Caliphate Journal of Science & Technology (CaJoST)

ISSN: 2705-313X (PRINT); 2705-3121 (ONLINE)

Research Article

Article Info

Received: 17th December 2018

Revised: 22nd February 2019

Accepted: 26th February 2019

Design and Implementation of a PC-based Temperature Data Logging System with Graphics User Interface

Sunday U. Ezeilo[1]*, Muhammad B. Abdullahi[2], Muhammad B. Abubakar[3] and Ibrahim Z. Yaroko [4]

[1] Department of Electrical and Electronic Engineering, Nigerian Defence Academy, Kaduna
[2]Department of Physics, Usmanu Danfodiyo University, Sokoto
[3]College of Science and Technology, Umaru Ali Shinkafi Polytechnic, Sokoto
[4]Department of Physics, Sokoto State University, Sokoto

Corresponding author's email: ezeilo.su@gmail.com

Cite this: *CaJoST,* 2019, 1, 38-43

Data acquisition systems have become inevitable tools for analyses in science and technology. With advancement in research and manufacturing processes, there is a growing demand for sophisticated yet cost-effective data logging systems to address the complexity in data acquisition and analyses. This work addresses the temperature data acquisition needs in research and manufacturing by implementing an embedded system suitable for monitoring and analyzing surrounding temperatures at configurable logging intervals. It uses a high precision temperature sensor to capture the ambient temperature. The captured data is processed on-chip and sent to a host Personal Computer (PC). Data logging is managed by a Graphics User Interface (GUI) application developed to run on the host PC. The application provides user-selectable logging interval, and an off-board database. The records are saved with the '.CSV' file extension, which is compatible with Microsoft Excel spreadsheet, enabling data analyses using built-in statistical methods integrated into spreadsheet applications.

Keywords: Data logger, Temperature, Microcontroller, Simulation, Graphics user interface

1. Introduction

Data logging is the measuring and recording of physical or electrical parameters over a period of time. Data loggers are used in a variety of applications such as in-vehicle data logging, environmental monitoring, structural health monitoring, and machine condition monitoring. Data loggers can either be PC-based, stand alone or both. PC-based data loggers offer more than just the basic PC connectivity of traditional stand-alone data loggers. With a PC-based data logger, the PC is part of the system, so the data logger can take advantage of the ever-increasing performance of a PC's processor, hard drive, display, and peripheral bus. These capabilities introduce several advantages over traditional data-logging methods [1].

One of the many data loggers developed by researchers is inexpensive microprocessor-based data logging system [2], which employed on-board EEPROM (Electrically Erasable Programmable Read-Only Memory) for logging. The stored data could be retrieved on demand via the host PC. Other works include Data Logger and Remote Monitoring System for Multiple Parameter Measurement Applications [3], incorporating both temperature and humidity logging. Of more interest is the utilization of SD (Secure Digital) card in this system, which offers high density memory capacity for data logging. Another system of this type is a Microcontroller-Based Temperature Monitoring and Logging System Suitable for Use in Hospitals [4]. It was implemented as a stand-alone system, using an on-board memory chip for data storage, and allowed user to retrieve data via an on-board LCD (liquid crystal display). It also provided for recorded data to be exported to the PC via the PC's serial interface. Design and development of a PC-based automated data logging system for measuring temperature was done by [5].

It included an optional on-board display device. This system used LabVIEW application as the third-party software to manage data records. One problem with the mentioned designs is that they have limited storage capacity as they employ on-board ROM for logging data. Data exporting is also an issue [5] as it can only be done manually which is not very convenient. The designs of [4] is bulky and requires a separate power source.

These shortcomings are addressed in the design described in this paper.

Thus, the aim of the present study is to design and implement a PC-based temperature data logging system with graphics user interface. The features include development of a Windows® database for data loggers as to make record management more reliable and interactive. It also provides a graphics user interface (GUI) which allows for user-selectable logging intervals. And finally, it adds more interactive features to the existing data loggers

2. Methodology

The development of this system comprises of circuit design algorithm, embedded software programming, Graphics User Interface (GUI) and hardware implementation. The block diagram in Figure 1 describes the system's operation.

