Embedded System Programming
Project 4

Task1:
Interrupt handling is a critical task , it has much impact on the responsiveness of the system. Interrupts are handled by the Interrupt Service Routine(ISR). While ISR is executing , the same interrupt cannot be processed. Thus it is important that ISR finishes it work as fast as possible, so that other interrupt can be processed. So to do this we defer time consuming or blocking operations to the bottom half generally done by tasklet, softirq or workqueues.

psmouse-base.c [2]

Whenever an activated mouse is moved or clicked , an interrupt is generated and mouse sends 3 bytes per movement or click. This interrupt is handled by the psmouse_interrupt ISR which handles the bytes and then decide whether to pass them for normal processing or gather them as a command response.
During the ISR it also checks if mouse is in the sync or not . If not , then it assigns the re-sync work to the workqueue since it is a time consuming work and it can also go to sleep which should not happen in the ISR.
In psmouse_resync() function :

a) It try to acquire mutex lock which might sleep and thus first reason for using workqueue.
b) It then sends PSMOUSE_CMD_DISABLE using ps2_sendbytes() and tries reconnecting after 20 msec for 3 times before reporting failure , as given in comments “ps2_sendbyte() can only be called from a process context.” [1] and thus another reason to use workqueue.
c) Polls to check if mouse is ready.
d) Now it tries to enable the mouse, if fails , sleep for 200 msec and retry again for 5 times before reporting failure.
e) Release the mutex lock.

The overall psmouse_resync work can take upto 1060 msec and also it tries to acquire mutex lock which can lead to blocking and also the sleep function where it is waiting for the response from mouse. These all scenarios make this task to be taken out of ISR and given to the workqueue. Thus making the ISR to return soon and process next interrupt generated by mouse .

serio.c [3]

In this file, it uses pre-build workqueue used for long events. And adds the work in that queue for processing. Function given as a work is serio_handle_event(). This function get the event by serio_get_event() function and performs task based on the type of event received.
Different kind of work performed are register port, reconnect port , rescan port, reconnect subtree , attach driver. Each of which is long task and can sleep during the execution and thus is taken out of the ISR top half and given to the workqueue.

References:
[1] : http://lxr.free-electrons.com/source/drivers/input/serio/libps2.c#L38
[2] : http://lxr.free-electrons.com/source/drivers/input/mouse/psmouse-base.c?v=3.6;a=arm
[3] : http://lxr.free-electrons.com/source/drivers/input/serio/serio.c?v=3.6;a=arm