I2C PROTOCAL -(INTER INTEGRATED CIRCUIT)

It is a bus interface connection protocal incorporated into devices for seriel communication. It was originally designed by phillips smiconductor in 1982.Recentely it was widely used protocal for short-distance communication.

SERIEL COMMUNICATION :

I2C is a seriel commuication protocal. So data transferred bit by bit along a single wire(SDA line).

SYNCHRONOUS :

Like spi, I2C is sychronous , So the output of bits is synchronized to the sampling of bits by a clock signal shared between the master and slave.The clock signal is always controlled by the master.

HALF DUPLEX :

I2C protocal is a half duplex protocal ,we can only transmit or receive a data at a time.

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.

According to I2C protocols, the data line can not change when the clock line is high, it can change only when the clock line is low. The 2 lines are open drain, hence a pull-up resistor is required so that the lines are high since the devices on the I2C bus are active low. The data is transmitted in the form of packets which comprises 9 bits. The sequence of these bits are

Start Condition – 1 bit

Slave Address – 8 bit

Acknowledge — 1 bit

Start and Stop Conditions :

START and STOP can be generated by keeping the SCL line high and changing the level of SDA. To generate START condition the SDA is changed from high to low while keeping the SCL high. To generate STOP condition SDA goes from low to high while keeping the SCL high, as shown in the figure below.

Repeated Start Condition :

Between each start and stop condition pair, the bus is considered as busy and no master can take control of the bus. If the master tries to initiate a new transfer and does not want to release the bus before starting the new transfer, it issues a new START condition. It is called a REPEATED START condition.

Read/Write Bit :

A high Read/Write bit indicates that the master is sending the data to the slave, whereas a low Read/Write bit indicates that the master is receiving data from the slave.

ACK/NACK Bit :

After every data frame, follows an ACK/NACK bit. If the data frame is received successfully then ACK bit is sent to the sender by the receiver.

Addressing :

The address frame is the first frame after the start bit. The address of the slave with which the master wants to communicate is sent by the master to every slave connected with it. The slave then compares its own address with this address and sends ACK.

Features of I2C Communication Protocol :

- Half-duplex Communication Protocol —

Bi-directional communication is possible but not simultaneously.

- Synchronous Communication —

The data is transferred in the form of frames or blocks.

Can be configured in a multi-master configuration.

- Clock Stretching –

The clock is stretched when the slave device is not ready to accept more data by holding the SCL line low, hence disabling the master to raise the clock line. Master will not be able to raise the clock line because the wires are AND wired and wait until the slave releases the SCL line to show it is ready to transfer next bit.

- Arbitration –

I2C protocol supports multi-master bus system but more than one bus
can not be used simultaneously. The SDA and SCL are monitored by the masters. If the SDA is found high when it was supposed to be low it will be inferred that another master is active and hence it stops the transfer of data.

- Serial transmission –

I2C uses serial transmission for transmission of data.

Used for low-speed communication.

Advantages :

- Can be configured in multi-master mode.
- Complexity is reduced because it uses only 2 bi-directional lines (unlike SPI Communication).
- Cost-efficient.
- It uses ACK/NACK feature due to which it has improved error handling capabilities.

Limitations :

- Slower speed.
- Half-duplex communication is used in the I2C communication protocol.

Pull-up Resistors :

Pull-up and Pull-down resistors are used to correctly bias the inputs of digital gates to stop them from floating about randomly when there is no input conditionThe most common method of ensuring that the inputs of digital logic gates and circuits can not self-bias and float about is to either connect the unused pins directly to ground (0V) for a constant low '0' input, (OR and NOR gates) or directly to Vcc (+5V) for a constant high '1' input (AND and NAND gates). Ok, lets look again at our two switched inputs from above.

switched inputs

This time, to stop the two inputs, A and B, from floating about when the corresponding switches, a and b are open (OFF), the two inputs are connected to +5V supply.You may think that this would work fine as when switch a is open (OFF), the input is connected to Vcc (+5V) and when the switch is closed (ON), the input is connected to ground as before, then inputs A or B always have a default state regardless of the position of the switch.However, this is a bad condition because when either of the switches are closed (ON), there will be a direct short circuit between the +5V supply and ground, resulting in excessive current flow either blowing a fuse or damaging the circuit which is not good news. One way to overcome this issue is to use a pull-up resistor connected between the input pin and the +5V supply rail as shown.

Pull-up Resistor Application :

By using these two pull-up resistors, one for each input, when switch 'A' or 'B' is open (OFF), the input is effectively connected to the +5V supply rail via the pull-up resistor. The result is that as there is very little input current into the input of the logic gate, very little voltage is dropped across the pull-up resistor so nearly all the +5V supply voltage is applied to the input pin creating a HIGH, logic '1' condition.When switches 'A', or 'B' are closed, (ON) the input is shorted to ground (LOW) creating a logic '0' condition as before at the input. However, this time we are not shorting out the supply rail as the pull-up resistor only passes a small current (as determined by Ohms law) through the closed switch to ground.By using a pull-up resistor in this way, the input always has a default logic state, either '1' or '0', high or low, depending on the position of the switch, thus achieving the proper output function of the gate at "Q" and therefore preventing the input from floating about or self-biasing giving us exactly the switching condition we require.While the connection between Vcc and an input (or output) is the preferred method for using a pull-up resistor, the question arises as how do we calculate the value of the resistance require to ensure the correct operation of the input.