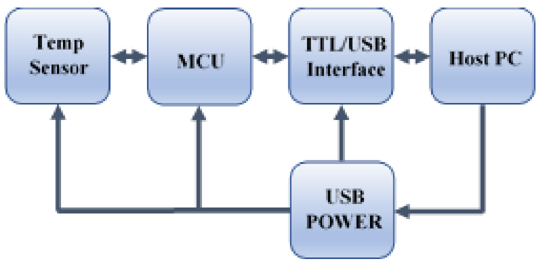


Figure 1. Design Block Diagram

2.1 Operation of the System

The Temperature Data Logger system consists of five functional units as shown in Figure 1. The temperature to be measured is sensed by the sensor unit and converted to a corresponding voltage signal. The Microcontroller Unit (MCU) sends a "Start Conversion" command to temperature sensor at regular intervals of time. The sensor then completes the conversion within a period of 187.5 milliseconds (max) and transmits the digital data to the MCU via "One-Wire" data interface. The MCU processes the received data as required and transfers the information to the host PC. Between the PC and the MCU is a serial communication interface in 'asynchronous' mode. The TTL/USB interface converts the TTL logic from the MCU to USB (Universal Serial Bus) logic compatible with the PC, and vice-versa. The host PC runs a GUI application that manages the data that the PC receives from the logger hardware. The PC also provides a regulated 5-volt dc supply to the logger hardware via the USB port.

The design of the Temperature data logger is in two phases; the hardware and software.

2.2 Hardware Design

The hardware design involves the design of the circuitry in each of the functional blocks (Figure 1). The circuit diagram of this system is shown in Figure 2, as captured from Proteus schematic editor.

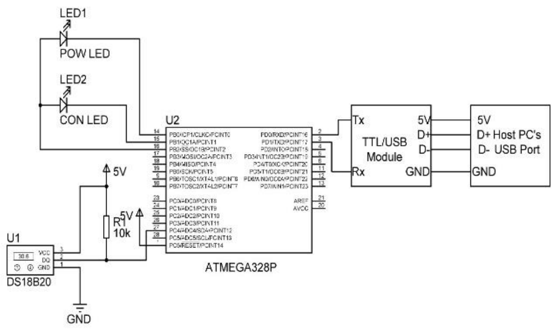


Figure 2. The Designed Cicuit Diagram

In the circuit digram (Figure 2), the temperature sensor, U1 (DS18B20) is a "1-Wire" digital thermomether configured in 10-bit mode for 0.25 ^{o}C step of conversion. R1 is a pull-up resistor for the 1-Wire data bus and has a value of 10 kΩ as suggested by the DS18B20's datasheet [6]. Block U2 (ATMEGA328P) is an 8-bit Microcontroller [7] used to establish data acquisition between the host PC and the temperature sensor. The TTL/USB module is an interface hardware used for MCU/PC communication. LED1 and LED2 are green and blue light emitting diodes respectively, for user interraction. LED1 blinks every 4 seconds to indicate that the device is powered, and remains OFF when not powered or connected and sending data to the host PC. LED2 remains ON as long as the device is sending data to the PC.

2.3 The Software Development

The software development for the system involves development of firmware for the microcontroller and the host PC. This section is reported in two parts as MCU Programming and GUI Development.

2.4 The MCU Programming

The program for the MCU was written in plane C-Language and compiled in integrated development environment (ICCAVR IDE) [8]. The Flowchart shown in Figure 3 describes the program design for the MCU to connect the logger hardware with the application on the host PC. It is designed to get data from the sensor, process them and send the result to the PC.

The MCU's built-in USART (Universl Synchronous-Asynchronous Receiver-

Transmitter) hardware was configured in asynchronous serial mode to implement communication between the PC and the MCU. The baud rate of 19200 was used for the communication as it offers moderate communication speed. The baud rate register value is calculated as follows:

$$\mathrm{UBRR0} = \frac{\mathrm{F_{osc}}}{16 \times \mathrm{BAUD}} - 1 \qquad (1)\ [7]$$

Where:

UBRR0 = USART baud rate register for USART0 (USART module-0);

F_{OSC} = frequency of the system clock (Hertz) = 8000000 (8MHz);

BAUD = required baud rate (bits per second) = 19200.

