Chapter 1 **Device Overview**

For vehicle network bus, communication protocol is greatly varied due to historic reason and many vehicle manufacturers. Generally, developer has no clue on how to develop a universal product related to vehicle network. Even if he tailors some functions for a vehicle, these functions cannot be completed for further function of product because of limitation of product hardware or understand manufacturer design. Currently, all existing protocol chips are tailored for the function of OBD2 diagnosis, along with undeveloped bus drive, few protocol support, incomplete function even if with support protocol, slow communication rate and bad extensibility. Many functions are unable to satisfy user's needs. We aim to implement the communication among all vehicles and finish all known, or unknown but predicable communication tasks with the simplest hardware, most reliable bus drive, at lowest cost, depending on the strongest extensibility and convenient user interface. Thanks to our efforts in research, test and development for years, we have developed the device based on large-scale integrated circuit technology, finishing the impossible task.

This device is full-functional vehicle network communication protocol controller, with built-in enhanced CAN2.0B Active module, J1850 (VPW 10.4KP and PWM 41.6KB) controller, and enhanced EUART controller. These three modules can simultaneously receive and send data at a high speed, which can be applied to vehicle diagnosis, signal control, vehicle gateway, bus data monitoring analysis, etc. Support all application layer protocols of ISO14230 （ KWP2000 ） /ISO9141-2/KW1281/LIN1.x/LIN2.x/SAE J1708/SAE J2610(SCI)/SAE J1850VPW/J1850PWM/ISO15765/SAE J1939/VW TP2.0/SAE J2411 link layer and physical layer. User-defined control mode can connect to J1850 physical bus, CAN physical bus, and any communication protocol of UART-based physical bus. Almost all common vehicle network communications can be supported by the simplest hardware circuit.

User dual-functional interface:

A. Use UART to connect, the Baudrate up to 10MBPS, easily connected to PC (personal computer).

B. Use SPI (serial peripheral interface) to connect MCU at high speed, communication rate up to 12MBPS, easier control and higher transmission efficiency.

1.1 Enhanced CAN2.0B Controller

- 8-frame transmit buffer, 16-frame FIFO receive buffer
- 16 receive filter and 3 masks
- Loopback, listen and listen all message mode. Used for message mode of diagnosis and bus monitor
- Wake-up in receiving CAN message
- Automatic processing to remote request.
- DeviceNet™ addressing mode
- Support ISO15765 auto-respond to multi-frame transmission
- Support J1939 auto-respond to multi-frame transmission
- Support VW TP2.0 auto-respond to multi-frame transmission
- Support SAEJ2411 single-line CAN (GM SWC)
- 2-channel CAN output, switch via internal electronic switch
- Fully support CAN 2.0B Active Versions, communication rate up to 1Mb/s:

1.2 Enhanced J1850 Controller

- 10 FIFO receive buffers
- 1 transmit buffer and internal transmit cache, actually identical to 2 FIFO caches
- Fully support J1850 technical code, support 10.4K VPW and 41.6KBPS PWM
- IFR1/IFR2/IFR3 in-frame respond and receive
- Support IFR3 automatic responds up to 10 bytes
- No-error arbitration transmit and receive, automatic hardware synchronization in transmiting
- High-speed CRC hardware check
- Support Single Header/Consolidated Header
- Support Single Header IFR (in-frame response)
- IFR1/IFR2 can simulate 8 physical ID responds
- ID/Physical Addr. Functional Addr. filtering
- Wake-up in receiving J1850 message

1.3 Enhanced EUART Controller

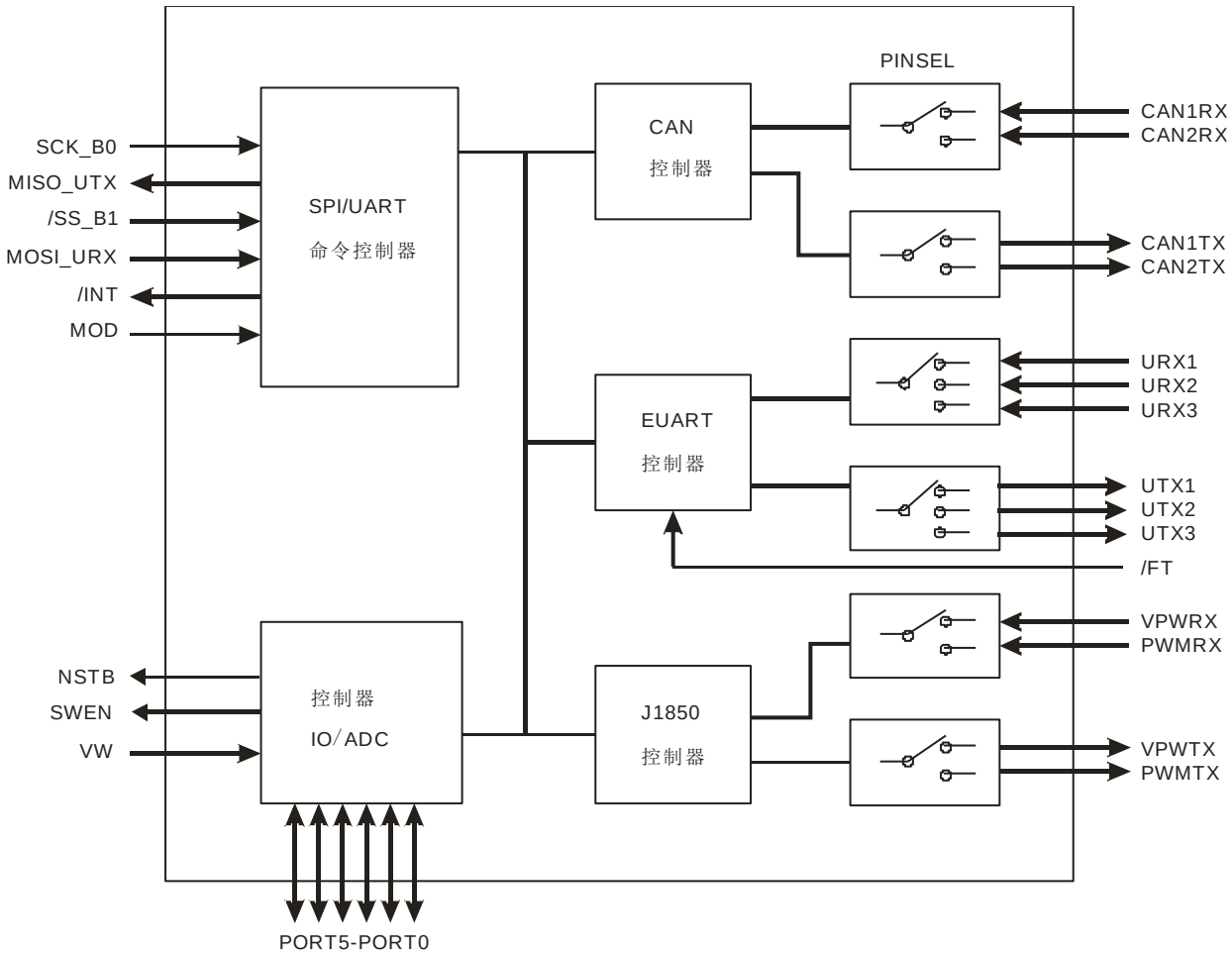
- Used as universal UART module, when there is 4-character transmit and receive FIFO buffer
- If start bit is detected, device programmable wake-up from sleep mode
- Support ISO14230 (KWP2000) frame mode, auto-identify all frame formats defined in ISO14230
- Support ISO9141-2 frame format
- Support user-defined frame format, support all manufacturer protocols (KWP1281 J2610 SCI J1708, etc.), theoretically support any UART-based communication on vehicle
- Support LIN1.x LIN2.x master mode, slave mode and auto-response
- Support the fast initialization of vehicle ECU diagnostic function and the simulation of 5 BaudRate initialization
- Support K-line logic analysis, used as utility tool for the development of diagnostic tester

- User can precisely set all times of all protocol frames
- TX, RX in 3-channel each can input and output pin via UART of internal electronic switch
- 2 FIFO receive buffers with up to 262 bytes in each frame
- Support EUART bus fault protection

1.4 Other Functions

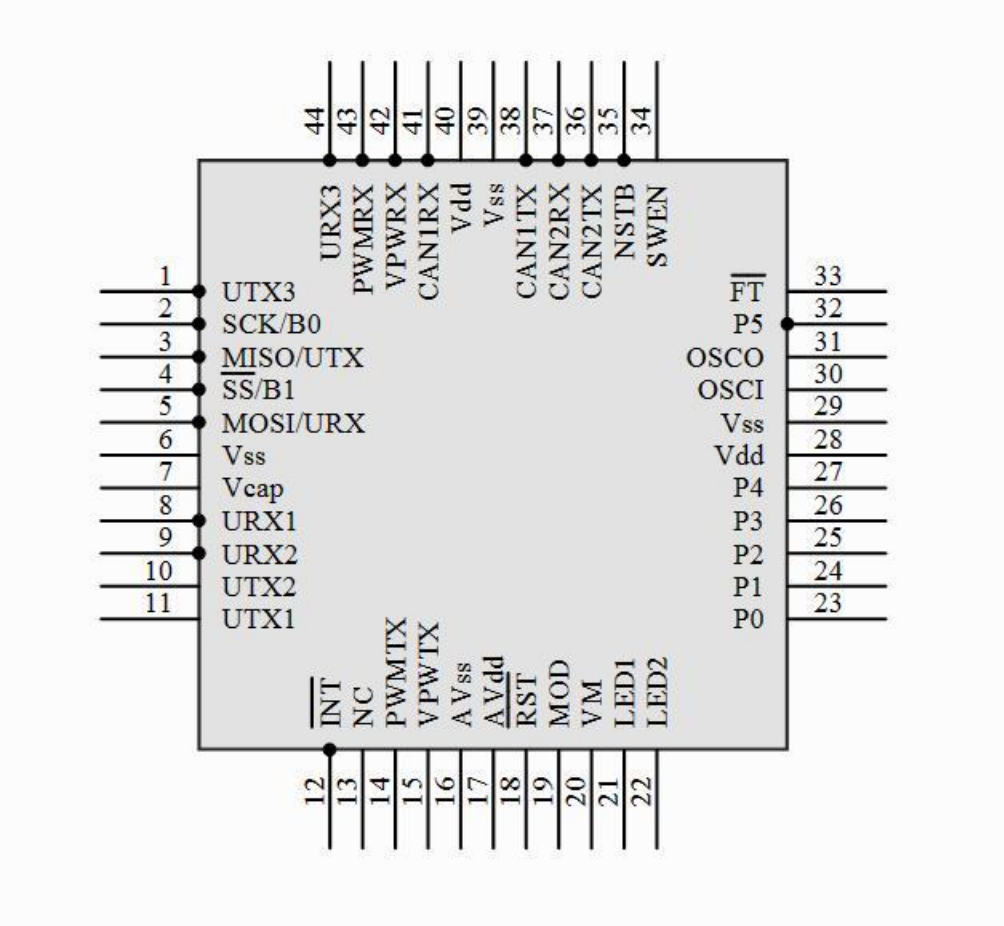
- 1-channel 10-bit ADC for measuring voltage of battery.
- 8 user-programmable multi-functional IO input and output
- 1 interrupt control pin output
- 3.3V (3.0-3.6V) operating voltage, low-power consumption and high-speed technology
- Industrial-grade temperature (-40°C to +85°C)
- Wake-up from sleep mode
- SPI and UART user dual-functional interface, TQFP44 10x10mmpin package

1.5 ET7190 Function Block Diagram



Chapter 2 IC Lead Function and Description

2.1 Pin Figure TQFP44



2.2 Description of Input/Output Pins

Name	Pin Type	Input Type	Description	State after Reset	5V
UTX1 UTX2 UTX3	O	-	EUART 1# Transmit pin EUART 2# Transmit pin EUART 3#Transmit pin	High impedance	Y
URX1 URX2 URX3	I	ST	EUART 1#input pin EUART 2#input pin EUART 3# input pin	High impedance	Y
SCK/B0	I	ST	When MOD=1, SPI clock pin When MOD=0, set pin as Baudrate of B0 connected with UART Built-in pull-up resistor	Input	Y
MOSI/URX	I	ST	When MOD=1, SPI MOSI When MOD=0, as input pin of RX connected with UART	Input	Y
MISO/UTX	O	-	When MOD=1, SPI MISO When MOD=0, as output pin of TX connected with UART	Output	-
/B1	I	ST	When MOD=1, SPI SS When MOD=0, set pin as Baudrate of B1 connected with UART Internal pull-up resistor. Low level is valid for device	Input	Y
	O	-	Interrupt output pin, low level is valid OC open circuit output, allow for external 5V pull-up	Output	-
VPWTX PWMTX	O	-	Output pin when J1850 operates in VPW mode Output pin when J1850 operates in PWM mode	High impedance	N
VPWRX PWMRX	I	ST	Input pin when J1850 operates in VPW mode Input pin when J1850 operates in PWM mode	High impedance	Y
MOD	I	CMOS	When user interface selects pin MOD=1, as SPI. When MOD=0, as UART interface Internal pull-up resistor	Input	N
VM	I	Analog	ADC switched simulated voltage input	Input	N
LED1 LED2	O	-	LED1 indicator output LED2 indicator output	Output	-
SWEN	O	-	IO output pin, set as OD open circuit output	Output	-
NSTB	O	-	IO output pin, set as OD open circuit output	Output	-
CAN1RX CAN2RX	I	ST	CAN 1#input pin CAN 2#input pin	High impedance	Y
CAN1TX CAN2TX	O	-	CAN 1#output pin CAN 2#output pin	High impedance	Y
P4 –P0	I/O	ST	Programmable input/output interface Allow for separate configuration as input, output, OD open circuit output, internal pull-up resistor	Input	N
P5					Y
	I	ST	Fault protection input of EUART module, low level is valid Pull up to Vdd if unimplemented	Input	N
	I	ST	Reset input, low level is valid for device	Input	N
AVDD	P	P	VDD of simulation module	-	-

AVSS	P	P	VSS of simulation module	-	-
VDD	P	P	VDD of logic and IO pin	-	-
VSS	P	P	VSS of logic and IO pin	-	-
Vcap	P	-	Filter capacitor connection of internal power	-	-
OSCI OSCO	I O	CMOS -	16M oscillator input oscillator output	-	-
NC			No connection, suspended		

Note: CMOS = CMOS compatible input or output Analog = Analog input P = Power
ST = Schmitt trigger input in CMOS level O = Output I = Input
Y/N= In the state of high impedance or input, if pin allows 5V input voltage

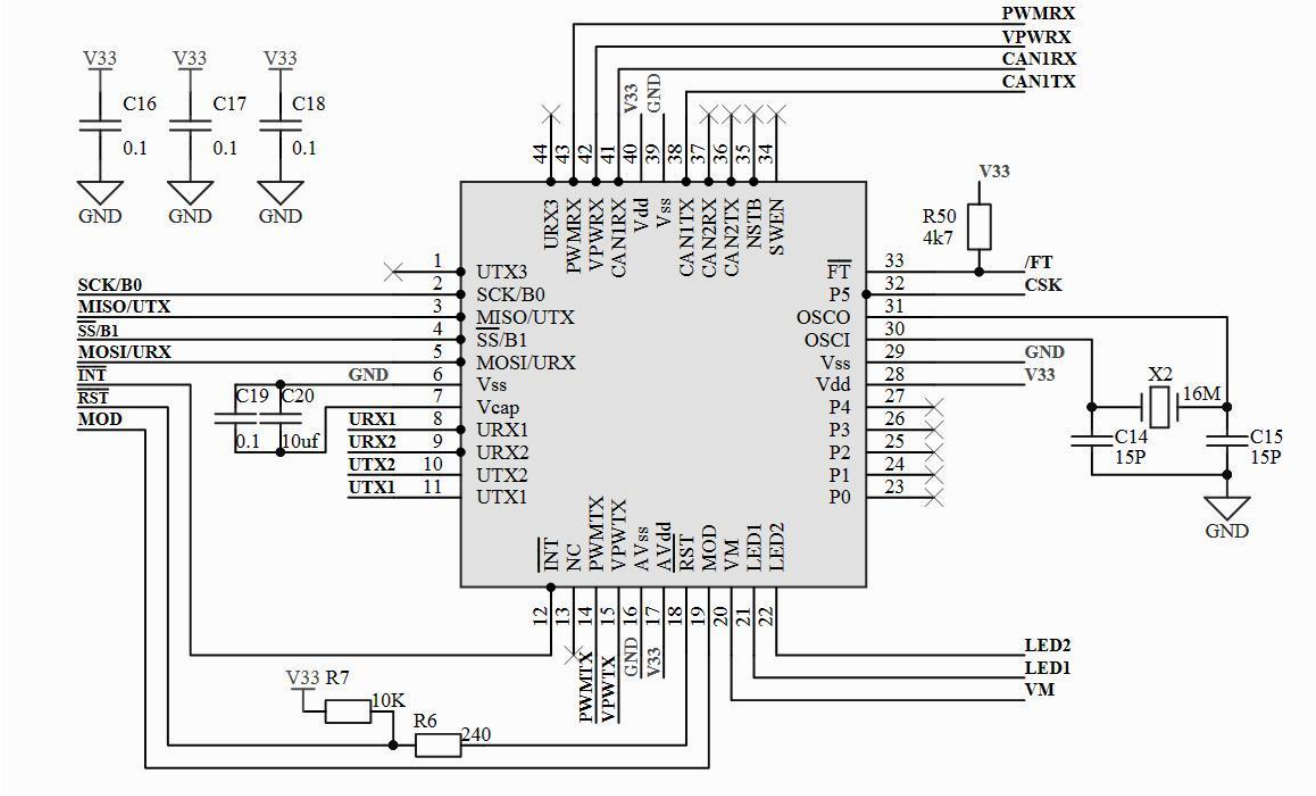
Chapter 3 Schematic Diagram of Typical Application

In the schematic diagram, Vcc is +5V power supply, V33 is 3.3V power supply. Vbat is the vehicle battery voltage (12V or 24V), Vbat should be serially connected to a rectifier diode for reverse protection, additionally with capacitator/ inductance filter circuit. Sometimes the interference signal of vehicle circuit may be strong. (Especially in caes of power generator or ignition circuit failure, strong interference may be created). Application system must ensure normal operation in case of strong interference.
ET7190 protocol is full-functional, which enables the data transmission of all protocols with almost all vehicle modules by simplest hardware, typical example applications are given below to implement the multi-functional vehicle communication interface, various special diagnoses/ECU refreshing and other special functions in this simple circuit, and show super suitability to nonstandard protocol standard.

3.1 Application of Universal OBD Diagnosis

3.1.1 ET7190 Main Device Circuit

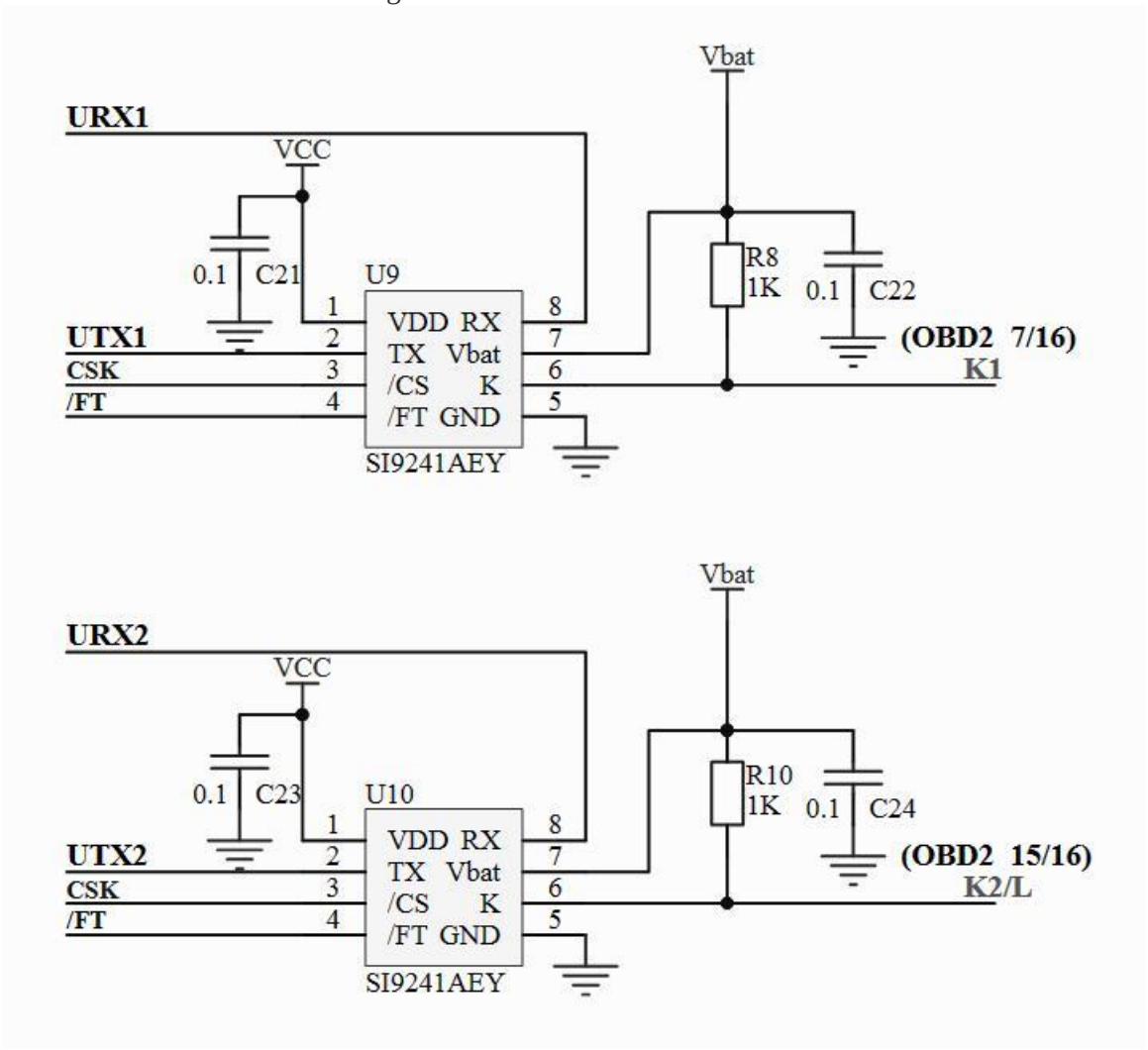
Standard OBD2 only uses partial ET7190 functions, if pin is unimplemented, P4:P0,UTX3/URX3 CAN2TX/CAN2RX NSTB/SWEN can be totally suspended, (for P4:P0,UTX3/URX3 CAN2TX/CAN2RX, internal pull-up resistor of pin can be configured to be valid, avoiding signal interference), NSTB/SWEN , LED1/LED2 are output pins, or directly suspended if unimplemented.
After powering-up SI9241A chip, if /FT adds protective action, a high level is required to add on /CS for reset, this schematic diagram uses P5 port to control SI9241A /CS pin.



For the connection with ET7190, user can use MCU to connect via SPI/UART or PC/Tablet, as shown in the circuit schematic diagram, see User and ET7190 Connection in Section 3.4.

3.1.2 EUART Connection (K Bus Drive)

3.1.2.1 Connection Schematic Diagram



3.1.2.2 Description of K-line Usage

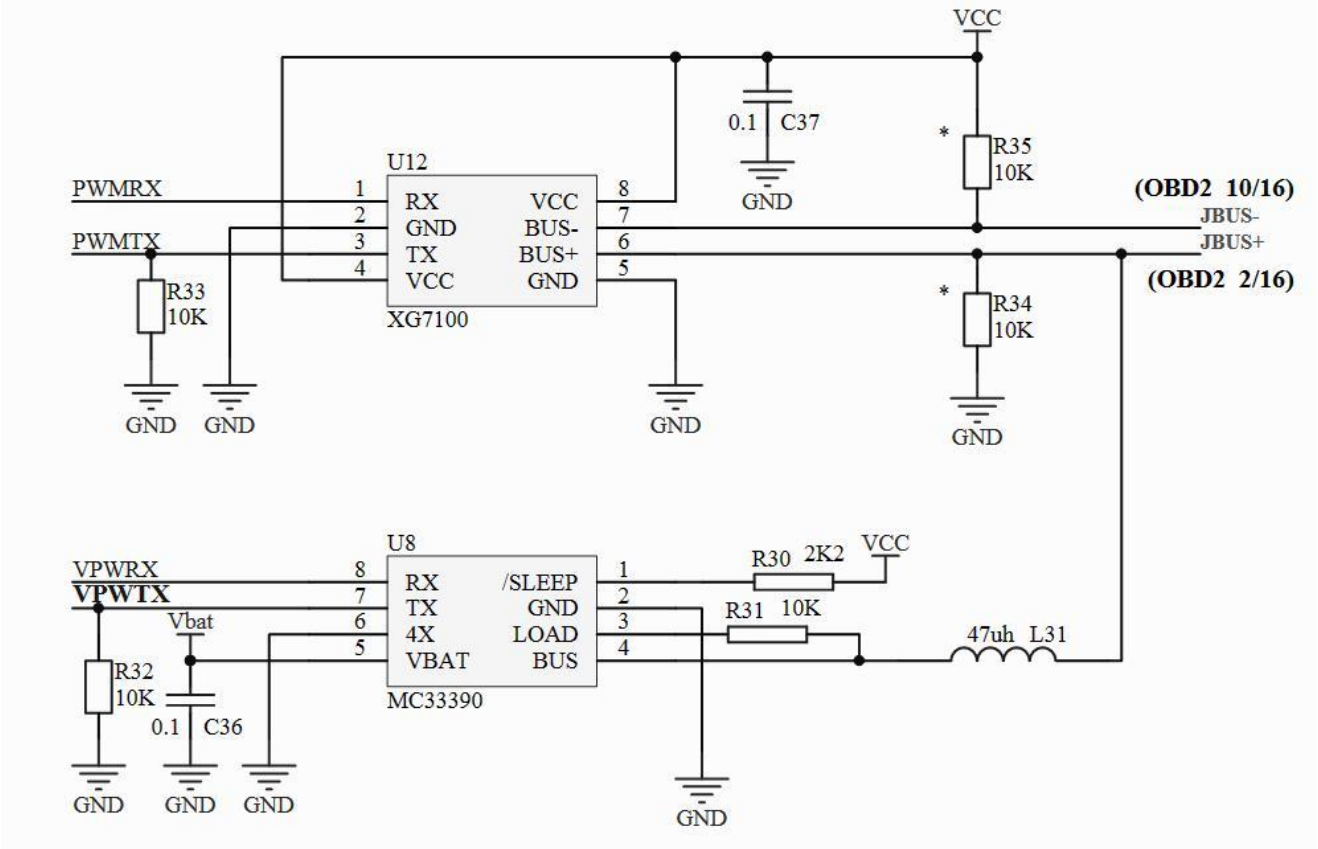
- /FT is SI9241A fault protection output (ODC), K-line short-circuit to ground or power source will trigger fault protection action, /FT level decreases, when UTX output signal will not transfer to K bus line, user needs to add a high level to /CS pin to reset.
- Vbat may fall into 12V-36V, which is applicable to vehicle in 12V/24V.
- When EUART control register UPINSEL is configured as UTX1/URX1 (two-way half-duplex), all communications are conducted on K1 line, (K2 pin is used as alternate L line), this case is only tailored for most application protocols, which is applicable to ISO9141 /ISO14230 （KWP200）/LIN1.x/LIN2.x/KW1281 and most commercial vehicle protocols.
- When EUART control register UPINSEL is configured as UTX1/URX2 (two-way half-duplex), K1 is used as SCI (SAE J2610) TX pin, K2 as SCI RX pin. If used as SCI communication, pay attention to bus line voltage, SCI bus voltage is 5V or 12V, where the schematic diagram only uses Vbat voltage, if more applications are required to adapt, P4：P0 port can be used to

control or convert SI9241 Vbat power supply voltage (In actual test, SI9241A allows min. Vbat to be 5V).

- When EUART control register UPINSEL is configured as UTX2/URX2, all communications are conducted on K2 line, (K1 line is invalid), this case is only tailored for some vehile placing some ECU K line on 15 pin at OBD2 interface, this configuration can be directly used for communication with no need to change hardware connection, where protocol support is not different from K1.

3.1.3 J1850 Bus Connection

3.1.3.1 Connection Schematic Diagram

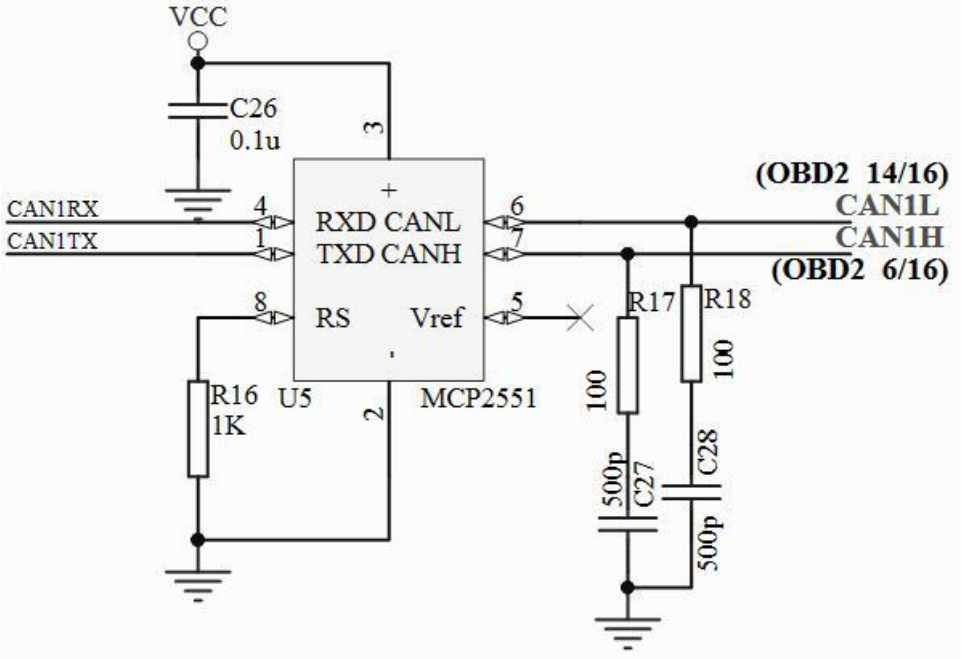


3.1.3.2 Description of J1850 Bus Usage

- XG7100 is bus drive chip specific for PWM, and MC33390 is VPW bus drive chip. Vehicle OBD2 interface defines VPW bus and PWM BUS+ on the same terminal (2/16 J1850 BUS+), so these two lines must be connected in the universal diagnostic device. XG7100 and MC33390 bus drive chip can avoid affecting the communication of the other bus line in recessive state. Bus drive chips are under the internal ESD protection.
- J1850 VPW is generally used on GM vehicle series, while J1850 PWM bus line used on FORD vehicle series.

3.1.4 CAN Bus Connection (Double-line)

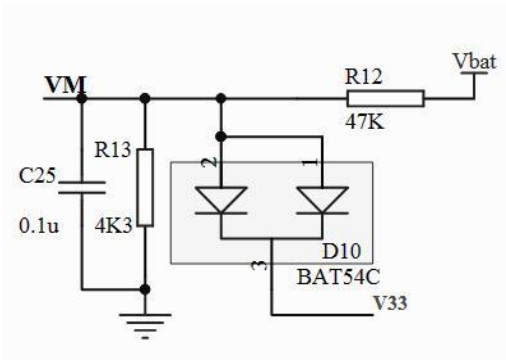
SWCAN single-line CAN doesn't fall into OBD2 definition, but a low-speed communication bus in GM vehicle, see single-line SWCAN in Section 3.2.3. Bus drive of double-line CAN is given below:



ET7190 is built in the multi-frame receive/transmit protocol controller in ISO15765 / J1939 /VW TP2.0 network layer, which can transmit or receive long message multi-frame data even if user doesn't attend to timing, frame-to-frame transmit and receive control response are automatically finished by ET7190. It is particularly applicable to WINDOWS or other non-real-time OS (operating system) in case that time cannot be precisely controlled. This also makes easy for user to handle transmission and reception.

3.1.5 Level Voltage Measurement (ADC)

BAT54C functions as voltage limit to prevent any voltage higher than 3.3v from connecting to VM measure pin.

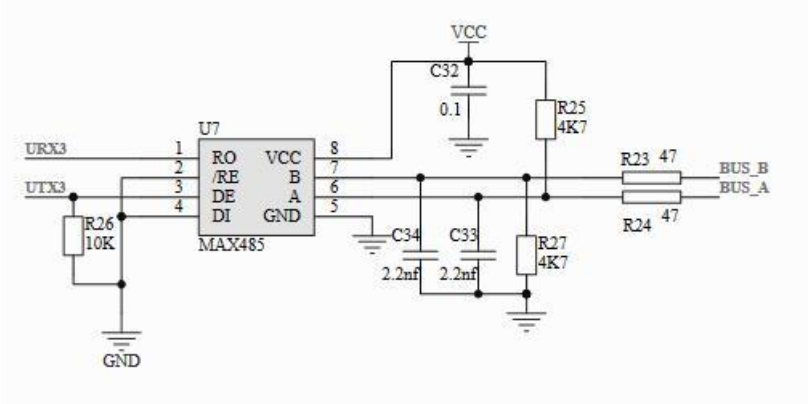


3.2 Other Vehicle Application Circuit

3.2.1 EUART UTX3/URX3 Connection (SAE J1708 Bus)

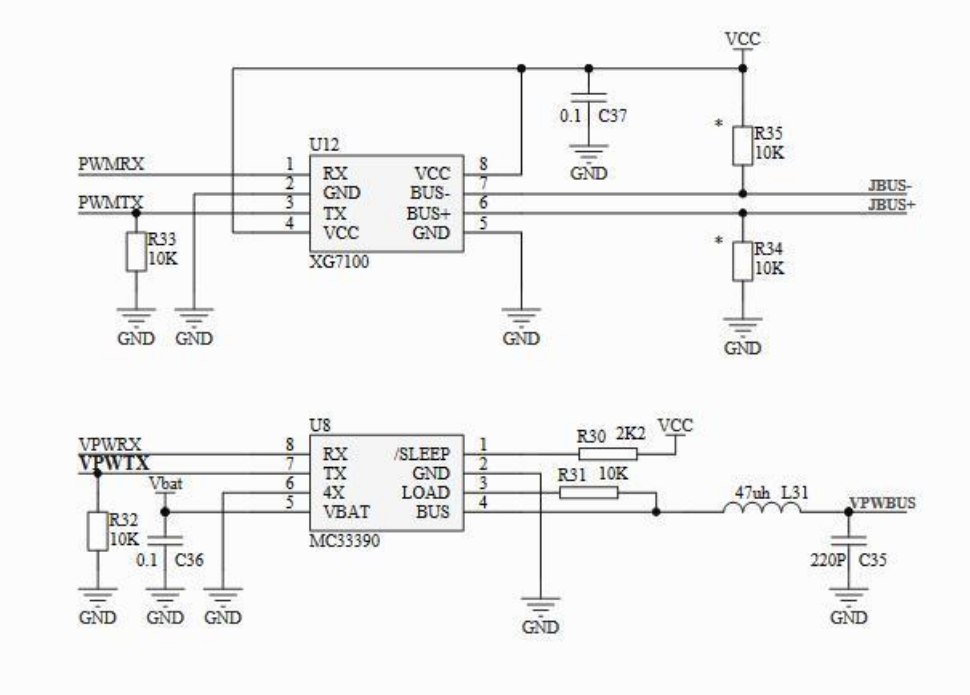
SAE J1708 is usually used in commercial vehicle, along with the application layer SAE J1587. Domestic heavy truck also applies SAE J1587 for the diagnosed varied application in K line, but J1708

timing and ISO9141 physical bus line are used. ET7190 EUART user-defined module can adapt to this application. Standard SAE1708 bus is as shown below. FORD DCP protocol also applies this bus line.

