Embedded systems – Exercise #3

1 Exercise purpose

This exercise introduces the use of an embedded operating system and its abstractions–threads, synchronization objects (mutexes, semaphores, queues and event flags) and memory pools.

Also, you will use the drivers you have written in the previous exercise in addition to a few new ones, and implement a device which sends and receives SMS message (Short Message Service), just like any modern cellular device.

2 System Description

The system you are going to build in this exercise will be composed of the following components:

1. ARC controller.
2. ROM for storing the firmware
3. RAM for storing data.
4. Timer.
5. [Input] An input panel with 12 keys.
6. [Output] A small screen for displaying text.
7. [Input, Output] A network interface.

The device has **32KB** of ROM starting at 0x0 and **64KB** of RAM starting at 0x80000.

3 Using the ThreadX OS

As you've seen in the lesson, ThreadX is an embedded OS, offering a thread-scheduler and many operating system primitives, such as synchronization primitives, synchronized data structures and memory management tools. You can find the full ThreadX manual and reference at the course website.

You should wisely use threads for the different tasks the device has to perform. Use the ThreadX operating system to utilize these threads. Use the other elements offered by ThreadX to allocate memory and synchronize your threads (whenever needed).

Following is an example of the main threads used in some hypothetical design (not necessarily optimal or even recommended):

- **Input Handler thread** – This thread is responsible for all of the input of the device – either via network, from input panel (buttons), UART or any other input. This thread must process each request serially.

- **Network Receive thread** – This thread will receive all of the packets from the network device and will handle them.
- **UI Thread** – This thread will be responsible for displaying the UI and updating it according to the events in the system (keystrokes, time passes by, incoming messages etc). The UI specification (how should it look and behave) can be found below in section 5 (User Interface Specification).

4 SMS network

You will use the network device driver to simulate a connection of your dev ice to an SMSC (Short Message Service Center).

The full protocol description, along with samples, can be found in the **SMS Protocol Specification** document on the course website.

As described in the protocol specification, your device must send periodic SMS_PROBE messages via the network card (at least every 100 milliseconds), and must be able to parse incoming responses (SMS_PROBE_ACK or SMS_DELIVER messages) to support **incoming** messages. **Note**, that incoming messages should be saved to the memory for later display (see section 5.2(a) below).

Also, in order to support **outgoing** messages, you device should be able to send SMS_SUBMIT when it is requested to, and parse the incoming response (SMS_SUBMIT_ACK).

You can access the SMSC directly from the web at http://embsys.comoj.com/SMSC.php where you can send SMS messages, see the pending messages and the devices that are "alive" in the network.

Note: You are given a pre-compiled library (embsys_sms_protocol.o) you can use to manipulate the various data structures required by the SMS protocol. Writing your own implementation to replace this functionality will be credited with a 10 points bonus. It is recommended to start with the given library, and only start the bonus if everything works and you have the time...

5 User interface specification

You have to implement a driver for a “monitor”, which can display strings on a device similar to a cellular phone.
The UI is conducted of four screens: Message listing, Message display, Message edit and Message number screens.

5.1 Special keys indicators

In each screen, the asterisk (*) and pound (#) keys might have a special usage.

The bottom row of the screen will indicate to the user the title of the special buttons (*, #), left and right aligned appropriately. The **whole** bottom row will be "selected".

5.2 Message listing screen

a. The device shows the list of incoming and outgoing text messages. Those text messages should be saved in memory when they are received by the device (for simplicity, a message is considered 'sent' if the user have sent it, regardless of whether the network device actually sent it).

b. By default the first message is selected. Your device needs to be able to store 100 messages.

c. Each line in the display shows a serial number of the message (between 0 and 99), the number it was received from and a single character indication whether the message is an incoming message (`I`) or outgoing message (`O`).

d. If a message arrives while the device is displaying this screen, it is expected that the list will be refreshed on the screen to indicate that a new message has arrived (add entry to the list)

e. When this screen is active the buttons "2" and "8" serve as "up" and "down" arrows, respectively.

f. The user can navigate using the "up" and "down" between the messages. When moving to a certain message the display shows it as "selected". The next message after the last one is the first message.

g. When selecting a message and pressing "OK", the message is opened and the display shows the ***message display*** screen.

h. The button "*" serves as "new message" (title: "New"). Pressing it will show the ***message edit*** screen.

i. The button "#" serves as "delete" button (title: "Delete"). Pressing it will delete the message, with the next message being selected.

5.3 Message display screen

a. The device shows the source of the message in the first line, which should be "selected", the time it was received in the second line, which should also be "selected", and the text of the message starting from the third line.

b. The text message is limited to 160 characters. If the full text doesn't fit the screen the user will be able to scroll up and down with the "2" and "8" buttons. In that case a ">" character will be displayed in the middle of the bottom line if the user can scroll down, and a "<" if the user can scroll up.

c. When this screen is active the button “*” serves as a cancel button (title: "Back"). Pressing it will return to the ***message listing*** screen, with the current message being selected.

d. The button “#” serves as “delete” button (title: "Delete"). Pressing it will delete the message, return to the ***message listing*** screen, with the next message being selected.

