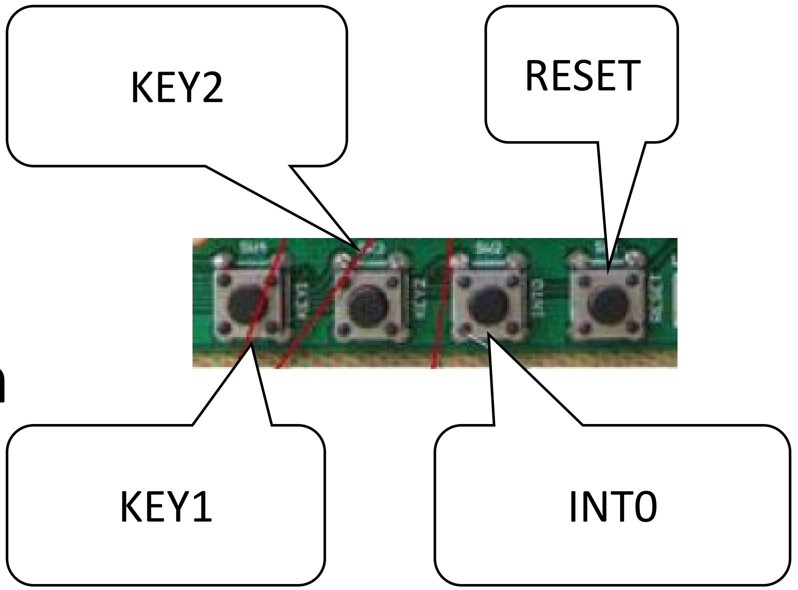
LPC1768 Exercises

Paolo Bernardi

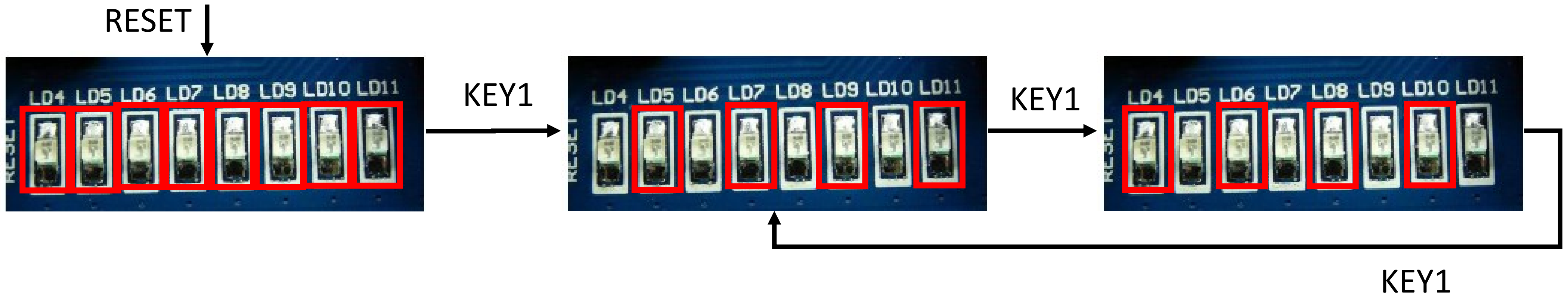
Requirements

- Install uVision 4.74 (available in the teaching portal /software)
- Download and use the 14b_sample_BUTTON_LED_NVIC_PCON project
    - Execute SW DEBUG

Altenate led lighting



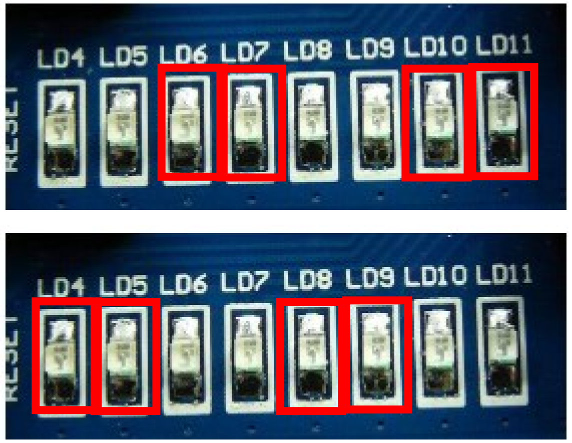
- Make the led assuming an alternate configuration
- RESET: all leds on
- KEY1:
  - when pressed setup alternance of on (odd) and off (even) leds
  - Any further (consecutive) pression produce a switch to alternance of off (odd) and (even) on leds



