CHAPTER 2

HARDWARE DETAILS OF INTEGRATED INSTRUMENTATION SYSTEM FOR
CAPACITANCE MEASUREMENT

The block diagram of the Microcontroller (ARM) – PC based Instrumentation system for the studies of frequency and temperature effect on capacitance measurement of materials and is shown in figure 2.1 and photograph 1.

![ Bloc Diagram of Microcontroller (ARM) – PC based Instrumentation system for studies of frequency and temperature effect on capacitance measurement. ]

Fig 2.1 : Block diagram of integrated Instrumentation system for the studies of frequency and temperature effect on capacitance measurement

The integrated instrumentation system for the capacitance measurement system consists of the following units. They are

1. Capacitance cell
2. Autobalance bridge
3. Programmable function generator (PFG)
4. Analog to digital converter using dual display multimeter
5. Temperature measurement & control unit
6. ARM processor
7. Serial Communication
8. Personal computer

The details of the hardware are presented below.

2.1 Capacitance cell

The capacitance cell consists of two circular discs (20 mm in diameter) of brass metal, whose faces are well machined and later polished with fine emery and separated by a small distance as shown in figure 2.2 (A) and 2.2 (B) front view and back view respectively. The two conducting plates are positioned parallel to each other so that one is fixed to base which is connected to a temperature measurement and control system and the other top plate is connected to a screw gauge. The top plate is separated by a cork which acts as insulation between the plate and the screw gauge and the length of leads is kept as small as possible (10 cm). These leads form interconnections between BNC, SRF-10 receptacle of 50 ohms impedance and the two conducting circular plates of the cell. With the help of screw gauge the thickness of the dielectric or the distance between the plates can be adjusted up to 10mm. The cell acts as a parallel plate capacitor with a solid being dielectric whose dielectric constant is to be measured. The entire arrangement is enclosed in a metal housing and is designed such that it is rigid which avoids capacitance variations due to environmental changes.
2.2 Autobalance bridge:

Figure 2.3 shows an autobalance bridge circuit, the current flowing through an unknown capacitor is equal to the current flowing through the feedback resistor used in the bridge circuit due to the fact the virtual ground is maintained at zero volts [9,10]. In the present study the polymer material is placed in the capacitance cell explained earlier and the properties of these material is studied in the 100 Hz to 1Mz frequency range and also from 40°C to 120°C. the two components magnitude and phase of the impedance are measured. Here a well known polymer HDPE is taken to start with as it is well referenced.
There are many references on autobalance bridge circuit for the measurement of impedance of basic electronic elements. J Prado, C Margo M Kouider, M Nadi [11] worked on this principle to evaluate a cell for dielectric properties measurement of biological material. James J. Ackmann [12] addressed two major limitations of two-terminal bridges; interfacial electrode impedance and impedances with high dissipation factor[13]. V. Dumbrava and L. Svilainis an impedance measurement system is presented, it explained the system capabilities of parallel RC circuit and transducer BVD electrical model. The study also stated that the performance of autobalance bridge method above 1MHz is very complicated [14].

2.3 Function generator

In the present work it is proposed to study the effect of frequency and temperature on the capacitance of the material. Hence a programmable function generator (PFC) is designed and developed as shown in figure 2.4 and it is interfaced with ARM processor.

The AD9834 is a low power direct digital synthesis (DDS) device capable of producing high performance sine and triangular outputs. The AD9834 has a standard serial interface that allows the part to interface directly with several microprocessors. The device uses an external serial clock to write the data/control information into the device. The serial clock can have a maximum frequency of 40 MHz maximum. The serial clock can be continuous, or it can idle high or low between write operations. When data/control information is being written to the AD9834, FSYNC is taken low and is held low until the 16 bits of data are written into the AD9834. The FSYNC signal frames the 16 bits of information being loaded into the AD9834. The AD9834 has a standard 3-wire
serial interface that is compatible with SPI, QSPI™, MICROWIRE™, and DSP interface standards. Data is loaded into the device as a 16-bit word under the control of a serial clock input (SCLK). The FSYNC input is a level triggered input that acts as a frame synchronization and chip enable. Data can only be transferred into the device when FSYNC is low. To start the serial data transfer, FSYNC should be taken low, observing the minimum FSYNC-to-SCLK falling edge setup time. After FSYNC goes low, serial data is shifted into the input shift register of the device on the falling edges of SCLK for 16 clock pulses. FSYNC can be taken high after the 16th falling edge of SCLK, observing the minimum SCLK falling edge to FSYNC rising edge time. Alternatively, FSYNC can be kept low for a multiple of 16 SCLK pulses and then brought high at the end of the data transfer. In this way, a continuous stream of 16-bit words can be loaded while FSYNC is held low, with FSYNC only going high after the

---

Fig 2.4 : Schematic diagram of 1MHz programmable function generator circuit
16th SCLK falling edge of the last word is loaded. The SCLK can be continuous, or alternatively, the SCLK can idle high or low between write operations but must be high when FSYNC goes low.

