Nigerian Journal of Technology (NIJOTECH)
Vol. 34 No. 2, April 2015, pp. **332 – 341**
Copyright© Faculty of Engineering,
University of Nigeria, Nsukka, ISSN: 1115-8443
www.nijotech.com
http://dx.doi.org/10.4314/njt.v34i2.18

STANDALONE GENERAL PURPOSE DATA LOGGER DESIGN AND IMPLEMENTATION

N. Bello[1]*, M. Ghraizi[2] and S. O. Adetona[3]
[1, 2] DEPARTMENT OF ELECTRICAL/ELECTRONIC ENGINEERING, UNIVERSITY OF BENIN, BENIN CITY. NIGERIA
[3]DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING, UNIVERSITY OF LAGOS, LAGOS. NIGERIA
E-mail addresses: [1] *nosabello@uniben.edu*, [2] *maherghraizi@gmail.com*, [3] *sadetona@unilag.edu.ng*

ABSTRACT
This paper describes the design of a general purpose data logger that is compatible with a variety of transducers, potentially permitting the measurement and recording of a wide range of phenomena. The recorded data can be retrieved to a PC via an RS-232 serial port. The standalone general purpose data logger centered on a single microcontroller unit (MCU) the PIC18F4520. The circuit takes an input range of 15 – 30V DC in addition; an in-built 9V rechargeable battery provides backup power in the absence of an external source. The microcontroller input form the Transduceroutput was conditioned within the allowable voltage range, the ADC calculates a binary value that is equivalent to its input (analog) voltage. The in-built 10-bit ADC of the PIC18F4520 converts the analog input to a digital output by successive approximation, the 8 MSBs of the ADC are stored whereas the 2 LSBs are discarded which makes the output of the ADC varies from 0 to 255. Although the PIC18F4520 features 256 bytes of non-volatile EEPROM data memory, four AT24C256 2-wire serial EEPROM chips were used for data storage. The four chips were cascaded to form a memory bank that connects to the MCU via a common single 2-wire bus. The hardware employed to achieve the RS-232 serial interface was a 9-pin D-shell serial connector and a voltage level translator chip (Maxim's MAX232). The connector simply allows the device to be coupled to a serial port of a PC using a standard serial cable. The TIMER1 module of the PIC18F4520 was configured such that its 16-bit register is incremented on the rising edge of the external clock signal applied to its input pin (pin 16). The input pins of the MCU that are dedicated to the switches are pulled up by resistors R11, R12, R13, R14, R15 and R16. The push buttons temporarily pull down the input signal when depressed. The output devices consist of a "Liquid Crystal Display" (LCD) device, four general purpose LEDs, and a buzzer. The LCD displays the momentary status of the data-logger in terms of date, time, sampling rate and nature of logged data. The LEDs are used to display the four MSBs of the measured data and the buzzer is used to raise an alarm if and when necessary but may be disabled in hardware by opening jumper J4. The design was implemented and simulated on Labcenterproteus and the components were mounted on a double layered PCB (Printed Circuit Board) that provides the tracks that make up the circuit.The prototype design realized was found to work satisfactorily.

Keywords*: Analog-to-Digital conversion (ADC), Sensor, Sample, Transducer, Interrupt Service Routine (ISR), Micro-Controller Unit (MCU)*

1. INTRODUCTION
The importance of accurate data recording in virtually all spheres of engineering has given rise to a constant demand for improved, cheaper, and more reliable standalone data loggers. Data logging has become indispensible in various engineering and scientific endeavors that range from unattended environmental monitoring to load profile recording for the purpose of energy consumption management. Data Loggers are based on digital processor. It is an electronic device that record data over time in relation to location using built in instrument in corporation with a sensor or external instruments in conjunction with sensors [1]. The process of data logging may be broken down into four main stages as shown in Figure 1. In the first stage, the parameter being measured is *sensed* by an appropriate transducer. The transducer produces an analog electrical signal that relays information about

*** Corresponding Author, Tel: +234-803-256-5711**

the phenomenon of interest. The output signal from the transducer requires signal conditioning to conform to a standard range and strength. Such conditioning may include amplification, impedance transformation and calibration. The second stage consists of *converting* the analog signal to a digital signal using an Analog-to-Digital Converter (ADC) [2]. At this stage, the signal may undergo preliminary processing in the way of interpretation or calibration. In the third stage, the data is logged by a processor and at regular intervals the processor takes the digital data provided by the ADC and *stores* it into memory. In the final stage, the stored data may be further *interpreted* by software such as spreadsheets and other data handling programs. This process is also known as offline processing [3].

Where the system requires that a PC is dedicated to the task of data logging, it is more appropriately called a data acquisition system whereas the term data logger usually refers to a standalone device. In this context, a data logger is a data acquisition system whereas a data acquisition system is not necessarily a data logger. Though PC based data acquisition systems may be practicable in laboratory settings, standalone data loggers are clearly better suited for application in remote sites as is the case in wind assessment studies.

Dedrick[5] designed and implemented a functional low cost data logger using PIC16C73A and serial EEPROM. The data logger works to create logs of ambient temperature over a stipulated period of fifteen days. Though, the design claimed to be adaptable with other sensing transducers for measuring physical phenomenon. In the design, two microcontrollers were used to achieve the functionality of measuring, logging and PC communication over serial port. One of the microcontrollers was dedicated to aid data acquisition and to create a successful log into a logically connected serial EEPROM the other one was used to manage communication with PC. This design falls under a limitation of measuring and logging only one phenomenon (Temperature) at a time, therefore it lacks the ability to measure and log multiple physical phenomena (Multichannel Logger). Furthermore, such design poses a low reliability index as two microcontrollers were used for its operation.

