ELECTRICAL ENGINEERING PROGRAM
California Polytechnic State University

CPE 439 Final Project

Objectives

- To implement a fairly complex RTOS project of your choosing –or –
- To develop a new CPE 439 course laboratory assignment that is more interesting than blinking LEDs

Introduction and Overview

Our focus this quarter with the CPE 439 experiments has been to learn and apply the basic objects and services provided by the FreeRTOS in an actual embedded system. While the experiments themselves were nothing to write home about, they arguably presented an introduction to developing an RTOS-based embedded system.

The final project provides you with an opportunity to apply an RTOS to a more complex system or to develop a CPE 439 course lab assignment that is more interesting than blinking LEDs. Note that you will have 3 weeks to work on this assignment.

Assignment:

Option 1:

- Develop your own RTOS system

Develop a new application or port an existing embedded system application to an RTOS-based system. The RTOS and associated hardware can be one of your choosing or you can simply use the AVR-based SDK500 board. The project should be meaningful and reasonably complex. At the very least, it should stretch you current knowledge of RTOSes and RTOS-based systems. You must interface to at least one peripheral through a protocol you have not yet used in this course. A quick online search suggested these applications that use an RTOS:

- MIDI Recorder
- Memory dump/viewer
- Waveform/wavetable performer
- Some sort of closed loop feedback control system

Option 2:

- Develop a new course laboratory assignment

Develop a new course laboratory assignment that is more interesting than blinking LEDs. Some ideas I have are to :

- Write a device driver for a peripheral device (such as an SD-Card Reader, ethernet controller, USB Controller, Keyboard/keypad controller, any other peripheral) and design a fun lab that makes use of the peripheral device in an RTOS context

- Design a lab that highlights the benefits of an RTOS. Implement the design in a non-RTOS and RTOS way and measure response times of critical events to quantify the responsiveness of an

RTOS and non-RTOS system. Include at least one peripheral interface.

Deliverables:

- A one-page project proposal: The proposal should describe what you propose to do for your final project. Include the following information:
  Project Title
  Parts List (what development board? What RTOS? What peripheral(s)? what cables?)
  Project Description – what exactly do you intend to do (include at least one graphic that can help depict your overall design)

- A one-page progress report: This progress report will be due 1.5 weeks into the project to update the instructor on the progress of your project.

- A presentation of your project: Your final presentation will take place during class during dead week. The presentation should be 5 minutes and follow the "Ignite" presentation format. http://en.wikipedia.org/wiki/Ignite_%28event%29 Each person should speak for an equal amount of time. The presentation should present what you did for your project in a factual and fun way.

- A final project report: This report should describe your project in detail, with emphasis on the various RTOS-based aspects of the project. The "report" for Option 2 (lab assignment) should be the written lab handout that will be given to the students that includes the objectives of the experiment, a description of what RTOS objects and services are explored, the procedure of the experiment, and the expected deliverables.

- Zipped Directory of your code: You should turn in all the code that you would need in order to modify and rebuild your project (that is, turn in your whole project directory). As usual, any code that you write should follow proper embedded C coding standards that we have practiced all quarter. For Option 2 (lab assignment), include a subfolder that has all the files that would need to be posted on PolyLearn for a student to be able to do the assignment.

Past Projects:

RTOS Bop It	Took apart a BopIt and implemented the game with an Arduino running DuinOS
Logic Analyzer on AVR	Created an 8 channel logic analyzer that can run on an Arduino running FreeRTOS
Real-Time Image Capture	Used a BeagleBoard running FreeRTOS to capture an image from a camera module when a hall effect sensor reads a voltage above a threshold simulating a possible manufacturing line scenario
Quantum Space Piano	Using FreeRTOS on STK500. Points a camera to an LED strip, and reads which LEDs are blocked to play different musical notes
Wacky Waveable Noise Maker	Maintains the position of the 'plug' from accelerometer data and leverages the use of RTOS to synchronize incoming data with outputting the appropriate sound
LaseRTOS	Simple Laser Tag game
Last Minute Hope	Sends a text message containing GPS coordinates when accelerometer senses large tremor
Vacuum Fluorescent Display	Designed a 9V to 60V boost converter controlled by the ATMega running FreeRTOS
RTOS Arcade	Uses FreeRTOS to implement game logic and interface to a display
Angry Alarm Clock	Interfaces a SpeakJet chip with the ATMega to speak when alarm goes off
Keyboard Interface	Makes a lab with SPI, UART, and PS2 interfaces