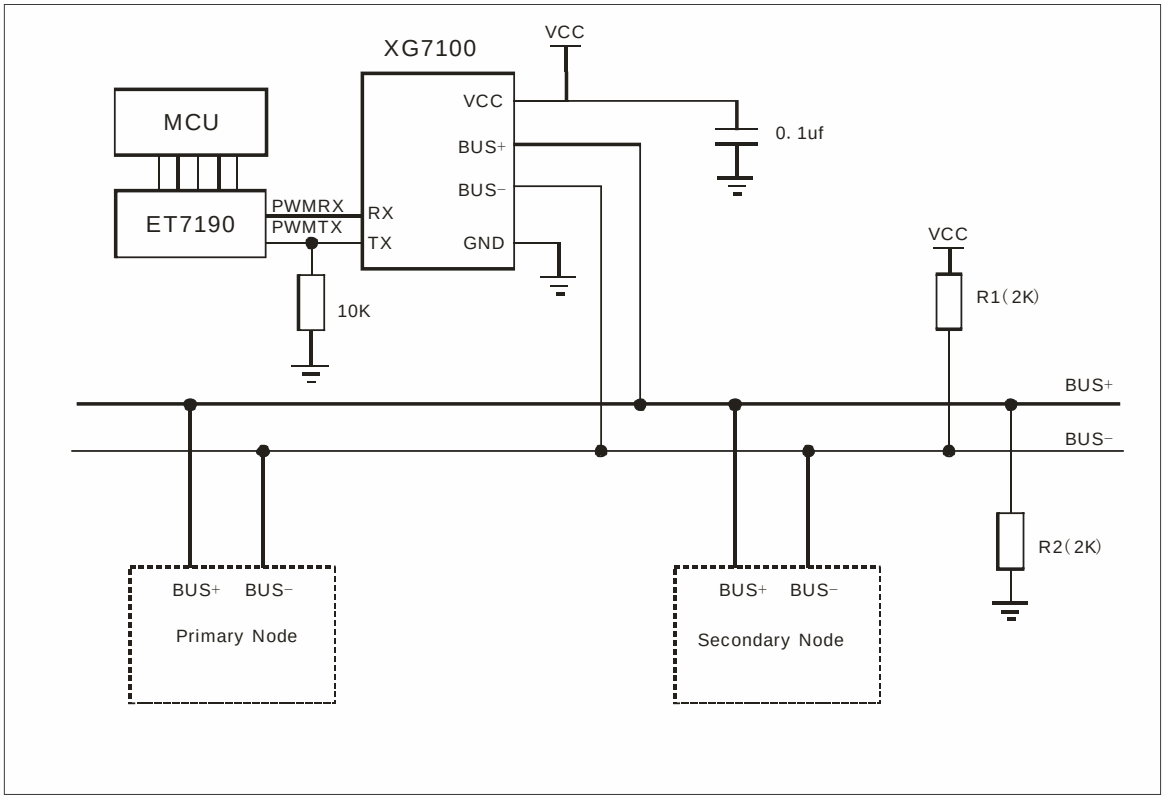


3.2.2 J1850 Bus Stand-alone Application

In special application, if only one physical connection is used, either VPW or PWM, the following stand-alone vehicle network can be applied. It is not required to connect R34/R35 in PWM to the internal vehicle network (Designed functions of diagnostic tester: Prevent bus from being suspended). J1850 network shares 2 of its used (2k2) resistors for pull-up and pull-down respectively.



3.2.3 J1850 PWM Network connection attention



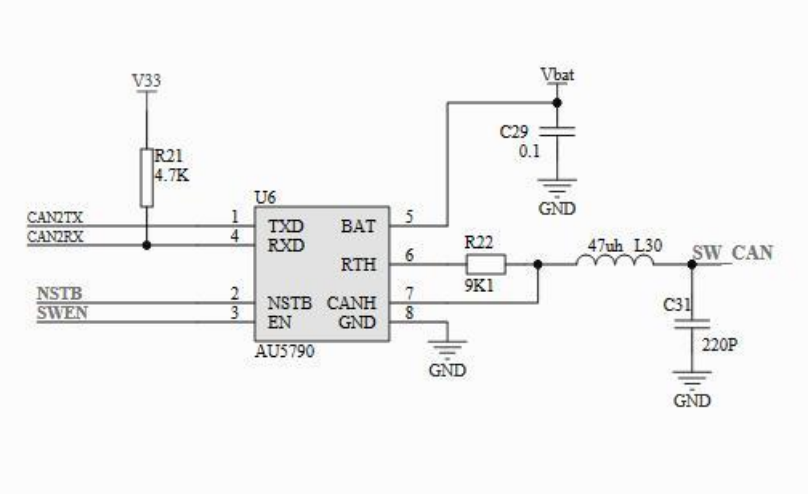
- If used as an external node (diagnostic instrument, etc.), There are alredy two bus resistance on the vehicle network . Do not need other resistance.

- If you use ET7190 as ECU simulation, the bus is not connected with the vehicle, the bus R1/R2 Resistor must be connected. In the ECU simulation, some diagnostic equipment is driven BUS by discrete components , the BUS signal waveform is not standard. The value of these two resistors will directly affect the diagnostic equipment to receive and transmit signal waveforms. That makes diagnosis Equipment can not normal communication.

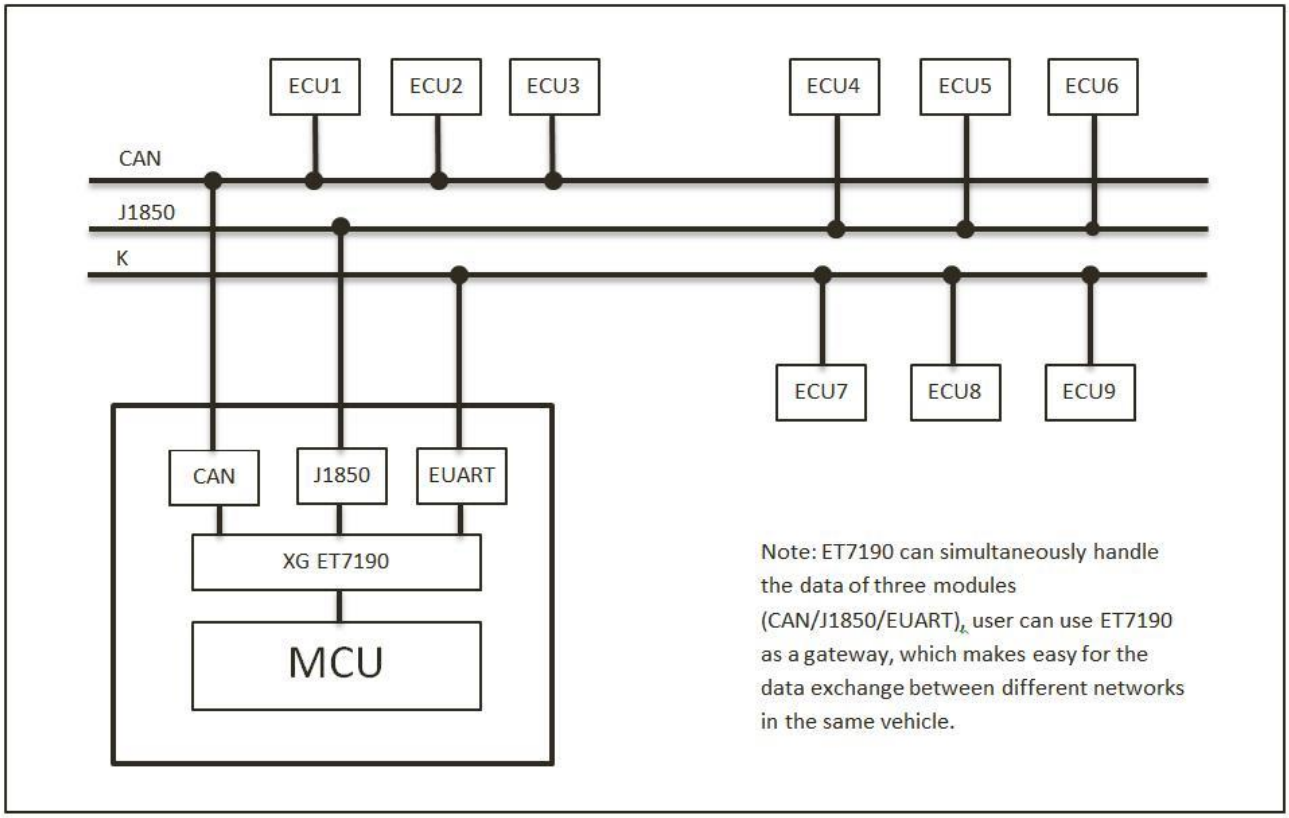
- If you only use two ET7190 development board to test, you can not pick up these two resistance. (due to the BUS of development board is Driven by a XG7100 Transceiver, the size of the resistance value effects the transmission signal waveform is very small. There are already two 10K resistances on the development Board.)

3.2.4 SWCAN (SAE J12411 Single-line CAN)

SAE J2411 standard is equivalent to GM3089 protocol by GM, which is widely applied in the body low-speed network of GM vehicle. Its upper layer protocol is also ISO15765. Application layer protocol is GMW3110. Generally connected to 8# terminal at OBD2 port on GM vehicle. Used for the application of body control module. Any group of pins (CAN1TX/CAN1RX or CAN2TX/CAN2RX) in ET7190 CAN can be connected to AU5790 single-line CAN drive chip. NSTB/SWEN controls AU5790 operation mode.



3.3 Application as Vehicle Gateway



3.4 User and ET7190 Connection

3.4.1 Connection with MCU

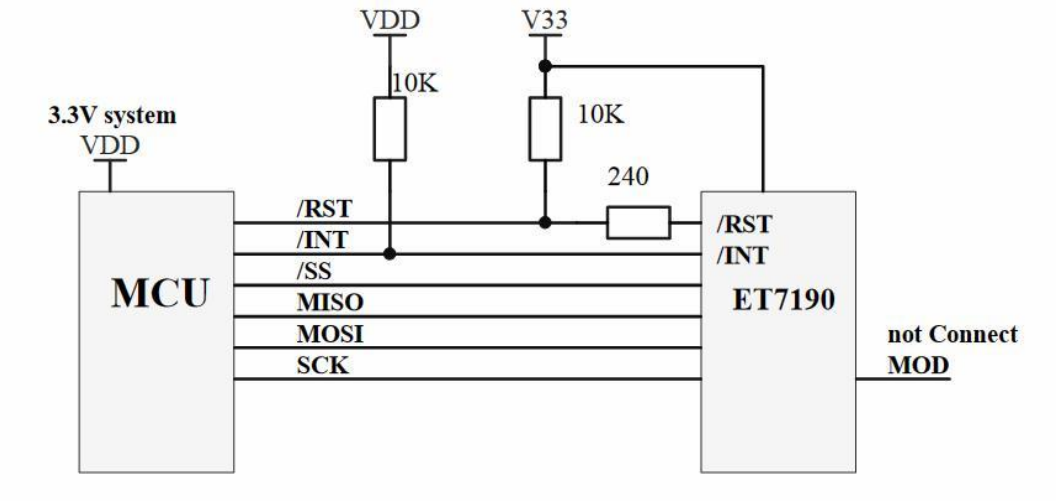


Figure: Connection with MCU via SPI and ET7190

- As shown above, MCU is a system in 3.3V, if in 5V, /RST pin cannot be directly connected, as shown below, level is converted by the same method (/RST pin must not exceed V33). Other pins can be directly connected to 5V system.
- MOD is set as 1 via SPI. Pin can be standoff, no need to externally connected with pull-up resistor (ET7190 is built in pull-up resistor)
- It is recommended to connect with MCU via SPI, because it actively transmits and receives ET7190 data to and from MCU. In this way, end command or start new command is implemented with higher efficiency via /SS pin at any time. In UART mode, data must be transmitted in the fixed command format because of no stand-alone control pin, see SPI/UART User Interface Control Command in Chapter 10 to carefully examine the difference.

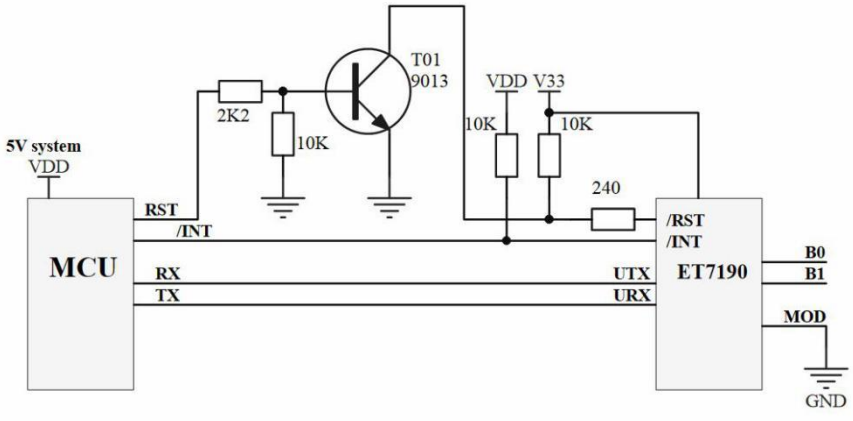
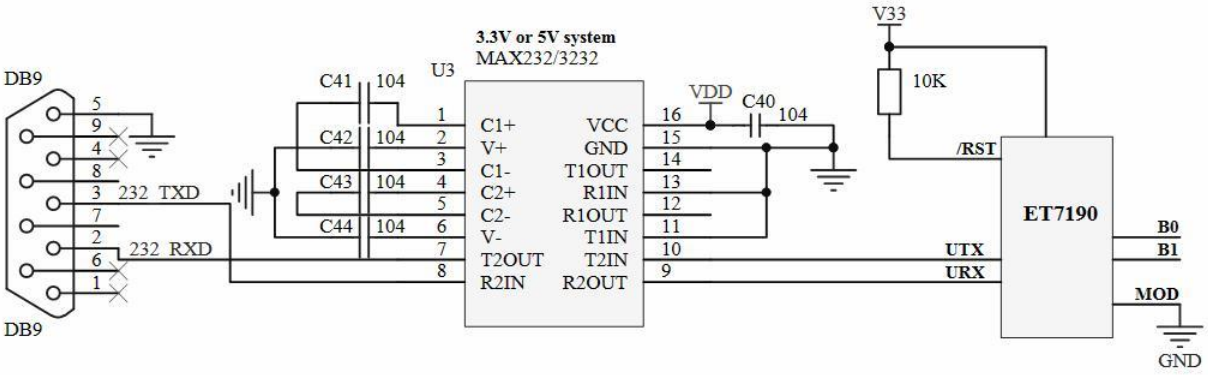


Figure: MCU Connection with ET7190 via UART

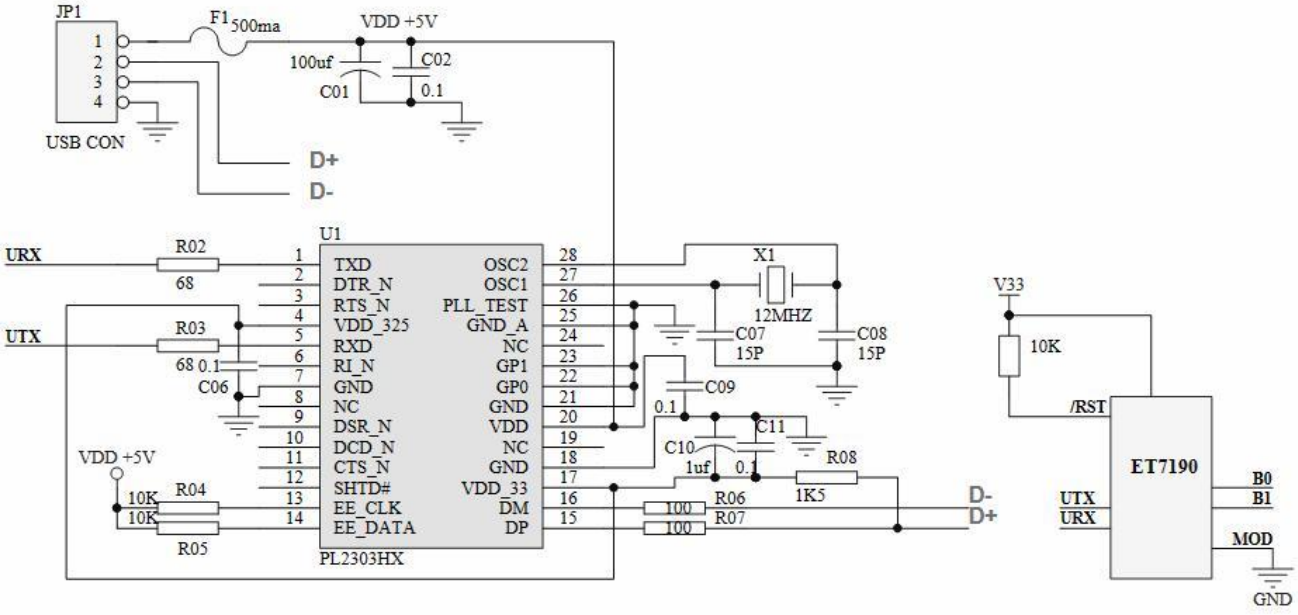
- As shown above, MCU is a sytem in VDD=5V, if in 3.3V, /RST pin can be directly connected via SPI.
- MOD is set as 0 via UART. (B0B1 is built in pull-up resistor) Use B0 or B1 to select or change UART initial Baudrate, control register DRVUBRG can dynamically change the communication Baudrate of user interface. Up to 10MBPS.
- UTX/URX can be directly connected with MCU. With no need of level switch.

3.4.2 Connection with PC RS232



PC RS232 serial port communication is as shown above, by which user only communicates with ET7190 via query mode.

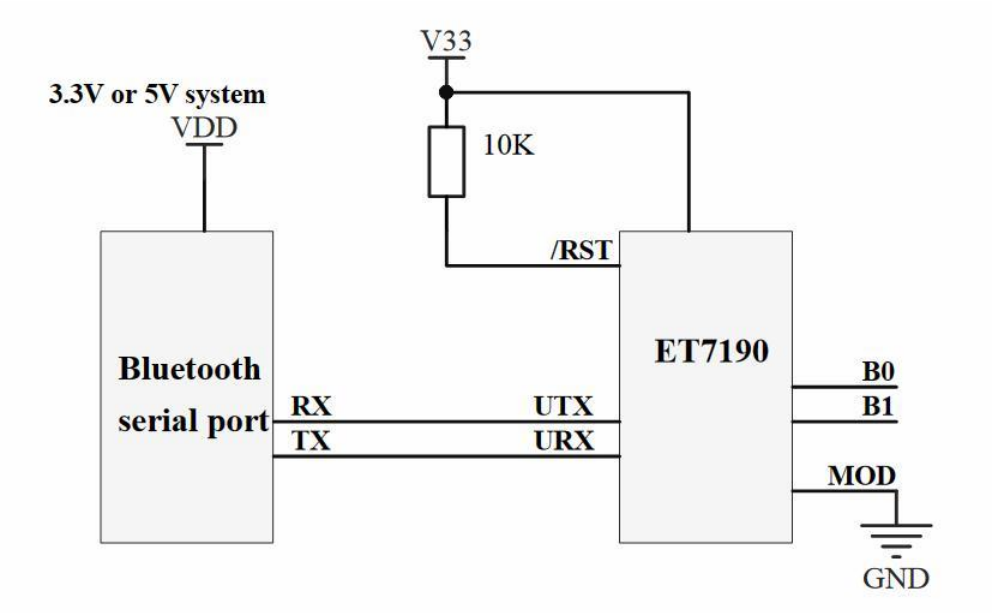
3.4.3 Connection with USB Serial Port Chip PL2303HX



This is not different from RS232 operation mode, where PL2303HX only serves as a serial port of USD-RS232 transition.

If MCU is unimplemented, it is recommended to use PL2303HX instead of MAX232 or other chips if ET7190 is intended to be directly connected with PC. Its advantages include cheap price of PL2303HX and very stable functionality. Communication Baudrate above MAX232 is applicable.

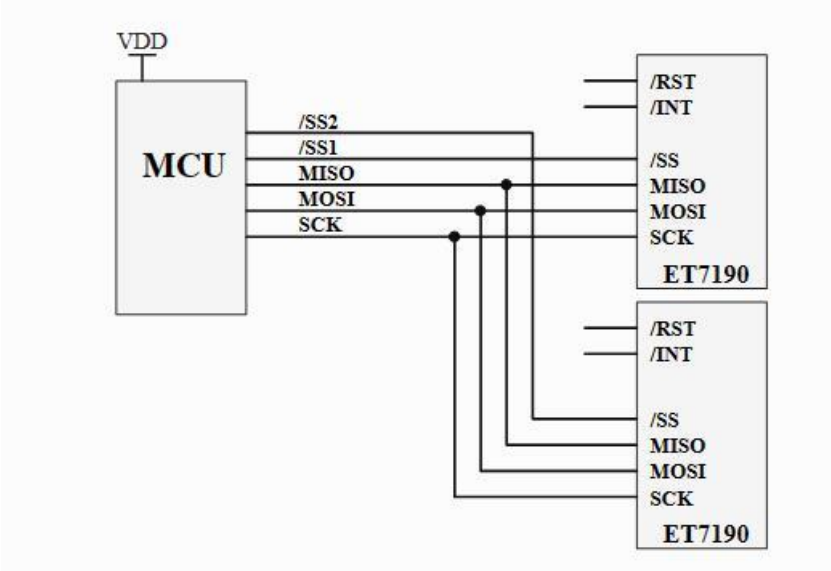
3.4.4 Connection with Wireless Bluetooth Serial Port



The same usage with RS232 connection, user programming shows no difference.

3.4.5 Connection with Multiple ET7190 Devices via SPI

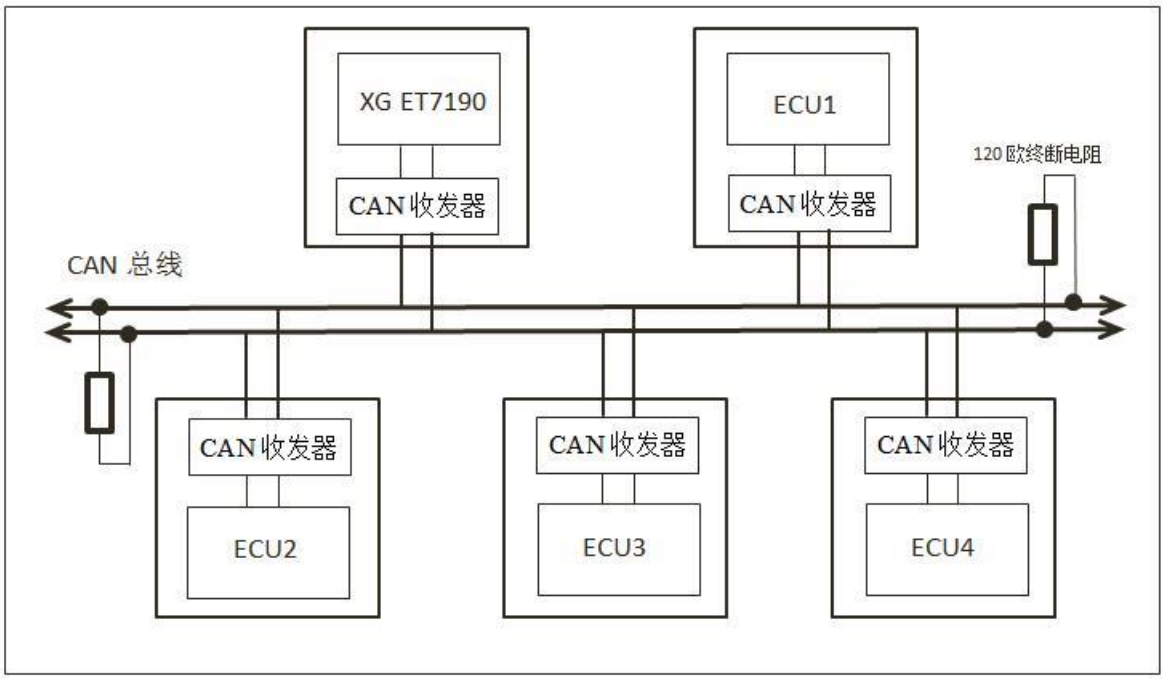
ET7190 /SS is controlled via IO port, MCU one-channel SPI allows for the connection of multiple ET7190 devices at the same time. Certainly user may connect two ET7190 via two stand-alone SPIs.



Chapter 4 Enhanced CAN Module

4.1 Introduction

Enhanced CAN module implements CAN 2.0B protocol, mainly used for industrial and vehicle application. This asynchronous serial data communication protocol can provide reliable communication in electrical noise environment. Fig.4-1 shows the typical double-line CAN bus topology.
Fig.4-1:



XG ET7190 CAN module supports the main characteristics below:

- 8-frame transmit buffer, 16-frame FIFO receive buffer
- 16 receive filter and 3 masks
- Loopback, listen and listen all message mode. Used for message mode of diagnosis and bus monitor
- Wake-up in receiving CAN message
- Automatic processing of remote send request
- DeviceNet™ addressing mode
- Support ISO15765 multi-frame transmission auto-respond
- Support J1939 multi-frame transmission auto-respond
- Support VW TP2.0 multi-frame transmission auto-respond
- Support SAEJ2411 single-line CAN (GM SWC)
- 2-channel CAN output, universal internal electronic switch
- Fully support CAN V2.0B Active technical code, communication rate up to 1Mb/s:

4.2 CAN Message Format

CAN bus protocol supports 5 frame types as follows:

- Data Frame ——Contain data transmitted from transmitter to receiver
- Remote Frame ——Transmit from one node in bus line, used to RTS data frame with the same identifier from another node
- Error Frame——Transmit from any node in case of checking error
- Overload frame——Provide extra time delay between continuous data frames or remote frames
- Interframe Space —— Provide space between continous frames

CAN 2.0B code also defines two extra data formats:

- Standard Data Frame——Used for standard message with 11 identifiers
- Extended Data Frame——Used for extended message with 29 identifiers

CAN bus norm includes 3 versions:

- 2.0A——Treat 29-bit identifier as error
- 2.0B Passive——Ignore message with 29-bit identifier
- 2.0B Active——Handle 11-bit and 29-bit identifier

ET7190 CAN module complies with CAN 2.0B Active norm, and enables enhanced function of message filtering.

Note: See Bosch CAN bus norm for more details about CAN protocol.

4.3 CAN Module Register

CAN module contains a lot of registers and register sets (buffers) with special function, which is used to configure message receive filter, message buffer, CAN frame transmit process, CAN bus bit definition and CAN frame time in multi-frame transmit. For multi-frame transmit and receive auto-respond of different protocols, data transmission via SPI or UART serial communication is available as long as relevant registers (sets) are appropriately configured.

Register Name	Addr.	Description	Reset Value HEX
CCTRL0	0x10	CAN control register 0, set CAN operation code	44
CCTRL1	0x11	CAN control register 1, used to abort all transmits and clear all receive buffers	00
CCTRL2	0x12	CAN control register 2, control NSTB and SWEN pin	00
CPINSEL	0x13	Select CAN 1TX/RX or CAN2TX/RX pin groups	00
CDNCNT	0x14	This register contains DeviceNet™ filter control bit	00
CINTF0	0x15	CAN error interrupt flag register 0	00
CINTF1	0x16	CAN error interrupt flag register 1	00
CINTE	0x17	CAN error interrupt enable register	00
CTERRCNT	0x18	CAN transmit error counter	00

CRERRCNT	0x19	CAN receive error counter	00
CCFG0	0x1A	Baudrate configuration register 0	00
CCFG1	0x1B	Baudrate configuration register 1	xx
CCFG2	0x1C	Baudrate configuration register 2	xx
CFEN0	0x1D	Receive filter enable register 0	3F
CFEN1	0x1E	Receive filter enable register 1	00
CFMSKSEL0	0x1F	Filter 3-0 Mask select register	xx
CFMSKSEL1	0x20	Filter 7-4 Mask select register	xx
CFMSKSEL2	0x21	Filter 11-8 Mask select register	xx
CFMSKSEL3	0x22	Filter 15-12 Mask select register	xx
CTXPRI0	0x23	Transmitter 3-0 Priority	00
CTXPRI1	0x24	Transmitter 7-4 Priority	00
CTXREQ	0x25	Message transmit request register	00
CTXABT	0x26	Message pending transmission request register	00
CRTREN	0x27	Automatic remote transmit enable register	00
CTXAUTOEN	0x28	After writing into transmit buffer, auto-start transmit register	FF
CPROCON	0x29	CAN multi-frame protocol control	00
CINTF2	0x2A	CAN multi-frame protocol transmit error flag register	00
CCFDL	0x2B	In ISO15765 multi-frame transmit, CETXB buffer data length doesn't contain FF frame data In J1939, data contained in CFTXB, CCFDL is the number of DT transmit frame In VW TP2.0, data length of CFTXB buffer	xx

4.3.1 CAN Multi-frame Auto-Transmit Time Register

For relevant standard defined time register, the following register only uses Command CMD_WR_REG to write value as 0x00-0xFF.
Only when ET7190 automatically transmits multiple frames, for more details about time, please refer to Chapter 5 Relevant Protocol Application.

Register Name	Addr.	Description	Reset Value
CN_BS	0x60	ISO15765 Time_N_BS =xx * 5ms	xx
CN_ST	0x61	ISO15765 Time_N_ST =xx * 100us	xx
CN_BR	0x62	ISO15765 Time_N_BR =xx * 100us	xx
CN_CR	0x63	ISO15765 Time_N_CR =xx * 5ms ET7190 未使用 Unimplemented	xx
CTP_T1	0x64	TP2.0 T1= xx * 5ms Time-out used by this Control mode for received telegrams	xx
CTP_T3	0x65	TP2.0 T3= xx * 100us minimum time when sending between consecutive telegrams	xx
CTP_TE	0x66	TP2.0 TE = xx * 5ms	xx

		Time-out for connection structure waiting for response	
CTP_T3A	0x67	TP2.0 T3 = xx * 1000us minimum time when sending between consecutive telegrams	xx
CTP_Tw	0x6F	TP2.0 Tw = xx * 5ms If Receiver is not ready, insert a delay of T_wait before sending the next Data telegram.	xx
CJ1939_T2	0x68	When J1939 T2= xx * 5ms receives multi-frame, maximum time awaiting time of DT frame	xx
CJ1939_T3	0x69	In J1939 T3= xx * 5ms multi-frame data transmission, maximum time of waiting receiver CTS	xx
CJ1939_TS	0x6A	In J1939 TS= xx * 1ms time slot for consecutive transmission of DT frame	xx
CJ1939_TR	0x6B	When J1939 TR= xx * 1m receives multi-frame, time delay for transmitting respond frame CTS/EOM	xx

4.3.2 CCTRL0: CAN Control Register 0

U-0	R-1	R-0	R-0	U-0	R/W-1	R/W-0	R/W-0
-	OPMODE<2:0>			-	REQOP<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 6-4 **OPMODE<2:0>**：Operation mode bits
000 = Set Normal Operation mode
001 = Set Disabled mode
010 = Set Loopback mode
011 = Set Listen mode
100 = Set Configuration mode

101 = Reserved
110 = Reserved
111 = Set listen all message modes

bit 2-0 **REQOP<2:0>**：Request Operation mode bits

000 = Set Normal Operation mode
001 = Set Disabled mode
010 = Set Loopback mode
011 = Set Listen mode
100 = Set Configuration mode
101 = Reserved
110 = Reserved
111 = Set listen all message modes

4.3.3 CCTRL1: CAN Control Register 1

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
-	-	-	-	-	CRXF	CMTXF	ABAT
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 2 **CTRXF**：Request bit for clearing receiver buffer
1 = Inform clear all receive buffers
0 = Module will automatically clear this bit when clearing is finished
bit 1 **CMTXF**：Terminate multi-frame transmit bit
1 = Inform to terminate transmission of multi-frame transmit buffer
0 = Module will automatically clear this bit when clearing is finished
bit 0 **ABAT**：Abort all transmit bits waiting to be handled
1 = Inform all transmit buffers to abort transmission
0 = Module will clear this set when all transmissions are aborted

In multi-frame transmit, CMTXF must be firstly used to abort multi-frame transmit before using ABAT to abort. CMTXF only clears transmit buffer CFTXB, but it doesn't abort CTXBx in CAN transmit buffer, CTRXF/CMTXF/ABAT can be simultaneously set ON to clear.

