REPORT FOR 2015 ECETDHA MINI-GRANT

DEVELOPMENT OF A REAL-TIME OPERATING SYSTEMS EDUCATION PLATFORM

Dr. Gang Sun
Department of Engineering technology & Industrial Distribution
Texas A&M University

A real-time operating system (RTOS) is an operating system that guarantees a certain capability within a specified time constraint, which is the key element for embedded product/system development, such as cell phones, smart TVs, and medical devices, etc.

Due to the lack of time and lab facilities, most U.S. Engineering Technology programs do not offer any courses related to real-time operating systems or they just emphasize concepts of RTOS and introduce basic theoretic topics (e.g., various software architectures, real-time multi-task scheduling strategies, and RTOS validation techniques, etc.) in their microcontroller courses. As a result, many students and product development engineers who have a good understanding of theory and concepts of RTOS do not have the confidence to map their knowledge onto implementations. To bridge the gap between conceptual understanding and concrete implementations, an RTOS platform has been established for students at Texas A&M University.

1. Conceptual Design

The RTOS platform mainly consist of two parts: the Modular Integrated Stackable Layer (MISL) intelligent layer and the analog system environment (ASE) board. Fig.1 illustrates the conceptual block diagram of the platform with many of the implemented features.

Fig.1. Conceptual block diagram of the RTOS platform.

The MISL intelligent layer, typically the TI-MSP430, can be directly interfaced to the ASE board. To integrate new and relevant devices and technologies into current embedded system education, the ASE board encompasses various analog and digital peripherals. Included in this suite of capabilities are GPIO outputs/inputs, LEDs, 7-segment displays, audio system, buzzer, switches, matrix keypad, and TFT LCD with touch screen, typical signal conditioning circuits such as A/D and D/A conversion for analog voltage simulation, battery life density measurement, 3-axis accelerometer, high-resolution external ADC converter, multiple analog signal generators, and motor control, etc. Several communication interfaces and protocols are also available such as UART (USB, RS-232/485, Bluetooth, and Zigbee), SPI (Ethernet, 2.4 G Wi-Fi, Micro SD card, and flash memory), I^2C (DAC and EEPROM), and 1-wire communication devices. Furthermore, the robust design of the ASE board supports interfacing to a number of other embedded intelligence boards such as the TI Launchpad development system. Students can program on-board features with other microcontrollers through header pins and jumpers. A fully populated ASE board with the MSP 430 layer installed is shown in Fig.2.

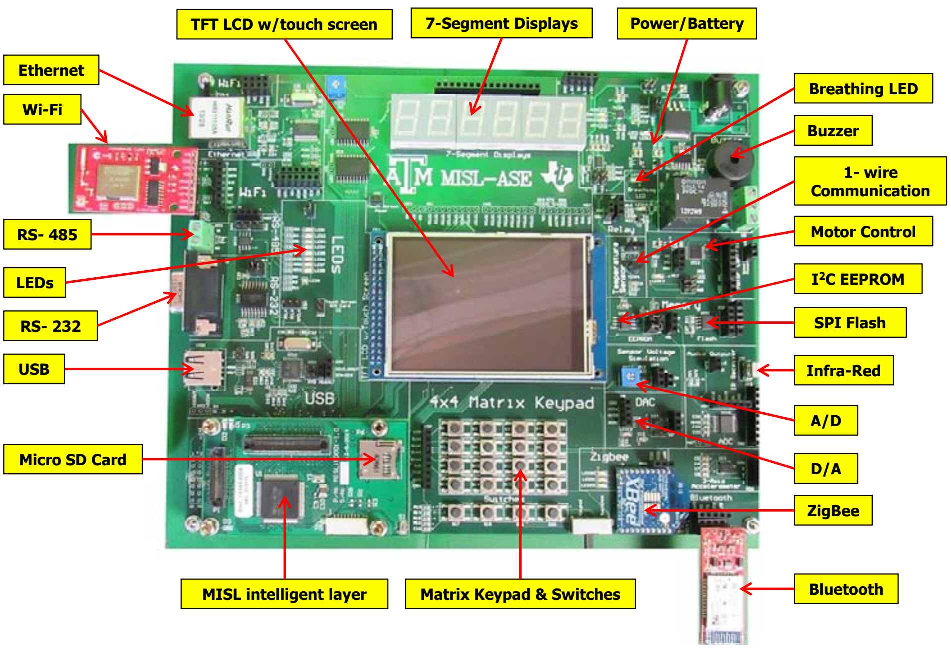

Fig.2. MISL-ASE platform with MSP-430 microcontroller.

2. Development of the RTOS Platform Based on MISI-ASE Board

The MISL-ASE platform incorporates many features that directly interface to the MSIL-MSP430 layer. The platform has four major parts: 1) MISL architecture – TI-MSP430 intelligent board; 2) GPIOs; 3) Signal conversion; and 4) Wired & wireless communications.

2.1 GPIO inputs and outputs

GPIOs are one of the most valuable resources in embedded systems providing the ability to communicate with the external world through digital lines that configured to operate as either inputs or outputs.

