I2C Protocol

→ Inter Integrated circuit protocol → I2C / I^2C / IIC

I2C was designed by Philips (NXT) for on-board IC communication (Short distance communication)

On-board IC communi. → On the same PCB connected Comm.

But now day company doing off-board communication.

I2C used for short distance communication → 2 meter distance Range.

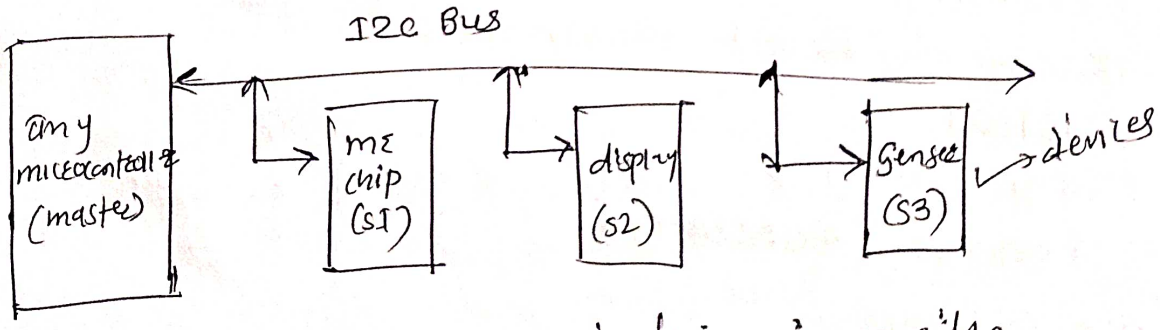
I2C designed for communication Between two IC (Integrated ci

USB device :- Ethernet to connect USB,

* UART device used for pair to pair communication, UART not designed for multiple slaves.

But why I2C so popular ?

→ We can connect multiple I2C device to I2C Bus. and you communicate with them.



In I2C multiple Slave communication is posible.

* In TV Remote we used RC5 and RC6 protocols used in Remotes.

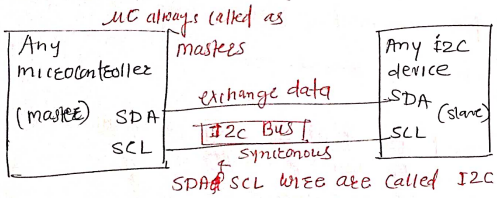
* I2C every device has its own address, now that's why master can communicate with perticular devices sensor, display, micro controller chip has separed own address.

* GPS we can not connect here, It's UART connecte compatable

* Features of I2c Protocols

1. I2C was designed by philips, now philisps ~~Remained~~ Renamed as NXP → "Next Expeelence" also it is on-board ic communicate

2. I2C is two wire communication Protocol.
   ① SCL → (Serial Clock)
   ② SDL → (Serial data)



SDA & SCL wire are called I2C bus.

List of i2c devices
+ AT24C08 (EEPROM)
(RTC) → Remote transmitter Receiver
+ DS1621 (Temp sensor)
+ ADXL345 (Accelerator)
+ OLED display
+ ADS1115 (ADC)
+ etc.

Why SCL used?
To generating clock pulse & clock pulse used for synchronizing / synchronization.

* Why used SDL?
→ Used to exchange data

* In the master and slave communication Only master can generate the clocks pulses,

3. I2C is half duplex and synchronous serial communication protocol.

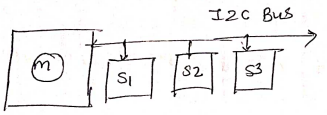
4. I2C is multislave and multi master communication protocol.

5. Speed of I2C procol (max upto 5mbps)
   ~~upto~~
   * Where we should do speed setting?
   → Only master we can do, gives and slave has by default speed setting. (company designed slave that it will gives default speed setting to slave).
   * Microcontroller are must faster than devices.
   → upto 100 kbps : standard speed mode.
   upto 400 kbps (master) master set to be Fast mode (FM)
   upto 1 Mbps : Fast mode plus (FM+)
   upto 3.4 Mbps : High speed mode (HS)
   upto 5mbps : Ultra high speed mode (UHS)
   ↓
   mega bits per second

Maximum speed of UART upto 1 mbps.

6. Every I2C device has it's own device address.
   Size of device address can be 7 bits / 10 bits.
   * The device address is given by cheap designer and we can not change it.

7. Maximum num of slaves on the bus
The maximum no. of slaves that can be connected on the I2C bus are limited by address space.



In i2c Bus same addresses of slave not allowed.
i.e. Restriction, feature or limitation.

all slave have 7-bit address.

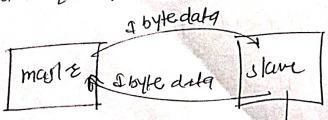
If you having: 2^7 = 128 slaves → 7 bit
2^{10} = 1024 → 10 bit

If you have combinations of 7 bit & 10 bit addressable slaves, 1024 + 128 = 1152 slaves

* Major problem in I2C is :- If master is not working then slave will become useless.

8. I2C is an acknolegment based protocol, master will get ack after every byte written into slave.

9. Data transfer direction is MSB first.

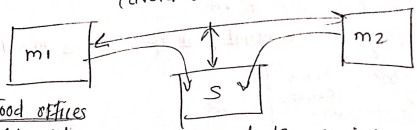


Responced by slave is called acknowledgement.

master & slave both send data ~~LSB~~ MSB to LSB.
Case of UART LSB first.

Slave will responce after one byte received.

10. I2C supports multi-master arbitration management (avoid data colision & curruption)



Friction, wait, struggle

Arbitration :- To avoid transmission conflicts on a n/w the technique which is used is called as arbitration.

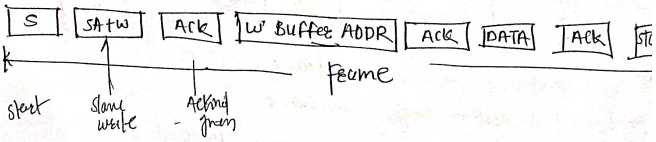
* * * * * Imp

* I2C EEPROM BYTE write algorithm ?

This communication is required, at least 8 step at most 10 step with this algorithm master can write into random location of the EEPROM memory

1. master issue start condition to beings this communication session.
2. master issue slave address with write option & master take acknolegment from slave.
3. master issue address of Buffer of slave where it has to write into.
5. master take acknolegment from slave.
6. master issue data for that write buffer.
7. master take acknolegment from slave.
8. master issue stop condition to end this communication sessions.

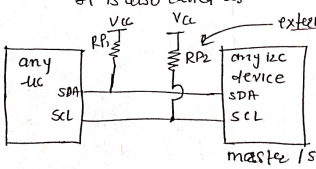
All steps in one lines



29/10/2021

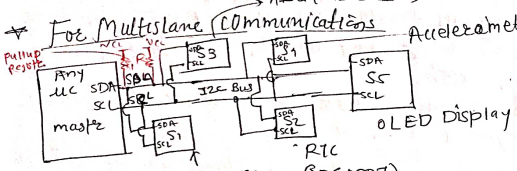
Hardware setup of I2C protocol.