4.3.4 CCTRL2: CAN Control Register 2

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	-	ODCTB	ODCEN	NSTB	SWEN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 3 **ODCTB**：NSTB output pin open-drain control
1 = NSTB Pin configured as open-drain output
0 = NSTB Pin configured as digital output

bit 2 **ODCEN**：SWEN output pin open-drain control
1 = SWEN Pin configured as open-drain output
0 = SWEN Pin configured as digital output
bit 1 **NSTB**：Port output bit
1 = NSTB Pin output as high level 1
0 = NSTB Pin output as low level 0
bit 0 **SWEN**：Port output bit
1 = SWEN Pin output as high level 1
0 = SWEN Pin output as low level 0

4.3.5 CPINSEL: CAN Pin Select

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
-	-	-	-	-	-	-	SEL
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 0 **SEL**：Pin select bit
1 = Select CAN2TX and CAN2RX as drive pin of current module

0 = Select CAN1TX and CAN1RX as drive pin of current module

4.3.6 CDNCNT: DeviceNet™ Filter Control Bit

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	DNCNT<4:0>				
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 4-0 **DNCNT<4:0>**：DeviceNet™ Filter numbering bit
00000 = Compare no byte of data
00001 = Compared bit 7 in byte of data 0 with EID<17>
00010 = Compared bit <7:6> in byte of data 0 with EID<17:16>
00011 = Compared bit <7:5> in byte of data 0 with EID<17:15>
00100 = Compared bit <7:4> in byte of data 0 with EID<17:14>
00101 = Compared bit <7:3> in byte of data 0 with EID<17:13>
00110 = Compared bit <7:2> in byte of data 0 with EID<17:12>
00111 = Compared bit <7:1> in byte of data 0 with EID<17:11>
01000 = Compared bit <7:0> in byte of data 0 with EID<17:10>
01001 = Compared bit <7:0> in byte of data 0 with bit <7> and EID<17:9> in byte of data 1
01010 = Compared bit <7:0> in byte of data 0 with bit <7:6> and EID<17:8> in byte of data 1
01011 = Compared bit <7:0> in byte of data 0 with bit <7:5> and EID<17:7> in byte of data 1
01100 = Compared bit <7:0> in byte of data 0 with bit <7:4> and EID<17:6> in byte of data 1
01101 = Compared bit <7:0> in byte of data 0 with bit <7:3> and EID<17:5> in byte of data 1
01110 = Compared bit <7:0> in byte of data 0 with bit <7:2> and EID<17:4> in byte of data 1
01111 = Compared bit <7:0> in byte of data 0 with bit <7:1> and EID<17:3> in byte of data 1
10000 = Compared bit <7:0> in byte of data 0 with bit <7:0> and EID<17:2> in byte of data 1
10001 = Compared bit <7:0> in byte of data 0, bit <7:0> in byte of data 1 with bit <7> and EID<17:1> in byte of data 2

10010 = Compared bit <7:0> in byte of data 0, bit <7:0> in byte of data 1 with bit <7:6> and EID<17:0> in byte of data 2
10011-11111 = Invalid select

4.3.7. CINTF0/CINTF1: CAN Error Interrupt Flag Register 0/1

CINTF0 Error Interrupt Flag

R/C-0	U-0	R/C-0	R-0	R/C-0	R/C-0	U-0	U-0
IVRIF	-	ERRIF	PROEIF	FIFOIF	RBOVIF	-	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **IVRIF**： Received invalid message interrupt flag bit
1 = Interrupt request generated
0 = No interrupt request generated
bit 5 **ERRIF**： Error interrupt flag bit (Multiple interrupt sources in CINTF1 register)
1 = Interrupt request generated
0 = No interrupt request generated
bit 4 **PROEIF**： Frame error interrupt flag bit (Multiple interrupt sources in CINTF2 register)
1 = Interrupt request generated (read only, set CINTF2 clear, this bit is automatically set clear)
0 = No interrupt request generated
bit 3 **FIFOIF**： FIFO Almost Full Interrupt Flag bit
1 = Interrupt request generated
0 = No interrupt request generated
bit 2 **RBOVIF**： Receive buffer overflow interrupt flag bit
1 = Interrupt request generated
0 = No interrupt request generated

CINTF1 Error Interrupt Flag

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
-	-	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 5 **TXBO**： Bus-off bit when transmitter is in a state of error
1 = Transmitter is in bus-off
0 = Transmitter is not in bus-off

bit 4 **TXBP**：Bus passive bit when transmitter is in a state of error
1 = Transmitter is in bus-off
0 = Transmitter is not in bus-off
bit 3 **RXBP**：Bus passive bit when receiver is in a state of error
1 = Transmitter is in bus-off
0 = Transmitter is not in bus-off
bit 2 **TXWAR**：Warning bit when receiver is in a state of error
1 = Transmitter is in error warning
0 = Transmitter is not in error warning
bit 1 **RXWAR**：Warning bit when receiver is in a state of error
1 = Transmitter is in error warning
0 = Transmitter is not in error warning
bit 0 **EWARN**：Warning bit when transmitter or receiver is in a state of error
1 = Transmitter or receiver is in error warning
0 = Transmitter or receiver is not in error warning

4.3.8 CINTF2 Multi-frame Message Error Register

R/C-0	R/C-0	U-0	U-0	U-0	U-0	U-0	R-0
OVFLW	TIMEOUT	-	-	-	-	-	MTXF
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Definition when PTYPE=000/001 ISO15765

U-0	R/C-0	R/C-0	U-0	R/C-0	R/C-0	R/C-0	R-0
-	TIMEOUT	SENDERR	-	ABT	SNERR	LOSSDT	MTXF
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Definition when PTYPE=010 J1939

U-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	U-0	R-0
-	TIMEOUT	SENDERR	CSNOTSUPP	NOTREADY	SNERR	-	MTXF
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Definition when PTYPE=011 VW TP2.0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **OVFLW**：Time overflow error flag
1 = When ISO15765 (PTYPE=000/001) transmits multi-frame, receiver overflows error
0 = Normal transmission
bit 6 **TIMEOUT**：Time overflow error flag
1 = When J1939 (PTYPE=010) transmits multi-frame, after transmitting RTS, wait for receiver CTS time overflow
1 = When TP2.0 (PTYPE=011) transmits multi-frame, after transmitting DT, wait for receiver response time overflow
1 = When ISO15765 (PTYPE=000/001) transmits multi-frame, after transmitting FF frame, wait for receiver FC frame time overflow

0 = Normal transmission

bit 5 **SENDERR**：Multi-frame transmit error flag

In J1939 (PTYPE=010) transmit RTS frame

1 = Transmit error occurs, generally caused by hardware

0 = Normal transmission

In TP2.0 (PTYPE=011) transmit Channel/Connection Setup /Data frame,

1 = Transmit error occurs, generally caused by hardware

0 = Normal transmission

bit 4 **CSNOTSUPP**：Receive module unsupported flag

When TP2.0 (PTYPE=011) processes Channel Setup/Connection setup,

1 = Receiver not responds

0 = Normal

bit 3 **NOTREADY/ABT**：Receive module unprepared receive flag

When **ABT** J1939 (PTYPE=010) receives multi-frame data, receive ABT

1 =In multiple transmission of multi-frame data, sender receives ABT, multi-frame transmission aborts

0 = Normal multi-frame transmission

NOTREADY When TP2.0 (PTYPE=011) transmits multi-frame data,

1 = When multi-frame transmission finishes, receiver responds to the last frame, but data reception is not possible for the time being. User must transmit the last frame data after a delay of 100ms. (Note: When TP2.0 automatically transmits multi-frame, if receiver cannot receives in the middle of multi-frame transmission, ET7190 module automatically delays CTP_Tw, after that transmits the next frame with no NOTREADY flag generated.)

0 = Normal multi-frame transmission

bit 2 **SNERR**：Frame number error flag

When J1939 (PTYPE=010) transmits multi-frame data, receives CTS response

1 =When multi-frame transmits, the received CTS frame number SN errors, exceeding the length

0 = CTS frame number SN has no error

0 =CTS

When TP2.0 (PTYPE=011) transmits multi-frame data, receive the response

When 1 = Multi-frame transmission, the received response frame number SN errors

0 = Frame number SN has no error

bit 1 **LOSSDT**：When J1939 receives multi-frame, data incomplete flag

1 = In multi-frame reception, time overflows, all DT data frames not received

0 = In multi-frame reception, normal reception

bit 0 **MTXF**：Multi-frame transmit data buffer state

1 = When multi-frame transmit data buffer is full, in TP2.0 protocol, Channel Setup instruction that transmits CTXB5 will be also set ON

0 = Multi-frame transmit data buffer is null

4.3.9 CINTE: Error Interrupt Enable Register

	U-0					U-0	U-0
IVRIE	-	ERRIE	PROEIE	FIFOIE	RBOVIE	-	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **IVRIE**：Receive invalid message interrupt enable bit
1 = Enable interrupt request
0 = Disable interrupt request
bit 5 **ERRIE**：Error interrupt enable bit (Multiple interrupt sources in CINTF1 register)
1 = Enable interrupt request
0 = Disable interrupt request
bit 4 **PROEIE**： Multi-frame message error interrupt enable bit (Multiple interrupt sources in CINTF2 register)
1 = Enable interrupt request
0 = Disable interrupt request
bit 3 **FIFOIE**：FIFO almost full interrupt enable bit
1 = Enable interrupt request
0 = Disable interrupt request
bit 2 **RBOVIE**：Receive buffer overflow interrupt enable bit
1 = Enable interrupt request
0 = Disable interrupt request

4.3.10 CTERRCNT/ CRERRCNT: Error Counter

CTERRCNT Transmit error counter

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
CTERRCNT<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
CRERRCNT<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CRERRCNT Receive error counter

4.3.11 CCFG 0/1/2: Baudrate Configuration Register

CCFG0 Baudrate Configuration Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SJW<1:0>		BRP<5:0>					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7-6 **SJW<1:0>： Synchronization Jump Width bits**
11 = Length = 4 x T_Q
10 = Length = 3 x T_Q
01 = Length = 2 x T_Q
00 = Length = 1 x T_Q

bit 5-0 **BRP<5:0>**： Baudrate Prescaler bits
11 1111 = T_Q = 2 x 64 x 1/F_{CAN}
. . .
00 0010 = T_A = 2 x 3 x 1/F_{CAN}
00 0001 = T_A = 2 x 2 x 1/F_{CAN}
00 0000 = T_Q = 2 x 1 x 1/F_{CAN}

CCFG1 Baudrate Configuration Register

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **SEG2PHTS**： PS2 Time Select bit
1 = Programmable
0 = The greater of SEG1PH bit max. value and information processing time (IPT)

bit 6 **SAM**： CAN bus line sample bit
1 = Bus line is sampled three times at the sample point
0 = Bus line is sampled once at the sample point

bit 5-3 **SEG1PH<2:0>**： PS1 length bits
111 = Length = 8 x T_Q
. . .
000 = Length =1 x T_Q

bit 2-0 **PRSEG<2:0>**： Propagation Segment Length bits
111 = Length = 8 x T_Q
. . .
000 = Length =1 x T_Q

CCFG2 Buadrate Configuration Register

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
-	-	-	-	-	SEG2PH<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 1-0 **SEG2PH<2:0>**： PS2 bits
111 = Length = 8 x T_Q
. . .
000 = Length =1 x T_Q

4.3.12 CFEN0/1 Receive Filter Enable Register

R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFEN0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFEN1

bit 7-0 **FLTENn**： Enable filter n bits (n=0-15)
1 = Enable filter n to receive message
0 = Disable filter n

4.3.13 CFMSKSEL0/1/2/3 Mask Select Register

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFMSKSEL0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFMSKSEL1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
F11MSK<1:0>		F10MSK<1:0>		F9MSK<1:0>		F8MSK<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFMSKSEL2

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
F15MSK<1:0>		F14MSK<1:0>		F13MSK<1:0>		F12MSK<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CFMSKSEL3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

F15MSK<1:0>： Filter 15 mask source bits
11 = No mask, (when ID must be identical before receiving)

10 =Receive mast register 2 includes mask value
01 = Receive mast register 1 includes mask value
00 = Receive mast register 0 includes mask value

F14MSK<1:0>： Filter 6 mask source bits (identical to F15MASK value)
…
F2MSK<1:0>： Filter 2 mask source bits (identical to F15MASK value)
F1MSK<1:0>： Filter 1 mask source bits (identical to F15MASK value)
F0MSK<1:0>： Filter 0 mask source bits (identical to F15MASK value)

4.3.14 CTXPRI 0/1 Transmitter Priority Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX3PRI<1:0>		TX2PRI<1:0>		TX1PRI<1:0>		TX0PRI<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CTXPRI0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX7PRI<1:0>		TX6PRI<1:0>		TX5PRI<1:0>		TX4PRI<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

CTXPRI1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

TXmPRI<1:0>： Message transmit priority bit m = 0-7 as CAN transmitter number
11 = Highest Message Priority
10 = High Intermediate Message Priority
01 = Low Intermediate Message Priority
00 = Lowest Message Priority

4.3.15 CTXREQ：Message Transmit Request Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX7REQ	TX6REQ	TX5REQ	TX4REQ	TX3REQ	TX2REQ	TX1REQ	TX0REQ
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

TXmREQ：Message transmit request bit, (m is transmit buffer 0-7)
1 =Request to send message. This bit is automatically cleared when the message is successfully sent
0 =Current message is successfully sent, no action will be generated if write in 0

4.3.16 CTXABT：Message Transmit Abort Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TX7ABT	TX6ABT	TX5ABT	TX4ABT	TX3ABT	TX2ABT	TX1ABT	TX0ABT
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

TXmABT：Message terminate transmit request bit, (m is transmit buffer 0-7)
1 = Message transmission will be terminated if write in 1
0 = Message transmission is successfully completed. No action is generated if write in 0, current register will not be cleared, this bit is automatically cleared when TXmREQ(CTXREQ) is set ON

4.3.17 CTXAUTOEN：Direct Transmit Enable Register

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

ENm：Direct transmit enable bits, (m is transmit buffer 0-7)

1 = When data is written into transmit area CTXBm, TXmREQ <CTXREQ> is set ON for direct transmission
0 = When data is written into transmit area CTXBm, transmission will not be started.

4.3.18 RTREN：Automatic Remote Transmit Enable Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

ENm：Automatic remote transmit enable bits, (m is the corresponding transmit buffer 0-7)
1 = When remote transmission is received, TXmREQ <CTXREQ> is set ON to start transmission.
0 = When remote transmission is received, TXmREQ <CTXREQ> is not affected

4.3.19 CPROCON Multi-frame Transreceive Control Register

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0
EN	OP1E	OP0E	-	-	PTYPE		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **EN**：Enable multi-frame protocol control
1 = Enable multi-frame transreceive automatic response

0 = Disable multi-frame transreceive automatic response

bit 6 **OP1E**：Option 1
When PTYPE=000/001 Standard ISO15765
1 = C15765RXFC1 Register set FC is valid
0 = C15765RXFC1 Register set FC is disabled
When PTYPE=010 Standard SAE J1939
1 = When RTS frame is received, CTS1 is enabled to automatically respond
0 = When RTS frame is received, CTS1 is disabled to automatically respond
When PTYPE=011 TP2.0
OP1E value is irrevelant

bit 5 **OP0E**：Option 0
When PTYPE=000/001 Standard ISO15765
1 = C15765RXFC0 Register set FC is valid
0 = C15765RXFC0 Register set FC is disabled

When PTYPE=010 Standard J1939
1 = When RTS frame is received, CTS0 is enabled to automatically respond
0 = When RTS frame is received, CTS0 is disabled to automatically respond
When PTYPE=011 TP2.0
OP1E value is irrevelant

bit 2-0 **PTYPE**：Multi-frame data handling ISO
1xx =Reserved, unavailable
011 =As per VW TP2.0
010 =As per Standard J1939
001 =As per ISO15765 Consolidated Addr.
000 =As per ISO15765 Standard Addr.

4.3.20 CCFDL Multi-frame Transmit Buffer Data length

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CCFDL<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

4.4 CAN Module Buffers (Register Set)

Name (Buffers)	Addr.	Length	Description	Reset Value
CRXMASK0	0x09	4	CAN Receive Filter Mask Standard Identifier Register 0	xx
CRXMASK1	0x0A	4	CAN Receive Filter Mask Standard Identifier Register 1	xx
CRXMASK2	0x0B	4	CAN Receive Filter Mask Standard Identifier Register 2	xx
CRXF0	0x0C	4	CAN Receive Filter Identifier Register 0	xx
CRXF1	0x0D	4	CAN Receive Filter Identifier Register 1	xx
CRXF2	0x0E	4	CAN Receive Filter Identifier Register 2	xx
CRXF3	0x0F	4	CAN Receive Filter Identifier Register 3	xx
CRXF4	0x10	4	CAN Receive Filter Identifier Register 4	xx
CRXF5	0x11	4	CAN Receive Filter Identifier Register 5	xx
CRXF6	0x12	4	CAN Receive Filter Identifier Register 6	xx
CRXF7	0x13	4	CAN Receive Filter Identifier Register 7	xx
CRXF8	0x14	4	CAN Receive Filter Identifier Register 8	xx
CRXF9	0x15	4	CAN Receive Filter Identifier Register 9	xx
CRXF10	0x16	4	CAN Receive Filter Identifier Register 10	xx
CRXF11	0x17	4	CAN Receive Filter Identifier Register 11	xx
CRXF12	0x18	4	CAN Receive Filter Identifier Register 12	xx
CRXF13	0x19	4	CAN Receive Filter Identifier Register 13	xx
CRXF14	0x1A	4	CAN Receive Filter Identifier Register 14	xx
CRXF15	0x1B	4	CAN Receive Filter Identifier Register 15	xx
CFPNT0	0x1C	4	Filter **0 – 7 Buffer Pointer Register**	0xFF FF FF FF
CFPNT1	0x1D	4	Filter **8-15 Buffer Pointer Register**	0xFF FF FF FF
CTXB0	0x1E	16	CAN Transmit Buffer 0	xx
CTXB1	0x1F	16	CAN Transmit Buffer 1	xx
CTXB2	0X20	16	CAN Transmit Buffer 2	xx
CTXB3	0X21	16	CAN Transmit Buffer 3	xx
CTXB4	0X22	16	CAN Transmit Buffer 4	xx
CTXB5	0X23	16	CAN Transmit Buffer 5	xx
CTXB6	0X24	16	CAN Transmit Buffer 6	xx
CTXB7	0X25	16	CAN Transmit Buffer 7	xx
C15765TXFC	0X26	4	ISO15765 Multi-frame Transmit Control Register Set	xx
C15765RXFC0	0X27	4	ISO15765 Receive Automatic Response FC Control Register Set 0	xx
C15765RXFC1	0X28	4	ISO15765 Receive Automatic Response FC Control Register Set 1	xx
CTP20CTRL	0X29	4	TP2.0 Multi-frame Transmit/Automatic Response Control	xx
CJ1939CTRL	0x34	4	J1939 Multi-frame Transmit/Receive RTS/CTS Control	xx
CRXB	0x3E	16*16	(Read) CAN Receive Buffer, FIFO Depth 16 Frames*16 Bytes	xx
CFTXB		256	(Write) Multi-frame Transmit Buffer, Length: 256 Bytes, write using CMD_WR_LB command	xx

4.4.1 CRXMSKn：CAN Receive Filter Mask Identifier Register Set n (n = 0-2)

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	-	MIDE	-	EID17	EID16
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE 2

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE 3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

SID<10:0>： Standard Identifier bit
1 =Filer compare operation includes SIDx bit
0 = Filer compare operation is irrelevant to SIDx bit

MIDE： Identifier Receive Mode Bit
1 =The corresponding message type of filter EXIDE bit is only matched (Standard or Extended Addr.)
0 = If filter is matched, then it matches with Standard or Extended Addr. message
(i.e.: If (Filter SID) = (Message SID) or (Filter SID/EID) = (Message SID/EID))\

EID<17:0>： Extended Identifier Bit
1 =Filer compare operation includes SIDx bit
0 = Filer compare operation is irrelevant to SIDx bit

4.4.2 CRXFn：CAN Receive Filter Identifier Register Set n(n = 0-15)

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	-	EXIDE	-	EID17	EID16
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x

EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

SID<10:0>： Standard Identifier Bit
1 = Message addr. bit SIDx must be set ON until it matches with filter
0 = Message addr. bit SIDx must be set Clear until it matches with filter
EXIDE： Extended Identifier Enable Bit
If MIDE = 1：
1 = Match only with message with extended identifier addr.
0 = Match only with message with standard identifier addr.
If MIDE = 0: Ignore EXIDE bit.
EID<17:0>： Extended Identifier Bit
1 = Message addr. bit EIDx must be set ON until it matches with filter
0 = Message addr. bit EIDx must be set Clear until it matches with filter

4.4.3 CFPNT0：Filter 0-7 Buffer Pointer Register Set

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F1BP<3:0>				F0BP<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F3BP<3:0>				F2BP<3:0>			
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F5BP<3:0>				F4BP<3:0>			
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F7BP<3:0>				F6BP<3:0>			
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

bit 31-28 **F7BP<3:0>**： Filter 7 Receiver Buffer Mask Bit
1111 = Data satisfying the filter condition will be received into receive FIFO buffer
1000-1110: Reserved, unavailable
0111 = Data satisfying the filter condition will be transmitted into transmit buffer 7 and used for automatic remote transmit
....
0001 = Data satisfying the filter condition will be transmitted into transmit buffer 1 and used for automatic remote transmit
0000 = Data satisfying the filter condition will be transmitted into transmit buffer 0 and used for automatic remote transmit

Bit 11-8 **F2BP<3:0>**： Filter 2 Receive Buffer Mask Bit (Identifical to bit 31-28 value)

bit 7-4 **F1BP<3:0>**： Filter 2 Receive Buffer Mask Bit (Identifical to bit 31-28 value)

bit 3-0 **F0BP<3:0>**： Filter 0 Receive Buffer Mask Bit (Identifical to bit 31-28 value)

4.4.4 CFPNT1：Filter 8-15 Buffer Pointer Register Set

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F9BP<3:0>				F8BP<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F11BP<3:0>				F10BP<3:0>			
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F13BP<3:0>				F12BP<3:0>			
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
F15BP<3:0>				F14BP<3:0>			
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

bit 31-28 **F15BP<3:0>**：Filter 7 Receive Buffer Mask Bit

1111 = Data satisfying the filter condition will be received into receive FIFO buffer
1000-1110: Reserved, unavailable
0111 = Data satisfying the filter condition will be transmitted into transmit buffer 7 and used for automatic remote transmit
....
0001 = Data satisfying the filter condition will be transmitted into transmit buffer 1 and used for automatic remote transmit
0000 = Data satisfying the filter condition will be transmitted into transmit buffer 0 and used for automatic remote transmit

Bit 11-8 **F10BP<3:0>**：Filter 2 Receive Buffer Mask Bit (Identical to bit 31-28 value)

bit 7-4 **F9BP<3:0>**：Filter 1 Receive Buffer Mask Bit (Identical to bit 31-28 value)

bit 3-0 **F8BP<3:0>**：Filter 0 Receive Buffer Mask Bit (Identical to bit 31-28 value)

4.4.5 CTXBn: Transmit Buffer n = 0-7

See Section 5.5, Message Transreceive Buffers

4.4.6 C15765TXFC：ISO15765 Multi-frame Message Transmit Control Register Set

Function of this control register set: Control in transmission of multi-frame message

R/W-x	U-x	U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x
LENM	-	-	-	FILTER<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
STM	BSM	BSIZE<5:0>					
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
N_AETX<7:0>							
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
N_AERX<7:0>							
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

bit 31-24 **N_AERX<7:0>**：Extended Addr. in Consolidated Addr. Mode, the received FC DATA0 must be in this value

bit 23-16 **N_AETX<7:0>**：Extended Addr. in Consolidated Addr. Mode, DATA0 value when transmitter transmits CF

bit 15 **STM**：ST Time Mode in transmission
1 = ST enforces CN_ST-defined interframe space
0 = ST automatically transmits the received FC interframe space

bit 14 **BSM**： BS Mode in transmission
1 = BS enforces the transmission by the number of BSIZE frames
0 = BS automatically handles the received FC by 15765-2

Bit 13-8 **BSIZE<5:0>**：BS (block size) in multi-frame transmission
BSM = 0 Irrelevant to BSIZE value,
BSM = 1 Enforce the transmission by BSIZE, when BSIZE must not be 0

bit 7 **LENM**：Length Mode, related to the transmission of last frame
1 = In transmission, CAN fixed length is 8 bytes, if the last frame is less than 8 bytes, it is transmitted in 8 bytes, with the subsequent byte value of CFTXB buffers. In multi-frame transmission, Data length exactly the same with integer frame must be introduced.
0 = In transmission, the last frame is transmitted by actual length

bit 3-0 **FILTER<3:0>**：Filter code 0-15 used when FC is received

4.4.7 C15765RXFC0：ISO15765 Receive Automatic Response FC Control Register Set

Function of this control register set, when FF set attached in this register set is received (the first frame of multi-frame message), ET7190 automatically transmits the preset FC (control frame) in CTXB6.

U-x	U-x	U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	-	FILTER<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BSIZE<7:0>							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
N_AE<7:0>							
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

U-x	U-x	U-x	U-x	U-x	U-x	U-x	U-x
-	-	-	-	-	-	-	-
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

bit 23-16 **N_AE<7:0>**：Extended Addr. in Consolidated Addr. Mode, the received FF AE must be in this value

Bit 15-8 **BSIZE<7:0>**：BS when multi-frame is received

0 = FC is transmitted only once, sender must transmit CF until all message are completed

1-255 = Counted by BSIZE, the next FC is transmitted when CF set in BSIZE is received

bit 3-0 **FILTER<3:0>**：Filter code 0-15 used when FC is received

4.4.8 C15765RXFC1：ISO15765 Receive Automatic Response FC Control Register Set 1

Function of this control register set, when FF set attached in this register set is received (the first frame of multi-frame message), ET7190 automatically transmits the preset FC (control frame) in CTXB7. Content is the same with above: C15765RXFC0.

4.4.9 CTP20CTRL：TP2.0 Multi-frame Transmit/Automatic Response Control Register Set

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CTSFLTN<3:0>				CHSFLTN<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ACK	STM	IDM	BSM	BSIZ<3:0>			
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	MNT<1:0>		MNTC<3:0>			
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

U-x	U-x	U-x	U-x	U-x	U-x	U-x	U-x
-	-	-	-	-	-	-	-
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"

-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

bit 21-20 **MNT<1:0>**：Number of repeats when Connection Seup fails, generally set 3

bit 19-16 **MNTC<3:0>**：Number of repeats when Channel Seup fails, generally set 10

Bit 15 **ACK**： DT Frame Automatic Response

0 =Automatic response when DT frame required to response is received

1 =No response when DT frame required to response is received

Bit 14 **STM**： Transmit Interframe T3 Time Mode

0 = T3 is automatically set by Connection setup message

1 =Enfored by CTP_T3(T3A)

Bit 13 **IDM**： ID Setting

0 = ID is automatically set by Channel setup respond frame (CTXB6 and CTSFLTN)
1 = ID not changed

Bit 12 **BSM：** BS Consecutive Transmit Mode
0 =BS is automatically set by Connection setup respond frame
1 = Enforced by BSIZ

Bit 11-8 **BSIZ<3:0>：** BS when multi-frame is transmitted, with the value as 1-F, but must not be 0

bit 7-4 **CTSFLTN<3:0>：** Filter code 0-15 when Connection Setup is done/normally connected

bit 3-0 **CHSFLTN<3:0>：** Filter code 0-15 when Connection Setup is done

4.4.10 CJ1939CTRL：J1939 Multi-frame Transmit/Receive RTS/CTS Control Register Set

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
R1FLTN<3:0>				R0FLTN<3:0>			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BSIZE<7:0>							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8

BYTE 1

U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	BAM	BSM	TXFLTN<3:0>			
Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16

BYTE2

U-x	U-x	U-x	U-x	U-x	U-x	U-x	U-x
-	-	-	-	-	-	-	-
Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24

BYTE3

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: This buffer can be handled as 32-bit integer <Bits 31: 0> if applied for the purpose of more convenience

Bit 21 **BAM：** BS Consecutive Transmit Mode
0 = Transmit multi-frame by RTS/CTS
1 = Transmit by BAM

Bit 20 **BSM：** Set BS mode when RTS is received
0 = BS is automatically set
1 = BS is enforced by BSIZ

bit 19-16 **TXFLTN<3:0>：** ID Filter code 0-15 used by CTS when multi-frame is received

Bit 15-8 **BSIZ<7:0>：** Maximum BS set by CTS when RTS is received, with the value as 1-F, but must not be 0

BSM=0 If RTS frame BS is greater than BSIZ, then CTS frame sets BS as BSIZ
If RTS frame BS is no greater than BSIZ, then CTS frame BS is RTS BS.
BSM=1 CTS frame sets BS always as BSIZ

bit 7-4 **R1FLTN<3:0>：** ID Filter code 0-15 used when RTSI is received

bit 3-0 **R0FLTN<3:0>：** ID Filter code 0-15 used when RTS0 is received

4.5 CAN Message Transreceive Buffer Block

The size of ET7190 message transreceive buffer block is 16 bytes. Its 16-byte block structure is identical in transmit and receive buffer. Transmit buffer block contains 8 buffers, namely CTXB0-CTXB7, receive buffer block CRXB is 16 frames*16 FIFO receive buffer block. Writing in CRXB doesn't mean writing receive buffer block, but CAN long data buffer block CFTXB when the transmission of multi-frame is written, ET7190 only maps to the same address, actual read and write are in two different buffer blocks.

4.5.1 Table of Transmit/Receive Buffer Block

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	SID5	SID4	SID3	SID2	SID1	SID0	**SRR**	**IDE**
1	-	-	-	SID10	SID9	SID8	SID7	SID6
2	EID13	EID12	EID11	EID10	EID9	EID8	EID7	EID6
3	-	-	-	-	EID17	EID16	EID15	EID14
4	-	-	-	**RB0**	**DLC<3:0>**			
5	EID5	EID4	EID3	EID2	EID1	EID0	**RTR**	**RB1**
6	DATA0							
7	DATA1							
8	DATA2							
9	DATA3							
10	DATA4							
11	DATA5							
12	DATA6							
13	DATA7							
14	-	-	-	-	-	-	-	-
15	-	-	-	**FILHIT<4:0>**				

SID<10:0>：Standard Identifier Bits

SRR： Replace Remote Request Bits
1 = Message will request remote transmission
0 = Normal message

IDE： Extended Identifier Bits
1 = Message will transmit extended identifier
0 = Message will transmit standard identifier

EID<17:6>：Extended Identifier Bits

EID<5:0>：Extended Identifier Bits

RTR： Remote Transmit Request Bits
1 = Message will request remote transmit
0 = Normal message

RB1： The held bit 1 user must set this bit ON as per CAN protocol.

RB0： The held bit 0 user must set this bit Clear as per CAN protocol.

DLC<3:0>：Data length Code Bits

DATA7:DATA0：CAN Message Bytes Data[7:0]

FILHIT<4:0>：The Selected Filter Code bits

The filter number written into this buffer will be coded (block only executes the writing for receive buffer, which will not be used for transmit buffer). User may determine the filter that hits data according to the received value of data frame.

4.5.2 Byte Definition of Buffers

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID5	SID4	SID3	SID2	SID1	SID0	SRR	IDE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 0

U-x	U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	SID10	SID9	SID8	SID7	SID6
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 1

SID<10:0>：Standard Identifier Bits

SRR： Replace Remote Request Bits

1 = Message will request remote transmission

0 = Normal message

IDE： Extended Identifier Bits

1 = Message will transmit extended identifier

0 = Message will transmit standard identifier

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID13	EID12	EID11	EID10	EID9	EID8	EID7	EID6
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 2

U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	-	EID17	EID16	EID15	EID14
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 3

EID<17:6>： Extended Identifier Bits

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	RB0	DLC3	DLC2	DLC1	DLC0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 4

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID5	EID4	EID3	EID2	EID1	EID0	RTR	RB1
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 5

EID<5:0>：Extended Identifier Bits

RTR： Remote Transmit Request Bits

1 = Message will request remote transmission

0 = Normal message

RB1： The held bit 1 user must set this bit clear as per CAN protocol.
RB0： The held bit 0 user must set this bit clear as per CAN protocol.
DLC<3:0>：Data Length Code Bits

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 0							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 6

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 1							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 7

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 2							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 3							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 9

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 4							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 10

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 5							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 11

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 6							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 12

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte of Data 7							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 13

CAN Message Byte Data[7:0]

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
-	-	-	-	-	-	-	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 14

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	-	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Message Buffer Byte 15

FILHIT<4:0>：The selected filter code bits

The filter number written into this buffer will be coded (block only executes the writing for receive buffer, which will not be used for transmit buffer). User may determine the filter that hits data according to the received value of data frame.

4.6 CAN Module Mode of Operation

User application program may select one of the following modes for CAN module. Including:

- Configuration Mode
- Normal Mode
- Listen Mode
- Listen All Message Mode
- Loopback Mode
- Disable Mode

User application program requests the desired mode by writing in the request operation mode bit REQOP<2:0> (CCTRL0<2:0>) of CAN control register 0. CAN acknowledges the access to requested mode by operation mode bit OPMODE<2:0> (CCTRL0<6:4>). Mode switch is simultaneously executed with CAN network. This means CAN controller will not change mode until it detects the bus idle sequence (11 recessive bits).

4.6.1 Configuration Mode

After hardware is reset, CAN module will be in Configuration mode(OPMODE<2:0> = 100). Error counter will be cleared, and all registers contain the reset value. CAN bit time control register(CCFG0/CCFG1 /CCFG2) is required to be modified, when CAN module must be in Configuration Mode.

4.6.2 Normal Mode

In Normal Mode, CAN module can transmit and receive CAN message. Normal Mode is requested by setting REQOP<2:0> bit (CCTRL0<2:0>) as 000 after the initialization. When OPMODE<2:0> = 000, this module will start the normal operation.

4.6.3 Listen Mode

Listen Mode is mainly used for bus monitoring rather than being involved in transmit process. Any node in Listen Mode will not generate response frame or error frame – but other nodes must execute this operation. Listen Mode can be used to detect the Baudrate in CAN bus.

4.6.4 Listen All Message Mode

Listen All Message Mode will be used for system debugging. Basically, all messages will be received whatever the message identifier is, even if error may exist. If Listen All Message Mode is activated, transmit or receive operation is identical to that in Normal Mode, but it will be transmitted to message buffer only when the error message is received.

4.6.5 Loopback Mode

Loopback Mode is used for self-check, permitting CAN module to receive its own message. In this mode, CAN transmit path will be internally connected to transmit path. And in this mode, “false” response will not be provided, consequently no other node is required to provide the response bit.

4.6.6 Disable Mode

Disable Mode is used to ensure safe closure before the device is set to Sleep Mode. In other words, CAN controller will not change mode until it detects the bus idle sequence(11 recessive bits). When the module is in Disable Mode, it will stop the clock, which has no effect on CPU or any other module. In case of bus activity or

CPU sets OPMODE<2:0> as 000, the module will be awakened. In Disable Mode, CANxTX will remain in recessive state.

4.7 Transmit CAN Message

This message transmitter is the node of message transmission. This node can maintain as the transmitter until the bus becomes idle or the arbitration fails. Transmission is started by writing data into message transmit buffer CTXB0-7 and set TXmREQ(CTXREQ) on. If the automatic start transmit bit related to ENm(CTXAUTOEN) is set on, then transmission is started when data is written into transmit buffer. In CAN protocol, please refer to BOSCH CAN protocols for the composition of standard frame and extended frame in message buffer, and the state of IDE, SRR, RTR, RB0 and RB1 bit of standard remote frame or extended remote frame.

4.7.1 Message Transmit Flow

User application program must execute the following tasks if message must be transmitted via CAN bus:

- Priority of the designated transmit buffer(if required)
- CAN message written into message transmit buffer CTXB0-7

The send request of buffer TXmREQ bit is set ON to start the message transmission. If ENm(CTXAUTOEN) related auto start transmit bit is set ON, then data is transmitted when transmit buffer writes data.

Message transmission is started by setting CAN transmit control register message transmit request bit TXmREQ(CTXREQ). If ENm(CTXAUTOEN) related auto start transmit bit is 1, TXmREQ is automatically set ON when writing message transmit buffer CTXBm, after message transmission is completed, TXmREQ bit will be automatically cleared. Before CAN sends SOF(Start-of-frame), the module will check all buffers prepared for transmission, in order to determine which buffer has the highest priority. Transmit buffer with the highest priority will be firstly transmitted.

Each transmit message buffer can indicate any of 4 user application program defined priorities by using CTXPRI0 CTXPRI1 TXmPRI bit, and all reset values are Priority 0.

TXmPRI<1:0> Message Transmit Priority：

11 = Highest Message Priority
10 = High Intermediate Message Priority
01 = Low Intermediate Message Priority
00 = Lowest Message Priority

For message buffer with the designated same priority defined by user application program, natural order priority exists. The natural order priority of transmit message buffer 7 is the highest. The priority defined by user application program will cover the natural order priority.

4.7.2 Data Frame Transmitted Code Example

Firstly, we assume that C-language structure and function is defined via SPI interface connected to ET7190. (For more details, please refer to Demoboard Demonstration Program)

Feature	SPI Mode - Operation that writes ET7190 Register
Parameter	Input: Register Val： Written Value

Function	void SpiWriteRegister(BYTE Register, BYTE Val)；

Feature	SPI Mode - Register set(buffer) that writes length byte(2-16)
Parameter	RgBuff:Buffer Addr. pData: Pointer written into data buffer DataLen: Data Length（<=16）
Function	void SpiWriteRegisterBuff(BYTE RegBuff,BYTE DataLen,BYTE * pData);

Feature	Transmit/Receive Buffer CAN Message Data Structure

```
typedef union{
   struct{
           unsigned int IDE          : 1；
           unsigned int SRR          : 1；
           unsigned int SID10_0      : 11；
           unsigned int              : 3；
           unsigned int EID17_6      : 12;
     unsigned int                 : 4;
     unsigned int DLC             : 4;
           unsigned int RB0          : 1；
           unsigned int               : 3;
     unsigned int RB1             : 1;
     unsigned int RTR             : 1;
     unsigned int EID5_0          : 6;
     unsigned char Data[8];
     unsigned char RESBYTE;
     unsigned char FILHIT;
        };
    WORD _word[8];
    BYTE  _byte[16];
}CANPACKET;
```

A. Code Example of Standard Frame Transmit

```
CANPACKET   canMsgBuf;                          //Define a CAN buffer variable
SpiWriteRegister(CTXAUTOEN,0x01);               //Set CTXAUTOEN：Transmission is automatically started
when CTXBO is written

canMsgBuf._word[0]     = 0;                     //RTR  RB0 RB1 SRR IDE  all  clr  to  0
canMsgBuf._word[1]     = 0;
canMsgBuf._word[2]     = 0;

//canMsgBuf.IDE              =  0b0;             //IDE=0;
//canMsgBuf.SRR              =  0b0;             //SRR=0
canMsgBuf.SID10_0          =  0x7DF;           //SID=7DF
canMsgBuf.DLC               =  8;              //Data Length
canMsgBuf.Data[0]= 0x10;                       //Data Field
canMsgBuf.Data[1]= 0x11;
canMsgBuf.Data[2]= 0x12;
canMsgBuf.Data[3]= 0x13;
canMsgBuf.Data[4]= 0x14;
canMsgBuf.Data[5]= 0x15;
canMsgBuf.Data[6]= 0x16;
canMsgBuf.Data[7]= 0x17;

SpiWriteRegisterBuff(CTXB0 ,  16 ,   canMsgBuf._byte);       //Write CTXBO, automatically start the transmission
```

B. Code Example of Extended Data Frame Transmit

```
CANPACKET   canMsgBuf;                          //Defined a CAN buffer variable
SpiWriteRegister(CTXAUTOEN,0x00);               //Set CTXAUTOEN：Transmission is not automatically started
when CTXB0 is written

canMsgBuf._word[0]     = 0;                     //RTR=0  RB0=0 RB1=0 SRR=0   IDE=0  all  clr
canMsgBuf._word[1]     = 0;
canMsgBuf._word[2]     = 0;

canMsgBuf.IDE              =  0b1;                  //IDE=1;  Extended Frame
//canMsgBuf.SRR            =  0b0;                  //SRR=0
canMsgBuf.SID10_0          =  (0x18DA33F1>>18);     //SID10_0  = 29BIT High 11 bits
canMsgBuf.EID5_0           =   ( 0x18DA33F1&0x3F);  //EID5_0    = 29BIT  Lowest 6 bits
canMsgBuf.EID17_6          =  (0x18DA33F1>>6);      // EID17_6  = 29BIT  Intermediate 12 bits
canMsgBuf.DLC               =  8;                   //Data Length
canMsgBuf.Data[0]= 0x10;                            //Data Field
canMsgBuf.Data[1]= 0x11;
canMsgBuf.Data[2]= 0x12;
canMsgBuf.Data[3]= 0x13;
canMsgBuf.Data[4]= 0x14;
canMsgBuf.Data[5]= 0x15;
canMsgBuf.Data[6]= 0x16;
canMsgBuf.Data[7]= 0x17;

SpiWriteRegisterBuff(CTXB0 ,  16 ,   canMsgBuf._byte);       //Write in CTXB0
SpiWriteRegister(CTXREQ,0x01);                               //TX0REQ=1 Start transmission
```

4.7.3 Abort Transmit Message

ABAT in CAN control register CTRL1 is set ON, which can request to abort all messages to be transmitted. To abort the specific message, message abort request bit TmABT(CTXABT) in connection with this message

buffer must be set ON. In these two cases, message will be aborted only before CAN module starts the message transmission in the bus.

