# Lab 3 Alarm Clock

This laboratory assignment accompanies the book, Embedded Systems: Real-Time Interfacing to ARM Cortex M Microcontrollers, ISBN-13: 978-1463590154, by Jonathan W. Valvano, copyright © 2016.

**Goals**
- Develop a graphics driver for the LCD that can plot lines and circles
- Design a hardware/software interface for a keyboard or individual switches
- Design a hardware/software driver for generating a simple tone on a speaker
- Measure supply current necessary to run the embedded system
- Implement a digital alarm clock using periodic interrupts

**Review**
- Valvano Section 3.4 on developing modular software
- Valvano Chapter 4 on Basic Handshake Mechanisms,
- Valvano Sections 4.5 and 4.10 on edge-triggered interrupts and the keypad
- Valvano Section 5.7 on periodic interrupts

**Starter files**
- **Lab3_Artist.sch** and your Lab 2 (**ST7735_4C123.zip**)

**Background**

Labs in EE445L are extremely open ended. For Labs 3, 4 and 5 you will be given a **requirements document**. Your TA will be considered your client or customer. A grade of B can be achieved by satisfying these minimum specifications. To achieve higher grades, you are expected to expand sections 2.1 and 2.5 describing what your system will do. You are free to make any changes to this document as long you achieve the educational goals for the lab. All changes must be approved by your TA. Excellent grades are reserved for systems with extra features and are easy to operate. You will need a LaunchPad, color LCD, speaker, switches, PN2222, 1N914, and some resistors for this lab.

**Required Hardware**

EK-TM4C123GXL, http://www.ti.com $12.99
Sitronix ST7735R Color LCD, http://www.adafruit.com/products/358 $19.99
8-Ω or 32-Ω speaker

**Required Parts**

| Parts | www.Mouser.com | www.element14.com | www.digikey.com | Price |
|---|---|---|---|---|
| PN2222 N-channel BJT | **512-PN2222ABU** | **PN2222ABU** | **PN2222BU-ND** | **$0.22** |
| 1N914B 4ns switching diode | **512-1N914B** | **1N914B** | **1N914B-ND** | **$0.10** |
| Tactile switch | **653-B3F-1050** | **B3F-1050** | **SW405-ND** | **$0.29** |

**Requirements document**

1. Overview

1.1. Objectives: Why are we doing this project? What is the purpose?

The objectives of this project are to design, build and test an alarm clock. Educationally, students are learning how to design and test modular software and how to perform switch/keypad input in the background.

1.2. Process: How will the project be developed?

The project will be developed using the TM4C123 board. There will be switches or a keypad. The system will be built on a solderless breadboard and run on the usual USB power. The system may use the on board switches and/or the on board LEDs. Alternatively, the system may include external switches. The speaker will be external. There will be at least four hardware/software modules: switch/keypad input, time management, LCD graphics, and sound output. The process will be to design and test each module independently from the other modules. After each module is tested, the system will be built and tested.

1.3. Roles and Responsibilities: Who will do what? Who are the clients?

EE445L students are the engineers and the TA is the client. Students are expected to modify this document to clarify exactly what they plan to build. Students are allowed to divide responsibilities of the project however they wish, but, at the time of demonstration, both students are expected to understand all aspects of the design.

1.4. Interactions with Existing Systems: How will it fit in?

The system will use the TM4C123 board, a ST7735 color LCD, a solderless breadboard, and be powered using the USB cable.

1.5. Terminology: Define terms used in the document.

Power budget, device driver, critical section, latency, time jitter, and modular programming. See textbook for definitions.

1.6. Security: How will intellectual property be managed?

The system may include software from Tivaware and from the book. No software written for this project may be transmitted, viewed, or communicated with any other EE445L student past, present, or future (other than the lab partner of course). It is the responsibility of the team to keep its EE445L lab solutions secure.

2. Function Description

2.1. Functionality: What will the system do precisely?

The clock must be able to perform five functions. 1) It will display hours and minutes in both graphical and numeric forms on the LCD. The graphical output will include the 12 numbers around a circle, the hour hand, and the minute hand. The numerical output will be easy to read. 2) It will allow the operator to set the current time using switches or a keypad. 3) It will allow the operator to set the alarm time including enabling/disabling alarms. 4) It will make a sound at the alarm time. 5) It will allow the operator to stop the sound. An LED heartbeat will show when the system is running.

2.2. Scope: List the phases and what will be delivered in each phase.

Phase 1 is the preparation; phase 2 is the demonstration; and phase 3 is the lab report. Details can be found in the lab manual.

2.3. Prototypes: How will intermediate progress be demonstrated?

A prototype system running on the TM4C123 board, ST7735 color LCD, and solderless breadboard will be demonstrated. Progress will be judged by the preparation, demonstration and lab report.