In this work, the data logger was designed around PIC18F4520 [4] which implements a real time clock by connecting a 32.768 KHz crystal oscillator. The microcontroller also features an Enhanced Universal Serial Asynchronous Receiver Transmitter (EUSART) module which is used for RS232 protocol serial communication and a 10 bit Analog-to-digital converter (ADC). These basic features were needed to successfully implement the general purpose data logger. In implementation, the microcontroller was configured to use four of its I/O pins as analog and digital input. These inputs require transducers or an already conditioned transducer output signal to be connected to it. In this paper, firmware refers to the program or code that is written to and run by the MCU Analog inputs are sampled and held for the internal ADC module to successfully convert the input voltage to a corresponding digital value. This value under the firmware running in the microcontroller is logged along with the value of the real time clock (RTC) and calendar, into a series of logically connected four serial (AT24C256) [7] EEPROMs. The logger provided functionalities to configure the sampling rate, which is the rate at which analog inputs are measured and logged. The microcontroller used the inbuilt EUSART module to transfer logged data to a PC over a serial port tethering cable on a retrieval command from the PC. Proteus was used for simulation and testing. After a successful testing of every component and codes, the PCB of the circuit was designed and realized using proteus PCB designer (ARES).

2. HARDWARE DESIGN AND REALIZATION

The hardware components that make up the data logger and their modes of interaction are represented by the block diagram in Figure 1.

2.1 Power

The circuit takes an input range of 15 – 30V DC. In addition, an in-built 9V rechargeable battery provides backup power in the absence of an external source. As depicted in Figure 2, when an external source is connected to the device a voltage regulator (LM7812) provides the charging voltage. A second voltage regulator (LM7805) provides the 5V required by the circuit for operationfrom either source. The charging of the internal battery is controlled by switching a Metal Oxide Silicon Field Effect Transistor (MOSFET) that connects the output of the 12V regulator to the battery, through a current limiting resistor (R17). Diodes (D2, D7, and D8) ensure appropriate current flow while the jumper (J1) provides a means to disable charging in the event that a non-rechargeable battery is being used. The variable potential divider consisting of the resistors R18, R19, and R20 supplies a proportional sample of the battery voltage to the MCU. The MCU in turn controls the charging by applying a control signal to the MOSFET's gate terminal through an inverting buffer – the2N2222 transistor (Q2).

2.2 The user I/O devices

Push button switches are provided on the prototype board. The switches enable the user to make changes to settings such as date and time and the sampling rate of the data logger. A toggle switch is used to select the mode of operation of the data logger – logging, retrieving or standby. The input pins of the MCU that are dedicated to the switches are all weakly pulled up by resistors R11, R12, R13, R14, R15 and R16. The push buttons temporarily pull down the input signal when depressed whereas the toggle switch may be set in any of three positions that would pull either of pins 9 and10 low in two of the cases and neither when left in the float position.

The output devices consist of a "Liquid Crystal Display" (LCD) device, four general purpose LEDs, and a buzzer. The LCD displays the momentary status of the data-logger in terms of date, time, sampling rate and nature of logged data. The LEDs are used to display the four MSBs of the measured data and the buzzer is used to raise an alarm if and when necessary but may be disabled in hardware by opening jumper J4.

2.3 Transducer

The maximum allowable voltage ratings on any pin of the PIC18F4520 are -0.3V and (Vdd + 0.3V) [4] The ADC reference voltages in this design are Vss and Vdd that is 0 and 5V. Hence, for any given transducer the output voltage must be conditioned to fall within the range of 0 - 5V. Another important consideration is the output impedance of the transducer cum conditioning circuit. In accordance with the manufacturer's recommendation the analog input's impedance has to be less than 2.5kΩ to ensure accurate ADC conversion [4]. A variable voltage divider circuit that is represented in Figure 3 was constructed for the purpose of simulating the output of a transducer cum conditioning circuit. The output of the circuit is sourced from an LM7805 regulator and a 9V battery.

2.4 Memory

Although the PIC18F4520 features 256 bytes of non-volatile EEPROM data memory, four AT24C256 2-wire serial EEPROM chips were used instead for the purpose of data storage. The four chips were cascaded to form a memory bank that connects to the MCU via a common single 2-wire bus. Cascading was achieved by allocating to each chip a unique address that was set in hardware by connecting each of the address pins (pins 1&2) [7] to logic low (Vss) or logic high (Vdd.) Each of the ICs features 256Kbits of memory. With three of the chips dedicated to the storage of 8-bit measurements a total of 98304:

$$\frac{No\ of\ bit \times 1024\ \times 3}{storage\ locations} = \frac{256\ \times 1024\ \times 3}{8}$$
$$= 98304 \qquad (1)$$

Data points may be stored. The fourth chip is reserved for the storage of information about the valid data entries. This information includes: the total number of valid sets, the total number of valid entries in each set, the starting date, starting time, the constant multiplier factor and the sampling rates at which each data set was acquired.

