CHARGING INFRASTRUCTURE

Prof. Apurv Kumar Yadav

Department of Electrical Engineering

Indian Institute of Technology Roorkee

Week-12

Lecture-57

Lec 57: DC Charging System (BEVC-DC001)-III

Hello everyone welcome to this lecture number 57 of NPTEL lecture series on charging infrastructure and today we continue our discussion on DC charging system using DC 001 charging systems. Now we gone through the communication over which the DC 001 charging system does the communication which is nothing but the CAN communication. We have studied some of the aspects of CAN communication, why CAN communication has been used, how the arbitrations are win by the different nodes who are communicating. And we have also seen the importance of CAN database files.

And we have also seen how the CAN data frame, which is 2.0a frame, 2.0b frame looks like. So in this lecture before starting let us recap in quickly what is the CAN database file we were discussing in the last class. So these are basically the text file containing different information of the CAN messages and in CAN message we know that we have signals signals or you can say data or you can say information. and for each signal for each data you are getting a raw data so you have to actually convert that raw data into the actual data or you can say the data which is being understood by the controller to perform the required action so you need to do certain operations on those raw data to actually obtain the actual data which can be understood by the controllers So there it will also define the parameters and the parameters are defined for corresponding signals and those parameters can be used to actually extract the real information from the raw data which are coming with the CAN messages.

and we have already discussed that in CAN message we can have multiple signals or data into it and in CAN database file generally provided by the device manufacturers and generally in CAN database file you have several informations like message name the message id which is one of the again important thing which defines the priority of the messages which are been pushed onto the CAN bus then dlc bit which is data length code bit then it will also have signals and channel name which are again what are the signals which are available in that particular CAN message then the size of that particular signal and the start bit of the particular signal in that CAN message because we can send 64 bits of information in one CAN message because we can have n number of signals in those 64 bits so in those 64 bits at which place that one particular signal is starting that is also incorporated by defining the start bit then you have the signal type what kind of signal it is either it is an integer or floating variable what kind of that particular signals you are expecting their byte order whether it is little indian or big indian that information is also been incorporated in the can database the scaling factors their units what will be the units either you know if it is the speed either it is rpm or meter per second different things related to units will also be defined then the range what is the maximum minimum value of that signal you can expect and then what is the offset which you need to give in order to obtain the real data from the raw data which are coming in the can message so let us take one example of a CAN message in that example we will see how we can extract the actual data from the raw data which is incorporated in the CAN message so let's see in the CAN message you have several bits which are aligned here you have messages

here you have start SOF then after SOF you have message id and different bits are defined then you have the actual message then you have the actual data and in that data you can go up to 64 bit maximum and then on the back side you have CRC you have acknowledgement and so on so in this thing let's say in the data we are just sending one signal so let's say the data is assume the data is 038 that corresponds to 1000 in decimal that data you are receiving and let's say in this particular CAN message you have a message ID and that message ID is incorporated in the CAN database and corresponding to that message ID it is been defined that how that data can be interpreted which are coming in and assume you had just one signal just one signal it is there and that one signal is that value the actual value of that signal is nothing but 0x03e8 which is 1000 decimal value now in this can database file you will have can message name will be there then you have message id will be there Then you have some key information of that CAN message, IDs and those things. Let us take DLC.

We will also define DLC bit. There are other aspects also, but we just take some few things. Let us say it has a scaling unit defined or scaling factor which has been defined. Let us take it will have offset value which has been defined and let's say it has the unit value. Now assume the CAN message which you are getting is let's say speed.

i mean example we are taking speed and the message id let us take 11 bit number so 11 bit number means we can say 0x any value let's take 1 followed by e followed by 2 any message id could be there so in the message id that number will be there 0x1 e2 which correspond to you if you open it bits you will find in binary format what it looks like and assume that it is sending the 16 byte of information so dlc value will be 2 why it will be 2 because you are sending 16 bits that means it corresponds to 2 bytes of information which are being sent and then scaling factor is there let us take the scaling factor defined as 0.2 and let us take the offset value is defined as 100 and let us take the speed unit is given in rpm so now this is the database which is been incorporated in the node or in the microcontroller so the moment this message is received first the message id is getting compared with the canned database message ids i mean in the database whatever the can message are there their message ids will be compared if it is okay fine if it is matching so then it is the valid message it will be used by this node so that message will be taken in and then the data will be extracted And in this the data we are getting assume the data will be 0x03e8.

