METU NORTHERN CYPRUS CAMPUS

Introduction to Microprocessors | Embedded Systems Development

EEE 347 | CNG 336
Module 4

SPRING 22|23

LAB MODULE #4: WORKING WITH STANDARD USER INTERFACE DEVICES, A/D CONVERTERS, AND MOTORS

4.1 OBJECTIVE

The fourth laboratory module targets the integration of standard user input-output interfaces, sensor signals processed by ATmega A/D converter, and a DC water-pump (motor) to the smart farming system. These components make up the last stage of your semester project. You will continue subsystem development and integration using high-level programming. It is very important that you start Module 4 with the Module 3 system in a fully functional state. Any missing pieces from Module 3 should be completed before your final design demonstration.

You will follow a modular approach as before, taking advantage of subroutines for the organization. You will then debug and simulate the code first on Microchip Studio, followed by the embedded system on Proteus, paying attention to different operation modes and corner cases. Design and simulation results should be well documented in your submitted report before your scheduled laboratory session. You will then be prepared to demonstrate your development and verification process quickly and effectively to the lab instructor by sharing the Microchip Studio and Proteus sessions directly from your computer and answering any questions. You need to be ready to create various scenarios by entering different operational system modes upon request. Your semester project that consists of accumulated content from Lab Modules 1-4 should be complete at this time.

4.2 PROBLEM DESCRIPTION

4.2.1 New Features

The following new features will be the primary additions in this module to the embedded system you developed in Module 3:

i. **User-side LCD display**: User output emulated by the Virtual Terminal (VT) at the Bluetooth interface in Module 3 will be replaced by a **20-character x 4-line LCD display**, that is connected to a new user side Atmega128 MCU. User-side MCU will receive messages through Bluetooth from the central smart data-logger to display on user screen.

ii. **User-side keypad**: The Virtual Terminal (VT) user input at the Bluetooth interface in Module 3 will be replaced a simple **4 x 3 matrix keypad** to enter simple numerical user choices. The keypad will directly communicate user selections to the new ATmega128 MCU from (i), which will then be sent to the central smart data-logger through Bluetooth.

iii. **Remote-node sensors**: Temperature, moisture, water level and battery level sensors will not be directly intergrated into your project but will be modeled as analog voltages (Vsource) that correspond to equivalent physical parameters. The new ATmega128 MCU integrated to the remote-sensor side will digitize these **analog voltage inputs connected to different channels** once every 30 minutes in real application and communicate them to the central smart data-logger through the Xbee interface, using the packet formatting (data packet and log request packet) established in previous lab modules. For the purposes of accelerated simulations, you will program **analog data acquisition/conversion cycles to be once every 10 seconds** for each of the four sensors.

iv. **Remote-node instant display**: Digitized 5-bit integer values of 4 sensors will be displayed simultaneously on a locally connected **16-character x 2 line LCD display** in hexadecimal format. Displayed values will be updated upon every A/D conversion. A message should be permanently displayed to "*change battery immediately*" when battery level is measured below 3.2 V.

v. **Remote-node water-pump**: A water-pump with a **DC motor** will be operated by the MCU from (iii) to water the plants at the location of the remote node. Watering frequency will be pre-programmed to be once a day for 10 minutes in real application using internal timer subsystem. For the purposes of accelerated simulations, you will program the watering frequency to be **once every 10 seconds for 5 seconds**. DC motor speed that will directly control the water flow rate will be programmed to be proportional to the soil moisture level (as measured by the moisture sensor).

4.2.2 Functions and Interfaces

The peripheral user node and remote node associated with the *smart data logger system* will be designed to complete the semester project. The subsystems at each node will be designed around an ATmega128 MCU, as depicted in Figure 4.1. Sensor analog inputs are converted to 5-bit digital values, as shown, and are displayed instantly at 16 **x** 2 LCD for Temperature (T), Moisture (M), Water Level (W), and Battery Level (B).

