - In [assembly](https://en.wikipedia.org/wiki/Assembly_language), [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B), [Pascal](https://en.wikipedia.org/wiki/Pascal_(programming_language)), [Modula2](https://en.wikipedia.org/wiki/Modula2) and similar languages:
      * **a machine-level** [**pointer**](https://en.wikipedia.org/wiki/Function_pointer) **to a function** may be passed as an argument to another (internal or external) function (CodeA in our example).
    - This is **supported by most compilers** and
      * + provides the advantage of using different languages together without special wrapper libraries or classes
    - One example may be the [Windows API](https://en.wikipedia.org/wiki/Windows_API) that is directly (more or less) accessible by many different languages, compilers and assemblers
* Sensor
  + Realsense SDK runs on
    - Linux
      * x86
    - Windows
    - Android
  + <https://www.intelrealsense.com/developers/>