0x means I am writing in terms of hexadecimal. So that will correspond to data is nothing but 1000 value in decimal value. So this raw data which I am getting is I mean raw data from the message which I am getting is a 1000 number. Now from this raw data we have to find out the actual data. So we can do the actual data to be equal to whatever raw data we are getting.

multiplied by scaling factor for that particular signal in the message so in this scan message having name is speed luckily you have just one signal that signal is having the scaling factor 0.2 or sometimes scaling factor can also be say the resolution that means what per bit will have the information you can say this scaling could be rpm per bit in this case and the offset value is 100 which is given and unit is RPM. So, this we can say raw data multiplied by scaling factor plus offset data, offset value which is been put in. So, the raw data was 1000. Scaling, this is given in the database, is been extracted from the database which is 0.2 and the offset value is given as 100 which is been extracted from the database.

So, this will give you 200 plus 100 and again the unit is in RPM. Finally, it is nothing but the 300 rpm. This is the actual data which corresponds to the speed value. So this is how the database is being incorporated in the controller and once the message is received the message IDs are compared with the CAN message ID which is been received or if it is a valid message ID it is taken in the controller and then the data will be extracted which is nothing but the raw data which you get it from the CAN message. Now that raw data has to be converted into actual data by multiplying it with the scaling factor which is like a resolution which tells what is the actual data when we multiply that factor with the raw data and plus the offset value which is being shown in this particular database file.

So like that you have to create the database and then in that way the two nodes can able to interact with each other. So, let us again recap about DC-001 standard. We have again input 3-phase AC. It supports up to 10 kW for 48V system, 15 kW for 72V system. And the detail communication is derived from IC6185-124B, which is again incorporated in AIS-138.

part two particularly an extra g which is released by aria india and also it is derived from gbt 27930 especially the digital communication expect it uses a can communication and it uses the gbt20234.3 connector with the standard 5 meter cable In the detail communication aspect of DC001, so it is primarily derived from GBT27930-2015 and is been incorporated in AIS138 part 2 as well. Now, the GBT27930 uses the SAEJ1939 or ISO 11898 standard to actually represent the physical layer and the data link layer. However, the application layer of GBT27930 is actually defines the representation of different parameters. How those parameters are grouped together?

What is their group numbers? What are their parameter group numbers? What are the IDs for messages? All those things will be defined in GBT27930. It uses a speed of 250 kbps.

It uses a CAN 2.0b identifier. That means it uses 29-bit identifier. which can actually have two rush hour 29 distinct messages. Then in this particular system, it has only two participants which are communicating with each other at a time. It has a charger, it has a BMS in the vehicle or primarily sometimes also called as the vehicle controller as well.

So this is the dedicated communication channel which has been created between the charger and the BMS. it does not have any connection with other CAN systems in the vehicle like the main CAN communication system which actually does communication inside the vehicle now this CAN communication is dedicated only for this charging systems and it has the fixed addresses the charger which is there will have the fixed address which is 56 in hexadecimal case and the bms will have the fixed address which is f4 hex that means if the charger says that they are dc zero zero compliance they have the address nothing but 56 hexadecimal if the vehicle or you can say bms in the vehicle says that they are dc zero zero compliant then they must have the address which is f4 in the hexadecimal case And the CAN data frames which are being transmitted between the charger and the BMS, they are transmitted in the form of protocol data unit. That means it defines how the CAN data frames are structured.

And they are particularly been defined in GBT27930 and also been incorporated in DC001 as well. Now, let us see how these things are defined, how the physical layer information and other things are defined in DC001. Now, we have already been discussing this thing that in physical layer defines the different physical properties of the signals which are shared between EV and EVSE or the charger here the informations like baud rate which is 250 kbps the identifier what kind of identifier the 27930-2015 standard is been following which is been same as been followed in DC001 so here they are using CAN 2.0b identifier which is a 29 bit identifier that means that you can accommodate 2 raised to the power 29 unique messages and the line must have an impedance of 120 ohm so that the impedance matching between the two CAN transducers can be obtained.

