Assignment 3 Measurement in Zephyr RTOS (100 points)

Two important performance metrics of RTOS are interrupt latency and context switching overhead. Interrupt latency is the total delay between the interrupt signal being asserted and the start of the interrupt service routine execution. This delay may be extended if an interrupt arrives when the RTOS is in a non-preemptive critical region. As for context switching overhead, it is the delay of context switching process of saving the context of the executing thread, restoring the context of the new thread, and starting the execution of the new thread.

In this assignment, you are requested to develop an application to measure interrupt latency and context switching overhead of Zephyr RTOS (version 1.10.0) on Galileo Gen 2 board. To generate interrupts, you can loop back either gpio or pwm output signals to a gpio input pin which has interrupt enabled at rising or falling edge. To trigger a context switch, you can let a thread unlock a mutex for which a thread of higher priority is waiting. To record the instants of event occurrences, you can read x86's Time Stamp Counter (TSC) which provides a much better granularity than typical OS timer.

Your application should perform 3 measurements in sequence: interrupt latency without background computing, interrupt latency with background computing, and context switching overhead. The background computing you need to test is a message passing operation between two threads via a message queue. For each measurement, 500 samples should be collected and saved in separate buffers. A shell module should be included in your application which can print out the collected samples from any one of buffer after the measurements. An alternative design of the measurement application is to use shell command to initiate operations of starting one of the three measurements and printing any results in the buffer.

All your application files should reside in the directory "zephyr/samples/measure_*n*" of Zephyr source tree where *n* is your team number. There should be no changes to the original Zephyr source (except the patch applied to zephyr/driver/gpio/gpio_dw.c). This implies that all measurements should be done in main and/or application threads. In addition, a report that contains the histogram diagrams of the measured latencies and overheads should be compiled.

References

- Zephyr 1.10.0 source code: https://github.com/zephyrproject-rtos/zephyr
- Zephyr documentation: http://docs.zephyrproject.org/1.10.0/index.html
- Intel Quark processor and Galileo board documentation: https://www.dropbox.com/sh/xkftt2l3bv7csny/AADBKBHlEwL2LiHTn4jDadqqa?dl=0

Due Date

The due date is 11:59pm, March 30.

What to Turn in for Grading

- Compress the directory "zephyr/samples/measure_*n*" into a zip archive file named cse522-teamX-assgn03.zip. The report on pdf format should be included in the zip file. Note that any object code or temporary build files should not be included in the submission. Submit the zip archive to Blackboard by the due date and time.
- Please make sure that you comment the source files properly and the readme file includes a description about how to make and use your software. Don't forget to add each team member's name and ASU id in the readme file.

- There will be 20 points penalty per day if the submission is late. Note that submissions are time stamped by Blackboard. If you have multiple submissions, only the newest one will be graded. If needed, you can send an email to the instructor and TA to drop a submission.
- Your team must work on the assignment without any help from other teams and is responsible to the submission in Blackboard. No collaboration between teams is allowed, except the open discussion in the forum on Blackboard.
- Failure to follow these instructions may cause deduction of points.
- Here are few general rule for deductions:
    - No make file or compilation error -- 0 point for the part of the assignment.
    - Must have "–Wall" flag for compilation -- 5-point deduction for each warning.
    - 10-point deduction if no compilation or execution instruction in README file.
    - Source programs are not commented properly -- 10-20-point deduction.
- ASU Academic Integrity Policy (http://provost.asu.edu/academicintegrity), and FSE Honor Code (http://engineering.asu.edu/integrity) are strictly enforced and followed.