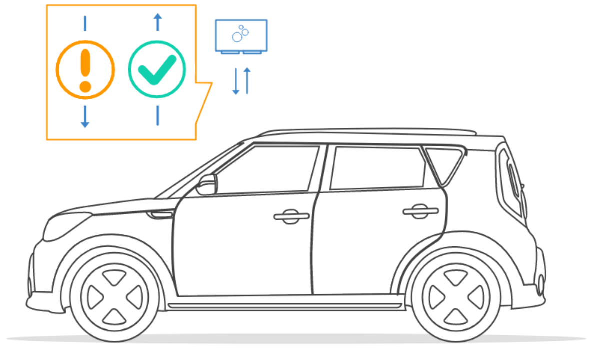
UDS
(Unified Diagnostic Services)

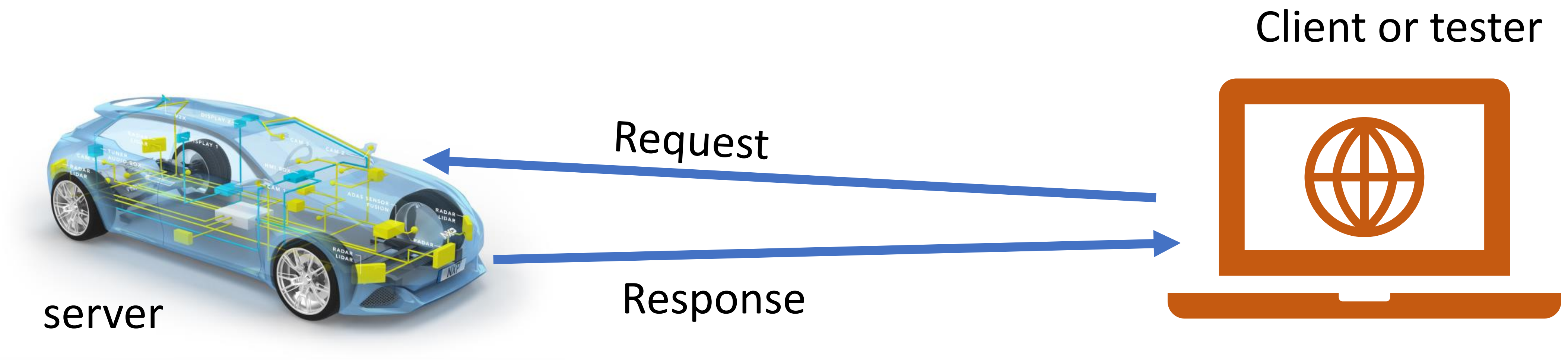
What is UDS?

- UDS stands for Unified Diagnostics Service, It is just a Transport Protocol
- Is a diagnostic communication protocol used in electronic control units (ECUs) within automotive electronics
- which is specified in the ISO 14229
- It is present in all modern cars
- Provides access to the Services offered by ECUs
- Allows to perform transmissions of up to 256 bytes
- Diagnostic tools are able to contact all ECUs installed in a vehicle, which has UDS services enabled.
- In contrast to the CAN bus protocol, which only uses the first and second layers of the OSI model, UDS utilizes the fifth and seventh layers of the OSI model.
- This makes it possible to interrogate the fault memory of the individual control units, to update them with new firmware, have low-level interaction with their hardware
- In practice, UDS communication is performed in a client-server relationship - with the client being a tester-tool and the server being a vehicle ECU



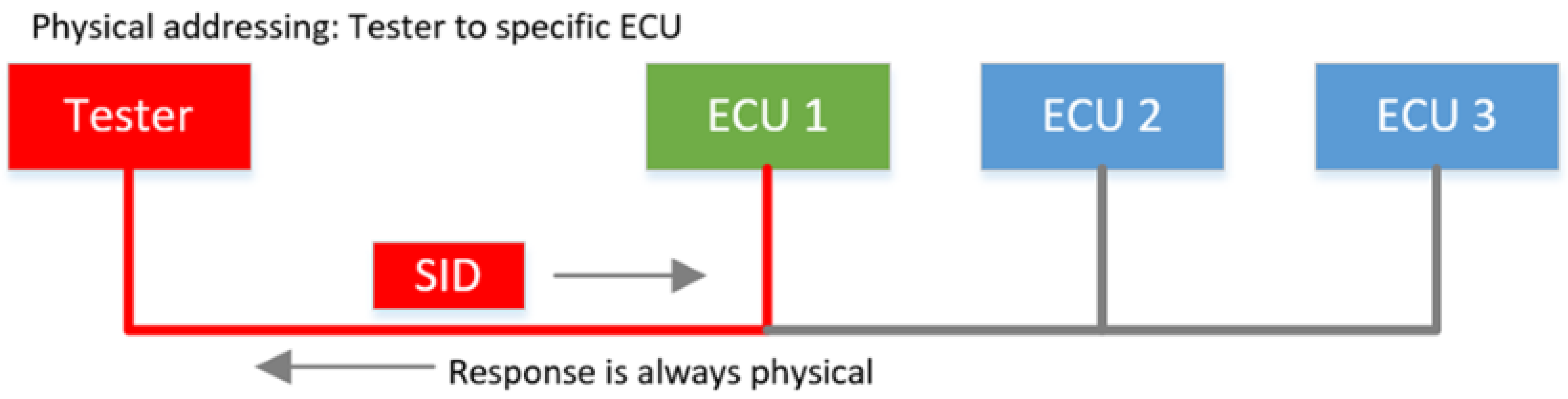
Use case of vehicle diagnostics tools using UDS:

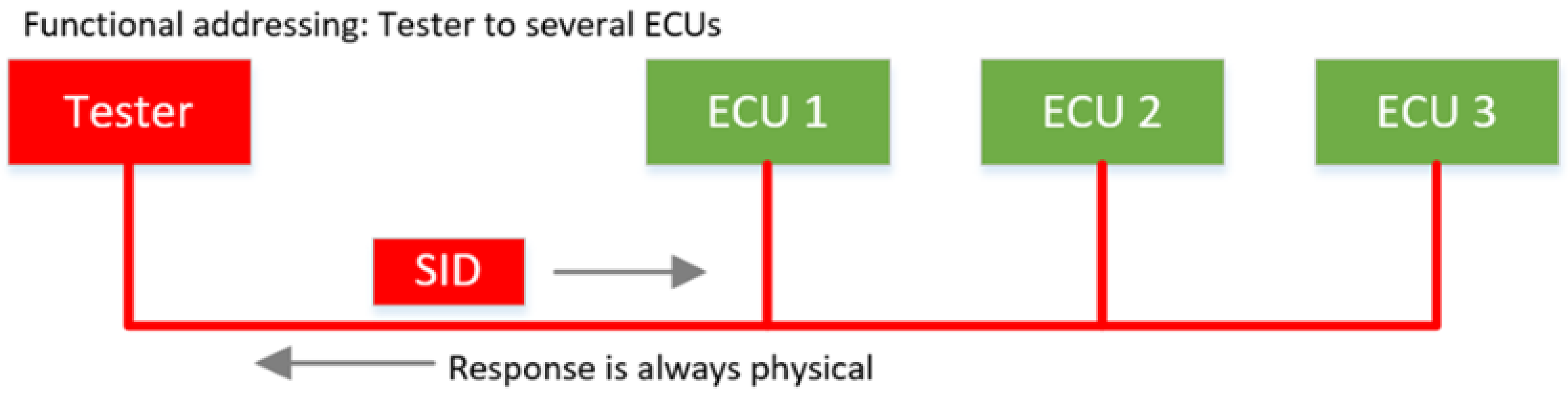
- Read/clear diagnostic trouble codes (DTC) for troubleshooting vehicle issues
- Extract parameter data
  - Temperatures
  - VIN(Vehicle Identification Number) etc…
- Initiate diagnostic sessions to e.g. test safety-critical features
- Modify ECU behavior
  - Resets (power OFF-ON)
  - Firmware flashing and settings modification etc…
- Reading and writing into memory of ECU