2.4. Performance: Define the measures and describe how they will be determined.

The system will be judged by three qualitative measures. First, the software modules must be easy to understand and well-organized. Second, the clock display should be beautiful and effective in telling time. Third, the operation of setting the time and alarm should be simple and intuitive. The system should not have critical sections. All shared global variables must be identified with documentation that a critical section does not exist. Backward jumps in the ISR should be avoided if possible. The interrupt service routine used to maintain time must complete in as short a time as possible. This means all LCD I/O occurs in the main program. The average current on the +5V power will be measured with and without the alarm sounding.

2.5. Usability: Describe the interfaces. Be quantitative if possible.

There will be two to four switch inputs. In the main menu, the switches can be used to activate 1) set time; 2) set alarm; 3) turn on/off alarm; and 4) display mode. The user should be able to set the time (hours, minutes) and be able to set the alarm (hour, minute). Exactly how the user interface works is up to you. After some amount of inactivity the system reverts to the main menu. The user should be about to control some aspects of the display configuring the look and feel of the device. The switches MUST be debounced, so only one action occurs when the operator touches a switch once.

The LCD display shows the time using graphical display typical of a standard on the wall clock. The 12 numbers, the minute hand, and the hour hand are large and easy to see. The clock can also display the time in numeric mode using numbers.

The alarm sound can be a simple square wave. The sound amplitude will be just loud enough for the TA to hear when within 3 feet.

2.6. Safety: Explain any safety requirements and how they will be measured.

The alarm sound will be VERY quiet in order to respect other people in the room during testing. Connecting or disconnecting wires on the protoboard while power is applied may damage the board.

3. Deliverables

3.1. Reports: How will the system be described?

A lab report described below is due by the due date listed in the syllabus. This report includes the final requirements document.

3.2. Audits: How will the clients evaluate progress?

The preparation is due at the beginning of the lab period on the date listed in the syllabus.

3.3. Outcomes: What are the deliverables? How do we know when it is done?

There are three deliverables: preparation, demonstration, and report.

**Preparation**

1) Copy and paste the requirements document from the lab manual. Edit it to reflect your design.

2) Draw a detailed circuit diagram showing all external hardware connections. We expect you to use PCBArtist (because this is the program with which we will be designing PCBs in Labs 6 and 7). Label all hardware chips, pin numbers, and resistor values. You do not have to show connections to components on the evaluation board. You must have in your possession all external hardware parts, but you do not have to construct the circuit. This parts requirement is relaxed for the labs first thing in the morning.

3) For each module you must have a separate header and code file. As stated earlier we expect at least four modules. As part of the preparation, you need to have the software designed, written and compiled. For the preparation, you do not need to have run or debugged any code. For the modules you have written include a main program that can be used to test it. The SysTick or timer module used to maintain time must be written at a low level, like the book, without calling TivaWare driver code. Other code (LCD, GPIO, and PLL) can use TivaWare driver code.

4) Write one main program that implements the digital alarm clock. Figure 3.1 shows the data flow graph of the alarm clock.

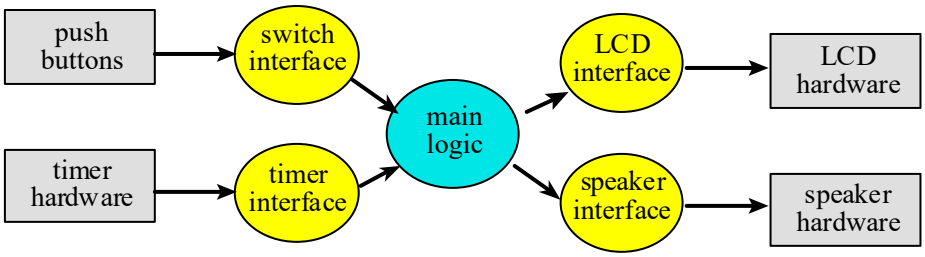


*Figure 3.1. Data flows from the timer and the switches to the LCD and speaker.*

Figure 3.2 shows a possible call graph of the system. Dividing the system into modules allows for concurrent development and eases the reuse of code.

