

“Heaven’s light is our guide”

Rajshahi University of Engineering & Technology



Department of Electrical & Electronic Engineering

Course No : EEE 4210

Course Title : Embedded System Design Sessional

Experiment no : 04

Name of the Experiment: Study on Serial Communication in 8051 Microcontroller using EdSim51 DI Software

Date of Experiment: December 13, 2023

Date of Submission: December 20, 2023

Submitted By

Md Maruf Hassan

Department : EEE

Roll : 1801105

Section : B

Submitted To

Md Mayenul Islam

Lecturer,

Department of EEE,

RUET

Experiment No.: 04

Experiment Name: Study on Serial Communication in 8051 Microcontroller using EdSim51 DI Software

Objectives:

1. To know about the various types of serial communication
2. To know about the interfacing of serial communication using 8051 microcontrollers

Theory:

The 8051 microcontroller employs parallel data transfer using eight data lines to interface with parallel I/O devices, transferring eight bits simultaneously. However, for long-distance communication, where parallel data transfer can be costly, serial communication is preferred. In serial communication, an 8-bit data byte is converted into serial bits using a parallel-in-serial-out shift register before being transmitted over a single data line, with the least significant bit transmitted first. Serial communication, also known as asynchronous communication, is a method of transmitting data one bit at a time over a single transmission line. This method is essential for interfacing the 8051 microcontrollers with various external devices like computers, peripheral chips, and other microcontrollers. The 8051 microcontroller features a full duplex serial port, facilitated by three special function registers. The SBUF register, an 8-bit register, handles data transmission and reception separately, requiring data to be placed in SBUF for transmission via TXD and holding received data for reading. The SCON register includes mode selection bits, serial port interrupt indicators (TI and RI), and a ninth data bit (TB8 and RB8) for transmission and reception. Additionally, the PCON register's SMOD bit (bit 7) controls the baud rate in asynchronous mode transmission.

Modes of serial communication:

1. Mode 0: The serial port operates in synchronous mode with data transmission and reception through RXD, while TXD is used for clock output, and the baud rate is set at 1/12 of the clock frequency.
2. Mode 1: SBUF functions as a 10-bit full duplex transceiver with 1 start bit, 8 data bits, and 1 stop bit; the baud rate is variable and determined by Timer 1 overflow rate.
3. Mode 2: Similar to Mode 1 but with the addition of a programmable 9th data bit, resulting in a total of 11 bits transmitted or received, and the baud rate is calculated as in Mode 1.
4. Mode 3: Similar to Mode 2, but the baud rate calculation is the same as in Mode 1.

RS-232 Standard: The RS232 interfacing standard, established by the Electronics Industries Association (EIA) in 1960, aims to ensure compatibility among data communication equipment from different manufacturers. However, due to its pre-logic family origin, RS232 uses voltage levels that are not TTL-compatible, with logic one represented by -3 to -25V (MARK) and logic zero by +3 to +25V (SPACE). To connect RS232 to a microcontroller system, voltage converters like the MAX232 are required to convert TTL logic levels to RS232 voltage levels and vice versa, with MAX232 IC chips commonly known as line drivers. Two types of connectors, DB9 and DB25, are used in the RS232 standard.

Assembly Program:

1. *A program for the 8051 to transfer letter 'A' serially at 4800- baud rate, 8 bit data, 1 stop bit continuously.*

Program:

```

ORG 0000H
LJMP START
ORG 0030H
START: MOV TMOD, #20H ; select timer 1 mode 2
MOV TH1, #0FAH ; load count to get baud rate of 4800
MOV SCON, #50H ; initialize UART in mode 2
; 8 bit data and 1 stop bit
SETB TR1 ; start timer
AGAIN: MOV SBUF, #'A' ; load char 'A' in SBUF
BACK: JNB TI, BACK ; Check for transmit interrupt flag
CLR TI ; Clear transmit interrupt flag
SJMP AGAIN
END

```

2. *A program for the 8051 to transfer the message 'EARTH' serially at 9600 baud, 8 bit data, 1 stop bit continuously.*

Program:

```

ORG 0000H
LJMP START
ORG 0030H
START: MOV TMOD, #20H ; select timer 1 mode 2

MOV TH1, #0FDH ; load count to get reqd. baud rate of 9600
MOV SCON, #50H ; initialise uart in mode 2
; 8 bit data and 1 stop bit
SETB TR1 ; start timer
LOOP: MOV A, #'E' ; load 1st letter 'E' in a