Therefore, $\mathrm{UBRR0} \cong 25$

Hence, from equation 1:

$$Baud\ Rate_{closest\ match} = \frac{f_{osc}}{16 \times UBRR0 + 1}$$

$$= 19231\ bps\ (bits\ per\ second).$$

Percentage error in the baud rate is also obtained as follows:

$$\mathrm{Error}\ (\%)$$

$$= \frac{\mathrm{Baud\ Rate_{closest\ match}} - \mathrm{Baud\ Rate}}{\mathrm{Baud\ Rate}} \times 100 \quad (2)$$

$$= \left(\frac{\mathrm{Baud\ Rate_{closest\ match}}}{\mathrm{Baud\ Rate}} - 1\right) \times 100 \cong 0.2$$

0.2% error is within the tolerance of 2% (max) recommended in the manufacturer's datasheet.

The DS18B20 temperature sensor uses a strict 1-Wire communication protocol to ensure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. The bus master initiates all these signals, with the exception of the presence pulse [6]. Therefore, transfer rate is computed as follows:

$$\mathrm{T} = \mathrm{T_{(bit)}} + \mathrm{T_{Bus}} \qquad (3)$$

Where

$\mathrm{T_{bit}} = 95\ \mu s$ = time for 1 bit; and $\mathrm{T_{Bus}} = 80\ \mu s$ = required Bus recovery time

$$T = 95\ \mu s + 80\ \mu s = 175\ us.$$

Hence, $\text{bit rate} = \frac{1}{\mathrm{T}} \cong 5.7\ \mathrm{kbps}$

The system clock frequency must be chosen above the calculated bit rate of 5.7 kbps. This was accommodated in the implementation, given the clock frequency, f_{CPU} = 8 MHz.

The MCU's built-in 16-bit timer module, timer1 was configured to run at 1 µs time base throughout the operation. Calculations for the timer configuration are as follows [7]:

$$\mathrm{f_{timer}} = \frac{\mathrm{f_{clock}}}{\mathrm{Prescaler}} \qquad (4)$$

where:

$\mathrm{f_{timer}}$ = frequency of the timer (Hertz),

$\mathrm{f_{clock}}$ = frequency of the system clock (Hertz) = 8000000

Prescaler = clock division factor = 8

Therefore, $\mathrm{f_{timer}} = \frac{8000000}{8} = 1000000\ \mathrm{Hz}$

$$= 1\mathrm{MHz}.$$

Hence, the period of the timer,

$$\mathrm{T_{timer}} = \frac{1}{\mathrm{f_{timer}}} = 1\ \mu s.$$

This time base was used to generate other integrals of a micro second throughout the code as shown in the flow chart of Figure 3.

2.5 Graphic User Interface Development

The GUI for this system was developed on Microsoft '.NET' platform for Windows OS (operating system) using Microsoft Visual C# (C-sharp) programming language.

2.6 Software Simulation

The design was pre-tested using simulation software. This involved simulations at different stages of the programming. Simulations helped to debug the code at different stages, even before the hardware production. Proteus Design Suite application was used for this task. Figure 4 shows a screen shot of the simulation in the Proteus.