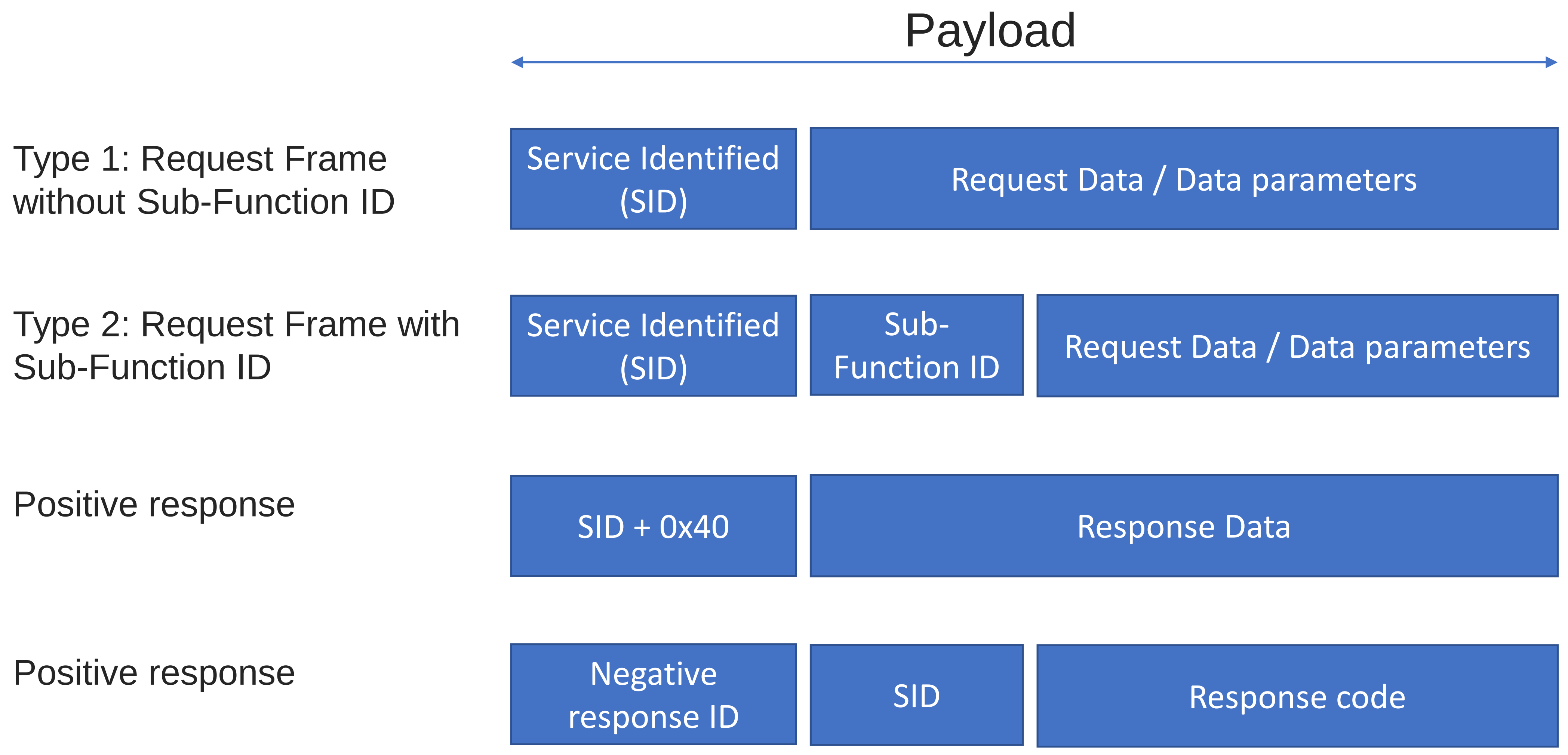
Physical and Functional request:

- Tester sends a request either to one (Physical addressing) or more(Functional addressing) ECU's.
- Then ECU unit or units sends positive or negative response

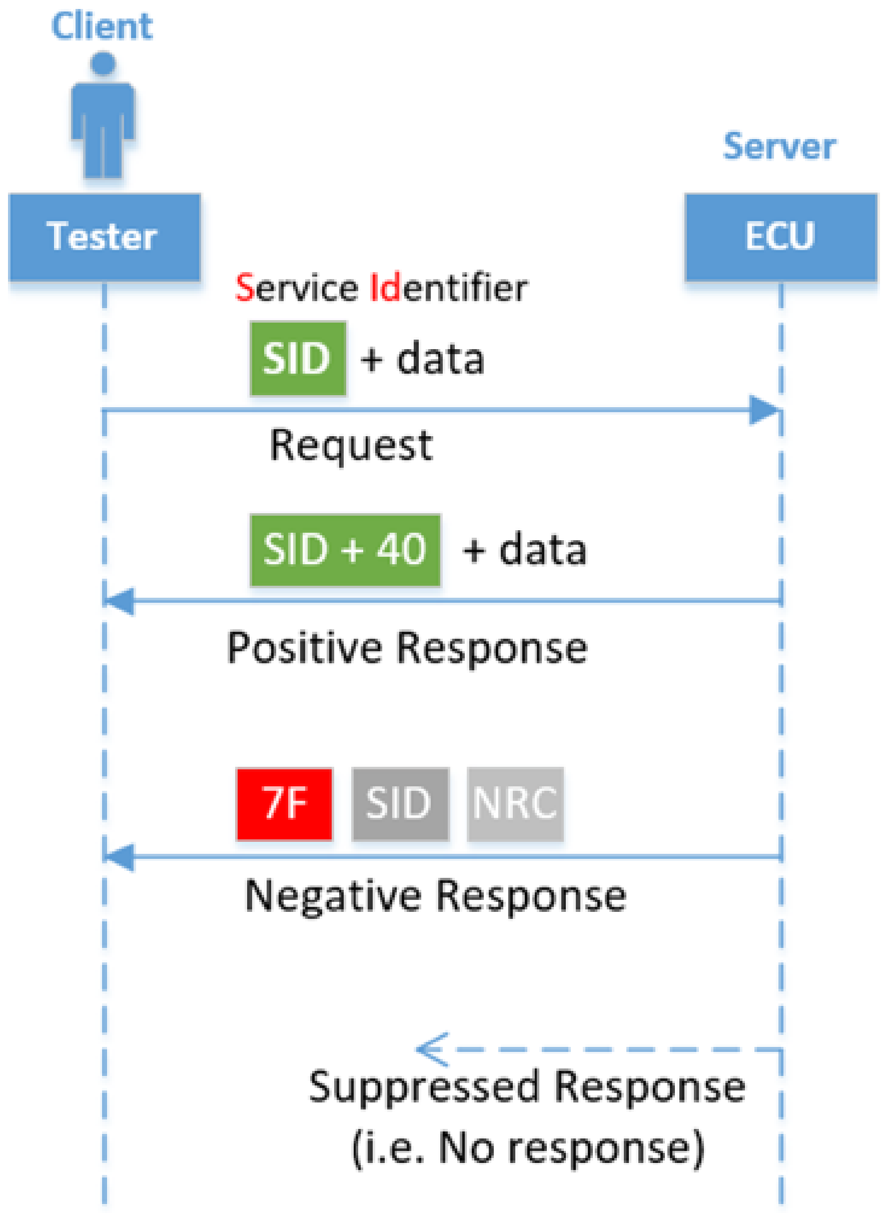

Physical addressing: Tester to specific ECU
Tester
ECU 1
ECU 2
ECU 3
SID
Response is always physical


Functional addressing: Tester to several ECUs
Tester
ECU 1
ECU 2
ECU 3
SID
Response is always physical

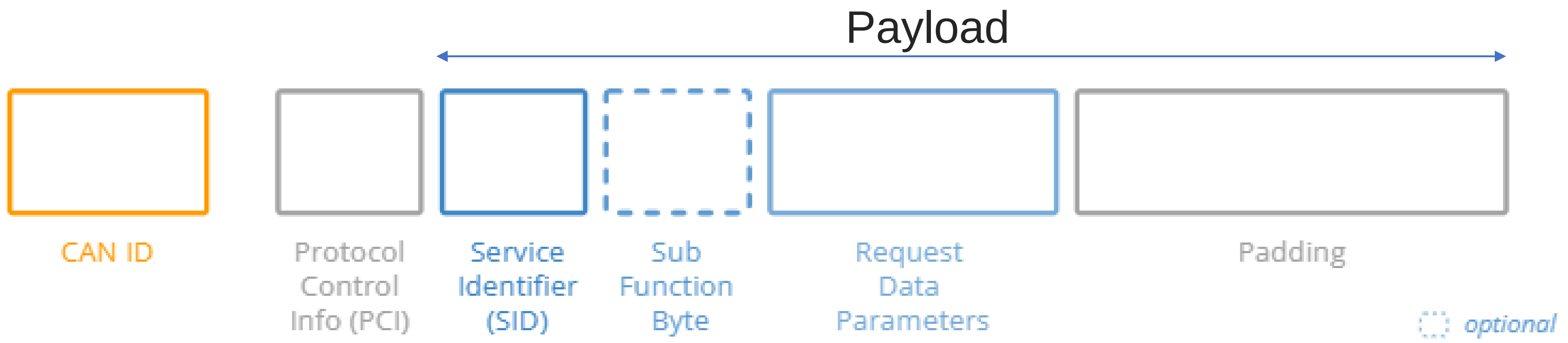
UDS request and response message format (General):



UDS request and response message format (General):



UDS request and response message format (complete overview):



ISO 15765-2 or ISO-TP (Transport Layer), is an international standard for sending data packets over a CAN-Bus. The protocol allows for the transport of messages that exceed the eight byte maximum payload of CAN frames. ISO-TP segments longer messages into multiple frames, adding metadata that allows the interpretation of individual frames and reassembly into a complete message packet by the recipient. It can carry up to 4095 bytes of payload per message packet.

ISO-TP prepends one or more metadata bytes to the payload data in the eight byte CAN frame, reducing the payload to seven or fewer bytes per frame. The metadata is called the Protocol Control Information, or PCI. The PCI is one, two or three bytes. The initial field is four bits indicating the frame type, and implicitly describing the PCI length.