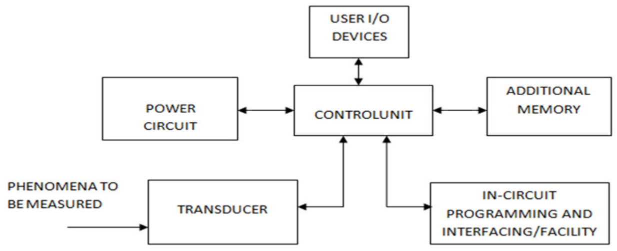


Figure 1: Block diagram of data logger

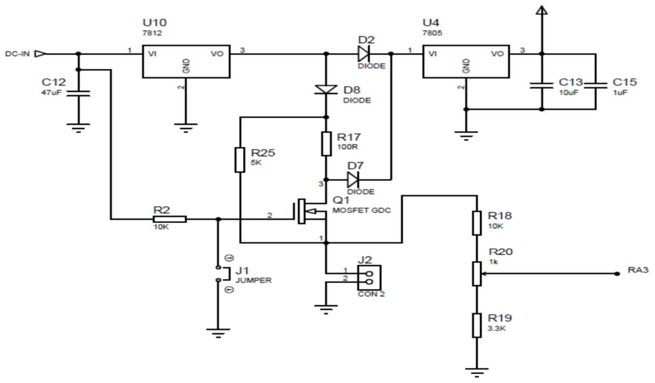


Figure 2: Power Supply Schematic

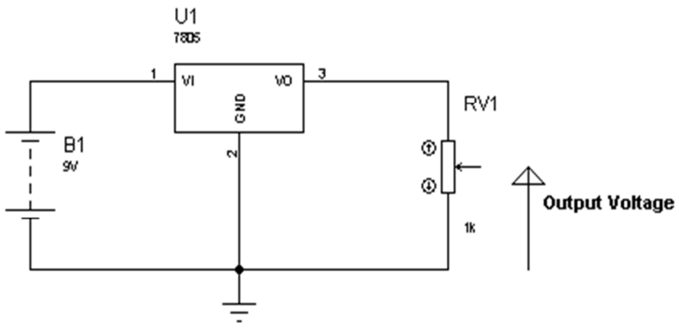


Figure 3: Transducer test Schematic

2.5 IN-Circuit programming/interfacing facilities

This unit consists of the hardware necessary to interface the device to a PC for reprogramming (or upgrading) the firmware and retrieving the stored data.

Communication with a PC for the purpose of data retrieval was achieved using the Universal Asynchronous Receiver/ Transmitter (UART) methodology. Since asynchronous communication does not require a clock signal, the serial link was realized using only two data lines of the PC's RS-232 serial port – one for transmission and the other for receiving of data.

The hardware employed to achieve the RS-232 serial interface was a 9-pin D-shell serial connector and a voltage level translator chip (Maxim's MAX232.) The connector simply allows the device to be coupled to a serial port of a PC using a standard serial cable. The MAX232 level translator provides the necessary conversion of the voltage signal levels assigned to logic high and logic low in the PC's RS-232 hardware (-12V and +12V, respectively) to those recognized by the MCU (+5V and 0V, respectively) and vice versa.

Microchip's PIC microcontrollers provide a simple serial programming interface that allows them to be programmed while in the application circuit. The In-Circuit Serial Programming (ICSP) feature of the MCU was made use of by furnishing a 6-pin header connector (J7) that connects to the circuit as shown in Figure 4. The prototype device was consequently programmed using PICkit2 from Microchip.

2.6 The control unit

The function of the control unit includes the maintenance of a real-time clock (RTC) and calendar in-code, ADC, Serial Communication and short term memory storage of sampled measurements. The heart of the control unit is the MCU – the PIC18F4520 from Microchip. The MCU derives its main clocking signal from a 4MHz crystal since an instruction cycle is equivalent to 4 clock cycles, the MCU operates at 1 MIPS (million instructions per second.)A second 32.768 kHz crystal provides the clock cycle that is used to maintain the RTC.In addition to the internal functions mentioned above, the MCU ensures the coordination and control of the system's peripheral devices that include the liquid crystal display (LCD), the light emitting diodes (LED) and the user controlled switches. The MCU also communicates with the EEPROM chips used for memory storage and the serial port of the PC during data retrieval.

The main features of the PIC18F4520 that led to its selection for the prototype are:

- The availability of a 10-bit ADC module that was used for ADC.
- The availability of 4 auxiliary timer modules (timers 0-3) one of which was used for the RTC.
- The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module used for serial communication with the PC, and
- The Master Synchronous Serial Port (MSSP) module that may be operated in Inter-Integrated Circuit (I^2C) mode for communication with the EEPROM.

2.6.1 The RTC and calendar

The TIMER1 module of the PIC18F4520 was configured such that its 16-bit register is incremented on the rising edge of the external clock signal applied to its input pin (pin 16.) With the interrupt on overflow feature of the timer enabled, an interrupt flag is raised whenever the counter overflows. Whenever an interrupt occurs, the program completes the execution of the present instruction then calls upon the ISR. When the ISR is completed the program resumes execution of the main code from where it was interrupted. Since the TIMER1 clock signal is derived from a 32.768 kHz clock then the period of the clock signal is:

$$T = \frac{1}{f} = \frac{1}{32.768kHz} \cong 0.0305\,ms \qquad (2)$$

Hence, the timer's counter will overflow every

$$2^{16} \times 0.0305ms = 2\,s \qquad (3)$$

By preloading the counter's register with a value of 32768 (or 2^{15}) that is by setting only the MSB of the counter then the timer is made to overflow every second. The Interrupt service routine (ISR) that updates the RTC and calendar is represented by Figure 5.

The ISR initially disables all interrupts. It then identifies the cause of the interrupt. In the firmware of the prototype device, the only enabled interrupt is that of timer1. Upon confirming that TIMER1 has overflowed, the interrupt flag is reset and the timer's registers are preloaded. The seconds counter is then incremented. If the seconds counter is 60 then it is reset and the minutes counter is incremented and so on. A mechanism that detects whether the present year is a leap year ensures that the calendar remains accurate.

2.6.2 The ADC

The function of the ADC is to calculate a binary value that is equivalent to its input (analog) voltage. The in-built 10-bit ADC of the PIC18F4520 converts the analog input to a digital output by successive approximation. For an n-bit ADC, the maximum output value is ($2^n - 1$). For the prototype, the 8 MSBs of the ADC are stored whereas the 2 LSBs are discarded. Hence, the output of the ADC varies from 0 to ($2^8 - 1$) or 255. The reference voltages of the ADC are Vss and Vdd (0V and 5V). Therefore, the input of the ADC may range between 0 to 5V. The resolution of the ADC, that is the minimum change in the input that may be detected, is calculated to be:

$$R = \frac{Range}{2^n} = \frac{5}{2^8} = \frac{5}{256} \cong 0.0195V \qquad (4)$$

In order to make use of MCU's in-built ADC, the flow diagram in figure 6 was implemented in the firmware. Initially, the ADC is configured and enabled. Then the input channel (pin1 or analog channel 0) is selected as the input source. A sample of the analog signal is then captured using a sample and hold circuit. A-D conversion begins after a delay that ensures that the sample and hold circuit has captured an accurate input value. Finally, the data may be read after the ADC has completed conversion process

2.7 The I²C link to the memory bank

Two-wire communication with the memory bank is made possible using inter – inter-computer communications protocol (I^2C). I^2C was developed by Phillips in the seventies to provide an interface between microcontrollers and peripheral devices that did not require a wired address bus as well as control and data buses [6].

