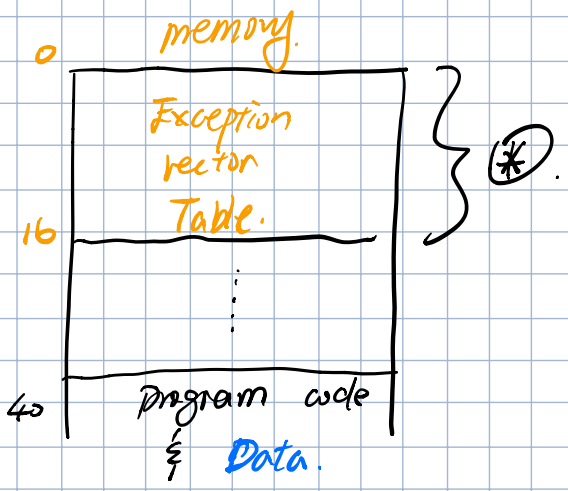


(*)

```
.section .vector, "ax"
    B   _Start
    B   SERVICE_UND
    B   OS_CODE
    B   SERVICE_ABORT_INST
    B   SERVICE_ABORT_DATA
.word.  0
    B   IRQ_HANDLER
    B   SERVICE FIQ
```

Interrupts. (Hardware-generated Exceptions)

Example: assume pushbutton KEY port is set up to cause interrupts.

```
_Start:  Inst
         Inst
         :
         Inst
```

Set up interrupts (*)

```
LOOP:    Inst
         Inst
```

← (*) Press KEY

0x18: B IRQ_HANDLER

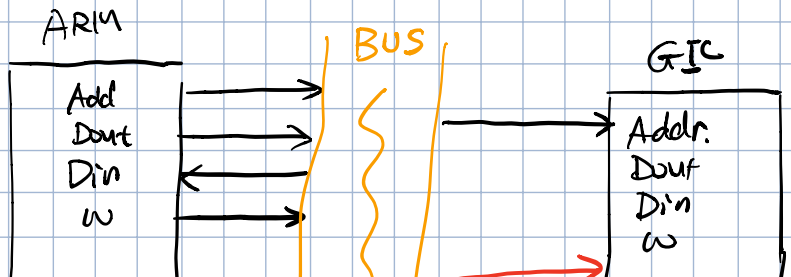
```
        :
      Inst.
        :
      B   LOOP

IRQ_HANDLER:  ....  ⊛ (later)
```

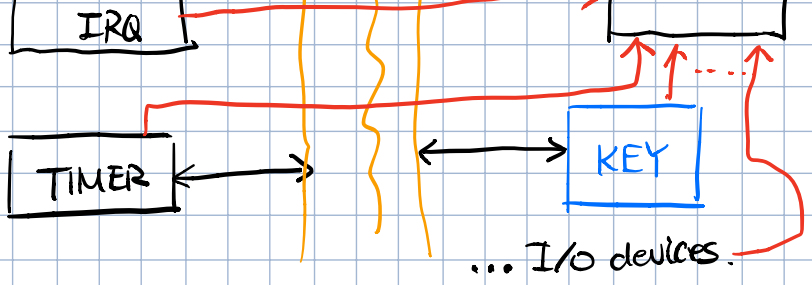
⊛ A hardware interrupt, or interrupt request (IRQ), is a signal sent by the I/O devices to the processor. The signal caused the processor to temporarily stop the execution of the current program and take an IRQ exception (load 0x18 in PC)

```
IRQ_HANDLER:  PUSH {R0 - R3, ....., LR}   // save registers used.
              :
              } code that checks the Interrupt Id
                                  ↙ KEYs port.
"If the interrupt id = 73, then"
              BL   KEY_ISR
              :
              POP {R0 - R3, ....., LR}
              SUBS  PC, LR, #4   // return from exception to main program.
```

⊛ When SUBS is executed w/ PC as the destination, this instruction also copies SPSR into CPSR, thereby returning from the exception. This action restores the flags for the main program, and sets the mode bits back to SVC.

Generic Interrupt controller (GIC)

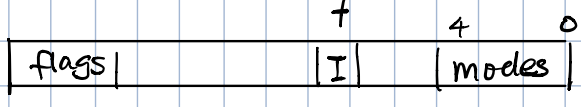




GIC: A memory mapped I/O device that receives IRQ signals from all other devices. The GIC drives the IRQ input of the processor. The GIC can be configured by your code to enable/disable any source of hardware interrupt. (all disabled by default.) The IRQ_HANDLER reads a register in the GIC to determine which I/O device to interrupt.

✱ Setting up interrupts.

CPSR:



"I" is called the interrupt mask bit. If set to 1 (by default), the interrupt registers (IRQ) are not visible to the processor. Your code has to set I=0 to use interrupts (using MSR)