4.7.4 Remote Transmit Request/Response

A. Remote Transmit Request

The node that desires to receive data frame with specific identifier can start another node to transmit the intended data via transmit remote frame. Remote frame may be in either standard or extended format.

Except the items below, remote frame is similar to data frame:

• RTR bit is recessive(RTR = 1)

• No data field(DLC = Irrelevant to length value)

To transmit remote frame, user application program must execute the following tasks:

• Write the remote frame into the message buffer accordingly. The transmitted identifier must be the same with the identifier of received data frame

• Set the buffer transmit request bit ON to start the transmission of remote frame

B. Remote Transmit Response

The node that serves as the remote frame request response source needs to configure the receive filter, in order to match with the identifier of remote request frame. Message transmit buffer 0-7 can respond to the remote request, so receive filter buffer pointer (FnBP) should point at one of 8 message buffers. CAN automatic remote transmit enable(ENm in RTREN) bit must be set ON, ENm in automatic enable transmit CTXAUTOEN must be set CLEAR, in order to automatically respond to the remote request frame.

When the module receives remote frame, TXmREQ in transmit buffer pointed by FnBP is automatically set ON to start the transmission.

This is the only case that a receive filter buffer pointer(FnBP) points to the transmitted message buffer. In normal case, all FnBPs point to FIFO. (FnBP=0b1111)

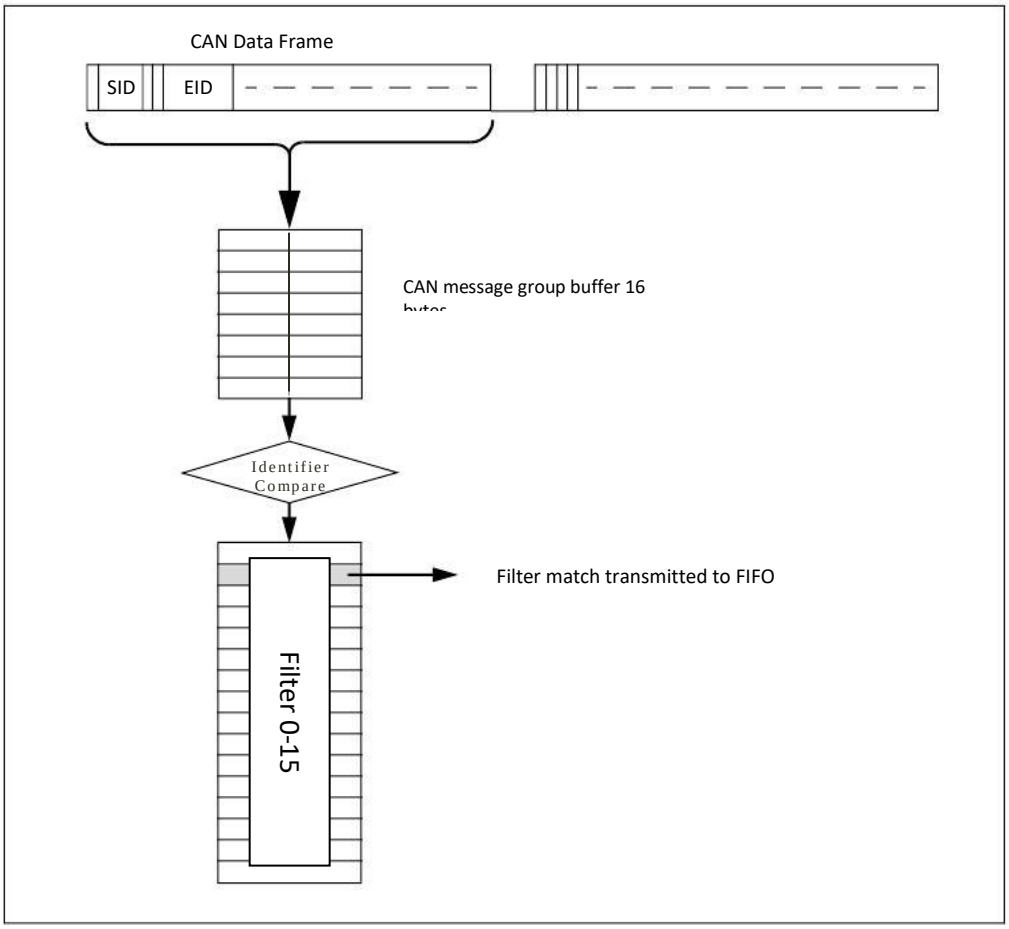
4.8 Receive CAN Message

CAN module can receive standard frame and extended frame in CAN bus nodes. The storage format of each received message in buffer is the same with that of transmit buffer, or each message occupies the space of 16 bytes. The next section will introduce two main stages about CAN receive process. Fig.4-2 illustrates the example of simplified receive process.

4.8.1 Message Receive and Receive Filter

Fig. 4-2 shows that each message transmitted in the bus is received into message buffer set, and its identifier field will be compared with receive filter defined by a group of users, 16 in total. Each received standard data frame contains a 11-bit standard identifier(SID), while each extended data frame contains a 11-bit SID and a 18-bit extended identifier(EID). If all bits transmitted to identifier are fully matched with the bit in any receive filter, then CAN module will receive the message into the receive buffer FIFO accordingly.

Fig.4-2:



4.8.2 Receive Filter

Receive filter CRXFn(n=0-15) can use the enable filter bit FLTENn in CAN receive filter enable register CFEN0 CFEN1 to enable or disable. “n” means register bits corresponding to the number of receive filter. Receive filter 0-15 is used to designate some identifier, its content will be transmitted to FIFO receive buffer only when received message contains the identifier. In which, each filter CRXFn(n=0-15) contains the standard and extended identifier.

4.8.3 Receive Filter Mask

Receive filter selects one of receive filter masks via the selected mask source select(FnMSK<1:0>) mask select bit in CFMSKSEL0/ CFMSKSEL1/ CFMSKSEL2/ CFMSKSEL3 register.

- CFMSKSEL0<FnMSK>：Filter 0-3 Mask Select
- CFMSKSEL2<FnMSK>：Filter 4-7 Mask Select
- CFMSKSEL2<FnMSK>：Filter 8-11 Mask Select
- CFMSKSEL3<FnMSK>：Filter 12-15 Mask Select

The selective values of FnMSK<1:0> include:

- 00 Select Receive Filter Mask CRXMSK0
- 01 Select Receive Filter Mask CRXMSK1

- 10 Select Receive Filter Mask CRXMSK2
- 11 No Select Receive Filter Mask, when it is equivalent to CRXMAK=0xFFFFFFFF

The module compares each bit in identifier with mask and filter to acknowledge whether to accept or reject message. Mask bit mainly decides which bits will apply the filter. If any mask bit is set clear, this bit will be automatically accepted whatever the value of filter bit is.
The following condition is understood that message has been successfully accepted:
SID_EID is CAN identifier transmitted in the bus, either n=0-15 or m = 0 - 3

(SID_EID & CRXMSKm) Compared with (CRXFn & CRXMSKm)

4.8.4 Message Type Select

The extended identifier enable bit EXIDE in CAN receive filter CRXFn register set is used to accept enable standard identifier message or extended identifier message. If CRXMSKn<MIDE> bit is set ON, then only the message type selected by CRXFn<EXIDE> bit will be received. If CRXMSKn<MIDE> is set CLEAR, then CRXFn<EXIDE> bit will be neglected, and all messages matched with the filter will be accepted.

EXIDE	**MIDE**	Select
0	1	Receive Filter Only Check Standard Identifier
1	1	Receive Filter Check Extended Identifier
x	0	Receive Filter Check Standard/Extended Identifier

4.8.5 Receive Filter Configuration

The filter setting in OBD2 standard diagnosis mode as indicated in example 6-1, 6-2 and message ID definition in OBD2 standard diagnosis mode are defined as follows:

OBD2 Diagnosed 11-bit ID Definition (ISO15765-4)

ID	Tester Request	ID	ECU Respond
7DF	External tester executes functional request for all ECUs		
7E0	Request the physical address for ECU1(Engine ECM)	7E8	ECU1 responds to the physical address of tester(Engine ECM)
7E1	Request the physical address for ECU2(Transmission TCM)	7E9	ECU2 responds to the physical address of tester(Transmission TCM)
7E2	Request the physical address for ECU3	7EA	EU3 responds to the physical address of tester
7E3	Request the physical address for ECU4	7EB	ECU4 responds to the physical address of tester
7E4	Request the physical address for ECU5	7EC	ECU5 responds to the physical address of tester
7E5	Request the physical address for ECU6	7ED	ECU6 responds to the physical address of tester
7E6	Request the physical address for ECU7	7EE	ECU7 responds to the physical address of tester
7E7	Request the physical address for ECU8	7EF	ECU8 responds to the physical address of tester

OBD2 Diagnosed 29-bit ID Definition(ISO15765-4)

ID	Description
18 DB 33 F1	External tester executes the request of diagnosis message for all ECUs
18 DA xx F1	External tester sends the diagnosis request to ECU with the physical address xx
18 DA F1 xx	ECU #xx responds message to tester

The ECU module address xx can be referred to: SAE 32178/1

Example 6-1: Code Example of Filter 11-bit Standard Data Frame
The code of configuration receive filter 0 is given, which configures it as the receive standard identifier 7E8-7EF message, and the receive filter mask register is used to mask SID<2:0> bit. Tester Functional Addr. Request ID: 7DF, ECU response ID: The ID in which the maximum 8 ECUs may simultaneously respond is: 0b111 1111 1xxx (7E8←→7EF)

```
CANPACKET   canMsgBuf;                          //Define a CAN transmit buffer variable
UINT32  uiID;

SpiWriteRegister(CFEN0,0x01);                  //Enable Receive Filter 0   bit0=1
SpiWriteRegister(CFMSKSEL0,0x00);               // Filter CRXF0 uses CRXMSK0 mask
uiID=(0x7F8<<18);                               //ID corresponds to SID10:0
SpiSetMask(CRXMSK0, uiID,1);                   //Set CRXMSK0 mask 7F8     SID<2:0>=0, no compare, MIDE=1
uiID=(0x7E8<<18);                               //ID corresponds to SID10:0
SpiSetFilter(CRXF0,uiID,0);                    //Set CRXF0 filter  SID<2:0>=0, no compare, so 7E8-7EF value is
irrelevant
                                                //EXIDE=0; only receive standard frame

SpiWriteRegister(CCTRL0,0x00);                 //Start CAN controller

SpiWriteRegister(CTXAUTOEN,0x01);              //Set CTXAUTOEN：  Transmission is automatically started
when CTXB0 is written

canMsgBuf._word[0]     = 0;                    //RTR  RB0 RB1 SRR IDE   SID EID   all  clr  to  0
canMsgBuf._word[1]     = 0;
canMsgBuf._word[2]     = 0;

//canMsgBuf.IDE              =  0b0;           //IDE=0;          11-bit Standard Frame
//canMsgBuf.SRR              =  0b0;           //SRR=0
canMsgBuf.SID10_0          =  0x7DF;          //SID=7DF
canMsgBuf.DLC               =  8;             // Data Length
canMsgBuf.Data[0]= 0x02;                      //Data Field PCI(SF Frame)
canMsgBuf.Data[1]= 0x01;                      //MOD
canMsgBuf.Data[2]= 0x00;                      //PID
canMsgBuf.Data[3]= 0x00;
canMsgBuf.Data[4]= 0x00;
canMsgBuf.Data[5]= 0x00;
canMsgBuf.Data[6]= 0x00;
canMsgBuf.Data[7]= 0x00;

SpiWriteRegisterBuff(CTXB0 ,  16 ,   canMsgBuf._byte);       //Write CTXB0, automatically start transmission
MOD_PID：0100 Request
```

Example 6-2: Code Example of Filter 29-bit Extended Data Frame
The code of configuration receive filter 0 is given, which configures it as the receive extended identifier 18 DA F1 xx message, and the accept filter mask register 0 is used. Tester Functional Addr. Request ID: 18 DB 33 F1, ECU response ID: 18 DA F1 xx

```
CANPACKET   canMsgBuf;                          //Define a CAN transmit buffer variable

SpiWriteRegister(CFEN0,0x01);                   //Enable Receive Filter 0   bit0=1
SpiWriteRegister(CFMSKSEL0,0x00);               // Filter CRXF0 uses CRXMSK0 mask
SpiSetMask(CRXMSK0,0x1FFFFF00,1);               //Set CRXMSK0 mask     EID<7:0>=0,no compare,MIDE=1
SpiSetFilter(CRXF0,0x18DAF100,1);               //Set CRXF0 filter    EXIDE=1; Extended frame is only received

SpiWriteRegister(CCTRL0,0x00);                  //Start CAN controller

SpiWriteRegister(CTXAUTOEN,0x01);               //Set CTXAUTOEN：Transmission is automatically started when
CTXB0 is written

canMsgBuf._word[0]     = 0;                     //RTR  RB0 RB1 SRR IDE   SID EID   all  clr  to  0
canMsgBuf._word[1]     = 0;
canMsgBuf._word[2]     = 0;

canMsgBuf.IDE               = 0b1;                          //IDE=1;          29-bit Extended Frame
canMsgBuf.SID10_0           = (0x18DB33F1>>18);             //SID
canMsgBuf.EID5_0            = (0x18DB33F1);                 //EID5_0    C coder can execute the automatic
truncation at high bits
canMsgBuf.SID17_6           = (0x18DB33F1>>6);              //EID17_6
canMsgBuf.DLC               = 8;                //Data Length, valid length is 3, Filled with 0
canMsgBuf.Data[0]= 0x02;                        //Data Field PCI (SF Frame) Command Length 2
canMsgBuf.Data[1]= 0x01;                        //MOD
canMsgBuf.Data[2]= 0x00;                        //PID
canMsgBuf.Data[3]= 0x00;
canMsgBuf.Data[4]= 0x00;
canMsgBuf.Data[5]= 0x00;
canMsgBuf.Data[6]= 0x00;
canMsgBuf.Data[7]= 0x00;

SpiWriteRegisterBuff(CTXB0 ,  16 ,   canMsgBuf._byte);      //Start CTXB0,automatically start transmission
MOD_PID：0100 Request
```

Note: User application program can configure several receive filters to implement the message multiple buffers by using the same value. In this case, the received message may match with several filters, ECAN module will assign the message to FIFO buffer, match filter with the smallest number. **FILHIT<4:0>** in the receive buffer is the match filter with the smallest number.

4.8.6 Message Received to Transmit Buffer CTXB 0-7

The receive filter buffer pointer(FnBP) normally points to FIFO receive buffer, which can be configured to point to message transmit buffer, provided that this transmit buffer CRTREN.ENm bit must be set ON, and if message is detected by the receive filter of this message buffer, then this message buffer will handle RTR rather than message storage. This is the only case that receive filter buffer pointer(FnBP) points to or configures the transmitted message buffer.

4.8.7 DeviceNet™ Filter

DeviceNet filter function is based on CAN 2.0A protocol, where 18-bit data field to the maximum will be compared with message receive filter EID in addition to SID. DeviceNet function is enabled or disabled via DeviceNet filter bit code control bit DNCNT<4:0> in CAN control register CDNCNT. IN DNCNT bit field, the designated value determines the number of data bit to be compared with message receive filter EID bit. If DNCNT<4:0> bit is set clear, DeviceNet will be disabled. 11-bit SID must be matched with SID<10:0> bit in the message receive filter and the first "n" data in message must be matched with EID<17:0> bit in the message receive filter before message is received.

Table 6-1 shows the filter compared via CiCTRL2 (DNCNT<4:0>) control bit configuration. For example, if DNCNT<4:0> = 00011, only 11-bit standard identifier is matched with SID receive filter(SID<10:0>), when the message with bit 7, bit 6 and bit 5 of data byte 0 matched with the extended identifier filter(EID<17:15>) will be received.

Table 6-1: DeviceNet™ Filter Bit Configuration

DeviceNet™ Filter Configuration **(DNCNT<4:0>)**	Receive Message Data Bit to be Compared (Byte <Bit>)	**EID Bit Used for Receive Filter**
00000	No compare	No compare
00001	Byte of Data 0<7>	EID<17>
00010	Byte of Data 0<7:6>	EID<17:16>
00011	Byte of Data 0<7:5>	EID<17:15>
00100	Byte of Data 0<7:4>	EID<17:14>
00101	Byte of Data 0<7:3>	EID<17:13>
00110	Byte of Data 0<7:2>	EID<17:12>
00111	Byte of Data 0<7:1>	EID<17:11>
01000	Byte of Data 0<7:0>	EID<17:10>
01001	Byte of Data 0<7:0> and Byte of Data 1<7>	EID<17:9>
01010	Byte of Data 0<7:0> and Byte of Data 1<7:6>	EID<17:8>
01011	Byte of Data 0<7:0> and Byte of Data 1<7:5>	EID<17:7>
01100	Byte of Data 0<7:0> and Byte of Data 1<7:4>	EID<17:6>
01101	Byte of Data 0<7:0> and Byte of Data 1<7:3>	EID<17:5>
01110	Byte of Data 0<7:0> and Byte of Data 1<7:2>	EID<17:4>
01111	Byte of Data 0<7:0> and Byte of Data 1<7:1>	EID<17:3>
10000	Byte of Data 0<7:0> and Byte of Data 1<7:0>	EID<17:2>
10001	Byte 0<7:0>, Byte 1<7:0> and Byte 2<7>	EID<17:1>
10010	Byte 0<7:0>, Byte 1<7:0> and Byte 2<7:6>	EID<17:0>
10011-11111	Invalid selection	Invalid selection

Special Cases

Some special cases may exist, where the number of data bits contained in message is less than the required number of bits in DeviceNet filter configuration.

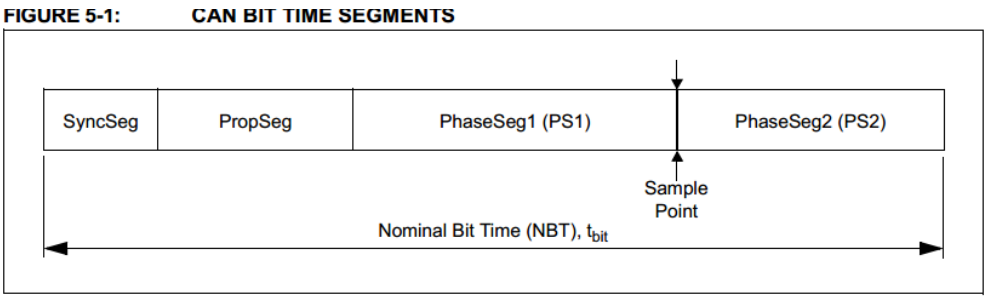
- Case 1 ——If DNCNT <4:0> is greater than 18, the selected number of bits by indicated user application program is greater than total number of EID bits, filter comparison will end at Bit 18 of data (Bit 6 of data byte 2). If SID matches with totally 18 data bits, message will be received.

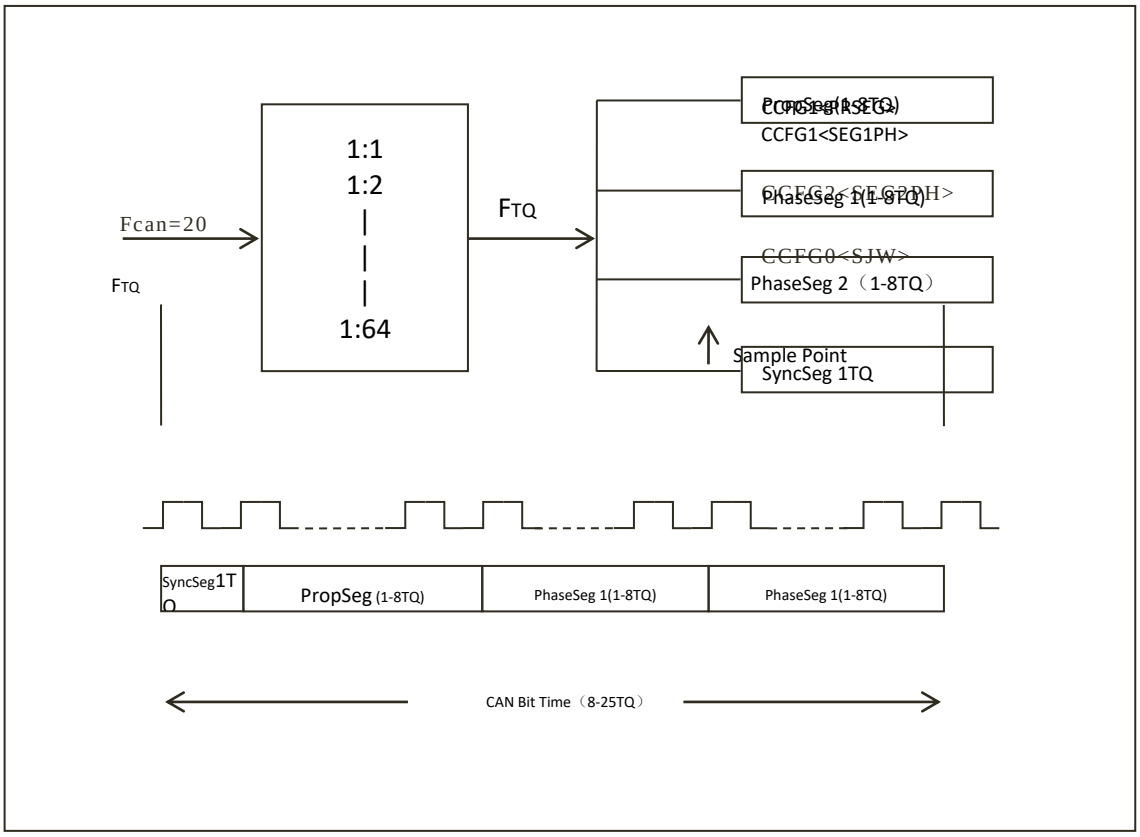
- Case 2 ——If DNCNT<4:0> is greater than 16, and receive message data length code(DLC) is 2(the valid load is indicated as 2 bytes of data), filter comparison will end at Bit 16 of data (Bit 0 of Data Byte 1). If SID matches with totally 16 data bits, message will be received.
- Case 3 ——If DNCNT<4:0> is greater than 8, and receive message DLC=1(the valid load is indicated as 1 byte of data), filter comparison will end at Bit 8 of data(Bit 0 of Data Byte 0). If SID matches with totally 8 data bits, message will be received.
- Case 4 ——If DNCNT<4:0> is greater than 0, and receive message DLC=0(no valid load of data is indicated), filter comparison will end at standard identifier. If SID is matched, message will be received.

4.9 CAN Bit Timing and Baudrate Setting

Nominal bit rate means the number of bits transmitted in CAN bus line. Nominal Bit Time = 1 ÷ Nominal bit rate There are 4 time segments per bit time to compensate any phase shift resulting from oscillator drift or propagation time delay. These time segments are not overlapped one another, and expressed in TQ(Time Quantum). One TQ is the fixed unit of time generated by oscillator clock. Total TQ of nominal bit time must be set between 8~25 TQ. Fig. 4-3 shows how the time quantum clock(FTQ) is obtained via system clock Fcan(20MHZ), and how to set different time segments.

Fig.4-3: CAN™ Bit Timing





4.9.1.1 Bit Time Segments

Each bit transmit time consists of 4 time segments.

- SyncSeg——This time segment is used to synchronize different nodes connected with CAN bus line. Bit edge is required to be in this time segment. SyncSeg is assumed as one TQ as per CAN protocol.
- ProgSeg——This time segment is used to compensate any time delay which may be caused by bus line or various transreceiver connected with this bus line.
- PhaseSeg 1——This time segment is used to compensate any possible error resulting from phase shift of edge. This segment may be extended in the period of resynchronization to compensate phase shift.
- PhaseSeg 2——This time segment is used to compensate any possible error resulting from phase shift of edge. This segment may be shortened in the period of resynchronization to compensate phase shift. PhaseSeg 2 time can be configured as programmable or designated by PhaseSeg 1 time.

4.9.1.2 Sample Point

Sample point means the time point when sample is obtained from CAN bit time interval and bus state is read and interpreted. This is between PhaseSeg 1 and PhaseSeg 2. CAN bus line can be sampled once or for three times at sample point, which is configured by sample CAN bus line bit SAM(CCFG1<6>) in CAN Baudrate configuration register.

- If CCFG1<SAM> = 1, CAN bus must be sampled at sample point for three times. Bit value will be determined by the majority of sample value appeared in three times of sampling.
- If CCFG1<SAM> = 0, CAN bus is only sampled at sample point once.

4.9.1.3 Synchronization

There are two mechanisms used for synchronization – Hard synchronization and resynchronization. Hard synchronization is processed once at start of frame. Resynchronization is processed in interframe.

- Hard synchronization is processed from recessive to dominant. Bit time restartes from this edge.
- Resynchronization is processed when some bit edge doesn't appear within the SyncSeg of message. One TQ of PhaseSeg will be shortened or extended according to phase errors in signal. The available maximum TQ is determined by SJW parameters(CCFG0<SJW>).
- The length of PhaseSeg1 and PhaseSeg 2 can be modified according to oscillator tolerance of transmit and receive nodes. Resynchronization can compensate any possible phase shift at transmit and receive nodes using different oscillators.
- Bit Extend ——If the oscillator rate of transmit nodes in CAN is slower than that of receive nodes, the next falling edge can be delayed by extending PhaseSeg 1 in bit time, thereby delaying the sample point.

- Bit Shorten——If the oscillator rate of transmit nodes in CAN is faster than that of receive nodes, the next falling edge can be delayed by shortening PhaseSeg 2 in bit time, thereby advancing the sample point.
- Synchronization Jump Width (SJW) ——SJW <1:0> bits (CCFG0<7:6>) in CAN Baudrate configuration register can control the extended or shortened TQ applied to time intervals between PhaseSeg 1 and PhaseSeg 2 to determine SJW. The time in this segment must not be longer than PhaseSeg 2 time. Its width can be 1-4 TQ.

4.9.1.4 Calculation of CAN Bit Time

Step 1: Calculate the frequency of TQ(F_{TQ})

If CAN bus Baudrate is F_{BAUD}, then $F_{TQ}=N*F_{BAUD}$(N must be 8-25)

Fcan is 20MHZ, F_{TQ} and Fcan must be in interger multiple, select the appropriate N TQ.

Step 2: Calculate the Baudrate Prescaller(CCFG0<BRP>) by the following formula:

CCFG0(BRP) = (20M/ F_{TQ}) – 1

Step 3: Select each bit time segment

Each bit time segment is selected using CCFG1/CCFG2 register.

Bit Time = SyncSeg + PropSeg + PhaseSeg 1 + PhaseSeg 2

Notes
1. (PropSeg + PhaseSeg 1) must be greater or equal to the length of PhaseSeg 2.
2. PhaseSeg 2 must be greater than SJW.

e.g.: CAN Bit Timing Calculation Example

Step 1: Calculate FTQ.

- If F_{BAUD} = 1 Mbps, and N(the number of TQ) = 20, then F_{TQ} = 20 MHz.

Step 2: Calculate Baudrate Prescaler(BRP).

-CCFG0<BRP> = (20000000/ FTQ) - 1 = 1 - 1 = 0。

Step 3: Select each bit time segment (20TQ in total).

- SyncSeg = 1TQ(constant).
- Based on system features, propagation delay is assumed to be 5 TQ.
- Sample point is assumed to be at 70% of nominal bit time. PhaseSeg 2 = 30% of nominal bit time = 6 TQ.
- Thus, PhaseSeg 1 = 20 TQ - (1 TQ + 5 TQ + 6 TQ) = 8 TQ.

4.10 CAN Error Management

4.10.1 CAN Bus Errors

CAN protocol has defined 5 different ways to detect error.

- Bit Error
- Response Error
- Format Error
- Filling Error
- CRC Error

Bit error and response error appear in bit level; while other three errors are found in message level.

4.10.1.1 Bit Error

The node to send a bit in bus line is accompanied with bus monitoring. When the monitored bit value is different from the transmitted bit value, it indicates that bit errors are detected; but there is one exception, recessive bits are transmitted in period of Filling bit flow of arbitration field or ACK time slot. In this case, bit errors will not appear when dominant bits are monitored. The transmitter that sends passive error frame and detects the recessive bits will not interpret this case as bit error.

4.10.1.2 Response Error

In message response field, transmitter will check if response time slot (transmitter sends it in recessive bit) contains a dominant bit or not. If not contained, it means that other nodes have not correctly received this frame. This indicates that response errors occur, so message must be repeatly transmitted. In this case no error frame is generated.

4.10.1.3 Format Error

When the bit field(EOF, interframe gap, response delimiter or CRC delimiter) in fixed format contains one or more illegal bits, it indicates that format error is detected. For receiver, the dominant bit in the last bit period after frame ends must not be deemed as format error.

4.10.1.4 Filling Error

In message field which should be coded as per Filling method, the bit time of the sixth consecutive equal bit level will detect the Filling error.

4.10.1.5 CRC Error

The node of transmit message needs to calculate and send CRC corresponding to the transmitted message. Each receiver in the bus line will execute the same CRC calculation with transmitter. If the calculated result is different from CRC value obtained from receive message, it indicates that CRC error

4.10.2 Error Limit

Each CAN controller in bus line will attempt to detect the said error in each message. If any error is identified, the node that such error is found will send an error frame, thus breaking bus communication. Other nodes will detect the error caused by error frame (if the original error is not yet detected) and execute the operation (i.e.: discard the current message). CAN module will maintain two error counters:

• Transmit Error Counter (CTERRCNT)

• Receive Error Counter (CRERRCNT)

There are some rules controlling incremental and/or decremental mode of these counters. In essence, the speed that transmitter with error detected increments the transmit error counter is faster than that of monitor node increments the receive error counter. This is because that there is a higher error chance for transmitter.

The node is in error active mode at the start. When either reading in two error counters is equal to or greater than 127, this node enters into the so-called state of “Error Passive”. When the reading of transmit error counter is greater than 255, this node enters into bus-off state.
• Error active node will transmit active error frame when an error is detected.
• Error passive node will transmit passive error frame when an error is detected.
• The node in bus-off will not transmit any data in bus line.

4.10.2.1 Transmitter in Error Passive

When the transmit error counter is equal to or greater than 128, transmitter error passive(CINTF1<TXBP>) bit will be set ON, and error interrupt(CINTF0<ERRIF>) is generated when it enters in error passive. If the transmit error counter is changed to be smaller than 128, hardware will automatically clear the transmit error passive flag.

4.10.2.2 Receiver in Error Passive

When the receive error counter is equal to or greater than 128, receiver error passive(CINTF1<RXBP>) bit will be set ON, and error interrupt(CINTF0<ERRIF>) is generated when it enters in error passive. If the receive error counter is changed to be smaller than 128, hardware will automatically clear the receive error passive flag.

4.10.2.3 Transmitter in Bus-off

When the transmit error counter is equal to or greater than 256, transmitter bus-off (CINTF1<TXBO>) will be set ON, and an error interrupt(CINTF0<ERRIF>) is generated.

4.10.2.4 Transmitter in Error Warning

When the transmit error counter is equal to or greater than 96, transmitter error passive(CINTF1<TXWAR>) bit will be set ON, and error interrupt(CINTF0<ERRIF>) is generated when it enters in error warning. If the transmit error counter is changed to be smaller than 96, hardware will automatically clear the transmit error passive flag(CINTF1<TXWAR>).

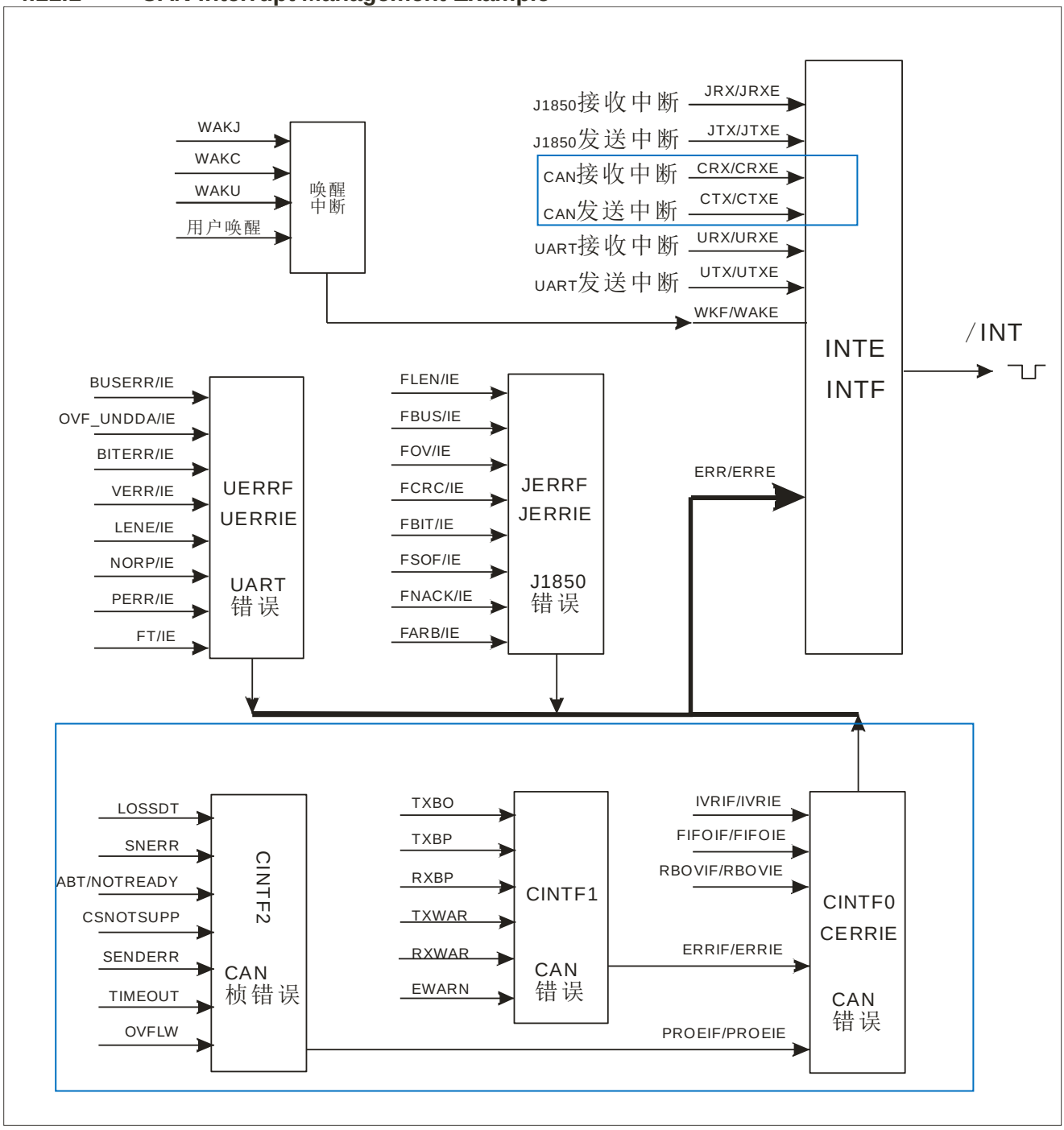
4.10.2.5 Receiver in Error Warning

When the receive error counter is equal to or greater than 96, receiver error passive(CINTF1<RXWAR>) bit will be set ON, and error interrupt(CINTF0<ERRIF>) is generated when it enters in error warning. If the receive error counter is changed to be smaller than 96, hardware will automatically clear the receive error passive flag. In addition, there is an error warning flag(CINTF1<EWARN>) bit, if one or more error counter is equal to or greater than error warning limit 96, this bit will be set ON. If the reading in either two error counters is smaller than error warning limit, EWARN will be reset.