Now in DC001 which is again derived from GBT27930-2015 standard, here the CAN messages are having certain formats particularly if you look the can message only two or three things are something which has to be standardized rest other things are quite standardized for example the message id format now what could be the message id format for the can messages which are being shared so here this standard 27930 will define what could be the data frame format it can have that means the other architecture of the data frame cannot be changed like the placement of srr bits sof bit ifs bit crc those things cannot be changed only thing the message id how the message ids are being defined for different can messages because you know there could be N number of manufacturers who are making their vehicle DC-001 compliant and there could be N number of manufacturers who will be making their charges to be DC-001 compliant so how one can ensure those CAN messages or you can say those CAN database file is same and in that the message IDs the messages will have the same kind of message ID so that's why they define message ID format now in the message id format they are transmitted in the form of protocol data unit which is also called as a pdu which is a protocol data unit now this protocol data unit will actually define how the message id has to be defined now in this message id as we know it is a 29 bit message id so each message has the unique id which is defined by 29 bits of information so 29 bit of message id is there

now in that 29 bit how those 29 bits are defined let us see that now in this case we have 29 bits that means if we take in terms of four bits so you will require eight cross four bits of information that means eight chunks of four bit you required in that the first three bits are kind of avoided so we will avoid these three bits initial three bits then you will have the three bit priority field so this one this one this one are actually the priority field now this is the Our message ID. This is the format of message ID in DC-001 and which has to be followed. You are adding one new message in the database file.

You are modifying any CAN message in the database file. You must follow this message ID format while defining the message IDs for different messages. Then after that you have a reserved bit. So this is your reserved bit. then after that you have the data page bit so you have data page bit data page bit after that you have 8 bit parameter group number which is a PGN which is parameter group number

Now, what is a parameter group number? So, different messages which are shared between the charger and the EV will have different parameters included in them and those parameters are combined together or those parameters are grouped together and that particular group will be given a number which is called as the parameter group number. So, we have a 8 bit parameter group number. So, let us say 8 bit PGN, PGN, PGN, PGN, PGN, PGN, PGN. So, these 8 bits are defined for PGN or you can say parameter group number.

So, this is your 8 bit parameter group. group number now then you have a 8 bit destination address so your destination could be either the BMS or could be the charger and here we know if we define our charger it has to be 56H and if it has to be BMS it should be 4FH the address value so we have the 8 bit destination address now this is your destination address and last 8 bits this is your source address bits This is 8 bit source address. So, you have 8 bit source address.

So, this will combine the form. If you see this bit combine the form 29 bits of information and along with this plus 3 bit of bits which has to be avoided and that will actually form your entire 32 bit information which is nothing but if you send 4 bytes of message ID. So that, so each message which are been there will have the message ID and that message ID must follow this format because once it is following that format one can easily able to extract what is this message all about from where it is going and where it has to go, what is the source address, what is the destination address and different other informations they will be easily able to extract it. So this is how our message ID format are been defined. Now as we have discussed that while charging there are different charging parameters which are grouped into different parameter groups which could also be say that whenever there are communication between the charger and the vehicle there will be a lot of messages which are exchanged at different stages of charging and those messages are having certain parameters

And those parameters are grouped together and those parameter groups are defined by the parameter group number, which is a PGL. And by knowing the PGL, one can easily able to understand what messages which are being shared, what will be the stage of the charging the vehicle are in or the system are in. One can easily able to identify those informations. And I have already indicated the BMS will have the fixed address or you can say the vehicle will have the fixed address which is F4 in hexadecimal. And the EV charger will have the fixed address which is nothing but 56 in hexadecimal and in decimal case it is 86.

So that's a fixed address which has been defined for the charger and the BMS or you can say the vehicle. Now, this application layer will be defining these parameters, what are those parameters and those parameter group numbers. So, those information related to the parameters and parameter group numbers are included in the application layer of GBT27930-2015, which has been also been incorporated in DC001. Now both the charger and EV are to be programmed to identify the data content in that data field with the help of the parameter group number of that message. So from the message ID the PGN will be extracted and once the PGN will be extracted one can able to know what parameters or what messages which are been there and what is the source what is the destination.