Three port pins of the microprocessor drives SCLK, SDATA and FSYNC of the AD9834. The clock, data and control signals are sent to AD9834 as explained in above paragraph so that it can be understandable and executed as per the user requirement. The performance of the designed programmable function generator is calibrated with protek 9305 function generator.

The wiring between RS232 interface and connection cable of computer:

![Diagram showing serial port connection details](Fig 2.5)

2.4 **Dual display multimeter**

In the present work the dual display was used as one of the key instrument for voltage measurement. The output of the Autobalance Bridge is a voltage signal in response to the material placed between the plates of capacitors. The change in output signal depends on many factors including an input voltage. In the present study the input voltage used is 1Vrms and this voltage is kept constant. For all other parameter variations, there is a need to measure the output voltage which is of the order of few millivolts or in some cases in microvolts. To accomplish this important requirement a 51/2 digit digital multimeter is used for measurement, display and to send the measured value to PC through serial port.
The salient features of the digital multimeter are given below

- A dual vacuum fluorescent display (VFD) that allows two properties of input signal to be displayed at the same time,
- Remote operation via RS-232 interface or IEEE-488 interface
- 1uV sensitivity in Vdc measurement,
- True RMS Vac with 20Hz to 100KHz bandwidth
- Selectable 2-wire and 4-wire resistance measurements.
- Resistance measurements up to 120MΩ with 1mΩ resolution
- Wide dc and ac current measurement ranges from 12mA to 12A
- Frequency measurements up to 1MHz with 0.01Hz resolution,
- dBm measurement with variable reference impedance from 2Ω to 8000Ω and audio power measurement capability.

**Basic measurement example**

This section describes the basic measurement procedures via operations in front panel.

**Voltage, resistance and frequency measurements**

Press the desired function button (Voltage, resistance or frequency) and connect the test leads (TL-36). The meter will select the appropriate range in the auto-range mode and an annunciator on the display will indicate measurement units.

This section describes how to operate the meter via standard RS-232 interface and gives detailed information of interface command sets used in the meter. The remote control operation allows the user either to manually operate the meter via a terminal or executes a host computer program automatically.
There are three types of commands, KEY commands, SET commands and QUERY commands.

KEY commands:

There are 16 pushbutton keys on the front panel of the meter. User may use the key commands <K1> to <K16> for directly simulating a single keystroke by pressing on the front panel push button via RS-232 interface. User may also use other key commands <K17> to <K20> for simulating combination keystroke functions.

SET commands

Unlike key commands, set command controls the meter operations through a string of commands. To set the meter at Vdc120V range, user may only need one command string

<S104>: S for setting,

1 for primary display

4 for 120V range.

QUERY commands

The purpose of query commands is used for requesting the meter to respond its current status. An example of a query command <R1> is used for requesting the meter to respond its primary display characters.

The wiring diagram for communicating PC through serial port from Digital Multimeter is shown in figure 2.6
2.5 Temperature measurement and control unit

Temperature has a significant effect on capacitance of a capacitor. Now a days there are only few materials which withstand for the change in temperature and sustain its capacitance over wide range. A. Rajendran and P. Neelamegam developed a setup with solid-state power controller with heater for furnace, the rate of heat is controlled by microcontroller and was able to heat the dielectric from 30 to 60°C[15]. Emine Dogru Bolat presented Matlab-SIMULINK based realtime temperature control of oven using different methods [16, 17].

As it is true that no instrument can perform measurements in entire range for a particular parameter. In the present study the author has chosen to measure at temperature range from room temperature to 120 °C required for the studies. A temperature controlled unit is developed and is interfaced with ARM controller. The details of the unit which range upto 400°C are presented below.

