UNIT- II

Syllabus

RISC vs CISC design philosophy, Harvard and Von-Neuman architectures, I/O – sensors and actuators, Communication Interfaces – Onboard (I2C, SPI, UART, 1-wire interface, parallel interface), External (RS-232 & RS-485, USB, IEEE 1394, IrDA, Bluetooth, Wi-Fi, ZigBee, GPRS. Application specific circuitry – reset, brownout protection, oscillator, RTC, Watchdog timer. Embedded firmware.

CISC and RISC design philosophy:

The architecture of the Central Processing Unit (CPU) operates the capacity to function from "Instruction Set Architecture" to where it was designed. The architectural design of the CPU is Reduced instruction set computing (RISC) and Complex instruction set computing (CISC). CISC has the capacity to perform multi-step operations or addressing modes within one instruction set. It is the CPU design where one instruction works several low-level acts. For instance, memory storage, loading from memory, and an arithmetic operation. Reduced instruction set computing is a Central Processing Unit design strategy based on the vision that basic instruction set gives a great performance when combined with a microprocessor architecture which has the capacity to perform the instructions by using some microprocessor cycles per instruction. This article discusses the difference between the RISC and CISC architecture. The hardware part of the Intel is named as Complex Instruction Set Computer (CISC), and Apple hardware is Reduced Instruction Set Computer (RISC).

What is RISC?

A reduced instruction set computer is a computer which only uses simple commands that can be divided into several instructions which achieve low-level operation within a single CLK cycle, as its name proposes "Reduced Instruction Set".

RISC Architecture

The term RISC stands for ''Reduced Instruction Set Computer''. It is a CPU design plan based on simple orders and acts fast.

This is small or reduced set of instructions. Here, every instruction is expected to attain very small jobs. In this machine, the instruction sets are modest and simple, which help in comprising more complex commands. Each instruction is of the similar length; these are wound together to get compound tasks done in a single operation. Most commands are completed in one machine cycle. This pipelining is a crucial technique used to speed up RISC machines.

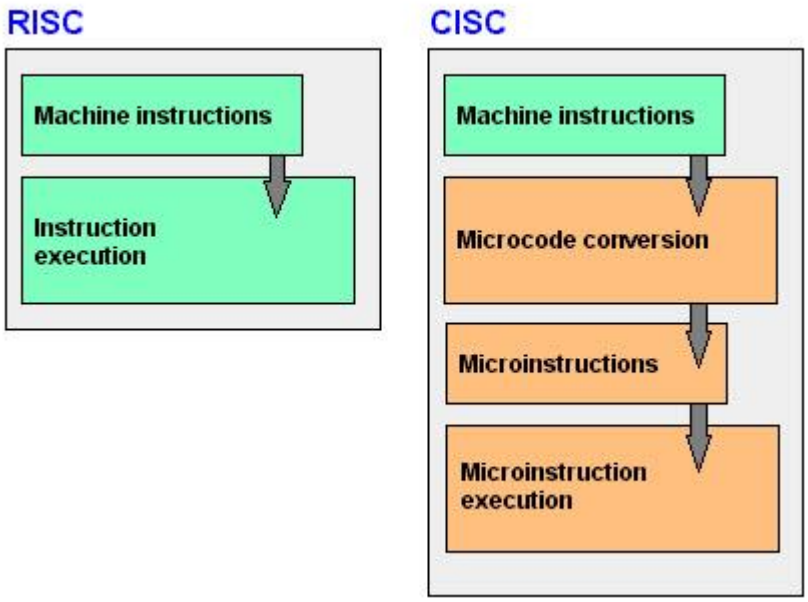
What is CISC?

A complex instruction set computer is a computer where single instructions can perform numerous low-level operations like a load from memory, an arithmetic operation, and a memory store or are accomplished by multi-step processes or addressing modes in single instructions, as its name proposes "Complex Instruction Set ".

CISC Architecture

The term CISC stands for ‘’Complex Instruction Set Computer’’. It is a CPU design plan based on single commands, which are skilled in executing multi-step operations.

CISC computers have small programs. It has a huge number of compound instructions, which takes a long time to perform. Here, a single set of instruction is protected in several steps; each instruction set has additional than 300 separate instructions. Maximum instructions are finished in two to ten machine cycles. In CISC, instruction pipelining is not easily implemented.



CISC vs RISC:

The following points differentiate a CISC from a RISC −

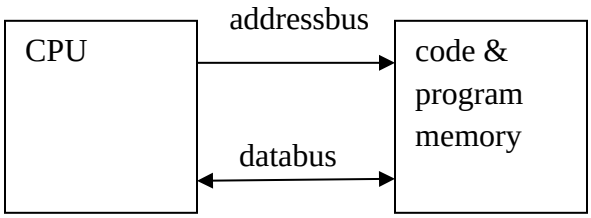
RISC	CISC
1. RISC stands for Reduced Instruction Set Computer.	1. CISC stands for Complex Instruction Set Computer.
2. RISC processors have simple instructions taking about one clock cycle. The average clock cycle per instruction (CPI) is 1.5	2. CSIC processor has complex instructions that take up multiple clocks for execution. The average clock cycle per instruction (CPI) is in the range of 2 and 15.
3. Performance is optimized with more focus on software	3. Performance is optimized with more focus on hardware.
4. It has no memory unit and uses a separate hardware to implement instructions..	4. It has a memory unit to implement complex instructions.
5. It has a hard-wired unit of programming.	5. It has a microprogramming unit.
6. The instruction set is reduced i.e. it has only a few instructions in the instruction set. Many of these instructions are very primitive.	**6.** The instruction set has a variety of different instructions that can be used for complex operations.
7. The instruction set has a variety of different instructions that can be used for complex operations.	**7.** CISC has many different addressing modes and can thus be used to represent higher-level programming language statements more efficiently.
8. Complex addressing modes are synthesized using the software.	8. CISC already supports complex addressing modes
9. Multiple register sets are present	9. Only has a single register set
10. RISC processors are highly pipelined	10. They are normally not pipelined or less pipelined
11. The complexity of RISC lies with the compiler that executes the program	11. The complexity lies in the microprogram
12. Execution time is very less	12. Execution time is very high
13. Code expansion can be a problem	13. Code expansion is not a problem
14. Decoding of instructions is simple.	14. Decoding of instructions is complex
15. It does not require external memory for calculations	15. It requires external memory for calculations
16. The most common RISC microprocessors are Alpha, ARC, ARM, AVR, MIPS, PA-RISC, PIC, Power Architecture, and SPARC.	16. Examples of CISC processors are the System/360, VAX, PDP-11, Motorola 68000 family, AMD and Intel x86 CPUs.
17. RISC architecture is used in high-end applications such as video processing, telecommunications and image processing.	17. CISC architecture is used in low-end applications such as security systems, home automation, etc.

Von-Neumann Architecture & Harvard Architecture:

When data and code lie in different memory blocks, then the architecture is referred as **Harvard architecture**. In case data and code lie in the same memory block, then the architecture is referred as **Von Neumann architecture**.

Von Neumann Architecture:

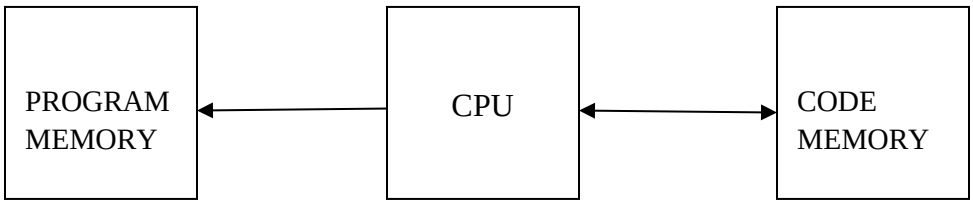
The Von Neumann architecture was first proposed by a computer scientist John von Neumann. In this architecture, one data path or bus exists for both instruction and data. As a result, the CPU does one operation at a time. It either fetches an instruction from memory, or performs read/write operation on data. So an instruction fetch and a data operation cannot occur simultaneously, sharing a common bus.



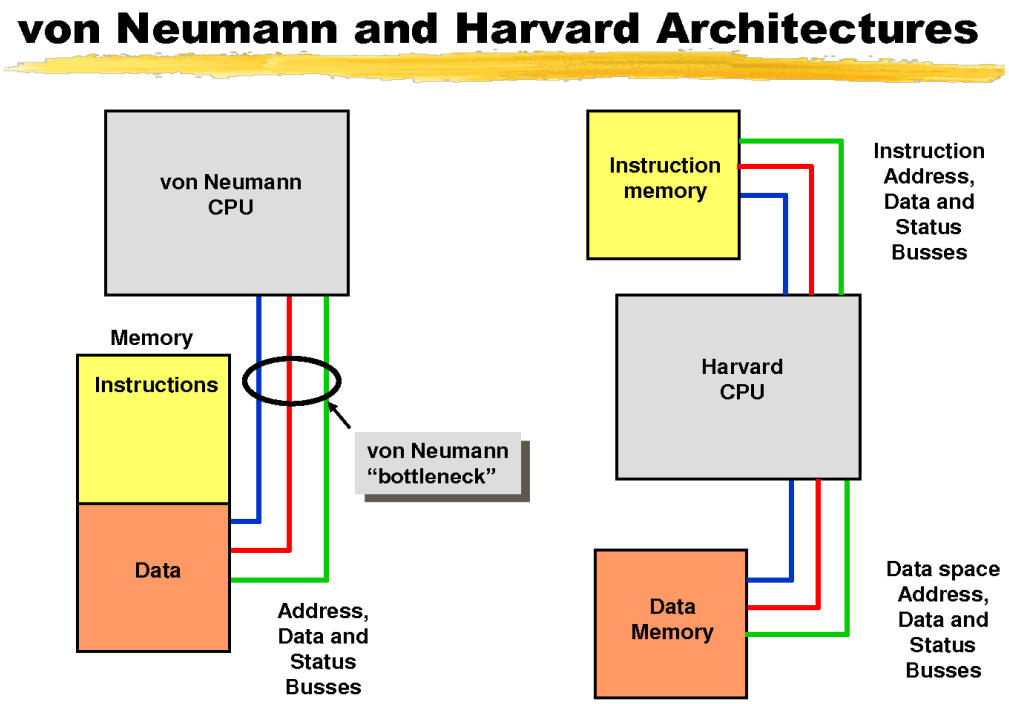
Von-Neumann architecture supports simple hardware. It allows the use of a single, sequential memory. Today's processing speeds vastly outpace memory access times, and we employ a very fast but small amount of memory (cache) local to the processor.