those things will be defined in that PGN and from there one can understand what is this message all about and within each parameter group there are certain signals or you can say in each message there are certain signals and this signal will have the unique signal parameter number which is called as the SPN so from the message from the message which are being shared between BMS and the charger the message ID is extracted from the message id the pgn is extracted and from the pgn the vehicle will come to know what are the parameter group what is the parameter group we are talking about and in the parameter group there will be several spn or you can say signal parameter number will be defined and from there one can easily extract which signals we are talking about by understanding the parameter group number and by understanding the definition of the parameters from the can database file Now in BEVC DC001 charging systems it has primarily the six stages of charging and each stage has a set of messages which is to be transmitted and received from the EVSEs or from the vehicle. whenever there is a message it is been sent from the sender to receiver sender could be either BMS or could be the charger controller or you can say the charger because it's a dedicated communication channel between the charger and the BMS that means you can say it is having just the two nodes now AIS 138 part 2 defines just the four stages however If you understand in a simple manner, you can say that it has the six stages.

You can say it has the handshake stage, then handshake harmonization stage. The AIS-138 says that these two comes under the handshake stage. then you have the parameter configuration stage, then you have the charging state and then you have the charging suspension stage and the end of charging stage. Now in AIS 1 through 8 part 2, these two things are combined together in the charging ending stage. So they define primarily they have

family they have the four stages and apart from that they also have the communication error stage now in every stages you have certain messages which has to be exchanged between the charger and the vehicle battery you can say battery or vehicle both are used interchangeably now in the handshake message you have charger handshake message you have vehicle handshake message or battery handshake message in the handshake organization stage you have the charger organization message you have the battery organization message or vehicle organization message because in ais they uses vehicle recognition message while the gbt27930 uses a battery recognition message then you have the parameter configuration stage where you have several set of messages like battery charge parameter message you have charger time synchronization message you have the charger maximum minimum output parameter message you have charger output ready message you have vehicle charge ready message and in the charging stage you have battery charge requirement message you have charger charge state message you have battery charge state one message you have battery charge stage two message so there are different kinds of messages which are exchanged in every stages and then in charging end stage you have charger stopping command message you have vehicle stopping command message you have charger statistical data in end of charging stage message you have vehicle statistical data message however this messages we are showing as per the AIS 138 now in AIS 138 they also have the another stage which is the communication error stage so this vehicle error message and charger receiving error message comes under the

communication error stage so they have also defined one more stage and they clubbed some stages and they have defined one extra stage in ais 138 now this message names we have taken it from ais part to an extra g which is actually been derived from 27930 However, in GBT27930 the message names and the parameters are quite different. That means the message name means this messages name which as I have mentioned in GBT27930 you have battery handshake message while in AIS138 part 2 we have the vehicle handshake message. And also some of the parameters which are defined in that message will also be slightly different. So one can see those standard GBT27930 and AIS138 part 2 standard and then able to understand what are those different messages and those things have been there.

However, fundamentally the structure will remain the same. That means in these stages, the messages will remain the same. Within that, the signals may be different. The data within those messages can be different. Now, these short forms or abbreviations we have used are from the GBT 27930 standard. However, we use these messages as defined in the AIS 138 Part 2. In BEVCDC001, we have already discussed that the application layer has been derived from GBT 27930-2015 with some changes. Those changes include the addition of some extra parameters.

Some of them are like the protocol version number, the version of the protocol the charger and the vehicle are using. So that has to be exchanged between the charger and the vehicle. Then they have included different types of batteries, the battery manufacturer name along with their standard ASCII code, the battery pack number, the battery pack product code, battery pack charging time, how many times it has been charged and discharged, battery pack property right mark—whether it is licensed or private—that has to be indicated. Then the vehicle identification number, different parameters in the vehicle or battery organization parameter message. Similarly, they will also have Charger number, the charging station location code, decarbonization result, etc. in the charger decarbonization message.

And they also have the last charging end reasons, the last charge status, start state of charge during last charging, end state of charge during the last charging, distance traveled since last charging. Those things will be the additional messages included in different stages. And then they have also defined the different fault codes, the different list of fault codes which can occur. So those things one can obtain from the document released by the Ministry of Heavy Industry, Government of India, related to BVC DC 001. And obviously, one can also go and see in detail the things related to this in AIS 138 Part 2, especially Annexure G. Now, as we discussed, whenever CAN communication happens between two nodes, they will have a certain CAN database file, and in the CAN database They will define the message IDs, the different parameters related to those messages, like what all signals are present, what are their resolution, what are their units, what are their scaling factors, what are their offsets. All those things will be defined.