4.11 CAN Interrupt

ECAN event interrupt contains multiple interrupt sources, and each of these interrupt sources can be idependently enabled. Interrupt flag(CINTF0/CINTF1/INTF) register contains interrupt flag, and interrupt enable(CINTE/INTIE) register contains interrupt enable bit.
Among 5 error interrupt sources(transmit error warning, receive error warning, transmit error passive, receive error passive and transmit bus-off), either one can set the error interrupt flag as 1<ERRIF>. Interrupt source of error interrupt is determined by reading CINTF1 register. All frame errors CINTF2 can set the frame error interrupt flag as 1 (CINTF0<PROEIF>), where interrupt source is read via CINTF2.
N event interrupt examples of ET7190 various interrupt sources are given in the figure.

4.11.1 CAN Interrupt Management Example



4.11.2 Transmit Buffer Interrupt

Message buffers 0-7 configured for message transmission will generate the transmit buffer interrupt(INTF<CTX>) after CAN message is transmitted. User must clear the transmit buffer interrupt by clearing and reading INTF bits.

4.11.3 Receive Buffer Interrupt

When message is successfully received and loaded in a receive buffer(FIFO), INTF<CRX> bit is set ON in module, and transmit buffer interrupt is removed by clearing and rading INTF bits.

4.11.4 Receive Buffer Overflow Interrupt

When message is successfully received, but FIFO buffer is fully occupied, CINTF0<RBOVIF> bit is set ON in module, and buffer receive interrupt is removed by clearing and reading FIFO receive buffer JRXBUFF.

4.11.5 FIFO Almost Full Interrupt Flag bit

When only an available buffer is left in FIFO, FIFO interrupt(CINTF0<FIFOIF>) will be activated when the module loads data in the last available buffer. FIFO almost full interrupt must be cleared by setting CINTF0<FIFOIF> bit Clear.

4.11.6 Error Interrupt

Error interrupt (CINT0F<ERRIF>) may be generated from 5 sources.

- Transmit Error Warning
- Receive Error Warning
- Transmit Error Passive
- Receive Error Passive
- Transmit Bus-off

Error interrupt must be removed by clearing CINTF0<ERRIF> bit via interrupt service program.

4.11.7 Wake-up Interrupt

In Sleep mode, the device will detect the bus activity by monitoring CAN receive pin(CANxRX), CAN1RX or CAN2RX is selected depending on CPINSEL. When bus activity is detected, wake-up (INTF<WAF>) interrupt will be generated. The source of wake-up interrupt is determined by reading SLEEP register. INTF<WKF> bit must be cleared by reading INTF via interrupt service program.

4.11.8 Receive Invalid Message Interrupt

For the error of any other type in a period of message reception, receive invalid message interrupt will be generated. IVRIF interrupt must be removed by clearing CINTF0<IVRIF> bit.

4.11.9 Multi-frame Handling Error Interrupt

For the automatic response when long message is automatically transmitted in segments or multi-frame message is received by ET7190, if any error under relevant standard is generated, it will be flagged PROEIF, and interrupt source is determined by reading INTF2. Interrupt must be cancelled by clearing 0 <INTF2>. Please refer to the description of relevant protocols in Chapter 7 for more details.

Chapter 5 CAN Module Procol Application

5.1 ISO16765 Application

ISO 15765, as a vehicle network protocol based on CAN bus line, has been extensively applied in vehicle network diagnosis. ISO 15765-2 precisely controls the communication time of network layer, and a comprehensive diagnostic service is provided for ISO14299-1, ISO15031-5 application layer.

5.1.1 ISO15765 Message Format

Message(frame) in 4 formats defined in ISO15765 is listed in the following table:

PDU Name Network Protocol Data Unit	N_PCI bytes(Network Protocol Control Information)				Remark
	BYTE 1(Byte of Data Field)		BYTE2	BYTE3	
	Bits 7-4	Bits3-0			
SF (Single Frame)	N_PCItype=0	SF_DL	N/A	N/A	Single-frame message
FF (First Frame)	N_PCItype=1	FF_DL		N/A	First frame in multi-frame message
CF (Consecutive Frame)	N_PCItype=2	SN	N/A	N/A	Consecutive frame in multi-frame message
FC (FlowControl)	N_PCItype=3	FS	BS	STmin	Control frame in multi-frame message

- All message transmitted under ISO15765 mut be one of these 4 formats. CAN frame length DLC is generally fixed as 8, while the insufficient part is Filled by 0. (Note: Don't mix CAN frame length DLC and SF_DL. CAN frame DLC parameters are set by transmitter, once accepted by receiver, the number of data bytes transmitted to network layer is determined. DLC parameters cannot determine the length of information, which is extracted from the opening N_PCI information.) OBD2 diagnosis applies Standard Addr. Mode.

- If the length of transmitted message is equal to or smaller than 7 bytes (6 bytes in Extended Addr. Mode), single frame is transmitted in SF format to complete the data transmission. If the data length of message is greater than 7 bytes(6 bytes in Extended Addr. Mode), meesage is segmented into multiple frames to be transmitted in FF/CF/FC format.

- Higher-order 4 Bits<7:4> in the first byte of CAN frame data field is N_PCItype distinguish frame format, which is valued at 0-3, representing SF/FF/CF/FC respectively. Other values are disabled.

- **SF (Single Frame):** N_PCItype=0 <Bits7:4>

  SF_DL<3:0> Data length of SF frame,
  Only the first byte in SF is PCI byte, and other bytes are transmitted data.
  SF_DL is valued at 1-7(1-6 in Extended Addr. Mode), and other values are disabled.
- **FF(First Frame):** N_PCItype=1 <Bits7:4>

FF_DL<11:0> In long message frame-to-frame transmission, FF is firstly transmitted, Lower-order 4 Bits(Bits3:0) in FF and the 2nd byte constitute 12-bit length FF_DL. The valid value of FF_DL must be 8-FFF(7-FFF in Extended Addr. Mode). Other values are invalid. ISO15765 enabled maximum data length of long message is 0xFFF(4095) bytes. In FF, PCI occupies 2 bytes, and others are valid bytes of data.

- **CF(Consecutive Frame):** N_PCItype=2 <Bits7:4>, PCI occupies 1 byte, and others are valid bytes of data.

  SN<3:0> Frame serial parameters 0-F.

  SN is used to indicate the sequence of consecutive frame. SN segment of long message starts counting from 0. FF is assigned as 0, no value in FF is indicated 0. But one message only has a FF, which represents serial 0. SN of the first CF must be 1, and SN circulates in 0-F when each CF SN+1 is transmitted. SN value must not be affected by FC.

- **FC(Frame Control):** N_PCItype=3 <Bits7:4>, PCI occupies 3 bytes, and other bytes are invalid.

  FS<3:0>Receiver Flow Status indication, indicating whether receiver is able to continue receiving data or not.

  FS=0： Enable sender to continue sending

  FS=1：Wait(WT) Transmission is not started until sender waits the next FC with FS=0.

  FS=2：Receiver data overflow, unable to continue receiving. Sender stops sending, error message is returned.

  FS=3~F: Disabled.

  BS：Definition of block size parameters, absolute number of CF that receiver enables sender to consecutively transmit, in segments of data transmission, the number of CF of the last block might be smaller than BS value.

  00：Represent sender transmits all frames of long message in one time, during which receiver will not send FC.

  1-FF：After sender transmits one block data in BS number, the subsequent data blocks must not be transmitted until receiver transmits one CF.

  STmin：Minimum time delay that sender is enabled to consecutively transmit CF interframe.

  00-7F：0-127ms

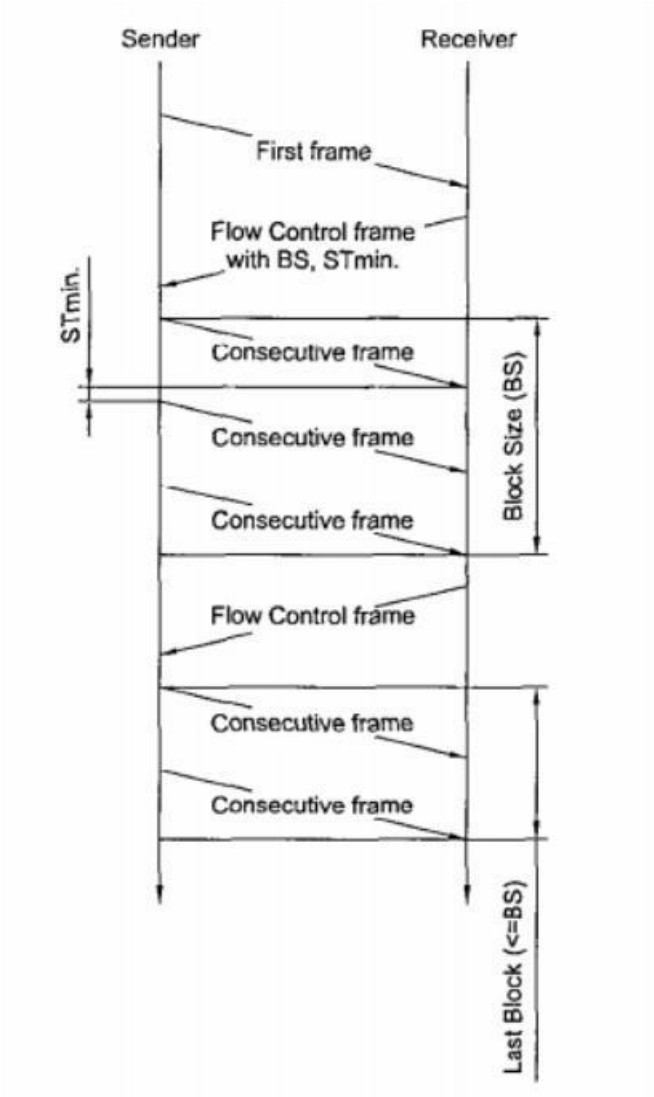
  80-F0：NA

  F1-F9：100us-900us

  FA-FF：NA

Note: Please refer to ISO15765-2 network protocol for detailed information.

5.1.2 Long Message Frame-to-Frame Transmission Process



5.1.3 Registers(Set) Related to ISO15765

- CINTF2：Multi-frame Transmit Flag Register
- CPROCON：Multi-frame Transreceive Protocol Control Register
- CCFDL：Multi-frame Transmit Buffer Data Length
- C15765TXFC：Multi-frame Transmit Control Register Set(4 bytes)
- C15765RXFC0：0# Multi-frame Receive Auto-respond FCO Register Set(4 bytes)
- C15765RXFC1：1# Multi-frame Receive Auto-respond FC1 Register Set(4 bytes)

- CN_BS： N_BS Time xx*5ms
- CN_ST： N_ST Time xx*100us
- CN_BR： N_BR Time xx*100us
- CN_CR： N_CR Time xx*5ms Not used by ET7190, no need to set

5.1.4 Transmit Buffer used under ISO15765

When ISO15765 protocol control is activated, CAN module uses 3 special transmit buffers for the transmission and reception of multi-frame message. User may use CTXB0-CTXB4 transmit buffer if SF or other data frames in no relation to C15765TXFC is transmitted.

- CTXB5： Specifically used for the transmission of FF, start long message multi-frame transmission, and module automatically generates CF.
- CTXB5： Specifically used for the transmission of C15765RXFC0 FC When multi-frame message is received.
- CTXB7： Specifically used for the transmission of C15765RXFC1 FC when multi-frame message is received.
- CFTXB: This is the buffer with 256 bytes, and the actual valid data length is 252 bytes to the maximum.

  User firstly writes the long message data(excluding data in FF) to be transmitted into CFTXB buffer, then the length of CCFDL is set(the actual length of buffer is 252 to the maximum, excluding data in FF), CAN starts the multi-frame transmission when FF is written into CTXB5, after sender receives FC responded from receiver, STmin/BS is automatically configured according to FC parameters, CF is automatically segmented to transmit against the buffer CFTXB datra, until the data transmission in CFTXB is completed.

  CCFDL must not be set greater than 252 bytes

- When C15765TXFC .LENM=1, if the data length of last frame is less than 8 bytes, the length of CAN module compel data frame DLC is 9, Fill bytes are behind the valid data in CFTXB.

  So, if C15765TXFC .LENM=1, actual data written by user into CFTXB must be the valid data(Stanard Addr. frame 7 or Extended Addr. frame 6) that CF one-frame is able to write in integer multiple, 0 is written in case of insufficient length, if it is not in integer multiple, the Filling data of the last frame may not be determined.

  If C15765TXFC .LENM=0, no Filling is needed, just write valid data into CFTXB.

- ET7190 The longest message that CAN module is able to automatically send frame-to-frame is the length of valid data in 252 + FF.

1. In Standard Addr. Mode: 252+6=258 bytes to the maximum.
FF_DL in FF transmitted by user must be equal to CCFDL+6
2. In Extended Addr. Mode: 252+5=257 bytes to the maximum.
FF_DL(total length of message) in FF transmitted by user must be equal to CCFDL+5

- If total length of message valid data exceeds the above length, user needs to handle FF/CF/FC, automatic segment of CAN module must not be used for the transmission of long message.

5.1.5 Configuration Register/Register Set/Transmit Buffer

(The following is in Standard Addr. Mode, illustrate OBD2 communication of diagnostic tester 7DF)

- A—Configure Transmit Control Register Set C15765TXFC

C15765TXFC.FILTER=0 Generally set as smaller filter number for precise match. This filter receives FC, if the frame with multiple IDs is required to receive in application, then multiple match must be set via the filter with bigger number. This filter seting is very important, CAN module only identifies FC data according to filter setting, ID value is not required to determine. So the filter must be precisely set.

C15765TXFC.BSM=0
C15765TXFC.STM=0
C15765TXFC .LENM=1 Frame length is fixed in 8 bytes
C15765TXFC.BSIZE=16 When BSM=0, value size is irrevalent, actual transmitted BS is determined by receiver's FC
N_AERX / N_AETX Standard Addr. Mode value is irrevalent

- B—Configure Receive Multi-frame Control C15765RXFC0 Response

C15765RXFC0.BSIZE=0 Generally as 0, sender sends in one time, or might be any other value
C15765RXFC0.FILTER=0 Identical to the filter number when multi-frame is transmitted. The same ID has receive and transmit
C15765RXFC0.N_AE Standard Addr. Mode value is irrevalent

- C—Configure Receive Multi-frame Control C15765RXFC1 Response

- C15765RXFC1.BSIZE=0 Generally as 0, sender sends in one time, or might be any other value

C15765RXFC1.FILTER=1 Receive the long message of another module(different IDs), Configure Filter 1
C15765RXFC1.N_AE Standard Addr. Mode value is irrelevant

- D—Control Register CPROCON

CPROCON.PTYPE=0b000 ISO15765 Standard Addr. Mode
CPROCON.OP0E =1 Enable Response FC0
CPROCON.OP1E =1 Enable Response FC1
CPROCON.EN =1 Enable Protocol

- E—Set FC Control Frame

CTXAUTOEN<EN7.EN6> must be set 0, the transmission will not be strated when data is written into CTXB6/7, the transmission of FC is automatically started when an valid FF is received.
FC0: Suppose tester communicates with the engine module, and with 11-bit ID, then ID is 7E0.
FC1: Suppose tester communicates with the transmission module, and with 11-bit ID, then ID is 7E1.
7E0 30 00 00 00 00 00 00 00: CTXB6 is written by FC0 complete FC
7E1 30 00 00 00 00 00 00 00: CTXB7 is written by FC0 complete FC

- E—Set Filter

Set Filter 0 receives 11-bit ID=7E8(engine module communicates with tester), no mask
Set Filter 1 receives 11-bit ID=7E9(transmission module communicates with tester), no mask
(No mask, ID must be fully matched before reception, if other ID data is required to receive in application, more filters may be configured)

CFEN0.EN0=1, CFEN0.EN1=1 Enable Filter 0 and 1

- F—Set Time

CN_BS=200 N_BS =200*5ms=1000ms
CN_ST=0 STmin=0*100us=0
CN_BR=10 C_BR =10*100us=1ms

5.1.6 CAN Module Long Message Automatic Frame-to-Frame Transmission Process

CAN module automatically transmits long message frame-to-frame according to C15765TXFC setting.

1. Relevant Register Configured as per 5.1.5
2. Enable CAN Module Normal Mode
3. Data Written in CFTXB Transmit Buffer
4. Data Length Written in CCFDL Transmit Buffer
5. FF Written in CTXB5 to Start Transmission
   After FF transmission is completed, CAN module waits receiver to reply FC, if no FC is received within N_BS, TIMEOUT<CINTF2> error will be generated, if interrupt enable will produce error interrupt, the transmission of long message is aborted. MTXF<CINTF2> is automatically cleared. CAN module enables receive or transmit other data in waiting for FC control.
   If FS=3-F of FC is received, CAN module is treated as invalid data, FC is continuously waited, N_BS is not recounted.
   If FS=2 of FC is received, OVFLW error will be generated, the transmission of long message is aborted. MTXF<CINTF2> is automatically cleared.
   If FS=1 of FC is received, CAN module will wait for the next FC. N_BS is recounted.
   If FS=0 of FC is received, CAN module starts loading data from CFTXB, CF is transmitted from CTXB5, and STmin is determined by FC. After BS is counted as 0, the next FC is expected to be transmitted from receiver, BS, STmin will be reset according to the new FC after the reception of FC, until the data transmission is completed.

6. Retransmission after Abort
   If the transmit error of long message is checked to abort, just follow Step 5 for retransmission, the retransmission of long message will be restarted when FF is written in CTXB5.
7. If transmission is actively aborted in the transmission of long message
   CCTRL1.CMTXF is set ON to abort the multi-frame transmission.

e.g.: 10 data transmitted to the receive module: 1,2,3,4,5,6,7,8,9,10

1. Write 7,8,9,10 in CFTXB, total length of <0,0,0,> is 7 bytes
   Because the data length is 10, FF transmits the first 6 data(1-6). For 4 data after CF transmission, if the fixed 8 bytes are transmitted when CAN is set, CF excludes PCI byte, valid data is in 7 bytes, 3 bytes need to Fill. So 3 “00” need to be written for Filling, if not, the Filling value of last CF is not determined. It is also enabled to not write as per ISO15765.
   The first 6 data are transmitted in FF, so they are not written in buffer.
2. The actual valid length in CFTXB when CCFDL=4 is written, (Filling byte is not counted)
3. Transmission is started when FF is written in CTXB5
   FF: 7E0 10 0A 01 02 03 04 05 06 (DLC=8，ID=7E0)
   In data frame, PCItype=1 FF_DL=0x00A (10)
   Actual data appeared in bus line when data in transmitted:
   TESTER →ECU: FF 7E0 10 0A **01 02 03 04 05 06**
   ECU→TESTER: FC 7E8 30 00 00 00 00 00 00 00(Sender will receive FC)
   TESTER→ECU: CF 7E0 21 **07 08 09 0A** 00 00 00

   Receiver ECU receives 2-frame data:
   FF 7E0 10 0A **01 02 03 04 05 06**
   CF 7E0 21 **07 08 09 0A** 00 00 00
   In this transmission, sender will receive one or more FCs according to concrete data. Application is generally neglected. The completion of long message transmission can be determined by reading CINTF2.MTXF status. After the transmission of long message, CTX<INTF> will be set ON.

Interrupt will be produced if interrupt is enabled. In transmission, FF and middle single frame CF CTX<INTF> will not be set ON.

5.1.7 CAN Module Automatic Response in Reception of Long Message

- If enabled, when FF is received, CAN module will automatically respond to one or more FCs according to the data length and BS value. The transmission of FC is not required at user end. User can automatically receive the complete long message of FF and one or more CF.
- Automatic response only sets 2 IDs simultaneously.
- After FC is automatically responded, if no CF data is transmitted from sender to CAN module, no error will be produced in CAN module. Whether data is complete or not, user should group it with data in CF according to the received FF_DL data length in FF.
- In whichever state of current FCx automatic response controller, the first FC response will be restarted when CAN module receives FF with the corresponding ID.
- Response Time: When FF is received, CN_BR is delayed and FC transmission is started. Or when the last CF in BS is received, if message is not completed, CN_BR will be delayed and the transmission of next FC will be started.

5.2 SAE J1939 Application

SAE J1939 protocol is vehicle network control protocol released by SAE(Society of Automotive Engineer), which uses CA N 2.0B as network core protocol based on CAN bus line. By now it has been one of CAN bus protocols extensively applied in vehicle industry, with communication rate at 250K bit/s. 29-bit extended frame format.

5.2.1 SAE J1939 Message Format

PDU(Protocol Data Unit), actually a frame with 29-bit ID and 8 data in CAN, constitutes the framework of information. No remote frame or 11-bit ID standard frame is used in J1939. As shown in Table below:

J1939 PDU						
SID10:0 EID17:0 (29 位 ID) (29-bit ID)						Data Field (CAN Data Field 8 Bytes)
P	R	DP	PF	PS(GE/DA)	SA	8 bytes
3	1	1	8	8	8	64 bits

- **P**<2:0>：Priority, 0 is the highest priority, 7 is the lowest priority.

- **R**：Reserved Bit, must be always 0

- **DP**：Data Page 1 bit
  Data Page Bit, generally Page 1 PGN is not used until Page 0 PGN is used up.
  It makes possible for further extension of PGN. In addition, it is also a field of PGN.

- **PF** (8bits): Determine the format of PDU
  PF is between 0 and 239 (0x00-0xEF), PDU format is PDU1

  PF is between 240 and 255 (0xF0-0xFF), PDU format is PDU2

- **PS** (8bits)：Specific PDU
  When PF is PDU1, this value represents —- Target Addr. DA
  When PF is PDU2, this value represents---Parameter group extension(GE), it is one field of PGN.
  Note: When PDU1, only 0xFF value is the global Target Addr.

- **SA** (8bits): Source Addr. (Physical Addr. of sender)

- Data field: 8 bytes, 64 bits

5.2.2 Definition of Parameter Group Number PGN

When the data field of CAN data frame is identified which parameter group it belongs, a 24-bit value is required. This 24-bit value firstly transmits the lowest byte (lowest byte is firstly transmitted, followed by intermediate byte, finally highest byte). PGN is a 24-bit value including: Reserved Bit, Data Page Bit, PDU Format Field(8-bit) and Group Extended Field(8-bit). These bit fields are converted to PGN by the following procedure. If PF value is smaller than 240(0xF0), PGN lowest byte is set Clear. Reserved Bit is always 0.

e.g.:

- 29-bit ID threshold is 0x18EA01F9. PF is 0xEA<0xF0, so the format is PDU1, PGN=0x00EA00 (DP=0), when PS threshold is Target Addr. DA=0x01.
- 29-bit ID threshold is 0x19FE08F9. PF is 0xFE>=0xF0, so the format is PDU2, PGN=0x01FE08, (DP=1), when PS threshold is Group Extension GE=0x08.

Several Important Parameter Group Numbers(PGN) under J1939

User-defined Mnemonic	PF Value	PGN Value	Description of Function
PF_ACKNOW	0xE8 (232)	0x00E800 (59392)	Aknowledge parameter group
PF_REQUEST	0xEA (234)	0x00EA00 (59904)	Request parameter group
The following 2 PGNs are used in long message multi-frame transmission			
PF_DT	0xEB (235)	0x00EB00 (60160)	Data message in multi-frame transmission
PF_TP_CM	0xEC (236)	0x00EC00 (60416)	Communication protocol, connection management

5.2.3 J1939 Message Types

Currently five message types are supported, which are: Command, Request, Radio/Response, Acknowledgement and Group Function. Message types are identified by the assigned PGN.

5.2.3.1 Command

Command message means those parameter groups which transmit command from one Source Addr. to specific Target Addr. or global Target Addr.. After Target Addr. receives the command message, action should be taken according to the received message. Both PDU1 format(PS is Target Addr.) and PDU2 format(PS is Group Extension) can be used as command. Command message can be used for drive control, address request, torque/speed control, etc.

5.2.3.2 Request

Request message is able to request information from global range or from specific Target Addr.. The request for specific Target Addr. is referred to as the request incremented to specific Target Addr.. PGN definition of "Request PGN" Parameter Group is given in the table below.

J1939 PDU Request PGN＝59904						
SID10:0 EID17:0 (29-bit ID)						Data Field (CAN Data Field 8 Bytes)
Default Priority P	R	DP	PF	DA	SA	Byte 1,2,3
6	0	0	0xEA	Target Addr.	Source Addr.	Requested PGN

If Target Addr. value is FF, so it is global Target Addr..

- Target Addr. must respond to the request for specific Target Addr.. If Target Addr. doesn't support the requested PGN, a NACK response must be sent to indicate that it doesn't support this PGN. Because some PGN are multiple packets, so a single-frame request response may have multiple CAN data frames.
- If this is a global request, when a node doesn't support some PGN, NACK response cannot be send. Request PGN can be directly sent to the specific Target Addr. to check if it supports the specific parameter group (whether the requested Target Addr. transmits the specific PGN).

Response to request must depend on whether this PGN is supported or not. If supported, the response device wil send the requested information. If PGN is acknowledged to be correct, control bytes are set 0, 2 or 3; if not supported, the response device will send the acknowledge PGN with value of control byte=1 as negative response.

5.2.3.3 Radio/Response

This message type may be message radio actively supplied by one device, or the response to command or request.

5.2.3.4 Acknowledgement

ACK has two forms. The first one is provided in CAN protocol, which consists of an “interframe” ACK, in order to acknowledge ther reception of a message at one or more nodes. In addition, if no CAN error frame appears, this message will be further acknowledged. If no error frame appears, it indicates that all other power-up devices that are connected to the bus line have correctly received this message.
The second form of ACK is provided in the application layer, which responds to the specific command, requested “Common Radio” or “ACK” or “NACK”. ACK message parameter group PGN is 59392(00E800).

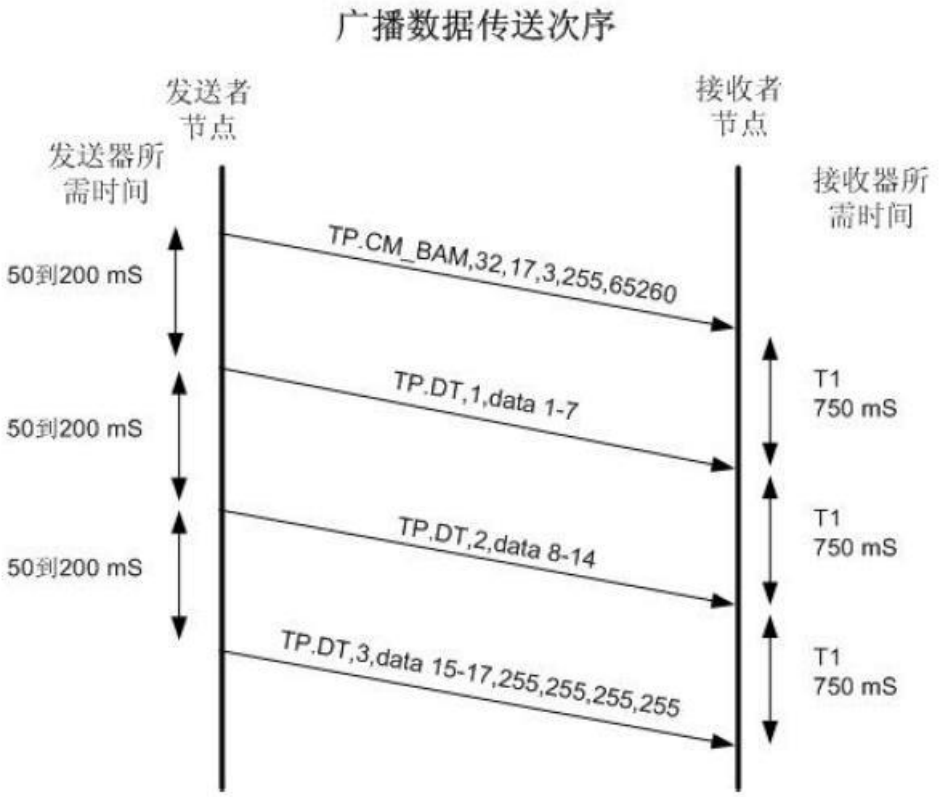
5.2.3.5 Group Function

This message type is used for special functional group (e.g.: special function, network management function, multi-frame transmission function). Each group function is identified by its PGN. Function is defined in data structure(generally the first byte in data field). Multi-frame transmission protocol is specified in the following chapter. Special group function can be used to resolve the conflict arising from CAN identifier used by different manufacturers in the transmission of special message. If necessary, an approach is provided for the reception and identification of special information. If the message defined under this standard is insufficient, the request of group function, ACK and/or NACK mechanism can be user-defined.
PGN 59904 request (refer to 5.4.2) can be used to check if the Target Addr. supports the specific parameter group or group function of some message type. If supported, the response device will send acknowledge PGN, in which the value of control byte is 0(Positive Acknowledgement) or 2(Access Denied) or 3(No Answer). If not supported, the response device will send acknowledge PGN, in which the value of control byte is 1(Negative Acknowledgement).

5.2.4 J1939 Long Message Frame-to-Frame Transmission

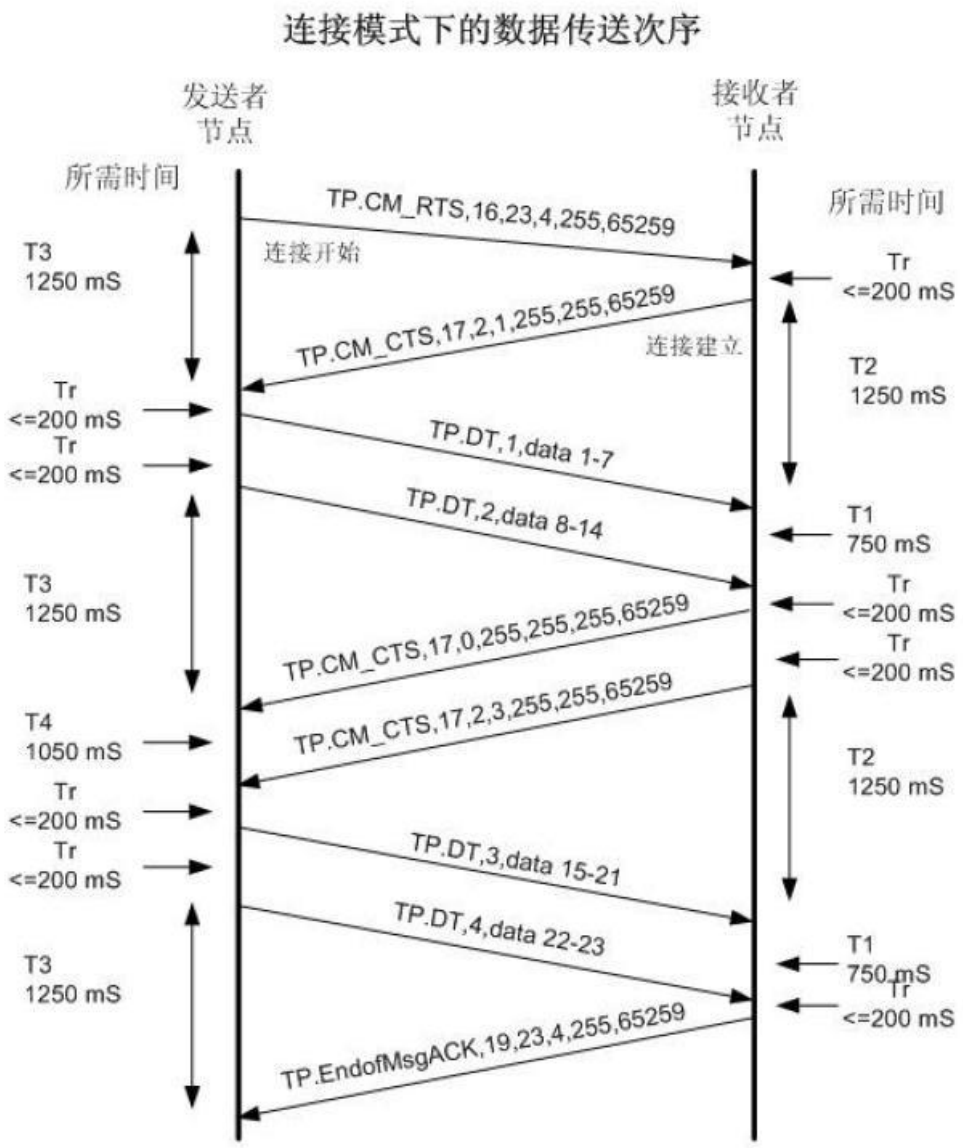
Transmission in BAM Radio Mode

Target Addr. of radio message must be global address FF. Firstly, sender transmits TP_CM_BAM message, indicating that a 17-byte message is subpackaged into 3 data packets, the message with PGN as 65260 will be sent. This is followed by the consecutive transmission of 3-frame data.



5.2.4.1 Long Message Transmission in RTS/CTS Connection Mode

Normally, data transmission is processed by the data flow mode as shown in the figure below. Sender transmits TP.CM_RTS message, indicating that a 23-byte message is subpackaged into 4 data packets that will be sent. PGN value of data packet member in transmission is identified as 65259.