Harvard Architecture:

The Harvard architecture offers separate storage and signal buses for instructions and data. This architecture has data storage entirely contained within the CPU, and there is no access to the instruction storage as data. Computers have separate memory areas for program instructions and data using internal data buses, allowing simultaneous access to both instructions and data. Programs needed to be loaded by an operator; the processor could not boot itself. In a Harvard architecture, there is no need to make the two memories share properties.



Von-Neumann Architecture vs Harvard Architecture:



The following points distinguish the Von Neumann Architecture from the Harvard Architecture.

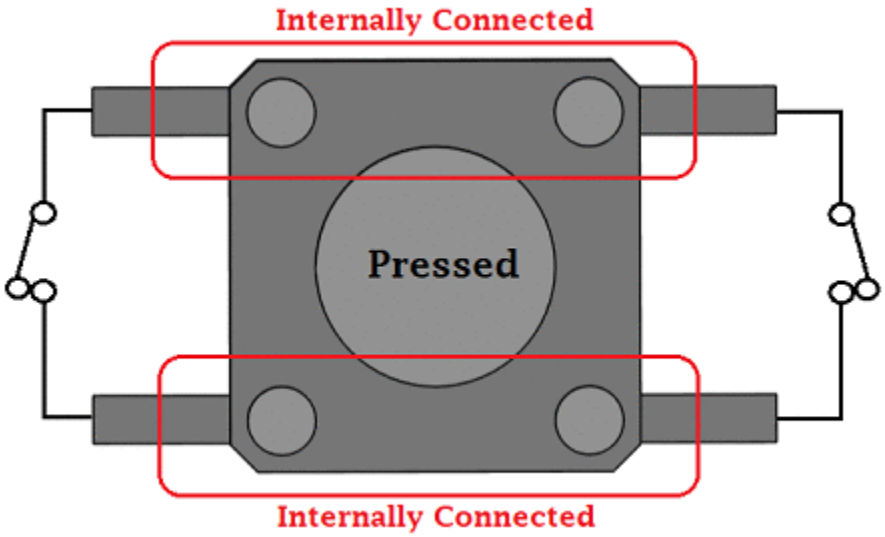
Von-Neumann Architecture	**Harvard Architecture**
Single memory to be shared by both code and data.	Separate memories for code and data.
Processor needs to fetch code in a separate clock cycle and data in another clock cycle. So it requires two clock cycles.	Single clock cycle is sufficient, as separate buses are used to access code and data.
Pipelining is difficult to implement	Pipelining is easy to implement
Slower in speed, thus more time-consuming.	Higher speed, thus less time consuming.
Less power consumption	High power consumption

Simple in design.	Complex in design.
Cost is low	Cost is high
Used in personal computers, laptops and workstations	Used in microcontrollers and signal processors

I/O – sensors and actuators:

Input & Output devices:

Push Button:



Push-Buttons are normally-open **tactile switches**. Push buttons allow us to power the circuit or make any particular connection only when we press the button. Simply, it makes the circuit connected when pressed and breaks when released. These are the most common buttons which we see in our daily life electronic equipment's.

7-Segment LED Display:

The 7-segment LED display is an output device for displaying alpha numeric characters. It contains 8 light-emitting diode (LED) segments arranged in a special fort. Out of the 8 LED segments. 7 are used for displaying alpha numeric characters and I is used for representing 'decimal point' in decimal number display. Figure 2.14 explains the arrangement of LED segments in a 7-segment LED display. The LED segments arc named A to G and the decimal

point LED segment is named as DP. The LED segments A to G and DP should be lit accordingly to display numbers and characters. For example, for displaying the number 4, the segments F, 0, B and C arc lit. For dis-playing 3, the segments A, B. C. D, G and DP are lit. For displaying the character 'd', the segments B. C, D, E and G arc lit.

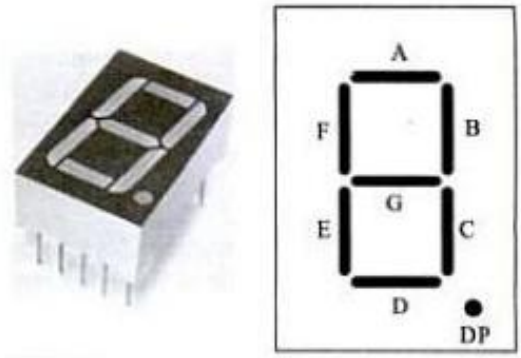


Fig. 2.14 **7-Segment LED Display**

All these 8 LED segments need to be connected to one port of the processor/controller for displaying alpha numeric digits. The 7-segment LED displays are available in two different configurations, namely: Common Anode and Common Cathode. In the common anode configuration, the anodes of the 8 segments arc connected commonly whereas in the common cathode configuration, the 8 LED segments share a common cathode line.

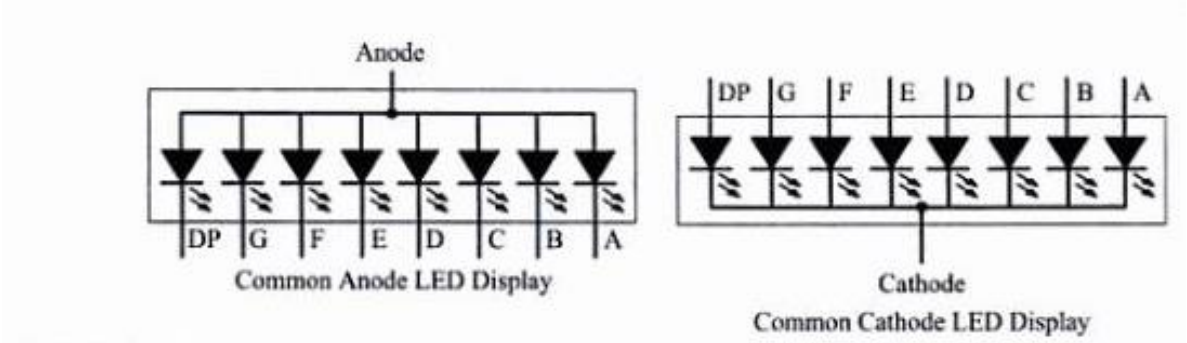


Fig. 2.15 **Common anode and cathode configurations of a 7-segment LED Display**

LED:

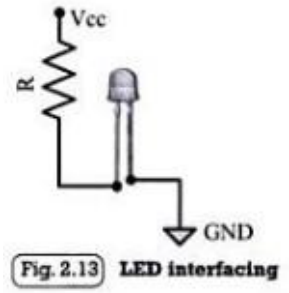


Fig. 2.13 **LED interfacing**

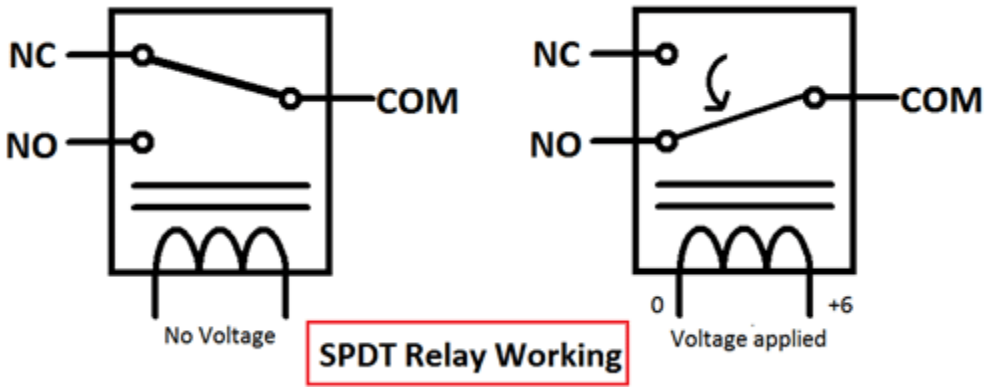
Light Emitting Diode (LED) Light Emitting Diode (LED) is an important output device for visual indication in any embedded system. LED can be used as an indicator for the status of various signals or situations. Typical examples arc indicating the presence of power conditions

like 'Device ON', 'Battery low' or 'Charging of battery' for a battery operated handheld embedded devices. Light Emitting Diode is a p-n junction diode and it contains an anode and a cathode. For proper functioning of the LED. the anode of it should be connected to +vc terminal of the supply voltage and cathode to the -vc terminal of supply voltage. The current flowing through the LED must be limited to a value below the maximum current that it can conduct. A resister is used in series between the power supply and the LED.

Relay:

Relays are switches that open and close circuits electromechanically or electronically. Relays control one electrical circuit by opening and closing contacts in another circuit. As relay diagrams show, when a relay contact is normally open (NO), there is an open contact when the relay is not energized. When a relay contact is Normally Closed (NC), there is a closed contact when the relay is not energized. In either case, applying electrical current to the contacts will change their state.

Relays are generally used to switch smaller currents in a control circuit and do not usually control power consuming devices except for small motors and Solenoids that draw low amps. Nonetheless, relays can "control" larger voltages and amperes by having an amplifying effect because a small voltage applied to a relays coil can result in a large voltage being switched by the contacts.