let us take some example one can go and see very much in detail in AIS 138 part 2 and extra G or GBT 27930 standard and then can able to implement those scanned database make those scanned database file as per those documents one can purchase those documents and use them now in this thing one of the message is vehicle status data now this message code is defined in GBT 27930 standard so they have the message code which is BSD and the source address is nothing but BMS or you can say the vehicle the destination address is the charger and they have the parameter group number so if you recall we have defined that in the can message frame it uses 29-bit identifier and in 29-bit identifier they will have the parameter group number which actually uses to identify what is this message all about that means it is indicating to which message so the vehicle statistic data is having the parameter group number nothing but zero zero one c zero zero in hexadecimal format we are indicating this h indicates the hexadecimal Similarly the battery charge requirement which is nothing but the message code is BCL and the source address is the BMS or vehicle. So vehicle is sending that message and the destination is nothing but the charger or you can say EVSE and it has the PGN which is 001000 and it is nothing but is the hexadecimal format what we have indicated.

Now, both the things have priority 6. They have also defined the data length. For example, vehicle statistics data has the data length of 7. Now, in this particular thing, what you have is you have the starting position. Now, they are saying the starting position with respect to the bytes.

What I mean by that is, since the data in this particular vehicle statistics data message is of 7 byte, that means you have 1st byte, 2nd byte, 3rd, 4th, 5th, 6th and 7th byte. Those bytes of information has been there. So the 1st byte which is there, this is let's say 1st byte, this is 2nd, this is 3rd, this is 4th, this is 5th, this is 6th, this is 7th. So, the first byte starting position is first byte is nothing but final SOC. So, whatever message you will receive in that message when you extract the data.

So, the first byte of data will be nothing but the final SOC. this particular data which is been there then the next message will be the minimum cell voltage which will have the starting position which is 2 and having the byte length 2 so that means 2 and 3 will actually corresponds to the minimum cell voltage then for the third signal or you can say the data which is The maximum cell voltage which starts from the byte number 4 and it has the length 2. So, these two things will be nothing but whatever the data which is been there in this byte 4 and 5 will corresponds to the maximum cell voltage value. then the byte number six which has nothing but the minimum battery temperature signal or data that means the sixth byte and having the length one so this one is nothing but your minimum battery temperature and the seventh byte is having the maximum battery temperature so this is the maximum battery temperature

So, whenever you receive the message, you extract the data and in the data, the different bytes indicate different signals or the data in that particular message. Similarly, in battery charge requirement message, the data length is 5 byte and the signals which are present in this particular message are nothing but voltage requirement signal, current requirement signal, charge mode signal and their length are defined and their starting positions are been defined. So this is how one can able to extract the data and from the data can able to extract the information corresponding to a specific signals and then can use the resolution and the scaling factor and the offset value to actually interpret it. the real information then let us take some other example of messages we have the charger max and mean output parameter message it has the message code which is the cml again it is the same thing which is defined in gbt 27930 the source address or you can say the source is the charger and the destination address is nothing but the bms so start from charger and goes to bms then it has the parameter group number which is nothing but zero zero eight zero zero hexadecimal case and we have the priority which is six the data length will be eight so if you see the first two bytes the first two bytes one two three four assume these are the bytes

The first 2 byte will correspond to the data which will provide the value corresponds to maximum output voltage. The second 2 byte because starting position is 2 is the minimum output voltage. The 5th and 6th byte starting from 5 it is the maximum output current. And then the 7th and 8th will be nothing but the minimum output current. That will be there.

so accordingly one can once they receive the message from the message they can extract the data from the data they can extract these different signals and they can actually interpret what is the actual data corresponding to that parameter then you can also see charger recognition message which is nothing but short form as CRM starts from the charger goes up to the BMS or you can say vehicle it has the parameter group number which is 000100 it has the priority 6 it has the data length which is 8 and it has a starting position having the three signals which is recognition result of one byte and starting position is one then the charger number which is the four byte starting from byte number two and then it has the charger and charge station location code which has the length three and start position is six so this is how they will define the different messages and those messages has to be included in CAN database file and those file has to be included in the EVSE controller on one side and the BMS controller on other side such that once the messages are received they can compared with those CAN database files and then can able to see whether it is a message which has to be consider or not and if it is there then it will be taken in the data will be extracted and from the data the different parameters corresponding to that particular message will be extracted. Further, we see that this particular message which is charge max mean output parameter has the cycle time of 250 millisecond or you can say parameter cycle of 250 millisecond.