It is also called as Bus setup.



External pull-up resistors ?
→ all the i2c devices are open drain devices. To adjust their (i2c slaves) input voltage level ext pull-up resistors must be connected on the bus.
Register values varies from 1kΩ to 10kΩ.

For Multislave communications



Slave 1 → EEPROM (AT24C08)

We can used two master, Use two master. Some slave connect master A & some slave connect to master B.

Master can communicate to all slave but one by one.

I2C Data Frames

1) I2C byte write frame
2) I2C byte read frame

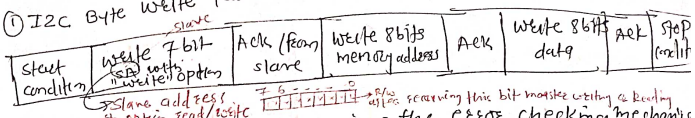
In UART we have only one data frame, There is no write or no read.

In I2C protocol there are two frames
1) → method of writing 1 byte into slave.
2) I2C by- → method of reading 1 byte from slave.

Protocol is always explain to master only.
1) master is writing → I2C byte write frame
2) master is reading → I2C byte read frame.

93.02 (completed)

1) I2C Byte write frame



I2C protocol they have not given the error checking mechanism.
If master send some data to slave (8 bit data) then if slave receive this data but due to some noise some bit are corrupted but still slave give you acknolegment, data is received so, I2C not check error.

Why in I2C error checking method not used ?
↳ Because I2C used for send the data for short distance. In short distance protocols data corruption chances very less. that's why.

~~I2C Hardware Setup / I2C Bus Setup~~

Acknolegment given by the slave.

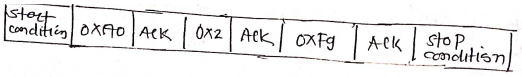
If slave is not available then there is no point of communication, If the slave is available & data transfer but slave not given the acknoledgment the master again send the data frame.

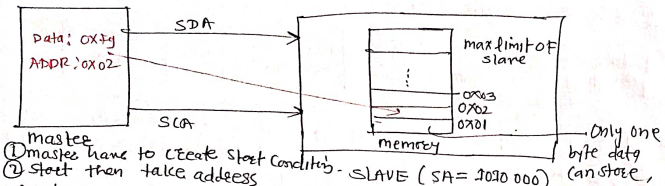
I2C Data Frame

Note: Every i2c slave has internal memory.
Memory size changes from device to device.

If slave not present you if slave not send acknolegment then send the error msg, we can send msg through LCD or Serial terminal (If error msg is max char then).
↳ It can display no. of char.

16 → 120 Ch





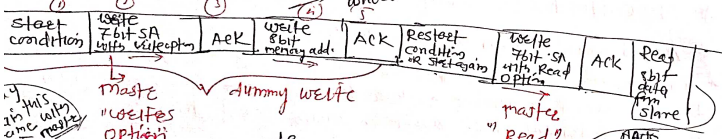
master

① Master have to create start condition - SLAVE (SA = 1010 000)
② Start then take address
-location of slave then slave give you acknolegment then again which location of memory you want to store data, so, take this address after again slave give ack, then that location save the data then stop condition. → This about write frame.

* If you want to store 2 byte data then?
↳ you have to used two memory location, we can not store 2 byte data at one memory location.

I2C Byte Read frame.

master has to write memory address from where master want to write the data.



master "writes option"

dummy write

master "Read" option

No ack given by master to slave

Frame explanation, generate
master has to start condition,
master has to write slave add with write option.
master has to check ack.
master has to write 8 bit memory address
master has to check the acknolegment.
After master has to generate restart condition
master has to read the 8 bit data from slave.
master has to write 8 bit data with Read option
-Ack master has to send acknolegment,
-master has to send No acknolegment (master to slave)
master has to generate stop frame.

In case of read frame, from start condition till ack is called dummy write. ① to ⑤.

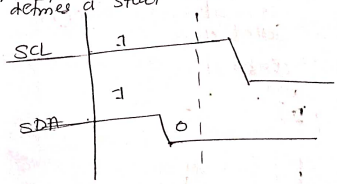
S → SA+W → A → m/c → A → data → A → P

S → SA+R → A → Read to write mem & read then its not allowe
↳ If not allowed

① If master write slave address with write condition then master allow to write memory address.

I2C Start Condition

A high to low transition on the SDA line while SCL is high, defines a start condition.

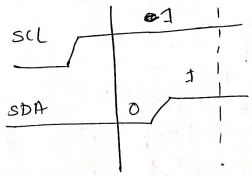


SDA = 1 } optional
SCL = 1 }

SDL = 1 }
SDA = 1 } starting condition
SDA = 0 ; }

I2C Stop Condition

- A low to high transition (change) on the SDA line while SCL is high, defines a stop condition.



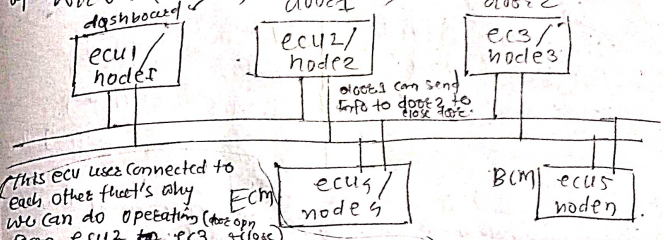
CAN Protocol

CAN → Controller Area Network. To improve the quality of automotive industry.

CAN Protocols → Replace Bunch of wires. with two wires, (complex wire with two wire). (It used firstly Automotive used But-Beez of its Feature

ECU → Electronics control unit its popular).

No. of microcontroller unite connect to only single wire. (use name)



In side of can protocol, It will work work on multimaster slaves.

ECM → Engine control module

BCM → Break Control Module

dashboard → It will take Information of seat, Battery Status Information, Engine control module, It will print the information on display, switch connected on the dashboard It can control the things, light, sroftenes,

Suppose :- In case door is close, ecu send sum Information to dashboard, then dashboard will not show Information. (If closed).

(CAN protocol implemented in 1983 by Robert bosh implemented 1991, first microcontroller bench used this CAN

VIBS → Advance Conteoll module (This also (Protected by Roboet bosh). (eneey ABS they peovid BCM)).

In that used BCM (Beeak conteoll modules) (one of sefty featuee of called)

There is difference Between Beeak conteoll module, (locking weels unlocking weels,

Old BCM	New BCM
There was deums Beeaking system.	Disk Beeak,
There was Rubbee,	ABS, This all Advance Beeaking system,
	Now we can conteoll like Speed easily.