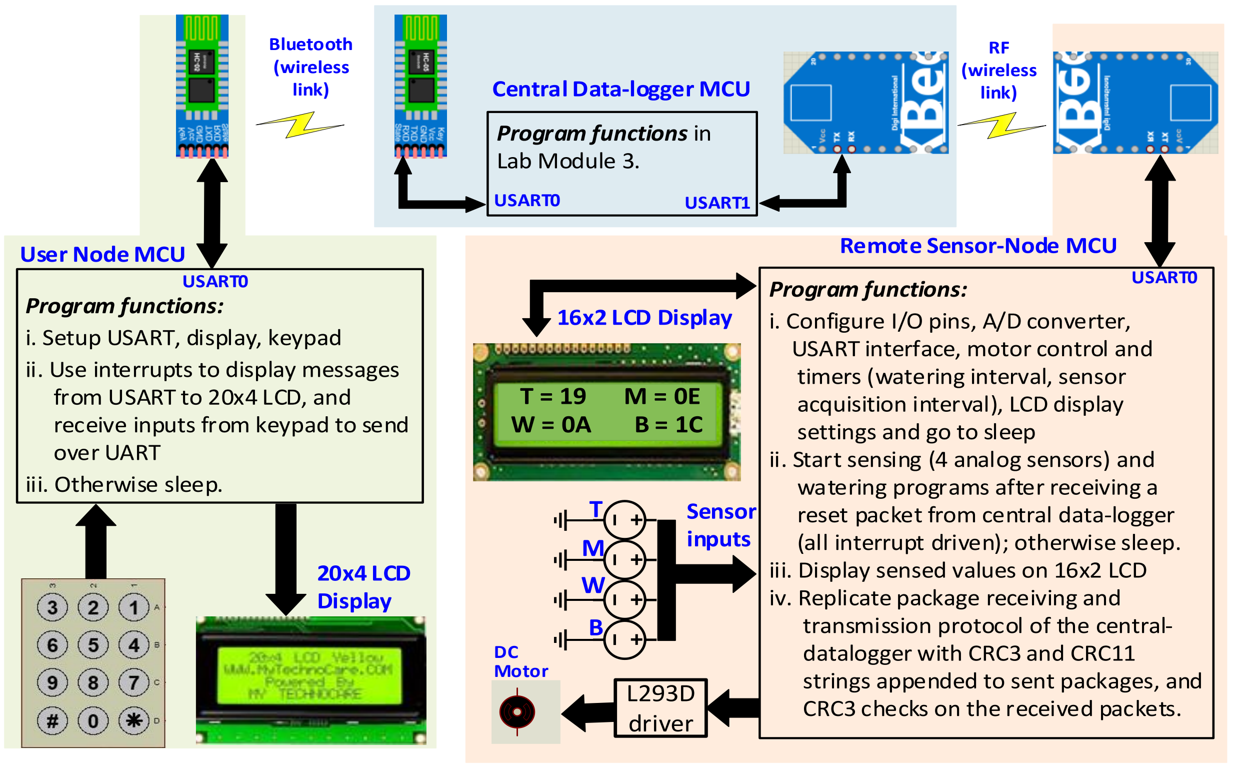
Bluetooth (wireless link)
Central Data-logger MCU
Program functions in Lab Module 3.
USART0
USART1
RF (wireless link)
User Node MCU
USART0
Program functions:
i. Setup USART, display, keypad
ii. Use interrupts to display messages from USART to 20x4 LCD, and receive inputs from keypad to send over UART
iii. Otherwise sleep.
20x4 LCD Display
Remote Sensor-Node MCU
USART0
16x2 LCD Display
T = 19 M = 0E
W = 0A B = 1C
Program functions:
i. Configure I/O pins, A/D converter, USART interface, motor control and timers (watering interval, sensor acquisition interval), LCD display settings and go to sleep
ii. Start sensing (4 analog sensors) and watering programs after receiving a reset packet from central data-logger (all interrupt driven); otherwise sleep.
iii. Display sensed values on 16x2 LCD
iv. Replicate package receiving and transmission protocol of the central-datalogger with CRC3 and CRC11 strings appended to sent packages, and CRC3 checks on the received packets.
T
M
W
B
Sensor inputs
DC Motor
L293D driver

Figure 4.1. Functions/interfaces of the smart farming system with user node and remote sensor node.

4.3 DESIGN AND REPORTING

4.3.1 Preliminary Work

a) The following figure is a block diagram of a successive approximation A/D converter. Explain the purpose of each part of the block diagram shown below. Do you think this A/D converter has parallel or serial digital interface? Explain.

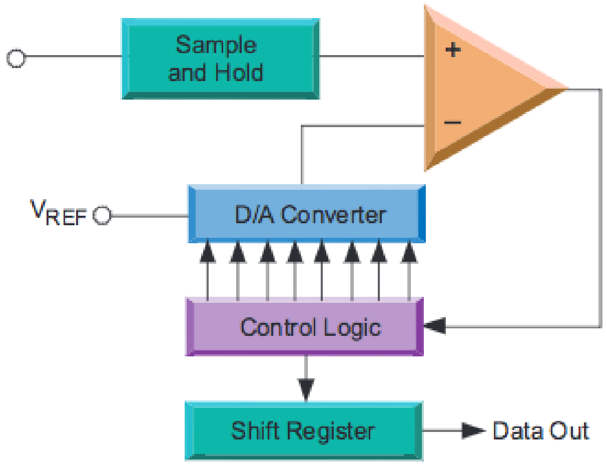

Figure 4.2. Successive Approximation ADC

b) If ATmega128 operates with an MCU clock frequency of 16-MHz, estimate the minimum possible single-ended A/D conversion time, showing corresponding MCU configuration requirements, and calculation. You may ignore the time it takes to initialize the analog circuitry, and may assume free-running mode.

c) The following table shows the pre-amplified single-ended voltage range corresponding to the output of each sensor at the remote node, and the corresponding digital values they should be converted to. Since the system uses only 5 bits (based on the protocol described in Lab Module 2) to represent each of the sensed parameters, calculate and fill in the table with the *effective resolution* of the system in terms of voltage. Also, describe how you may use the existing 10-bit A/D in ATmega128 in obtaining the provided 5-bit adjusted values.