The measurement unit consists of a temperature sensor (thermocouple) and associated circuitry. A triac is used to control and vary temperature with the help of heater wire wound around one of the electrodes of the capacitor, which is one of the plates of a capacitor. The schematic diagram of temperature measurement and control is as shown in figure 2.7. The thermocouple (K-type Chromel/Alumel) is precision
temperature sensor, which can be easily calibrated. They operate as a 2-terminal voltage source to be easily calibrated which has a range of –200°C to +1250°C. The output of thermocouple is very weak to be processed which is of the order of microvolts. To amplify this signal an instrumentation amplifier is used which converts signal from microvolts to millivolts to be easily read by, a 10 bit ADC, an on chip of ARM microprocessor.

The whole process of measurement and control is accomplished with the help of an opamp op-07 as a comparator circuit. And an opto coupler is used to separate AC section from DC section and only trigger points are taken in to microprocessor circuit. From ARM processor a PWM pulse is generated taking into reference of the ZCD signal. The PWM signal that passes through a transistor to trigger BT136 triac, as it is dangerous to combine both high voltage AC and low voltage DC signals. These two signals are separated by using an opto coupler MOC3041. Across BT136 the resistive coil is connected as load. Depending on the ON time of PWM signal the temperature is generated and stabilized. If temperature required as low very low ON time is maintained and if temperature required as high, the ON time of the PWM signal is increased. If the temperature required and measured is same the ON time is maintained till temperature required is changed to high or low.

Both measurement and control are done by the PC using MATLAB as a processing tool. At first the user sets the temperature range with step level. By obtaining this data, MATLAB program sends necessary commands to ARM microprocessor. Once
Fig 2.7 : Schematic diagram of temperature measurement and control circuit
to the ARM processor, the temperature is measured, depending on the temperature requirements. The PWM signal is generated and ON time of the PWM signal is maintained. Once the desired temperature is attained, a serial data is sent to MATLAB indicating that the desired temperature is obtained. Until MATLAB completes its process and tells the ARM microprocessor to increase the temperature, the PWM signal holds on maintaining the same temperature. The process continued till maximum temperature is attained.

2.6 ARM Microprocessor (LPC2148)

Introduction

The LPC2141/2/4/6/8 microcontrollers are based on a 16/32 bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, that combines the microcontroller with embedded high speed flash memory ranging from 32 kB to 512 kB. A 128-bit wide memory interface and unique accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30% with minimal performance penalty. Due to their tiny size and low power consumption, LPC2141/2/4/6/8 are ideal for applications where miniaturization is a key requirement, such as access control and point-of-sale. A blend of serial communications interfaces ranging from a USB 2.0 Full Speed Device, multiple UARTS, SPI, SSP to I2Cs and on-chip SRAM of 8 kB up to 40 kB, make these devices very well suited for communication gateways and protocol converters, soft modems, voice recognition and low end imaging, providing both large buffer size and high processing power. Various 32-bit timers, single or dual 10-bit ADC(s), 10-bit DAC, PWM channels and 45 fast GPIO lines with up to nine edge or
level sensitive external interrupt pins make these microprocessors particularly suitable for industrial control and medical systems. The microprocessor used for the preset setup is ARM based LPC2148. This has many advanced features apart from 60MHz core and low cost, this is the key for battery operated and timing applications to be built around it. The schematic diagram of ARM (LPC 2148) is shown in figure 2.8 and its features and interfacial details are discussed below

**Features**

- 16/32-bit ARM7TDMI-S microcontroller in a tiny LQFP64 package.
- 8 to 40 kB of on-chip static RAM and 32 to 512 kB of on-chip flash program memory. 128 bit wide interface/accelerator enables high speed 60 MHz operation.
- In-System/In-Application Programming (ISP/IAP) via on-chip boot-loader software. Single flash sector or full chip erase in 400 ms and programming of 256 bytes in 1 ms.
- Embedded ICE RT and Embedded Trace interfaces offer real-time debugging with the on-chip Real Monitor software and high speed tracing of instruction execution.
- USB 2.0 Full Speed compliant Device Controller with 2 kB of endpoint RAM. In addition, the LPC2146/8 provide 8 kB of on-chip RAM accessible to USB by DMA.
- One or two (LPC2141/2 vs. LPC2144/6/8) 10-bit A/D converters provide a total of 6/14 analog inputs, with conversion times as low as 2.44 μs per channel.
- Single 10-bit D/A converter provide variable analog output.
- Two 32-bit timers/external event counters (with four capture and four compare channels each), PWM unit (six outputs) and watchdog.
- Low power real-time clock with independent power and dedicated 32 kHz clock input.
• Multiple serial interfaces including two UARTs (16C550), two Fast I2C-bus (400 kbit/s), SPI and SSP with buffering and variable data length capabilities.
• Vectored interrupt controller with configurable priorities and vector addresses.
• Up to 45 of 5 V tolerant fast general purpose I/O pins in a tiny LQFP64 package.
• Up to nine edge or level sensitive external interrupt pins available. 60 MHz maximum CPU clock available from programmable on-chip PLL with settling time of 100 µs.
• On-chip integrated oscillator operates with an external crystal in range from 1 MHz to 30 MHz and with an external oscillator up to 50 MHz.
• Power saving modes include Idle and Power-down.
• Individual enable/disable of peripheral functions as well as peripheral clock scaling for additional power optimization.
• Processor wake-up from Power-down mode via external interrupt, USB, Brown-Out Detect (BOD) or Real-Time Clock (RTC).
• Single power supply chip with Power-On Reset (POR) and BOD circuits: – CPU operating voltage range of 3.0 V to 3.6 V (3.3 V ± 10 %) with 5 V tolerant I/O pads.

