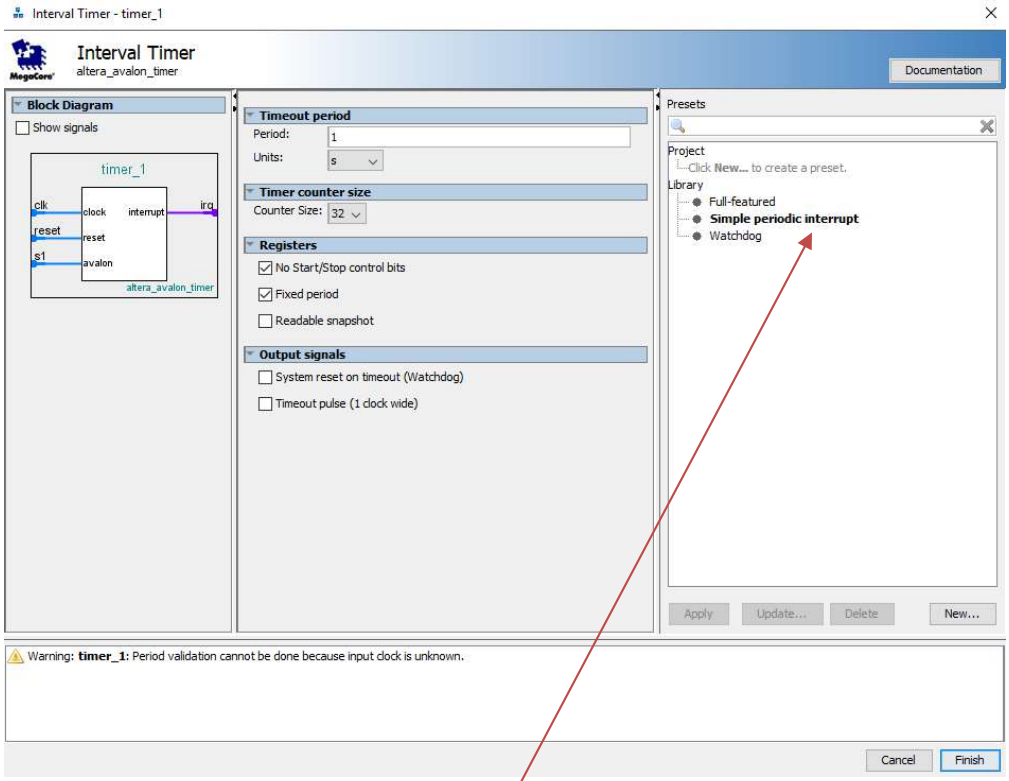
Education Objective
The educational objective of this demo is to become familiar with registering interrupts and writing interrupt service routines in C for the NIOS II processor.

Technical Objective
The technical objective of this laboratory is to design an embedded system for the Nios II processor and DE1-SoC that will update the count on the LEDs every one second. An interval timer core is used to generate an interrupt every one second.

1. Open Quartus II and create a new project. Name the project int_demo.
2. Open SOPC builder and name the system nios_system.
3. Add the following components

    - Nios II/e processor
    - 32K On-chip memory
    - 8-bit output PIO named leds
    - JTAG Uart
    - Sysid
4. Add Peripherals > Microcontroller Peripherals > Interval Timer.

