Jason McKeon
ETEC 471
Final Project Description
Digital Altimeter
December 10, 2003
Introduction:

When spending time in the mountains hiking, biking, or traveling by car, one would often want to know their current altitude above sea level. Knowing the altitude can be important in locating different regions on a map. The only time that the altitude would be known is if it is posted on informational signs or read from difficult to read analog altimeters. This is why I propose to design a digital altimeter. Many high end commercial altimeters are expensive, which often makes them unavailable to many consumers. The cost of the proposed design would only be a fraction of the cost of a commercial device.

In addition to measuring altitude, the proposed design will measure temperature and atmospheric pressure. All of this information will be displayed on an LCD. The temperature and pressure measurements will be used to calculate the current altitude in either meters or feet. When a user wants to track the differences in altitude they have traveled, the digital altimeter will have the option of storing altitude values over a certain period of time.

General Description:

The digital altimeter will be a battery operated portable device. This is a critical function of the device since it will be designed to measure altitudes on outdoor trips. Important hardware components of the project include the pressure sensor, temperature sensor, keypad, and LCD display. Each sensor transmits critical information which will be displayed on a 2 x 16 character LCD. Altitude measurements will be derived from both the temperature and pressure sensors. In addition, the direct pressure and temperature measurements will be monitored through the user interface of the digital altimeter. The user input device will be a 16 button keypad. A sketch of the project hardware assembly with maximum dimensions is shown in Figure 1.
The final dimensions of the digital altimeter in its final enclosure will not exceed 6 inches wide, 8 inches tall, and 1.5 inches thick. During the final stages of development, the altimeter will be placed within a plastic enclosure. This enclosure will increase its portability and allow it to withstand harsh conditions that may be present outdoors. In addition, the altimeter will be powered by four AA alkaline batteries, which are widely available. Rechargeable nickel-metal hydride batteries could also be used. As shown in Figure 1, the user interface will consist of a 2 x 16 LCD display, a 16 button keypad, and an on/off button.

**Detailed Functional Description:**

The digital altimeter will be implemented using the Motorola MC9S12DP256 (9S12) microcontroller. It is not the most ideal microcontroller for the project, but it is the best choice due to programming code familiarity and a large number of available resources for this part. Another microcontroller that could have been used for the altimeter is the HC08. It has all of the features necessary to implement the design. Specifications for one model from this series include 2 Kbytes of RAM, 1 Kbytes of EEPROM, and 60 Kbytes of Flash EEPROM. Also included is an
A/D converter and SPI interface. These are the same resources that will be used on the 9S12 microcontroller.
The detailed block diagram of the project hardware is shown in Figure 2. Critical components shown are the MC9S12DP256 microcontroller, digital temperature sensor, 2 x 16 character LCD, 16 button keypad, absolute pressure sensor, and the battery operated power supply. Each of the components with their circuit connections are shown in the block diagram.

All microcontroller resources used in the design are shown in the detailed block diagram. The resources of the 9S12 used will be the analog to digital converter (A/D), serial peripheral interface (SPI), general purpose input/output ports (GPIO, Ports A, B, K, and P), and the internal memory. Outputs from the 6 Volt battery source and the pressure sensor will be connected to the A/D converter. A/D conversions of the battery source will be used by the microcontroller software to detect low battery voltage. Also, the A/D converter will convert the pressure sensor output voltage, which is the critical variable for the altitude calculation. Communication between the digital temperature sensor and the microcontroller will occur over the SPI. The SPI will supply the input clock signal through the SCK to the sensor. Both the 2x16 character LCD and the 16 button keypad will interface with the 9S12 through Ports A, B, and K. The internal memory of the 9S12 will contain all software and variables.

