

SEP 2018

VOL 52

Introduction to SPI Interface

by Piyu Dhaker

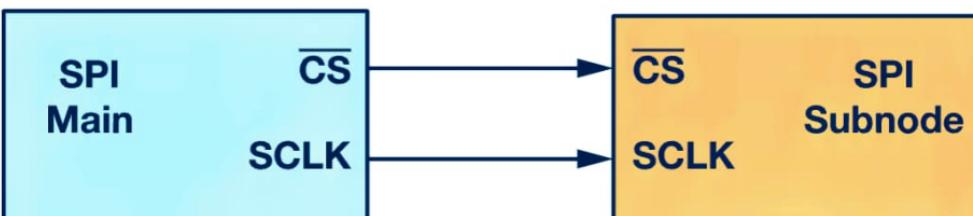
 [Download PDF](#)

Serial peripheral interface (SPI) is one of the most widely used interfaces between microcontroller and peripheral ICs such as sensors, ADCs, DACs, shift registers, SRAM, and others. This article provides a brief description of the SPI interface followed by an introduction to Analog Devices' SPI enabled switches and muxes, and how they help reduce the number of digital GPIOs in system board design.

SPI is a synchronous, full duplex main-subnode-based interface. The data from the main or the subnode is synchronized on the rising or falling clock edge. Both main and subnode can transmit data at the same time. The SPI interface can be either 3-wire or 4-wire. This article focuses on the popular 4-wire SPI interface.

 [Give Feedback](#)

Interface



X I2C Communication Proto...
From geeksforgeeks.org – delivered by



GEEKSFORGEEKS

I2C Communication Protocol

I2C stands for **Inter-Integrated Circuit**. It is a bus interface connection protocol incorporated into devices for serial communication. It was originally designed by Philips Semiconductor in 1982. Recently, it is a widely used protocol for short-distance communication. It is also known as Two Wired Interface(TWI).

Working of I2C Communication Protocol :

It uses only 2 bi-directional open-drain lines for data communication called SDA and SCL. Both these lines are pulled high.

Serial Data (SDA) – Transfer of data takes place through this pin.

Serial Clock (SCL) – It carries the clock signal.

I2C operates in 2 modes –

- Master mode
- Slave mode

Each data bit transferred on SDA line is synchronized by a high to the low pulse of each clock on the SCL line.

Open In App

Vdd



+ Project

Sig

Platforms Projects Contests

DC motor interfacing with 8051

Small DC Motor



Small DC Motor

DC motor converts electrical energy in the form of Direct Current into mechanical energy.

- In the case of the motor, the mechanical energy produced is in the form of a rotational

ChatGPT 3.5



Certainly! Below is a simple assembly language program for an 8051 microcontroller to rotate a DC motor continuously using a motor driver:

Assembly

Copy code

```
ORG 0x0000
```

```
MOV P1, #0x00 ; Initialize port P1 for motor control  
MOV TMOD, #01H ; Set timer 0 in mode 1  
MOV TH0, #OFEH ; Set initial value for Timer 0 (adjust for  
desired speed)  
MOV TL0, #OFEH ; Set initial value for Timer 0 (adjust for  
desired speed)
```

```
START:    ; Main loop
```

```
    MOV P1, #0x06 ; Set appropriate pins for clockwise  
rotation
```

```
    SETB TR0    ; Start Timer 0
```

```
WAIT_CLOCKWISE:
```

```
    JNB TF0, WAIT_CLOCKWISE ; Wait for Timer 0  
overflow
```

```
    CLR TR0    ; Stop Timer 0
```

```
    CLR TF0
```

```
    MOV P1, #0x09 ; Set appropriate pins for  
counterclockwise rotation
```

```
    SETB TR0    ; Start Timer 0
```

```
WAIT_COUNTERCLOCKWISE:
```

```
    JNB TF0, WAIT_COUNTERCLOCKWISE ; Wait for  
Timer 0 overflow
```

```
    CLR TR0    ; Stop Timer 0
```

```
    CLR TF0
```

```
    JMP START    ; Repeat the loop
```

```
END
```



Message



ChatGPT 3.5



TE YOU

Classify the modes of communication possible using 8051 µC



CHATGPT

The 8051 microcontroller supports various modes of communication, including:

1. Serial Communication Modes:

- **UART (Universal Asynchronous Receiver/Transmitter):** Asynchronous serial communication.
- **USART (Universal Synchronous/Asynchronous Receiver/Transmitter):** Synchronous or asynchronous serial communication.

2. Parallel Communication Modes:

- **Parallel I/O Ports:** Direct parallel data transfer through the microcontroller's I/O ports.

3. Interrupt-Driven Communication:

- Utilizing interrupts to handle communication events efficiently.

4. I2C (Inter-Integrated Circuit):

- A multi-master, multi-slave serial communication protocol for devices to communicate over short distances.

5. SPI (Serial Peripheral Interface):

Message



ChatGPT 3.5



1. Serial Communication Modes:

- **UART (Universal Asynchronous Receiver/Transmitter):** Asynchronous serial communication.
- **USART (Universal Synchronous/Asynchronous Receiver/Transmitter):** Synchronous or asynchronous serial communication.

2. Parallel Communication Modes:

- **Parallel I/O Ports:** Direct parallel data transfer through the microcontroller's I/O ports.

3. Interrupt-Driven Communication:

- Utilizing interrupts to handle communication events efficiently.