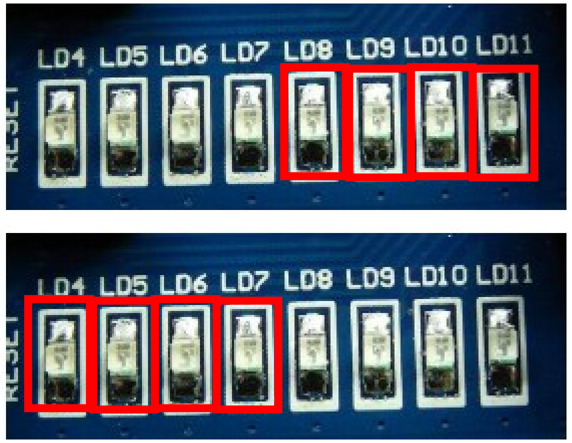
Challenge yourself

- Use Key2 and INT0 to add 2- and 4-leds alternances
- The system needs to switch from a configuration to another as soon as a button is pushed.

KEY2

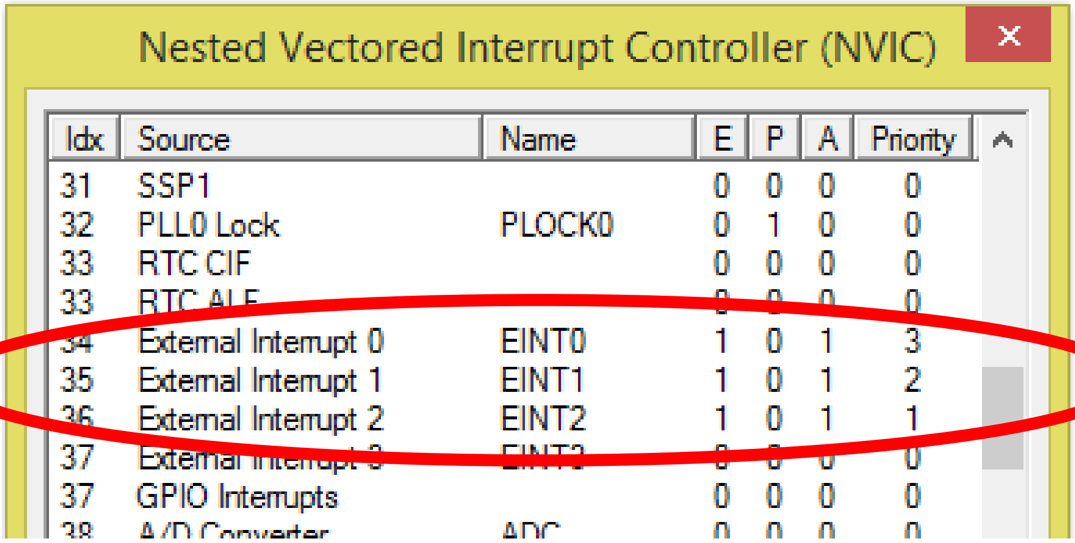


INT0

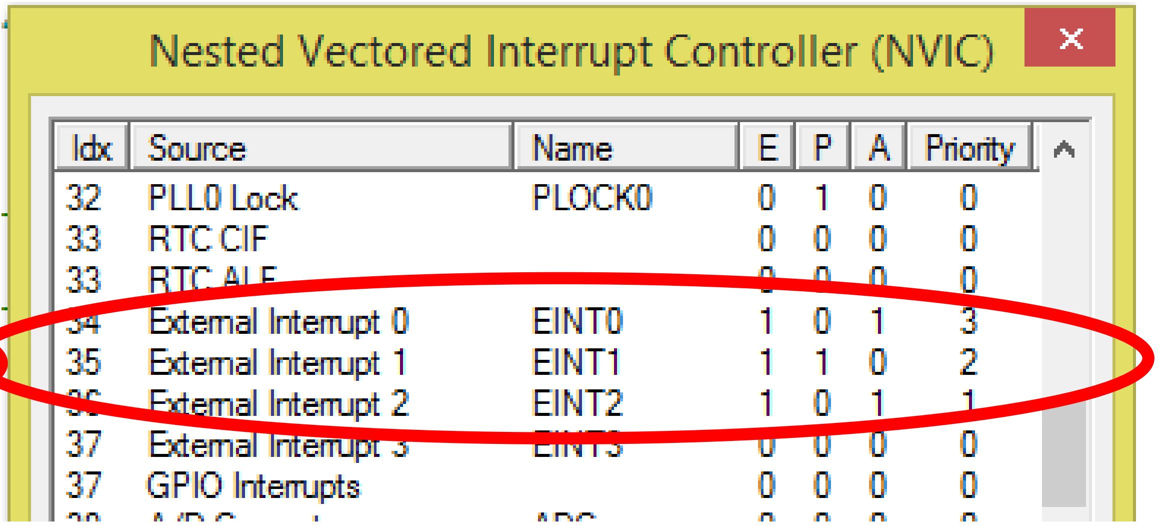


Experiment nested interruptions and priority

- Try to use first the sw and then the hw debugger to reproduce <u>with the help of breakpoints</u> the following Nested Vectored Interrupt Controller (NVIC – pg 73 UM) configurations.