Sensor:

We live in a World of Sensors. You can find different types of Sensors in our homes, offices, cars etc. working to make our lives easier by turning on the lights by detecting our presence, adjusting the room temperature, detect smoke or fire, make us delicious coffee, open garage doors as soon as our car is near the door and many other tasks. All these and many other automation tasks are possible because of Sensors.

Let us consider a measurement system. It is composed of an input device which senses the environment or surrounding to generate an output and, a signal processing block which processes the signal from input device and an output device which presents the signal to human or machine operator in a more readable and usable form.

A sensor is a device that responds to any change in physical phenomena or environmental variables like heat, pressure, humidity, movement etc.

Another unique definition of a Sensor is as follows: It is a device that converts signals from one energy domain to electrical domain.

Sensor classification

(1) Based on the quantity being measured

- Temperature: Resistance Temperature Detector (RTD), Thermistor, Thermocouple
- Pressure: Bourdon tube, manometer, diaphragms, pressure gauge
- Force/ torque: Strain gauge, load cell
- Speed/ position: Tachometer, encoder, LVDT
- Light: Photo-diode, Light dependent resistor etc

(2) Active and passive sensors: Based on power requirement sensors can be classified as active and passive. Active sensors are those which do not require external power source for their functioning. They generate power within themselves to operate and hence called as self-generating type. The energy for functioning is derived from the quantity being measured. For example piezoelectric crystal generate electrical output (charge) when subjected to acceleration. Passive sensors require external power source for their functioning. Most of the resistive, inductive and capacitive sensors are passive (just as resistors, inductors and capacitors are called passivedevices).

(3) Analog and digital sensor: An analog sensor converts the physical quantity being measured to analog form (continuous in time). Thermocouple, RTD, Strain gauge are called analog sensors. A digital sensor produces output in the form of pulses (1's and 0's). Encoders are example of digitalsensors.

(4) Inverse sensors: There are some sensors which are capable of sensing a physical quantity to convert it to other form and also sense the output signal form to get back the quantity in original form. For example a piezoelectric crystal when subjected to vibration generates voltage. At the same time when a piezo crystal is subjected to varying voltage they begin to vibrate. This property make them suitable to use in microphone and speakers.

Different types of sensors:

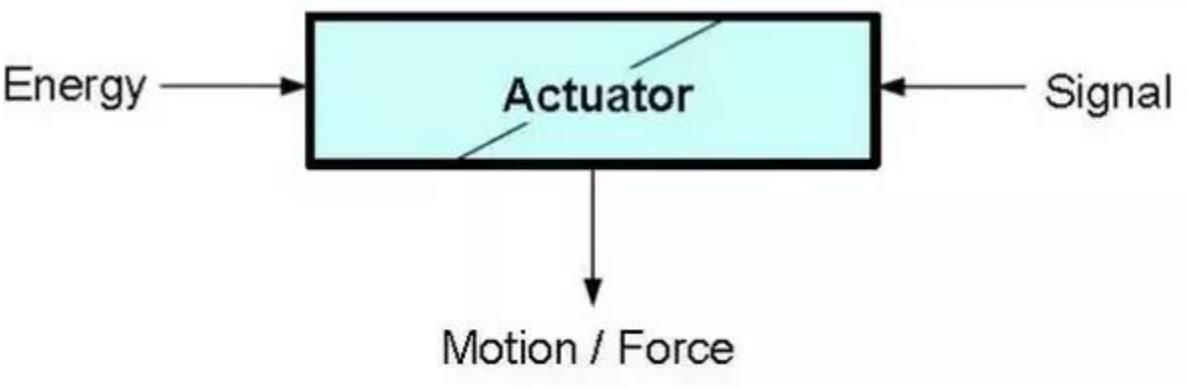
The following is a list of different types of sensors that are commonly used in various applications. All these sensors are used for measuring one of the physical properties like Temperature, Resistance, Capacitance, Conduction, Heat Transfer etc.

- Temperature Sensor - it measures the changes in the temperature.In a Temperature Sensor, the changes in the Temperature correspond to change in its physical property like resistance or voltage.

- Proximity Sensor - A Proximity Sensor is a non-contact type sensor that detects the presence of an object. Proximity Sensors can be implemented using different techniques like Optical (like Infrared or Laser), Ultrasonic, Hall Effect, Capacitive, etc.
- Accelerometer - Accelerometers are devices that measure acceleration, which is the rate of change of the velocity of an object. They measure in meters per second squared (m/s^2) or in G-forces (g). A single G-force for us here on planet Earth is equivalent to 9.8 m/s^2, but this does vary slightly with elevation (and will be a different value on different planets due to variations in gravitational pull). Accelerometers are useful for sensing vibrations in systems or for orientation applications.
- IR Sensor (Infrared Sensor) - IR Sensors or Infrared Sensor are light based sensor that are used in various applications like Proximity and Object Detection. Different applications where IR Sensor is implemented are Mobile Phones, Robots, Industrial assembly, automobiles etc.
- Pressure Sensor - Pressure is defined as the applied force by a liquid or gas on a surface and it is usually measured in units of force per unit of surface area. Common units are Pascal (Pa), Bar (bar), N/mm2 or psi (pounds per square inch).
- Light Sensor - The **light sensor** is a passive devices that convert this "**light** energy" whether visible or in the infra-red parts of the spectrum into an electrical signal output. **Light sensors** are more commonly known as "Photoelectric Devices" or "Photo **Sensors**" because the convert **light** energy (photons) into electricity (electrons).
- Ultrasonic Sensor - An Ultrasonic Sensor is a non-contact type device that can be used to measure distance as well as velocity of an object. An Ultrasonic Sensor works based on the properties of the sound waves with frequency greater than that of the human audible range.
- Smoke, Gas and Alcohol Sensor – Used to detect smoke, gas and alcohol levels
- Touch Sensor – A **touch sensor** is a type of equipment that captures and records physical **touch** or embrace on a device and/or object. It enables a device or object to detech **touch**, typically by a human user or operator. A **touch sensor** may also be called a **touch** detector.
- Humidity Sensor - **Humidity Sensor** is one of the most important devices that has been widely in consumer, industrial, biomedical, and environmental etc. applications for measuring and monitoring **Humidity**. **Humidity** is **defined** as the amount of water present in the surrounding air

Actuator:

An **actuator** is something that converts energy into motion. It also can be used to apply a force. An actuator typically is a mechanical device that takes energy — usually energy that is created by air, electricity or liquid — and converts it into some kind of motion. That motion can be in virtually any form, such as blocking, clamping or ejecting. Actuators typically are used in manufacturing or industrial applications and might be used in devices such as motors, pumps, switches and valves.



Hydraulic: In a hydraulic actuator, a cylinder or fluid-based motor uses the power of hydraulics to create mechanical action. The motion can be either straight, rotating or oscillating. The fact that liquids don't compress well makes a hydraulic actuator very powerful. The hydraulic power can be applied to one or both sides of a piston.

Pneumatic: With a pneumatic actuator, a vacuum or compressed air is used to make energy into action. They are good for making a large linear or rotating motion with a small amount of pressure. Because of the use of high pressure to power them, pneumatic actuators respond quickly and are good for stopping and starting and so are good for main engine controls.

Electrical: The most common kind of actuator is an electrical one which can take electrical energy from DC or AC and turn it into mechanical energy. Because this is a clean and easily available technology, electrical actuators are popular for many types of industry and things like multi-turn valves. Examples for electric actuators are as follows:
AC motor -- An **AC motor** is an electric motor driven by an alternating current (AC).
DC motor - A **DC motor** is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy.
Servo motor - A **servo motor** is an electrical device which can push or rotate an object with great precision. If you want to rotate and object at some specific angles or distance, then you use **servo motor**.
Stepper motor: A **stepper motor**, also known as **step motor** or stepping motor, is a brushless DC electric motor that divides a full rotation into a number of equal steps.

Thermal (Magnetic): Another name for thermal actuators is "shape memory alloys" (SMMs) because these types of actuators use SMMs to create high power density energy for commercial applications. Advantages of thermal actuators are that they are cost-effective, small and lightweight.

Mechanical: Using pulleys, gears, rails and chains, a mechanical actuator converts the rotating motion of physical objects into linear motion. One common example is rack and pinion steering systems.

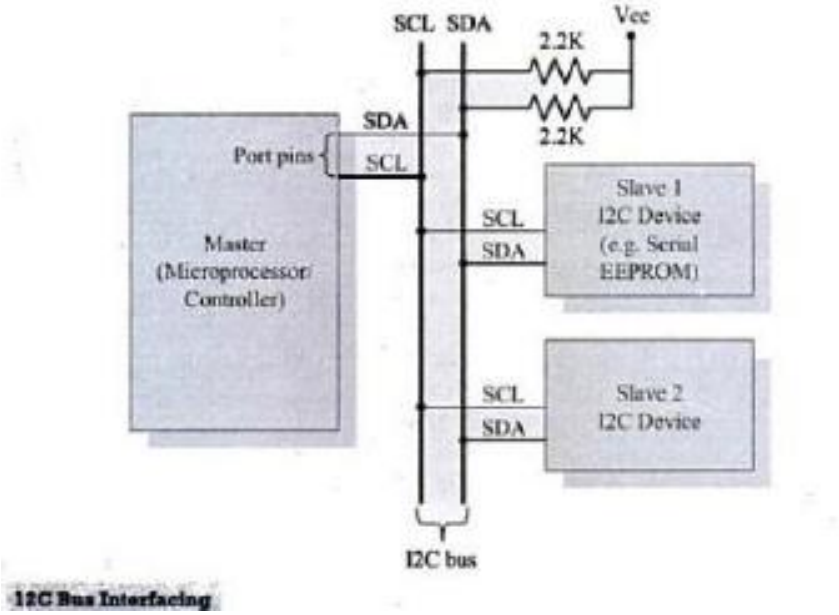
COMMUNICATION INTERFACE

Communication interface is essential for communicating with various subsystems of the embedded system and with the external world. For an embedded product, the communication interface can be viewed in two different perspectives; namely; Device/board level communication interface (Onboard Communication Interface) and Product level communication interface (External Communication Interface). Embedded product is a combination of different types of components arranged on a printed circuit board (PCB).