4. I2C (Inter-Integrated Circuit):

- A multi-master, multi-slave serial communication protocol for devices to communicate over short distances.

5. SPI (Serial Peripheral Interface):

- A synchronous serial communication protocol for communication between multiple peripheral devices.

These modes enable the 8051 microcontroller to interact with a variety of devices and systems, making it versatile in embedded applications.

Screenshot saved

Message





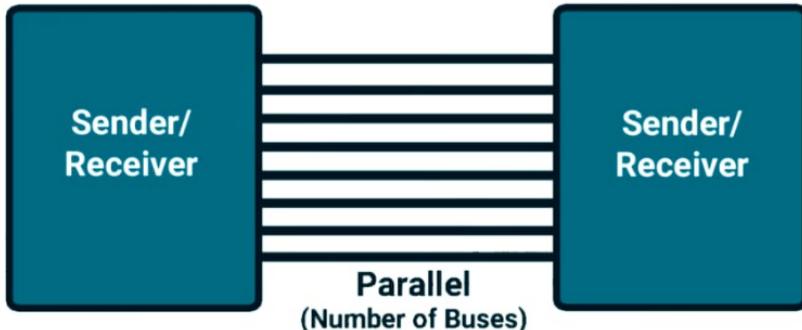
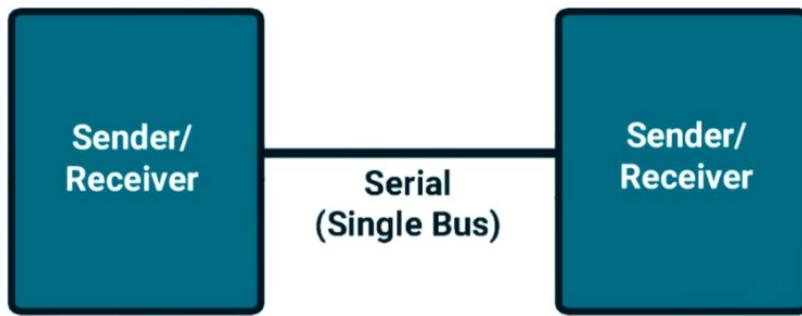
+ Project

Sign In

[Platforms](#) [Projects](#) [Contests](#)

8051 UART

Introduction



8051 UART Serial vs Parallel Communication

Serial communication means to transfer data bit by bit serially at a time, whereas in parallel communication, the number of bits that can be transferred at a time depends upon the number of data lines available for communication.

Two methods of serial communication are

- Synchronous Communication: Transfer of bulk data in the framed structure at a time



Search tutorials, courses and ebooks...



Microcontrollers 8051 Input Output Ports

[Previous](#)[Next](#)

8051 microcontrollers have 4 I/O ports each of 8-bit, which can be configured as input or output. Hence, total 32 input/output pins allow the microcontroller to be connected with the peripheral devices.

- **Pin configuration**, i.e. the pin can be configured as 1 for input and 0 for output as per the logic state.



+ Project

Sign In

[Platforms](#) [Projects](#) [Contests](#)

8051 Interrupts

Introduction to 8051 Microcontroller Interrupts

An interrupt is an event that occurs randomly in the flow of continuity. It is just like a call you have when you are busy with some work and depending upon call priority you decide whether to attend or neglect it.

The same thing happens in microcontrollers. 8051 architecture handles 5 interrupt sources, out of which two are internal (Timer Interrupts), two are external and one is a serial interrupt. Each of these interrupts has its interrupt vector address. **The highest priority interrupt in 8051 Microcontroller is the Reset**, with vector address 0x0000.

8051 interrupt vector table

Vector Address: This is the address where the controller jumps after the interrupt to serve the ISR (interrupt service routine).

Interrupt	Flag	Interrupt vector address
-----------	------	--------------------------

Timer & Counter of 8051 μ C

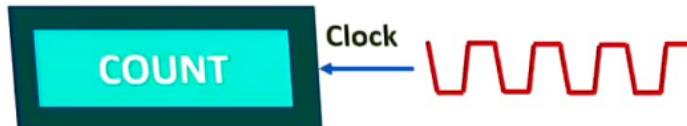
- 8051 has Two 16 bits Timers T0 & T1, working as up counters. T0 and T1 is further divided into 8bits of registers TH0-TL0 and TH1-TL1.



- How to Load Count?
- This Timers are Up counter.
- So, on given clock it will increment by 1.
- When it reaches to FFFFH, it will rolls back to 0000H and during that it will generates Timer Overflow interrupt.

$$\text{Count} = \text{FFFFH} - \text{Value} + 1$$

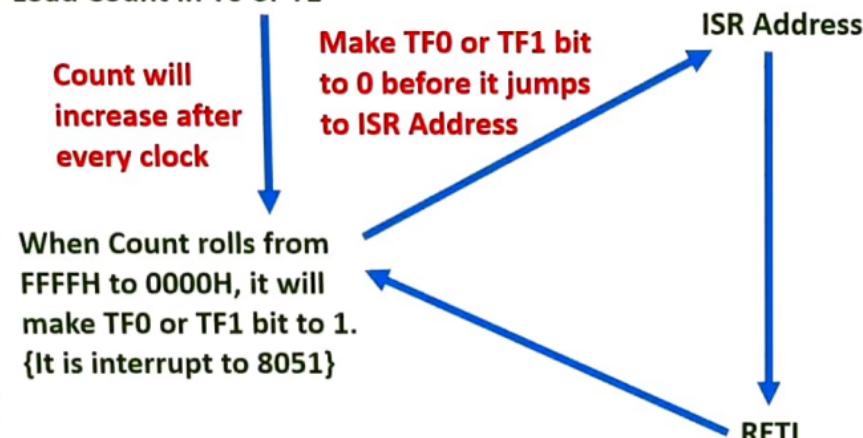
- So, if you wants to count 9 then Count = FFFF - 9 + 1 = FFF7H
- MOV TH0, #FFH
MOV TL0, #F7H



- This count loaded in T0 or T1 will increment after every clock.