An important feature of the 9S12 is the large amount of memory available. It includes 256Kbytes of Flash EEPROM, 12Kbytes of RAM, and 4Kbytes of EEPROM. Each form of memory will be utilized for the functions of the altimeter. The main software code will be stored in the Flash EEPROM. Large amounts of code can be stored in this section of memory. Having a large amount of memory will be useful since the altimeter will have a complex user interface, requiring relatively large amounts of software code. Temporary program variables will be stored in the microcontroller RAM. Another purpose of the RAM is to debug program code during the software development process. The much larger RAM of the 9S12 compared to the earlier HC12
will allow significant amounts of program code to be temporarily stored for debugging purposes. An important memory type for digital altimeter functionality will be the 4K EEPROM. This section of memory will be used for data logging. It will store altitude measurements over a period of time. In the event that the altimeter loses power or is turned off, this information will be retained in the non-volatile EEPROM. Altitude measurement values will be in the form of 16-bit words. Therefore, the EEPROM will be able to store 256 altitude measurements. As a result, the microcontroller will be capable of storing altitude values every 15 seconds for a maximum period of approximately one hour.

There will be five main hardware components in the project. These are the digital temperature sensor, absolute pressure sensor, power supply, 2 x 16 LCD display, and the 16 button keypad. How these components are interfaced with the microcontroller is described above and in Figure 2. One possible digital temperature sensor that may be used in the project is the Maxim MAX6629. It is a 12 bit plus sign digital temperature sensor that communicates with the microcontroller through the SPI. The advantage of this sensor is that it is a read-only device. Therefore, the microcontroller only needs to provide a SCK input to the device and read its output through the SPI. Output from the temperature sensor will be in the form of a 13 bit serial bit stream. Examples of how the digital output is interpreted are shown in Table 1.

<table>
<thead>
<tr>
<th>Temperature (deg C)</th>
<th>Digital output (binary)</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>0 1001 0110 0000</td>
</tr>
<tr>
<td>125</td>
<td>0 0111 1101 0000</td>
</tr>
<tr>
<td>0.0625</td>
<td>0 0000 0000 0001</td>
</tr>
<tr>
<td>0</td>
<td>0 0000 0000 0000</td>
</tr>
<tr>
<td>-25</td>
<td>1 1110 0111 0000</td>
</tr>
</tbody>
</table>

*Table 1: Digital Temperature Sensor Output*
Digital outputs of other digital temperatures sensors have a format similar to the MAX6629, making the part interchangeable with others. Pressure measurements will be taken using an analog pressure sensor such as the Motorola MPX5100 absolute pressure sensor. The output of the pressure sensor will be connected to the A/D converter of the 9S12 at the AN0 input. This value will then be converted to pressure (kPa) using an equation relating voltage to pressure, supplied by the manufacturer.

The power supply of the system will use a source of four AA batteries which are equal to 6 volts at full charge. To reduce the battery output voltage to the 5 volts necessary for the 9S12 and hardware, a step-down DC-DC converter will be used. Two DC-DC converters that could be used for the power supply are the MAX639 and MAX710 DC-DC converters. Both converters are capable of supplying up to 200 mA of current to the system. The system will have the option of operating in either a sleep mode or a low power mode to reduce battery consumption. When normal system operation is desired with a lower power output, the system will be able to operate at a lower clock frequency.

Additional hardware components are the RS232 interface and the piezo buzzer. The RS232 interface circuitry will be connected to the SCI of the 9S12. This will allow the microcontroller to send and receive data from a serial port of a PC. The piezo buzzer will be used to sound an alarm when the altimeter reaches a user-defined altitude.

If there is time during the project development process, a Real Time Clock IC may be added to the system to provide a time display. This will allow a user to keep track of the current time and date.
Software Description:

The software for the digital altimeter will be written in both C and assembly language. Most of the software will be in C programming language because it is a higher level language that will allow more complex program modules to be written in a shorter amount of time. Assembly language will be used to initialize the microcontroller and perform other low level tasks. To control the various software tasks, the MicroC/OS-II preemptive kernel will be used. It will allow tasks to be completed faster than using a time-slice scheduler. This will be important since the digital altimeter has several variables that need to be sampled within a limited period of time. The following are the software modules and real-time kernel resources that will be used in the project.

LCD: The LCD module will control display initialization, cursor movements, and character alignment. It will control the display configuration for the various user interface modes and display temperature, pressure, and altitude information. An already existing module will be used for these functions that will be modified to meet design specifications.

KEYPAD: The keypad module will control the functions of the 16 keys on the keypad. It will control program flow when a user presses a key and allow user input in certain user modes. This module will also be modified from an existing module.