The I^2C communication bus consists of a clock line (SCL) and a serial data line (SDA.) The SCL and SDA lines are both pulled up by external resistors to allow them to be driven by multiple devices. In the prototype data logger, a single master – the MCU – communicates with the memory bank that consists of four slave EEPROM devices having unique hardware addresses. The MCU initiates communication by sending a start signal down the bus – simply by first pulling down the SDA line and then pulling down the SCL line as well. The data to be transmitted is then sent serially to or from the slave with the negative edge of the clock signal being used to latch in the data. At the end of transmission, a stop signal is sent down the line. The stop signal consists of allowing the SCL line to go high followed by allowing the SDA line to go high as well.

The command that is sent by the master to the slave is usually of the format given in Figure 6.

.

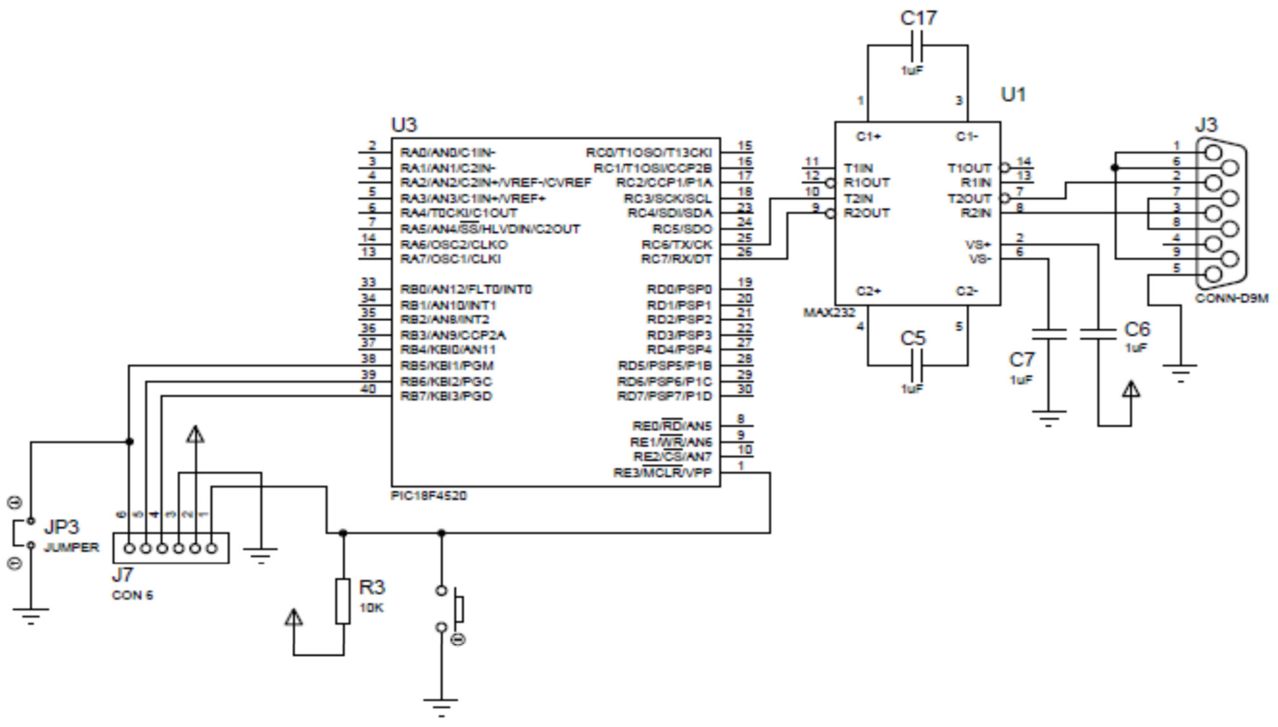
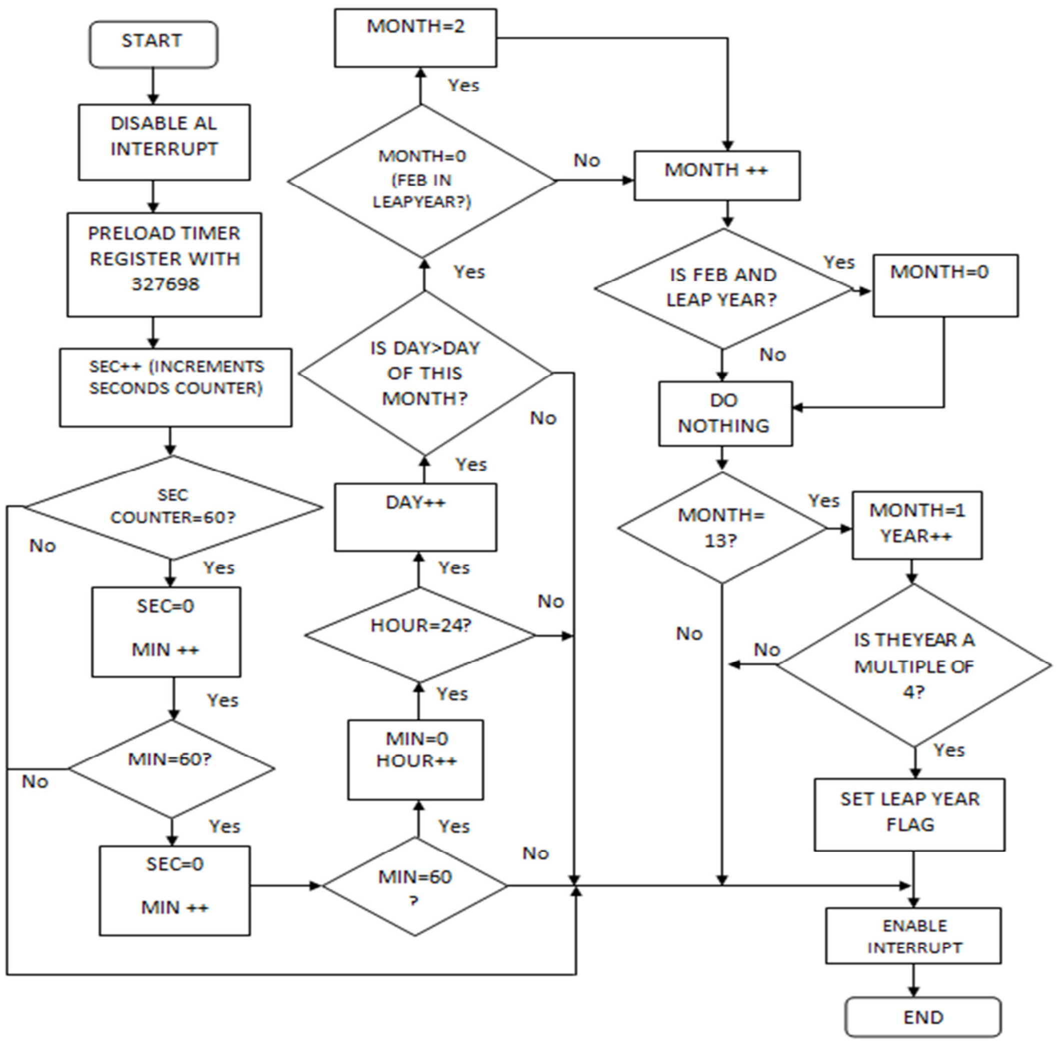