- Timer or Counter? •
- If clock to the count is given by internal clock of 8051 then it will be timer and if clock is given by external clock on T0 and T1 then it will counter.
- That is to be configured by TMOD register of 8051.
- T/C bit will decide timer or counter configuration of 8051.
- How Timer / Counter works?

Load Count in T0 or T1



- ISR Address of Timer 0 and Timer 1
- Timer 0 ISR Address is 000BH and Timer 1 ISR Address is 001BH

ChatGPT 3.5



Illustrate the interfacing of 8051 µC to Counters.



CHATGPT

Interfacing a 8051 microcontroller (μ C) with counters typically involves connecting the microcontroller to external counter ICs, such as the 74LS90 or 74LS93. Here's a basic outline:

1. **Select Counter IC:** Choose a counter IC based on your requirements (e.g., 74LS90 is a decade counter).
2. **Power Connections:** Connect VCC and GND pins of the counter IC to the appropriate power supply levels.
3. **Clock Input (CLK):** Connect the CLK input of the counter IC to a suitable pin on the 8051 μ C that provides a clock signal. This could be a timer output or any other clock source.
4. **Clear Input (CLR):** Connect the CLR (clear) input of the counter IC to a pin on the 8051 for resetting the counter if needed.
5. **Output Pins:** Connect the output pins (Q0, Q1, Q2, etc.) of the counter IC to the input pins of the 8051. The 8051 will read the count.
6. **Read/Control:** Implement your 8051 code,

Message



ChatGPT 3.5



is a decade counter).

2. **Power Connections:** Connect VCC and GND pins of the counter IC to the appropriate power supply levels.
3. **Clock Input (CLK):** Connect the CLK input of the counter IC to a suitable pin on the 8051 µC that provides a clock signal. This could be a timer output or any other clock source.
4. **Clear Input (CLR):** Connect the CLR (clear) input of the counter IC to a pin on the 8051 for resetting the counter if needed.
5. **Output Pins:** Connect the output pins (Q0, Q1, Q2, etc.) of the counter IC to the input pins of the 8051 that will read the count.
6. **Read/Control Logic:** In your 8051 code, implement the logic to read the counter values and control the counter operations, like clearing or presetting the count.
7. **Programming:** Program the 8051 with the necessary code to handle the counter interface.

Remember to check the datasheets of the microcontroller and counter IC for specific details regarding pin configurations, voltage levels, and timing requirements.

Screenshot saved

Message



← 6-memory-and-io-interfacing



TOPIC 6

MEMORY AND I/O INTERFACING

MEMORY INTERFACING

i. External ROM (program memory) Interfacing

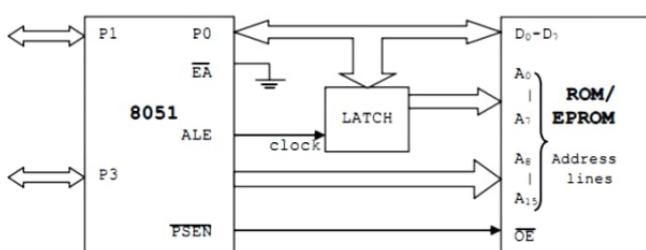


FIGURE 1 INTERFACING OF ROM/EPROM TO μ C 8051.

above figure shows how to access or interface ROM to 8051.
port 0 is used as multiplexed data & address lines.
it gives lower order (A_7-A_0) 8 bit address in initial T cycle & higher order (A_8-A_{15}) used as data bus.
8 bit address is latched using external latch & ALE signal from 8051.
port 2 provides higher order ($A_{15}-A_8$) 8 bit address.
PSEN is used to activate the output enable signal of external ROM/EPROM.

ii. External RAM (data memory) Interfacing

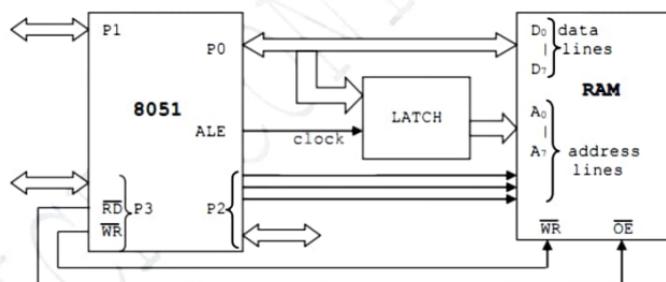


FIGURE 2 INTERFACING OF RAM (DATA MEMORY) TO μ C 8051.