Nested Vectored Interrupt Controller (NVIC)

Idx	Source	Name	E	P	A	Priority
31	SSP1		0	0	0	0
32	PLL0 Lock	PLOCK0	0	1	0	0
33	RTC CIF		0	0	0	0
33	RTC ALF		0	0	0	0
34	External Interrupt 0	EINT0	1	0	1	3
35	External Interrupt 1	EINT1	1	0	1	2
36	External Interrupt 2	EINT2	1	0	1	1
37	External Interrupt 3	EINT3	0	0	0	0
37	GPIO Interrupts		0	0	0	0
38	A/D Converter	ADC	0	0	0	0

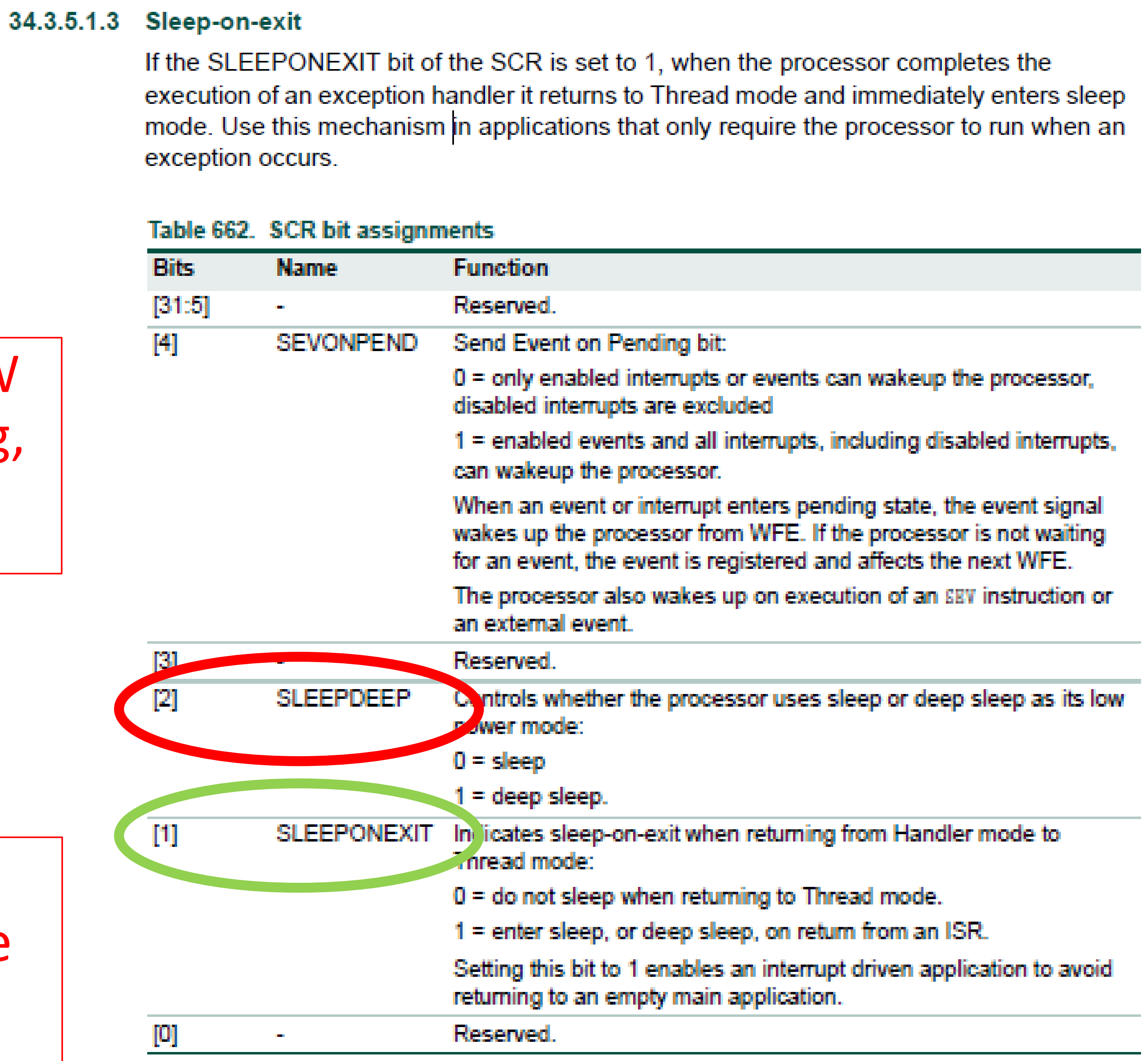

Nested Vectored Interrupt Controller (NVIC)