TEMPERATURE: This module will handle communication between the microcontroller and the temperature sensor. It will read the digital output of the sensor and convert that to a decimal value that can be displayed on the LCD and used in the altitude calculation.
PRESSURE: The pressure module will handle A/D conversions of the pressure sensor output. It will allow current pressure readings to be displayed on the LCD and make the data available for use in the altitude calculation.

EEPROM: This is an already existing module that prepares the on-chip EEPROM for programming. It will also be modified to input altitude measurements.

MAIN: The main program module will contain all of the user interface functions of the altimeter and sleep mode routines. It will also contain the equations for calculating pressure readings, current altimeter values, and unit conversions. The major algorithms of the software will be contained within this module. One of the algorithms will be responsible for sampling the variable parameters from hardware devices a specific intervals and performing the necessary calculations. Another algorithm will use the results of calculations to display values to the LCD and monitor keypad inputs.

MICRO-C/OS-II: This preemptive kernel will be responsible for the task scheduling of the microcontroller. It will be used in favor of a time-slice routine because it is able to execute higher priority tasks and improves the overall response time. The function of the kernel will be to call the other software modules.

**User Interface Description:**

The user interface of the digital altimeter will consist of the LCD display, keypad, and the on/off switch. Outputs of the LCD will contain standard text characters. Information displayed will include user input prompts, several different display configurations, and current parameter measurements. A user will be able to change the altimeter mode and input information using the keypad. The keypad has 16 buttons which include buttons for numbers 0-9, letters A, B, C, and D, and the characters ‘*’ and ‘#’. Number keys and asterisk character will be used when the user
is prompted to set a desired altitude in alarm mode or select user options. Letter keys A-D will be used to switch between the different altimeter modes. The pound character will allow the user to toggle between English units and metric units.

An on/off switch will also be included in the user interface to disconnect the battery power from the rest of the circuit. This will allow the batteries to last longer if the altimeter is not to be used for a long period of time. To reduce battery consumption without disconnecting the batteries from the circuit, a sleep mode will be implemented. The asterisk button of the keypad will be used to put the system in sleep mode if the altimeter is not currently recording altitude values or prompting the user to enter a reference altitude. Sleep mode will also be activated if the altimeter is not used for a period of time. If the altimeter is currently recording altitude values, pressing the asterisk button will reduce the system clock frequency, but not activate the sleep mode. This allows the altimeter to continue to take measurements while reducing the overall power consumption. A sketch of the user interface layout and keypad functionality is shown in Figure 3.

![Figure 3: User Interface Layout](image)

altimeter is currently recording altitude values,
The digital altimeter will operate in four different modes. Each mode will have a different display configuration on the LCD. Pressing the keys A, B, C, or D will allow the user to switch between the different modes. Figure 5 shows which mode is entered if one of these keys is pressed. Upon power up or reset, the initial mode will be Altitude (Figure 4.1). This mode displays the current altitude and temperature in metric units. The units can be converted between metric and English in any mode by pressing the pound key.

<table>
<thead>
<tr>
<th>ALT:</th>
<th>280m</th>
<th>ATM. PRESSURE</th>
<th>LCD Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMP:</td>
<td>23 °C</td>
<td>101 kPa</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 4.1: Altitude Mode  Figure 4.2: Pressure Mode*

Pressure mode displays the current atmospheric pressure (Figure 4.2). This allows the user to track the change in pressure when the altitude is fixed, effectively having the altimeter act as a barometer. The third mode is the Reference mode (Figure 4.3). In this mode, the user can enter a desired altitude to be detected using the number keys. When this altitude is reached, a piezo buzzer will sound. The Altitude Tracking mode controls the altitude tracking feature of the altimeter (Figure 4.4). Altitude tracking records altitude values over a period of time so a user can track the different altitudes that have been reached. This mode allows the user to start or stop collecting altitude measurements or send the data to a PC. From a PC, the user will be able to choose whether or not to receive data from the altimeter. There may be additional commands that could be executed from a PC. The format of the data will be standard ASCII text.
A fifth mode of the digital altimeter will be the calibration mode, which is not one of the four main functional modes. This mode will allow the user to make manual calibrations to increase altitude accuracy. Pressing the ‘1’ button while in the Altitude or Pressure modes will allow a user to enter the calibration mode. In this mode, the user will be able to set the altitude of the digital altimeter to a known reference altitude such as sea level. Also, a base temperature value will be able to be set to further increase the accuracy of the altitude measurements.