* Aiebag conteol system: (when vehicle crash into something the electronic conteol unit (ECU) for the aiebag system eeact in milliseconds.)

* Role of CAN peotocol In cae,

* why we used ecu in cae?

→ To exchange the data from one ecu1 to second ecu2, Internally communication to each other. (If we have two wire are called as CAN BUS.)

(In CAN peotocol CRC error detection is there if single bit error then it will detect.)

* Aleeady we have UART BUS Then why we used CAN BUS?

→ Because UART BUS cannot cancel noice. Because of outside we have wind noice, because of this type of noice can be coreupt the data, when we will teansfee the data get coreupted (noice cancelation option is not UART) noice?

* We can used any other mateeial, to reduce insulator

→ so, UART not used multimastee, It is designed for PL to PL, Eeeoe detection ... is not effective.

* Why I2C & SPI is not used for the cae? (I2C & SPI used for Onboard communication (diff. communication chanel for Integrated)).

→ Because I2C & SPI designed for short distance. This is offboard communication.

suppose, One teuck suppose, In feont of ecu1 eeeoe detected to send the other ecu2, for that there will be distance more, more noice of engine, so, there peefeeable is CAN only. It will teansfee long distance data easily without any noice inteeupt.

→ whenever deivee sitting in the feont of teuck and it send some Indication it should go feom one ecu to second.

CAN - ISO 11898 → standaed Numbee, which can eepeesent can peotocol.

CAN is a seeial communication technology used especially for eeliable data exchange between electronics conteol ~~system~~ units (ECUs) in the automobile without using any host computee.

* OBD Onboard diagnostics?

What is OBD? (On-boaed diagnostics)

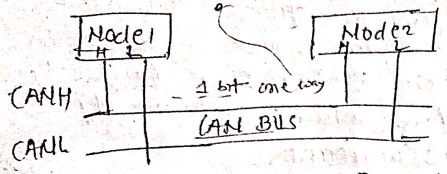
→ It's vehicles self capability of Reposting eeeoes. (It look like digital multimetee).

* UDS peotocol → UDS peotocols used inside of the CAN peotocol.

On ~~boad~~ board Dignostics

① can is too, wire protocol
— CANH → CANHigh
— CANL → CAN low

② Half duplex



CAN Bus used inside cars, buses, trucks, JCB,

③ Can protocol used different voltage method use for data bit transmission.

⚠ Our Hardware they will twist cable, used to avoid EMI → Electronics magnetic Interference shield → error

④ It is multimaster slaves used.

⑤ Can protocols is Broadcast

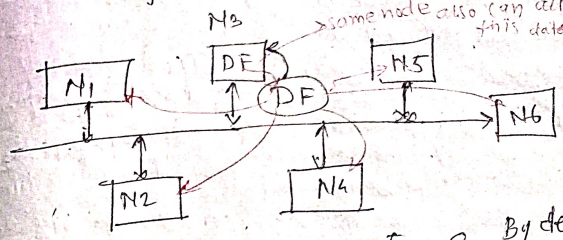
⑥ Asynchronous serial communication protocol.

→ Electronics Control unit.

Every ECU → master

⑦ IF one ECU not work is the car but other all ecu will work properly.

* IF any ECU can access this frame, that is called all ECU can access this, that is called Broad casting. (Default feature of CAN).



IF N3 create dataframe, By default it will be broadcasted. But we need to give information only that perticular ECU (Node) then there CAN provide one setting :- message acceptance filter (This is part of protocol setting only) setting (This setting in ECU), with the help of this setting which ECU should accept this dataframe msg, & which should the Ignore.

message acceptance filter → Only available for accepting the msg of which have to be Ignore.

Interrupt :- is the signal / disturbance which breaks the normal flow of execution of CPU to divert it's attention to another flow of execution with higher priority as against the current flow execution.

→ There is 8 Interrupt

① ISR → Interrupt service Routine :- Refer block of code ~~interrupt~~ to be executed in responce to the interrupt misd.

② Interrupt Vector :- Refer starting address of where ISR begins from.

③ Interrupt vector Table :- collection of one or more interrupt vetors at same predefined location of CPU architecture

④ Hardware Interrupt :- Refer an asynchronous event,

⑤ Software Interrupt :-

⑥ Maskable Interrupt :

⑦ Non-maskable Interrupt :

⑧ Non Spurious (Hyped) Interrupts :→ maskable & Non-maskable Interrupt are called Non-Spurious Interrupts.

In ARM two External Interrupts :

IRQ → Interrupt Request

FIQ → Fast Interrupt Request

Arbitration :- The arbitration mechanism is used to ensure that only one master has access to the bus at any one time. The arbiter performs this function by observing a number of different sequences to use the bus and deciding which is currently the high priority master requesting the bus.

CAN → 2

In-vehicle protocols

In luxuries car more than 90 ECU's.

All buses connected to the ~~to the~~ single bus, this is the network only and this network in side vehicle is called In-vehicle electronics networks. is nothing but the except can protocol only.

* every ECU have a microcontroller, that's why it a master.

* maximum speed of CAN protocol is 1 mbps bcoz 40 meters range.

After 40 meters if you want to used them decrease the speed then you can communicate. (Signaling Rate) mbps.

Eg: 500 meters at 125 kbps (kilobit per second

* what is Arbitration ?

* MAX Node on CAN Bus :- Upto 30 Nodes we can connect, If you want to connect more than 30 Nodes then you have connect more bus input impedance. (low-voltage, high current) is called high impedance (result) for you have to give high current.

* CAN available in two versions (CAN 2.0) → Bosh disigned

① Standard CAN (version 2.0A) 11 bit identi

② Extended CAN (version 2.0B) 29 bit identi

Identifiers Represent type of msg.

1. engine ~~control~~ temperature message ECU_3
2. light control message (front) ECU 2
3. light control message (rear) ECU 4
4. Crash detection message ECU 1 (ecu has connected front side of car)
5. ABS message ECU 2
   ↓
   Airbag ECU

Priority 1, 2, 3, 4, 5.

It will get highest priority.

when in can protocol any node send to data frame inside data frame the number will decide the which will get the highest priority, this called identifier no.

① For Standard CAN → support 8 bytes

2^{11} → 208 0 to 207 this no. are the used as identifier in data frame.

② For Extended CAN →

2^{29} →

Two layer of CAN protocol

— (ISO 11898-1) → Data link layer → handle moving data into and out of a physical link of network.

— (ISO 11898-2/3) → physical layer

↳ high-speed CAN (speed ≥ 125 kbps)

↳ low speed CAN (speed < 125 kbps) → hardware equipment

→ physically connects two stations

★ CAN protocol support two topology.