2.7 SERIAL COMMUNICATION UNIT

The salient features of serial port interface are given below
• 16 byte Receive and Transmit FIFOs
• Register locations conform to ‘550 industry standard.
• Receiver FIFO trigger points at 1, 4, 8, and 14 bytes.
• Built-in fractional baud rate generator with autobauding capabilities.
• Mechanism that enables software and hardware flow control implementation
Given the status of U0IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. The U0IIR must be read in order to clear the interrupt prior to exiting the Interrupt Service Routine.

The UART0 RLS interrupt (U0IIR[3:1] = 011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the UART0 Rx input: overrun error(OE), Parity Error (PE), Framing Error (FE) and Break Interrupt (BI). The UART0 Rx error condition that set the interrupt can be observed via U0LSR[4:1]. The interrupt is cleared upon an U0LSR read. The UART0 RDA interrupt (U0IIR[3:1] = 010) shares the second level priority with the CTI interrupt (U0IIR[3:1] = 110). The RDA is activated when the UART0 Rx FIFO reaches the trigger level defined in U0FCR[7:6] and is reset when the UART0 Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level. The CTI interrupt (U0IIR[3:1] = 110) is a second level interrupt and is set when the UART0 Rx FIFO contains at least one character and no UART0 Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any UART0 Rx FIFO activity (read or write of UART0 RSR) will clear the interrupt. This interrupt is intended to flush the UART0 RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of 100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

Architecture

The architecture of the UART0 is explained in this section. The VPB interface
provides a communications link between the CPU or host and the UART0. The UART0 receiver block, U0RX, monitors the serial input line, RXD0, for valid input. The UART0 RX Shift Register (U0RSR) accepts valid characters via RXD0. After a valid character is assembled in the U0RSR, it is passed to the UART0 RX Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The UART0 transmitter block, U0TX, accepts data written by the CPU or host and buffers the data in the UART0 TX Holding Register FIFO (U0THR). The UART0 TX Shift Register (U0TSR) reads the data stored in the U0THR and assembles the data to transmit via the serial output pin, TXD0.

The UART0 Baud Rate Generator block, U0BRG, generates the timing enables used by the UART0 TX block. The U0BRG clock input source is the VPB clock (PCLK). The main clock is divided down per the divisor specified in the U0DLL and U0DLM registers. This divided down clock is a 16x oversample clock, NBAUDOUT.

The interrupt interface contains registers U0IER and U0IIR. The interrupt interface receives several one clock wide enables from the U0TX and U0RX blocks. Status information from the U0TX and U0RX is stored in the U0LSR. Control information for the U0TX and U0RX is stored in the U0LCR.

UART1 is identical to UART0, with the addition of a modem interface.

- 16 byte Receive and Transmit FIFOs.
- Register locations conform to '550 industry standard.
- Receiver FIFO trigger points at 1, 4, 8, and 14 bytes.
- Built-in fractional baud rate generator with autobauding capabilities.
- Mechanism that enables software and hardware flow control implementation.
• Standard modem interface signals included with flow control (auto-CTS/RTS) fully supported in hardware (LPC2144/6/8 only).

Interrupts handling procedure is explained in this section. Given the status of U1IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. The U1IIR must be read in order to clear the interrupt prior to exiting the Interrupt Service Routine.

