The legacy API consists of methods defined in libbsp/shared/src/rq-legacy.c . The prototypes for these functions are written in cpukit/include/rtems/irq.h
Following are the rules/recommendations that must be followed while updating legacy code from RTEMS BSPs. They are:

- The replacement of legacy methods should happen like this:

BSP_get_current_rtems_irq_handler() → Obsolete
BSP_install_rtems_irq_handler() → rtem_interrupt_handler_install()
BSP_install_rtems_shared_irq_handler() → rtems_interrupt_handler_install()
BSP_remove_rtems_irq_handler() → rtems_interrupt_handler_remove()
BSP_rtems_irq_mngt_set() → bsp_interrupt_initialize()
BSP_rtems_irq_mngt_get → Obsolete

- All uses of rtems_irq_connect_data type and other data types from legacy API should be removed and replaced with suitable data types if required.

rtems_irq_connect_data → Use components of this data type individually
rtems_irq_number → rtems_vector_number
rtems_irq_hdl_param → void *
rtems_irq_hdl → void *

- Remove functions if they're defined but not used, or if they're empty.
- Insert prototypes for non-global functions(having no declarations in a header file) in order to remove any compiler warnings.
- Make functions static if they're only used in that specific file.
- Obtain status of install() & remove() functions in a rtems_status_code status and check it's value with an assert().
- While using install() method from the new API, the on() function needs to be called if it's defined, and it's not empty.
- While using remove() method from the new API, the off() function needs to be called before, if it's defined, and it's not empty.
- The implementation of is_enabled/is_on() helper from previous structure should be removed if it's not used already.
- Inserting a declaration of a global function in the same file is a bad hack.