RTAI Modules

- rtai
  - Core Module
    - initializes all of its control variables and structures.
    - makes a copy of the idt_table.
    - makes a copy of the Linux irq handlers entry addresses.
    - initializes the interrupts chips (ic) management specific functions.
  - But when you install rtai with the usual `insmod rtai` command nothing happens, as rtai is a dormant module. You must specifically
    - mount it when is needed by other modules calling `rt_mount_rtai()` to activate it.
    - unmount it as well when it is not required anymore, by calling `rt_umount_rtai()`, which put rtai back into its bed to sleep.
  - The mount call activates rtai and, even if you do not use any of its services Linux work toward the hardware is filtered by rtai. The most important thing happening when you mount RTAI is that from that very instant Linux is no more in power of disabling/enabling interrupts. From that point on rtai will assure that interrupt enables/disables will be consistent intra Linux but Linux could be preempted at any time by the higher authority of rtai, the only master of the hardware.

- rtai_sched
  - The real time scheduler module is in charge of distributing the CPU to different tasks present in the system, including Linux. The scheduling occurs when tasks perform certain system calls and on timer handler activation (each 8254 interrupt).
  - The scheduler makes it elected the first highest priority task in a READY state. RTAI considers the priority 0 as the highest priority and 0x3fffFfff the lowest.
  - Linux is given priority 0x7fffFfff. Given a priority level, the first initialized task will be the first elected and will run to completion unless a
task with a higher priority is elected or it terminates or the task calls a blocking system function.

- RTAI supports both periodic and oneshot mode for the real time scheduler. You have three different schedulers:
  - $UP$, only for uniprocessors
  - $SMP$, for multiprocessors
  - $MUP$, only for multiprocessors

- The scheduler services are:
  - Task functions
  - Timing functions
  - Semaphore functions
  - Mailbox functions
  - Intertask communication functions

- All the functions can be used with any scheduler. Note that rtai is mounted automatically when you load rtai_sched.

- rtai_fifos
  - The module that implements the fifo services for RTAI.

- Many applications appear to benefit from a synergy between the real-time system side and the Linux side, for example for managing the data logging and displaying. Simple fifo buffers are used to do this; they are called real time fifos. The real-time task interface includes creation, destruction, reading and writing functions, performed by the rtai_fifos module.

- Linux user processes, on the other hand, see rt-fifos as ordinary character devices. Note that on the module side you always have only non blocking put/get, so that any different policy should be enforced by using appropriate user handler functions.

- Even if fifos are strictly no more required in RTAI, because of the availability of LXRT, fifos are kept for both compatibility reasons and because they are very useful tools to be used to communicate with interrupt handlers, since they do not require any scheduler to be installed.
In this sense you can see this new implementation of fifos as a kind of universal form of device drivers, since once you have your interrupt handler installed you can use fifo services to do all the rest.

- rtai_shm
- lxrt
- rtai_pqueue
- rtai_pthread
- rtai_utils