While if we see the charger recognition parameter message will nothing but have the cycle parameter which is 250 millisecond. Similarly, the vehicle statics data is having the cycle time of 250 ms and the battery charge requirement will have the cycle time of just 50 ms. So, this is how the cycle time of these messages will be defined and those things will also be included in AIS 138 part 2 standard. Here it is nothing but charge optimization parameter message. So those things you can get it from AIS 138 part 2 and extra G. Now let us take one example of vehicle statics data message.

Now as we have discussed it has nothing but if we see our discussion it has nothing but 1, 2, 3, 4, 5, 5 signals in them. And those signals are defined as the final SOC, the minimum cell voltage, maximum cell voltage, minimum battery temperature and maximum battery temperature. And these fields are mandatory. This signal has to be there. Some signals you will see that from the standard, some signals are optional.

It can be included, cannot be included. But this particular, all the signals are mandatory signals which has to be there. As I have indicated in the standard, you will also see the data resolution. You will also see the offset. if the offset is not given that it indicates that the offset value is nothing but 0 0 in all the things in all the messages it is 0 0 that we can take now in this thing we have the resolution for example the first byte information

It is the first byte. I should not say bit. It is the first byte information. It is, you know, the first byte is having the resolution 1 percentage per bit. The second byte has 0.01 volt per bit.

The fourth byte is 0.01 volt per bit. The sixth one is 1 degree Celsius per bit. And the seventh one which is the maximum battery temperature signal having the resolution 1 degree Celsius per bit. and we know that this vehicle static data message has the PGN which is nothing but 001C00 obviously in hexadecimal format it is been there that is nothing but my PGN value for this so let us understand from an example how we can able to extract the real data from the CAN message which we received so assume you got the CAN message and you have extracted the message ID So, you have extracted the message ID or you can say 29-bit identifier.

And at the same time, you have also extracted the message from the CAN message. So, both things you have extracted. Now, message ID is, you know, I am writing in terms of hexadecimal case 0x181C56F4. And let's say we have the message, which is nothing but 466. 0 1 2 C 0 1 A 4 0 A 3 7 0 0 so again if you see this I am defining in terms of hexadecimal format so this is first byte this is second this is third this is fourth this is fifth sixth seventh eighth so this is first second third

fourth, fifth, sixth, seventh and eighth. So, this is the data which we have already extracted and we have this data and these are nothing but the raw data. and these are nothing but the byte order which i have indicated so these things these are expressed in the hexadecimal format so that's why the byte order representing binary it will be defined by four bits so that's why the two hexadecimal numbers will actually constitute the byte now in this case we have received this message so in this case we know that from our discussions of how the data frames are we represented that means protocol data unit format we know that we have first 3 byte which is we have to neglect then the 3 will be the priority field then we have the reserve byte then we have data bit then we have pgn for 8 bits and then we have destination address and then the source address so let us try to define this one so what it indicates if you try to write this one in binary format is nothing but 0001 1 0 0 0 then we have 0 0 0 1 the C will be represented as 1 1 0 0 so this is 1 this is 8 this is 1 this is C then we have 5 which will be represented in the form of

0 1 0 1 5 0 1 1 0 6 we have 1 1 1 1 f and we have 0 1 0 0 which is 4. so from our understanding of protocol data unit of pdu format we understood that these first three bits are to be neglected or avoided then these three bits which is the priority bits and here the priorities are given as 6 and that you can also see from the vehicle statistic data the priority is given as 6 now then we have next bit which is the reserve bit and then we have the data page bit Now after the first 8 bits what we have is the next 8 bits is nothing but my PGN number. Now since we have seen that our PGN is nothing but 001C00. So what you will see is that this is our PGN which is 1C and it has to be appended with 00 at the front and 00 at the back.

So the moment this PGN is being instructed this indicates that it is nothing but your vehicle statistic data. Then we have after this, this 2 byte is nothing but is your destination address and this is if you recall the 5 6 is given as the address for the charger controller or the charger which is been there and after that this 2 will be the source address which is defined for the BMS so obviously this message is sent from the BMS to the charger once the charging session ended so that is the status of the vehicle or the statistic of the vehicle which is to be sent to the EVSE and that will be stored for metering and for future logging so this is how once you obtain the message you extract the message id and from the message id the things related to priority, the things related to PGR number because that will indicate what will be the message all about and what will be the destination, what will be the source. And accordingly, one can able to identify what is this message all about, whether this message is correct message or not.