```

ACALL LOAD ; call load subroutine
 MOV A, #'A' ; load 2nd letter 'A' in a
 ACALL LOAD ; call load subroutine
 MOV A, #'R' ; load 3rd letter 'R' in a
 ACALL LOAD ; call load subroutine
 MOV A, #'T' ; load 4th letter 'T' in a
 ACALL LOAD ; call load subroutine
 MOV A, #'H' ; load 4th letter 'H' in a
 ACALL LOAD ; call load subroutine
 SJMP LOOP ; repeat steps
 LOAD: MOV SBUF, A
 HERE: JNB TI, HERE ; Check for transmit interrupt flag
 CLR TI ; Clear transmit interrupt flag
 RET
 END

Output from EdSim51 DI Software:

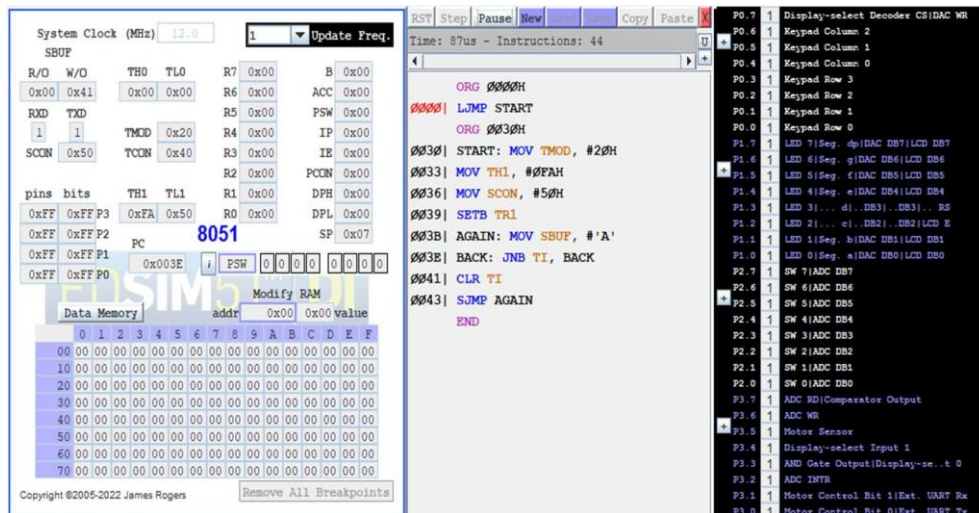


Fig. 4.1: Output for program 1

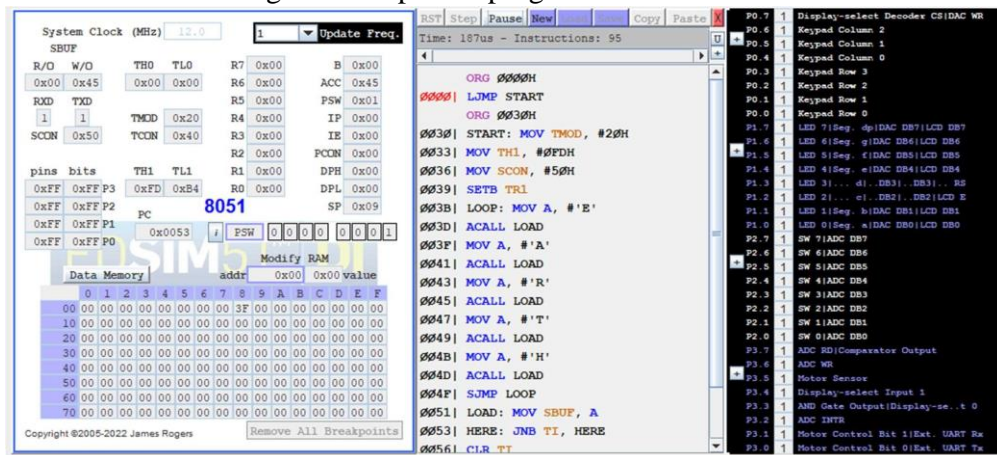


Fig. 4.2: Output for program 2

Discussion and Conclusion:

In conducting the experiment on Serial Communication in the 8051 Microcontroller using EdSim51 DI Software, two distinct programs were developed to assess the serial transmission capabilities. The first program efficiently transferred the letter 'A' serially at a baud rate of 4800, utilizing 8 bits of data and 1 stop bit in a continuous loop. The second program demonstrated the versatility of the 8051 microcontrollers by serially transmitting the message 'EARTH' at a faster baud rate of 9600, also employing 8 bits of data and 1 stop bit continuously. This experiment enhances understanding and proficiency in utilizing the 8051 microcontrollers for diverse serial communication applications.