Onboard Communication Interfaces

Onboard Communication Interface refers to the different communication channels, buses for interconnecting the various integrated circuits and other peripherals within the embedded system.

Inter Integrated Circuit (12C) Bus: The Inter Integrated Circuit Bus (I2C-Pronounced **I square C') is a synchronous Bi-directional half duplex (one-directional communication at a given** point of time) two wire serial interface bus. The concept of I2C bus was developed by 'Philips semiconductors in the early 1980s. The original intention of 12C was to provide an easy way of connection between a microproccesor/rnicro***controller system and*** the peripheral chips in television sets. The l2C bus comprise of two bus lines, namely; Serial Clock—SCL and Serial Data-SDA. SCL line is responsible for generating synchronization clock pulses and SDA is responsible for transanitting the serial data across devices. I2C bus is a shared bus system to which army number of I2C devices can be connected. Devices connected to the 12C bus can **act** as either 'Maslen' device or 'Slave' device. The 'Master' device is responsible for controlling the communication by initiating/terminating data transfer. sending data and generating necessary synchronisation clock pulses. 'Slave' devices wait for the commands from the master and respond upon receiving the commands .'Master' and 'Slave' devices can act as either transmitter or receiver. Regardless whether a master is acting as transmitter or receiver. the synchronization clock signal is generated by the 'Master' device only. I2C supports multi masters on the same bus The following bus interface diagram shown in Fig. illustrates the connection of master and slave devices on the 12C bus.



12C Bus Interfacing

The sequence of operations for communicating with an I2C slave device is listed below:

I. The master device pulls the clock line (SCL) of the bus to 'HIGH'
2. The master device pulls the data line (SDA) 'LOW', when the SCL line is at logic 'HIGH' (This is the 'Stan' condition for data transfer) •
3. The master device sends the address (7 bit or 10 bit wide) of the 'slave' device to which it wants to communicate, over the SDA lion. Clock pubes are generated at the SCL line for synchronizing the bit reception by the slave device. The MSB of the data is always transmitted first. The data in the bus is valid during the 'HIGH' period of the clock signal
.4.The master device scads the Read a Write bit (Bit value = I Read operation; Bit value =. 0 Write operation) according to the requirement
5. The master device waits for the acknowledgement bit from the slave device whose address is sent on the bus along with the Readitirite operation command. Slave devices connected to the bus compares the address received with the address assigned to them
6. The slave device with the address requested by the master device responds by sending an ac-knowledge bit (Bit value - I) over the SIM line
7. Upon receiving the acknowledge bit the Master device sends the &bit data to the slave device over SDA line. if the requested operation is 'Write to device'. If die requested operation is 'Read from device', the slave device sends data to the Mailer ova the SDA line
8. The master device waits for the acknowledgement bit (torn the device upon byte transfer complete for a write operation and sends an acknowledge hit to the Slave device for a read operation
9. The master device terminates the transfer by pulling the SDA line 'HIGH' when the clock line SCL is at logic (Indicating the 'STOP' condition)

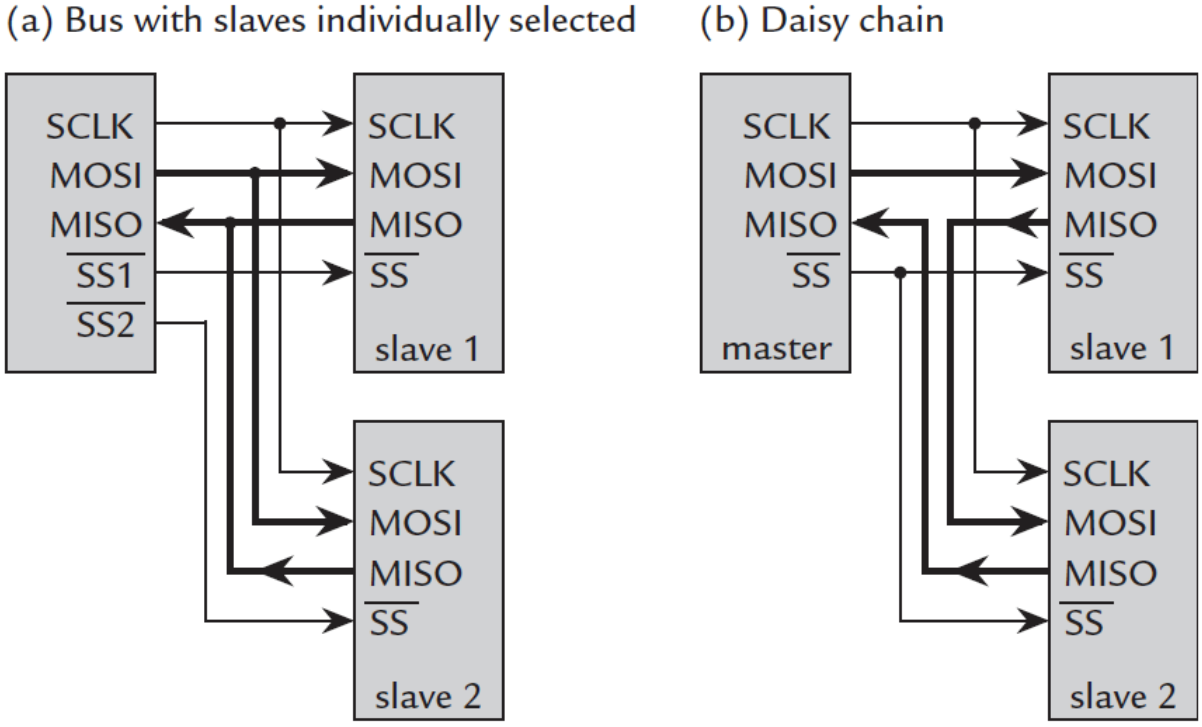
I2C bus supports three different data rates. They are Standard mode (Data rate up to 100kbits.see (100 kbps)), Fast mode (Data rate up to 400kbits:sec 1400 kbps)) and High speed mode (Data rate up to 3.4Mbitsithe (3.4 Mbps)). The first generation I2C devices was designed to support data rates only upto 100kbps. The new generation I2C devices are designed to operate at data rates up to 3.4Mbits/sec.

Serial Peripheral Interface (SPI) Bus: The Serial Peripheral Internal Bus (SPI) is a synchronous bi-directional full duplex four-wire serial interface bus. The concept of SP1 was introduced by Motorola. SPI is a single master multi-s1ave system. It is possible to have a system where more than one SPI device can be master, provided the condition only one master device is active at any given point of time is satisfied. SPI requires four signal lines fur communication. They are:

Master Out Slave In (MOSI): Signal line carrying the data from master to slave device. It is also known as Slave Input/Slave Data In (SI/DI)
Master In Slave Out (MISO): Signal line carrying the data front slave to master device. It is also known as Stave Output (SO/SDO).
Serial Clock (SCLK): Signal line For carrying the clock signals
Slave select: signal line for slave device select. It is an active low signal

The bus interface diagram shown in Fig illustrates the connection of master and slave devices on the SPI The master device is responsible for generating the clock signal. It selects the required slave device by inserting the corresponding slave device's slave select signal 'LOW’. The data

out line (MISO) of all the slave devices when not selected floats at high impedance state. The serial data transmission through SPI bus is fully configurable. SPI devices contain a certain set of registers for holding these configurations. SPI works on the principle of 'Shift Register'. The master and slave devices contain a special shift register for the data to transmit or receive. The size of the shift register is device dependent Normally it is a multiple of 8. During transmission from the master to slave, the data in the master's shift register is shifted out to MOSI pin and it enters the shift register of the slave device through the MOSI pin of the slave device.



At the same time the shifted register out data bit from the slave device's shift register enters the shift register of the master device through MISO pin.When comprised to 12C, SPI bus is most suitable for applications requiring transfer data in 'streams'. The only limitation is SPI doesn't support an acknowledgement mechanism.

Universal Asychrounous Receiver Transmitter(UART):

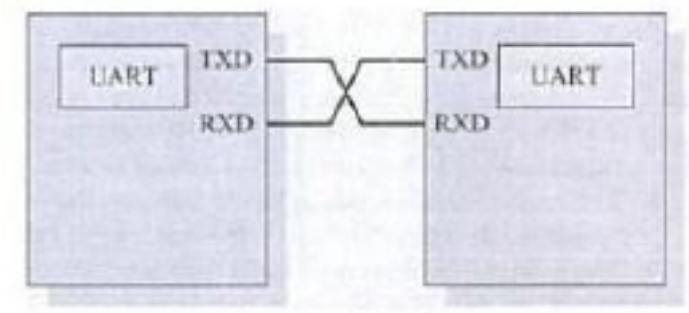


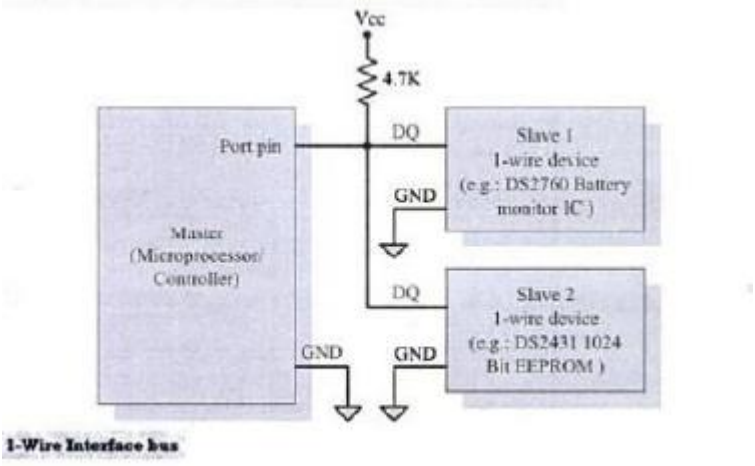
Fig. 2.28 **UART Interfacing**

UART based data transmission is an asynchronous form of serial data transmission. UART based serial data transmission doesn't require a clock signal to Synchronise the transmitting end and receiving end for transmission. Instead it relies upon the pre-defined agreement between the transmitting device and receiving device. The serial communication settings (Baudrate. number of bits per byte, parity, number of start bits and stop bit and flow control) for both transmitter and receiver should be set *as* identical. The start and stop of communication is indicated through inserting special bits in the data streamn. While sending a byte of data, a start bit is added fist and a stop bit is added at the end of the bit stream. The least significant bit of the data byte follows the 'start' bit.