Figure 4: ICSP and Serial Port interface Schematic



1

Figure 5: Flow diagram for RTC and Calendar

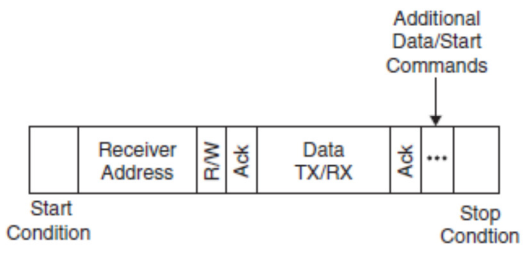


Figure 6: Command format for I²C communication.

The data sheet of the AT24C256 Serial EEPROM [7] chips chosen for this work offers instructions for the addressing, reading, and writing operations. Since the chip has a size of 256 Kbits, which is equal to:

$$256 \times 1024 = 262144\ bits$$
$$262144 \div 8 = 32768\ bytes$$
$$log_2\ 32768 = 15\ bits$$

Therefore, 15bits are required to uniquely address each byte slot in the memory table. The address is thus sent as two words separated by an acknowledgement. Then data to be written at the desired address is sent followed by an acknowledgement from the EEPROM chip and finally by a stop condition.

2.8 The UART communication to PC

The communication between the PC and the Logger was enhanced by the EUSART module in the microcontroller. The microcontroller Pins used for serial communication are RC6 (TXD) and RC7 (RXD). The microcontroller was configured to operate asynchronously this method of EUSART communication does not require a clock signal for synchronization. At an RS-232 data output (TX) on The Personal Computer side, logic 0 is defined as equal to or more positive than +5V, and logic 1 is defined as equal to as or more negative than -5V. In other words, the data uses negative logic, where the more positive voltage is logic 0 and the more negative voltage is logic 1 [8]. Prior to the differences in voltage level, MAX232 was used to interface the PC to the microcontroller. The EUSART module has a transmitter and Receiver buffer. When to transmit, data to be transmitted are loaded into the transmitter buffer (TXREG register). Control electronics 'pushes' data toward the TX pin in synchronization with internal clock. To receive data, when a START bit is detected, data is transferred to the receiver shift register RSR through the RX pin. When the STOP bit has been received, the data is transferred to the receiver buffer (RCREG register) [9].

2.9 The circuit diagram of the data logger

The complete circuit diagram of a General Purpose Data Logger is shown in Figure 7.

3. IMPLEMENTATION

The power supply was design to accomplish the logger's functionality to switch between external power and battery using a MOSFET as the switch, and also to give a constant 5VDC power to the circuit. On power-on of the logger the firmware initializes the LCD, ADC, EUSART and the I2C module. From the circuit diagram depicted in Figure 10, the four (4) channel sensor port (J10) for transducer inputs were connected to the RA0 to RA2 and RA4 of the microcontroller. During ADC initialization, pins (RA0, RA1, RA2 and RA4) are configured to operate as analog input to the device. On the arrival of analog signal on any of the analog inputs, the internal ADC of the Microcontroller samples and produces an equivalent digital representation of the signal's amplitude (in voltage). Special firmware routine processes the acquired ADC value and stores it in the serial EEPROM bank over the I^{2}C bus interface. Read ADC values are also displayed on the 16x2-LCD, which gives a visual presentation of the read value. The implemented Real Time Clock (RTC) facilitates comprehensive attributes to when the read ADC value is being sampled. Special routine ensures a proper storage of each ADC samples and the time the sampling tool place. Logged data are somewhat meaningful on an interface or platform where the data can easily be analyzed and produce more information
To achieve this, the logger implemented a serial communication with a Personal Computer (PC) using the EUSART module of the microcontroller. RC6 and RC7 pins of the microcontroller are Transmitter and Receiver pins respectively..