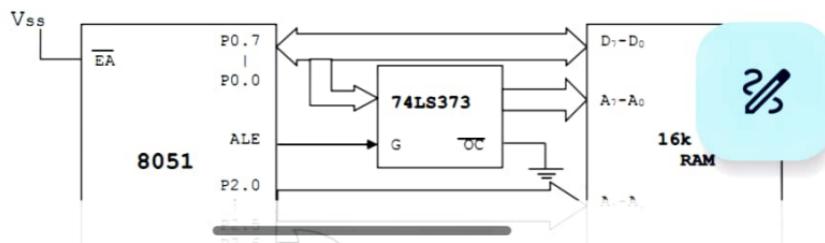
above figure shows how to connect or interface external RAM(data memory) to 8051.
port 0 is used as multiplexed data & address lines.
address lines are decoded using external latch & ALE signal from 8051 to provide lower order (A_7-A_0) address lines.
port 2 gives higher order address lines.
RD & WR signals from 8051 selects the memory read & memory write operations respectively.

RD & WR signals: generally P3.6 & P3.7 pins of port 3 are used to generate memory read and memory write signals.
remaining pins of port 3 i.e. P3.0-P3.5 can be used for other functions.

LINEAR AND ABSOLUTE DECODING

i. Absolute Decoding

all higher address lines : decoded to select memory chip for specific logic levels.
for other logic levels memory chip is disabled.
generally used in large memory systems.
figure below shows memory interfacing using absolute decoding.



RD & WR signals: generally P3.6 & P3.7 pins of port 3 are used to generate memory read and memory write signals.
remaining pins of port 3 i.e. P3.0-P3.5 can be used for other functions.

LINEAR AND ABSOLUTE DECODING

i. Absolute Decoding

all higher address lines : decoded to select memory chip for specific logic levels.
for other logic levels memory chip is disabled.
generally used in large memory systems.
figure below shows memory interfacing using absolute decoding.

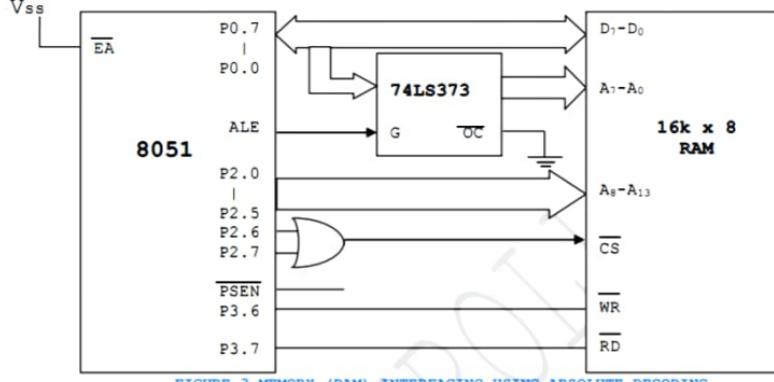


FIGURE 3 MEMORY (RAM) INTERFACING USING ABSOLUTE DECODING.

ii. Linear Decoding (Partial Decoding)

for small systems : individual higher order address lines used to select memory chip.

replacing the hardware by decoding logic.

reducing the cost of decoding, drawback is- multiple addresses. as shown in figure below, A14 line is directly connected to chip select line, A15 line not connected anywhere, kept open. so, status of A15- not considered for generation of chip select signal.

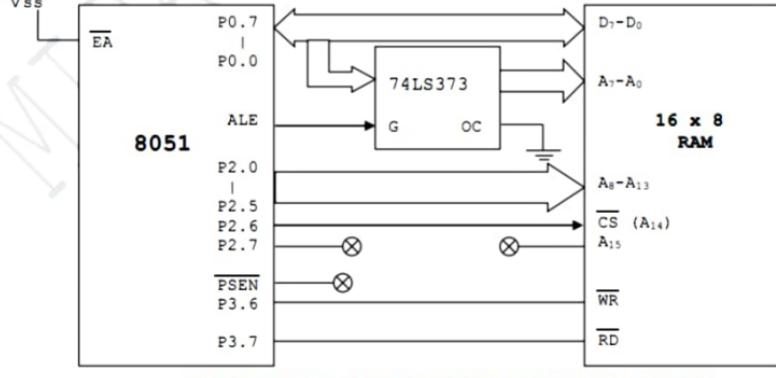


FIGURE 4 MEMORY (RAM) INTERFACING USING LINEAR DECODING.

Address Mapping (Memory Map)

i. Absolute Decoding

Address	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	HEX adrs.	
starting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
end	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH

ii. Linear Decoding

Address	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	HEX adrs.	
starting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
end	x	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH

ii. Linear Decoding (Partial Decoding)
for small systems : individual higher order address lines used to select memory chip.

replacing the hardware by decoding logic.

reducing the cost of decoding, drawback is- multiple addresses. as shown in figure below, A₁₄ line is directly connected to chip select line, A₁₅ line not connected anywhere, kept open. so, status of A₁₅- not considered for generation of chip select signal.

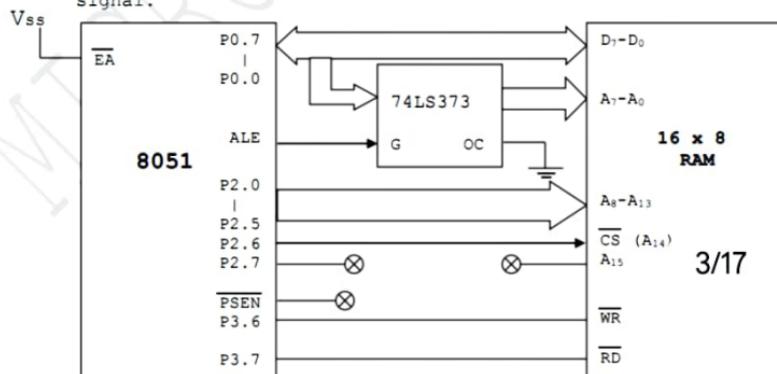


FIGURE 4 MEMORY (RAM) INTERFACING USING LINEAR DECODING.