UDS request and response message format (complete overview) continue …

List of protocol control information (PCI) field types

Type	Code	Description
Single frame (SF)	0	The single frame transferred contains the complete payload of up to 7 bytes (normal addressing) or 6 bytes (extended addressing)
First frame (FF)	1	The first frame of a longer multi-frame message packet, used when more than 6/7 bytes of data segmented must be communicated. The first frame contains the length of the full packet, along with the initial data.
Consecutive frame (CF)	2	A frame containing subsequent data for a multi-frame packet
Flow control frame (FC)	3	the response from the receiver, acknowledging a First-frame segment. It lays down the parameters for the transmission of further consecutive frames.
	4..15	Reserved

UDS request and response message format (complete overview) continue …

CAN-TP Header

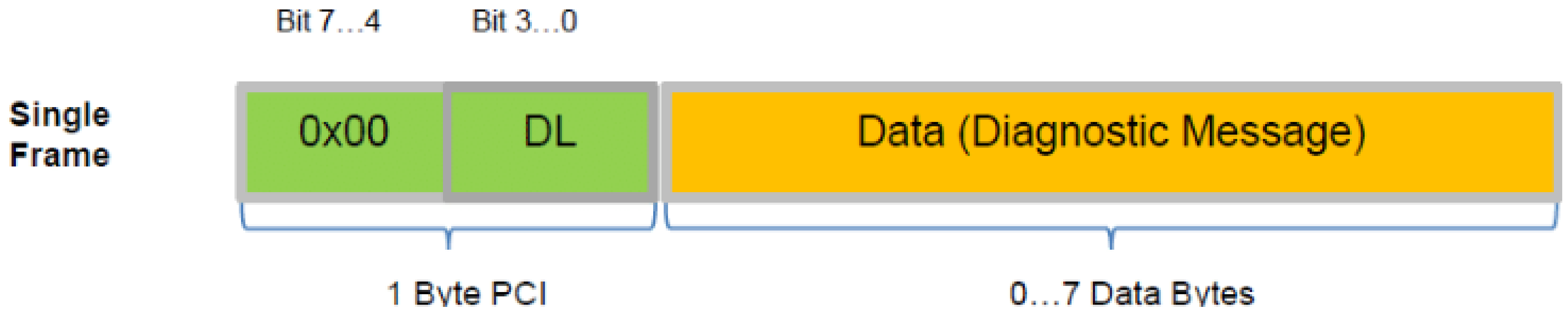
Bit offset	7 .. 4 (byte 0)	3 .. 0 (byte 0)	15 .. 8 (byte 1)	23..16 (byte 2)	....
Single	0	size (0..7)		Data A	Data B
First	1	size (8..4095)	Data A	Data B	
Consecutive	2	index (0..15)	Data A	Data B	Data C
Flow	3	FC flag (0,1,2)	Block size	ST	

A message of seven bytes or less is sent in a single frame, with the initial byte containing the type (0) and payload length (1-7 bytes). A message longer than 7 bytes requires segmenting the message packet over multiple frames. A segmented transfer starts with a First Frame. The PCI is two bytes in this case, with the first 4 bit field the type (type 1) and the following 12 bits the message length (excluding the type and length bytes). The recipient confirms the transfer with a flow control frame. The flow control frame has three PCI bytes specifying the interval between subsequent frames and how many consecutive frames may be sent (*Block Size*).

UDS request and response message format (complete overview) continue …

SINGLE FRAME (SF)

If the data payload is 7-bytes or less than that, then the single frame will be used for the data transfer in CAN-TP Protocol. Where the first byte of the data field is used for addressing. Again this byte is divided into two divisions, where the MSB 4-bit is used for addressing of TP frame type called PCI (protocol information). The LSB 4-bit is used for the DLC (Data Length Code).

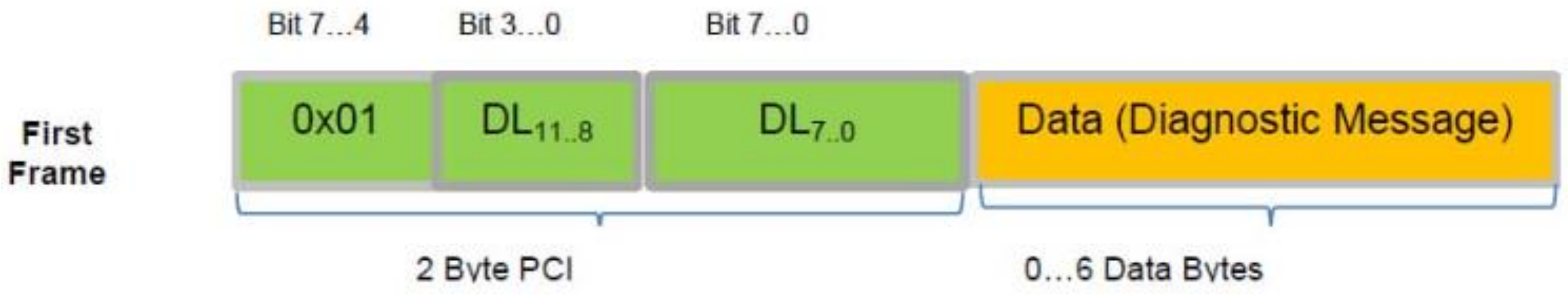


CAN-TP: Single Frame Format

UDS request and response message format (complete overview) continue …

FIRST FRAME (FF)

The first frame is an initial message of the multi-frame message packet in CAN-TP Protocol. It is used when more than 6 or 7 bytes of data segmented must be communicated. The first frame contains the length of the full packet, along with the initial data. In FF, the first 2-bytes are used for the PCI, where the MSB 4-bit of the 1st byte is used for the type of frame and the LSB 4-bit and next 1-byte (total 8+4 = 12 bit)of the CAN data field is used for DLC (2^12 = 4095 data bytes). so in FF, only 6-bytes of the data can be transferred for the first time. This frame is responsible for sending the information to the receiver about the information as to how many total data bytes he is going to send.

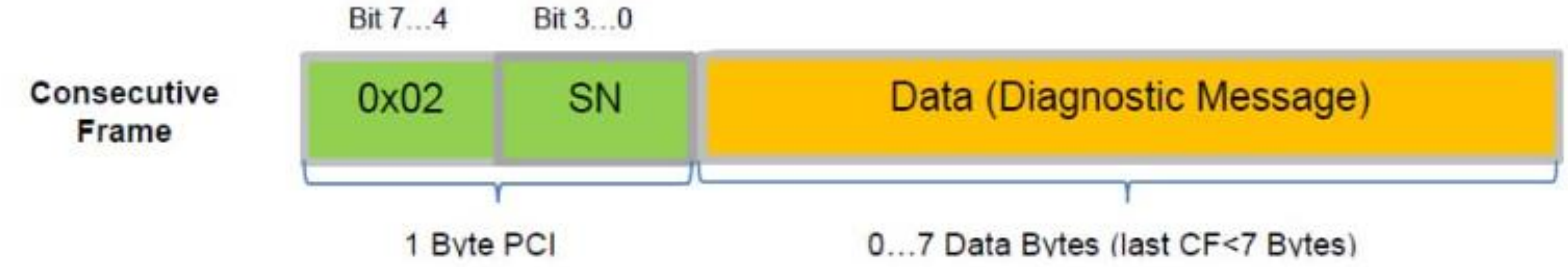


CAN-TP: First Frame Format

UDS request and response message format (complete overview) continue …

CONSECUTIVE FRAME (CF)

The primary task of the transport protocol or you can also CAN-TP Protocol is to transfer messages which cannot be transmitted as a single Protocol Data Unit (PDU) due to their length. Messages which contain more data that can be transmitted within a single PDU are segmented by means of the transport protocol and divided into multiple, separate PDUs. This procedure can also be implemented on the data link layer. The segmentation of the message must then be carried out in PDUs of the corresponding data link protocol.
In CF, the 1st byte is used for PCI byte, whereas the MSB 4-bit is defined for the type of frame and the LSB 4-bit is defined for the Sequence number. The sequence number always starts at 1 and increments with each frame sent (like as 1, 2,…, 15, 0, 1,…), with which lost or discarded frames can be detected. Each consecutive frame starts at 0, initially for the first set of the data in the first frame will be considered as 0th data. So the first set of CF(Consecutive frames) starts from "1". There afterwards when it reaches "15", it will be started from "0" by resetting the buffer register.