Idx	Source	Name	E	P	A	Priority
32	PLL0 Lock	PLOCK0	0	1	0	0
33	RTC CIF		0	0	0	0
33	RTC ALF		0	0	0	0
34	External Interrupt 0	EINT0	1	0	1	3
35	External Interrupt 1	EINT1	1	1	0	2
36	External Interrupt 2	EINT2	1	0	1	1
37	External Interrupt 3	EINT3	0	0	0	0
37	GPIO Interrupts		0	0	0	0

Experiment SLEEP-ON-EXIT

- WARNING: be sure your SW version is perfectly working, then try on hardware

- SLEEPDEEP mode could make the board unusable and MUST BE AVOIDED



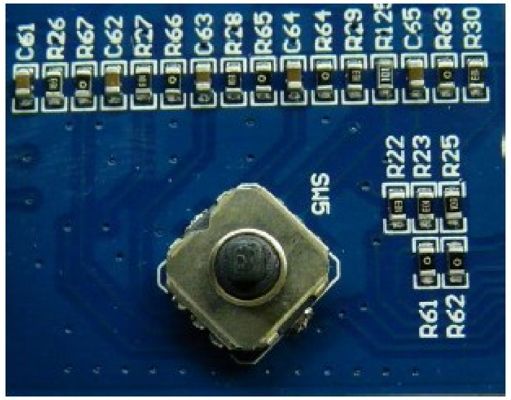
34.3.5.1.3 Sleep-on-exit

If the SLEEPONEXIT bit of the SCR is set to 1, when the processor completes the execution of an exception handler it returns to Thread mode and immediately enters sleep mode. Use this mechanism in applications that only require the processor to run when an exception occurs.

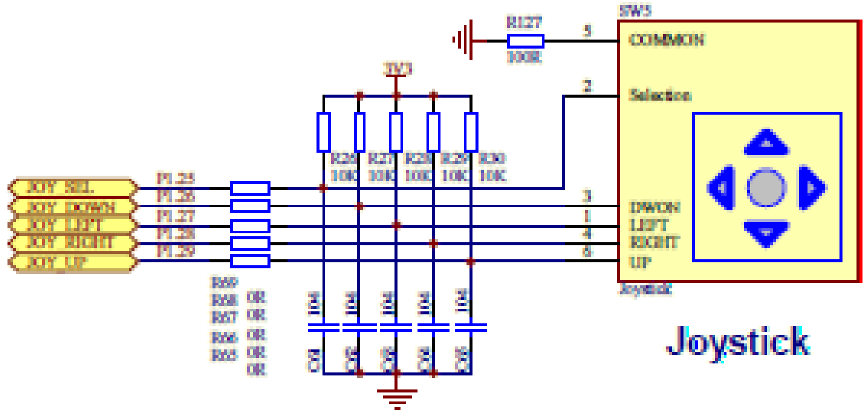
Table 662. SCR bit assignments

Bits	Name	Function
[31:5]	-	Reserved.
[4]	SEVONPEND	Send Event on Pending bit: 0 = only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded 1 = enabled events and all interrupts, including disabled interrupts, can wakeup the processor. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an `SEV` instruction or an external event.
[3]	-	Reserved.
[2]	SLEEPDEEP	Controls whether the processor uses sleep or deep sleep as its low power mode: 0 = sleep 1 = deep sleep.
[1]	SLEEPONEXIT	Indicates sleep-on-exit when returning from Handler mode to Thread mode: 0 = do not sleep when returning to Thread mode. 1 = enter sleep, or deep sleep, on return from an ISR. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	-	Reserved.

Build a joystick based project


Figuur 19 Joystick 5-Way Switch

- Setup the joystick for led control
  - Choose the behaviour you prefer in terms of led lighting as response to joystick movements (←↑
- Create a new project starting from the one used so far
  - Create joystick folder for libraries
  - Create function files (.h .c) (see 12_ at slide 7)



- The libraries of some peripheral core according to the following convention
  - *peripheral*.h /* prototypes */
  - lib_*peripheral*.c /* base functions */
  - IRQ_*peripheral*.c /* interrupt service routines */
  - funct_*peripheral*.c /* advanced user functions */