Enabling Interrupt Generation for an I/O Device

- By default an I/O device won't produce an IRQ signal and sent to the GIC. You code has to enable interrupts in each I/O device.

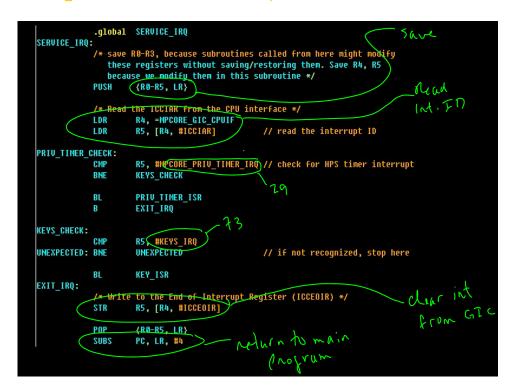
E.g. KEYs port:

Address						
0xFF200050						Data
58	unused	1	1	1	1	Interrupt mask 1
5C						Edgecapture 2

- (1) Your program stores a 1 for each KEY that should cause an interrupt
- 2 Your KEY_ISR code will read the Edgecapture register to see which KEY was pressed. You code has to clear the bits(s) in Edgecapture (or else the same interrupt will occur repeatedly)

Summary: steps required to use Interrupts

- 0. Provide the Exception Vector Table (at least 0x18)
- 1. Initialize banked SP for IRQ mode
- 2. (Initialize SP for SVC mode, like before)
- 3. Configure the GIC to enable interrupts (code supplied)
- 4. Enable interrupt generation in (each) I/O device
- 5. Set I=0 in CPSR
- 6. Provide IRQ_HANDLER code, which queries the GIC to determine the source of the interrupt. It then calls the subroutine for the I/O device (e.g. KEY_ISR)
- 7. Provide code for the interrupt service routine for each I/O device (e.g. KEY_ISR)
- 8. The IRQ_HANDLER has to clear the interrupt from the GIC





ICCIAR: Who caused the interrupt? ICCEOIR: Clear interrupt

Example of an ISR:

```
LDR
LDR
STR
                                             R0, =0xFF200050
R1, [R0, #0xC]
R1, [R0, #0xC]
                                                                                    // base address of KEYs parallel port
// read edge capture register
// clear the interrupt
KEY_ISR:
                          TST
BEQ
  HK_KEY3:
                                             R1, #0b1000
END_KEY_ISR
                                                                                    // KEY 3 pressed?
                            LDR
LDR
EOR
STR
                                             R0, =0xFFFEC600
R1, [R0, #8x8]
R1, R1, #1
R1, [R0, #0x8]
                                                                                    // timer base address
// read timer control register STOPED?
// toggle the enable bit
// write to the timer control register
 END KEY ISR:
                            MOV
                                       PC. LR
                                                          x00 "E" bit
                                                          ( to stop/start timer)
```