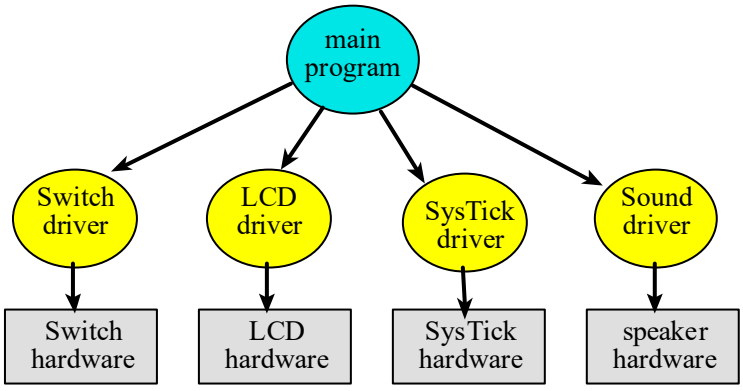


*Figure 3.2. A call graph showing the four modules used by the alarm clock.*

**Procedure**

1) Build and test any external hardware needed. Debug each module separately. Debug the overall alarm clock. Measure how long it takes to update the graphical time on the LCD. Identify all shared I/O ports and global variables. I.e., list all the permanently allocated variables that have read or write access by more than one thread. Next, consider what would happen if the interrupt occurred between any two instructions of the main program. Remember high priority interrupts can suspend lower priority ISRs. Look for critical sections, and if you find any remove them. Document in your lab manual that each shared object is not critical. During checkout, the TA may ask you to prove that your system has no critical sections.)

2) Record the +5 V and +3.3V power line voltages versus time signal and measure the rms noise level.

3) Make sure the lab power supply is set at +5V. Remove the USB power cable and carefully power your alarm clock system using a +5V lab power supply. Double check the positive and negative connections before turning it on. If you are at all unsure about this measurement, ask your TA for help. Measure the required current to run the alarm clock. Take a measurement with and without the alarm sounding.

**Deliverables (exact components of the lab report)**

A) Objectives (final requirements document)

B) Hardware Design

Showing all external components connected to the TM4C123 board. You do not need to show hardware components on the TM4C123 board.

C) Software Design (upload your files as instructed by your TA)

If you organized the system different than Figure 3.1 and 3.2, then draw its data flow and call graphs

D) Measurement Data

Plot the +5 and +3.3 supply voltages versus time and record the rms magnitudes

Plot the speaker voltage (or output voltage) versus time during an alarm sound

Measurements of current required to run the alarm clock, with and without the alarm

E) Analysis and Discussion (give short 1 or two sentence answers to these questions)

1) Give two ways to remove a critical section.

2) How long does it take to update the LCD with a new time?

3) What would be the disadvantage of updating the LCD in the background ISR?

4) Did you redraw the entire clock for each output? If so, how could you have redesigned the LCD update to run much faster, and create a lot less flicker?

5) Assuming the system were battery powered, list three ways you could have saved power.

**Checkout**

Using the first main program, you should be able to demonstrate all the "cool" features of your LCD graphics driver. Next, download the digital alarm program. Demonstrate that your digital alarm clock is stand-alone by turning the power off then on. The digital alarm clock should run (the time will naturally have to be reprogrammed) without downloading the software each time.

**Hints**

1) The requirements document should change a couple of times during the lab as you determine features that can and cannot be performed.

2) You can interface an 8 or 32-Ω speaker using a NPN transistor like PN2222 to an output port. The resistor controls the loudness of the sound. Please make it quiet, try 1 kΩ. The maximum $I_{CE}$ of the transistor must be larger than 3.3V/8Ω or (3.3V/32Ω). The speaker has inductance, so the 1N914 diode is used as a snubber to remove back EMF when the transistor switches off. If you toggle the output pin in the background ISR, then sound will be generated. See Figure 7.13 in the book.

3) You must be careful not to let the LCD show an intermediate time of 1:00 as the time rolls over from 1:59 to 2:00. You must also be careful not to disable interrupts too long (more than one interrupt period), because a time error will result if any interrupts are skipped.

4) You may use 32-bit or 64-bit timer modes on the TM4C microcontrollers. However, it is good practice to refer to the errata for the microcontroller you are using. The errata describe bugs and flaws not listed in the data sheet.

5) If you use the on-board switches then you must activate the internal pull-up resistors. In particular, you will set the corresponding bits in the GPIO_PORTF_PUR_R register. These on-board switches are simply SPST switches to ground. When the switch is pressed, the signal goes to 0V (ground). When the switch is not pressed, the internal pull-up makes the signal go high (3.3V.) Furthermore, coming up out of a reset PF0 is locked, and thus if you use PF0 you will need to unlock it.

6) Learn to use PCBartist. You will need to be proficient with this application during Labs 6 and 7. Using it now for simple circuits will be an efficient use of your time.

7) One way to design the graphical interface is to draw the static components of the clock in Paint, save as BMP, and convert the BMP into C code, and then use **`ST7735_DrawBitmap`** to draw the static image on the screen

Then to create the hour and minute hand, use the line draw function from Lab 2.