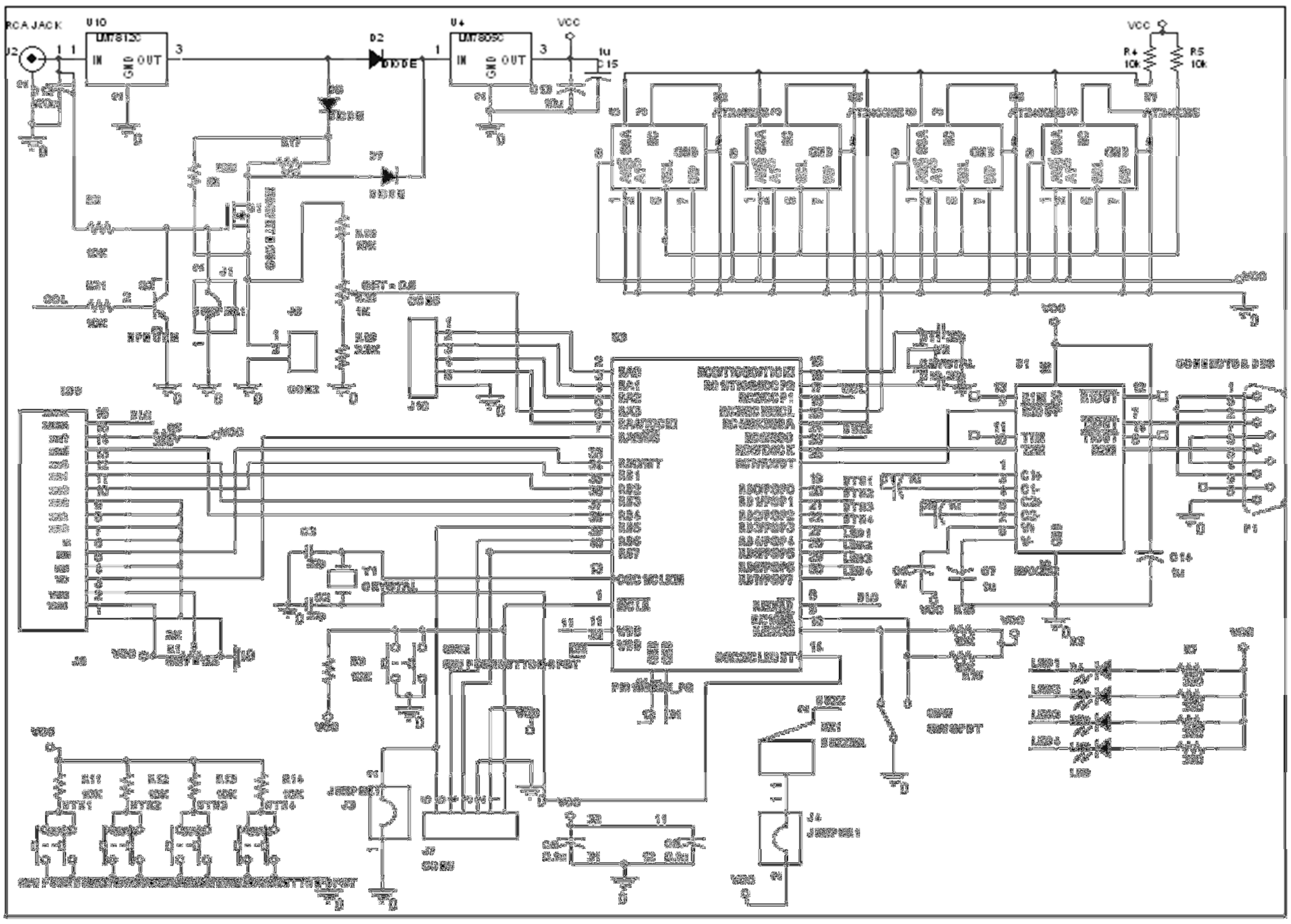


Figure 7: Circuit Diagram of General Purpose Data Logger

RC6 was connected to T2IN of MAX232 and RC7 was connected to R2OUT of the MAX232. The MAX232 is a voltage level translator between the voltage levels that exist as "0" or "1" on the PC side and the microcontroller. C17 and C5 were used to bias the MAX232 for a successful voltage level translationWhen user switches to interface communication mode, commands are executed to start transmitting all logged data that are the digital representation of the measured analog transducer values to PC over a serial port cable.

4. RESULTS AND ANALYSIS

The desired functions of the data logger were first achieved by simulation in Proteus. The results achieved in the simulation were consequently reproduced in the circuit, initially on bread board and then on the PCB as depicted in Figures 8 and 10 respectively. The analog input used for the experiments was the output of a simple voltage divider potentiometer circuit that was varied between 0 – 5V. The bread board circuit (Figure 8) as well as the Printed circuit board (Figure 10) both carry out the functions of maintaining a real time clock, performing analog to digital conversion of the measured signal, displaying and recording the measured samples as well as interfacing to a personal computer (PC) for the purpose of data retrieval and In – Circuit Serial Programming (ICSP). The functionality of the ADC, RTC and LCD were easily verified by simply observing the results and comparing with those obtained in the simulation. The I^2C communication to the Serial EEPROM chips was verified by manually removing the ICs from the bread board (and consequently from their sockets on the PCB) and placing them in the TOP2049 programmer. The programmer was then used to 'read out' the values stored during the experiment. The values agreed with the input value and the set parameters. The retrieved information, that is the logged data, is retrieved to a Personal Computer (PC) serially via RS-232 communication port. The USART terminal of MikroC® compiler IDE was used for the process of data retrieval.

The USART terminal was used to select the COM port to which the device is connected and to select the appropriate Baud rate (9600 bps in this case) and communication parameters. The device is then switched to interface mode via the toggle switch. A captured sample of the data retrieval process is shown in Figure 9. The values are fed out serially to the terminal. The values that are fed to the PC may then be stored and further post processing may be carried out in the way of plotting graphs and deducing statistics. The prototype was then further tested with a USB-to-serial converter with no compromise in functionality.