“Heaven’s light is our guide”

Rajshahi University of Engineering & Technology



Department of Electrical & Electronic Engineering

Course No : EEE 4210

Course Title : Embedded System Design Sessional

Experiment no : 05

Name of the Experiment: Study on Programmable Logic Controller (PLC) using LOGOSoft Comfort Software and GOT7 PLC Trainer OMRON CP1E 40 I/O

Date of Experiment: December 13, 2023

Date of Submission: December 20, 2023

Submitted By

Md Maruf Hassan

Department : EEE

Roll : 1801105

Section : B

Submitted To

Md Mayenul Islam

Lecturer,

Department of EEE,

RUET

Experiment No.: 05

Experiment Name: Study on Programmable Logic Controller (PLC) using LOGOSoft Comfort Software and GOTT PLC Trainer OMRON CP1E 40 I/O

Objectives:

1. To know about the programmable logic controller (PLC)
2. To know how PLC is implemented in automation system
3. To know how to control electrical appliances using PLC
4. To learn how to construct basic gates (AND, OR, NOT, NOR, NAND, etc.) using PLC

Theory:

A Programmable Logic Controller (PLC) is a versatile and ruggedized digital device widely employed in industrial automation. Its primary function is to monitor and control various processes within an industrial system. PLCs are designed to withstand harsh environments and are equipped with programmable memory, enabling the execution of customized control functions. They have replaced traditional relay-based control systems, offering increased flexibility, reliability, and efficiency in automating complex processes across industries such as manufacturing, energy, and transportation. PLCs operate by receiving input signals from sensors, processing this information using programmed logic, and then activating output devices such as motors, valves, and relays to regulate the industrial processes. This adaptability and programmability make PLCs a fundamental component in modern automation, facilitating precise control, real-time monitoring, and rapid response to dynamic industrial environments.

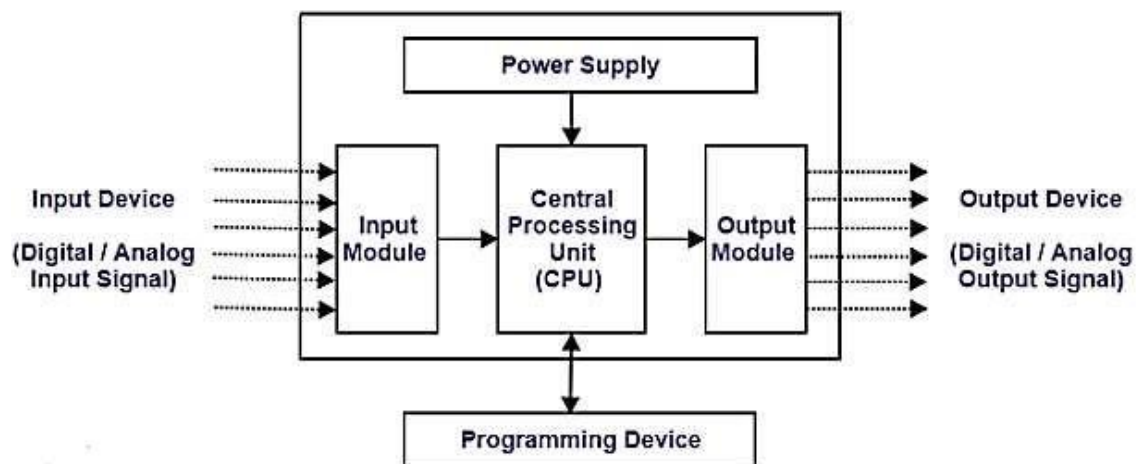


Fig. 5.1: Block diagram of PLC

PLC works by following a user-defined program to monitor inputs from sensors and devices, make decisions based on that data, and then send control signals to actuators and other equipment. The operation of PLC is described below:

1. **Sensing:** PLCs gather information from the physical world through various sensors like temperature sensors, pressure sensors, and limit switches. These sensors convert realworld conditions into electrical signals that the PLC can understand.
2. **Processing:** The core of the PLC is the Central Processing Unit (CPU), which executes the program written by the user. This program is essentially a set of instructions that tell the PLC what to do based on the sensor inputs. The CPU analyzes the input data, applies the logic defined in the program, and makes decisions about the system's operation.
3. **Actuation:** Based on the decisions made by the CPU, the PLC sends control signals to various actuators like relays, valves, motors, and drives. These actuators then physically affect the process or equipment, carrying out the desired actions.
4. **Communication:** PLCs can also communicate with other devices and systems, such as Human-Machine Interfaces (HMIs) for operator interaction, Supervisory Control and Data Acquisition (SCADA) systems for monitoring and control, and even other PLCs for coordinated operation of complex systems.

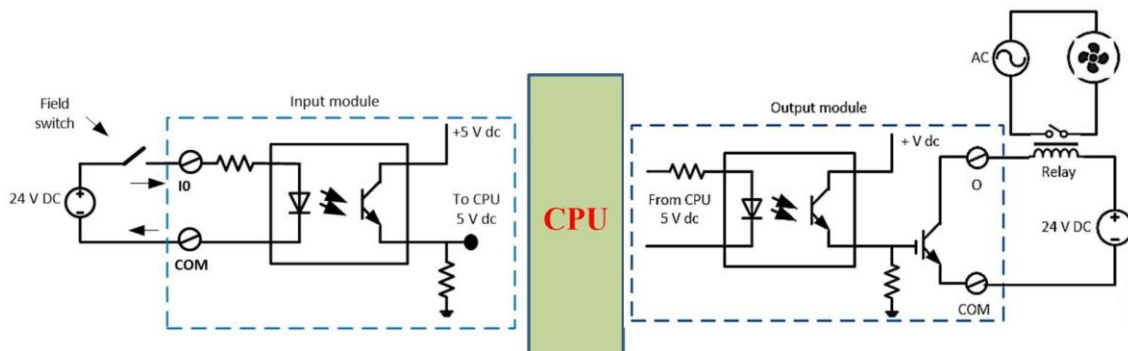






Fig. 5.2: Internal connection of a PLC

PLC programming languages are designed to create control algorithms for Programmable Logic Controllers. Common PLC programming languages include:

1. **Ladder Logic (LL):** Resembles electrical relay logic diagrams with rungs representing control circuits. Well-suited for discrete control applications and easy to understand for those familiar with electrical schematics.
2. **Function Block Diagrams (FBD):** Uses graphical blocks to represent functions, with connections between blocks indicating data flow. Ideal for complex control tasks and facilitates modular programming.
3. **Structured Text (ST):** Resembles high-level programming languages, such as C or Pascal, using text-based code. Suitable for complex mathematical operations and algorithmic control logic.
4. **Sequential Function Charts (SFC):** Represents control logic as a series of steps or states with transitions between them. Effective for modeling sequential processes and statebased control.

PLC Programming using Ladder Diagram Table:

5.1: Basic symbols of ladder diagram

Sl. No.	Symbol	Description
1		Normally open contact
2		Normally close contact
3		Output coil (Relay coil): Not connected to input
4		Output coil (Relay coil): Connected to input

1. OR Gate

Table 5.2: Truth table of OR gate

Input		Output
I1	I2	Q1
0	0	0
0	1	1
1	0	1
1	1	1

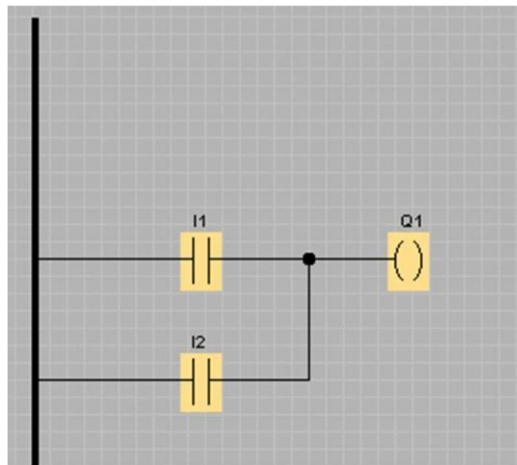


Fig. 5.3: Implementation of OR gate using ladder diagram of PLC in LOGO!Soft Comfort Software