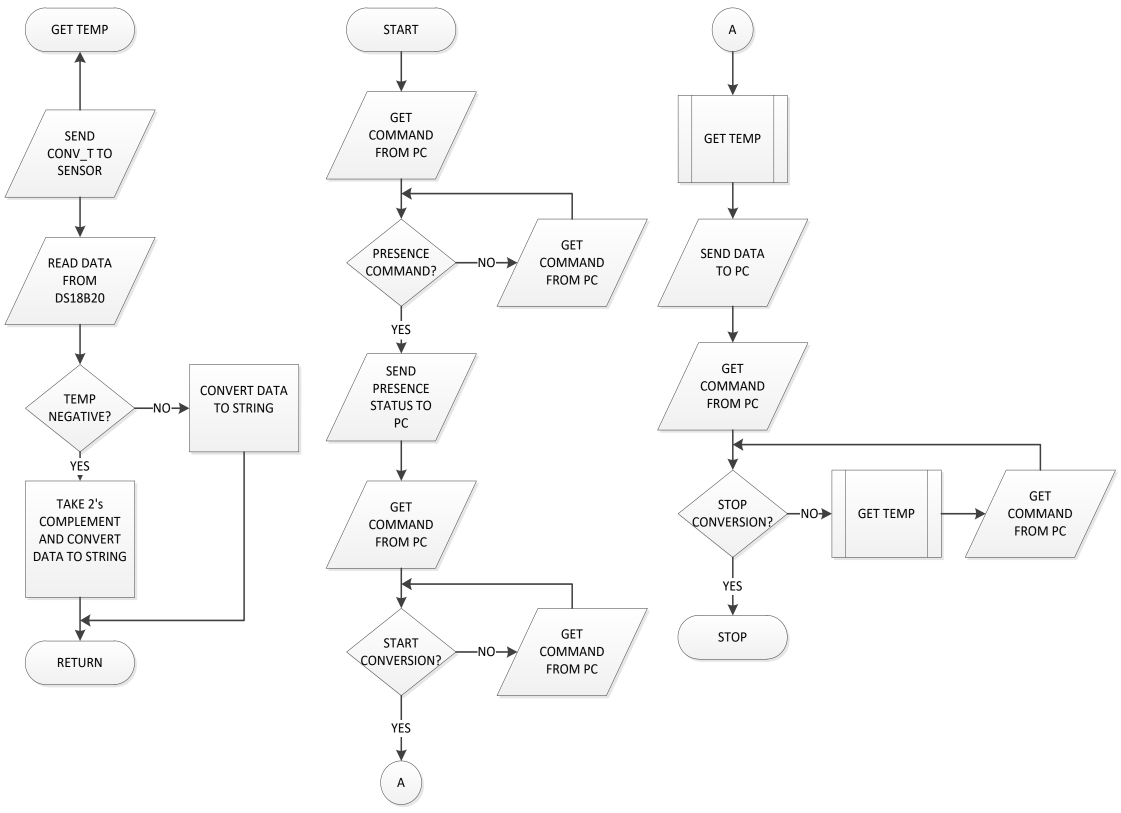


Figure 3. The Microcontroller Program Flow Chart

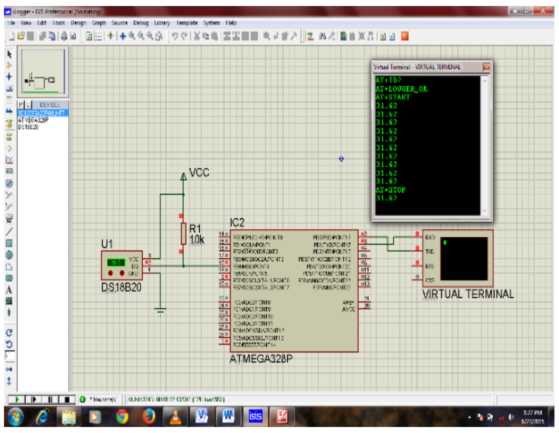

```
AT+ID?
AT+LOGGER_OK
AT+START
31.62
31.62
31.62
31.62
31.62
31.62
31.62
31.62
31.62
31.62
AT+STOP
31.62
```

Figure 4. Simulation Screenshot

2.7 The Device Hardware Construction

Hardware construction involves wiring and packaging of the designed circuit. This requires arrangement and soldering of various components and sub-circuits that make up the system.

2.8 PCB Design and Fabrication

The PCB layout was designed and routed in Proteus Design Suite, producing the circuit schematic, PCB layout, and components' layout required for soldering. A snapshot of the PCB layout is shown in Figure 5a. The layout was transferred onto a single-sided Copper board by means of "hot Iron" heat transfer method. A chemical etching process was used to obtain the required tracks on the board. The chemical (etchant) used for this process was Ferric (Iron III) Chloride with warm water. The holes for components were drilled after etching using a hand drill with drill bits of suitable sizes. Figure 5 (b) and (c) picture the copper and components sides of the circuit board respectively.

2.9 Packaging of the Finished Circuit Board

This is the final stage of the construction process. After the circuit was implemented, it was cased in suitable housing. It was enclosed in a portable and suitably-sized already-made plastic case. The casing made provision for the USB connection to the PC. A picture of the cased packaged hardware is shown in Figure 5d.