In all user modes, error detection software will monitor the inputs for invalid key presses. All keys that are not used in a certain modes will have no effect when pressed. The state diagram of Figure 5 shows how the user interface of the digital altimeter will function. Keys that will be responsible for most mode changes are keys A-D.
Note: Pressing the ‘#’ key in modes A-C changes the units displayed, nothing else changes.

Figure 5: System State Diagram
Development Plan:

Development of the digital altimeter project will occur during winter and spring quarters. Critical parts are to be ordered late in fall quarter and during winter break. There should not be any delivery problems for critical parts since the lead time for most of the parts is less than two weeks. Significant time during winter quarter will be spent learning how to program in C and assembling the hardware components of the project. Most of the C programming will be learned in ETEC 454. Shown below is the weekly schedule of tasks that will be completed late fall quarter through spring quarter.

Fall Quarter 2003

Week 11: Get 9S12 development board and LCD. Order remaining parts needed for the project.

Week 12: Finish final project description and begin studying C programming.

Winter Break 2003

Week 1: Assemble and test battery operated power supply.

Week 2: Study C programming language.

Week 3: Continue studying C programming.

Winter Quarter 2004

Week 1: Get all hardware components ready for testing.

Week 2: Begin testing pressure sensor and learning how to interface it with the A/D converter of the 9S12.

Week 3: Test temperature sensor and begin placing hardware components on a prototype board.

Week 4: Set up keypad and LCD.
Week 5: Begin programming some of the software tasks.

Week 6: Modify LCD and keypad modules and interface with the microcontroller.

Week 7: Work on temperature and pressure sensor modules.

Week 8: Learn how to integrate modules with MicroC/OS.

Week 9: Work on the main program module.

Week 10: Continue working on software.

Week 11: Prepare for and take finals.

Spring Quarter 2004

Week 1: Interface hardware components with 9S12.

Week 2: Work on software modules.

Week 3: Complete hardware interface and prepare for hardware design review.

Week 4: Assign program tasks in MicroC/OS-II.

Week 5: Finish writing user interface and test program modules.

Week 6: Begin integrating hardware and software components for testing. Have software prepared for software system presentations.

Week 7: Test temperature and pressure sensor for accuracy.

Week 8: Conduct field tests of the altimeter to determine the accuracy of the altitude readings.

Week 9: Have all critical software complete for code reviews.

Week 10: Finish fine tuning the project, place it in an enclosure, and demonstrate the project.

Week 11: Prepare for and take finals.
Development Hardware and Software:

Most of the project development will occur in the ET340 lab. Important software tools that will be used in developing the project include CodeWright, Noral debugging software, and the Introl C compiler. Hardware tools that will be used in the lab are the Hewlet Packard 54645D Mixed Signal Oscilloscope, digital multimeters, the Noral debugging pod, and a PC. Additional power supply and sensor testing may occur in the ET338 lab. Since the project is relatively small, no additional lab space will be required.

Demonstration Prototype:

The demonstration prototype will include the 9S12 development board and an additional prototype board. All hardware components will be soldered to a prototype board. If time permits, the hardware may be soldered to a custom PCB. For the demonstration, the project will be placed in a rugged plastic enclosure. This will allow portability and prevent hardware damage. On presentation day, the project will be set up for users to test its operation. An instruction sheet will be provided to describe the user interface functions of the altimeter. A PowerPoint slideshow will present the details and functionality of the digital altimeter.

Electrical Specifications:

Project Specifications:

- Altitude Accuracy: ± 10m (target accuracy, with calibration)
- Altitude Resolution: 1m
- Altitude Range: 0-2000m
- Atmospheric Pressure Range: 15 to 115 kPa
- Temp. Sensor Accuracy: ± 1 °C
- Temp. Sensor Resolution: ± 1 °C
- SCI Baud Rate: 9600 bps
- SCI Mode: 8-bit mode, no parity

Power Requirements:
- Four AA alkaline batteries (6 volts)
  - Estimated life: Approximately 105 hours (alternating between full-power and sleep-modes)
- DC power from wall transformer (for testing purposes)
  - Total worst-case power dissipation: 95.78 mA

Special Environmental Requirements:
- Operating temperature range: 0 to 50 °C
- Humidity: Altimeter will be sensitive to changes in humidity. Effect may be reduced by using altimeter during relatively stable weather conditions and making necessary calibrations.