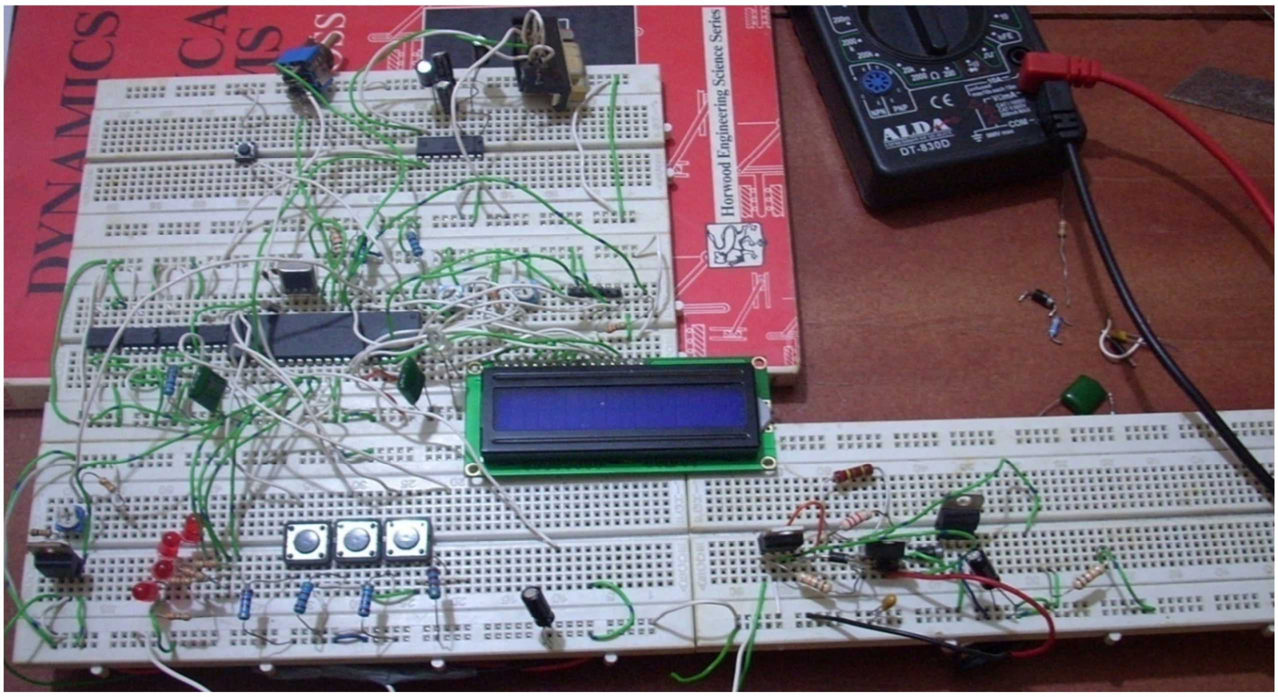
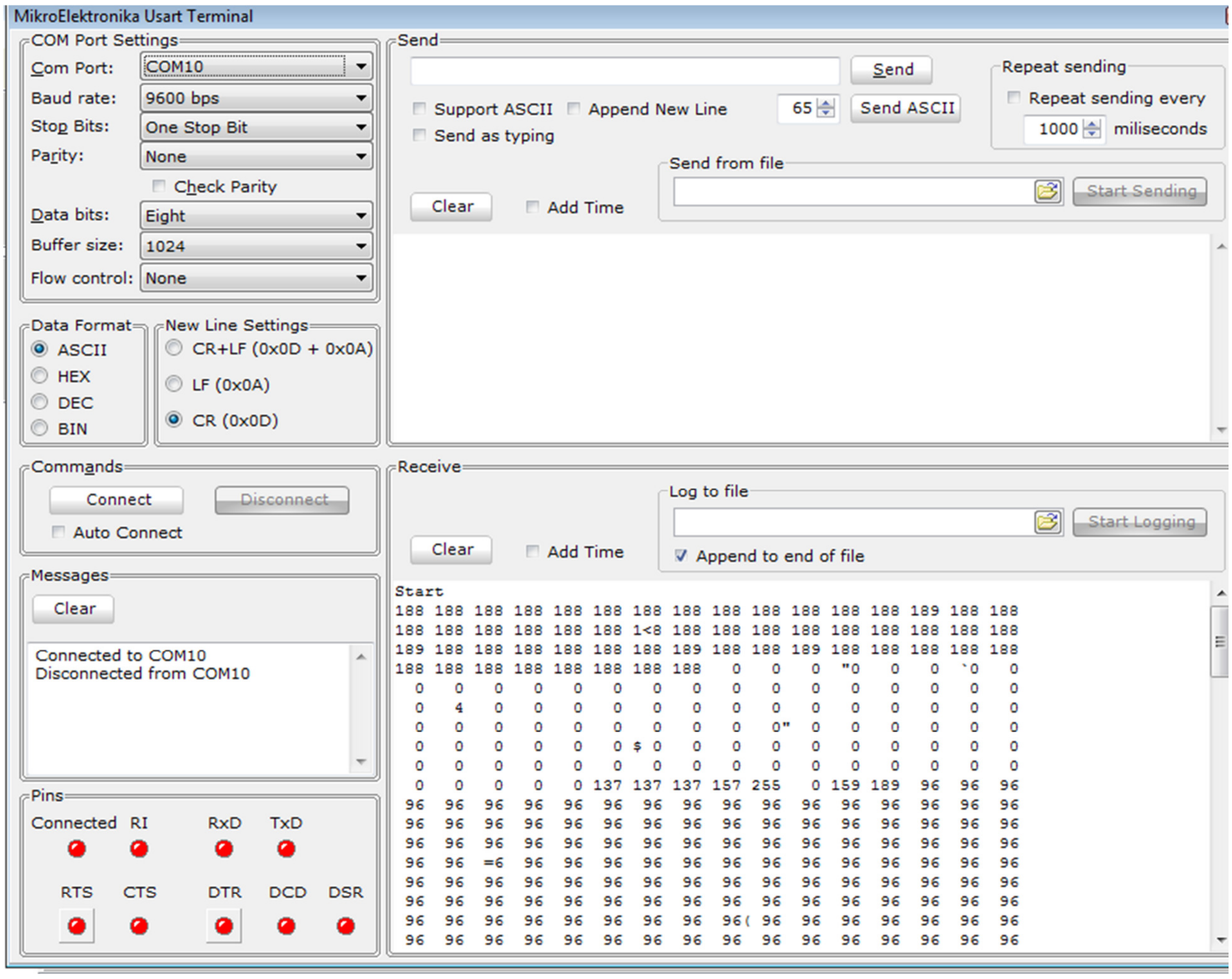