(1) GPIO-outputs: when configured as outputs, GPIOs can control the binary status of external devices by setting an external voltage level to either a logic high or logical low. The typical external devices are LEDs, buzzers, LCDs, and relays. GPIOs have been implemented on most development boards (e.g., TI lanchpads). The MISL-ASE platform not only includes those devices, but includes three unique features.

a) Breathing LED (PWM): The Breathing LED is a feature that simulates a breathing response by fading the LED intensity in and out. It is controlled by using the PWM duty cycle function of timers and GPIOs.

b) Mutilpe 7-segment displays: The MISL-ASE platform incorporates 6 common cathode 7-segment displays that are controlled by 2 octal D-type latches (74HC573). One latch is used to select which display to use and the other latch is used to select what to display. The latches are controlled through the latch enable (LE) pins to prevent the latches from overriding each other. This noval display design only uses one I/O port (8-bit I/O pins) to control multiple displays instead of using multiple GPIO ports.

(c) TFT LCDs with touch screen: TFT (Thin-film transistor) LCDs with touch screen are widely used in mobile phones, handheld video game systems, personal digital assistants, navigation systems projectors, etc. On the MISL-ASE platform, a 3.2" TFT LCD Display module is employed, which includes the display controller (ILI9341), the touch IC (XPT2046), and a SD card. The MISL- MSP430 layer uses 20 GPIO pins and a SPI interface to communicate with this TFT LCD display.

(2) GPIO-inputs: Mechanical contact switches are one of the components most commonly interfaced to microcontrollers as a typical GPIO input example. There are two GPIO input features on the MISL-ASE platform: the 4x4 matrix keypad and the 4 switches. The keypad uses one of the GPIO ports (8 pins) of the microcontroller, where 4 pins are for the columns and 4 pins for the rows. The column pins are shared with the 4 switches. A 4PDT switch is used to switch between the two features.

2.2 Signal Conversion

One common task for embedded systems is to measure analog signals from the outside world, e.g., pressure, temperature, humidity, flow rate, etc. On the ASE board, the analog-to-digital conversion part includes several typical analog devices interfaced to the MSP430 microcontroller: 1) Potentiometer: it is used to simulate a varying analog input voltage; 2) 3-axis accelerometer (ADXL335): The X, Y, and Z channel outputs of the ADXL335 are connected to three of the 12-bit internal A/D converter inputs of the microcontroller to detect the static acceleration of gravity in tilt-sensing applications, and dynamic acceleration resulting from vibration, shock and motion; and 3) 16-bit, 8-channel SAR external ADC (ADS8345): it is also available so students can gain experience with a high-resolution ADCs instead of the 12-bit internal ADCs.

2.3 Wired and wireless communications

Most popular wired & wireless communication interfaces and protocols are available on the MISL-ASE platform. These include: 1) four-wire communication networks: SPI (Ethernet, 2.4 G Wi-Fi, Micro SD card, and flash memory); 2) two-wire communication networks: UART (USB, RS-232/485, Bluetooth, and ZigBee); and 3) two-wire communication networks: I2C (DAC and EEPROM).

(1) SPI: the Serial Peripheral Interface (SPI) is a synchronous serial bus standard with full-duplex capability to support communications between a master (e.g., microcontroller) and one or multiple slave peripheral devices. On the ASE board, the Ethernet, 2.4G Wi-Fi modules (TI-CC3000 and nRF24L01), the Micro SD card, and the flash memory (SST25VF016B) are able to communicate with the MISL-MSP430 stack via SPI interfaces. For example, the interface between the MSP430 microcontroller and the Ethernet controller chip (ENC28J60) is based on the SPI bus protocol. The SI, SO, SLCK, and CS pins of the Ethernet chip are connected to SPI pins (MOSI, MISO, SCLK, and CS) of the microcontroller. With this Ethernet feature, many different devices and equipment can be remotely accessed, monitored, and controlled through internet, such as web-based home automation, remote environmental monitoring, Voice Over IP, etc.

(2) UART: The Universal Asynchronous Receiver / Transmitter (UART), is a very useful feature of microcontrollers for communicating serial data (text, numbers, etc.) to computers or devices. The MISL-ASE platform not only contains typical UART standards (RS232, RS485, and USB), but provides new wireless communication standards (ZigBee and Bluetooth) via UARTs. ZigBee is a reliable, low-cost and low-power, wireless networks based upon the 2.4 GHz IEEE 802.15.4 radio protocol. The ZigBee module (XBee2) is used on the platform and allows students to create wireless distributed sensor networks to remotely monitor and control electromechanical systems in their home, office, or other areas. The RN-42 is a Class 2 certified Bluetooth module, which can be connected to the ASE board to communicate with personal electronic devices, e.g., cell phones, wearable devices, etc.

(3) I^2C: The Inter-Integrated Circuit bus (I^2C) is a synchronous serial communication protocol to support on-board interconnection of integrated circuit devices. On the platform, the I^2C uses two lines (SDA and SCL) to establish a half-duplex communication bus between the master (MSP430) and slave devices:16K EEPROM (24C16) and Digital-to-Analog converter (DAC5574).