PCB Size Limits:
- PCB will be no larger than 3 inches wide, 4 inches long. With the components attached to the PCB, it should not have a thickness greater than 1 inch. Since the digital altimeter is designed to be a portable device, the dimensions should be as small as possible.
## Preliminary Parts List:

<table>
<thead>
<tr>
<th>Part</th>
<th>Part Number</th>
<th>Source</th>
<th>Quantity</th>
<th>Price</th>
<th>Power Dissipation</th>
<th>Lead Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC-DC converter</td>
<td>MAX639</td>
<td>Maxim</td>
<td>1</td>
<td>$2.87</td>
<td>20uA</td>
<td>Have</td>
</tr>
<tr>
<td>Regulator circuit</td>
<td>N/A</td>
<td>Mouser</td>
<td>1</td>
<td>$6.12</td>
<td>110uA</td>
<td>Have</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>MC9S12DP256</td>
<td>Digi-Key</td>
<td>1</td>
<td>$19.40</td>
<td>65mA</td>
<td>2 weeks</td>
</tr>
<tr>
<td>LCD</td>
<td>N/A</td>
<td>Todd Morton</td>
<td>1</td>
<td>$12.82</td>
<td>3mA</td>
<td>Available</td>
</tr>
<tr>
<td>Pressure Sensor</td>
<td>MPX5100</td>
<td>Digi-Key</td>
<td>1</td>
<td>$14.30</td>
<td>10mA</td>
<td>2 weeks</td>
</tr>
<tr>
<td>Temp. Sensor</td>
<td>MAX6629</td>
<td>Maxim</td>
<td>1</td>
<td>$1.39</td>
<td>650uA</td>
<td>Have</td>
</tr>
<tr>
<td>Keypad</td>
<td>GH5010-ND</td>
<td>Digi-Key</td>
<td>1</td>
<td>$11.30</td>
<td></td>
<td>2 weeks</td>
</tr>
<tr>
<td>Piezo Buzzer</td>
<td>273-074</td>
<td>Radio Shack</td>
<td>1</td>
<td>$2.99</td>
<td>7mA</td>
<td>Have</td>
</tr>
<tr>
<td>Battery Holder</td>
<td>270-391</td>
<td>Radio Shack</td>
<td>1</td>
<td>$1.69</td>
<td></td>
<td>1 week</td>
</tr>
<tr>
<td>Enclosure</td>
<td></td>
<td>Radio Shack</td>
<td>1</td>
<td>$4.99</td>
<td></td>
<td>1 week</td>
</tr>
<tr>
<td>PCB</td>
<td></td>
<td>Radio Shack</td>
<td>1</td>
<td>$2.49</td>
<td></td>
<td>Have</td>
</tr>
<tr>
<td>Resistors</td>
<td>ET340</td>
<td>variable</td>
<td>1</td>
<td>$1.50</td>
<td></td>
<td>Available</td>
</tr>
<tr>
<td>Capacitors</td>
<td>ET340</td>
<td>variable</td>
<td>1</td>
<td>$1.50</td>
<td></td>
<td>Available</td>
</tr>
<tr>
<td>100uH Inductor</td>
<td>580-18104</td>
<td>Mouser</td>
<td>1</td>
<td>$1.44</td>
<td></td>
<td>Have</td>
</tr>
<tr>
<td>Diode</td>
<td>1N5817</td>
<td>Mouser</td>
<td>1</td>
<td>$0.20</td>
<td></td>
<td>Have</td>
</tr>
<tr>
<td>SOT23 Adapter</td>
<td>33206CA-ND</td>
<td>Digi-Key</td>
<td>1</td>
<td>$2.07</td>
<td></td>
<td>2 weeks</td>
</tr>
<tr>
<td><strong>Total:</strong></td>
<td></td>
<td></td>
<td></td>
<td><strong>$87.07</strong></td>
<td></td>
<td><strong>95.78mA</strong></td>
</tr>
</tbody>
</table>