The UART1 RLS interrupt (U1IIR[3:1] = 011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the UART1RX input: overrun error(OE), parity error (PE), framing error (FE) and break interrupt (BI). The UART1 Rx error condition that set the interrupt can be observed via U1LSR[4:1]. The interrupt is cleared upon an U1LSR read. The UART1 RDA interrupt (U1IIR[3:1] = 010) shares the second level priority with the CTI interrupt (U1IIR[3:1] = 110). The RDA is activated when the UART1 Rx FIFO reaches the trigger level defined in U1FCR7:6 and is reset when the UART1 Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level.

The CTI interrupt (U1IIR[3:1] = 110) is a second level interrupt and is set when the UART1Rx FIFO contains at least one character and no UART1 Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any UART1 Rx FIFO activity (read or write of UART1 RSR) will clear the interrupt. This interrupt is intended to flush the UART1 RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of
100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

**Architecture**

The VPB interface provides a communications link between the CPU or host and the UART1. The UART1 receiver block, U1RX, monitors the serial input line, RXD1, for valid input. The UART1 RX Shift Register (U1RSR) accepts valid characters via RXD1. After a valid character is assembled in the U1RSR, it is passed to the UART1 RX Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The UART1 transmitter block, U1TX, accepts data written by the CPU or host and buffers the data in the UART1 TX Holding Register FIFO (U1THR). The UART1 TX Shift Register (U1TSR) reads the data stored in the U1THR and assembles the data to transmit via the serial output pin, TXD1. The UART1 Baud Rate Generator block, U1BRG, generates the timing enables used by the UART1 TX block. The U1BRG clock input source is the VPB clock (PCLK). The main clock is divided down per the divisor specified in the U1DLL and U1DLM registers. This divided down clock is a 16x oversample clock, NBAUD OUT. The modem interface contains registers U1MCR and U1MSR. This interface is responsible for handshaking between a modem peripheral and the UART1. The interrupt interface contains registers U1IER and U1IIR. The interrupt interface receives several one clock wide enables from the U1TX and U1RX blocks. Status information from the U1TX and U1RX is stored in the U1LSR. Control information for the U1TX and U1RX is stored in the U1LCR. The LPC2xxx devices currently have two on-chip UARTS. They are both identical to use except UART1 has
additional modem support. Both peripherals conform to the “550 industry standard” specification. Both have a built-in baud rate generator and 16 byte transmit and receive FIFOs.

ARM Board-LPC214X is an evaluation board for LPC2148 ARM7TMDI based microcontroller. The LPC2148 microcontroller has 512KB of internal flash and 32+8K RAM. Following are the salient features of the board.

Power supply DC 6.5V with power LED. On-board linear regulators generate +3.3V/500mA and +5v/500mA from power supply. USB connector (as alternate power source). The connectors available on board are, Extension headers for all microcontroller pins, RS232 connectors (2), VGA connector, PS/2 connector, JTAG connector, SD/MMC connector, USB B-type connector with Link-LED. 2 line X 16 character LCD with backlight control.

2.7.1 Connecting the hardware

To test all the features on the ARM Board necessary to the present work we need the following accessories: program download(ISP) via serial port, 8 controllable LEDs, USB cable, DB-9 straight Full and Half modem serial cable

Once you have all these accessories connected to the ARM board you can run through a simple test to verify that all the peripherals are working fine. It is highly recommended that you test all the peripherals as soon you receive the ARM Board.

2.7.2 Programming ARM Board

ARM board can be programmed through wiggler clone JTAG or through serial port using ‘Flash Magic’. It is a freeware windows utility used download the hex file format onto the ARM board. If your PC does not have a serial port; use a USB to serial
Fig 2.8 Schematic diagram of ARM development board
converter to download the hex file using the Flash Magic utility. For programming with JTAG your system should have a parallel port and the supporting IDE which can communicate to the processor core over JTAG interface.

2.7.3 Programming ARM board Through ISP

The ARM board can be programmed through ISP in two modes:

1. Manual Mode
2. Auto Mode

1. Manual Mode

To program in Manual mode you need a half serial cable (which just has TX, RX and GND wire connected). Set the jumper to pins 1 & 2 of J26 and connect the half serial cable to UART0 (J5) and power the board. To make the board enter programming mode:

- Hold down SW2 (isp) and SW3 (reset), then release SW3 first and finally SW2
- The controller enters the bootloader mode if during reset the SW2 pin is low

2. Auto Mode

To program in Auto mode you need a full serial cable. Set the jumper to pins 2 & 3 of J26 and connect the full serial cable to UART0 (J5). When ARM board is powered ON black boxes will be displayed on LCD. Open Flash Magic tool, select the appropriate COM port, set the Baud rate to less than or equal to 38400 bps, select device as LPC2148, interface as 'None (ISP)' and oscillator frequency as 12MHz. Specify the path of your HEX file and click START. The status is shown at the bottom on the Flash Magic window.