Data frame handled in the foregoing transmission process is listed below:
<DA> is Node A of receiver address, and <SA> is Node B of sender address

Direction	ID(HEX)	DATA(HEX)	PGN in Current Frame	Remark
A→B	18EC <DA.SA>	10 17 00 04 FF EB FE 00	TP_CM_RTS PF=EC PGN=00EC00	A sends RTS(16=0x10) Transmitted data PGN=65259(00FEEB) Total data length is 23 bytes, 4 frames
B→A	18EC <SA.DA>	11 02 01 FF FF EB FE 00	TP_CM_CTS PF=EC PGN=00EC00	B responds to receiver CTS(17=0x11) B notifies sender A, start sending data from Frame 1, 2

				frames can be consecutively sent. Transmitted data PGN=65259
A→B	18EB <DA.SA>	01 xx xx xx xx xx xx xx	TP_DT PF=EB PGN=00EB00	A sends Frame 1 data 1-7 to B Xx is the sent data
A→B	18EB <DA.SA>	02 xx xx xx xx xx xx xx	TP_DT PF=EB PGN=00EB00	A sends Frame 2 data 8-14 to B After A sends 2 frames, wait for the next receiver B's CTS
B→A	18EC <SA.DA>	11 00 FF FF FF EB FE 00	TP_CM_CTS PF=EC PGN=00EC00	B makes CTS response, Because data frame that B responds to the receivable is 0, A is unable to send data, keep waiting.
B→A	18EC <SA.DA>	11 02 03 FF FF EB FE 00	TP_CM_CTS PF=EC PGN=00EC00	B responds to the receiver CTS(17=0x11) B notifies sender A, start sending data from Frame 3, 2 frames can be consecutively sent. Transmitted data PGN=65259
A→B	18EB <DA.SA>	03 xx xx xx xx xx xx xx	TP_DT PF=EB PGN=00EB00	A sends Frame 3 data 15-21 to B
A→B	18EB <DA.SA>	04 xx xx FF FF FF FF FF	TP_DT PF=EB PGN=00EB00	A sends Frame 4 data 22-23 to B After A sends the completed data, wait for B's reception of completed response TP_END_MSG
B→A	18EC <SA.DA>	13 17 00 04 FF EB FE 00	TP_END_MSG PF=EC PGN=00EC00	B responds to receiver ENDofMsg (19=0x13) B notifies sender A, 23 bytes are successfully received, 4-frame data Transmitted data PGN=65259

Note: Please refer to SAE J1939-21 text for more details about J1939

5.2.5 Registers(Set) Related to J1939

- CINTF2：Multi-frame Transmit Flag Register
- CPROCON：Multi-frame Transreceive Protocol Control Register

- CCFDL：The number of Multi-frame Transmit Buffer Data Frame, 36 frames to the maximum
- CJ1939CTRL：Multi-frame Transmit Control Register Set(4 bytes)
- CJ1939_T2 1939 T2 = xx * 5ms When the multi-frame is received, maximum waiting time of DT
- CJ1939_T3 J1939 T3= xx * 5ms When the multi-frame is sent, maximum waiting time of receiver CTS
- CJ1939_TSJ1939 TS= xx * 1ms Time slot in consecutive transmission of DT
- CJ1939_TS J1939 TS= xx * 1ms
- CJ1939_TR J1939 TR= xx * 1m When the multi-frame is received, the response frame CTS/EOM delay needs to be transmitted
- CJ1939_TR J1939 TR= xx * 1m

5.2.6 Transmit Buffer Used under J1939

When J1939 protocol control is activiated, CAN module applies 4 special transmit buffers to the transmission and reception of multi-frame information. If user needs to transmit other frame data, CTXB0-CTXB3 transmit buffer may be used.

- CTXB4：Specifically used for the transmission of RTS or TP_CM_BAM frame, activate the multi-frame transmission of long message.
- CTXB5：Specifically transmit DT consecutive data frame automatically generated by module.
- CTXB6： When RTS/CTS multi-frame message is received, specifically used for the transmission of CTS0.
- CTXB7： When RTS/CTS multi-frame message is received, specifically used for the transmission of CTS1.
- CTXAUTOEN<EN7.EN6.EN5> must be set Clear, data transmission will not be started when data is written in CTXB5/6/7, data transmission is automatically started when valid RTS or other frames are received against the transmission of DT/CTS/EOM.
- CFTXB: This is buffer with 256 bytes, actual length of valid data is 252 bytes to the maximum.

User firstly writes long message data to be transmitted in CFTXB buffer, then set CCFDL(the number of frames of DT required by data in CFTXB) When RTS or TP_CM_BAM is written in CTXB4, J1939 starts the multi-frame transmission. If CJ1939CTRL.BAM=0, when the sender receives CTS responded from receiver, data in buffer CFTXB is automatically segmented and transmitted in DT consecutive frame, with interframe time slot as J1939_TS, until the data transmission in CFTXB is completed.

If CJ1939CTRL.BAM=1, CAN module automatically starts segmenting and transmiting data in the buffer CFTXB in DT consecutive frame, until the data transmission in CFTXB is completed. Interframe time slot is CJ1939_TS.

CCFDL number of frame must not set greater than 36. (J1939 DT each frame is 7 valid data. 36*7=252 bytes)

- If the length of last frame is less than 8 bytes, CAN module compel data frame length DLC is 8, Filling bytes follow the valid data in CFTXB(Generally must as FF).

- ET7190 CAN Module: The longest message that J1939 can automatically send frame-to-frame is 252 data length. If total length of valid data exceeds 252, user needs to handle RTS/CTS/DT, the automatic segment function in CAN module J1939 must not be used for the transmission of long message. (J1939 provides that the longest message is 255 frames, i.e.: 255*7=1,785 bytes)

5.2.7 Configuration Register/Register Set/Transmit Buffer

- A—Configure J1939 Protocol Control Register Set CJ1939CTRL

CJ1939CTRL.R0FLTN=0 Generally set as smaller filter number for precise match. Target Addr. is the address of this module. This filter receives RTS that is transmitted to this device, if multiple frames of Target Addr. need to be received in the application, the filter with bigger number is set as multiple match. This filter setting is very important.

CJ1939CTRL.R1FLTN=0 The same as above, generally the same with Setting 1, because ET7190 has 2 CTS automatic response controls, generally 2 modules are possible to simultaneously launch RTS message to this device address, ET7190 may simultaneously handle 2 connections depending on different source addresses of transmitter, and process RTS/CTS/EOM message.

CJ1939CTRL.TXFLTN=0 In transmission, the filter in reception of CTS message is generally the same with that in reception.

CJ1939CTRL.BSIZE=16 Maximum BS is between 1 and 255

CJ1939CTRL.BSM=0 In reception, BS is automatically set

CJ1939CTRL .BAM=0 Mode when the long message is transmitted

- B—Control Register CPROCON

  CPROCON.PTYPE=0b010 J1939 Standard Addr. Mode
  CPROCON.OP0E =1 Enable Response CTS0
  CPROCON.OP1E =1 Enable Response CTS1
  CPROCON.EN =1 Enable Protocol

- C—Set CTS automatic response frame, transmit buffer CTXB6/7
  CTXAUTOEN<EN7.EN6.EN5> must be set 0, transmission will not be strated when data is written in CTXB6/7, CTS transmission automatically triggers one or more transmissions in handling DT after an valid RTS is received.

  CTXB6/ CTXB7 only sets ID, and DLC must be 8, data field content is irrelevant, when RTS is received, ET7190 is able to automatically fill in 8-byte data field, ID is generally set as:

  SID:EID=1C EC <DA> <SA> CAN transmit frame length field DLC=8

  Wherein: 1C is Priority P=7 PF=EC
  DA may be any value, RTS sender's Physical Addr. is automatically filled when ET7190 receives RTS.
  SA is the address of this device, or Source Addr., for example, the address of diagnostic tester is F9, then SID:EDIT=1C EC 00 F9.

- D—Set Transmit Buffer CTXB5(DT Transmission ID)
  CTXAUTOEN<EN5> must be set Clear, transmission will not be started when data is written in CTXB5, transmission is automatically started when an valid CTS is received against the transmission of DT.

  CTXB5 only sets ID, and DLC must be 8, data field content is irrelevant, when CTS is received,
  ET7190 only extracts data from CFTXB, 8-byte data field is automatically filled in, ID is generally set as:
  SID:EID=1C EB DA SA transmit frame length field DLC=8
  Wherein: 1C is Priority P=7 PF=EB (TP_DT)

DA is the receiver's address. For example, the engine module is 01(DA must be FF in caes of BAM transmission)

SA is the address of this device, or Source Addr., for example, the address of diagnostic tester is F9, then SID:EDIT=1C EB 01 F9.

- E—Set Filter

  J1939 Set Filter is generally as:

  Set Filter 0 is used to receive all message when Target Addr. is the address of this device(SA=0xF9), Mask 0

  Set Filter 1 is used to receive all message when Target Addr. is global address(0xFF), and Mask 0 is used to receive BAM message.

  Set Filter 2 is used to receive global PGN parameters in PF=0xF0-0xFF, and Mask 1 is used to receive parameters in PDU2 format. If all PDU2 are not required to receive, PDU can be precisely matched by configuring Filter 2-15.

  CFEN0.EN0=1, CFEN0.EN1=1 ,CFEN0.EN2=1 Enable Filter 0, 1 and 2

- F—Set Time

CJ1939_T2=250	250*5ms=1250ms	
CJ1939_T3=250	250*5ms=1250ms	
CJ1939_TS =100	100ms	Transmit gap of consecutive DT
CJ1939_TR =100	100ms	After RTS is received, time delay of response CTS

5.2.8 CAN Module J1939 Long Message Automatic Frame-to-Frame Transmission Process

CAN module automatically transmits long message(RTS/CTS) frame-to-frame according to CJ1939CTRL setting.

1. Configure related register as per 5.2.7
2. Enable CAN module in normal mode
3. Write all data of long message in CFTXB transmit buffer
4. Write DT ID in CTXB5 buffer, note that Target Addr. and Source Addr. must be correct, DLC=8
5. Write the number of frames in CCFDL transmit buffer (note that it is the number of frames rather than the length of buffer)

Transmission is started when RTS is written in CTXB4

CAN module waits for receiver to reply CTS/ABT/EOM after RTS is completely transmitted.

- If CTS/ABT/EOM is not received within CJ1939_T3, TIMEOUT<CINTF2> error will be generated, if interrupt is enabled, error interrupt will be generated and the transmission of long message will be aborted. MTXF<CINTF2> is automatically cleared.
- If CTS BS=0 is received, CAN module will keep waiting for CTS, and CJ1939_T3 is recounted.

- If total number of SN>CCFDL famres of CTS is received, SNERR error will be generated, and the transmission of long message will be aborted. MTXF<CINTF2> is automatically cleared.
- If TP.CM_EndofMsgAck is received, CAN module J1939 long message is completely transmitted. CTX<INTF> is set ON when long message is completely transmitted. If interrupt is enabled, interrupt will be generated.
- If TP.CM_Abort is received, NOTREADY/ABT error will be generated, and the transmission of long message is aborted. MTXF<CINTF2> is automatically cleared.

6. Retransmission after Abort

If the transmission error of long message is checked to abort, just follow Step 6 to restart the transmission, RTS is written in CTXB4 to restart the transmission of long message.

7. If the transmission is actively aborted in the transmission of long message
CCTRL1.CMTXF is set ON, multi-frame transmission will be enforced to abort.

e.g. 1: 10 data, 1,2,3,4,5,6,7,8,9,10, will be sent to the receive module. Receiver module address 01, sent in F9, PGN of transmit number is 00FEE3

1. 1,2,3,4,5,6,7,8,9,10 are written in CFTXB, total length of <FF,FF,FF,FF> is 14 bytes, FF is Filled if the length of last frame is insufficient. The length of DT valid data is 7 bytes, 4 bytes need to be additionally Filled. So 4 FFs are written to Fill, if not written, the Filling value of the last DT is undetermined.
2. Write CCFDL=2 Actual valid number of frames in CFTXB
3. Write CTXB5, ID=1C EB 01 F9 DLC=8 (PF=EB TP_DT)
4. Write RTS in CTXB4 to start transmission
ID= 18 EC 01 F9 （PF=EC DA=01 SA=F9 P=6 DP=0）
DATA= 10 0A 00 02 FF E3 FE 00 (The meaning of each byte is carefully analyzed, PGN is in red letter)

Actual transmitted data appeared in bus line are:

TESTER→ECU: RTS 18 EC 01 F9 10 0A 00 02 FF E3 FE 00

ECU→TESTER: CTS 1C EC F9 01 11 02 01 FF FF E3 FE 00

TESTER→ECU: DT 1C EB 01 F9 01 **01 02 03 04 05 06 07**

TESTER→ECU: DT 1C EB 01 F9 02 **08 09 0A** FF FF FF FF

ECU→TESTER: EOM 1C EC F9 01 13 0A 00 02 FF E3 FE 00

e.g. 2: 10 data, 1,2,3,4,5,6,7,8,9,10, are sent to the receive module in BAM mode. Receiver module address must be FF, sent in F9, PGN of transmitted data is 00FEE3

Set CJ1939CTRL.BAM=1

1. 1,2,3,4,5,6,7,8,9,10 are written in CFTXB, total length of <FF,FF,FF,FF> is 14 bytes, FF is Filled if the length of last frame is insufficient. The length of DT valid data is 7 bytes, 4 bytes need to be

additionally Filled. So 4 FFs are written to Fill, if not written, the Filling value of the last DT is undetermined.

2. Write CCFDL=2 Actual valid number of frames in CFTXB
3. Write CTXB5, ID=1C EB 01 F9 DLC=8 (PF=EB TP_DT) DA must be FF
4. Write RTS in CTXB4 to start transmission

   ID= 18 EC FF F9 （PF=EC DA=FF SA=F9 P=6 DP=0）

   DATA= 20 0A 00 02 FF E3 FE 00 (The meaning of each byte is carefully analyzed, PGN is in red letter)

   Actual transmitted data appeared in bus line are:

   TESTER→ECU: BAM 18 EC FF F9 20 0A 00 02 FF E3 FE 00
   TESTER→ECU: DT 1C EB FF F9 01 **01 02 03 04 05 06 07**
   TESTER→ECU: DT 1C EB FF F9 02 **08 09 0A** FF FF FF FF

   Note: J1939 sets out that only one BAM message is allowed to appear in the bus line at the same time.

5.2.9 CAN Module Automatic Response in Reception of RTS Message

- If enabled, CAN module will automatically start one CTS response according to the data length and frame number of data when RTS is received. The complete RTS/CTS long message will be automatically received at the user end, and TP_CM_EOM message is sent when current data frame is completely received.

- Automatic response only sets PGN transmission of 2 long message simultaneously.

  After CTS is automatically responded, if no DT data is transmitted from sender to CAN module within CJ1939_T2, LOSSDT error will be produced in CAN module. If interrupt is enabled, error interrupt will be generated.
- Response Time: When RTS is received, CJ1939_Tr is delayed and CTS transmission is started. Or when the last DT in BS is received, if message is not completed, CJ1939_Tr will be delayed and the transmission of next CTS will be started. After the last DT is received, CJ1939_Tr is delayed, and the next TP_CM_EOM message transmission is started.

- Abort frame will not be transmitted and SN is automatically incremental according to the serial numberwhen CAN module automatically handles, and the retransmission of SN is not required.

5.3 VW TP2.0 Application

TP2.0 diagnosis is done before the communication of one ECU module. Channel setup is firstly required, after that, connection setup must be immediately finished, and ET7190 can automatically finish 2 communication setups with ECU module. Timing and ID are automatically set.

TP2.0 uses 11-bit ID standard frame format, communication BaudRate is 500KBPS.

5.3.1 TP2.0 Channel Setup

5.3.1.1 Channel Setup Message Structure

ID(11bits)	1.byte	2.byte	3.byte	4.byte	5.byte	6.byte	7.byte
200	Destination	OpCode	Parameter				

- ID : The fixed tester address 0x200 before channel setup of tester
- Destination: ECU address (Target Addr.)
- Opcode: Message of dynamic channel setup

  0xC0 Request channel setup
  0xD0 Channel positive respond
  0xD6 Channel negative respond, application type is not supported
  0xD7 Channel negative respond, application type is temporarily not supported
  0xD8 Channel negative respond, free resource is temporarily unavailable

- Parameter: The Parameter data area is assigned according to the opcode

Diagnosis Request Channel Setup Message
In diagnostic communication, this is command message of channel setup, DLC=7

ID(11bits)	1.byte	2.byte	3.byte	4.byte	5.byte	6.byte	7.byte
200	Destination	Opcode	00	0x10	WORD<10:0>RX_ID_A		01
200	xx	0xC0	00	0x10	00	03	01

- Detination: The 11-bit ID lower-order 8 bits of ECU Physical Addr., and other data is not variable. Specifically for diagnosis.
  e.g.: CAN gateway: 0x1F, instrument: 0x07
- WORD<10:0>RX_ID_A is 11-bit receive ID after channel setup of tester, as 0x300(Active module receive ID, invariable, A mnemonic, Active means active module)
- See table below: ECU respond channel setup 11-bit ID is 0x2xx, xx= Detination,
  e.g.: 0x207 instrument, 0x21F CAN gateway

ECU Channel Setup Positive Respond

ID(11bits)	1.byte	2.byte	3.byte	4.byte	5.byte	6.byte	7.byte
2xx	Destination	Opcode	WORD<10:0> =TX_ID_P		WORD<10:0> =RX_ID_P		01
2xx	0x00	0xD0	0x300		xxx		01

- RX_ID_P is the ID of ECU module receive data, subsequent communication, tester must send this ID to communicate with ECU module.
- TX_ID_P is the ID of ECU module transmit data, this ID must be equal to RX_ID_A(0x300) at request. Tester must receive the message in this ID. (In TX_ID_P, P means Passive(passive module)
- The meaning of WORD<10:0> is a 16-bit interger consisting of Byte 3, 4(5.6 bytes), with lower-order 11 bits selected. Higher-order 5 bits can be neglected.
- ID：11-bit ID of ECU respond channel setup is 0x2xx, e.g.: 0x207 instrument, 0x21F CAN gateway

5.3.2 Transport Protocol Data Unit(TPDU) afte Channel Setup

TPDU message type after channel setup includes the following 7 formats.

Message Type Data Length DLC	Abbr.	11-bit ID	TPDU 8 bytes							
			BYTE 0(PCI)	1	2	3	4	5	6	7
Data(DLC Variable)	DT	RX_ID_A Or RX_ID_P	**TPCI**	D/-	D/-	D/-	D/-	D/-	D/-	D/-
Acknowledge DLC=1	ACK		**TPCI**	-	-	-	-	-	-	-
Connection Setup DLC=6	CS		0xA0	BS	T1	T2	T3	T4	-	-
Connection ack DLC=6	CA		0xA1	BS	T1	T2	T3	T4	-	-
Connection Test DLC=1	CT		0xA3	-	-	-	-	-	-	-
Break DLC=1	BR		0xA4	-	-	-	-	-	-	-
Disconnect DLC=1	DC		0xA8	-	-	-	-	-	-	-

D is the valid data of message, "-" means this byte is not transmitted, the length of message must match with message type in transmission.

- **RX_ID_A**: Receive ID of Active(Tester) in channel setup
  RX_ID_P: Receive ID of Passive(ECU) in channel setup

- **TPCI**：Protocol control information byte of DT and ACK, see Table below

- **BS**：Maximum value in block transport, must be between 1 and 15.
  Module indicates that sender is allowed to consecutively transmit the maximum BS DTs. After BS DTs are consecutively transmitted, one ACK must be responded by receiver.

- **T1/ T2/T3 /T4：** Time bytes of transport protocol control
  Calculation of time bytes: T=n*Tb

  Bit7:6 2 bits represent Tb(time base)
  00 : Tb=100us
  01 : Tb=1 ms
  10 : Tb=10ms
  11: Tb=100ms
  Bit5:0 Time unit n
  Note: If the byte value is FF, this time is not used.
  e.g.: Suppose T1 byte value as 0x4F, then n=15 Tb=1ms T1=n*Tb=15ms

Definition of Transport Protocol Control Information(TPCI) of DT and ACK

TPDU Frame Type	Code	TPCI Byte 1							
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Data Frame	DT	0	0	AR	EOM	SN			
Respond Frame	ACK	1	0	RS	1	SN			

- **AR：** Whether receiver's response is required after DT is transmitted
  0=Response is required
  1=Response is not required

- **EOM：** Whether all data of message has been completely transmitted(End of Message)
  0=Data not completed
  1=All data transmission completed

- **SN：** Cyclic number from 0x0 to 0xF
  DT: SN is the cycle of sender from 0x0 to 0xF, each module is independent once one DT SN+1 is transmitted.
  ACK: SN is DT SN+1 of request response.

- **RS**: Receive state, ACK respond module notifies the current state of receiver to transmit module
  0=Unprepared to receive, CAN module will delay CTP_Tw, transmit the next DT data in case of automatic transmission of long message.
  If user transmits DT in application, 100ms must be delayed to transmit DT.
  1=Prepared to receive data

5.3.3 Diagnosis Request/Response Message Format – Data Definition of Application Layer

DT Data Frame	TPDU 8byte							
	Byte1	2	3	4	5	6	7	8
1 DLC=8	TPCI EOM=0	**0x00/0x80**	**Len=14**	D	D	D	D	D
2 DLC=8	TPCI EOM=0	D	D	D	D	D	D	D
3 DLC=3	TPCI EOM=1	D	D	-	-	-	-	-

- Byte 2 of the first DT Sender transmited message(might be one or more DT message):
  00=Represent no subsequent message after this message is completely transmitted.
  80=Reprsent more message followed the transmission of this message, so receiver must continue receiving the following message.
  1. Multiple information may occur when ECU message is read.
  2. If ECU is busy, one negative message may be sent by ECU, and tester is made to keep waiting until the subsequent positive message is received.
- Byte 3 of the first DT of sender transmitted information represents the data length of this message, up to 255 bytes. A 14-byte long message is shown in the table. EOM=1 in TPCI byte represents the complete transmission of message.

An example of one request ECU information process is illustrated below(Instruction 1A 9B):
Note that red letters represent the first 2 data bytes of the first DT in each message, ECU responds 5 message once tester sends one 1A9B diagnosis request, the first 3 are negative message among 5 message responded from ECU(might be ECU unprepared data, in order to prevent the requester T1 time overflow, ECU must send one message within T1), after 3 T1s, ECU has prepared data, and 2 sent ones are actual data message. Tester must handle each message according to the application layer protocol. The following data are carefully analyzed to understand TP2.0 data processing.

790 11 00 02 1A 9B
300 B2
300 11 80 03 7F 1A 78 (Note: 7F 1A 78 represents negative respond, KWP2000 definition)
790 B2
300 12 80 03 7F 1A 78 (Time T1, if sender unprepares the data, receiver continues waiting)
790 B3
300 13 80 03 7F 1A 78
790 B4
300 24 80 24 5A 9B 31 4A 44 (Note: 1A+0x40=5A respresents respond, KWP2000 definition)
300 25 39 30 37 33 37 39 20 (Valid message 1, 6 DTs in total, 36 bytes long)
300 26 20 20 30 31 30 34 03
300 27 00 51 21 00 00 00 00

300 28 00 4E 4D 41 42 53 20
300 19 4D 4B 37
790 BA
300 2A 00 0D 5A 30 4D 20 20 (Valid message 2, 3 DTs in total, 13 bytes long)
300 2B 20 20 20 20 20 20 20
300 1C 20
790 BD

Through the above analysis, TP2.0 transmit/receive format needs to be understood:

- Transmission of only single DT data message
- Data transmission of multiple DT message(long message)
- Consecutive transmission of multiple message(long message or single-frame message) and reception of multiple message
- Response of DT message receiver

Note: One TP2.0 message may consist of one or more DTs, depending on the length of message.

5.3.4 TP2.0-related Registers(Set)

- CINTF2： Multi-frame Transmission Flag Register
- CPROCON： Transreceive Protocol Control Register
- CCFDL： Length of Multi-frame Transmit Buffer
- CTP20CTRL： Multi-frame Transmit Control Register Set(4 bytes)
- CTP _T1 Overflow Time of Receiver Waiting for DT. *5ms
- CTP _T3 Interval of Consecutive Transmit DT. *100us
- CTP _TE Overflow Time of Waiting for Passive Response in Channel Setup/Connection Setup.*5ms
- CTP _T3A Interval of Consecutive Transmit DT. *1ms
- CTP_TwTwait : Time delay of module transmit DT when receiver unprepares to receive. *5ms

5.3.5 Transmit Buffer under TP2.0

When TP2.0 protocol control is activated, CAN module uses 4 special transmit buffers for the automatic frame-to-frame transmission of long message/communication setup/automatic response. CTXB0-CTXB3 transmit buffer may be used when user needs to transmit other frame data.

- CTXB4：Specifically used for the first DT of long message, multi-frame transmission of long message is started. And the subsequent DT is transmitted.
- CTXB5：Specifically send the command of Channel Setup, used for the initialization process of communication setup.
- CTXB6：Specifically send the command of Connection Setup, automatically triggered after the command of Channel Setup is transmitted.
- CTXB7：Specifically automatically respond to DT that needs response.

- CTXAUTOEN<EN7.EN6> must be cleared, transmission will not be started when data is written in CTXB6/7, the command of Connection Setup accepts ECU response after user sends the command of Channel Setup, and the transmission of Connection Setup is automatically started by CAN module.

- CFTXB This is 256-byte buffer, and the actual length of valid data is 252 bytes to the maximum.

  User firstly needs to write the data of long message to be sent in CFTXB buffer, then CCFDL(the length of valid data in CFTXB, excluding the first DT in CTXB4 transmittd by user) is set, when the first DT is written in CTXB4, (data field of the first DT is required to company with TP2.0 SN and format, the length byte in data format is CTXBUF length+5), and multi-frame transmission is started.

  After the first Dt is completely transmitted, data in the buffer CFTXB is automatically segmented to transmit DT consecutive frame, with interframe time slot as T3. If the number of consecutive frame equals to BS, transmitter will send one DT that needs receiver's response, after receiver normally responds, the subsequent DT is continuously transmited, until the data transmission in CFTXB is completed. If the state of receiver's respond frame RS=0, the subsequent DT will not be transmitted until after CTP_Tw(100ms).

  ET7190 CAN module J1939 can automatically transmit the message frame-to-frame up to 252+5 data length(actually up to 255 due to the limit of length bytes). If total length of message valid data is greater than 255, user needs to processs DT BS/T3 and length bytes(as defined in application layer), CAN module automatic segment function must not be used to transmit the long message.

5.3.6 Configuration Register/Register Set/Transmit Buffer

- A—Configure TP20 Protocol Control Register Set CTP20CTRL

  CTP20CTRL.CHSFLTN=1 Generally set as smaller filter number for precise match, the received value by receive filter is the ID responded by ECU in Channel Setup. For example, instrument is 0x207, and gateway is 0x21F.

  CTP20CTRL.CTSFLTN=0 Generally set as smaller filter number for precise match, the received value by receive filter the ID responded by ECU in normal communication after

Channel Setup is completed. Normally as 0x300. Note: In Channel Setup, CAN module allows CAN module to automatically set the filter ID with this number.

CTP20CTRL.BSIZ= 15 Default BS
CTP20CTRL.BSM=0 Automatically set BS in Connection Setup
CTP20CTRL.IDM=0 CTSFLTN filter ID and CTXB6 ID automatic setting
CTP20CTRL.STM=0 T3 Automatic Setting
CTP20CTRL.ACK=0 DT automatic response started
CTP20CTRL.MNTC=10 The number of retransmission when Channel Setup fails
CTP20CTRL.MNT=3 The number of retransmission when Connection Setup fails

- B—Control Register CPROCON

CPROCON.PTYPE=0b011 TP20 Protocol Mode
CPROCON.OP0E =0 Irrelevant
CPROCON.OP1E =0 Irrelevant
CPROCON.EN =1 Enable Protocol

- C—Set Transmit Buffer CTXB6(CTXB6 is required to set only in Channel Setup and Connection Setup)

CTXAUTOEN<EN6> must be cleared, and the transmission will not be started when data is written in CTXB6,

CTXB6 content is CS(Connection Setup) command. Its format is shown in xx-xx

CTXB6 ID can be automatically set in CAN module(BSM=0),

- D—Set Transmit Buffer CTXB7(DT Automatic Response)

CTXAUTOEN<EN7> must be cleared, and the transmission will not be started when data is written in CTXB7,
CTXB7 content is the ID automatically responded in this module, data field content is irrelevant, DLC length must be 1.

When CAN module receives DT that requires response, the respond data field is automatically set, but ID and DLC in CAN module will not be modified. CTXB7 ID must be RX_ID_P value in Channel Setup. This value must be set after Channel Setup. (Because it is unknown which ID can be received by ECU prior to Channel Setup)

- E—Set Filter
  Generally TP20 Set Filter is provided below:
  Set Filter 0 is all message with Receiver Target Addr. as 0x300.
  Set Filter 1 is the message responded from ECU in Receive Channel Setup(2xx). Xx is ECU Physical Addr.
  CFEN0.EN0=1, CFEN0.EN1=1 Enable Filter 0, 1

- F—Set Time

  CTP20_T1=20 20*5ms=100ms
  CJ1939_T3=50 50*100us=5ms
  CJ1939_TE =20 20*5ms=100ms
  CJ1939_Tw =20 20*5ms=100ms Time delay when receiver RS=0

5.3.7 CAN Module TP20 Long Message Automatic Frame-to-Frame Transmission Process

1. Relevant Register Configured as per 5.3.6
2. Enable CAN Module Normal Mode
3. All Data Written in CFTXB Transmit Buffer
4. Data Length Written in CCFDL Transmit Buffer
5. FF Written in CTXB4to Start Transmission
   When the first DT is completely transmitted, the module automatically generates the consecutive DT from CFTXB.
   - When the last DT in BS is transmitted, this DT requires the response from receiver. (If the data length is greater than a BS, PCI byte AR=0, EOM=0). The subsequent DT is continuously transmitted in reception of normal response.
   - When the last DT of long message is transmitted, this DT(PCI byte AR=0， EOM=1) requires the response from receiver. After the normal response is received, transmission completed. MTXF is cleared. INTF<CTX> transmit completion flag is set ON. If interrupt is allowed, interrupt will be generated.
   - Receiver responds, if ACK RS=0; it represents that receiver is not well-prepared to receive data, sender will delay CTP20_Tw(100ms) to send the next DT.
     Receiver responds, if ACK SN is not equal to DT SN+1, the transmission of CAN module long message will be aborted, MTXF is cleared. INTF2<SNERR> error flag is set ON. If interrupt is allowed, interrupt will be generated.
   - If DT requires response, the generated error TIMEOUT<CINTF2> will be set ON if no ACK is received within CTP20_T1. If interrupt is allowed, error interrupt will be generated and the transmission of long message will be aborted. MTXF<CINTF2> is automatically cleared.

6. Retransmission after Abort
   If the transmit error of long message is checked to abort, just follow Step 5 for retransmission, the retransmission of long message will be restarted when the first DT is written in CTXB4.

7. If transmission is actively aborted in the transmission of long message, CCTRL1.CMTXF is set ON to abort the multi-frame transmission.
8. Time Slot T3 in Consecutive Transmission of DT
   - If CTP20CTRL.STM=0 T3 time is automatically set according to ECU response in the instruction of Connection Setup, if no Channel Setup is processed before communication, multi-frame transmission is directly processed, when T3 is CTP20_T3(T3A) set time.
   - If CTP20CTRL.STM=1 T3 time is CTP20_T3(T3A) set time. This time will not be changed in the process of Connection Setup.
   - The time when data is written in CTP20_T3 is 100us per frame, while it is 1ms per frame when data is written in CTP20_T3A, the value written in these two registers is an operation targeted at the same T3 timer. The following writing is valid.

5.3.8 CAN Module TP20 Automatic Response to DT

In DT transmission, TP20 may sometimes require receiver to make ACK response, CAN module sets the automatic response function, and receiver will automatically respond to ACK if this function is enabled, user needn't to transmit the response frame. This process of reception tends to be more convenient.

- If CTP20CTRL.ACK=0. CAN module will automatically send one ACK(CTXB7 set ID, DT receive filter must be numbered corresponding to CTSFLTN) when CAN module receives DT that requires response.

- Automatic respond ACK ID is CTXB7 setting, DLC must be 1, ACK PCI data is automatically Filled by CAN module according to the received DT SN, ACK respond RS receive state is always 1, CAN module thinks it always prepared to receive data, so user operates the automatic response function. 16 FIFOs of CAN module must not overflow in data processing.

- If CTP20CTRL.ACK=1, CAN module will not automatically transmit ACK, all frame responses require user to respond.

5.3.9 TP2.0 Communication Setup Process

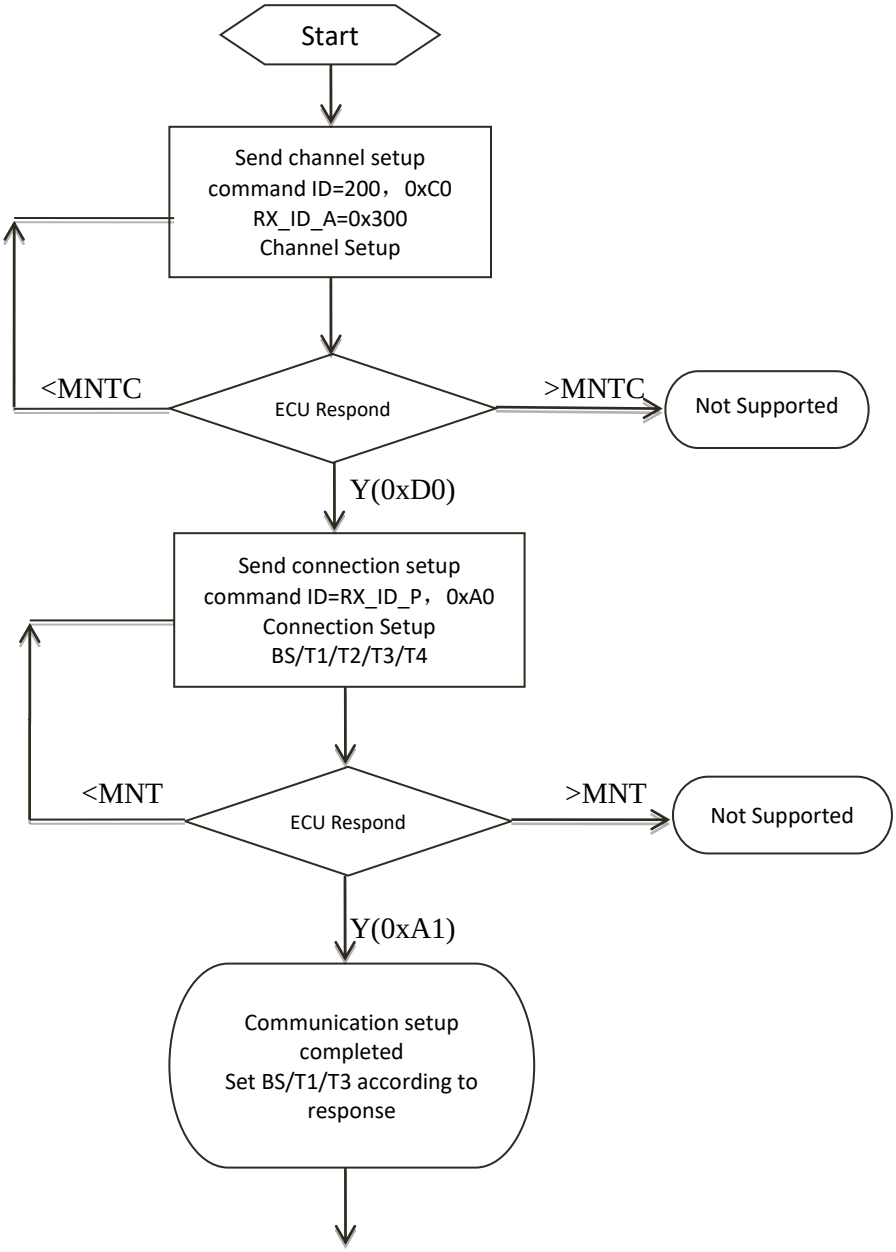
Communication Setup between tester(Active Module) and instrument(Passive Module) is listed below for the purpose of analysis.