① line Topology → The host directly connect all nodes over one bus line is a line topology

② Bus Topology.

all the various devices in the network are connected to a single cable or line

★ low cost, lightweight Network;

Only two wires are used for ECU connection, and one shield used, not that that's why between light weight network, nowday ECU and MC price less that's why low cost

Standard can support 8 bytes while (CAN FD protocol support 64 bytes data in one frame

CAN Flexible Data-rate (CAN FD) ⇒ Advance version of CAN

Interview Question

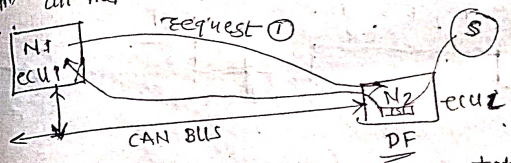
★ What is Difference Between CAN and CAN FD? ~~Advance Version~~

→ Standard CAN and Extended CAN is a CAN Both are CAN protocol version, CAN FD is a advance version.

→

★ Can support Arbitration (If multimaster happen in protocol then arbitrary will come in picture)

★ Remote Request facility all node can transmit & Receive the data.



One ECU can send the request to other ECU to send the data frame.

★ For reduce the traffic they have given this Remote Request Facility.

Remote Request Facility :- It says like that don't send me unnecessary data, (whenever I gave the request then send me data.

This is not an automatic process, for that we have to write a logic.

* Support autoretransmission of frame.

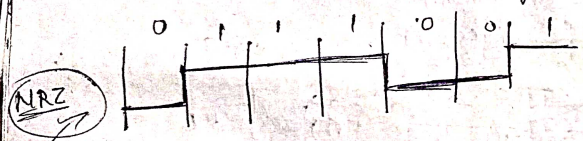
DF → Data Frame → Inside data frame, there is binary. (Inside of DF)

* Every Node are Trans-Receiver.
* Any node detect Error.
* 32 time Node can transmit data 33 node will get block (some time only).
* Protocol supports different Error, bit error, ack error, form error, CRC error, Stuff error.

ACK :- Acknowledgement

CRC : Cyclic - Redundancy check.

* NRZ → Non-Return Zero. (This is taken care by hardware)



Application of CAN Protocol

Elevators, escalators, Building automation, Medical Instruments and equipment.

→ If voltage is +ve it will send '1' after five '0' bit, if -ve then it will send '0', after five '1' bit.

MARCH 24

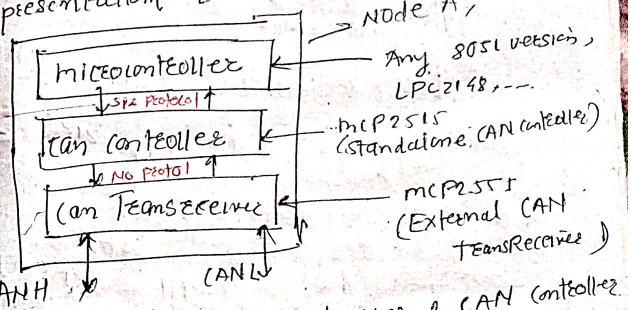
Now, what is component inside of ECU.

* Inside every ECU you have ① microcontroller, ② can controller, ③ can Transreceiver.

Apart from that, some ECU having ① sensor, ② LED & Bulbs ③ motors ④ Display ⑤ switches

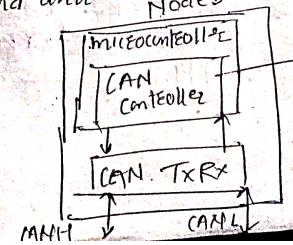
used with door ecu, power window Option } LED-Bulbs ①, ②, ③ components sensor, are not mandatory, But microcontroller, can controller, Can Transreceiver used compulsory every ecu.

Representation ECU



* We can used, microcontroller & CAN controller single unit, and CAN TransReceiver in second unit.

Node B or ECU B

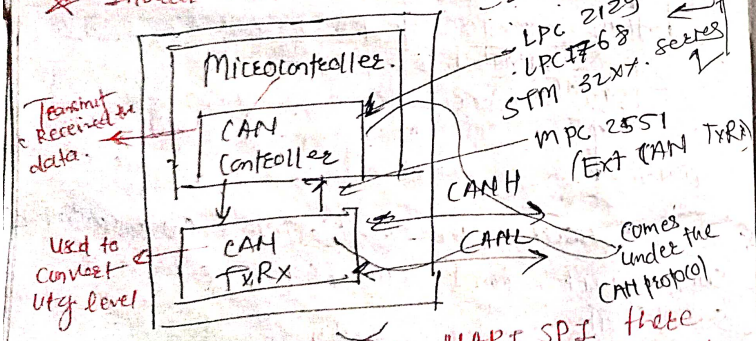


* Why company used external CAN controller?

→ Because there many companies used, they knit their own, work on that, they will not change total project operation they used externally CAN, But if CAN externally used then we need to add protocol, if it is inbuild CAN controller → no need protocol (company don't want to change project But want to add new feature then company used the external CAN controller. If used the external CAN controller then you have to used protocol) For communicating SPI protocols used here.

* There is no protocol used between CAN controller and CAN Transceiver.

* Inbuilt CAN controller.



ARM controllers
LPC 2129
LPC1768 series
STM 32XX
MPC 2551 (EXT CAN TxRx)

* IF there Inbuilt I2C, UART SPI there is not IC numbers, If it is external protocol then there is IC Number, LPC 2129, --- etc

In ECU there is DLL and ~~Phy~~ PL
① Data link layer ② Physical layer

CAN controller → is called Data link layer, which receive data given from CAN controller it link through the DLL.

CAN TxRx → Physical layer.

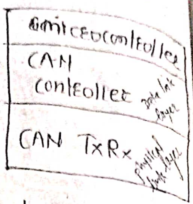
A Role of CAN controller (called)
1. create & send data frame (messages)
2. receive & store messages (data / remote frame) (It has inbuilt buffers)
3. error handling
4. bit-stuffing
5. bus-off handling
6. retransmission of lost message
7. arbitration (observed all bus & then send which Bus has higher priority)
8. baudrate setting / speed of transmission
9. message accepting setting
10. Interupt microcontroller upon transmitting & receiving as well.

* Suppose you have 10 ECU for broadcast it's baudrate should be same (other wise they will not synchronised).