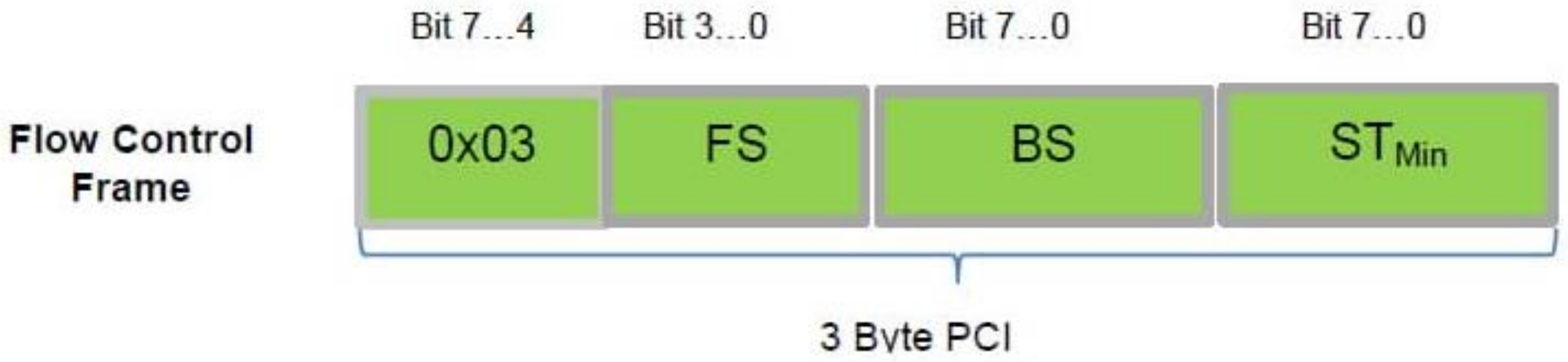


CAN-TP: Consecutive Frame Format

UDS request and response message format (complete overview) continue …

FLOW CONTROL FRAME (FC)

The flow control mechanism of the CAN-TP Protocol is used to configure the sender to match the properties of the receiver (timing, available receive buffer, readiness to receive). The FCF is always sent by the receiver so that to inform about the receiver capability of how the transmitter is going to send the data.



CAN-TP: Flow Control Frame Format

UDS request and response message format (complete overview) continue …

Flow Control

Bit offset	7 .. 4 (byte 0)	3 .. 0 (byte 0)	15 .. 8 (byte 1)	23..16 (byte2)
Description	type	if the transfer is allowed	Block Size	Separation Time (ST), minimum delay time between frames (end of one frame and the beginning of the other)
Single	type = 3	(0 = Continue To Send, 1 = Wait, 2 = Overflow/abort)	0 = remaining "frames" to be sent without flow control or delay	<= 127, separation time in milliseconds.
Single	type = 3	(0 = Continue To Send, 1 = Wait, 2 = Overflow/abort)	> 0 send number of "frames" before waiting for the next flow control frame	0xF1 to 0xF9 UF, 100 to 900 microseconds.

The initial byte contains the type (type = 3) in the first four bits, and a flag in the next four bits indicating if the transfer is allowed (0 = Clear To Send, 1 = Wait, 2 = Overflow/abort). The next byte is the block size, the count of frames that may be sent before waiting for the next flow control frame. A value of zero allows the remaining frames to be sent without flow control or delay.

UDS request and response message format (complete overview) continue …

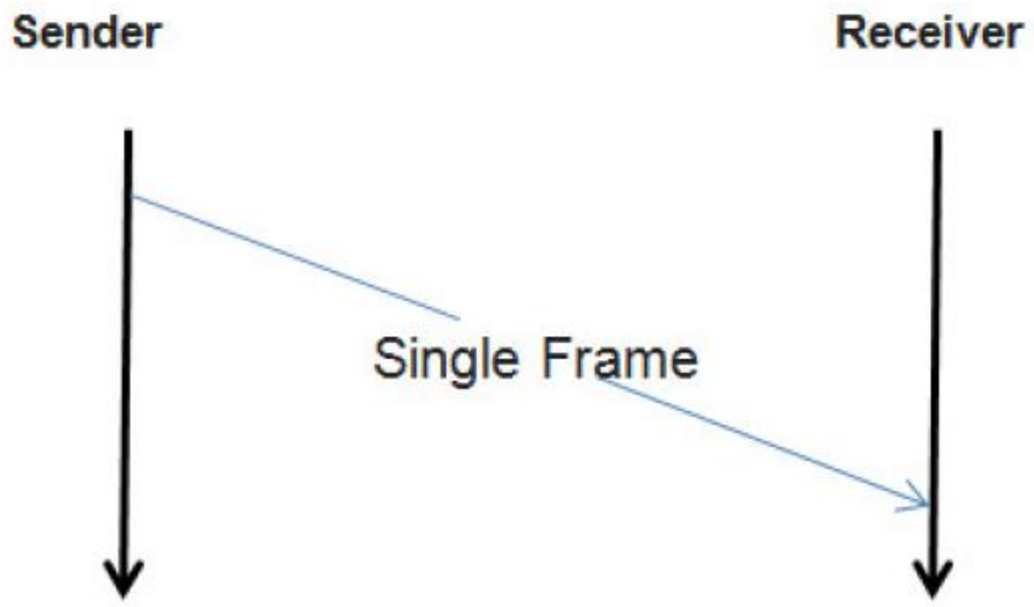
Flow Control continue…

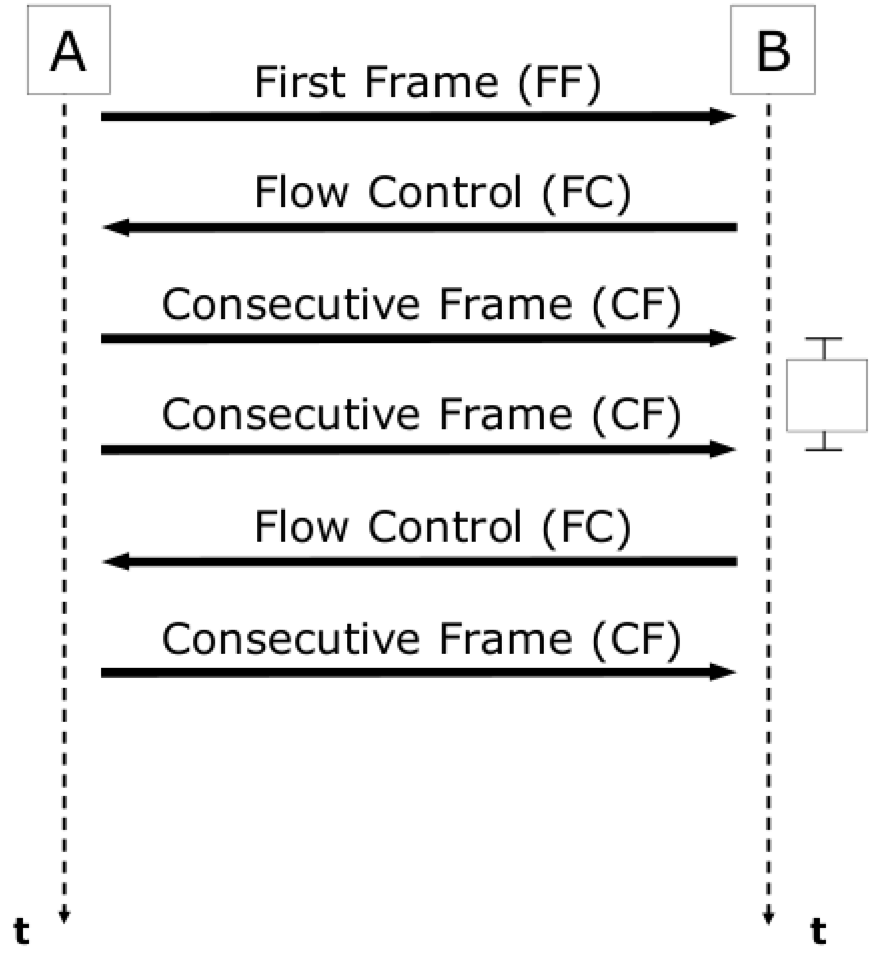
The third byte is the Separation Time (ST), the minimum delay time between frames. ST values up to 127 (0x7F) specify the minimum number of milliseconds to delay between frames, while values in the range 241 (0xF1) to 249 (0xF9) specify delays increasing from 100 to 900 microseconds. Note that the Separation Time is defined as the minimum time between the end of one frame to the beginning of the next. Robust implementations should be prepared to accept frames from a sender that misinterprets this as the frame repetition rate i.e. from start-of-frame to start-of-frame. Even careful implementations may fail to account for the minor effect of bit-stuffing in the physical layer.

The sender transmits the rest of the message using Consecutive Frames. Each Consecutive Frame has a one byte PCI, with a four bit type (type = 2) followed by a 4-bit sequence number. The sequence number starts at 1 and increments with each frame sent (1, 2,..., 15, 0, 1,...), with which lost or discarded frames can be detected. Each consecutive frame starts at 0, initially for the first set of data in the first frame will be considered as 0th data. So the first set of CF(Consecutive frames) start from "1". There afterwards when it reaches "15", will be started from "0". The 12 bit length field (in the FF) allows up to 4095 bytes of user data in a segmented message, but in practice the typical application-specific limit is considerably lower because of receive buffer or hardware limitations.