5.4 Message edit screen

a. The device shows a form that allows entering the message text.

b. When this screen is active the number buttons serve to enter text:

   i. Each button has an associated list of characters it can enter

   ii. When pressing the button for the first time it shows the first character in the list

   iii. If pressing the button again within half a second, it will show the next character on the list.

   iv. If the same button was not pressed within half a second, or another button was pressed, the cursor will move to the next location.

   v. The characters associated with each button are:

      1 - “.” (period), “,” (comma),”?” (question mark), “1”

      2 - “a”, “b”, “c”, “2”

      3 - “d”, “e”, “f”, “3”

      4 - “g”, “h”, “i”, “4”

      5 - “j”, “k”, “l”, “5”

      6 - “m”, “n”, “o”, “6”

      7 - “p”, “q”, “r”, “s”, “7”

      8 - “t”, “u”, “v” , “8”

      9 - “w”, “x”, “y”, “z”, “9”

      0 – “ “ (space), “0”

c. Pressing “OK” at any stage shows the ***message number*** screen.

d. Pressing “*” at any stage will cancel the operation and the device will return to the ***message listing*** screen (title: "Back").

e. Pressing “#” at any stage will delete the last character and the cursor will move to the previous location (title: "Delete").

5.5 Message number screen

a. The user enters the destination phone number using the number buttons. The maximum length of the number is 8 (decimal) digits, with no hyphens.

b. Pressing “OK” sends the message. After the message is sent the device will return to the ***message listing*** screen.

c. The button “*” serves as a cancel button. Pressing it will return to the ***message listing*** screen (title: "Back").

6 Slave Mode – *15 points bonus*

Many cellular phones today allow you to connect them to your PC (usually via USB) and control them from a convenient GUI: Reading all the messages, deleting un-wanted messages, and even sending new messages. This is called “**Slave Mode”** (meaning – the cellphone is “slave” and the PC is “master”, telling it when to do).

You are required to implement this feature; To make it simple, we will not ask you to implement USB, but use the driver you already wrote for the UART device in order to allow communication with the PC.

The **Slave Mode Protocol** specification is similar to the protocol used in exercise 2, and follows the exact same rules. Please refer to section 4 in exercise 2 and refresh with the details.

6.1 Messages sent from PC to your device

GetMessagesCount

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	“MC”
End Marker	1 byte	1 byte	0x2E

ReadMessage

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	“RM”
Message Index	1 byte	2 bytes	The index of the stored message to retrieve
End Marker	1 byte	1 byte	0x2E

DeleteMessage

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	“DE”
Message Index	1 byte	2 bytes	The index of the stored message to delete
End Marker	1 byte	1 byte	0x2E

SendMessage

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	“SM”
Destination Length	1 byte	2 bytes	N, the number of bytes in the number, must be between 1 and 12, but you cannot assume that (you must verify that and drop illegal packet)
Destination	N bytes	2N bytes	The destination to send to
Message Length	1 byte	2 bytes	M, the number of characters in the message
Destination	M bytes	2M bytes	The message to send
End Marker	1 byte	1 byte	0x2E

UpdateMessage

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	“UM”
Message Index	1 byte	2 bytes	The index of the stored message to update
New Message Length	1 byte	2 bytes	M, The number of characters in the new message
New Message	M bytes	2M bytes	The new message. The message with the given index as above should keep its timestamp, index and sender/receiver data, but change its content.
End Marker	1 byte	1 byte	0x2E

6.2 Messages sent from your device to the PC

Note, that all the messages in this section are responses to the messages sent from the PC. These messages are to be sent after the processing of the request has completed:

Invalid Request

No matter what request was sent, if your device cannot interpret the message properly, you should reply with this response:

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"IR"
End Marker	1 byte	1 byte	0x2E

GetMessagesCount Response

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"MC"
MessageCount	1 byte	2 bytes	The number of messages stored on the device's flash
End Marker	1 byte	1 byte	0x2E

DeleteMessage Response

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"DE"
Status	1 byte	2 bytes	0 = Message deleted successfully 1 = Invalid message index 2 = Any other error
End Marker	1 byte	1 byte	0x2E

SendMessage Response

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"SM"
Status	1 byte	2 bytes	0 = Message sent successfully 1 = Other error
End Marker	1 byte	1 byte	0x2E

ReadMessage Response

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"RM"
Message Index	1 byte	2 bytes	The index of the retrieved message (same as in the request)
Time	T byte	2T bytes	The time the message was received/sent
Destination Length	1 byte	2 bytes	N, the number of bytes in the destination number
Destination	N bytes	2N bytes	The destination the message was sent to
Source Length	1 byte	2 bytes	M, the number of bytes in the source number

Source	M bytes	2M bytes	The source the message was sent from
Message Length	1 byte	2 bytes	K, the number of characters in the message
Destination	K bytes	2K bytes	The message
End Marker	1 byte	1 byte	0x2E

SendMessage Response

Field	Data Length	Actual field length	Notes
Message Code	2 bytes	2 bytes	"UM"
Status	1 byte	2 bytes	0 = Message updated successfully 1 = Other error
End Marker	1 byte	1 byte	0x2E

7 Exercise credit

The maximal grade for this EX (including bonuses) is 115.

Good luck!