In the 'Step 4 - Options' check 'Verify after programming' and 'Fill unused flash' options.
Checking the 'Set Code Read Prot' option will not allow you will program with JTAG. So keep it unchecked unless required.

1. In Auto mode under the 'Options' tab select 'Advanced options'. In this under 'Hardware Config' tab make sure the options 'Use DTR and RTS to control RTS and P0.14' and 'Keep RTS asserted while COM port open' are checked. The values of T1 and T2 are set to 100ms and 200ms by default.

2. After programming the board in Auto mode you should disconnect the serial cable from J5. This is a known issue.

2.7.4 USB Boot-loader

The BlueBoard-lpc214x can now be pre-loaded with the USB boot-loader firmware*. With the USB boot-loader you can simply drag and drop your compiled binary file onto the device as you would with any USB flash drive, and it will update itself accordingly. No need for serial port for flashing. To make the board enter Boot-loader mode:

After loading USB boot-loader firmware* to the BlueBoard-LPC214X, hold down SW1 (USBBL) and SW3 (RESET), then release SW3 first and finally SW1. Now the pre-loaded USB boot-loader allows you to enumerate the board as a Mass Storage Device. Drag the compiled binary file and drop onto the device and reset the board using SW3, and the firmware runs. USB boot-loader firmware* is provided as NGX_BT_LDR.hex. The Boot-Loader Examples and BIN files can be found at USB Boot-loader
2.8 Personal computer (PC)

The era of personal computer has started in early 70’s, it was intended for personal use. Even today the computer has same functional units, namely input, processing and output unit. The advancement is only in processing unit which included many functionalities like increased memory, serial, parallel, USB, LAN, WI-FI, audio etc., most of the advanced instruments build with microprocessors/controllers do include one of the communication protocol, this allows the user to control the instrument remotely.

In the present study the author uses serial port of a personal computer (PC) to transfer the data to and from ARM processor and programmable multimeter. The communication process is carried out and controlled by the GUI of the MATLAB program which is executed in PC. The requirements for running KEIL and MATLAB softwares are as below

Keil[6]

Computer must meet certain minimum system requirements for the evaluation of software package to function properly. Windows XP SP2, Windows Vista or Windows 7, Mouse or Similar Pointing Device Available Hard Disk Space for C51: 200Mbytes, 1GBytes of RAM, Pentium Class PC

MATLAB[7]

OS supported are Windows 7, (sp1, sp2), XP (sp) etc., and to be processor specific are any intel or AMD with SSE2 instruction set, more than 3GB for installation and atleast 1GB RAM is required.
Microcontroller transfer the data in two ways as parallel and serial. In parallel data transfers, often 8 or more lines (wire conductors) are used to transfer data to device that is only a few meters away, lot of data can be transferred in short time by using many wires. Data is transferred to a device located many meters away serial port is used.

To allow compatibility among data communication equipment made by various manufacturers, an interfacing standard called RS232 was set by the Electronics Industries Association (EIA) in 1960. In 1963 it was modified and called RS232A in this we use RS232. RS232 is the most widely used serial I/O interfacing standard. This standard is used in PCs and numerous types of equipment. Its input and output voltage levels are not TTL compatible. In RS232, a 1 is represented by –3 to –25V, while a 0 bit is +3V to +25V for this reason, to connect any RS232 to Microcontrollers system we must use voltage converters such as MAX232 to convert the TTL logic levels to the RS232 voltage level, and vice versa. Max 232 IC chips are commonly referred to as line drivers. The connection between a PC and microcontroller requires a minimum of three pins. TXD, RXD, and ground.

The MATLAB high-performance language for technical computing integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. You can watch the Getting Started with MATLAB video demo for an overview of the major functionality. If you have an active Internet connection, you can also watch the Working in the Development Environment video demo, and the Writing a MATLAB Program video demo. This collection includes the following topics: Introduction Describes the components of the MATLAB system Matrices and Arrays How to use MATLAB to
generate matrices and perform mathematical operations on matrices Graphics How to plot data, annotate graphs, and work with images Programming How to use MATLAB to create scripts and functions, how to construct and manipulate data structures Data Analysis How to set up a basic data analysis Creating Graphical User Interfaces Introduces GUIDE, the MATLAB graphical user interface development environment. Desktop Tools and Development Environment. Information about tools and the MATLAB desktop External Interfaces Introduces external interfaces available in MATLAB software.