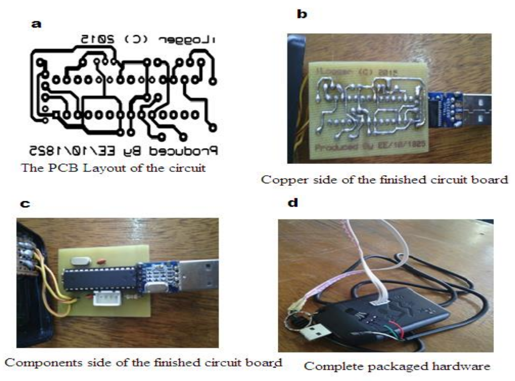

a
The PCB Layout of the circuit
b
Copper side of the finished circuit board
c
Components side of the finished circuit board
d
Complete packaged hardware

Figure 5. Hardware Production

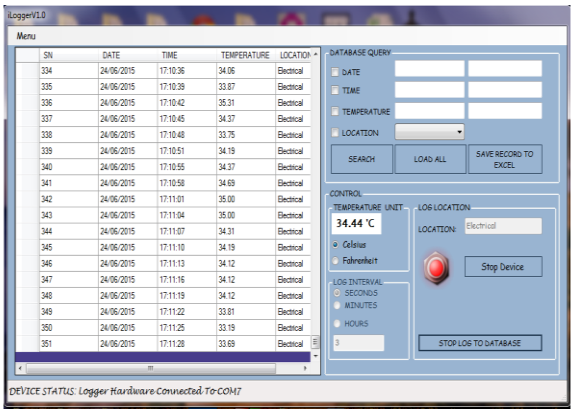
3. Tests, Results and Discussion

3.1 Tests

Performance evaluation was carried out after the implementation of the whole system. The finished hardware was connected to the PC via a USB port. It was switched on to test if it was working. For the purpose of reliability, further tests were conducted by leaving it ON over a period of time and at different locations, while monitoring the performance as it obtained data. The temperature as measured by the constructed device was used to compare the equivalent temperature readings taken by a standard thermometer under the same conditions.

3.2 Result of the Tests

The system was then used to retrieve the logged record and saved with ".CSV" (comma-separated values) file extension then accessed in a spreadsheet Microsoft Excel and SPSS applications. A snapshot of the GUI with data logging in progress is shown in Figure 6.



SN	DATE	TIME	TEMPERATURE	LOCATION
334	24/06/2015	17:10:36	34.06	Electrical
335	24/06/2015	17:10:39	33.87	Electrical
336	24/06/2015	17:10:42	35.31	Electrical
337	24/06/2015	17:10:45	34.37	Electrical
338	24/06/2015	17:10:48	33.75	Electrical
339	24/06/2015	17:10:51	34.19	Electrical
340	24/06/2015	17:10:55	34.37	Electrical
341	24/06/2015	17:10:58	34.69	Electrical
342	24/06/2015	17:11:01	35.00	Electrical
343	24/06/2015	17:11:04	35.00	Electrical
344	24/06/2015	17:11:07	34.31	Electrical
345	24/06/2015	17:11:10	34.19	Electrical
346	24/06/2015	17:11:13	34.12	Electrical
347	24/06/2015	17:11:16	34.12	Electrical
348	24/06/2015	17:11:19	34.12	Electrical
349	24/06/2015	17:11:22	33.81	Electrical
350	24/06/2015	17:11:25	33.19	Electrical
351	24/06/2015	17:11:28	33.69	Electrical

TEMPERATURE UNIT: 34.44 'C (Celsius)
LOG INTERVAL: SECONDS, 3
LOCATION: Electrical

DEVICE STATUS: Logger Hardware Connected To COM7

Figure 6. A snapshot of the GUI with data logging in progress

The results obtained during different tests are tabulated in Table 1. The system's error was then calculated using:

$$Average\ System\ Error = \frac{\sum_{i=1}^{n} \frac{Thermo\ Temp_i - Logger\ Temp_i}{Logger\ Temp_i}}{n} \qquad (5a)$$

$$Average\ \%\ System\ Error = Average\ System\ Error \times 100 \qquad (5b)$$

Where:

Logger Temp = temperature reading obtained from the Logger;

Thermo Temp = temperature reading from a standard thermometer;

i = i_{th} result and n = number of readings.

