CS4271 Assignment 3

March 29, 2010

Submission Notes

- This assignment is due by 11:59 PM, Wednesday, 21-st April, 2010. No late submissions!
- This is an individual assignment. Acts of plagiarism are subjected to disciplinary action by the university. Please refer to `http://www.comp.nus.edu.sg/students/plagiarism/` for details on plagiarism and its associated penalties.
- *Submission Instructions*: (Failure to follow these instructions may result in deduction of marks) Submit one file named YourMatricNumber.pdf containing all your answers to the IVLE Workbin Folder HW3-Submission.

Question 1 [5 marks]

Deadline Monotonic (DM) priority ordering is a fixed priority ordering scheme where priorities assigned to processes are inversely proportional to the *length* of their deadline. Thus, the process with the shortest deadline is assigned the highest priority, the longest deadline process has the lowest priority. If there exists a schedule that meets all the deadlines with the above mentioned fixed priority scheme, then DM will produce a feasible schedule.

Suppose you are going to design part of an Unmanned Aerial Vehicle (UAV) application which involves an automated pilot as well as obtaining navigation instructions via radio. You want to run the following four periodic tasks on a single processor under Deadline Monotone (DM) scheduling. Note that the deadlines of the tasks are different from their periods.

Table 1: Description of UAV tasks

Task name	Description	Period (ms)	Deadline (ms)
ap_spi_cmd1	executes command received through SPI serial link	88.34	68.64
fbw_radio_mancmd	manages radio command orders	180.16	158.72
ap_main_getcmd	stabilization task at autopilot	128.84	32.16
fbw_servo_nav	manages servo commands	322.18	258.78

Assume you have three systems S_1, S_2 and S_3 with different choices of processors and main memory. The particulars of these three systems are listed in Table 2.

Assume the cost of these systems are $S_1 > S_3 > S_2$. Use the following configuration for all other parameters: *pipeline:* out-of-order, *superscalarity:* 1, *IF queue size:* 4, *RB size:* 8, *cache:* direct-mapped, *Number of cache sets:* 32, *cache block size:* 8, *branch predictor:* disabled (perfect prediction).

Which of the systems (if any) you would like to use (the cheapest one that makes all the tasks schedulable under DM)? Write down the *step-by-step* details of how your decision was made.

Table 2: Description of available systems

System	Processor frequency (MHz)	Main memory latency (CPU cycles)
S_1	500	30
S_2	500	50
S_3	400	30

Question 2 [2.5 + 2.5 = 5 marks]

Analyze the given program q2.c in the folder question2 with the following configuration: *pipeline:* out-of-order, *superscalarity:* 1, *IF queue size:* 4, *RB size:* 8, *cache:* direct-mapped, *Number of cache sets:* 4, *cache block size:* 8, *Main memory latency:* 30 cycles, *branch predictor:* disabled (perfect prediction). Now, answer the following questions:

1. Report both the Worst Case Execution Time (WCET) and Simulation Time in CPU cycles and compute the over-estimation ratio. Find out the reason of over-estimation. State your findings clearly in the report (Your reasoning must be explained in the context of this particular program).

2. Can you re-write the source program (q2.c) so that the over-estimation is reduced ? State clearly in your report why your changes reduce the over-estimation ratio compared to the original version of the program and also include the modified source code in the report (Note that just providing the modified source without any explanation is not going to give you any marks). You will get more points if you can achieve substantial reduction in over-estimation ratio with minor changes. Is the simulation result also improved with your transformation ? Justify why or why not.

About Chronos

Chronos (`http://www.comp.nus.edu.sg/~rpembed/chronos`) performs timing analysis of embedded software (written in C) through static analysis. In particular, Chronos estimates Worst Case Execution Time (WCET), which is the upper bound on the execution time of a program over all possible data inputs on a specific hardware platform.

To use the pre-installed Chronos on "tembusu" cluster for the purpose of this assignment, you can follow the steps below:

1. Launch ssh secure client (installed in every machine of SOC). Go to "Edit menu" and go to "settings". Goto "Profile settings->Connection->Tunneling". Check the box "Tunnel X11 connections". Goto "File menu" and click "Save settings". For a graphic help please refer to `https://www.comp.nus.edu.sg/cf/x/xwin32-sshsecureshell.html`. You need to give your SOC unix id and password to check this help.

2. Log in `sunfire.comp.nus.edu.sg` with a SSH Secure Shell Client using your SOC account and password.

3. Log in `tembusu.comp.nus.edu.sg` from sunfire using the command "ssh -X tembusu".

4. Check for a ".bashrc" (note the ".") file in your home directory. If available, edit the file by putting the following two lines at the end of the file:

   ```
   export PATH=$PATH:/home/s/sudiptac/TA-4271/CHRONOS_HOME/bin
   alias chronos="cd /home/s/sudiptac/TA-4271/CHRONOS_HOME/gui/; ./gui.sh"
   ```

   If the file is not in your home directory, then create one with that name (i.e. ".bashrc") by putting the above two lines in ".bashrc" file. Save and close the file ".bashrc".

5. Type the command "source ∼/.bashrc" (Note the "." again).
6. Open a X11 connection to support the Chronos GUI by using X-Win32 / Xming. "Xming" is already installed in Embedded Systems Teaching Labs. If you want to connect from outside NUS, you can download the free version of Xming at `http://sourceforge.net/projects/xming`. After downloading, you can install and launch the "xming" application with the *default* settings.
7. Download the "benchmarks.zip" from IVLE. Unzip it ("unzip benchmarks.zip") into a folder under your tembusu account. It should contain five sub-folders (*ap_spi_cmd1*, *fbw_radio_mancmd*, *ap_main_getcmd*, *fbw_servo_nav*, *question2*) under a floder named "benchmarks".
8. Type the following command in your SSH to launch Chronos GUI: "chronos". Note that all the necessary settings for this copy of Chronos have already been done. Please do not change them. Please also make a note that you have to use *only this installation of Chronos* to do Homework 3.
9. To analyze/simulate a particular benchmark, you should do the following:
   (a) Open a benchmark by choosing to open the folder containing the benchmark. One folder should contain exactly one benchmark (one main method).
   (b) After the source code, CFG, and assembly code are displayed on Chronos GUI, you can set the loop bounds using Option / Loop bound constraints. If no loop bound constraints are required, it means all the loop bounds have already been automatically set by the Chronos data flow analysis.
   (c) Before analyzing/simulating the benchmark, you need to set the processor architecture on which the program runs in Run / Processor configuration.
   (d) After all these have been done, you can estimate the programs WCET or simulate it by clicking Run / Estimate or Run / Simulate respectively. The result will be shown in the bottom panel. For the four benchmarks used in Question 1, you do not need simulation on them. However, for Question 2, you need both estimation and simulation.
10. In your next login to tembusu, you only need to type "source ∼/.bashrc" command before launching "chronos".

END OF HOMEWORK 3