Address Mapping (Memory Map)

i. Absolute Decoding

Address	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	HEX adrs.
starting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
end	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH

ii. Linear Decoding

Address	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	HEX adrs.
starting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
end	x	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH

Comparison between Full address (Absolute) & Partial address (Linear) Decoding.

Full Address (Absolute) Decoding		Partial Address (Linear) Decoding
i. all higher address lines are decoded to select memory or I/O device.		i. few or individual address lines are decoded to select memory or I/O device.
ii. more hardware : decoding logic.		ii. less hardware : decoding logic. (sometimes none.)
iii. decoding circuit : higher cost.		iii. decoding circuit : less cost.
iv. No multiple addresses.		iv. multiple addresses possible.
v. used in large systems.		iv. used in small systems.

Solved Examples:

Example 1: Design a μController system using 8051. Interface the external RAM of size 16k x 8.

Solution: Given, Memory size: 16k

that means we require $2^8 = 16k :: n$ address lines here $n=14 :: A_0$ to A_{13} address lines are required.

A_{14} and A_{15} are connected through OR gate to CS pin of external RAM. when A_{14} and A_{15} both are low (logic '0'), external data memory(RAM) is selected.

Address Decoding (Memory Map) for 16k x 8 RAM.

Address	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	HEX adrs.
starting	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000H
end	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFFH

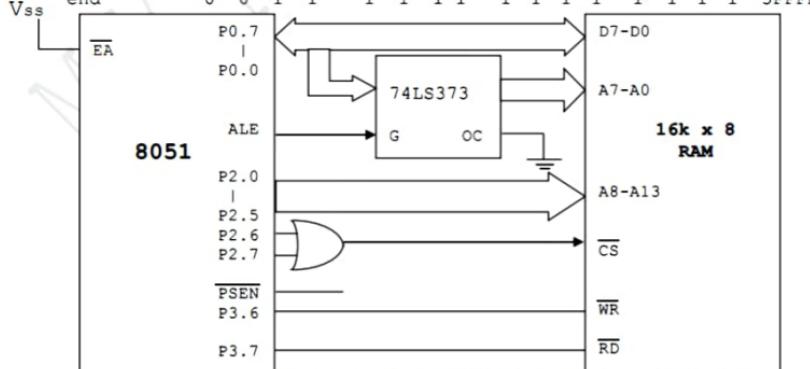


FIGURE 5 16K X 8 MEMORY (RAM) INTERFACING TO μC 8051.



Search tutorials, courses and ebooks...



HTML

CSS

Javascript

SQL

Python

Java

Interfacing DAC with 8051 Microcontroller

Microcontroller

Microprocessor

Computers

In this section we will see how DAC (**Digital to Analog Converter**) using **Intel 8051 Microcontroller**. We will also see the sinewave generation using DAC.

The Digital to Analog converter (DAC) is a device, that is widely used for converting digital pulses to analog signals. There are two methods of converting digital signals to analog signals. These two methods are binary weighted method and R/2R ladder method. In this article we will use the MC1408 (DAC0808) Digital to Analog Converter. This chip uses R/2R ladder method. This

ChatGPT 3.5



the DAC0808 to the desired analog output.

Assembly Language Program (ALP) for 8051:
Write an assembly program to send sine wave values to the DAC0808. The following is a simplified example using 8-bit values for the sine wave:

Assembly

Copy code

ORG 0H ; Set the origin to 0

MOV DPTR, #SineTable ; Load address of sine wave table

MOV R0, #0 ; Initialize index

LOOP:

MOV A, R0 ; Load index into accumulator

MOVX A, @DPTR ; Load sine wave value from table

MOV P1, A ; Send value to DAC0808 or external DAC

ACALL Delay ; Insert a delay for controlling the frequency

INC R0 ; Increment index

CJNE R0, #8, LOOP ; Loop until all values are sent

SJMP LOOP ; Infinite loop

Delay:

; Implement a delay routine as needed

; This can be achieved using software loops or timer interrupts

; Adjust the delay to control the frequency of the sine wave

RET

SineTable:

DB 128, 183, 215, 255, 215, 183 ; 8-bit sine wave values (adjust as needed)

Message



ChatGPT 3.5



LOOP:

```
MOV A, R0 ; Load index into accumulator  
MOVX A, @DPTR ; Load sine wave value from table  
MOV P1, A ; Send value to DAC0808 or external DAC
```

ACALL Delay ; Insert a delay **for** controlling the frequency

INC R0 ; Increment index

CJNE R0, #8, **LOOP** ; Loop until all values are sent

SJMP **LOOP** ; Infinite loop

Delay:

```
; Implement a delay routine as needed  
; This can be achieved using software loops or timer  
interrupts  
; Adjust the delay to control the frequency of the sine  
wave  
RET
```

SineTable:

DB 128, 183, 215, 235, 244, 235, 215, 183 ; 8-bit sine wave values (adjust **as** needed)

END

Note: The values in the SineTable are just placeholders. You may need to replace them with appropriate values for a smoother sine wave. Additionally, you'll need to customize the delay routine based on your specific requirements.

Remember to refer to the datasheets of the 8051 microcontroller and DAC0808 for detailed pin configurations and specifications.