The features of serial port interface using MATLAB software are explained in detail below

Serial communication is the most common low-level protocol for communication between two or more devices. Normally, one device is a computer, while the other device can be either a modem, or a printer or another computer, or a measuring instrument such as an oscilloscope or a function generator. As the name suggests, the serial port sends and receives bytes of information in a serial fashion — one bit at a time. These bytes are transmitted using either a binary (numerical) format or a text format.

The MATLAB serial port interface provides direct access to peripheral devices such as modems, printers, and scientific instruments that you connect to your computer’s serial port. This interface is established through a serial port object. The serial port object supports functions and properties that allow you to

- Configure serial port communications
- Use serial port control pins
• Write and read data
• Use events and callbacks
• Record information to disk

Over the years, several serial port interface standards have been developed. These standards include RS-232, RS-422, and RS-485 - all of which are supported by the MATLAB serial port object. Of these, the most widely used interface standard for connecting computers to peripheral devices is RS-232.

For many serial port applications, you can communicate with your device without detailed knowledge of how the serial port works. If your application is straightforward, or if you are already familiar with the previously mentioned topics, you might want to begin with The Serial Port Session to see how to use your serial port device with MATLAB software.

The original serial port interface standard was given by RS-232, which stands for Recommended Standard number 232. The term RS-232 is still in popular use, and is used in this thesis when referring to a serial communication port that follows the TIA/EIA-232 standard.

Primary communication is accomplished using three pins: the Transmit Data pin, the Receive Data pin, and the Ground pin. Other pins are available for data flow control, but are not required.

Using the synchronous protocol, all transmitted bits are synchronized to a common clock signal. The two devices initially synchronize themselves to each other, and continually send characters to stay synchronized. Even when actual data is not really being sent, a constant flow of bits allows each device to know where the other is at any
given time. That is, each bit that is sent is either actual data or an idle character. Synchronous communications allows faster data transfer rates than asynchronous methods, because additional bits to mark the beginning and end of each data byte are not required.

Using the asynchronous protocol, each device uses its own internal clock, resulting in bytes that are transferred at arbitrary times. So, instead of using time as a way to synchronize the bits, the data format is used.

In particular, the data transmission is synchronized using the start bit of the word, while one or more stop bits indicate the end of the word. The requirement to send these additional bits causes asynchronous communications to be slightly slower than synchronous. However, it has the advantage that the processor does not have to deal with the additional idle characters. Most serial ports operate asynchronously.

The Serial Port Session

The serial port session comprises all the steps you are likely to take when communicating with a device connected to a serial port. These steps are:

Create a serial port object — create a serial port object for a specific serial port using the serial creation function. Configure properties during object creation if necessary. In particular, you might want to configure properties associated with serial port communications such as the baud rate, the number of data bits, and so on.

Connect to the device — connect the serial port object to the device using the fopen function. After the object is connected, alter the necessary device settings by configuring property values, read data, and write data.
Configure properties — To establish the desired serial port object behavior, assign values to properties using the set function or dot notation. In practice, you can configure many of the properties at any time including during, or just after, object creation. Conversely, depending on your device settings and the requirements of your serial port application, you might be able to accept the default property values and skip this step.

Write and read data — Write data to the device using the fprintf or fwrite function, and read data from the device using the fgetl, fgets, fread, fscanf, or readasync function. The serial port object behaves according to the previously configured or default property values.

Disconnect and clean up — when you no longer need the serial port object, disconnect it from the device using the fclose function, remove it from memory using the delete function, and remove it from the MATLAB workspace using the clear command.

Microcontroller transfers the data in two ways as parallel and serial. In parallel data transfers, often 8 or more lines (wire conductors) are used to transfer data to device that is only a few meters away. Example of parallel transfers is printers. Each uses cable with many wire strips. Although in such cases a lot of data can be transferred in a short amount of time by using many wires in parallel, the distance should be short. To transfer to a device located many meters away, the serial method is used.

To allow compatibility among data communication equipment made by various manufacturers, an interfacing standard called RS232 was set by the Electronics Industries Association (EIA) in 1960. In 1963 it was modified and called RS232A in this
we use RS232. RS232 is the most widely used serial I/O interfacing standard. This standard is used in PCs and numerous types of equipment. Its input and output voltage levels are not TTL compatible. In RS232, a 1 is represented by −3 to −25V, while a 0 bit is +3V to +25V for this reason, to connect any RS232 [8] to Microcontrollers system we must use voltage converters such as MAX232 to convert the TTL logic levels to the RS232 voltage level, and vice versa. Max 232 IC chips are commonly referred to as line drivers. The connection between a PC and microcontroller requires a minimum of three pins (TXD, RXD and ground).