3. RTOS Course

ESET 489: Embedded Real-Time Operating Systems
Spring 2016

Instructor: Dr. Gang Sun
Office: Fermier 305
Phone: (979) 845-0675
Email: gangsun@tamu.edu
Office Hours: by appointment

This syllabus is a tentative course schedule. The policies and dates presented here are subject to change at the discretion of the instructor.

Lecture: MW: 9:10 – 10:00 am, CHEM 255
Laboratory: Thompson 101A F 3:50-6:30 pm (sec. 501), T 1:50-4:30 pm (sec. 502)
W 5:00-7:40 pm (sec. 503)

Course Learning Outcomes:

1. Demonstrate knowledge of the vocabulary of Embedded Kernel design.
2. Design and implement an embedded application with at least 5 tasks1.
3. Demonstrate an understanding of sharing memory, dynamic allocation/de-allocation, and stack usage in an embedded design.
4. Demonstrate ability to use binary semaphores, counting semaphores and messages queues.
5. Demonstrate ability to design an embedded application using diagrams and text descriptions.
6. Demonstrate knowledge of defensive design techniques including error handling, time outs, and watch dog timers.

Prerequisites: The course prerequisites are explicit. The prerequisites include successful completion of ESET 369 and ENTC 350 with a grade of C or better. These prerequisites are designed to ensure that the student has a good foundation in digital logic, electrical circuits and the C Programming Language.

Textbook: Simon, D. E., An Embedded Software Primer, Addison-Wesley, Boston, 1999.
μC/OS-III: The Real-Time Kernel, Jean J. Labrosse, Micrium Press, 2010.

Reference: C Primer Plus, Sixth Edition, Stephen Prata, 2014.
Real-Time Systems, Jane W. S. Liu, Prentice Hall, 2000

Student Work: At the start of the semester we assume the concept of a level playing field. That is all students have the same resource of time to complete the studies/work required for successful completion of the course.

With respect to the time-frame for course work it begins on the first day of class and ends when the final examination is over. Each student has the same set of resources available to achieve the goals and objectives of the course. The challenge is to apply oneself to achieve the goals and objectives in a timely manner.

Student work includes lecture and laboratory attendance, reading and study, laboratory work, programming, problems solving, etc. The competition starts now and each student's efforts will be evaluated.

As the result of the level playing field all work is assigned to all students at the same time and all work is due at the same time.

Laboratory requirements: All laboratory work must be completed to meet the minimum requirements for a passing grade in the course. Laboratory work may include exercises, assignments and a final project.

Each student is required to maintain an engineering journal. In this journal you will record the progress of your work, questions, references, contact information, web addresses, etc. Each student is expected to prepare for laboratory work. When you work with the professor you may be asked to review your journal entries.

In preparation for laboratory work sessions each student should read related reference material, conduct appropriate research and work related problems.

Attendance: Regular lecture and laboratory attendance is required. Attendance during all regularly scheduled examinations is required. Students who are absent from a lecture or examination must submit medical or other appropriate documentation in order to classify the absence as excused. A student who is absent from an examination without submitting medical documentation or other appropriate documentation that substantiates an emergency will receive a zero for the examination grade and there will be no make-up examination.

Each student is responsible for all material presented during regularly scheduled lecture and laboratory sessions.

Grades: Your course grade will be based on the following points:

Middle-term Exam:	25%
Final Exam	35%
Laboratory	20%
Project:	15%
Quizzes & Participation:	5%
Total:	100%

A: 90%-100%, B: 80%-89%, C: 70%-79%, D: 60%-69%, F: <60%, incomplete Lab, or an F score in Lab.

One middle-term exam and a final exam will be given. Since the course content builds on prior material, all exams are to some extent cumulative. No make-up exams will be given unless circumstances are exceptional. The term project is your opportunity to demonstrate your skills learned in this course. The term project can be done in groups of up to two students. At the end of the semester, students will be asked to evaluate their team members' contribution to the project which will be included in the grading.

Tentative Schedule of Topics:

1. **Introduction**

- Application Structures; Single Thread, Foreground/Background, Real-Time Kernel
- Kernel/RTOS vocabulary
- Time Slice Concepts

2. **Key Embedded Concepts**

- Interrupts
- Time Dependence
- Understanding and preventing race conditions
- Tasks as standalone programs
- Task/Program Contexts
- Task Switching Pending

3. **uC/OS-III**

- Installation and use of uC/OS-III development environment
- Creation of a single task

4. **Timer Operation**

- One-Shot Timers
- Periodic Timers
- Time of Day

5. **Inter-task Communications**

- Kernel Communication Diagrams
- Binary Semaphores
- Counting Semaphores
- Mail Boxes
- Message Queues
- Priority Inversion
- Deadlocks
- Mutexes

6. **Design Principles**

- Timing
- Priorities
- Defensive Programming