For proper communication the 'Transmit line' of the sending device should be connected to the receive line of the receiving device. In addition to the serial data transmission function UART provides hardware handshaking signal support for controlling the serial data flow. UART chips are available from different semiconductor manufacturers. National Semiconductor's 8250 UART chip is considered as the standard setting UART. It was used in the original IBM PC. Now a days most of the microprocessors/controllers are available with integrated UART functionality and they provide built-in instruction support for serial data transmission and reception.

I-Wire Interface:

I-wire interface is an asynchronous half-duplex communication protocol developed by Maxim Dallas Semiconductor.It is also known as Dallas I-Wire protocol. It makes use of only a single signal line (wire) called DQ for communication and follows the master-slave communication model. One of the key feature of I-wire bus is that it allows power to be sent along the signal wire as well. The 12C slave devices incorporate internal capacitor (typically of the order of 800pF) to power the device from the signal line. The I-wire interface supports a single master and one or more slave devices on the bus. The bus interface diagram shown in Fig illustrates the connection of master and slave devices on the 1-wire bus.



1-Wire Interface bus

Every I -wire device contains a globally unique 64 bit identification number stored within it. This unique identification number can be used for addressing individual devices present on the bus in case there are multiple slave devices connected to the I-wire bus.

The identifier has three parts :an 8 bit family code. ,48 bit serial number and an 8 bit CRC computed from the first 56 bits. The sequence of operation for communicating with a 1-wire slave device is listed below.

I. The master device sends a 'Reset pulse on the 1-wire bus.

2. The slave device(s) present on the bus respond with a 'Presence' pulse.

3. The master device sends a ROM command (Net Address Command followed by the 64bit address of the device). This addresses the slave device(s) to which it wants to initiate a communication.

4. The master device sends a read/write function command to read/write the internal memory or register of the slave device

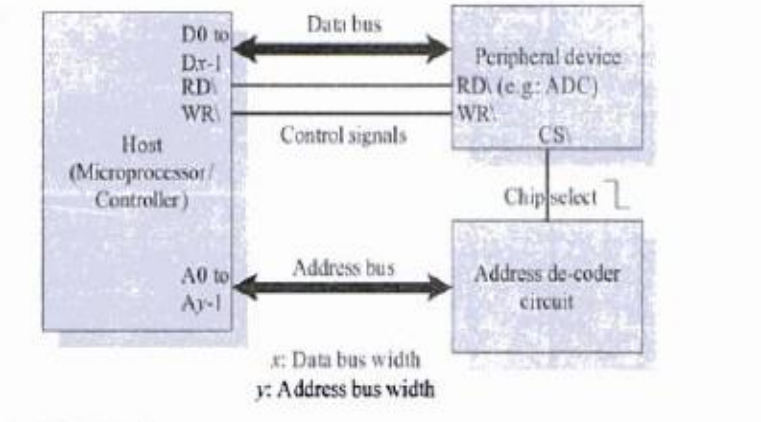
5. The master initiates a Read data/Write data from the device or to the device.

All communication over the I wire bus is master initiated. The communication over the I -wire bus is divided into timeslots of 60 microseconds.

Parallel interface:

The on-board parallel interface is normally used for communicating with peripheral devices which are memory mapped to the host of the system. The host processor/controller of the embedded system contains a parallel bus and the device which supports parallel bus can directly connect to this bus system. The communication through the parallel bus is controlled by the control signal interface between the device and the host. The 'Control Signals' for communication includes Read'/Write' signal and device select signal. The device normally contains a device select line and the device becomes active only when this line is asserted by the host processor. The direction of data transfer (host to Device or Device to host) can be controlled through the control signal lines for 'Read' and 'Write'. Only the host processor has control over the 'Read' and 'Write' control signals.

The device is normally memory mapped to the host processor and a range of address is assigned to it. An address decoder circuit is used for generating the chip select signal for the device. When the address selected by the processor is within the range assigned for the device,. the decoder circuit activates the chip select line and thereby the device becomes active. The processor then can read or write from or to the device by asserting the corresponding control line (Read and Write respectively).



Parallel Interface Bus

External Communication Interface:

The External Communication Interface refers to the different communication channel buses used by the embedded system to communicate with the external world. The following section gives an overview of the various interfaces for external communication.

RS-232 C & RS-485 RS-232 C (Recommended Standard number 232. revision C from the Electronic industry Association) is a legacy, full duplex, wired. asynchronous serial communication interface. The RS-232 interface is developed by the Electronics Industries Association (EIA) during the early 1960s. RS-232 extends the UART communication signals for external data communication.

UART uses the standard TTL/CMOS logic (Logic 'High' corresponds to bit value 1 and Logic 'Low' corresponds to bit value 0) for bit transmission whereas RS-232 follows the EIA standard for bit transmission, As per the EIA standard. a logic *0s is represented with voltage between +3 and +25V and a logic 1 is represented with voltage between -3 and -25V. The RS-232 interface defines various handshaking and control signals for communication apart from the 'Transmit' and 'Receive' signal lines for data communication, RS-232 sup-ports two different types of Conn ions, namely; DB-9: 9-Pin connector end DB-25; 25-Pin connector Figure 2.31 illustrates the connector details for DB-9 and DB-25.

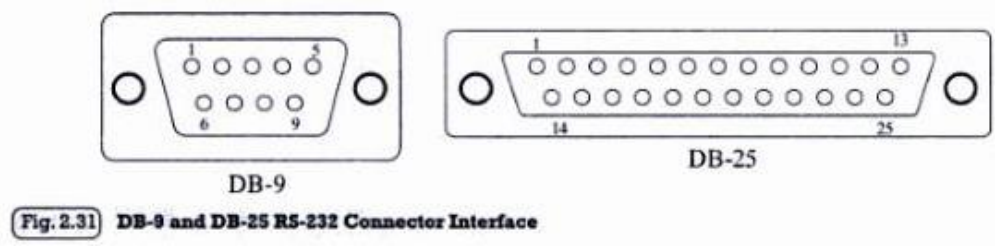


Fig. 2.31 **DB-9 and DB-25 RS-232 Connector Interface**

The 25 pin DB connector contains two sets of signal lines for transmit, receive and control lines. Nowadays DB-25 connector is obsolete and most of the desktop systems are available with DB-9 connectors only.

As per the EIA standard RS-232 C supports baudrates up to 20Kbps (Upper limit 19_2 Kbps) The com-monly used baudrates by devices arc 300bps, 1200bps, 2400bps, 9600bps, 11.52Kbps and 19.2Kbps. 9600 is the popular baudrate setting used for PC communication. The maximum operating distance supported by RS-232 is 50 feet at the highest supported baudrate.

Embedded devices contain a UART for serial communication and they generate signal levels forming to TTL/CMOS logic. A level translator IC like MAX 232 from Maxim Dallas semiconductor is used for convening the signal lines from the UART to RS-232 signal lines for communication. On the receiving side the received data is converted back to digital logic level by a converter IC. Converter chips contain converters for both transmitter and receiver.

Though RS-232 was the most popular communication interface during the olden days, the advent of other communication techniques like Bluelooth, USB, Firewire, etc are pushing down RS-232 from the scenes. Still RS-232 is popular in certain legacy industrial applications. RS-232 supports only point-to-point communication and not suitable for multi-drop communication. It

uses single ended data transfer technique for signal transmission and thereby more susceptible to noise and it greatly reduces the operating distance.

RS-422 is another serial interface standard from EIA for differential data communication. It supports data rates up to 100Kbps and distance up to 400 ft. The same RS-232 connector is used at the device end and an RS-232 to RS-422 convener is plugged in the transmission line. At the receiver end the conversion from RS-422 to RS-232 is performed. RS-422 supports multi-drop communication with one transmitter device and receiver devices up to 10.

RS-485 is the enhanced version of RS-422 and it supports multi-drop communication with up to 32 transmitting devices (drivers) and 32 receiving devices on the bus. The communication between devices in the bus uses the 'addressing' mechanism to identify slave devices.

Universal Serial Bus (USB):

Universal Serial Bus (USB) is a wired high-speed serial bus for data communication. The first version of USB (USB1.0) was released in 1995 and was created by the USB core group members consisting of Intel, Microsoft, IBM, Compaq, Digital and Northern Tele-com. The USB communication system follows a star topology with a USB host at the center and one or more USB peripheral devices/USB hosts connected to it.

A USB host can support connection up to 127, including slave peripheral devices and other USB hosts. Figure 2.32 illustrates the star topology for USB device connection. USB transmits data in packet format. Each data packet has a standard formal_ The USB communication is a host initiated one. The USI3 host contains a host controller which is responsible for controlling the data communication, including establishing connectivity with USB slave

The physical connection between a USB peripheral de-vice and master device is established with a USB cable. The USB cable supports communication distance of up to 5 metres. The USB standard uses two different types of connector at the ends of the USB cable for connecting the USB peripheral device and host device. 'Type A' connector is used for upstream connection (connection with host) and Type B connector is used for downstream connection (connection with slave device). The USB connector present in desktop PCs or laptops arc examples for 'Type A' USB connector. Both 'Type A and Type B connectors contain 4 pins for communication. The Pin details for the connectors arc listed in the table given below.