Table 4.1 Sensed physical parameter range and resolution at the remote node

Parameter	Min. (V)	Max. (V)	Min. (digital)	Max. (digital)	Eff. Resolution (mV)
T	2.0	4.0	0x03	0x1B	
M	1.8	4.2	0x02	0x1E	
W	2.0	2.8	0x04	0x1A	
B	3.0	5.0	0x01	0x01F	

d) Given the rotational speed of the waterpump (motor) will vary between **20% to 80%**, depending on the moisture (M) level at the remote node, calculate and indicate relevant PWM generation settings you plan to program in the motor control section of your remote node solution. What will be the default motor speed before any data has been received from moisture sensor? Why?

e) Outline the main differences between 16x2 LCD discussed in lectures, and 20x4 LCD to be utilized at the user node.

f) Sketch a system schematic diagram that has the full smart farming system, including 3 ATmega128 MCUs and their connectivity to the periperhal components. Your sketch should be organized and readable, preferably using a drawing application such as Visio. Pin level connectivity should be clear for each pin of each omponent.

g) Sketch an algorithmic flowchart to accurately show the program executed in the Remote Sensor Node MCU.

h) Sketch an algorithmic flowchart to accurately show the program executed in the User Node MCU.

i) Considering your answers to (f-h), and sytem farming system representation in Figure 4.1, use component datasheets to investigate estimated minimum (IDLE) and maximum (ACTIVE) power dissipation for components in your system, including times when both wireless transmission interfaces are active into your worst-case power scenario. Complete the blanks in Table 4.1.

Table 4.2 Idle and worst-case system power estimation

Component Power (mW)	Approx. best-Case (IDLE)	Approx. worst-Case (ACTIVE)
MCU (Central)		
MCU (User-node)		
MCU (Remote-node)		
Bluetooth Interface		
Xbee Interface		
16x2 LCD display		
20x4 LCD display		
Motor driver	0	200
Waterpump	0	1000

4.3.2 Design

Follow a similar implementation approach as in Module 3 for high-level C functions to divide and conquer the design. Do a task division with your partner to develop required last pieces of the system. Once you are convinced each piece works correctly, integrate them together into a single system in Proteus. The embedded C-code that will run on each of the three MCUs should be well annotated, and easy to modify with use of #define lines for parameters instead of hardcoded values in the code.

- Use Microchip Studio to create two new projects on top of the project you carried over Module 3 for the central smart datalogger system. Use the new projects to develop the embedded code for user node and remote node.
- After each project successfully compiles, and is proven correct after debug, you may move on to generating Proteus project with three MCUs and peripherals. All the new periperals you need to add to your system in Module 4 should already be in the Proteus parts library: KEYPAD-PHONE (numeric keypad), L293D (motor driver), LM016L (16x2 LCD), LM044L (20x4 LCD), Motor (water pump), VSOURCE (sensors).

4.3.3 Verification

Use Proteus to create Module4_Proteus project. Instantiate and connect ATMEGA128 microcontrollers, Xbee, HC-05 and new peripheral components as in Figure 4.1. Make sure your COM settings are configured correctly for each wireless communication device, and wireless pairing is properly done using

VSPE, as in Module 3. Load your debugged hex codes to three MCUs, and run different cases, especially demonstrating each mode of operation, communication with user and with remote sensor, user node operations, remote sensor node operations, and correct logging of data to internal and external memory. Take screen captures (PrtScr) again for the critical cases to include in your report. If you would like to add additional LEDs to certain DEBUG pins on parallel ports to make your debugging job easier, you are welcome to do so. It is recommended that you utilize Proteus VSM debugging mode, and use stepwise simulation to watch activities by selecting visual aids under “Debug” menu, such as Watchwindow, CPU registers, Data Memory, I/O registers, etc.

4.3.4 Report

- Follow the strict guidelines and format described in detail in the first lab manual to complete a concise and comprehensive report.
- Dedicate a paragraph in the introduction section to discussing how you divided the design, implementation, and verification tasks between two team members.
- Your report should represent your team’s work only, and should clearly document your solutions to preliminary exercises (4.3.1), your modular and organized code, MCU and full embedded system critical verification results along with Proteus schematics. Each simulation screen you include should be carefully annotated and explained in a paragraph. If there are any problematic cases that do not function correctly, these should be discussed.
- Your conclusions from the overall project experience should be carefully considered and written.
- Include calculations in your Conclusion section on estimated energy requirement (in Joules) of the overall system in one day (24 hours), using the numbers in Table 4.2. Take advantage of **real application assumptions** provided in Section 4.2.1 when calculating energy requirements to make sure you are not overly pessimistic.