* How will you gate the baudrate? → baudrate (no. of bit per signal unit)
→ use one register, used formula :
* Can controller Interupt facility also.
* CAN controller is nothing but Data link layer
* ISO 11-8 → it is a roll of data link layer.
* Ro

Role of physical link layer OR CAN TxRx

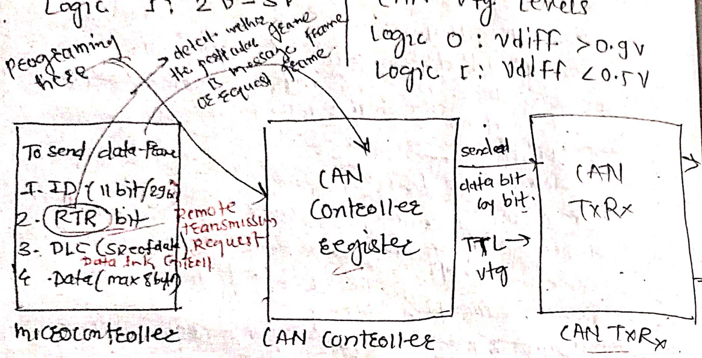
1. TTL (Transistor logic) vtg level nothing But microcontroller vtg. level. & Vcc - verm
2. To protect CAN controller.



- Can Control is inside microcontroller

TTL voltage levels
Logic 0: 0V - 0.8V
Logic 1: 2V - 5V

CAN vtg. Levels
Logic 0: Vdiff > 0.9V
Logic 1: Vdiff < 0.5V

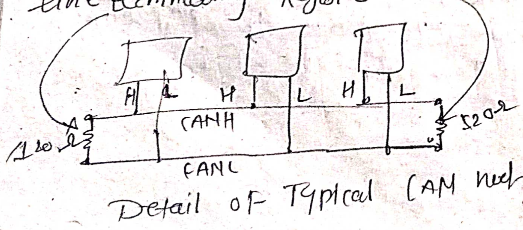


- We can gives data from sensor also.
- If data is zero then also it will send 1byte.

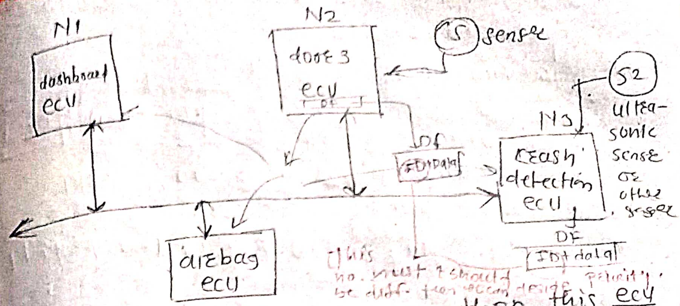
* data frame is nothing But Sequence of Binary.

→ Hardware setup are also called Bus setup

Line terminating Register



Detail of Typical CAN node.



suppose:- N2 door not close then this ecu create a data frame and send, Inside data frame you have [ID + DATA], whenever this data frame Return on Data frame, it will goes broadcasted. But here only dasboard ecu should Receive msg, (N1), so, for that we have to give setting is ~~message control unit~~ mesg acceptance filter

(CAN vtg)

* when door 3 (N2) and crash detection (N3) send the data frame at a time, that time has to send the first crash detection data frame (frame).

* In can protol message (frame) are having priority. Priority of mgs depend on Identifier.

* size of Identifier is 11 bits (version 2.0A) Standard Can → 2^{11} → 2048
or it can be 29 bits (version 2.0B) extended CAN
~~2^29~~ → 512GB → Identifiers are available)
2^{29} = ??

* message priority of messages.
→ Smaller the ID Number, higher the message priority.

Example:- Priority with ID

10, 500, 300, 25

Message (data frames) ID

1. Crash detection ----> ID (Highest priority → Less ID No get first priority) (1)
2. Door status (open/close) ---> 25 (2)
3. Engine temperature ---> 500 (4) (Lowest Priority)
4. Seat belt status ---> 300 (3)

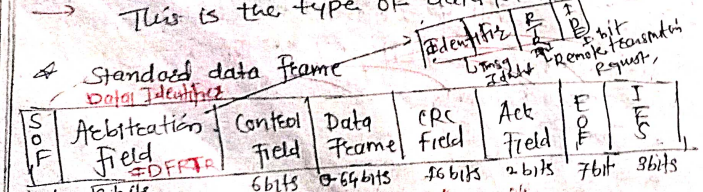
* Identifiers play major Role in the priority

* Data Frame:- This frame is used to send a data within a CAN Network. (Using Programmable)
* Remote Frame:- This frame is used to request a data frame from ECUs / Nodes.
* Error Frame:- This frame is used to represent error (transmitted when any ECU detect the error). (Non-programmable, once will still generate these frame automatically when a condition is met)
* Overload Frame:- This frame generated when node is busy. (In todays date ECUs will not generate overload frame)
(This frame less importance in the CAN protocol)

* Difference Between standard data frame & Extended data frame.

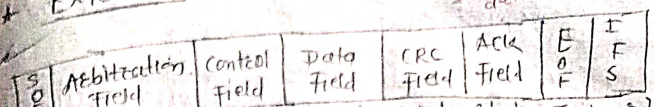
→ This is the type of data frame in CAN.

* Standard data frame (Data Identifier)

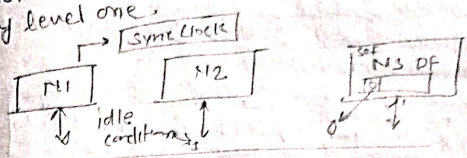


SOF → Start of frame
EOF → End of frame
CRC → Cyclic redundancy check
IFS → Inter-frame space

* Extended data frame (MCP251)



SOF (1 bit) → Idel condition of CAN Bus Represented by Level one.

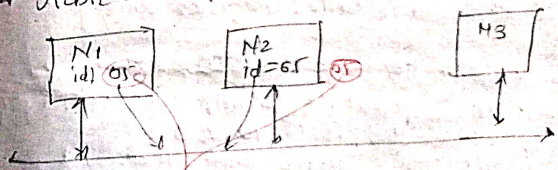


Clock synchronization When the any one node start of the frame, all node clock synchonised same. (Baudrate must be same) If diff they will not synchonised

In CAN protocol 0 → Dominant Bit (High, प्रभावी)
1 → Recessive Bit (कमजोर दबेगा)

It is single dominant bit (0).

* Arbitration Field (12 bits) (first ID)



If both Id are same, If both node becomes winner in arbitration process then both node will transmit data frame after Identifier field. That lead to "bit error".

Data length code → to send data - to data frame (How many bytes of data)

* Control Field (6 bits)
  1. ID (05)
  2. RTR bit (0) — Remote Request frame
  3. DLC (2)
  4. Data (10,20)

5	4	3	2	1	0
IDE	RB0	DLC3	DLC2	DLC1	DLC0
0	0	0	0	1	0

DLC field

Data length code

* Data Field (max 8 bytes)
  This field consists of a data to be transmitted within a data-frame.

This is for experience people only.

One data frame you can send 8 byte data

If you want send more than 8 byte data in CAN used CAN tp layer

CRC Field: (16 bits) (cyclic Redundancy code — error checking method)

15	...	1	0
15 bits CRC value			CRC delimeter

↳ cyclic Redundancy click.