UDS request and response message format (complete overview) continue …

Flow Control continue…





UDS request and response message format (complete overview) continue …

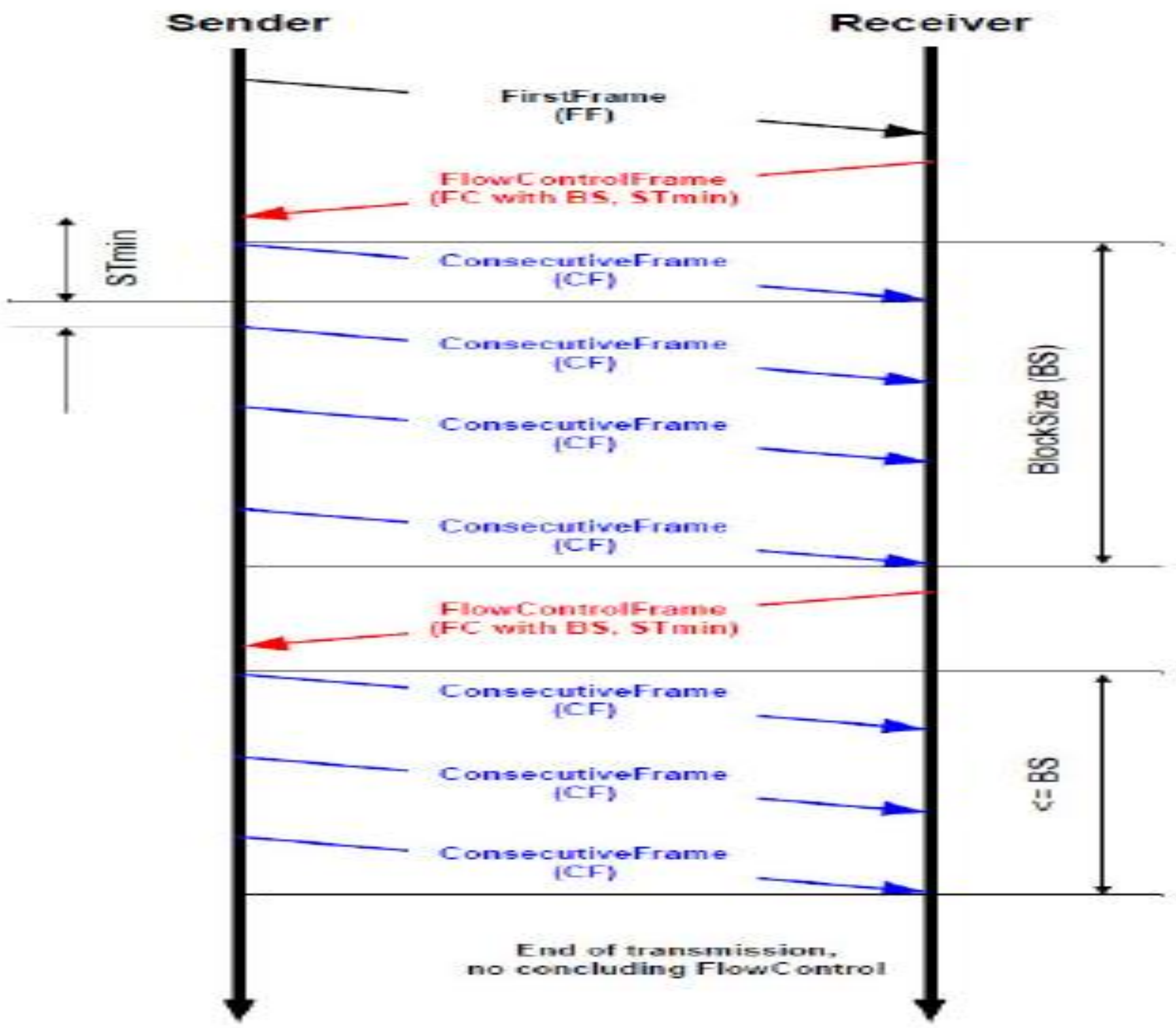
Flow Control continue…
timing Requirements of
flow control

- STmin: shows the minimum separation time between consecutive frames to be noticed.
- Tester:

P2Client: Timeout value for the client to wait after the successful transmission of a request message till the start of incoming response message.

P2Client_max: is the maximum value of P2Client

P2*Client_max: Enhanced timeout for the client to wait after the reception of a negative response message with negative response code 0x78 for the start of incoming response messages.



UDS request and response message format (complete overview) continue …

Flow Control continue…

timing Requirements of flow control

Message timing

- ECU:

P2Server: performance timer of ECU, and is either loaded with P2Server_max or P2*Server_max value.

P2Server_max: performance value loaded when server receive a request. Server either process the request and send back a response in time or the request processing is still going and the timeout(P2Server_max value) occur then server send back a negative with NRC=0x78 for "requestCorrectlyReceived-ResponsePending" to notify about pending final response.

P2*Server_max: performance requirement for the server to start after the transmission of a negative response message with negative response code 0x78. In case the server can still not provide the requested information within the enhanced P2*Server_max, then a further(number of time depend on configuration) negative response message including negative response code 0x78 can be sent by the server.

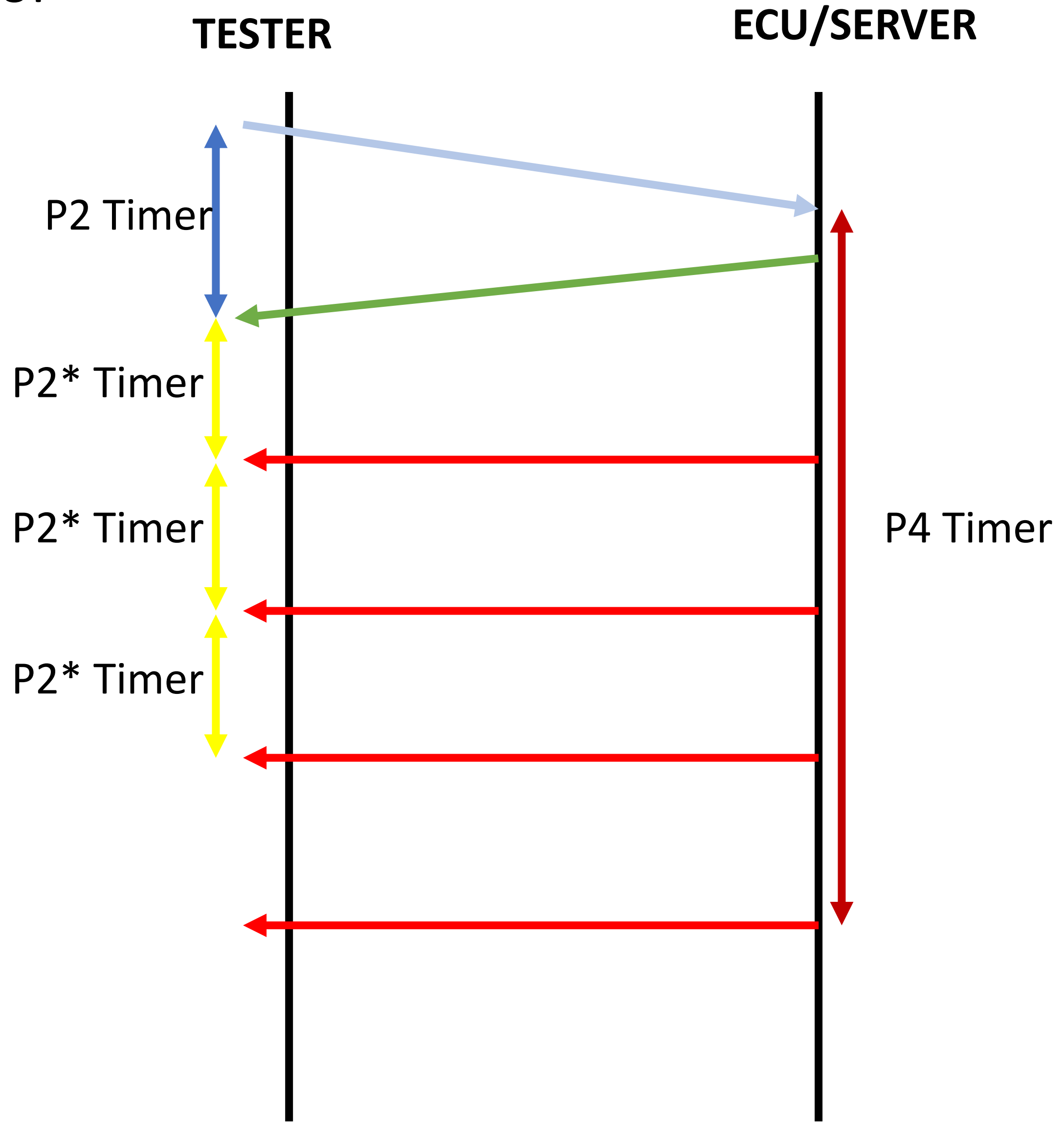
- P4Server: is performance requirement time, which is period between the reception of a request and the start of transmission of the final response(which can be either a positive response or negative response with NRC is not 0x78).