And if it is not, then they will discard it. If it is, then they will take the message and then extract the message data. Now, this is vehicle statistical data message. now we have recognized that this is a vehicle statistic data now let us understand how the message data is being interpreted so now we know that in our message five signals and the first signal is nothing but the final SOC which starts from the byte number one and having the length one byte so our first thing is nothing but our the final SOC is nothing but your first byte which is 46 obviously in hexadecimal format it is been shown then the second signal is minimum cell voltage so minimum cell voltage which is starting from the byte 2 having the length 2 so byte 2 having the length 2 so it is nothing but 0 1 2 C in hexadecimal format

then we have the message which is the maximum cell voltage which if you see which starts from the 4th byte and having the length which is 2 so if we see 4th byte and having the length 2 which is nothing but 01A4 in hexadecimal format and then we have the minimum battery temperature minimum battery temperature which is again just the 1 byte and starting byte is 6th byte 6th byte is nothing but 0A and then finally what we have is we have the maximum battery temperature again it is a single byte and having nothing but 37H now again this data we have assumed that all those scenarios related to Little Indian and Big Indian has been considered and this is the data which we have extracted from the message and it can be directly used and to actually extract the real information. So that thing is being taken care I mean those things we have already assumed that whether it is been arranged in Little Indian or Big Indian depending upon the standard you can see and we have only assumed here that we have obtained the data from the message. Now this particular thing that indicates the final SOC is 46 hexadecimal that indicates it is nothing but 70 in terms of decimal.

Minimum cell voltage is 012C in hexadecimal case. Now when we do the decimal it is nothing but 300. this is 0 1 a 4 hexadecimal means nothing but 420 in decimal and it is nothing but 10 in decimal and this corresponds to 55 in decimal form now we know that when we have to extract the information we have to use the data resolution to be nothing but one percentage per bit and the offset value is zero so what it indicates that my real data or the actual data is nothing but whatever the raw data which I am getting so these are all the raw data whatever the raw data which I am getting multiplied by the resolution which is 1% per bit plus 0 because my offset value is 0 so this is my resolution this is my offset and this will indicate that the final SOC value is nothing but 70% similarly for the minimum cell voltage we know that our data resolution is 0.1 volt per bit and offset is 0 so this you will get 300 into 0.01 volt per bit

plus 0 and that indicates nothing but equal to 3 volt similarly the maximum cell voltage is nothing but 0.01 volt 0.01 volt per bit resolution and value is 420 plus 0 which is 4.2 volt which we get then we have the 10 so 10 and the resolution value is 1 degree per celsius per bit and offset is 0 which indicate is nothing but 10 degree celsius and the maximum temperature is nothing but 37 into again 1 degree celsius per bit resolution offset is 0 which indicate is 37 degree celsius so this is how once you receive the message you extract the message id from the message id you recognize which is the actual message and what is the destination address source address if those things are correct then you see the data you extract the raw data from that data of the can message and once you achieve the raw data you multiply it with the resolution and then sum it up those things with the offset and to get the actual data which is been there and then this data can be used to log the information or can be used to show it to the user who is actually want to see the status of the vehicle which was being charged so this is how one can obtain that message and from the message one can extract the actual information So, this is one example we are showing.

Similarly, there are other messages which have been there, other resolutions for corresponding to those messages which have been there, the signal corresponding to those messages which have been there and the offset corresponds to those messages which have been there which can be used to extract the different information in the real time. so this is how the data has been exchanged we have in this particular lecture we have understood how the messages are been defined how the message ids are been defined once the message ids are defined how from the message id the different parameters can be extracted because in the message we have several sets of signals and For each message, we have to see their description of the CAN messages and from the CAN messages, we extract the signals, values, raw data and from the raw data using the resolution and offset defined in the database file, we will represent the actual data and that actual data will then be used by the individual controllers to perform required operation. So, this is how the communication is being taken place when the DC0 charger is getting plugged into the EV. Thank you very much for listening to this lecture.

We will see you in the next lecture with a new portion, which corresponds to the charging stations and their components. Thank you.