Pin no:	Pin name	Description
1	V_{BUS}	Carries power (5V)
2	D–	Differential data carrier line
3	D+	Differential data carrier line
4	GND	Ground signal line

USB uses differential signals for data transmission. It improves the noise immunity. USB interface has the ability to supply power to the connecting devices. Two connection lines (Ground and Power) of the USB interface arc dedicated for carrying power. It can supply power up to 500 mA at 5 V. It is sufficient to operate low power devices. Mini and Micro USB connectors arc available for small form factor devices like portable media players.

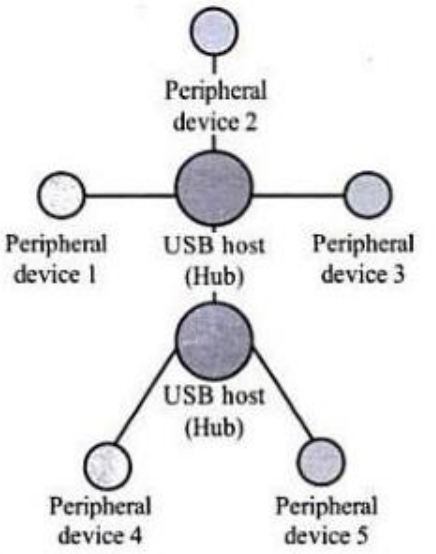


Fig. 2.32 **USB Device Connection topology**

USB supports four different types of data transfers, namely; Control, Bulk, Isochronous and Interrupt. **Control transfer** is used by USB system software to query, configure and issue commands to the USB device**. Bulk transfer** is used for sending a block of data to a device. Bulk transfer supports error checking and correction. Transferring data to a printer is an example for bulk transfer. Isochronous data transfer is used for real-time data communication. In **Isochronous transfer**, data is transmitted as streams in real-time. Isochronous transfer doesn't support error checking and re-transmission of data in case of any transmission loss. All streaming devices like audio devices and medical equipment for data collection make use of the isochronous transfer. **Interrupt transfer** is used for transferring small amount of data Interrupt transfer mechanism makes use of polling technique to sec whether the USB device has any data to send. The frequency of polling is determined by the USB device and it varies from Ito 255 milliseconds. Devices like Mouse and Keyboard, which transmits fewer amounts of data, uses Interrupt transfer.

USB.ORG (www.usb.org) is the standards body for defining and controlling the standards for USB communication. Presently USB supports four different data rates namely; Low Speed (1.5Mbps), Full Speed (12Mbps). High Speed (480Mbps) ,Super Speed (4.8Gbps). Super Speed + (10 Gbps). There is a move happening towards wireless USB for data transmission using Ultra Wide Band (UWB) technology. Some laptops are already available in the market with wirless USB support.

IEEE 1394 (Firewire)

IEEE 1394 is a wired, isochronous high-speed serial communication bus. It is also known as High Performance Serial Bus (HPSB), The research on 1394 was started by Apple Inc. in 1985 and the standard for this was coined by IEEE. The implementation of it is avail-able from various players with different names. Apple Inc's (www.apple.com) implementation of 1394 protocol is popularly known as Firewire. LYNX is the 1394 implementation from Sony Corporation (www.sony.com) and Lynx is the implementation from Texas Instruments (www.ti.com).

1394 supports peer-to-peer connection and point-to-multipoint communication allowing 63 devices to be connected on the bus in a tree topology. 1394 is a wired serial interface and it can support a cable length of up to 15 feet for interconnection. The 1394 standard has evolved a lot from the first version IEEE 1394-1995 released in 1995 to the recent version IEEE 1394-2008 released in June 2008. The 1394 standard supports a data rate of 400 to 3200Mbps/second.

The IEEE 1394 uses differential data transfer (The information is sent using differential signals through a pair of twisted cables. It increases the noise immunity) and the interlace cable supports 3 types of connectors. namely: 4-pin connector, 6-pin connector (alpha connector) and 9 pin connector (beta connector). The 6 and 9 pin connectors carry power also to support external devices (In case an embedded device is connected to a PC through an IEEE 1394 cable with 6 or 9 pin connec-tor interface, it can operate from the power available through the connector.) It can supply unregulated power in the range of 24 to 30V. (The Apple implementation is for battery operated devices and it can supply a voltage in the range 9 to 12V.)

1394 is a popular communication interface for connecting embedded devices like digital camera, camcorder, scanner to desktop computers for data transfer and storage. Unlike USB interface (except USB OTG), IEEE 1394 doesn't require a host for communicating between devices. For example, you can directly connect a scanner with a printer for printing. The data rate supported by1394 is far higher than the one supported by USB2.0 interface. The 1394 hardware implementation is much costlier than USB implementation.

Infrared (IrDA)

Infrared (IrDA) is a serial. half duplex. line of sight based wireless technology for data communication between devices. It is in use from the olden days of communication and you may be very familiar with it. The remote control of your TV, VCD player, etc. works on Infrared data communication principle. Infrared communication technique uses infrared waves of the electromagnetic spectrum for transmitting the data.

IrDA supports point-point and point-to-multipoint communication. provided all devices involved in the communication arc within the line of sight. The typical communication range for IrDA lies in the range 10 cm to 1m. The range can be increased by increasing the transmitting power of the IR device. IR supports data rates ranging from 9600bits/second to 16Mbps.

Depending on the speed of data transmission IR is classified into Serial IR (SIR), Medium IR (MIR). Fast IR (FIR). Very Fast IR (VFIR) and Ultra Fast IR (UFIR). SIR supports transmission rates ranging from 9600bps to 115.2kbps. MIR supports data rates of 0.576Mbps and 1.152Mbps. FIR supports data rates up to 4Mbps. VFIR is designed to support high data rates up to 16Mbps. The UFIR specs arc under development and it is targeting a data rate up to 100Mbps.

IrDA communication involves a transmitter unit for transmitting the data over IR and a receiver for receiving the data. Infrared Light Emitting Diode (LED) is the IR source for transmitter and at the receiving end a photodiode acts as the receiver. Both transmitter and receiver unit will be present in each device supporting IrDA communication for bidirectional data transfer. Such IR units are known as 'Transceiver'. Certain devices like a TV remote control always require

unidirectional communication and so they contain either the transmitter or receiver unit (The remote control unit contains the transmitter unit and TV contains the receiver unit).

'Infra-red Data Association' is the regulatory body responsible for de-fining and licensing the specifications for IR data communication. IrDA communication has two essential parts: a physical link part and a protocol part. The physical link is responsible for the physical transmission of data between devices supporting IR communication and protocol part is responsible for defining the rules of communication. The physical link works on the wireless principle making use of Infrared for communication. The IrDA specifications include the standard for both physical link and protocol layer. The IrDA control protocol contains implementations for Physical Layer (PHY). Media Access Control (MAC) and Logical Link Control (LLC). The Physical Layer defines the physical characteristics of communication like range, data rates, power. etc.

IrDA is a popular interface for file exchange and data transfer in low cost devices. IrDA was the prominent communication channel in mobile phones before Bluetooth's existence. Even now most of the mobile phone devices support IrDA.

Bluetooth

Bluetooth is a low cost, low power. short range wireless technology for data and voice communication. Bluetooth was first proposed by 'Ericsson' in 1994. Bluetooth operates at 2.4GHz of the Radio Frequency spectrum and uses the Frequency Hopping Spread Spectrum (FHSS) technique for communication. Literally it supports a data rate of up to 1Mbps and a range of approximately 30 feet for data communication. Like IrDA. Bluetooth communication also has two essential parts: a physical link pan and a protocol part. The physical link is responsible for the physical trans-mission of data between devices supporting

Bluetooth communication and protocol part is responsible for defining the rules of communication. The physical link works on the wireless principle making use of RF waves for communication. Bluetooth enabled devices essentially contain a Bluetooth wireless radio for the transmission and reception of data. The rules governing the Bluetooth communication is implemented in the 'Bluetooth protocol stack'. The Bluetooth communication IC holds the stack. Each Bluetooth device will have a 48 bit unique identification number.

Bluetooth communication follows packet based data transfer. Bluetooth supports point-to-point (device to device) and point-to-multipoint (device to multiple device broadcasting) wireless communication. The point-to-point communication follows the master-slave relationship. A Bluetooth device can function as either master or slave. When a network is formed with one Bluetooth device as master and more than one device as slaves, it is called a Piconet. A Piconet supports a maximum of seven slave devices.

Bluetooth is the favorite choice for short range data communication in handheld embedded devices. Bluetooth technology is very popular among cell phone users as they arc the easiest communication channel for transferring ringtones. music files, pictures, media files, etc. between neighboring Blue-tooth enabled phones. The Bluetooth standard specifies the minimum

requirements that a Bluetooth device must support for a specific usage scenario. The Generic Access Profile (GAP) defines the requirements for detecting a Bluetooth device and establishing a connection with it.

All other specific usage profiles are based on GAP. Serial Port Profile (SPP) for serial data communication. File Transfer Profile (FTP) for file transfer between devices, Human Interface Device (HID) for supporting human interface devices like keyboard and mouse arc examples for Bluctooth profiles. The specifications for Bluetooth communication is defined and licensed by the standards body 'Blue-tooth Special Interest Group (SIG)'.

Wi-Fi

Wi-Fi or Wireless Fidelity is the popular wireless communication technique for networked communication of devices. Wi-Fi follows the IEEE 802.11 standard. Wi-Fi is intended for network communication and it supports Internet Protocol (IP) based communication. It is essential to have device identities in a multipoint communication to address specific devices for data communication. In an IP based communication each device is identified by an IP address, which is unique to each device on the network.