To determine the value, the formula is simple Ohms law. As per the ohms law, the formula is

V = I x R

R = V/I

PULL-DOWN RESISTORS :

On the other hand, a pull-down resistor is used to ensure that inputs to logic systems settle at expected logic levels whenever external devices are disconnected or of high impedance. It ensures that the wire is at a defined low logic level even when there are no active connections with other devices. The pull-down resistor holds the logic signal near to zero volts (0V) when no other active device is connected. It pulls the input voltage down to the ground to prevent an undefined state at the input. It should have a larger resistance than the impedance of the logic circuit. Otherwise, it will make the input voltage at the pin on constant logical low value no matter the position of the switch. When the switch is open, the voltage of the gate input is pulled down to the level of the ground. When the switch is closed, the input voltage at the gate goes to Vin. Without the resistor, the voltage levels would virtually float between the two voltages.Just like the pull-up resistor in the first figure, the pull-down resistors in this circuit also ensures that the voltage between VCC and a microcontroller pin is actively controlled when the switch is open. Unlike the pull-up resistor, the pull-down resistor pulls the pin to a low value instead of high value. The pull-down resistor which is connected to the ground or 0V sets the digital logic level pin to default or 0 until the switch is pressed and the logic level pin becomes high. Therefore, the small amount of current flows from the 5-V source to the ground using the closed switch and pull-down resistor preventing the logic level pin from getting shorted with the 5-V source.

APPLICATIONS OF PULL DOWN RESISTORS :

- pull-down resistors are frequently used in interfacing devices like interfacing a switch to microcontroller.
- Most of the microcontrollers have inbuilt programmable pull up/pull down resistors.So Interfacing a switch with a microcontroller directly is possible.
- In general, pull up resistors are often used than pull down resistors, although some microcontroller families have both pull-up and pull-down resistors.

- These resistors are often used in A/D converters to provide a controlled flow of current into a resistive sensor
- Pull-up and pull-down resistors are used in I2C protocol bus, wherein the pull-up resistors are used to allow a single pin to act as an I/P or O/P.
- When it is not connected to a I2C protocol bus, the pin floats in a high impedance state. Pull down resistors are also used for outputs to afford a known O/P

LINUX BOOTING PROCESS :

1. BIOS

BIOS stands for Basic Input/Output System. In simple terms, the BIOS loads and executes the Master Boot Record (MBR) boot loader.When you first turn on your computer, the BIOS first performs some integrity checks of the HDD or SSD.Then, the BIOS searches for, loads, and executes the boot loader program, which can be found in the Master Boot Record (MBR). The MBR is sometimes on a USB stick or CD-ROM such as with a live installation of Linux.Once the boot loader program is detected, it's then loaded into memory and the BIOS gives control of the system to it.

2. MBR

MBR stands for Master Boot Record, and is responsible for loading and executing the GRUB boot loader.The MBR is located in the 1st sector of the bootable disk, which is typically /dev/hda, or /dev/sda, depending on your hardware. The MBR also contains information about GRUB, or LILO in very old systems.

3. GRUB

Sometimes called GNU GRUB, which is short for GNU GRand Unified Bootloader, is the typical boot loader for most modern Linux systems.The GRUB splash screen is often the first thing you see when you boot your computer. It has a simple menu where you can select some options. If you have multiple kernel images installed, you can use your keyboard to select the one you want your system to boot with. By default, the latest kernel image is selected.The splash screen will wait a few seconds for you to select and option. If you don't, it will load the default kernel image.

4. Kernel

The kernel is often referred to as the core of any operating system, Linux included. It has complete control over everything in your system.In this stage of the boot process, the kernel that was selected by GRUB first mounts the root file system that's specified in the grub.conf file. Then it executes the /sbin/init program, which is always the first program to be executed. You can confirm this with its process id (PID), which should always be 1.The kernel then establishes a temporary root file system using Initial RAM Disk (initrd) until the real file system is mounted.

5. Init

At this point, your system executes runlevel programs. At one point it would look for an init file, usually found at /etc/inittab to decide the Linux run level.

6. Runlevel programs

Depending on which Linux distribution you have installed, you may be able to see different services getting started. For example, you might catch starting sendmail …. OK.These are known as runlevel programs, and are executed from different directories depending on your run level. Each of the 6 runlevels described above has its own directory

ROLE OF KERNEL:

kernel is the main component of a Linux operating system (OS) and is the core interface between a computer's hardware and its processes. It communicates between the 2, managing resources as efficiently as possible.With an impressive numbers of code lines, the Linux kernel is one of the most prominent open source projects and at the same time, the largest available one. The Linux kernel constitutes a piece of software that helps with the interfacing of hardware, being the lowest-level code available that runs in everyone's Linux operating system.

- It provides a set of portable hardware and architecture APIs that offer user space applications the possibility to use necessary hardware resources

- It helps with the management of hardware resources, such as a CPU, input/output peripherals, and memory
- It is used for the management of concurrent accesses and the usage of necessary hardware resources by different applications.

FIRST IMPRESSION ON ZEPHYR RTOS :

- A small kernel
- A flexible configuration and build system for compile-time definition of required resources and modules
- A set of protocol stacks (IPv4 and IPv6, Constrained Application Protocol (CoAP), LwM2M, MQTT, 802.15.4, Thread, Bluetooth Low Energy, CAN)
- A virtual file system interface with several flash file systems for non-volatile storage (FATFS, LittleFS, NVS)
- Management and device firmware update mechanisms

----

regards

-yuvarajasekar.