Table 1. Test Results

Time	TEST 1			TEST 2			TEST 3		
(Min)	Thermo-meter Temp (°C)	Logger Temp (°C)	% Error (%)	Thermo Temp (°C)	Logger Temp (°C)	% Error (%)	Thermo Temp (°C)	Logger Temp (°C)	% Error (%)
1	32.00	33.12	-3.5000	3.00	3.25	-8.3333	74.50	74.63	-0.1745
2	35.31	36.44	-3.2002	2.90	3.13	-7.9310	77.30	77.25	-0.0647
3	35.35	36.44	-3.0835	2.70	3.00	-11.1111	79.59	79.75	-0.2010
4	34.23	35.94	-4.9956	2.75	3.00	-9.0909	83.08	83.50	-0.5055
5	32.05	34.19	-6.6771	2.57	2.75	-7.0039	85.78	86.25	-0.5479
6	32.21	32.31	-0.3105	2.58	2.75	-6.5891	88.12	88.50	-0.4312
7	33.51	34.69	-3.5213	2.50	2.63	-5.2000	90.01	90.75	-0.8221
8	31.76	32.37	-1.9207	2.49	2.50	-0.4016	90.89	91.25	-0.3961
9	31.29	33.69	-7.6702	2.46	2.50	-1.6260	92.15	92.75	-0.6511
10	33.72	34.75	-3.0546	2.30	2.36	-2.6087	95.63	96.00	-0.3869
	Average System Error (%)		-3.7934			-5.9896			-0.4052

3.3 Discussion of Results

The results obtained after testing showed high precision and reliability as indicated in Table 1. This indicates that the system is suitable for temperature data logging. The magnitude of the average system error observed, in the table, is less than 6%, which complies with the accuracy of the temperature sensor used as specified in the device's datasheet as ±0.5 °C over the range of -10°C to +85°C [6].

The Logger GUI application has the flexibility in managing the data history such as querying by one or more fields. The result in Figure 6 shows that at any time, the user can query the database either by date range, time range, temperature range, location, or any combination of these fields. The result obtained showed that the system was also able to retrieve record saved with ".CSV" (comma-separated values) file extension. Compared to [2], [4], the design is an improvement as the use of PC's storage provides high logging memory density when compared to using on-board ROM which places significant limit to logging. Apart from this, there is an improvement in data management which the GUI provides due to the provision of several logging fields in the database that makes it easier to manage data history, such as database query. Finally, the GUI in this design allows for raw data export in CSV format which can be processed by data analysis tool such as Microsoft Excel, Matlab, SPSS, etc

4. Conclusion

In this work, we carried out the design and implementation of a PC-Based temperature data logger. The system was observed to be capable of performing the required task based on the satisfactory results obtained during tests, with tolerable errors. The application design was based on Microsoft Windows OS (Operating system). The test result showed that the system can measure and record temperatures with high reliability.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] National Instruments, 'Data Loggers,' , https://www.ni.com/data_logger/ , 2017

[2] R.D. Robert, D. H. John and D. M. Brooks, ‘An Inexpensive, Microprocessor-Based Data Logging System’, *Computers & Geosciences,* 26:1059-1066. 2000

[3] G. S. Nhivekar, and R.R Mudholker, ‘Data Logger and Remote Monitoring System for Multiple Parameter Measurement Applications’, *e -Journal of Science & Technology (e-JST)* pp. 55-62 2011.

[4] I. G. Saidu M. Momoh and A.S. Mindaudu, ‘Temperature monitoring and Logging System Suitable for Use in Hospitals, Incorporating GSM Text Messaging’, *International Journal of Information Sciences and Techniques (IJIST)* 3:11-26, 2013

[5] A. Al Mamun, K. Sundaraj, N. Ahmed, M. Rahman, and N. A. Uddin, Design and Development of a PC-Based Automated Data Logging System For Measuring Temperature, *Journal of Engineering and Applied Sciences* 8:960-968. 2013

[6] MAXIM Innovation Delivered Revolutionary iButton Digital Temperature and Humidity Data Loggers, pp. 6-7. 2011

[7] Atmel Corporation Atmel- microcontroller - ATmega328/Datasheet 2016

[8] Imagecraft, JumpStarter C for AVR – C Compiler for Atmel AVR, https://imagecraft.com/help/PDF/ICCAVR.pdf 2016.