* Interruptions (asynchrone généré par HW)
* Exceptions (synchrone généré par proc) : les appels systemes en font partis

1 interrupt handler pour chaque IRQ et donc pour chaque device

Interrupt handling : Top half et Bottom half :

* Les top désactive IRQ et refilent le taf aux bottom (execution ultra rapide) : il s’agit des interrupt handlers.
* Les bottom eux procede au traitement de l’IRQ avec haut niveau de prio mais IRQ activés

SoftIRQ, tasklets, workqueues dans cette catégorie

Register an interrupt handler :

/\* request\_irq: allocate a given interrupt line \*/

int request\_irq(unsigned int irq,

irq\_handler\_t handler,

unsigned long flags,

const char \*name,

void \*dev)

Enlever l’inscription d’un interrupt handler :

void free\_irq(unsigned int irq, void \*dev)

Handler :

static irqreturn\_t intr\_handler(int irq, void \*dev)

Execution d’un IRQ :

unsigned int do\_IRQ(struct pt\_regs regs)

Il va acquitter le HW et désactiver les IRQ sur cette ligne puis appeler le bon handler si aucun n’est activé sur cette ligne puis appel handle\_IRQ\_event()