P4Server_max: is the maximum value of P4Server

UDS request and response message format (complete overview) continue …

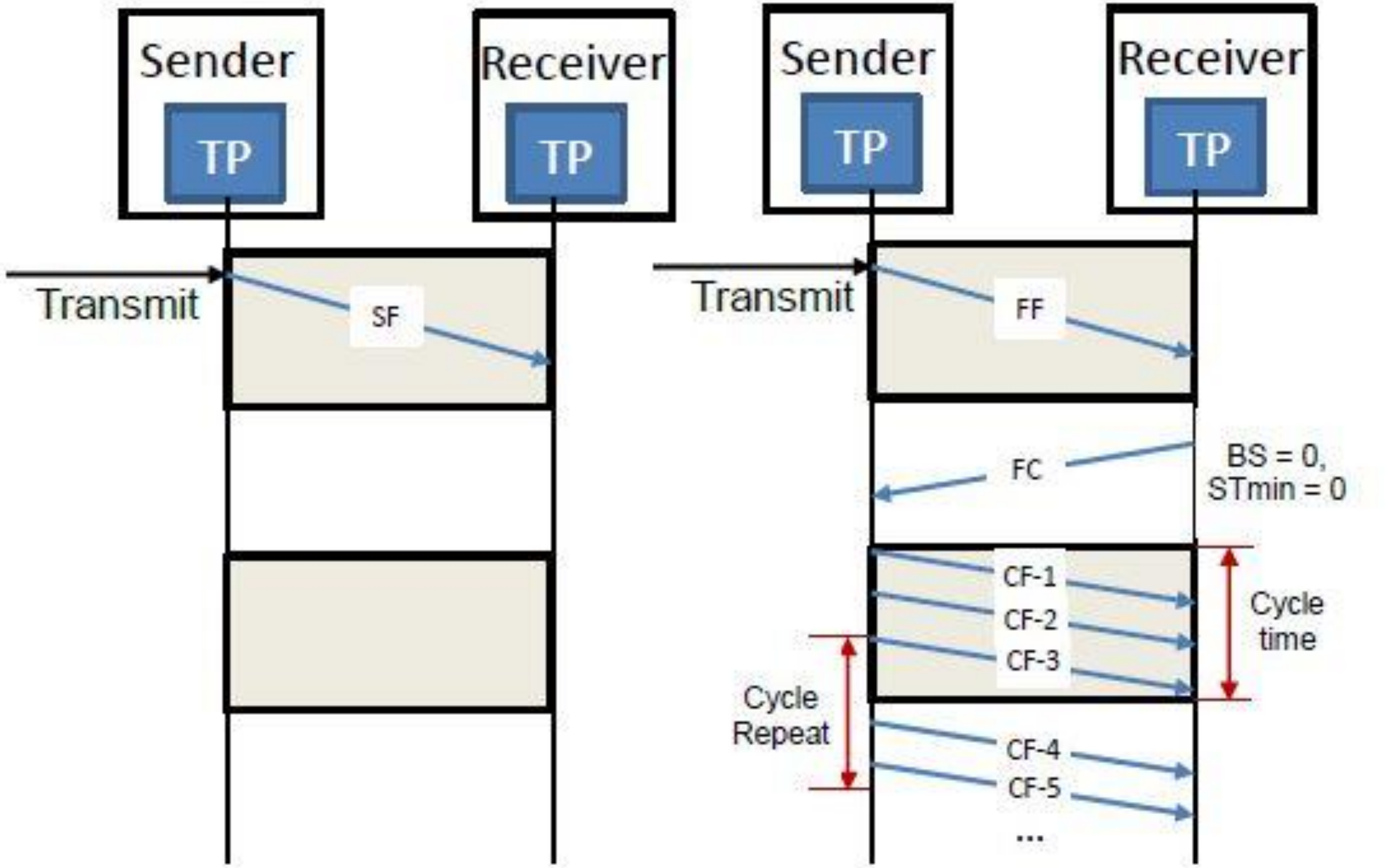
Flow Control continue…

timing Requirements of flow control



UDS request and response message format (complete overview) continue …

Flow Control continue…



UDS request and response message format (complete overview) continue …

Flow Control continue…

EXAMPLE:

FF: 10 64 xx xx xx xx xx xx

In the above, the client is sending a CAN data frame where the first byte is defining as 0x10. The MSB 4-bit is 1 which defines it is the first frame and the LSB 4-bit plus next 1-byte is having a total 12-bit which defines the CAN TP data length. This value is 0x64 and in decimal, it is 100 bytes. So from there, the server understood that the client is going to send a total of 100 bytes of data. Next 6-bytes are xx means any data the client can send. So from here, we understood that the first 2-bytes of data are Protocol control information which is not the real data. The next 6-bytes are real data that the server needs it. Then after this data received by the server, it will calculate its own receiver buffer memory defined in its programs and as per this, it will make a decision that how and when the next 94 bytes (100 – 6) of data client is going to send. Then the server will send the flow control frame like below as:

FC: 30 0A 14 xx xx xx xx xx

In the above frame, 4-MSB of the first byte is 3 means flow control frame and the next 4-LSB is flow status having 0 means clear to send. The clear to send means the server informs the client that the server is now free so that the client can start sending the periodic messages in UDS protocol frame format. The second byte is 0A is the block size nothing but the number of consecutive frames the client can send periodically. then third byte is 0x14 is the minimum separation time nothing but the time delay in between the two consecutive frames to give some time to server to be ready to receive the next consecutive frame periodically seamlessly.

Next page continue …

UDS request and response message format (complete overview) continue …

Flow Control continue…

EXAMPLE continue… :

After the client received the flow control frame it will start sending the consecutive frames periodically as below:

```
20 xx xx xx xx xx xx xx
21 xx xx xx xx xx xx xx
22 xx xx xx xx xx xx xx
23 xx xx xx xx xx xx xx
24 xx xx xx xx xx xx xx
25 xx xx xx xx xx xx xx
26 xx xx xx xx xx xx xx
27 xx xx xx xx xx xx xx
28 xx xx xx xx xx xx xx
29 xx xx xx xx xx xx xx
```

Next page continue …

In above the 10 consecutive frames sent to the server where the first 4-MSB defines the consecutive type and the second 4-LSB defines the index or the number of consecutive frames sending to the server. Since the server sent to the client for 10 periodic messages should be sent by the client periodically, the client sent it and after then it will wait for the next flow control frame from the server since the total 100 bytes need to be sent to the server. In the first frame, the client 6-bytes of data and in the next 10 consecutive frames having 7-bytes in each frame is 70 (10×7) plus the 6 bytes having a total 76 bytes. The remaining 24 (100-76) bytes will continue in the next phase. The total bytes of data transfer in a segment format is called periodic messages in UDS protocol

UDS request and response message format (complete overview) continue …

Flow Control continue…

EXAMPLE continue… :

After the total 10 periodic messages received by the server, it will send the next flow control frame as below::

30 00 14 xx xx xx xx xx

Then the client will receive this frame and after receiving it will understand that all the other bytes are already defined and in second byte it is 00 means now the client can send unlimited consecutive frames to server. So after receiving this frame the client again will start to send the consecutive frames like as below:

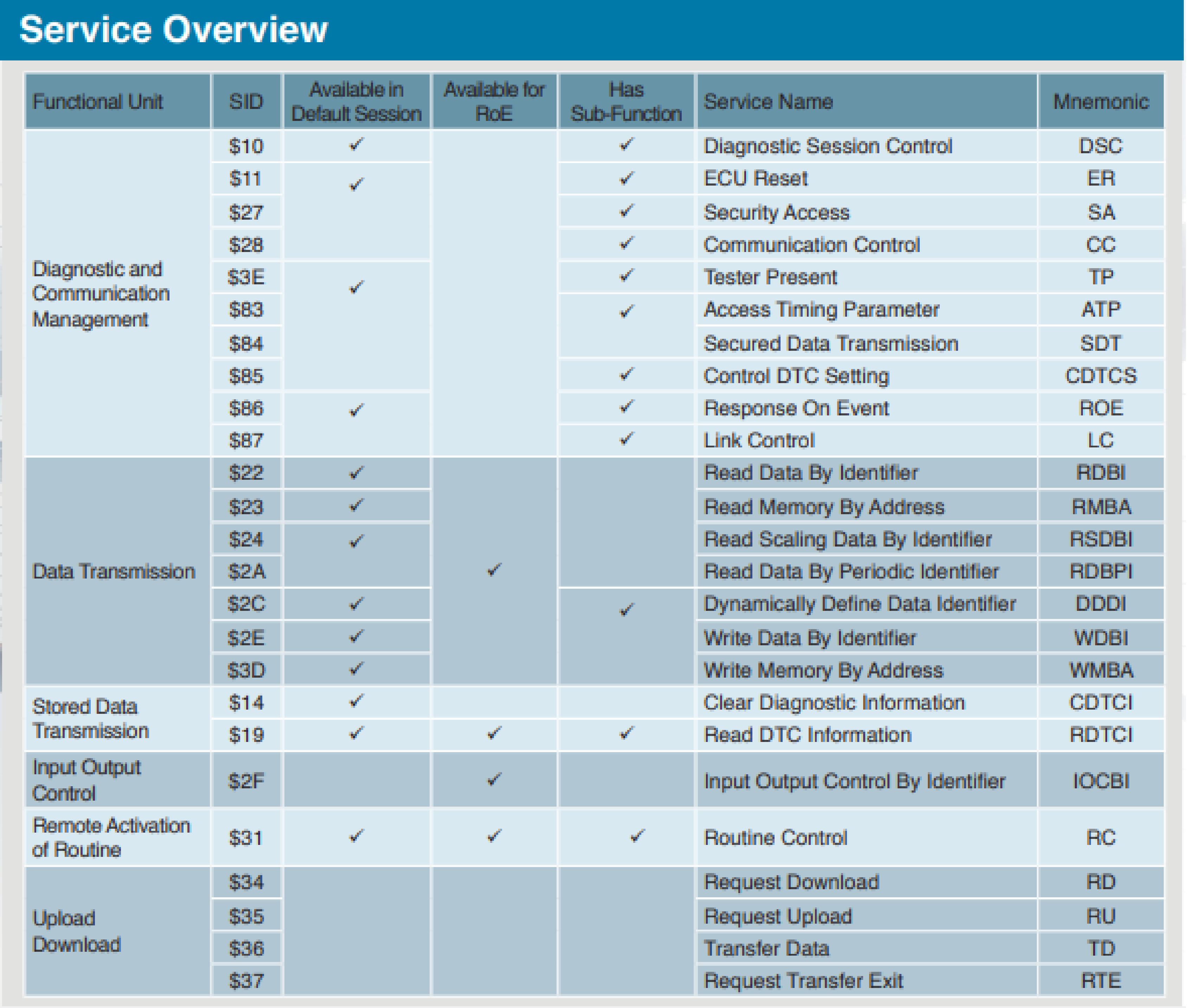
20 xx xx xx xx xx xx xx

21 xx xx xx xx xx xx xx

22 xx xx xx xx xx xx xx

23 xx xx xx 00 00 00 00

*I*n above after sending of 24 bytes, the client stops the periodic message send. After then the client-server communication will end. Like this, the UDS protocol uses the periodical message send to the server for multi-byte message communication.



Service Overview

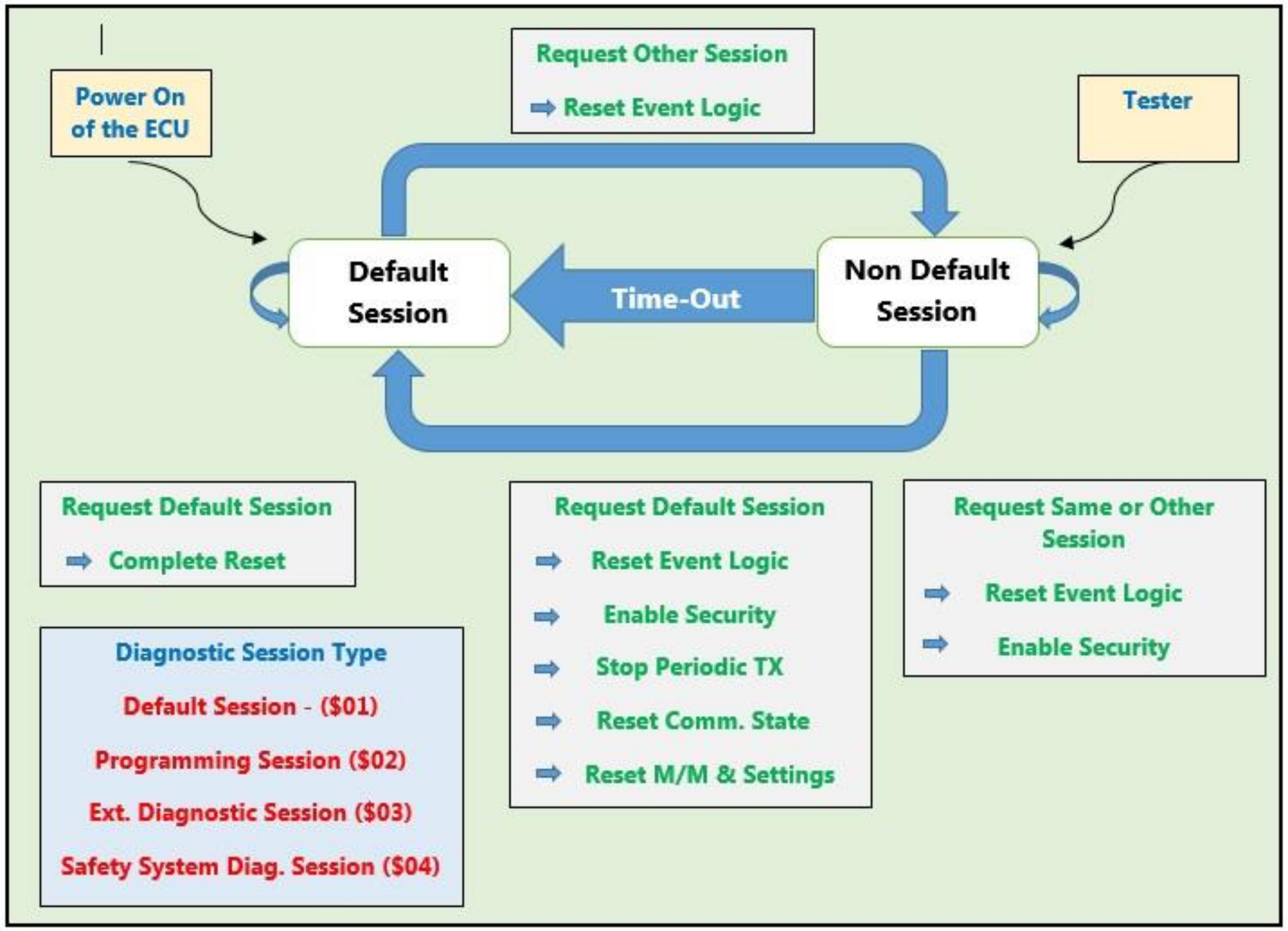
Functional Unit	SID	Available in Default Session	Available for RoE	Has Sub-Function	Service Name	Mnemonic
Diagnostic and Communication Management	$10	✓		✓	Diagnostic Session Control	DSC
	$11	✓		✓	ECU Reset	ER
	$27			✓	Security Access	SA
	$28			✓	Communication Control	CC
	$3E	✓		✓	Tester Present	TP
	$83			✓	Access Timing Parameter	ATP
	$84				Secured Data Transmission	SDT
	$85			✓	Control DTC Setting	CDTCS
	$86	✓		✓	Response On Event	ROE
	$87			✓	Link Control	LC
Data Transmission	$22	✓			Read Data By Identifier	RDBI
	$23	✓			Read Memory By Address	RMBA
	$24	✓			Read Scaling Data By Identifier	RSDBI
	$2A		✓		Read Data By Periodic Identifier	RDBPI
	$2C	✓		✓	Dynamically Define Data Identifier	DDDI
	$2E	✓			Write Data By Identifier	WDBI
	$3D	✓			Write Memory By Address	WMBA
Stored Data Transmission	$14	✓			Clear Diagnostic Information	CDTCI
	$19	✓	✓	✓	Read DTC Information	RDTCI
Input Output Control	$2F		✓		Input Output Control By Identifier	IOCBI
Remote Activation of Routine	$31	✓	✓	✓	Routine Control	RC
Upload Download	$34				Request Download	RD
	$35				Request Upload	RU
	$36				Transfer Data	TD
	$37				Request Transfer Exit	RTE

UDS services overview

UDS: Service 0x10 (Diagnostics session control)

Diagnostic session makes sure the communication between the ECU and the diagnostic tool. Different types of diagnostics sessions:

Default Diagnostic Session,
Programming Session
Extended Diagnostic Session.
Safety System Diagnostic Session.
After power on, ECU will be switched to a Default Diagnostic Session. After receiving different request from Diagnostic Tool, the ECU will be switched to different diagnostic session.



UDS: Service 0x11 (ECU Reset)

ECU Reset Service Identifier (0x11) is a service Identifier in UDS protocol. The sub-function parameter reset Type is used by the ECU Reset request message to describe how the server has to perform the reset and all the different ECU reset types are defined in sub-functions which are shown in below table.

SBF	SBF Name	Description
0x01	Hard Reset	This is Equivalent to power reset
0x02	key Off On Reset	It is Equivalent to Ignition On-Off-on
0x03	Soft Reset	It is Equivalent to watch-dog reset of a micro-controller
0x04	Enable Rapid Power Shut Down	It is nothing but the deep sleep mode
0x05	Disable Rapid Power Shut Down	to reverse back from deep sleep mode to ECU run state
0x06 – 0x3F	ISO SAE Reserved	This range of values is reserved by this document for future definition.
0x40 – 0x5F	vehicle Manufacturer Specific (OEM)	This range of values is reserved for vehicle manufacturer specific use.
0x60 – 0x7E	System Supplier Specific	This range of values is reserved for system supplier specific use.
0x07	ISO SAE Reserved	This value is reserved by this document for future definition.