Receive Buffer → is called Register Only

Suppos node A generate the ID, RTR(0) DLC(2), DATA(10,20), & 15 bits CRC value and it send to the node B means it will broadcasting, after that node (B) will receive this data and generate same 15 bit CRC value and matching with node A Then if data & value is same there will be no error, otherwise error, and no reply, there will be acknowledgement will generated.

* RTR It has to write '0'.

10-11-2021

node A (Tx) — CRC delimiter — Ack slot — Ack delimiter — node B (Rx)

sending data frame	received info into receive buffer
1. ID (5)	ID
2. RTR(0) (Remote Request)	RTR
3. DLC(2) CRL	DLC
4. DATA (10,20)	DATA
15 bits CRC value	15 bits CRC value

Generate its own 15 bits CRC value

Check match or not

Predefined time.

If it comes not 2 bits then there is error

CAN BUS

Error checking taken by CAN controller in that Data link layer.

If there is error then node A will retransmitt the data, it happened automatically.

+ EOF → Bus Become free. (Represent end of transmission).
+ EOF → is sequence of 7 consecutive recessive bits (1111 111)

IFS (3 bits) (Inter Frame Space)
↳ It is used to provide a small delay (or pause) between the last frame & next frame.

CAPL scripting is a c-like language.
Sending & receiving a data-frame using CAPL script

* CANoe & CAN Analyser.

"kapple"

CAPL → Communication Access programming language.

1. It is a scripting language that is used to access CAN protocol with logical operation s.
2. ~~which~~ CAN n/w using the script code which is almost like-c.
3. This script can be used with vector CANoe & vector CAN Analyser.
4. Using CAPL we can do simulation also,

It will work like Plash magic.

example:-

```
includes {}
variables {}
on start{
  write ("Hello world!!");
}
```

```
int i;
for(i=0;i<5;i++)
```

13/11/21

Data Frame:- To exchange data.

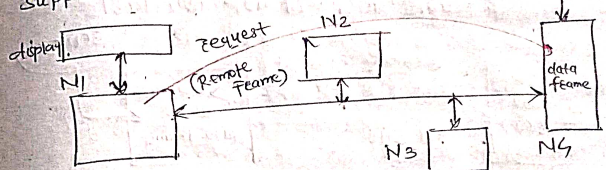
Remote frame:- To request data from other nodes.
(data → Frame)

Data Field → is not available in Remote frame

* Remote Frame:- the main purpuse of Remote Frame to reduce the trafic,

sensor output always 4 byte.

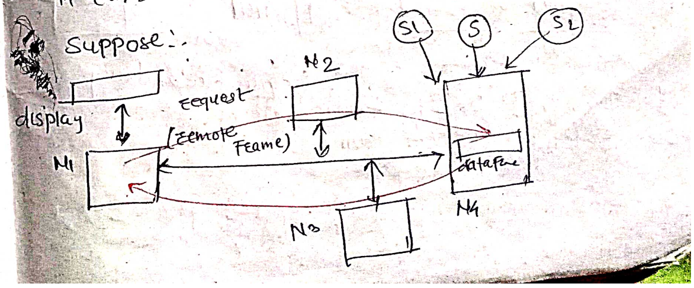
suppose!



when data transfer to N4 through the senser, But N4 not send that data automatically to all ECU's that time if N1 want ~~som~~ data, N1 will request to the N4 to send the data Because of that also trafic will reduced, If other Node whant data that will request Only, # so, that work ~~done by~~ is called Remote Frame, to request to send data.

Means N4 send the Remote Frame to N1. For that N1 has to create data Frame then it can be send ~~data~~ remote frame.

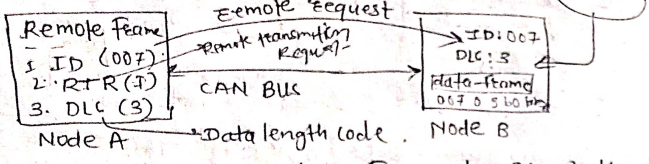
Suppose:-



If there is two or more sensee in above diagram then ~~you~~ ~~name~~ N will send the Request to Ng but from which sensee data should receive; so for that every sensee have separate Identifiee.
dlc ; id ;

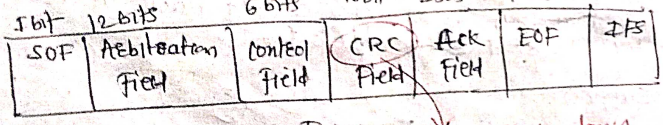
Rule to Respond Remote Frame

In Remote Frame data will not available.



Node B has to data Frame to accept the Request to Remote Frame, Both has to same ~~be~~ Identifiee, same DLC. ~~and~~ +

Remote Frame & Data Frame is same accept data field → It is not in Remote Frame and RTR value is 1, & Dataframe RTR bit is 0.



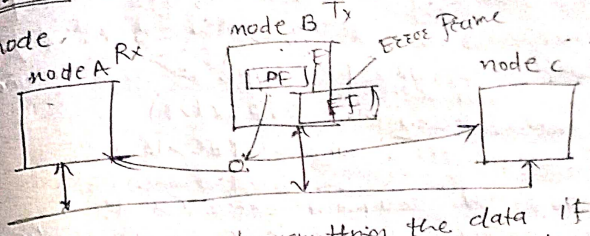
Types of Remote Frames

1. Standard Remote Frame
2. Extended Remote Frame

15/11/21

* Using dashboard, It can say that which door is close or open.

* Error Frame :- Suppose, there are 3 node,



~~If~~ when node B transmitting the data if he detect the some error then ~~and~~ time node B first send the error to Error Frame and every node have capability to detect the error Transmitted as well as Receive

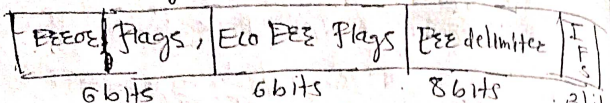
Error Handling ~~for~~ controlled by ~~data~~ Frame (Taken care by data link layer) totally depend on the data link layer, Error Handling ~~both~~ controlled by CAN controllee.

Total Five error in Error Frame:
two are detected by transmittee → 2
~~three~~ by detected by Receive. —3

If the transmitted node detect the error then transmitted node send error Frame. some for Receive, If the error ~~detected~~ ~~or Data Frame~~ transmitted on the BUS then all node will receive the error.

If ongoing transmittion is there then all data will get stop. Error has highest priority than Data Frame & Remote Frame.

Error has only three Field!.



Error Flags depend on type of error frame

1. Active Error Frame (TEC < 128)
2. Passive Error Frame (TEC > 128)

(TEC) → Transmit error Counter.
This counter increments every time when a node sends an err frame.

(REC) → Receive Err Counter &
↪ This counter is incremented upon receiving an error frame.

TEC & REC values is this are numbers.