5.3.9.1 Communication Example between Tester and Instrument

Sequence	CAN Frame	Frame Type	Description
1	→200 07 C0 00 10 00 03 01	0XC0 Channel Setup	Tester sends the command of Channel Setup, ID=0x200 Target Addr.: 0x07 Instrument(ECU)
2	←207 00 D0 00 03 51 07 01	0xD0	ECU Channel Setup response, ID=0x207 TX_ID_P=0x300 is the ID when ECU transmits CAN after Channel Setup, RX_ID_P=0x751 is the ID when ECU receives Can after Channel Setup. After that, tester must transmit data in ID:0x751, and receive the data from instrument module in ID as 0x300
3	→751 A0 0F 8A FF 32 FF	CS(0xA0) Connection setup	Tester sends the command of Connection Setup BS=15 T1=100ms(0x8A) T3=5ms(0x32) T2/T4 Normally as FF unused
4	←300 A1 0F 8A FF 4A FF	CA(0xA1)	Tester POSTIVE RESPOND BS=15 T1=100ms T3=10ms(0x4A)
Diagnosis process follows the completion of the said Communication Setup Process			
5	→751 10 00 02 10 89	DT TPCI=0x10	Send 1089 diagnosis mode instruction, instruction length is 2 bytes SN=0 Start from 0, this is the first DT of tester AR=0 Receiver's response is required EOM=1 Data transmission is completed, only one frame
6	←300 B1	ACK TPCI=0xB1	ECU responds to tester, SN is tester DT SN+1 Indicate the instruction of tester is correctly received, (RS-=1 means that ECU has well-prepared to receive the next data, if tester transmits)
7	←300 10 00 02 50 89	DT	ECU starts to respond to tester 1089 request

		TPCI=0x10	SN=0 Start from 0, this is the first DT of ECU AR=0 Receiver's response is required EOM=1 Data transmission completed, only one-frame 5089 indicates this function is supported
8	→751 B1	ACK	Tester to ECU respond frame RS=1
9	→751 11 00 02 1A 9B	DT TPCI=0x11	Send Instruction 1A9B, read ECU information, command length is 2 bytes SN=1, this is the second DT of tester AR=0 Receiver's response is required EOM=1 Data transmission completed, only one frame
10	←300 B2	ACK	Indicate that tester instruction RS=1 is correctly received
11	←300 21 00 30 5A 9B 34 46 30 300 22 39 31 30 39 30 30 43 300 23 20 20 30 31 32 30 03 300 24 00 08 5F 07 EA 05 EC 300 25 04 74 4B 4F 4D 42 49 300 26 49 4E 53 54 52 2E 20 300 27 4D 37 33 20 48 31 33 ←300 18 20	ECU reply DT long message Len=0x30 48 valid bytes	SN=1 /EOM=0/ AR=1 No need to respond, data uncompleted SN=2 . . . . . SN=8 / EOM=1 /AR=0, data completed, need to respond
12	→751 B9	ACK	Tester to ECU respond frame RS=1
	751 12 00 04 31 B8 00 00 300 B3 300 29 00 10 71 B8 01 02 01 300 2A 03 01 04 01 06 01 07 300 1B 01 08 01 0C 751 BC	ECU reply 16-byte message	Read function select judge Instruction 31 B8 00 00
	→751 A3	CT	Connection Hold in idle, if no other instruction is sent from diagnosis, one CT command must be sent every 1S.
	←300 A1 0F 8A FF 4A FF	CA	ECU Connection response
	….		
	→751 A8	DC	Disconnect
	←300 A8	DC	Disconnect

5.3.9.2 TP20 Communication Setup Process Chart



5.3.9.3 Handling Procedure of CAN Module Automatic Communication Setup

1. Set CTXB6, Connection Setup instruction

2. Write CTXB5, send Channel Setup instruction, when INTF2<MTXF> is set ON, after transmission is completed, CAN module will wait for ECU response, if ECU gives no or error response within CTP_TE, CAN module will retransmit, if ECU gives no normal response when the number of retransmission is greater than MNTC, CAN module will set error flag CSNOTSUPP<INTF2> as ON, and clear INTF2<MTXF>. If interrupt is allowed, error interrupt will be generated.

3. After ECU normally responds to the instruction of Channel Setup, CAN module will automatically set the filter as TX_ID_P, and automatically set CTXB6 ID as RX_ID_P, user must set RX_ID_P as all transmit frame IDs in current module according to the respond frame RX_ID_P, (Id in automatic respond frame CTXB7 and ID that you intends for data transmission must be set RX_ID_P).

4. CAN module automatically starts CTXB6 transmission, (CS indicates Connection Setup)
5. CAN module waits for receive ECU to respond to CA.

   - If ECU normally responds to CA, CAN module will automatically set BS and T3 according to Can parameters, the multi-frame transmission is processed according to the automatically set BS and TS in multi-frame transmission. Communication Setup is completed, MTXF is automatically cleared, INTF<CTX> transmit completion flag is set ON. If interrupt is allowed, interrupt will be generated.

   - If ECU gives no or error response within CTP_TE, CAN module will retransmit CTXB5(Connection Setup), If ECU gives no normal response when the number of retransmission is greater than MNT, CAN module will set error flag CSNOTSUPP<INTF2> as ON, and clear INTF2<MTXF>. If interrupt is allowed, error interrupt will be generated, and INTF2<MTXF> is cleared.

6. CAN module may make other diagnoses after completion of Communication Setup.

After Communication Setup, if user makes no operation, Connection Hold must be done by sending CT(0xA3) within 1S, or ECU will break the connection channel, Channel Setup must be restarted if communication is intended to resume.

5.4 Application of Other CAN Protocols

- For J1939/ISO15765/VWTP2.0 ET7190 CAN control module with built-in response mechanism of transmission and reception of long message, when user operates on ET7190 via WINDOWS or other non-real-time ports, user can better handle it with no need to consider time.
- For message with the length over ET7190 automatic frame-to-frame transmission capacity, or application of any other protocol, user may need to consider the time of transmit frame. If ET7190 is connected using microcontroller(real-time system), user can transmit or receive CAN message in precise time, theoretically can be applied to any application based on CAN physical bus with no limit.
- For WINDOWS or any other non-real-time OS, if user directly connects ET7190 via serial port, user may not precisely control the time of message transmission/reception, some limit may occur when any other CAN range is applied. User should select UI mode according to the real use.

Chapter 6 Enhanced J1850 Module

J1850 bus is the standard issued by SAE(Society of Automotive Engineers), which is later applied in US vehicle plants, such as Ford, General Motor(GM), Chrysler. While most of US vehicle plants apply J1850 standard, instrument may vary from plant to plant, Ford's design of physical layer is different from that in GM, Chrysler. Though GM and Chrysler share the same physical layer, they have different forms of application layer, which means that these 3 plants have applied 3 protocols.
In J1850 bus line, message is transmitted in the form of digital signals, and the priority of dominant bits of digital signal is higher than that of recessive bits. When the bus line is occupied by the message of high priority, the message of low priority is caused to abort transmission, and the aborted message is only transmitted when the bus line is idle for the avoidance of information loss due to the message conflict. J1850 protocol uses full-frame comparison to determine if the message transmitted in bus line should be accepted or not, namely comparing from the start-of-frame bit by bit, until the frame ends. This approach will not damage the frame structure and content, this damage-free resolution of conflict also serves as the core of this protocol, so user may transmit data frame from time to time as CAN bus line, with no consideration of bus conflict. J1850 protocol doesn't precisely define the possible frame error. However, it takes CRC check method to detect the error frame. When the transmit site detects an error from its transmitted message, the transmission will be automatically interrupted. Upon the reception of this error frame, the site receiving this message will entirely abandon this frame. J1850 application layer is mainly used for the prompt of operation information and fault diagnosis. SAE has further expanded this protocol that has been written in J2178 protocol.
In respect of bus class of vehicle electronics, J1850 belongs to Class B bus, which runs at a rate about 20kbps~125kbps, comparatively, Class A bus runs at a lower rate below 20kbps, and K bus line belongs to Class A. Class B bus is primarily used for vehicle information center, instrument display, fault detection and diagnosis, etc.

As previously described, J1850 takes different approaches of physical layer, one is the data transmission in Pulse Width Modulated (PWM), in which 2 lines are used in the mode of differential arrangement for data transmission, up to 41.6kbps; the other is Variable Pulse Width(VPW), in which only 1 line is used for data transmission, up to 10.4kbps.

ET7190 J1850 module has the following features:

- 10 FIFO receive buffers
- 2 FIFO transmitters
- Fully support J1850 technical standard, support 10.4K VPW and 41.6KBPS PWM
- IFR1/IFR2/IFR3 in-frame respond and receive
- Support the data up to 10 bytes or IFR3 transmission
- All-frame error-free arbitration transmission and IFR,
- High-speed CRC hardware check
- Support Single Header/Consolidated Header
- Support Single Header IFR(in-frame respond)
- IFR1/IFR2 can simulate 8 physical ID responses
- Filtering of ID/Physical Addr./Functional Addr.
- Wake-up in reception of J1850 message

6.1 J1850 Message Format

Total length of J1850 message is no greater than 12 bytes(excluding special bits like SOF, EOD, EOF, IFS, NB). Header may be 1 bytes or 3 bytes long. J1850 defines Data as 0-8(10) bytes. CRC is 1 byte long. IFR is 0-8(10) bytes long, J1850 has 4 message formats, as shown in the figure below:

Type 0 - None

Type 1 - Single Byte From a Single Responder

Type 2 - Single Byte From Multiple Responders

Type 3 - Multiple Bytes From a Single Responder

Note: How to understand IFR: When the data frame is transmitted by sender, it only ends at EOD, IFR is the in-frame respond data by receiver, which is responded by receive module in accordance with configuration requirement. IFR = In Frame Respond

1. TYPE No IFR Data

   When Header contains 1 byte, ET7190 J1850 module allows the data with up to 10 bytes, plus CRC up to 12 bytes.

When Header contains 3 bytes, the data length is up to 8 bytes.

2. TYPE 1 IFR Data with Single Byte

   In this mode, there is only one IFR data, the responded ECU may be multiple, but the minimum ID obtains the bus, other IDs abort the IFR transmission after IFR is arbitration error.
   In this mode, ET7190 permits when Header contains 1 byte, the data length is up to 9 bytes, when Header contains 3 bytes, the data length is up to 7 bytes.

3. TYPE 2 contains IFR data with multiple bytes, IFR sender is the transmission of multiple ECUs, one-byte ID is transmitted by each ECU.

   In this mode, multiple ECUs simultaneously responds and transmits IFR data, bus arbitration is firstly obtained by smaller ID, along with the completion of transmission. All IDs with bigger value restart IFR transmission as soon as the previous ID is completely transmitted, until all responded IDs are completely transmitted. ET7190 J1850 module contains 1 byte in header, if the data is in 0 byte, up to 10 IFR bytes are allowed.

4. TYPE 3 contains IFR data with multiple bytes, IFR responder is 1 ECU module.
   IFR3 longest possible length ET7190 is also allows with10 bytes. If multiple modules simultaneously send IFR3, as full-frame comparison is made in J1850 module transmission, finally only IFR3 with the smaller value obtains the bus line. Actually it permits no simultaneous IFR3 data transmission and response on the same frame by multiple modules. This must be avoided in network configuration.

   This frame form is usually applied to the configuration with less changes or the transmission of state data in network, for example, Node A aims to obtain the configuration or state of Node B. A transmits a specific Header with the data length of 0, Node B firstly puts state data in IFR3 data buffer, when this data-free Header is received by J1850 module at Node B, IFR3 in-frame respond is directly made from J1850 controller. Actually CPU at Node B doesn't involve in the transmission at that time, thereby saving the occupation of bus bandwidth and accelerating the respond time. Meanwhile, CPU needn't involve in the data output.

   IFR3 may have one CRC byte, but it might not be used due to different manufacturers, this is only CRC check of data byte in IFR. IFR CRC is valid or invalid when ET7190 can be configured as TYPE3.

Note: Please refer to SAE J1850 standard text about J1850 bit definition, bus arbitration or other contents

6.2 J1850 Protocol Header Form

6.2.1 Single Byte Header

In this mode, one network only allows for this mode, no Consolidated Header should co-exist in single network. This mode contains 256-message ID identifiers in total, which is used for the earlier vehicle network, but it is rarely seen now. ID form is given below:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Message ID (256)

6.2.2 1 Byte Form of Consolidated Header

1 Byte, 3 Byte Form of consolidated header may simultaneously communication in the same network, which are differentiated according to (Bit4) H=1 in ID, 1 Byte Form contains 128 IDs, and each of these IDs is given below:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
x	x	x	H=1	x	x	x	x
Message ID (128)							

6.2.3 3 Byte Form of Consilidated Header

3 Byte Form H bit is always 0, Header content is given below:

BYTE 1								BYTE 2	BYTE 3
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Target Addr.	Source Addr.
P	P	P	H	K	Y	Z	Z		
Priority			0	Message Type					

Message Type—SAE J2178-1

Msg Type	KYZZ	Response(K)	Adde.(Y)	IFR Type	Message Type
0	0000	Required	Functional	2	Function
1	0001	Required	Functional	1	Broadcast
2	0010	Required	Functional	2	Function Query
3	0011	Required	Functional	3	Function Read
4	0100	Required	Physical	1	Node to Node
5	0101	Required	Physical	-	Reserved -MFG
6	0110	Required	Physical	-	Reserved -SAE
7	0111	Required	Physical	-	Reserved -MFG
8	1000	Not Allowed	Functional	0	Function Command/Status
9	1001	Not Allowed	Functional	0	Function Request/Query
10	1010	Not Allowed	Functional	0	Function Ext. Command/Status
11	1011	Not Allowed	Functional	0	Function Ext. Request/Query
12	1100	Not Allowed	Physical	0	Node to Node
13	1101	Not Allowed	Physical	0	Reserved -MFG
14	1110	Not Allowed	Physical	0	Acknowlegement
15	1111	Not Allowed	Physical	0	Reserved -MFG

6.3 J1850 Module Register

J1850 module transmits and read data via special registers, when J1850 module is operated by user. The special registers are listed below:

Name	Addr.	Description	Reset Value
JERRF	0x04	J1850 Error Register	0x00
JERRIE	0x05	Error Interrupt Flag Enable Register	0x00
IFR3B0LEN	0x06	IFR Type 3 Buffer 0 Data Length	0x00

IFR3B1LEN	0x07	IFR Type 3 Buffer 1 Data Length	0x00
JCON0	0x08	J1850 Control Register 0	0x02
JCON1	0x09	J1850 Control Register 1	0x7E
JSTAT	0x0A	J1850 State Register	0x00
YZZ0	0x50	Correspond to 8 bytes in YZZCON register set Register writes only, read and use read buffer command once, read 8 bytes	0x00
YZZ1	0x51		0x00
YZZ2	0x52		0x00
YZZ3	0x53		0x00
YZZ4	0x54		0x00
YZZ5	0x55		0x00
YZZ6	0x56		0x00
YZZ7	0x57		0x00

6.3.1 JERRF Error Flag Register

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
FARB	FNACK	FSOF	FBIT	FCRC	FOV	FBUS	FLEN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **FARB**： Arbitration Error Flag Bit
1 = Arbitration error generated, in transmission of data frame, if the arbitration error continuously occurred in the module over the times data set in JCON1<ARBCNT>, FARB is set ON. Data transmission is aborted.
0 = Arbitration error not generated

bit 6 **FNACK**： IFR Error Flag
1 = Error generated, when IFR frame required for reception is transmitted, no IFR is received.
0 = Error not generated

bit 5 **FSOF**： SOF Error Flag Bits
1 = Error start bits received
0 = Error not generated

bit 4 **FBIT**： Error Bit Flag (Error will be generated in reception of BRK or SOF when data frame is accepted)
1 = Error bit received
0 = Error not generated

bit 3 **FCRC**： CRC Error Flag Bits
1 =Message abandoned in reception of message CRC error
0 = Error not generated

bit 2 **FOV**： Receive Buffer Overflow Flag Bits
1 = Receive Buffer FIFO Overflow
0 = Error not generated

bit 1 **FBUS**：Bus Error Flag Bits
1 = Bus-to-ground or short circuit or bus transreceiver error
0 = Error not generated
bit 0 **FLEN**：Frame Long Flag Bits
1 =Total length of the received frame exceeds 12 bytes
0 = Error not generated

6.3.2 JERRIE Error Interrupt Enable Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARBIE	NACKIE	SOFIE	BITIE	CRCIE	OVIE	BUSIE	LENIE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ARBIE**：Arbitration Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 6 **NACKIE**：IFR Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 5 **SOFIE**：SOF Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 4 **BITIE**：Bit Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 3 **CRCIE**：CRC Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 2 **OVIE**：Receive Buffer Overflow Error Interrupt Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 1 **BUSIE**：Bus Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt
bit 0 **LENIE**：Frame Long Error Interrupt Enable Bits
1 = Enable Error Interrupt
0 = Disable Interrupt

6.3.3 IFR3B0LEN IFR Type3 Data Buffer 0 Valid Length

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x	R/W-0
IFR3B0LEN<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- IFR3B0LEN maximum value is 10, and total length of data frame must not be greater than 12 bytes.
- If IFR3 contains CRC check, this length contains CRC byte. Data written in buffer may not write CRC, CRC value of IFR3 data is automatically set by J1850 module.

6.3.4 IFR3B1LEN IFR Type3 Data Buffer 1 Valid length

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x	R/W-0
IFR3B1LEN<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- IFR3B1LEN maximum value is 10, and total length of data frame must not be greater than 12 bytes.
- If IFR3 contains CRC check, this length contains CRC byte. Data written in buffer may not write CRC, CRC value of IFR3 data is automatically set by J1850 module.

6.3.5 JCON0 Control Register 0

R/W-0	U-0	R-0	U-0	U-0	U-0	R/W-1	R/W-0
ON	-	CFG	-	-	-	IFR3CRCEN	HEAD
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ON**： J1850 Module Enable
1 = Enable J1850
0= Disable

bit 5 **CFG**：Module Configuration State
1 = Module in Configuration
0 = Module Configuration Completed
bit 1 **IFR3CRCEN**：CRC in IFR3
1 = CRC byte contained in IFR data field
0 = No CRC byte contained in IFR data field
bit 0 **HEAD**：Header Form
1 = Single Header
0 = One byte or three byte of Consilidated Header

6.3.6 JCON1 Control Register 1

R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0
MOD	RETIFR0	RENACK	RETACK	ARBCNT<1:0>		NAKCNT<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend：C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **MOD**：J1850 Module Work Physical Mode
1 = PWM 41.6KBPS When the module drives PWMTX/PWMRX pin
0 = VPW 10.4KBPS When the module drives VPWTX/VPWRX pin
bit 6 **RETIFR0**：In data transmission, transmit frame with no need of IFR is returned to the receive buffer.
1 = Transmit Frame Return
0 = No Return
bit 5 **RETNACK**：In data transmission, transmit frame with respond error is returned to the receive buffer.
1 = Transmit Frame Return
0 = No Return
bit 4 **RETACK**：In data transmission, transmit frame with IFR is returned to the receive buffer.
1 = Transmit Frame Return
0 = No Return

bit 3:2 **ARBCNT**：Retransmission Number of Arbitration Error

In simultaneous transmission of data frame by multiple modules, data frame with lower priority will lose arbitration, when the transmission is aborted. Transmission is restarted when the module with higher priority completes the transmission. If transmission is unable when the number of restart reaches the set number of ARBCNT, the module will abort the transmission, error flag FARB is set. If interrupt is allowed, error interrupt will be generated.

11 = Transmit 4 times,
10=Transmit 3 times,
01=Transmit 2 times,
00=Only transmit once,

Bit1:0 **NACKCNT**：Retransmission Number when IFR is not received.
If IFR is required, if no IFR is received after the number of retransmit NACKCNT is set, the error flag FNACK will be set. The transmission of data frame is completed. INTF<JTX> is not set ON.

11 = Transmit 4 times,
10= Transmit 3 times,
01= Transmit 2 times,
00= Only transmit once,

Remark: When J1850 module is operated, reception and transmission are processed simultaneously, the transmitted data frame can be simultaneously accepted in FIFO receive buffer, the transmitted data frame can be set to return or not depending on user needs. For the transmit frame requiring the reception of IFR, user can set RETACK as ONby which user can process IFR value. RETNACK and RETIFR0 can be generally set Clear, with no need of reception.

6.3.7 JSTAT State Register

R/W-0	R/W-0	U-0	R-0	U-0	R-1	R-0	R-0
CRXBF	CTXBF	-	TXF	-	IFS	BUSY	TRMT
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **CRXBF**：Clear FIFO Receive Buffer
1 =Execute to clear FIFO receive buffer, automatically clear after removal
0 = No Execute
bit 6 **CTXBF**：Clear Transmit Buffer
1 = Execute to clear FIFO receive buffer, automatically clear after removal
0 = No Execute

bit 4 **TXF**：J1850 Transmit Buffer full, read only
1 = J1850 Transmit Buffer full
0 = J1850 Transmit Buffer empty
bit 2 **IFS**：Bus Release Flag, read only
1 =Bus released, enable data transmission
0 =Bus unreleased
bit 1 **BUSY**：Transmitter busy, read only
1 =Transmitter busy
0 = Transmitter idle
bit 0 **TRMF**：Transmit Shift Register full flag bits, read only
1 =Transmit shift register full
0 = Transmit shift register empty

6.3.8 YZZi Control Register i=0-7

- Valid only in data frame of 3 byte form of Consolidated Mode
- Control register YZZ0-YZZ7 corresponds to Byte 0-7 of YZZCON in control register set, different from writing YZZCON, YZZi permits fast writing of 1 byte, only with1 byte changed. YZZCON needs to write 8 bytes in one time. YZZi only writes rather than reads. Normally, YZZCON setting is

not required to modify after control setting is completed. But when IFR3 in-frame respond data is required to modify, IFR3SEL setting should be changed, and data must be switched between 2 IFR3B0s and IFR3B1 buffer, rapidly processing when YZZi is used and written.

- Function of YZZi Control Register:
  In data frame of 3 byte form of Consolidated Mode, if J1850 module receives 1 byte data, according to the YZZ value(000-111) of in-frame Byte 1 ID corresponding to YZZi controller setting, if K=0 for this ID, J1850 module processes IFR for this data frame, and the responded content is set as per YZZi.

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	NB	IFRT<1:0>		IFR3SEL	IDSEL<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 6 **NB**：JCON1<MOD>=1, in VPW mode, NB bit. This bit is neglected when JCON1<MOD>=1, PWM.
1 = NB is Active short Pulse
0 = NB is Active long Pulse

bit 5:4 **IFRT<1:0>**：IFR Type Mode.
11 = Respond in IFR type 3
10 = Respond in IFR type 2
01 = Respond in IFR type1
00 = IFR type0 No respond

bit 3 **IFR3SEL**：In IFR Type3 mode, select the data buffer
1 =Select data in IFR3B1 buffer to transmit
0 = Select data in IFR3B0 buffer to transmit

bit 2:0 **IDSEL<2:0>**：In IFR Type1/2 mode, select the byte in JIDBUFF buffer
111=Select Byte 7 in JIDBUFF buffer as ID transmission
110= Select Byte 6 in JIDBUFF buffer as ID transmission
101= Select Byte 5 in JIDBUFF buffer as ID transmission
100= Select Byte 4 in JIDBUFF buffer as ID transmission
011= Select Byte 3 in JIDBUFF buffer as ID transmission
010= Select Byte 2 in JIDBUFF buffer as ID transmission
001= Select Byte 1 in JIDBUFF buffer as ID transmission
000= Select Byte 0 in JIDBUFF buffer as ID transmission

6.4 J1850 Module Control Register Set (Buffer)

Name (Buffer)	Addr.	Length	Description	Reset Value
YZZCON	0x00	8	3 byte Header IFR control register set	00
IDIFRCON	0x01	256	In Single Header or one byte header of Consilidated Mode, IFR control register set, write-use long buffer write command	00
HDFILTER	0x02	256	Receive filter control register set, write-use long buffer write command	07
IDBUFF	0x03	8	In IFR type 1/2 mode, the responded ID buffer	xx
IFR3B0	0x04	10	In IFR Type 3 mode, the responded data buffer 0	xx
IFR3B1	0x05	10	In IFR Type 3 mode, the responded data buffer 1	xx
JTXB	0x06	16	J1850 transmit buffer	xx
JRXB	0x3D	16*10	J1850 FIFO receive buffer, FIFO depth 10 frames	xx

6.4.1 YZZCON 3 byte Header IFR Control Register Set

- The same as description of Control Register YZZi (i=0-7)

- IFR Respond of YZZ=000 in YZZCON[0] Control ID
  IFR Respond of YZZ=001 in YZZCON[1] Control ID
  IFR Respond of YZZ=010 in YZZCON[2] Control ID
  IFR Respond of YZZ=011 in YZZCON[3] Control ID
  IFR Respond of YZZ=100 in YZZCON[4] Control ID
  IFR Respond of YZZ=101 in YZZCON[5] Control ID
  IFR Respond of YZZ=110 in YZZCON[6]Control ID
  IFR Respond of YZZ=111 in YZZCON[7] Control ID

6.4.2 IDIFRCON ID IFR Conrol Register Set

Valid only in Single Header or 1 byte Header of Consilidated Header
When J1850 message header has only 1 byte, IFR in reception is controlled by this register. If it is only applied in diagnosis, all bytes of IDIFRCON can be cleared, because the frame of single header doesn't require IFR respond in diagnosis.

IFR Respond of IDIFRCON [0] Control ID=0x00
IFR Respond of IDIFRCON [0] Control ID=0x00
IFR Respond of IDIFRCON [0] Control ID=0x00
……
IFR Respond of IDIFRCON [0] Control ID=0x00
IFR Respond of IDIFRCON [0] Control ID=0x00

IDIFRCONi (i=0-255) Control Byte Form

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	NB	IFRT<1:0>		IFR3SEL	IDSEL<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 6 **NB**：JCON1<MOD>=1, in VPW mode, NB bit. This bit is neglected when JCON1<MOD>=1,PWM
1 = NB is Active short Pulse
0 = NB is Active Long Pulse

bit 5:4 **IFRT<1:0>**：IFR Type Mode
11 = Respond in IFR type3 mode
10 = Respond in IFR type2 mode
01 = Respond in IFR type1 mode
00 = IFR type0 No respond

bit 3 **IFR3SEL**：In IFR Type3 mdoe, select the data buffer
1 =Select the data in IFR3B1 buffer to transmit
0 = Select the data in IFR3B0 buffer to transmit

bit 2:0 **IDSEL<2:0>**：In IFR Type1/2 mode, select the byte in JIDBUFF buffer
111= Select Byte 7 in JIDBUFF buffer as ID transmission
110= Select Byte 6 in JIDBUFF buffer as ID transmission
101= Select Byte 5 in JIDBUFF buffer as ID transmission
100= Select Byte 4 in JIDBUFF buffer as ID transmission
011= Select Byte 3 in JIDBUFF buffer as ID transmission
010= Select Byte 2 in JIDBUFF buffer as ID transmission
001= Select Byte 1 in JIDBUFF buffer as ID transmission
000= Select Byte 0 in JIDBUFF buffer as ID transmission

6.4.3 HDFILTER Receive Filter Control Register Set

- When message is in single header: According to ID
  If HDFILTER[ID] IDEN=1, this message is accepted.

- When message in 3 byte form(H=0): According to Byte 1 Bit Y and Byte 2 Target Addr. of header
  If Y=0(Physical Addr.). If HDFILTER[addr] FUNEN=1, this message is accepted.
  If Y=1(Physical Addr.). If HDFILTER[addr] PHYEN=1, this message is accepted.

- HDFILTER can implement:
  Separate filtering of 256 IDs of message in single header.
  Separate filtering of 256 Functional Addr., 256 Physical Addr. of message in 3 byte form.

HDFILTERi (i=0-255) Byte Form

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1
-	-	-	-	-	FUNEN	PHYEN	IDEN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 2 **FUNEN**： Message with Receive Functional Addr. = i(Y=0, Functional Addr. filtering of 3 byte header)
1 =Accept
0 =Reject
bit 1 **PHYEN**： Message with Receive Physical Addr. = i(Y=0, Physical Addr. filtering of 3 byte header)
1 =Accept
0 =Reject
bit 0 **IDEN**： Receive message with ID=i(H=1 or Single Header)
1 =Accept
0 =Reject

6.4.4 IDBUFF IFR Type1/2 Message IFR ID Buffer

R/W-xx	R/W-xx	R/W-xx	R/W-xx	R/W-xx	R/W-xx	R/W-xx	R/W-xx
ID0	ID1	ID2	ID3	ID4	ID5	ID6	ID7
Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7

- Buffer length is 8 bytes, 8 IDs can be informed.
- If J1850 module requires IFR1/IFR2, one ID is selected for IFR according to the message feature and IDSEL.

6.4.5 IFR3B0 IFR3 Data Buffer 0

It is used to store the data responded from IFR Type 3, with the length up to 10 bytes, its valid length is determined by IFR3B0LEN, IFR3B0 buffer data must be firstly written before writing IFR3B0LEN. CRC is reset according to IFR TYPE3 transmit length when J1850 module writes the length register. IFR3B0LEN must be rewritten after each change of buffer data.
Note: When IFR3 data varies, incomplete data may occur for the avoidance of IFR3 data change when the data frame responds. Two alternate buffers must be used to change data.

6.4.6 IFR3B1 IFR3 Data Buffer 1

It is used to store the data responded from IFR Type 3, with the length up to 10 bytes, its valid length is determined by IFR3B1LEN, IFR3B1 buffer data must be firstly written before writing IFR3B1LEN. CRC is

reset according to IFR TYPE3 transmit length when J1850 module writes the length register. IFR3B1LEN must be rewritten after each change of buffer data.
Note: When IFR3 data varies, incomplete data may occur for the avoidance of IFR3 data change when the data frame responds. Two alternate buffers must be used to change data.

6.5 J1850 Receive/Transmit Buffer Structure

The size of receive and transmit buffer of J1850 message is 16 bytes. Transmit and receive buffer has the same structure definition of 16 bytes. Transmit buffer is JTXB, while receive buffer JRXB is 10F*16 FIFO receive buffer.

Table of Transmit/Receive Buffer Structure

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DIR	-	-	-	HEADTYPE	NB	-	FVOP
1	DATALEN<3:0>				IFRLEN<3:0>			
2	-							
3	PT<3:0>				-	-	-	-
4	DATA0							
5	DATA1							
6	DATA2							
7	DATA3							
8	DATA4							
9	DATA5							
10	DATA6							
11	DATA7							
12	DATA8							
13	DATA9							
14	DATA10							
15	DATA11							

DATA<11:0>： Store the complete frame data with the length up to 12 bytes

PT<3:0>： Protocol Flag
This value is neglected in writing JTXB
When JRXB buffer is read, PT is always equal to 2, represented as J1850 message, ET7190 is user-defined. No special meaning

IFRLEN<3:0>： IFR length(If IFR3 contains CRC, the length includes IFR3 CRC byte)

This value is neglected when the transmitted data writes JTXB, if JRXB is read, the length of IFR data accepted in frame

DATALEN<3:0>： Frame data length, including Header byte length, excluding IFR length, maximum value is 12

DIR： Direction Bits

This bit is neglected in writing JTXB
DIR=0 represents output frame(output return frame) when JRXB buffer is read, DIR=1 represents this frame as receive frame

HEADTYPE：This bit is neglected in writing JTXB
When JRXB buffer is read
0 = one byte header or three byte of Consilidated header
1 =Single header

NB：NB bit in IFR respond of VPW mode (This bit is always 0 in PWM)
This bit is neglected in writing JTXB(NB value is one bit transmitted when receiver processes IFR. Sender will not send a NB bit)

When JRXB buffer is read, represent NB bit in IFR

1 =NB is Active short Pulse
0 = NB is Active long Pulse

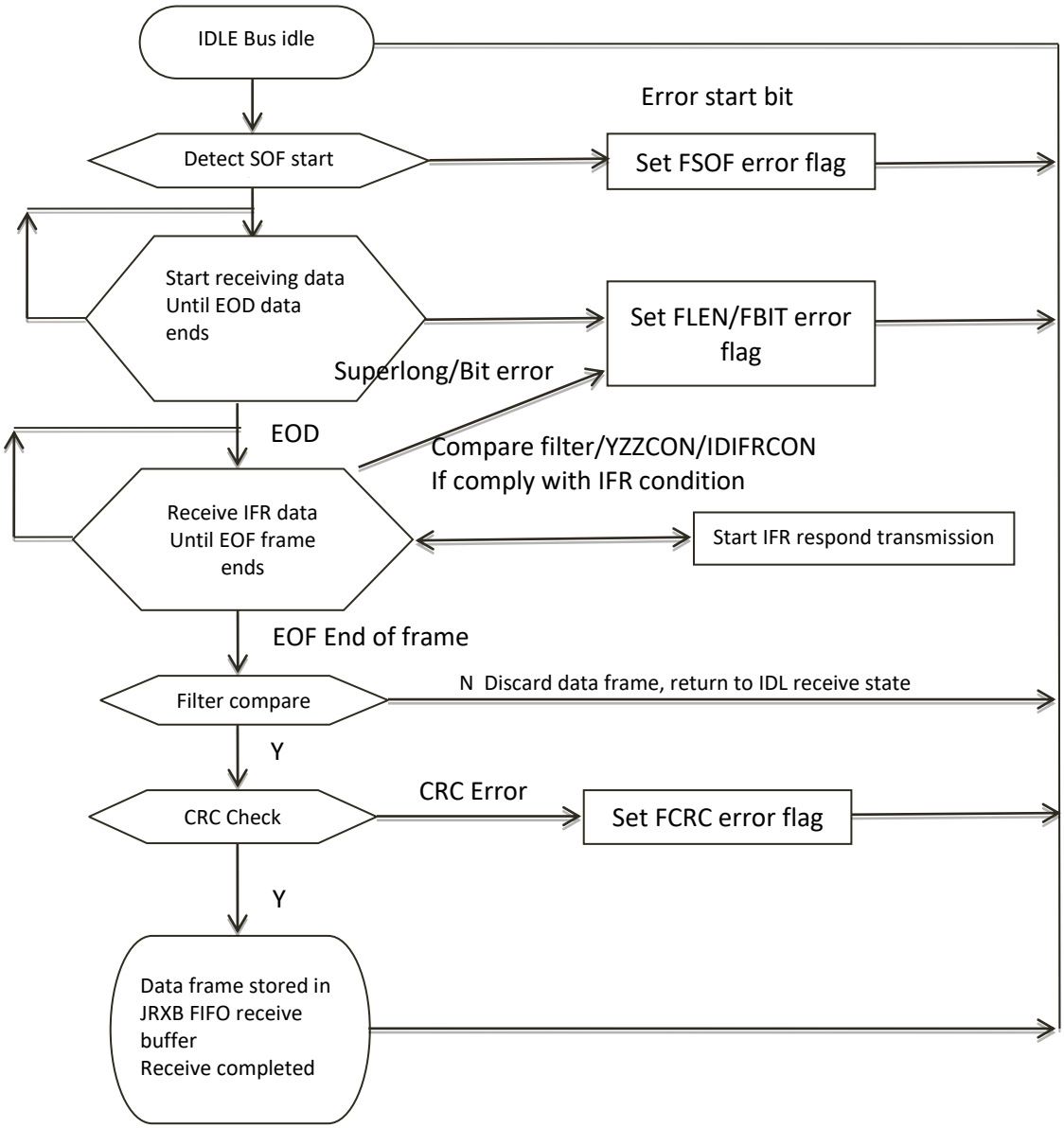
FVOP：Total Physical Mode Flag, this bit is neglected in writing JTXB(Flag of VPW or PWM)
When JRXB buffer is read
In VPW mode, FVOP is always equal to 0
In PWM mode, FVOP is always equal to 1
The following message structure is defined in C

```
typedef struct
{    union
     {
          UINT32  u32Info;
      struct
      {
        unsigned char FVOP            : 1;   //VPW or PWM Flag
        unsigned char            : 1;   //Reserved
        unsigned char NB              : 1;   //VPW_NB Bit
        unsigned char HEADTYPE      : 1;
        unsigned char                 : 3;   //Reserved
              unsigned char   DIR             : 1;   //Direction

              unsigned char          DATA_Len       : 4;   // Data Length
           unsigned char          IFR_Len          : 4;   // IFR Length
              unsigned char                              ;    //Reserved Byte
              unsigned char                      : 4;    //Reserved
              unsigned char          PT              : 4;
            }FrameInfo;
      };
      unsigned char     Data[12];
}  J1850_DATAPACKAGE,*LP J1850_DATAPACKAGE;
```

6.6 Receive J1850 Message

6.6.1 J1850 Module Receive Message Process



INTF<JRX> is set ON after reception, if interrupt is allowed, receive interrupt will be generated, user can read FIFO buffer data, if FIFO is completely read empty, JRX is automatically cleared. If interrupt is ususued, user can inquire INTF<JRX> flag, J1850 FIFO receive buffer data is read.