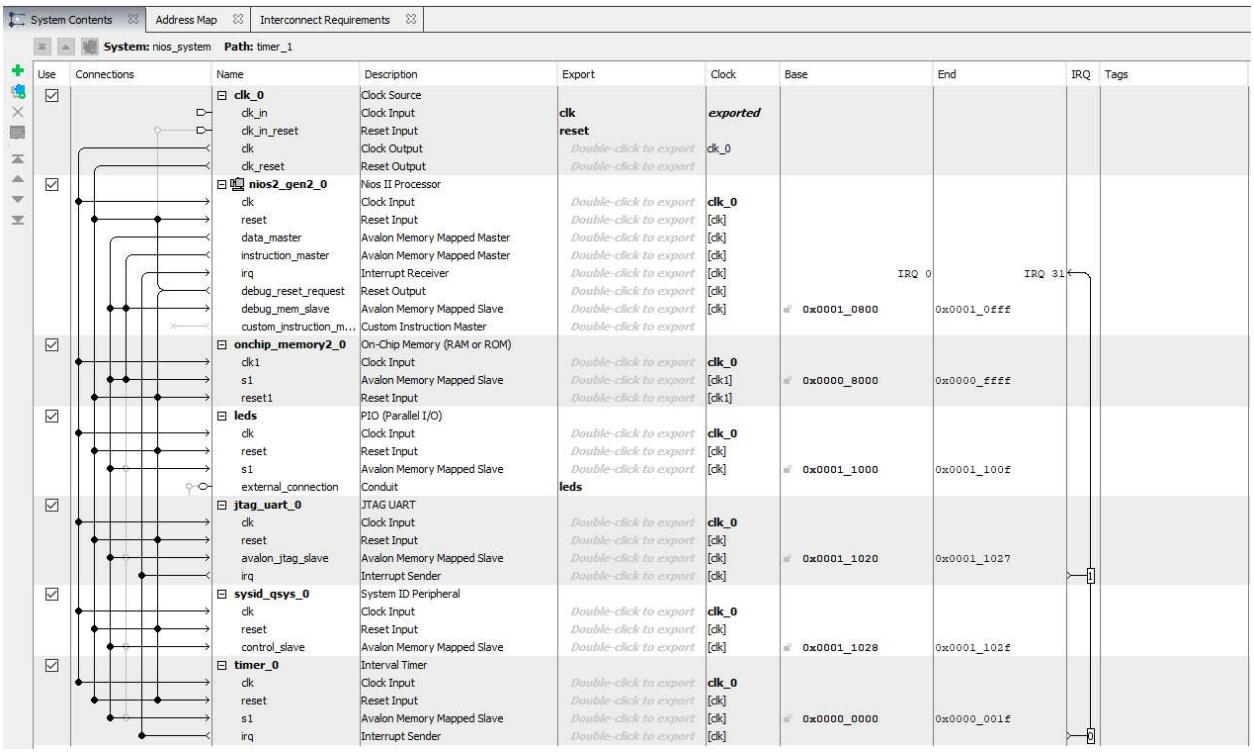


    - Double click on 'Simple periodic interrupt'.
      The simple periodic interrupt generator will act as its name implies and interrupt the processor at the rate specified by the timeout period. Edit the timeout period to 1 sec.

5. Edit the CPU to make the reset and exception vector within the onchip_memory

6. Choose System > Auto-Assign Base Addresses to ensure that none of the components overlap
7. Scroll to the right so that the column labeled IRQ is visible. Notice that the timer has been assigned interrupt number 1 and the JTAG UART is assigned interrupt 0. This means that the JTAG UART is the higher priority interrupt. To change the interrupt priority, click on the number and type a new number.
    - Switch the priority of these two interrupts

8. Your system should look like the following:



9. Generate the system. Remember to choose VHDL.
10. While the system is generating, download the design files from MyCourses. Move the int_demo.vhd file into your project directory. Open it in Quartus. Notice that the nios_system component only has 3 port signals. The periodic interrupt timer does not have any signals that go outside of the nios_system
11. Add the <project>/nios_system/synthesis/niod_sytem.qip file to the project
12. Import the pin assignments
13. Compile the design
14. Program the board
15. Open NIOS II Software Build Tools for Eclipse
16. Create a new NIOS II Application and bsp from template
17. Add int_demo.c to the application folder

18. Open the system.h file and find the timer constants. Notice that the TIMER_0_IRQ constant has a value of 0 which matches value we assigned in the SOPC Builder. Notice also that the timeout period constants matching what was specified when the component was added to the system.
19. Examine the C program. The first function is the interrupt service routine for the timer. The microprocessor will jump to this ISR once the interrupt is detected. The first thing that is done is to clear the interrupt. This is so the next interrupt can be caught. An ISR always has to clear the interrupt since a flag is set when the interrupt occurs and it needs to be cleared so that the processor can catch the next one.

    Go to the main program. The first instruction registers the interrupt so that the processor can accept it. This needs to be done for all interrupts in the system. In this example, the JTAG UART was also given an interrupt number in the SOPC Builder. However, no JTAG UART interrupts will be caught by the processor since it is not registered here.

20. Copy system.h from the bsp folder to the application folder
21. Right click on the bsp, choose NIOS II > Generate BSP
22. Right click on the bsp and build Project. Right click on the application and build Project

23. Choose debug as > NIOS II hardware
    - Set a breakpoint at the interrupt_handler
    - Click on resume
    - Step through the ISR until the LEDs change
    - Remove the breakpoint
    - Click resume

Video Submission:
Demo the breakpoint and the functional C code/LED changes. Explain what is happening in the video, or with an accompanying document.
Submit ONLY the video to the dropbox.