2. AND Gate

Table 5.3: Truth table of AND gate

Input		Output
I1	I2	Q1
0	0	0
0	1	0

1	0	0
1	1	1

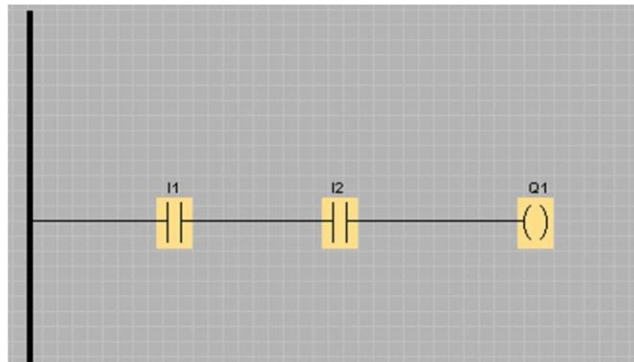


Fig. 5.4: Implementation of AND gate using ladder diagram of PLC in LOGO!Soft Comfort Software

3. NOT Gate

Table 5.4: Truth table of NOT gate

Input	Output
I1	Q1
0	1
1	0

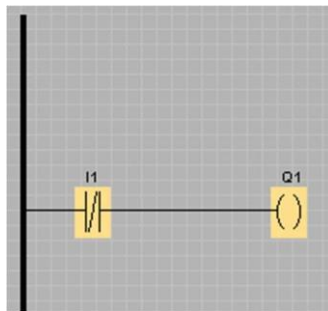


Fig. 5.5: Implementation of NOT gate using ladder diagram of PLC in LOGO!Soft Comfort Software

4. NAND Gate

Table 5.5: Truth table of NAND gate

Input		Output
I1	I2	Q1
0	0	1
0	1	1
1	0	1
1	1	0

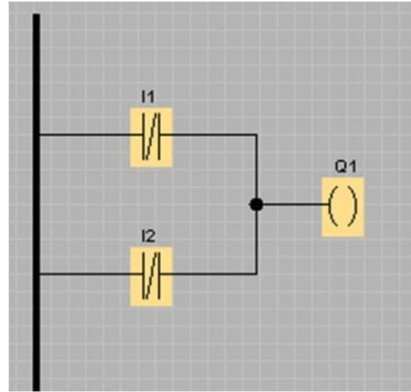


Fig. 5.6: Implementation of NAND gate using ladder diagram of PLC in LOGO!Soft Comfort Software

5. Ex-OR Gate

Table 5.6: Truth table of Ex-OR gate

Input		Output
I1	I2	Q1
0	0	0
0	1	1
1	0	1
1	1	0

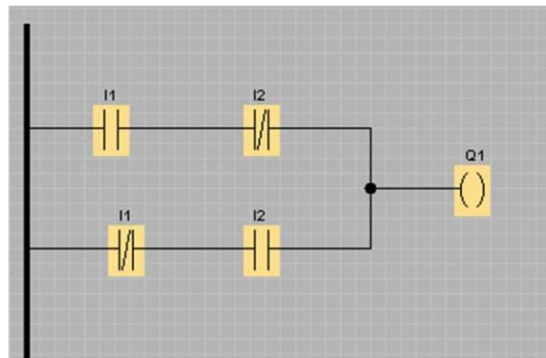


Fig. 5.7: Implementation of Ex-OR gate using ladder diagram of PLC in LOGOSoft Comfort Software



Fig. 5.8: Implementation PLC program in GOTT PLC Trainer OMRON CP1E 40 I/O

Discussion and Conclusion:

In the experiment studying Programmable Logic Controller (PLC) using LOGOSoft Comfort Software and the GOTT PLC Trainer OMRON CP1E 40 I/O, I successfully designed ladder diagrams for fundamental logic gates such as OR, AND, NOT, NAND, and Ex-OR in LOGOSoft Comfort Software. The exercise allowed practical application and visualization of logical operations through the intuitive graphical interface of LOGOSoft. The integration with the GOTT PLC Trainer and OMRON CP1E 40 I/O facilitated hands-on experience in implementing these logic gates on an actual PLC system. So, it can be concluded that the experiment successfully demonstrated the practical implementation of fundamental logic gates using the LOGOSoft Comfort Software and GOTT PLC Trainer OMRON CP1E 40 I/O, providing valuable insights into the application of Programmable Logic Controllers in industrial automation.