2.9 Hardware description

The system designed and developed for the present study, in which the dielectric cell acts as a capacitor C is connected in the autobalance bridge circuit using a BNC connector. The instruments (programmable function generator and dual display multimeter) and ARM processor are integrated with personal computer through RS 232 standard interface using MATLAB as a tool to measure capacitance variations along with graphical user interface (GUI) for online graphical representation of frequency and temperature effect on capacitance materials.

In Autobalance Bridge a Resistor (R) and Capacitor (Cx) are connected as shown in figure 2.2. The designed programmable function generator AD9834 can generate frequencies from 10 Hz to 1MHz. Depending on a sample glass transition temperature a safe temperature zone is generated to avoid any damage to parallel plates of capacitor. The glass transition temperature is an important parameter for any sample, here in particular it is important because increasing temperature beyond certain level may deform the shape of the sample upon which the entire measurement is
based. The temperature can be measured and controlled by sending appropriate command signals to ARM processor from MATLAB GUI. MATLAB program identifies and initializes the instruments PROTEK and ESCORT through serial port before starting the measurement; the software needs inputs from user (resistor and frequency values) and START command. Protek instrument gets commands from personal computer to generate 1Vrms voltage and frequency 1 KHz. The Escort multimeter gives measured AC voltage across Capacitor under test for frequency under test.

The coil wound around lower plate of the capacitor is connected to a power source of 220V and triac. The gate of triac is controlled by a microprocessor port pin which is configured as PWM. A K-type thermocouple is used to measure the temperature around the plates of a capacitor. As the material is subjected to same closed environment the temperature attained to plates is transferred to the material under study without any loss. The thermocouple is connected to an instrumentation amplifier thereby amplifying signal to required level. The analog signal is then fed to ARM microprocessor to an ADC input, this signal is further processed to get the temperature of the capacitor.

The designed MATLAB Graphical User Interface (GUI), which controls and collects the data from designed function generator, multimeter for voltage across capacitor under test Cx, thermocouple and a triac is shown in figure 2.8. GUI collects data from the instruments, stores and computes capacitance using equations (1) and (2) for capacitor under test. The characteristics of designed instrument is compared and calibrated with a standard instrument and the results are tabulated in table 2.1.
Table 2.1. Comparison of some of the instrument parameters of the fabricated instrument with the HIOKI 3532-50 LCR HITESTER instrument

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Parameter</th>
<th>HIOKI 3532-50 LCR HITESTER</th>
<th>Fabricated instrument</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Capacitance</td>
<td>0.32pF to 370mF</td>
<td>1pF to 100uF</td>
</tr>
<tr>
<td>2</td>
<td>Dielectric loss</td>
<td>0.00001 to 9.99999</td>
<td>0.0001 to 9.9999</td>
</tr>
<tr>
<td>3</td>
<td>Accuracy</td>
<td>$Z : \pm 0.08% \text{ rdg.} \ \Theta : \pm 0.05^\circ$</td>
<td>±0.2%</td>
</tr>
<tr>
<td>4</td>
<td>Measurement frequency</td>
<td>42 Hz to 5 MHz</td>
<td>100Hz to 1MHz</td>
</tr>
<tr>
<td>5</td>
<td>Measurement signal levels</td>
<td>10 mV to 5 V rms</td>
<td>10mV to 20Vrms</td>
</tr>
<tr>
<td>6</td>
<td>Display screen</td>
<td>LCD with backlight / 99999 (full 5 digits)</td>
<td>Personal Computer screen</td>
</tr>
<tr>
<td>7</td>
<td>Settings in memory</td>
<td>Maximum 30 sets</td>
<td>Unlimited through programming</td>
</tr>
</tbody>
</table>
References

1. TIA/EIA-232-F, Interface Between Data Terminal Equipment and Data Circuit Terminating Equipment Employing Serial Binary Data Interchange.
   Serial communication unit (MAX 232) is used to transfer the data to print screen or hyper terminal of the personal computer.


14. V. Dumbrava, L. Svilainis, the automated complex impedance measurement system, Electronics and Electrical Engineering, 2007, Nr.4(76), ISSN 1392-1215.