Introduction to UDS Negative Response Codes

UDS protocol is implemented in such a way that if the tester is requesting any task or UDS service request, then the server should complete the work and send the response for completion. But suppose due to any technical or environmental malfunction, the server or ECU is not able to complete due to the wrong request or anything, then the server will send the negative response message to the server with Negative Response Codes (NRC). By the way, the tester can easily analyze and fix the issue or repair the vehicle easily

All the negative response codes are split into three categorized from 0x00 – 0xFF ranges, such as:

0x00: The purpose of this Negative Response Codes (NRC) is to available for a positive response parameter value for server internal implementation.
0x01 – 0x7F: This range is used for the communication-related negative response codes.
0x80 – 0xFF: This range is used as the Negative Response Codes (NRC) for specific conditions that are not correct at that instant of time when the request is received by the server. These response codes could also be utilized whenever the response code 0x22 is listed as valid. By the way to report more specifically why the requested action can not be taken for a positive response.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x00	Positive Response	The 0x00 is a special and initiative value that can not be used for ant NRC. It is this parameter value reserved for several internal code implementation reason.
0x01 – 0x0F	ISO SAE Reserved	This range of NRC is reserved for the future definition that maybe increase the diagnostic functionality more.
0x10	General Reject	If there is none of the NRC is available in UDS standard document to meet the needs of the implementation, then you can use this NRC for your ECU.
0x11	Service Not Supported	Suppose the client has sent a request with a Service Identifier. That is not supporting in your ECU by your OEM. Even if you can also say as per the requirement or as per the OEM. But the service engineer or client has requested that unknown SID request to the server. Then the server will send the 0x11 NRC response message to the client that it is an unknown Service Identifier.
0x12	Sub-function Not Supported	Suppose the client has sent a valid and known Service identifier. But the Sub-function Identifier is unknown or invalid. then the server will send 0x12 Negative Response Code to the client.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x13	Incorrect Message Length Or Invalid Format	If the client has sent any diagnostic request message whose length or the frame format does not match with the prescribed length or the format for that specified service (SID), then the server will send this NRC.
0x14	Response Too Long	If the client has sent a diagnostic request and the response message length is more than the Transport protocol maximum length, then the server will send this NRC. The CAN-TP has 4096 bytes has a maximum length, if the server will send more than that, then this NRC will be sent by the server to the client. Ex: suppose you requested more number of DIDs at a time to read the data if it will cross that limit, then this NRC will get generated by the server.
0x15 – 0x20	ISO SAE Reserved	This range of NRC is also reserved for the future definition that maybe increase the diagnostic functionality more.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x21	Busy Repeat Request	If the server is too busy with some other operation or you can say any other client request in a multi-client environment, then the server will send this NRC to the client. This NRC is supported by each SID. Ex: Suppose there are two clients connected to your vehicle simultaneously. One client has already requested a service that is under process means incomplete and the second client is requesting another request to the server then the server will send the NRC 0x21 to the second server to wait for some time. This time will be defined in the document that you need to check.
0x22	Conditions Not Correct	Before proceeding any service from the client, the server will check for prerequisite conditions are met or not. If not then the server will send this NRC. Ex: you can say the ECU operational low threshold or high threshold voltage, temperature, etc.
0x23	ISO SAE Reserved	This NRC is also reserved for the future definition.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x24	Request Sequence Error	If the client will send the diagnostic message non-sequentially, then the server will send this NRC to the client. Ex: Suppose in security access SID, the client sent security key first and then seed request, then the server will send this NRC. because in this 0x27 service, the client should send the seed request first and then the security key.
0x25	No Response From Sub-net Component	Suppose the diagnostic request service sent by the client to the server, but the received ECU is not the target ECU and it is a gateway. So that the Gateway ECU will send this request to the target ECU and that should perform the operation and send back the response message to the Gateway ECU, and then it will send the response message. If the Gateway ECU does not receive the response message from that target ECU or server then the Gateway ECU will send this NRC to the client. This NRC supported by each SID.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x26	Failure Prevents Execution Of Requested Action	If the client requested service and at that time a fault occurred in server ECU after receiving of this ECU so that at least one DTC status bit for TestFailed, Pending, Confirmed or TestFailedSinceLastClear set to 1, then the server will send this NRC to the client. Basically this failure condition prevents the server from performing the requested action so that the server will not able to execute the client request.
0x27 – 0x30	ISO SAE Reserved	This NRC is also reserved for the future definition.
0x31	Request Out Of Range	If the server received a client request with a parameter (DID, RID) that is out of range, then the server will send this NRC. Suppose in your ECU, the DID or RID having some range used and the client requesting it which is not supported in your ECU or might be not supported in this active session, it will send this NRC.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x32	ISO SAE Reserved	This NRC is also reserved for the future definition.
0x33	Security Access Denied	The server will send this NRC if the security strategy is not satisfied with the server. The Server will send this NRC if any of the below conditions will not satisfy: Server test conditions are not met. Service message sequence (first diagnostic session and then security access service request). The server needs to unlock by the client for some read or write access to the server, but without unlock, a client trying to access the protected memory.
0x34	ISO SAE Reserved	This NRC is also reserved for the future definition.
0x35	Invalid Key	If the server received a security key from the client that is not matching with the server-generated key, then the server will send this NRC.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x36	exceed Number Of Attempts	Basically, in each server, there will be a security retrieval counter and it will be having a limit like 3 or 5 times. So if a client will send the wrong security key more than the defined counter value, the server will send this NRC to the client.
0x37	Required Time Delay Not Expired	Once the client will send a wrong security key, client should wait the defined time, and then it should send the key again. But if the client will send the security key before that then the server will send this NRC.
0x38 – 0x4F	Reserved By Extended Data Link Security Document	This is reserved for future implementation of security layer related NRC
0x50 – 0x6F	ISO SAE Reserved	This NRC is also reserved for the future definition.
0x70	Upload Download Not Accepted	Sometimes due to some fault conditions in the server or server memory, it will not accept any upload or download request from the client.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x71	Transfer Data Suspended	During the data transfer, sometimes due to some failure of any kind of fault, the server will not able to write data onto the memory. So in that situation, the server will send this NRC to the client.
0x72	General Programming Failure	If the server detects an error during the erasing or programming of the memory location of permanent memory (e.g. NVM/Flash/EEPROM), then the server will send this NRC to the client.
0x73	Wrong Block Sequence Counter	Basically whenever a multi-frame consecutive data packets sent by the client to server. In the Consecutive Frame, there will be a Frame index that will increase in each next frame. The server also receives that and compares it with the previous value that should be +1 in each current frame received. If there is any miss match, the server will send this NRC to the client.
0x74 – 0x77	ISO SAE Reserved	This NRC is also reserved for the future definition.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x78	Request Correctly Received-Response Pending	This NRC indicates that the request message from client successfully received by the server. All the requested parameters are also valid. But the server is not ready or due to busy or it is taking some more time to execute the client request in the defined server parameter. So if the client sends another request or the server P2 Client time out happened, the server will send this NRC to the client to inform that to wait for some more time period nothing but the P2* Client timing value. After the execution, the server will send either a positive or negative response message.
0x79 – 0x7D	ISO SAE Reserved	This NRC is also reserved for the future definition.
0x7E	Sub-function Not Supported In Active Session	This NRC will return by the server if the requested sub-function Identifier is not supported in this session. basically, in UDS protocol there are 3 sessions. In each session what services and their sub-functions will support it will be decided by the OEM. If the tester will request a service with a correct subfunction for this service Identifier but that is not supported in this session, then obviously the server will send this NRC to the client. But remember there might be a possibility that the same sub-function and service Identifier will support in any other session.
0x7F	Service Not Supported In Active Session	This NRC will return by the server if the requested service identifier is not supported in this session.

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x80	ISO SAE Reserved	This NRC is also reserved for future definition.
0x81	Rpm Too High	This NRC indicates that the requested action sent by the client will not be taken because the server prerequisite condition for RPM is higher than the defined value in ECU or server.
0x82	Rpm Too Low	This NRC indicates that the requested action sent by the client will not be taken because the server prerequisite condition for RPM is lower than the defined value in ECU or server.
0x83	Engine Is Running	
0x84	Engine Is Not Running	
0x85	Engine Run Time Too Low	
0x86	Temperature is Too High	

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x87	Temperature is Too Low	
0x88	Vehicle Speed is Too High	
0x89	Vehicle Speed is Too Low	
0x8A	Throttle/Pedal is Too High	
0X8B	Throttle/Pedal IS Too Low	
0X8C	Transmission Range Is Not In Neutral	
0x8D	Transmission Range is Not In Gear	
0x8E	ISO SAE Reserved	
0x8F	Brake Switch(es) Not Closed (Brake Pedal not pressed or not applied)	
0x90	Shifter Lever Not In Park	
0x91	Torque Converter Clutch is Locked	
0x92	Voltage is Too High	

Introduction to UDS Negative Response Codes

NRC Code Value	NRC Code Name	Description
0x93	Voltage Too Low	
0x94 – 0xEF	Reserved For Specific Conditions Not Correct	
0xF0 – 0xFE	Vehicle Manufacturer Specific Conditions Not Correct	
0xFF	ISO SAE Reserved	

References:

https://en.wikipedia.org/wiki/Unified_Diagnostic_Services

https://piembsystech.com/can-tp-protocol/

https://automotiveembeddedsite.wordpress.com/uds/