Figure 8: The Bread boarded circuit



Com Port: COM10
Baud rate: 9600 bps
Stop Bits: One Stop Bit
Parity: None
Check Parity
Data bits: Eight
Buffer size: 1024
Flow control: None

Data Format: [✓ ASCII] HEX DEC BIN

New Line Settings: CR+LF (0x0D + 0x0A) LF (0x0A) [✓ CR (0x0D)]

Support ASCII · Append New Line · Send as typing · 65 · Repeat sending every 1000 miliseconds · Add Time · Auto Connect · [✓ Append to end of file]

```
Connected to COM10
Disconnected from COM10
```

```
Start
188 188 188 188 188 188 188 188 188 188 188 188 188 189 188 188
188 188 188 188 188 188 1<8 188 188 188 188 188 188 188 188 188
189 188 188 188 188 188 188 189 188 188 189 188 188 188 188 188
188 188 188 188 188 188 188 188   0   0   0  "0   0   0  `0   0
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  0   4   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0  0"   0   0   0   0   0   0
  0   0   0   0   0   0 $ 0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0 137 137 137 157 255   0 159 189  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  =6  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96( 96  96  96  96  96  96  96
 96  96  96  96  96  96  96  96  96  96  96  96  96  96  96  96
```

Figure 9: The Recorded data retrieved to a PC

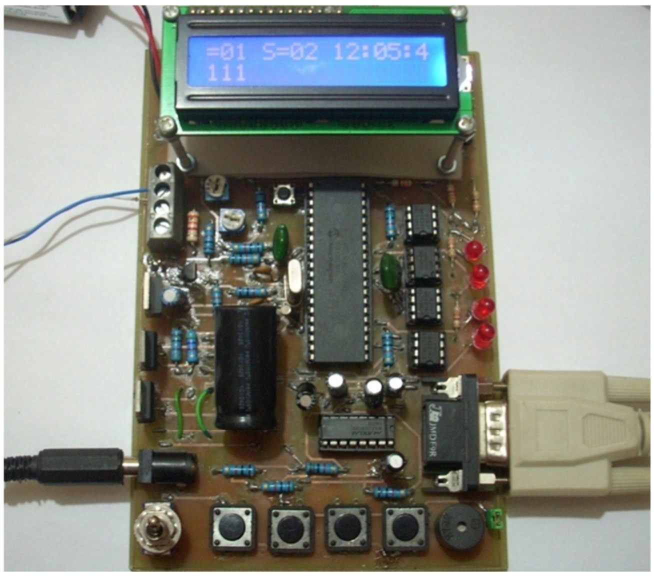


Figure 9: The Data logger PCB prototype

5. CONCLUSION AND RECOMMENDATION

This work has shown that it is possible to build a prototype data logger centered on a single Micro Controller Unit (MCU.) The construction of the prototype has also further shown that it is possible to meet the technical specifications set out in the design problem while remaining within the desired budget.

The objectives set out in the introductory chapter have been met by the prototype design. The selection of a versatile yet low power MCU to implement most of the crucial functions of the data logger has proven to be cost effective in terms of both power consumption and price.

It is recommended that this design be adopted and used for variety of field works such as a weather station.

Optimization of power consumption may be further promoted by implementing a few conservation practices such as automatic backlight control and employing the power-managed modes supported by the PIC18F4520 when possible.

The incorporation of a multimedia card or compact flash card for the storage of data or possibly a USB host environment that would allow the connection of a USB mass storage device instead of the present EEPROM chips seems a viable idea that would greatly enhance the storage capabilities of the data logger.

And finally, a graphical display may replace the alphanumeric LCD provided in the prototype allowing for greater scope in online analysis of the logged data and possibly presenting a historical trace of logged data as opposed to reporting only present measurements.

6. REFERENCES

[1] S. S Badhiye, Dr. P. N. Chatur, B.V. Wakode "Data logger system: A Survey" *International of Computer Technology and Electronics Engineering (IJCTEE),* Vol4 Issue 6, pp 24 - 26 Feb 2012, ISSN 2249-6343.

[2] R. H. Walden, "Analog to Digital Converter Survey and Analysis", *IEEE Journals on Selected Areas on Communication*, Vol 17 No 4 pp 539 - 550, July 16 1999.

[3] M Harnendez, V. Rybnikov, F. Sanchez, "Offline mass data processing using online computing resources at HERA-B", *Nuclear Instruments and methods in Physics research.* A 502 (2003) pp 471 -474.

[4] Microchip Technology Inc., "PIC18F2420/ 2520/ 4420/ 4520 Data Sheet", 2004, www.microchip.com

[5] R. R. Dedrick, J. D. Halfman, D. B. Mckinney, "An inexpensive microprocessor based data logging system". *Journal of Computers and Geoscience,* Vol 26 issue 9-10, pp 1059 – 1068.

[6] M. Predko, *"Programming and Customizing the PIC Microcontroller",* 3rd edition. New York: McGraw-Hill, 2008.

[7] Atmel Corporation, "Two-wire Serial EEPROMs", 2007 www.atmel.com

[8] J. Axelson, "Serial Port Complete", 2007, Lakeview Research LLC,ISBN 978-1931448-07-9

[9] MikroElektronika, "PIC Microcontrollers - Programming in C"*,* www.mikroe.com