Wi-Fi based communications require an intermediate agent called Wi-Fi router/Wireless Access point to manage the communications. The Wi-Fi router is responsible for restricting the access to a network, assigning IP address to devices on the network. routing data packets to the intended devices on the network. Wi-Fi enabled devices contain a wireless adaptor for transmitting and receiving data in the form of radio signals through an antenna. The hardware pan of it is known as Wi-Fi Radio.

Wi-Fi operates at 2.4Ghz or 5Ghz of radio spectrum and they co-exist with other ISM band devices like Bluetooth. Figure 2.33 illustrates the typical interfacing of devices in a Wi-Fi network.

Fig. 2.33 Wi-Fi network

For communicating with devices over a Wi-Fi network, the device when its Wi-Fi radio is turned ON, searches the available Wi-Fi network in its vicinity and lists out the Service Set Identifier (SSID) of the available networks. If the network is security enabled, a password may be required to connect to a particular SSID. Wi-Fi employs different security mechanisms like Wired

Equivalency Privacy (WEP) Wireless Protected Access (WPA), etc. for securing the data communication. Wi-Fi supports data rates ranging from 1Mbps to 150Mbps (Growing towards higher rates as technology progresses) depending on the standards (802.11a/b/g/n) and access/modulation method. Depending on the type of antenna and usage location (indoor/outdoor). Wi-Fi offers a range of 100 to 300 feet.

ZigBee

ZigBee is a low power. low cost. wireless network communication protocol based on the IEEE 802.15.4-2006 standard. ZigBee is targeted for low power. low data rate and secure applications for Wireless Personal Area Networking (WPAN). The ZigBee specifications support a robust mesh network containing multiple nodes. This networking strategy makes the network reliable by permitting messages to towel through a number of different paths to get from one node to another.

ZigBee operates worldwide at the unlicensed bands of Radio spectrum. mainly at 2.400 to 2.484 GHz. 902 to 928 MHz and 86&0 to 868.6 MHz. ZigBee Supports an operating distance of up to 100 metres and a data rate of 20 to 250Kbps. In the ZigBee terminology, each ZigBce device falls under any one of the following ZigBee device category. ZIgflec Coordinator (ZC)/Network Coordinator. The ZigBcc coordinator acts as the root of the Zig-Bcc network. The ZC is responsible for initiating the ZigBce network and it has the capability to store information about the network. ZIgReo Router (ZR)/Full function nosier (FFD): Responsible for passing information from device to another device or to another ZR. ZigBee End Device (ZED)/Reduced Function Device (RFD) End device containing ZigBee functionality for data communication. It can talk only with a ZR or ZC and doesn't have the capability to act as a mediator for transferring data from one device to another.

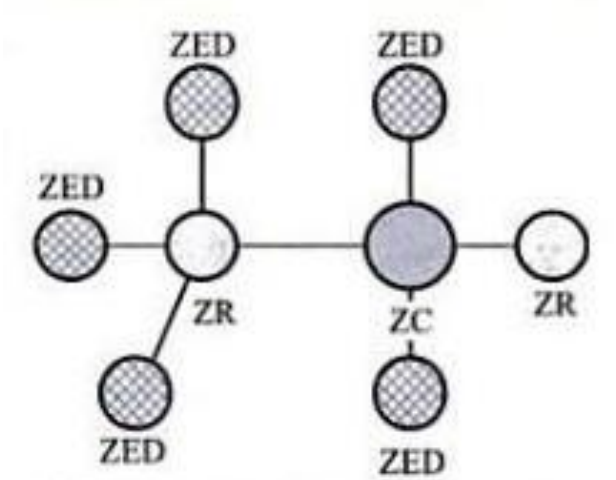


Fig. 2.34 A ZigBee network model

The diagram shown in Fig. 2.34 gives an overview of ZC, ZED ZED ZED and ZR in a ZigBee network ZigBee is primarily targeting application areas like home & industrial automation. energy management, home control security. Medical, patient tracking. logistics & asset tracking and sensor networks & active RFID. Automatic Meter Reading (AMR), smoke detectors, wireless telemetry, HVAC control, heating control, lighting controls, environmental controls. etc.

are examples for applications which can make use of the Zig-Bee technology. The specifications for ZigBee is developed and managed by the ZigBee alliance (www.zigbee.org), a non-profit consortium of leading semiconductor manufacturers, technology providers, OEMs and end-users worldwide.

General Packet Radio Service (GPRS)

General Packet Radio Service (GPRS) is a communication technique for transferring data over a mobile communication network like GSM. Data is sent as packets in GPRS communication. The transmitting device splits the data into several related packets. At the receiving end the data is re-constructed by combining the received data packets.

GPRS supports a theoretical maximum transfer rate of 171.2kbps. In GPRS communication, the radio channel is concurrently shared between several users instead of dedicating a radio channel to a cell phone user. The GPRS communication divides the channel into 8 timeslots and transmits data over the available channel. GPRS supports Internet Protocol (IP). Point to Point Protocol (PPP) and X.25 protocols for communication.

GPRS is mainly used by mobile enabled embedded devices for data communication. The device should support the necessary GPRS hardware like GPRS modem and GPRS radio. To accomplish GPRS based communication, the carrier network also should have support for GPRS communication.

GPRS is an old technology and it is being replaced by new generation data communication techniques like EDGE, High Speed Downlink Packet Access (HSDPA), etc. which offers higher bandwidths for communication.

EMBEDDED FIRMWARE

Embedded firmware refers to the control algorithm (Program instructions) and or the configuration settings that an embedded system developer dumps into the code (Program) memory of the embedded system. It is an un-avoidable part of an embedded system. There are various methods available for developing the embedded firmware. They are listed below.

1. Write the program in high level languages like Embedded C/C++ using an Integrated Development Environment (The IDE will contain an editor, compiler, linker. debugger, simulator, etc. IDEs are different for different family of processors/controllers. For example, Keil micro vision3 IDE is used for all family members of 8051 microcontroller, since it contains the generic 8051 compiler C51).
2. Write the program in Assembly language using the instructions supported by your application's target processor/controller.

The instruction set for each family of processor/controller is different and the program written in either of the methods given above should be converted into a processor understandable machine code before loading it into the program memory.

The process of converting the program written in either a high-level language or processor/controller specific Assembly code to machine readable binary code is called 'HEX File Creation'. The methods used for HEX File Creation' is different depending on the programming techniques used. If the program is written in Embedded C/C++ using an IDE, the cross compiler included in the IDE converts it into corresponding processor/controller understandable 'HEX File'.

If you arc following the Assembly language-based programming technique (method 2), you can use the utilities supplied by the processor/controller vendors to convert the source code into 'HEX File'. Also third party tools arc available. which may be of free of cost, for this conversion. For a beginner in the embedded software field, it is strongly recommended to use the high level language based development technique.

The reasons for this being: writing codes in a high level language is easy, the code written in high level language is highly portable which means you can use the same code to run on different processor/controller with little or less modification. The only thing you need to do is re-compile the program with the required processor's IDE, after replacing the include files for that particular processor.

Also the programs written in high level languages are not developer dependent. Any skilled programmer can trace out the functionalities of the program by just having a look at the program. It will be much easier if the source code contains necessary comments and documentation lines. It is very easy to debug and the overall system development time will be reduced to a greater extent.

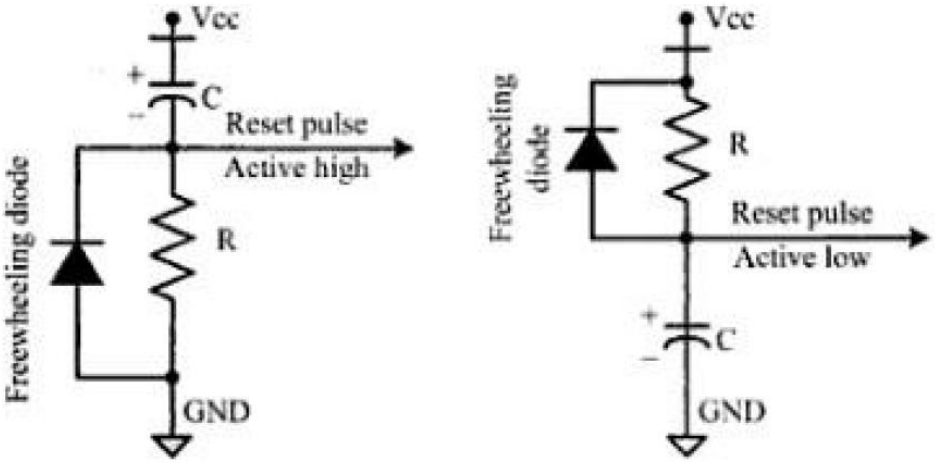
The embedded software development process in assembly language is tedious and time consuming. The developer needs to know about all the instruction sets of the processor/controller or at least s/he should catty an instruction set reference manual with her/him. A programmer using assembly language technique writes the program according to his/her view and taste. Often he/she may be writing a method or functionality which can be achieved through a single instruction as an experienced person's point of view, by two or three instructions in his/her own style. So the program will be highly dependent on the developer. It is very difficult for a second person to understand the code written in Assembly even if it is well documented.

Reset Circuit

The reset circuit is essential to ensure that the device is not operating at a voltage level where the device is not guaranteed to operate, during system power ON. The reset signal brings the internal registers and the different hardware systems of the processor/controller to a known state and starts the firmware execution from the reset vector (Normally from vector address Ox0000 for conventional processors/controllers.

The reset vector can be relocated to an address for processors/controllers supporting bootloader). The reset signal can be either active high (The processor undergoes reset when the reset pin of the processor is at logic high) or active low (The processor undergoes reset when the reset pin of the processor is at logic low). Since the processor operation is synchronized to a clock signal. the

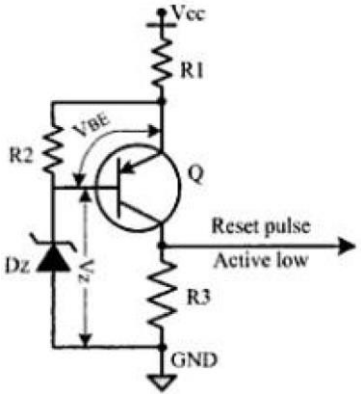
reset pulse should be wide enough to give time for the clock oscillator to stabilize before the internal reset state starts.