* Error Confinment (कारावास) mechanism is CAN ECU's.
→ If Error generated continuous then error Confinment mechanism.
* The node who sent error frame will increment TEC value by 8.
* The node who receive err frame will increment REC val by 1.
* Upon successful transmission of data/remote frame TEC val. will decrement by 1.
* Upon receiving a data/remote frame, REC val will decrement by 1.

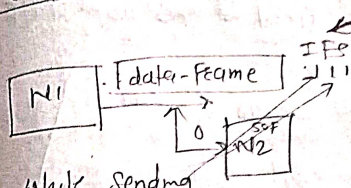
TEC > 255 → Bus-off state
Then that node enter into Bus-off state.

(6-11-21)

* Overload Frame

This frame generated when node is busy.

Overload Flags (6bits)	0	Overload delimiter (8 bits)	IFS (3bits)



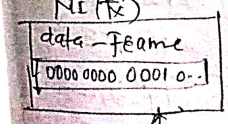
While sending these two bits on the bus, if zero detected, the 0V frame will be generated.

If zero detected here, interpreted as SOF while sending the IFS after that frame, if SOF send data, that the overload frame will say hey stop here IFS sending data.

(Bit error, ack error, form error, CRC error, stuff error)

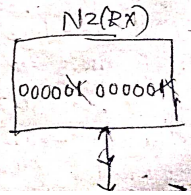
* what is Bit stuffing? (Stuff)
→ To ensure enough transitions to maintain synchroniza tion, A bit of opposite polarity is inserted after five consecutive bits of the same polarity.
• The stuffed data frames are destuffed by the receiver.

Suppose:



Stuffed bit

000000 1000001



If N1 (Transmit Five consecutive 0 bits (arithmetic operation) after that. it will send opposite bit '1', then there zero will not consider. Bit error
00000 1 → This bit is called stuffed bits and Removing stuffed bit called destuffing.

* Bit stuffing can't be affected on following fields, → cyclic Redundancy check
- CRC Delimiter → 16 bit
- Ack Field → 2 bit
- EOF → 7 bit

Above field sizes are fixed & are not stuffed.

* Different types of Error in CAN

1. Bit monitoring } → Bit level
2. Bit stuffing
3. Form Error (Frame check) } → message level.
4. Acknowledgment check.
5. Cyclic Redundancy check

→ Transmitted node

(2) Bit stuffing Error → If more than Five level occurs on the bus, of same level a stuff error signaled.

(3) Form Error → Some part of CAN message have a Fixed format, If a CAN Controller detect an invalid value in one of these fixed fields, a form error signaled.

(4) Ack:gment check :- Transmitter wait for acknowledgment → If Tx. can't detect dominant Ack ... Acknowledgment

(5) CRC Error :- If Node1 one sending some CRC value to Node2, then node2 will try to match this value If value not matched then Node 2 will think data is currupted & node2 will generated the error frame.
→ This called CRC error.

* RTOS :- Real-time System (eg. airbag system, autopilot, missiles, etc...)
→ It is work differently as compare OS. RTOS Perform on the tasks, but is specially designed to run applications with very precise timing and a high degree of reliability.

* Can bit time ? →
* How many SFR's in CAN ?
→ CAN1 → 13 SFR's } 26 SFR's
CAN2 → 13 SFR's

quanta → CAN time measuring in quanta
↓
Bit

* Bus Arbitration → will check priority which slave should select first depend on this priority.

... Universal Asynchronous Receiver Transmitter

* What is UART? (Gordon bell American engineer)

→ UART designed for serial communication. There is inbuilt UART in microcontroller. If we have In built UART we will get TXD and RXD two pins.
TXD → Transmit data.
RXD → Receive Data.

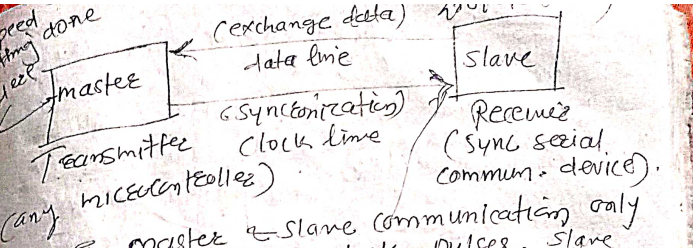
* UART is ~~Asynchronous~~ an on-chip peripheral of microcontroller designed for ~~asynchronous~~ serial communication where data format & speed of communication is configurable. Boudrate is called communication speed.

* Serial Communication :- is process of sending one bit at a time. (eg. Bluetooth module, USB device, etc.)

* Parallel Communication :- It is a process of sending 1 byte at a time.
eg. old printers, alphanumeric LCD, etc...

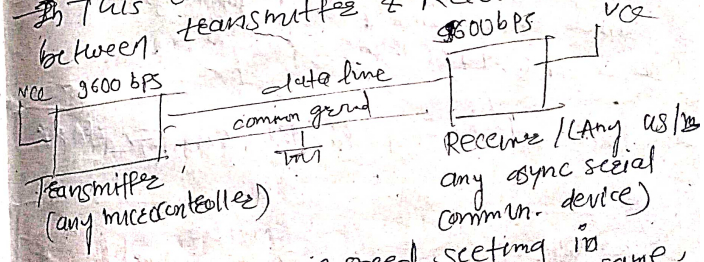
* There two type of serial communication
+ Synchronous serial communication
+ Asynchronous serial communication.

(1) Synchronous Serial Communication :-
→ A common clock line is used between transmitter & receiver for synchronization.
→ Here master can send 1 bit using 1 clock pulse.



→ In master → slave communication only master can generate clock pulses. Slave is not master / Intelligent to generate clock pulse.
Here master can send 1 bit using 1 clock pulse.
In synchronous serial communication <= 400kbps speed we can set for master.
How to know speed of sheet ?
=> datasheet of the device.

* Asynchronous serial Communication :-
This common clock lines not used between transmitter & Receiver.



→ Using the same speed setting in transmitter & receiver must be same, then we can async commu will synchronize ?

ARM Controller work on 3.3 V+.

* List of asynchronous serial Comm devices
  + Bluetooth module
  + GPS module
  + GSM module
  + RFID Reader ✓
  + Xbee module
  + Wifi module
  + etc..