Message



[Log In](#)[Menu](#)[Computer Science Courses](#) / [Course](#) / [Chapter](#)

IO Ports & Buses: Uses & Examples

Lesson Transcript

Instructor: Sudha Aravindan

Computers are equipped with a series of ports and buses for input and output of information. Review basic computer terms, such as processor and memory, as well as the types of ports and buses, including peripherals, parallel ports, SCSI, and USB.

[Get started](#)

Table of Contents

- [Ports & Bus](#)
- [Processor](#)
- [Memory](#)
- [Peripherals & Parallel Port](#)
- [SCSI & USB](#)
- [Lesson Summary](#)

[Show](#)

Ports & Bus

Katie is a high-level gamer. One of her games requires both the left and right buttons of her trackpad to be clicked at the same time. This is difficult to do, so she had to use an external mouse instead. She had a wired mouse in her laptop bag with a USB connector. Katie looked for a

= ChatGPT 3.5



mention the control signals required for accessing peripheral devices.



CHATGPT

I/O expansion buses are pathways that facilitate communication between a computer's central processing unit (CPU) and peripheral devices. Common examples include PCI, USB, and SATA. Control signals are crucial for accessing peripherals, including:

1. **Read/Write Control Signals:** Indicate whether data transfer is a read or write operation.
2. **Address Lines:** Specify the memory address for data transfer, helping identify the peripheral.
3. **Data Lines:** Carry the actual data between the CPU and peripheral device.
4. **Chip Select (CS):** Enables or disables communication with a specific peripheral.
5. **Interrupt Request (IRQ):** Notifies the CPU of an event or data ready condition from a peripheral.
6. **Clock Signals:** Synchronize data transfer between the CPU and peripherals.
7. **Reset Signals:** Reset or reset the peripheral device.

Message



ChatGPT 3.5



facilitate communication between a computer's central processing unit (CPU) and peripheral devices. Common examples include PCI, USB, and SATA. Control signals are crucial for accessing peripherals, including:

- 1. Read/Write Control Signals:** Indicate whether data transfer is a read or write operation.
- 2. Address Lines:** Specify the memory address for data transfer, helping identify the peripheral.
- 3. Data Lines:** Carry the actual data between the CPU and peripheral device.
- 4. Chip Select (CS):** Enables or disables communication with a specific peripheral.
- 5. Interrupt Request (IRQ):** Notifies the CPU of an event or data ready condition from a peripheral.
- 6. Clock Signals:** Synchronize data transfer between the CPU and peripherals.
- 7. Reset Signals:** Initialize or reset the peripheral device.

These control signals ensure effective communication and coordination between the CPU and connected peripherals.

Screenshot saved

Message





Home → 8051 Microcontroller, General

8051 Microcontroller Instruction Set

April 6, 2021 By Ravi Teja

In the previous tutorial on 8051 Microcontroller, we have seen the Introduction of 8051, the Architecture of 8051 and the Memory Organization of the 8051 Microcontroller. Continuing further, we will take a look at the 8051 Microcontroller Instruction Set and the 8051 Addressing Modes in this tutorial.

[Outline](#)



Introduction to 8051 Microcontroller Instruction Set

Writing a Program for any Microcontroller consists of giving commands to the



INSTRUCTION SET OF 8051

Instruction Set of 8051

The process of writing program for the microcontroller mainly consists of giving instructions (commands) in the specific order in which they should be executed in order to carry out a specific task. As electronics cannot "understand" what for example an instruction "if the push button is pressed- turn the light on" means, then a certain number of simpler and precisely defined orders that decoder can recognise must be used. All commands are known as INSTRUCTION SET. All microcontrollers compatible with the 8051 have in total of 255 instructions, i.e. 255 different words available for program writing.

At first sight, it is imposing number of odd signs that must be known by heart. However, It is not so complicated as it looks like. Many instructions are considered to be "different", even though they perform the same operation, so there are only 111 truly different commands. For example: ADD A,R0, ADD A,R1, ... ADD A,R7 are instructions that perform the same operation (addition of the accumulator and register). Since there are 8 such registers, each instruction is counted separately. Taking into account that all instructions perform only 53 operations (addition, subtraction, copy etc.) and most of them are rarely used in practice, there are actually 20-30 abbreviations to be learned, which is acceptable.

3.1 Types of instructions



Menu ▾

Login



Search tutorials, courses and ebooks...



HTML

CSS

Javascript

SQL

Python

Java

Addressing modes of 8051

Microprocessor 8085

In this section, we will see different addressing modes of the **8051 microcontrollers**. In 8051 there are 1-byte, 2-byte instructions and very few 3-byte instructions are present. The opcodes are 8-bit long. As the opcodes are 8-bit data, there are 256 possibilities. Among 256, 255 opcodes are implemented.

The clock frequency is 12MHz, so 64 instruction types are executed in just 1 μ s, and rest are just 2 μ s. The Multiplication and Division operations take 4 μ s to execute.