The reset signal to the processor can be applied at power ON through an external passive reset circuit comprising a Capacitor and Resistor or through a standard Reset IC like MAX810 from Maxim Dallas (www max im-ic.com). Select the reset IC based on the type of reset signal and logic level supported by the processor/controller in use. reset circuitry and they don't require external reset circuitry. Figure 2.35 illustrates a resistor capacitor based passive reset circuit for active high and low configurations. The reset pulse width can be adjusted by changing the resistance value R and capacitance value C.

Brown-out Protection Circuit

Brown-out protection circuit prevents the processor. controller from unexpected program execution behavior when the supply voltage to the processor/controller falls below a specified voltage. It is essential for battery powered devices since there arc greater chances for the battery voltage to drop below the requir.x1 threshold. The processor behavior may not be predictable if the supply voltage falls below the recommended operating voltage. It may lead to situations like data corruption.



A brown-out protection circuit holds the processor/controller in reset state. when the operating voltage falls below the threshold, until it uses above the threshold voltage. Certain processors/controllers support built in brown-out protection circuit which monitors the supply voltage internally. If the processor/controller doesn't integrate a built-in brown-out protection

circuit, the same can be implemented using external passive circuits or supervisor 1Cs. Figure 2.36 illustrates a brown-out circuit implementation using Zener diode and transistor for processor/controller with active low Reset logic.

The Zener diode Dz and transistor Q forms the heart of this circuit. The transistor conducts always when the supply voltage V., is greater than that of the sum of VBE and V, (Zener voltage). The transistor stops conducting when the supply voltage falls below the sum of VBE and Vz. Select the Zener diode with required voltage for setting the low threshold value for Vcc. The values of RI. R2. and R3 can be selected based on the electrical characteristics (Absolute maximum current and voltage ratings) of the transistor in use. Microprocessor Supervisor ICs like DS1232 from Maxim Dallas (www.maxim-ic.com) also provides Brown-out protection.

Oscillator Unit:

A microprocessor/microcontroller is a digital device made up of digital combinational and sequential circuits. The instruction execution of a microprocessor/controller occurs in sync with a clock signal. It is analogous to the heartbeat of a living being which synchronizes the execution of life. For a living being. the heart is responsible for the generation of the beat whereas the oscillator unit of the embedded system is responsible for generating the precise clock for the processor.

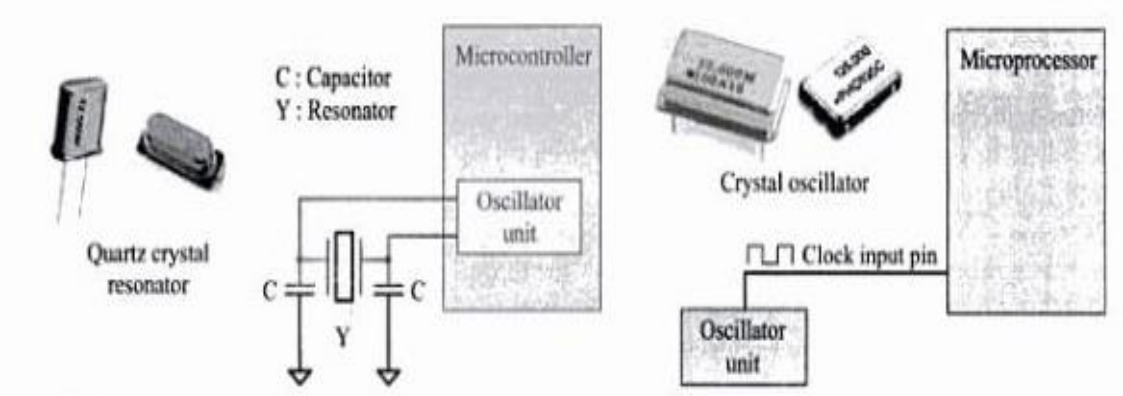


Fig. 2.37 Oscillator circuitry using quartz crystal and quartz crystal oscillator

Figure 2.37 illustrates the usage of quartz crystal/ceramic resonator and external oscillator chip for clock generation.

Certain processors/controllers integrate a built-in oscillator unit and simply require an external ceramic resonator/quartz crystal for producing the necessary clock signals. Quartz crystals and ceramic resonators arc equivalent in operation, however they possess physical difference. A quartz crystal is normally mounted in a hermetically sealed metal case with two leads protruding out of the case.

Certain devices may not contain a built-in oscillator unit and require the clock pulses to be generated and supplied externally. Quartz crystal Oscillators arc avail-able in the form chips and they can be used for generating the clock pulses in such a cases. The speed of operation of a processor is primarily dependent on the clock frequency.

However, we cannot increase the clock frequency blindly for increasing the speed of execution. The logical circuits lying inside the processor always have an upper threshold value for the maximum clock at which the system can run. beyond which the system becomes unstable and non-functional. The total system power consumption is directly proportional to the clock frequency. The power consumption increases with increase in clock frequency. The accuracy of program execution depends on the accuracy of the clock signal. The accuracy of the crystal oscillator or ceramic resonator is normally expressed in terms of +/-ppm (Pans per million).

Real-Time Clock (RTC)

Real-Time Clock (RTC) is a system component responsible for keeping track of time. RTC holds information like current time (In hours, minutes and seconds) in 12 hour/24 hour format, date. month. year. day of the week, etc. and supplies timing reference to the system. RTC is intended to function even in the absence of power. RTCs are available in the form of Integrated Circuits from different semiconductor manufacturers like Maxim/Dallas. ST Microelectronics etc.

The RTC chip contains a microchip for holding the time and date related information and backup battery cell for functioning in the absence of power, in a single IC package. The RTC chip is interfaced to the processor or controller of the embedded system. For Operating System based embedded devices, a timing reference is essential for synchronizing the operations of the OS kernel. The RTC can interrupt the OS kernel by asserting the interrupt line of the processor/controller to which the RTC interrupt line is connected.

The OS kernel identifies the interrupt in terms of the Interrupt Request (IRQ) number generated by an interrupt controller. One IRQ can be assigned to the RTC interrupt and the kernel can perform necessary operations like system date time updation, managing software timers etc when an RTC timer tick interrupt occurs. The RTC can be configured to interrupt the processor at predefined intervals or to interrupt the processor when the RTC register reaches a specified value (used as alarm interrupt).

Watchdog Timer

In desktop Windows systems, if we feel our application is behaving in an abnormal way or if the system hangs up. we have the + Alt s Del' to come out of the situation. What if it happens to our embedded system? Do we really have a 'Ctrl + Alt + Del' tr take control of the situation? Of course not, but we have a watchdog to monitor the firmware execution and reset the system processor/microcontroller when the program execution hangs up.

A watchdog timer, or simply a watchdog. is a hardware timer for monitoring the firmware execution. Depending on the internal implementation. the watchdog timer increments or decrements a free running counter with each clock pulse and generates a reset signal to reset the processor if the count reaches zero for a down counting watchdog. or the highest count value for an up-counting watchdog. lithe watchdog counter is in the enabled state. the firmware can write a zero (for up counting watchdog implementation) to it before starting the execution of a piece of code (subroutine or portion of code which is susceptible to execution hang up) and the watchdog

will start counting. lithe firmware execution doesn't complete due to malfunctioning, within the time required by the watchdog to reach the maximum count, the counter will generate a reset pulse and this will reset the processor (if it is connected to the reset line of the processor).

If the firmware execution completes before the expiration of the watchdog timer you can reset the count by writing a 0 (for an up-counting watchdog timer) to the watchdog tinier register. Most of the processors implement watchdog as a built-in component and provides status register to control the watchdog timer (like enabling and disabling watchdog functioning) and watchdog timer register for writing the count value. lithe processor/controller doesn't contain a built in watchdog timer, the same can be implemented using an external watchdog timer IC circuit.

The external watchdog timer uses hardware logic for enabling/disabling. resetting the watch-dog count, etc instead of the firmware based 'writing' to the status and watchdog timer register. The Microprocessor supervisor IC DS1232 integrates a hardware watchdog timer in it. In modem systems running on embedded operating systems, the watchdog can be implemented in such a way that when a watchdog timeout occurs. an interrupt is generated instead of resetting the processor. The interrupt handler for this handles the situation in an appropriate fashion. Figure 2.38 illustrates the implementation

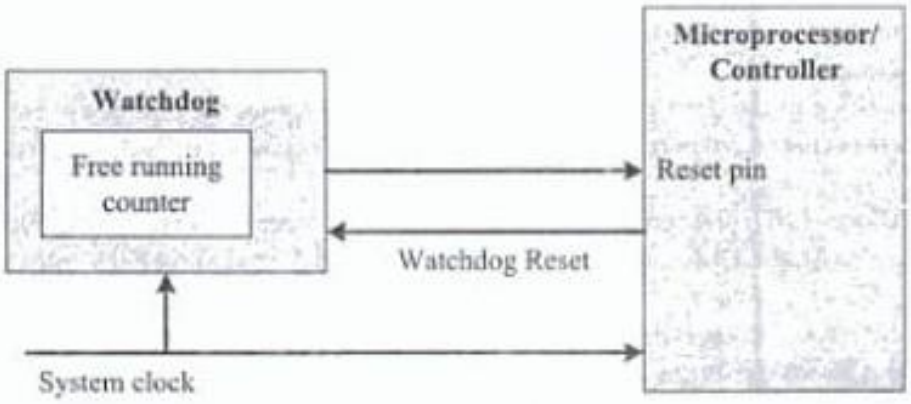


Fig. 2.38 Watchdog timer for firmware execution supervision