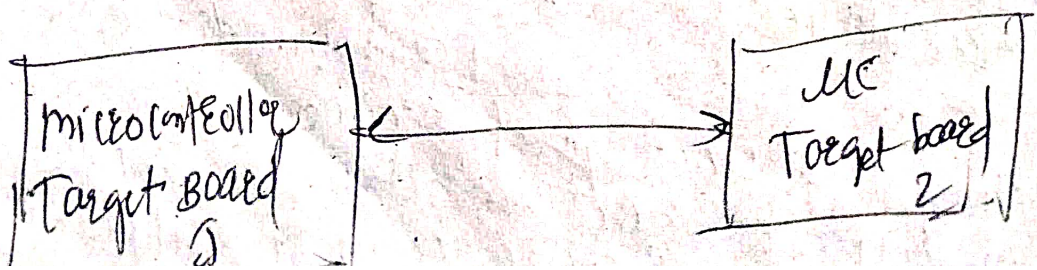
LPC 2129 मध्ये CAN available आहे but LPC 2148 मध्ये CAN available नाही, आणि LPC2148 ला Flash memory & RAM ही available आहे Comm devices

* List of Synchronous serial
  + EPROM
  + Real-time clock (RTC)
  + Digital accelerometer
  + Digital thermometer (temperature sensor)
  + OLED display
  + etc...

* Asynchronous serial Comm. based protocols
  UART
  CAN
  CAN FD & USB

* Sync serial Comm based protocols
  I2C, SPI, LIN etc.

* Understanding Real-time case of UART

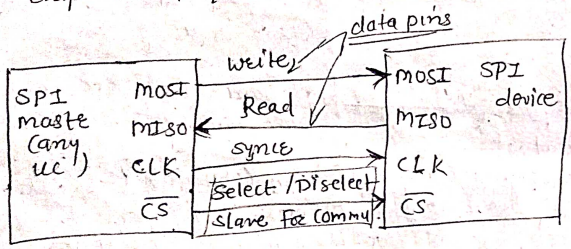


SPI Protocol → 10-04-2023

SPI is designed by motorola for on-board IC communication (or) for short distance communication).

Features :- SPI

1. SPI was designed by motorola for short distance communication.
2. SPI is 4 wired commu. protocol
   - Master Out, Slave In (MOSI)
   - Master In, slave out (MISO)
   - Serial clock (SCK / CLK)
   - Chip select / Slave select ($\overline{CS}$ / $\overline{SS}$)

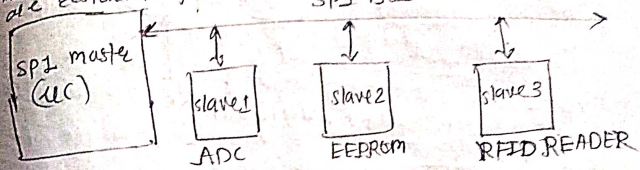


By giving logic '0' you can select slave commu. By given '1' you can deselect.

3. Hardware Complexity will Increase when you Increase the slave in SPI.
   * In I2C Hardware Complexity will not Increase when you Increase the slave.

This four wire is called as SPI bus.
MOSI, MISO, CLK, CS.

4. MOSI pins & MISO pins are data pins
5. It is Full duplex commu. protocol because here two wire used. Because here used synchronous serial commu. protocol.
6. SPI is multislave communication protocol.
7. maximum number of slaves that can be connected on the SPI Bus, are restricted by the hardware.



8. Speed of communication
   - SPI does not define any speed limit; implementation often go over 10 mbps.
9. Acknowledgement is not supported by the SPI.
10. Direction of data transmission can be MSB first or LSB first.
11. SPI has 4 modes of operation. And modes are depends on the two parameters.
    - clock phase (CPHA)
    - clock polarity (CPOL)

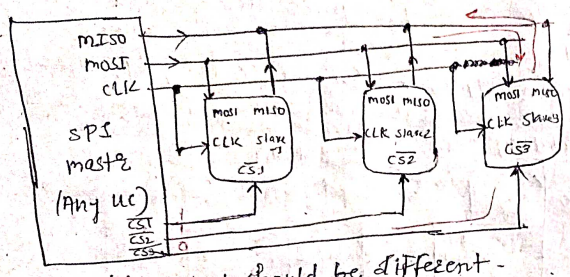
CPHA: Represents on which edge (Rising / Falling) data has to be sampled (read).

CPOL: Represents the base value of clock pulse.

mode	CPOL	CPHA	DATA sampling	Data transition
0	0	0	Rising edge	Falling edge
1	0	1	Falling edge	Rising edge
2	1	0	Falling edge	Rising edge
3	1	1	Rising edge	Falling edge

To connect three slaves 6 pins are required

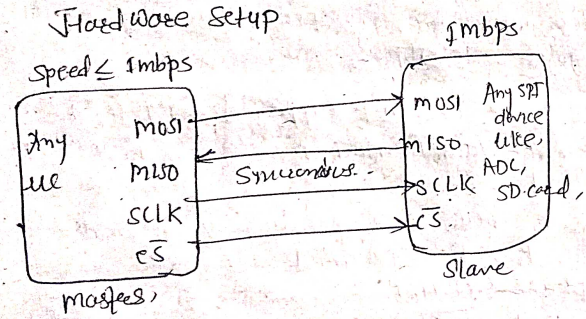
H/w setup: Maste & 3 Independent slaves



chip select should be different.

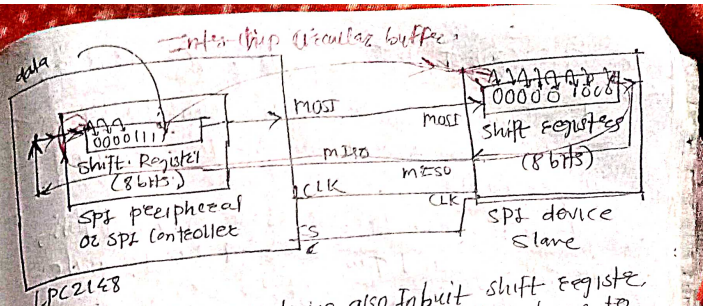
✱ Working of SPI Protocols.

Hardware setup



✱ Peripheral Method

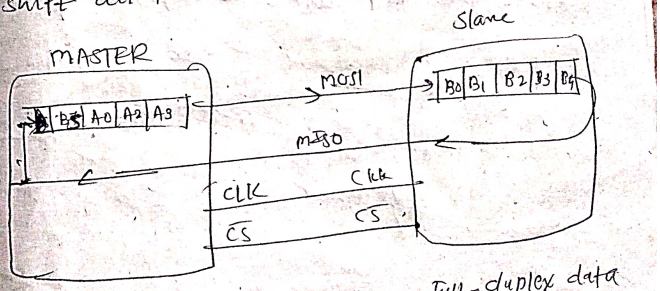
In Any microcontroller there are inbuilt SPI controller or SPI peripheral.



Working :- In SPI device also Inbuit shift register, If you want send data, we have to write data shift register only. Here Shift register is nothing but the special Function Register After transmitting data Internally flag will set. data will shifted serially, Serial output, parallel Input. (PISO) or (SOPI).

Whenever master generate one clock Pulse, that full duplex communication occure.

After generating one clock pulse master will shift all the bits towards to right.



- During each SPI clock cycle, a full-duplex data transmission occures. That's means, the master sends a bit on the most line & the slave reads it, while the slave sends a bit on the MISO line & the master reads it.