TCON & TMOD of 8051 μ C

- ❑ 8051 has Two 16 bits Timers T0 & T1, working as up counters. T0 and T1 is further divided into 8bits of registers TH0-TL0 and TH1-TL1.
- ❑ If T0 & T1 counts internal clock pulses, then it is timer.
- ❑ If T0 & T1 counts External clock pulses, then it is Counter.
- ❑ Timer action is controlled by TCON and TMOD registers.
- ✓ **TCON register – {Bit Address TCON.7 to TCON.0}**

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
-----	-----	-----	-----	-----	-----	-----	-----

- ❑ **TF1 and TF0 – Timer Overflow Flag**
- ❑ SET 1 = When timer 1 and timer 0 overflows, when timer roll overs to all 0's.
- ❑ Clear 0 = When processor executes ISR after overflow. {For Timer 1 ISR address is 001BH and Timer 0 ISR address is 000BH}
- ❑ **TR1 and TR0 – Timer Run Control Bit**
- ❑ SET 1 = Start Counting Timer.
- ❑ Clear 0 = Halts Timer.
- ❑ **IE1 and IE0 – External Interrupt bit**
- ❑ SET 1 = when 8051 receives interrupt on INT1 and INT0.
- ❑ Clear 0 = when ISR executed. {For INT1 ISR address is 0013H and INT0 ISR address is 0003H}

- ❑ **IT1 and IT0 – External Interrupt Type bit**
- ❑ SET 1 = INT1 and INT0 must be –ve edge trigger.
- ❑ Clear 0 = INT1 and INT0 must be low level trigger.
- ✓ **TMOD register – Timer Mode Control register**



- ❑ **C/T – Counter / Timer Type bit**
- ❑ SET 1 = Acts as Counter. {External Frequency on T1 & T0}
- ❑ Clear 0 = Acts as Timer. {Internal Frequency Fosc/12}
- ❑ **GATE – Gate Enable Control bit**
- ❑ SET 1 = Timer Controlled by Hardware. {INTX Signal}
- ❑ Clear 0 = Timer independent on INTX signal.

- ❑ **M1 & M0 – Mode Control bits**

M1	M0	Timer Mode
0	0	Timer Mode 0
0	1	Timer Mode 1
1	0	Timer Mode 2
1	1	Timer Mode 3



Menu ▾

Login



Search tutorials, courses and ebooks...



HTML

CSS

Javascript

SQL

Python

Java

Interrupt structure of 8051

Microprocessor 8085

Now in this section, we will see the interrupt structure of Intel 8051 microcontroller.

Interrupts are basically the events that temporarily suspend the main program, pass the control to the external sources and execute their task. It then passes the control to the main program where it had left off.

8051 has five interrupts. These interrupts are INT0, INT1, T0, T1, TI/RI. All of the interrupts can be enabled or disabled by using the IE (interrupt enable) register.

8051 Memory Organization

download.mikroe.com



8051 Memory Organization

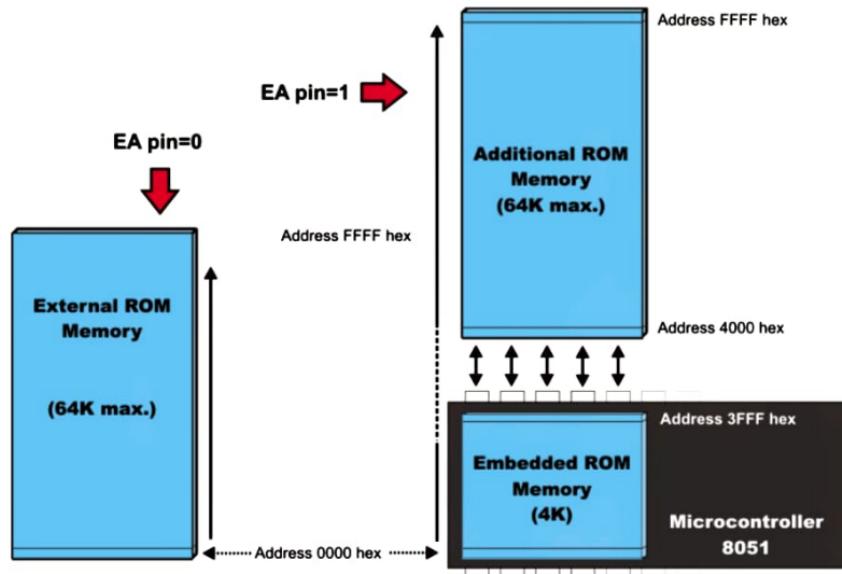
The 8051 microcontroller's memory is divided into Program Memory and Data Memory. Program Memory (ROM) is used for permanent saving program being executed, while Data Memory (RAM) is used for temporarily storing and keeping intermediate results and variables.

Program Memory (ROM)

Program Memory (ROM) is used for permanent saving program (CODE) being executed. The memory is read only. Depending on the settings made in compiler, program memory may also be used to store constant variables. The 8051 executes programs stored in program memory only. `code` memory type specifier is used to refer to program memory.

8051 memory organization allows external program memory to be added.

How does the microcontroller handle external memory depends on the pin EA logical state.



Internal Data Memory

Up to 256 bytes of internal data memory are available depending on the 8051 derivative. Locations available to the user occupy addressing space from 0 to 7Fh, i.e. first 128 registers and this part of RAM is divided in several blocks. The first 128 bytes of internal data memory are both directly and indirectly addressable. The upper 128 bytes of data memory (from 0x80 to 0xFF) can be addressed only indirectly.

Since internal data memory is used for CALL stack also and there is only 256 bytes split over few different memory areas



- ✓ In-store shopping
- ✗ In-store pickup



i
x



☰ Electronics and Communications



[Home](#) > Microcontrollers

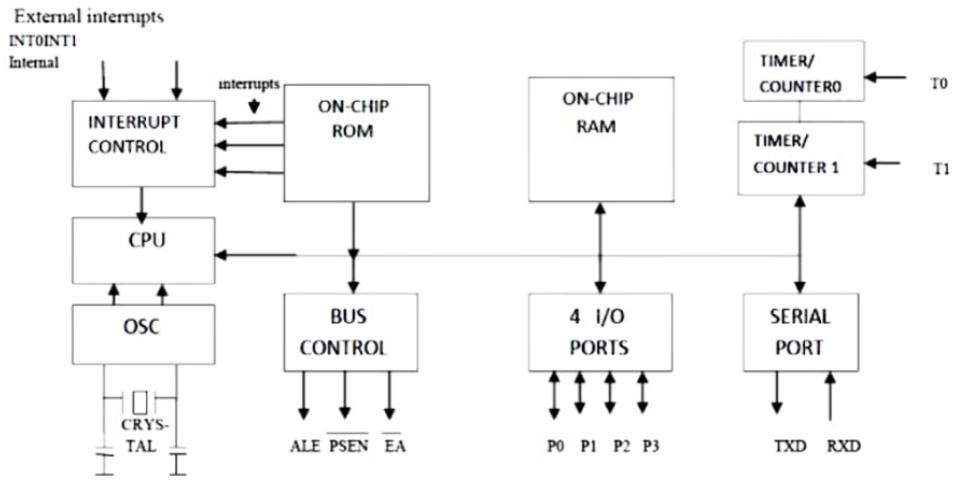
Block Diagram of Microcontroller 8051



by [Sreejith Hrishikesan](#) • July 22, 2019

0

Block Diagram of Microcontroller 8051



CPU – The CPU (Central Processing Unit) comprising of ALU (Arithmetic and Logic Units) and Control units

What is Embedded System Design : Steps in the Design Process

An Embedded system is a controller, which controls many other electronic devices. It is a combination of embedded hardware and software. There are two types of embedded systems **microprocessors** and **micro-controller**. Micro-processor is based on von Neumann model/architecture (where program + data resides in the same memory location), it is an important part of the computer system, where external processors and peripherals are interfaced to it. It occupies more area and has more power consumption. The application of the microprocessor is personal computers. This article discusses steps involving in embedded system design.



70

Different types of Microcontroller Programming used in Embedded Systems

155

14

28

Share

Tweet

Pin

LIVE CHAT



Types of Microcontroller

Need Help? [Chat with us](#)



Types of Microcontroller

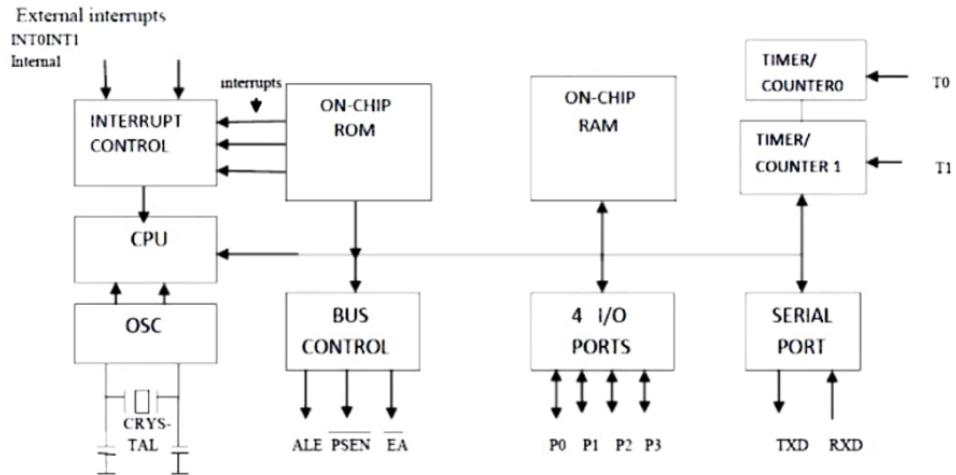
A microcontroller is a small computer on a single integrated circuit. It

[≡ Electronics and Communications](#)[Home](#) > Microcontrollers

Block Diagram of Microcontroller 8051

by **Sreejith Hrishikesan** • July 22, 2019

Block Diagram of Microcontroller 8051



CPU – The CPU (Central Processing Unit) comprising of ALU (Arithmetic and Logic Units) and Control units

ALU - Arithmetic and Logic Unit performing the arithmetic and logical operations. These operations are multiplication, addition, subtraction, logical AND, OR etc. To do these operations one operand must be in Accumulator, and another may in B register or in general purpose register. The ALU

Table 1. MSP430™ and MSP432™ Core Comparison

Name	MSP430™	MSP432™
Data size	16 bit	32 bit
Program bus width	16-bit (CPU) or 20-bit (CPUX) address bus	32 bit
Bus type	16-bit MSP430 bus	AHB
Architecture	Von Neumann (Princeton): data op and instruction fetch share same bus	Harvard: separate data and instruction buses
Instruction Set	RISC, MSP430 proprietary	RISC, thumb and thumb2
Instruction size	16-bit (and 16-bit for each operand)	16-bit and 32-bit
Instruction Cycle (typical)	1-4 cycles	1-2 cycles
Pipeline	None	3-stage pipeline
Prefetch buffer	128 bit	128 bit
Power Modes	Active, LPM0-LPM4, LPMx.5	Active, Low Frequency, LPM0, LPM3, LPMx.5
Debug Interface	MSP430 4-wire JTAG, and 2-wire SBW	ARM JTAG in 4-wire and 2-wire modes
Math support	Hardware Multiplier (MPY)	Hardware Multiplier and Divider, DSP extension, and integrated FPU