6.6.2 IFR in Reception of J1850 Message

1. When J1850 module operates, bus state is monitored, receiver will accept each frame data(including the transmitted data frame), if data frame is received, when sender completes the delivery of data(receiver accepts EOD bit), the module will determine if this is the data frame required by this module according to filter setting. If so, IFR respond or no respond is selected depending on IDIFRCON or YZZCON setting.

2. When JCON0<HEAD>=1, the module is operated in Single Header mode, all data frames is in Single Header, as shown in Fig.6.2.1, the module processes IFR respond according to the received Message ID and IDIFRCON configuration.

IDIFRCON[ID]<IFRT>=0 TYPE 0 No respond, (TYPE=0 usually prevails in single header, rather than IFR1/2/3)
IDIFRCON[ID]<IFRT>=1 TYPE 1 Respond value is Byte IDIFRCON[ID]<IDSEL> in IDBUFF.
IDIFRCON[ID]<IFRT>=2 TYPE 2 Respond value is Byte IDIFRCON[ID]<IDSEL> in IDBUFF.
IDIFRCON[ID]<IFRT>=3 TYPE 3 Respond, respond value IDIFRCON[ID]<IFR3SEL>, the selected IFR3B0 or IFR3B1 buffer.
If the physical connection is VPW. IFR1/2/3 respond NB value is IDIFRCON[ID]<NB>, if multiple modules simultaneously respond, the setting of NB=0 shows higher priority than that of NB=1. NB=0 module obtains the bus line. PWM format has no NB bit.

3. When JCON0<HEAD>=0, the module is operated in Consolidated Mode, when the header form is determined by Byte 1 ID H Bit<Bit4> of data frame.

- When H=1, as One byte header form. IFR respond is the same with 2, the module makes IFR respond according to the received Message ID and IDIFRCON configuration. (Up to 128 IDs)
- When H=0, as 3 byte header form. When K<bit3>=1 in the received ID, howeverYZZCON is configured, the module will not make IFR respond. Handled as TYPE0 mode as per Standard J1850.
- When H=0, as 3 byte header form. In reception of K<bit3>=0 in ID, this module makes IFR respond according to YZZCON configuration. (n is YZZ bit value 0-7 in Message ID)
  YZZCON[n]<IFRT> =0 TYPE 0 No respond.
  YZZCON[n]<IFRT> =1 TYPE 1 Respond value is Byte YZZCON[n]<IDSEL> in IDBUFF.
  YZZCON[n]<IFRT> =2 TYPE 2 Respond value is Byte YZZCON[n]<IDSEL> in IDBUFF.
  YZZCON[n]<IFRT> =3 TYPE 3 Respond, respond value YZZCON[n]<IFR3SEL>, the selected IFR3B0 or IFR3B1 buffer.
  If the physical connection is VPW. IFR1/2/3 respond NB value is YZZCON[n]<NB>, if multiple modules simultaneously respond, the setting of NB=0 shows higher priority than that of NB=1. NB=0 module obtains the bus line. PWM format has no NB bit.

6.6.3 Setting Procedure of User IFR3 Data Buffer

When IFR3 data changes, for the avoidance of data frame in respond, IFR3 data is changed with the occurrence of incomplete data. Two alternate buffers IFR3B0, IFR3B1 must be used to change data. This is usually used for the configuration with few change or state data transmission in network. (This message form will not used in diagnosis)

1. IFR3B0 data and IFR3B0LEN valid length are written.
2. IFR3SEL in YZZCON(IDIFRCON) Register is set Clear(IFR3B0 data will be used as respond data if data frame is received and IFR3 is required to respond).
3. If IFR3 respond data is required to update, IFR3B1 data and IFR3B1LEN valid length are well set.
4. If IFR3SEL in YZZCON(IDIFRCON) Register is set ON(IFR3B1 data will be used as respond data if data frame is received and IFR3 is required to respond).
5. If update is required, execute procedure is switched to IFR3B0, the buffer IFR3B0, IFR3B1 is used as IFR3 respond data by repeating Step 1-5.

6.6.4 Listen Only Bus Data Frame

J1850 module has no specific Listen Mode, YZZCON and IDIFRCON are just set clear without impact on bus work, if it is required to listen the bus data, no IFR respond will be made in reception of data frame. All or partial data is received by setting the filter HDFILTER. J1850 module can receive any desired frame data(including IFR information between other modules).

6.7 Transmit J1850 Message

6.7.1 Transmit Message Flow

User application program just writes J1850 in message transmit buffer JTXB if message is intended to transmit via J1850 bus. After it is written in buffer, JSTAT<TXF> buffer full flag will be set ON. The module will operate as follows:

- J1850 module will start the message transmission when the bus release(IFS) is detected.
- In EOF state, if SOF start signal is transmitted from other module, the module will synchronize the transmission, when at least 2 modules transmit at the same time, in case of arbitration error, the module will abort the transmission, current data frame is received, and retransmission is done until the bus release.
- If the number of arbitration failure equals to the number of user-defined JCON1<ARBCNT>, the module will abort the transmission. Error flag JERRF<FARB> is set ON, if interrupt is allowed, error interrupt will be generated.

- If the transmitted message is the IFR-responded message required for receiver, if no IFR is received in transmission, retransmission will be done. If the number of retransmission equals to the number of user-defined JCON1<NAKCNT>, IFR is still not received, the module will abort the transmission. Error flag JERRF<FNACK> is set ON, if interrupt is allowed, error interrupt will be generated.

Whether the transmitted message requires the response or not, it is determined by YZZCON or IDIFRCON setting.

- If the message transmission is normally completed, transmit complete flag bit INTF<JTX> is set ON, if interrupt is allowed, transmit complete interrupt will be generated.

6.7.2 Abort Transmit Message

If message is stored in JTXB transmit buffer, the transmission is not started, transmission is aborted when JSTAT<CTXBF> is set ON, JSTAT<TXF> is automatically cleared, if the transmission is started, message will not be aborted.

6.7.3 Transmit Example

e.g.: One 68 6A F1 01 00 request is required to transmit

```
J1850_DATAPACKAGE J1850SendPackage;      //Define one message buffer
J1850SendPackage.u32Info=0;                    //All bits are cleared,
J1850SendPackage.FrameInfo.DATA_Len=6;    //Length is 6 bytes, must contain CRC length
J1850SendPackage.Data[0]=0x68;
J1850SendPackage.Data[1]=0x6A;
J1850SendPackage.Data[2]=0xF1;
J1850SendPackage.Data[3]=0x01;
J1850SendPackage.Data[4]=0x00;
SpiWriteRegisterBuff(JTXB, 16,(unsigned char *) &J1850SendPackage); //Write JTXB buffer to start the transmission
```

6.8 Configuration in OBD2 Diagnosis

6.8.1 OBD2 Mode VPW

In OBD2 diagnosis, the tester request application protocol header form in J1850 VPW Protocol is 68 6A F1.

Byte 1 0x68 is ID, wherein,
<Bit7:5>=3 Represent Priority(0b011)
<Bit4>H=0 Represent 3 Byte Header.
<Bit3>K=1 Represent no need of IFR respond.
<bit2>Y=0 Represent Functional Addr.
<Bit3:0>KYZZ=1000 Represent Message Type TYPE0 as per 6.2.3

Byte 2 6A is Target Addr. of functional request
Byte 3 F1 is Physical Addr. of tester
ECU responded protocol header form is 48 6B xx(xx is ECU transmitted Physical Addr.)
Byte 1 0x68 is ID, wherein,
<Bit7:5>=2 Represent Priority(0b010)
<Bit4>H=0 Represent 3 Byte Header.
<Bit3>K=1 Represent no need of IFR respond.
<bit2>Y=0 Represent Functional Addr.
<Bit3:0>KYZZ=1000 Represent Message Type TYPE0 as per 6.2.3
Byte 2 6A is Target Addr. of functional request
Byte 3 xx is ECU Physical Addr. (multiple ECU responds might be used)

6.8.2 OBD2 Mode PWM

In OBD2 diagnosis, the tester request application protocol header form in J1850 PWM Protocol is 61 6A F1.
Byte 1 0x61 is ID, wherein,
<Bit7:5>=3 Represent Priority(0b011)
<Bit4>H=0 Represent 3 Byte Form.
<Bit3>K=0 Represent no need of IFR respond.
<bit2>Y=0 Represent Functional Addr.
<Bit3:0>KYZZ=0001 Represent Message Type TYPE1 as per 6.2.3
Byte 2 6A is Target Addr. of functional request
Byte 3 F1 is Physical Addr. Of tester
ECU responded protocol header form is 41 6B xx(xx is ECU transmitted Physical Addr.)
Byte 1 0x41 is ID, wherein,
<Bit7:5>=2 Represent Priority(0b010)
<Bit4>H=0 Represent 3 Byte Form.
<Bit3>K=0 Represent no need of IFR respond.
<bit2>Y=0 Represent Functional Addr.
<Bit3:0>KYZZ=0001 Represent Message Type TYPE1 as per 6.2.3
Byte 2 6B is Target Addr. of functional request
Byte 3 xx is ECU Physical Addr.(Multiple ECUs might be used)

6.8.3 Configuration J1850 Module Control Register(Set)

When OBD2 diagnosis is in J1850 bus line, whether in VPW or PWM mode, the definition of message type conforms to Standard SAE J2178-1, because ECU-responded Functional Addr. is also the same(0x6B), so we just keep the same configuration. General setting is provided below:

- According to the message type in 6.2.3, YZZCON control register set is set. Because only PWM in OBD2 applies the response when YZZ=1, so we only set YZZCON[1], all other settings are cleared(IFR respond in no reception). In VPW, because K=1 in ID requires no IFR respond, J1850 module doesn't respond to K=1 message.

YZZCON[0]=0
YZZCON[1]=0x10 (IFRT=1 IDSEL=0 IFRSEL=0, this is the respond setting when YZZ=001)
YZZCON[2]=0
YZZCON[3]=0

YZZCON[4]=0
YZZCON[5]=0
YZZCON[6]=0
YZZCON[7]=0

Detailed operation command, CMD_WR_REG can be used to write into YZZCONi register one by one, or CMD_WR_BUFF is written to the buffer in one time.

- Set IDBUFF
  IDBUFF[0]=0xF1 as the Physical Addr. of tester makes IFR1 respond. IDBUFF[1-7] value is irrelevant, because YZZCON[1] IDSEL selects 0 ID byte in the buffer, while others will not be applied.

- Because IFR3B0, IFR3B1 in no use, so there is no need to set.

- Set Filter(Only data frame with Functional Addr. 0x6B is received)
  All IDENs in HDFILTER[0 -255] are cleared, because no data frame with single header will be received.
  All PHYENs in HDFILTER[0 -255] are cleared, because no Physical Addr. will be received.
  All FUNENs in HDFILTER[0 -255] are cleared, then FUNEN in HDFILTER[0x6B] is set ON. Only message with Functional Addr. as 0x6B is received.

For the setting of operation command of filter, CMD_FILL_BUFF_BYTE can be used to Fill all HDFILTER register sets with 0, then CMD_WR_BUFF_BYTE command is used to write 0x6B address buffer value in 0x04(bit2 FUNEN=1). 2 commands are used to finish the setting of filter.

- Set JCON1
  In VPW, JCON1=0x0E is set
  In PWM, JCON1=0x8E is set

- Set JCON0 Start J1850 Module(JCON0<HEAD> must be cleared in OBD2)
  JCON0=0x80 Start Module, JCON0=0x00 stops the module work
- Transmit J1850 OBD2 Request
  e.g: MOD:01 PID:00 PWM Mode
  (FrameInfo:00 06 00 00) 61 6A F1 01 00 is written into JTXB (DATA_LEN=6, any value of CRC Byte) Start RTS.

6.8.4 GM Vehicle SRS System Example(VPW)

To better understand J1850 configuration, another example of GM SRS using VPW diagnostic mode is illustrated.

3 byte header of tester is 6C 58 F1 , ECU responded 3 byte header is 6C F1 58. 0x58 SRS safety bag Physical Addr., 0xF1 is Physical Addr. Of tester.(Generally Physical Addr. is used for the diagnosis of vehicle ABS and other ECUs)
Tester, SRS transmitted data frame ID is 6C, wherein ID

<Bit7:5>=3 Represent Priority(0b011)
<Bit4>H=0 Represent 3 byte header.
<Bit3>K=1 Represent no need of IFR (No IFR is made by the module when K=1).
<bit2>Y=1 Represent Physical Addr.
<Bit3:0>KYZZ=1100 Represent the message type TYPE0 as per 6.2.3

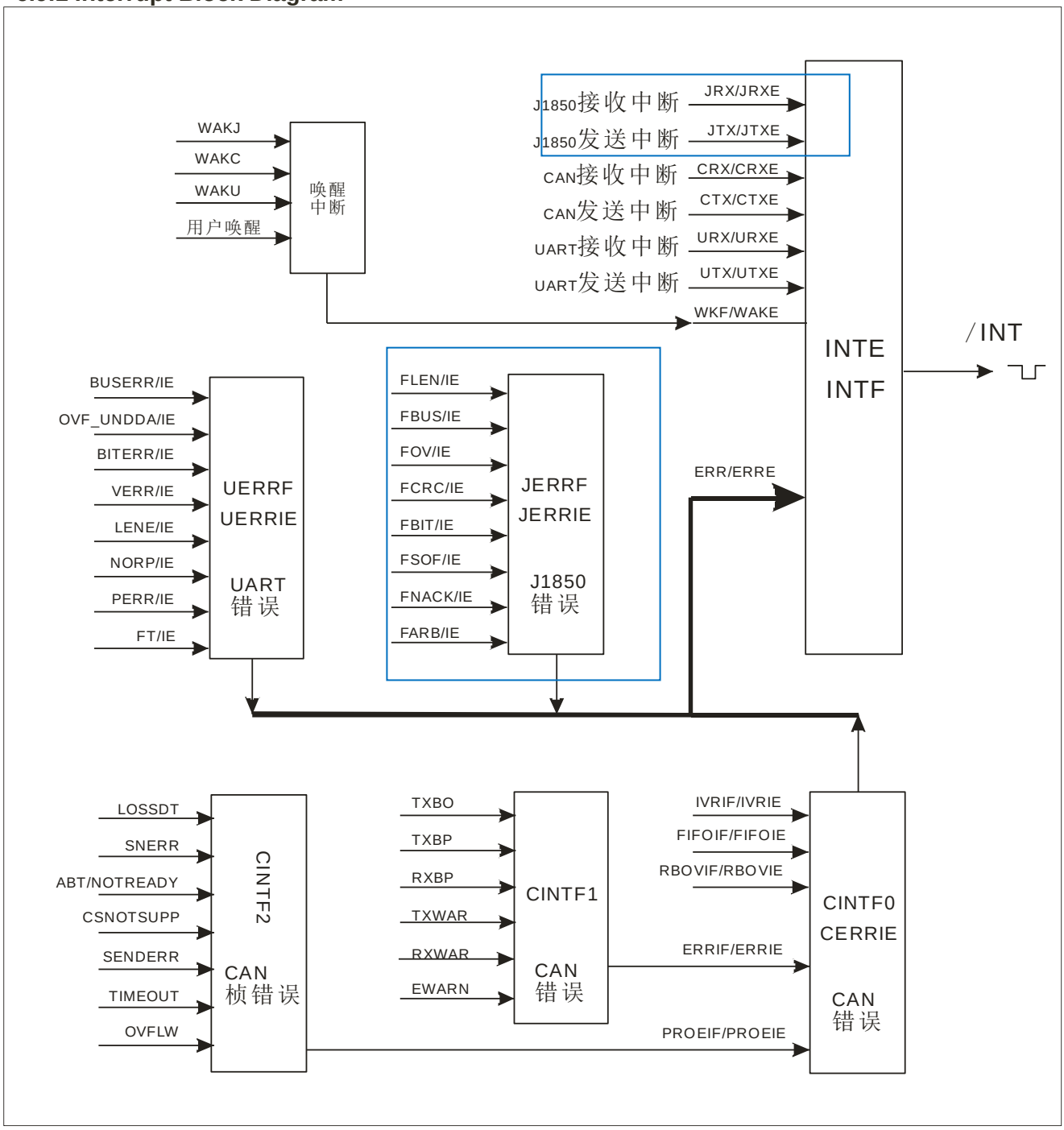
All processes of setting are defined in 6.8.3, it is only required to change the filter to only receive the data frame with Physical Addr. of 0xF1:

For the setting of operation command of filter, CMD_FILL_BUFF_BYTE can be used to Fill all HDFILTER register sets with 0, then CMD_WR_BUFF_BYTE command is used to write 0xF1 offset address buffer value in 0x02(bit1 PHYEN=1). 2 commands are used to finish the setting of filter.

Send the request to read TroubleCodes(19 02 FF 00):
(00 08 00 00) 6C 58 F1 **19 02 FF 00** is written into JTXB (DATA_LEN=8, any value of CRC Byte) Start RTS. So ECU returned TroubleCodes can be read(multiple frames or only one frame)

6.9 Interrupts Management

6.9.1 Interrupt Block Diagram



6.9.2 Transmit Buffer Interrupt

After the message is transmitted from message buffer JTXB, transmit interrupt(INTF<JTX>) is generated. User must read INTF to clear JTX transmit buffer interrupt.

6.9.3 Receive Buffer Interrupt

When message is successfully received and packed in one receive buffer JRXB(FIFO), INTF<JRX> bit is set ON in the module, and the buffer receive interrupt is cleared by reading null FIFO receive buffer JRXB.

6.9.4 Error Interrupt

Error interrupt(JEERF) is generated from 8 sources: The generation of J1850 error interrupt is controlled by JEERIE.

- Arbitration Error
- IFR Error
- SOF Error
- Bits Error
- CRC Error
- Receive Buffer Overflow

  When the message is successfully received, but FIFO buffer is already full, JERRF<FOV> bit is set ON in the module.
- Bus Error
- Data Frame Superlength

  If the total length of received data frame exceeds 12 bytes, JERRF<FLEN> bit is set ON in the module.

Error interrupt must be removed by clearing JERRF bit in the interrupt service program.

6.9.5 Wake-up Interrupt

In Sleep Mode, the device will detect the bus activity by monitoring J1850 receive pins(VPWRX/PWMRX), VPWRX is still selected by PWMRX depending on JCON1<MOD>. When the bus activity is detected, wake-up(INTF<WAF>) interrupt will be generated. Wake-up interrupt source is determined by reading SLEEP register. INTF<WKF> bit must be removed by reading INTF in the interrupt service program.

Chapter 7 Enhanced EUART Module

7.1 Introduction

Enhanced UART(Universal Asynchronous Receiver Transmitter) module is the serial I/O block provided by ET7190 device. Its implementation is based on UART bus vehicle protocol communication.
Main features of UART module:

- Used as UART module, when there is 4-character-deep transmit and receive FIFO buffer. At that time, full-duplex data transmission is done by UxTX and UxRX pins. Even parity check, odd parity check or parity none check is optional, and one or two stop bit is optional. Hardware automatic BaudRate feature.
- When the start bit is detected, the device programmable wakes up from Sleep Mode.
- Support ISO14230(KWP2000) frame mode, ISO14230-defined all frame formats are automatically identified.
- Support ISO9141-2 frame format.
- Support user-defined frame format, support all kinds of manufacturer protocols(KWP1281 J2610 SCI J1708, etc.), theoretically, support any communication based on UART vehicle.
- Support LIN1.x LIN2.x master mode, slave mode frame format and automatic response.
- Support the simulation of vehicle ECU diagnostic function fast initializationand 5 BaudRate initialization.
- Support K-line logic analysis, used as practical tool to develop the diagnostic tester.
- User can precisely set all various times of all protocol frames
- Each 3-channel TX, RX can input and output pins via UART controlled by the internal electronic switch.
- 2 FIFO receive buffers, the length of each frame is up to 262 bytes.
- Support EUART bus failure protection function

7.2 EUART Control Register

Name	Addr.	Description	Reset Value HEX
UFRMCON	0x2C	Frame Mode Control Register	01
UPINSEL	0x2D	UART Drive Pin Select	00
UFRMV	0x2E	Frame Check Control Register	00
UFLENCON	0x2F	Frame Length Control Register	00
UFLAND	0x30	“And Value” in frame length setting (Bit Masked Clear if operated)	FF
UFTXSTAT	0x31	State Register in Frame Transmission	20
INITLT	0x32	Low Level Time of ISO14230 Fast Initialization	xx
INITHT	0x33	High Level Time of ISO14230 Fast initialization	xx
INITADDR	0x34	5BUAD Slow Initialization Time and Trigger Addr.	xx
UMODE	0x35	EUART Control Register	xx
USTAT0	0x36	EUART State Register 0	10
USTAT1	0x37	EUART State Register 1	01
UTRXREG	0x38	EUART Single Byte Transmit/Receive Register, only operated when UFRMCON<FMOD> is 0	00
UBITADJ	0x39	Time Fine-tuning of Fast initializationand Slow Initialization	00
UERRF	0x3A	EUART Error Register	00
UERRIE	0x3B	EUART Error Interrupt Enable Register	00
UPRICON	0x3C	J1708 Priority Control	00
The following registers are read only			
ABAUD	0x40	Enable BaudRate measurement, generally operated only when UFRMCON<FMOD> is cleared	-
UTXBRK	0x41	Transmit sync intervals, generally operated when UFRMCON<FMOD> is cleared	-
UTXP1T	0x42	Interbyte space in frame transmission(0-255ms)	-
UFREET	0x44	Bus release time. Data is only transmitted from transmit buffer to the bus after EUART is released in the bus.(0-255ms)	-
URXP1T	0x45	Allowed maximum interbyte space in frame reception(0-255ms)	-
UTXVT0	0x46	In frame transmission, maximum waiting time in reception of positive/or reverse check byte(0-255ms)	-
URXVT1	0x48	In frame reception, time delay in transmission of check bytes(0-255ms)	-

7.2.1 UFRMCON Frame Mode Control Register

Note: For any operation on UFRMCON, EUART will be restarted or closed.

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-1
ON	SIMU	LAB	-	-	FMOD<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ON**：EUART Module Enable
1 = Enable EUART
0 = Disable
bit 6 **SIMU**：Enter ECU Simulation State
Only when FMOD=1/3
1 = Enter ECU Simulation State
0 = Normal State
When FMOD=0/2/4/5/6/7, SIMU must be kept cleared rather than ON
bit 5 **LAB**：In LIN Mode: When sync interval is received
This bit is only valid when FMOD=2, when FMOD is any other value, this bit is neglected.
1 = In LIN Mode, when sync interval is received, BaudRate is automatically detected
0 = Detect no BaudRate
bit 2:0 **FMOD<2:0>**：EUART Module Mode of Operation
000 = UART Character Mode – This mode is identical to one common UART serial communication port, FIFO transmit and receive buffer, each with 4 bytes, user must handle all data and times in the application program by himself.
001 = ISO14230 Frame Mode
010 = LIN1.x LIN2.x Frame Mode
011 = User-defined Frame Mode(ISO9141-2, KW1281, J1708, etc.) USERMOD
100 = SCI Half Duplex(J2610, every byte check can be processed)
101 = SCI Full Duplex Mode(J2610)
110 = Reserved, unavailable
111 = UART timing reading, logic waveform analysis

7.2.2 UPINSEL EUART Drive Pin Select Register

Note: For any operation on UPINSEL, please keep off EUART(UFRMCON<ON>=0).

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	-	RXSEL<1:0>		TXSEL<1:0>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 3:2 **RXSEL<1:0>**：EUART RX Input Pin Setting
00 = Select URX1 as EUART module RX pin
01 = Select URX2 as EUART module RX pin

10 = Select URX3 as EUART module RX pin
11 = Reserved

bit 1:0 **TXSEL<1:0>**：EUART TX Output Pin Setting
00 = Select UTX1 as EUART module TX pin
01 = Select UTX2 as EUART module TX pin
10 = Select UTX3 as EUART module TX pin
11 = Reserved

7.2.3 UFRMV Check Control Register

This register value is only valid when UFRMCON<FMOD>=3/4/5, and this value is neglected when UFRMCON<FMOD> is in any other mode.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FRVEN	FTVEN	VFINV	VEND	ENCS	OFFSET<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **FRVEN**：In reception, enable each byte check of frame
0 = disable
1 = enable

bit 6 **FTVEN**：In transmission, enable each byte check of frame
0 = disable
1 = enable

bit 5 **VFINV**：The value in each bye check of frame
0 = Use the original value in check
1 = Use the reverse value in check

bit 4 **VEND**：Whether each byte check of in-frame data includes the last byte of frame or not, when each byte check is enabled, if there is no length byte in frame, the last byte must be checked, VEND must be set ON, because receiver doesn't know which one is the last byte of frame.
0 = Exclude
1 = The last byte also requires check.

bit 3 **ENCS**：ChKSUM in Reception
0 = No CHKSUM is done in reception
1 = There are 2 methods of calculation for CHKSUM, as determined by UFLENCON<CSM>.

bit 2:0 **OFFSET<2:0>: Length Offset** 0-7
The length byte of one frame in actual application doesn't always represent the actual total length of frame, which generally means the length of in-frame valid data, and the total length of frame generally requires an additional offset value. When EUART is in reception, the total length of data frame is obtained depending on the length byte of frame and OFFSET.

7.2.4 UFLENCON Frame Length Control Register

This register value is only valid when UFRMCON<FMOD>=3/4/, and this value can be neglected when UFRMCON<FMOD> is in any other mode.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EN	LRR<2:0>			CSM	LENP<2:0>		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **EN**：Enable Length Control in Reception
0 = disable
1 = enable

bit 6:4 **LRR<2:0>**：Length Byte Right Shift 0-7 Bits
Byte with data length information is contained in frame, the length might be some bits between bytes, and the actual length of frame is calculated by shifting rightward to the utmost.

bit 3 **CSM:** Frame and Check Calculation Method in Reception
0 = ISO14230 CHKSUM
1 = SAE J1708 CHKSUM

bit 2:0 **LENP<2:0>**：Frame Length Byte Bit (0-7)
0 = Data Length Byte is Byte 1 in Data Frame
1 = Data Length Byte is Byte 2 in Data Frame
2 = Data Length Byte is Byte 3 in Data Frame
3 = Data Length Byte is Byte 4 in Data Frame
4 = Data Length Byte is Byte 5 in Data Frame
5 = Data Length Byte is Byte 6 in Data Frame
6 = Data Length Byte is Byte 7 in Data Frame
7 = Data Length Byte is Byte 8 in Data Frame

7.2.5 UFLAND Control Register

This register value is only valid when UFRMCON<FMOD>=3/4/5, and this value is neglected when UFRMCON<FMOD> is in any other mode.

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
UFLAND<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Enable frame length control in reception: When the length of data frame is calculated, "AND" calculation is made between the right-shifted length byte(LRR) value and UFLAND, and "0" bit in UFLAND is masked.

7.2.6 UFTXSTAT Frame Transmit State Register

R-0	R-0	R-1	U-0	R/W-0	R/W-0	U-0	U-0
TXBF	TXBUSY	BUSFREE	-	CRXBF	CTXBF	-	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **TXBF**：Transmit Buffer UTXB State, Read Only
0 = Transmit Buffer UTXB Empty.
1 = Transmit Buffer UTXB Full.

bit 6 **TXBUSY**：Transmitter State Bit, Read Only
0 = Transmitter Idle.
1 = Transmitter Busy.

bit 5 **BUSFREE**：Bus Release State Bit, Read Only. Bus Release is Controlled by UFREET Time
0 = Bus unreleased.
1 = Bus released, data transmission is available.

bit 3 **CRXBF**：Enforce Clear URXB FIFO Receive Buffer
0 = Unexecuted
1 = Set ON, after URXB receive buffer is cleared, this bit is automatically set Clear

bit 2 **CTXBF**：Enforce Clear UTXB Transmit Buffer
0 = Unexecuted
1 = Set ON, after URXB receive buffer is cleared, this bit is automatically set Clear and TXBF is set Clear

7.2.7 Low Level Time of INITLT Fast initialization

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
INITLT<7:0> 复位值=25 Reset Value = 25							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

INITLT<7:0>: Set as 0-255 ms

7.2.8 High Level Time of INITHT Quick Initialization

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
INITHT<7:0> Reset Value =25							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

INITHT<7:0> : Set as 0-255 ms

7.2.9 INITADDR 5 Addr. in BaudRate Initialization

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
INITADDR<7:0> Reset Value =0x33							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

7.2.10 UMODE Mode Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	-	-	URXINV	BRGH	PDSEL<1:0>		STSEL
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **WAKE**： In Sleep Mode, Start Bit Wake-up Enable Bit is Detected
1 = Enable Wake-up
0 = Disable Wake-up
bit 4 **URXINV**： Receive Polarity Reverse Bits
1 = UxRX idle state is set Clear
0 = UxRX idle state is set ON
bit 3 **BRGH**： High BaudRate Select Bits
1 = High Speed
0 = Low Speed
bit 2-1 **PDSEL<1:0>**： Odd/Even Parity Check and Data Select Bits
11 = 9-bit data, no parity check
10 = 8-bit data, odd parity check
01 = 8-bit data, even parity check
00 = 8-bit data, no parity check
bit 0 **STSEL**： Stop Select Bits
1 = 2 Stop Bits
0 = 1 Stop Bit

7.2.11 USTAT0 State Register 0

U-0	U-0	U-0	R-1	R-0	R-0	R/C-0	R-0
-		-	RIDLE	PERR	FERR	OERR	URXDA
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 4 **RIDLE**：Receiver Idle Bits(Read Only)
1 = Receiver Idle
0 = Receiving data
bit 3 **PERR**：Odd/Even Parity Check Error State Bits(Read Only)
1 = Odd/even check error of current character detected
0 = Odd/even check error undetected
bit 2 **FERR**：Frame Error State Bits(Read Only)
1 = Frame error of current character detected
0 = Frame error undetected
bit 1 **OERR**：Receive Buffer Overflow Error State Bits(Clear/Read Only)
1 = Receive buffer overflowed
0 = Receive buffer not overflowed(When the OERR bit in ON state is cleared, receive buffer and RSR reset will be set in empty state)

bit 0 **URXDA**：Whether the Receive Buffer includes Data Flag Bit or Not(Read Only)
1 = Receive buffer contains data, at least one or more characters is readable
0 = Receive buffer empty

7.2.12 USTAT1 State Register 1

U-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R-0	R-1
-	UTXINV	-	-	UTXBRK	UTXEN	UTXBF	TRMT
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 6 **UTXINV**：Transmit Polarity Reverse Bits
1 = UxTX idle state is set Clear
0 = UxTX idle state is set ON
bit 3 **UTXBRK**：Transmit Gap Bits
1 = Whichever the state of transmitter is, UxTX pin is driven in low level(sync interval transmit – start bit is tailed with 12 "0", then followed with 1 start bit)
0 = Disable or complete the transmission of sync interval characters

bit 2 **UTXEN**：Transmit Enable Bits

1 = Enable UARTx transmitter, UxTX pin is controlled via UARTx(if UARTEN = 1)

0 = Disable UARTx transmitter, all pending transmission are aborted, and the buffer is reset. UxTX pin is controlled via port.

bit 1 **UTXBF**：Transmit Buffer Full State Bits(Read Only)

1 = Transmit buffer already full

0 = Transmit buffer not full, at least one or more characters are writable

bit 0 **TRMT**：Transmit Offset Register Empty Bits(Read Only)

1 = Transmit offset register is empty and transmit buffer is empty(the last transmission is completed)

0 = Transmit offset register is empty, transmission is in process or queuing in transmit buffer

7.2.13 UTRXREG Receive/Transmit Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
URX/UTX Register							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

When UFRMCON<FMOD> equals to 0, the register used in transmission or reception of single byte is actually read and written by two separate 4-byte-deep FIFO registers URX and UTX in the hardware. When FMOD is not 0, this register is not readable or writable, or it may lead to the confused state of block frame. Data is directly transmitted to the bus by writing UTRXRG, and data is read from FIFO by reading UTRXREG.

7.2.14 UBITADJ Initial Time Fine-tune Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OP	ADJVAL<6:0>						
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- In 5 BaudRate or fast initializationfor K bus line, the bit time of initialization waveform is fine-tuned, each frame is 6.4us. When OP=0, time increases; when OP=1, time decreases. ADJVAL is valued between 0 and 127, the maximum regulation time is ±812.8us.
- 5 BaudRate initialization is the regulation of 200ms bit time.
- Fast initializationis the regulation of INITLT and INITHT time.
- The function of this register may sometimes fine-tune the nonstandard application. Generally it is not required to regulate the reset value as Clear.

7.2.15 UERRF Error Flag Register

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
FT	PERR	NORP	LENE	VERR	BITERR	UNDDA/OVF	BUSERR
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **FT**：External Hardware Protection Action
1 = FT failure protection generated
0 = Error not generated
bit 6 **PERR**：Identifier Odd/Even Parity Check Error, (ID check error in LIN mode and when odd or even parity check is applied)
1 = Odd/Even parity check error.
0 = Error not generated
bit 5 **NORP**：LIN Slave Not Responded
1 = Slave not responded
0 = Error not generated
bit 4 **LENE**：Frame Length Error(when the enable length is controlled, UFLENCON<EN>=1)
1 = Received frame length is insufficient
0 = Error not generated
bit 3 **VERR**：Each Byte Check Error in Transmission of Data Frame
1 = Error generated
0 = Error not generated
bit 2 **BITERR**：Stop Bit Error or Bus Competition Generated
1 = Error generated
0 = Error not generated
bit 1 **UNDDA/OVF**：Unknown Data(Frame Mode) Received /Receive Buffer Overflow
bit 1 **UNDDA/OVF**：
1 = Undefined nonstandard data received
0 = Error not generated
bit 0 **BUSERR**：Bus Error(Generally caused by hardware failure)
1 = Error generated in transmission
0 = Error not generated

7.2.16 UERRIE Error Interrupt Enable Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FTIE	PERRIE	NORPIE	LENEIE	TVERRIE	BITERRIE	UNDDAIE/OVFIE	BUSERRIE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **FTIE**：External Hardware Protection Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt
bit 6 **PERRIE**：Identifier Odd/Even Check Error Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt
bit 5 **NORPIE**：LIN Slave Not Responded Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt
bit 4 **LENEIE**：Frame Length Error Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt
bit 3 **TVERRIE**：Each Byte Check Error Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt
bit 2 **BITERRIE**：Error Interrupt Enable Bits due to Stop Bit Error or Bus Competition
1 = Enable error interrupt
0 = Disable interrupt
bit 1 **UNDDAIE/OVFIE**：Unknown Data Error Interrupt Enable Bits Received/Buffer Overflow
1 = Enable error interrupt
0 = Disable interrupt
bit 0 **BUSERRIE**：Bus Error Interrupt Enable Bits
1 = Enable error interrupt
0 = Disable interrupt

7.2.17 UPRICON J1708 Priority Control Register

R/W-0	U -0	U -0	U -0	U -0	R/W-0	R/W-0	R/W-0
J1708E	-	-	-	-	PRI		
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **J1708E**：Enable J1708 Priority
1 = Enable
0 = Disable
bit 2:0 **PRI<2:0>**：Transmitter Priority
000 = Transmitter Priority 1, Highest
001 = Transmitter Priority 2
010 = Transmitter Priority 3
011 = Transmitter Priority 4
100 = Transmitter Priority 5

101 = Transmitter Priority 6
110 = Transmitter Priority 7
111 = Transmitter Priority 8, Lowest

7.2.18 ABAUD Automatic BaudRate Enable Bits(Write Only)

This register is write only with no read command, it is actually a command of starting BaudRate measurement, which is generally used when UFRMCON<FMOD> is set Clear and unavailable in any other mode.

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
-	-	-	-	-	-	-	ABAUD
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 0 **ABAUD**： Automatic BaudRate Enable Bits
1 = Enable the measurement of BaudRate of the next character. Receive sync field(55h) is required; hardware clears when completed.
0 = Disable the measurement of BaudRate or measurement already completed

7.2.19 UTXBRK Transmission Sync Interval(Write Only)

This register writes only with no read command, actually a command of transmission sync interval, which is generally used when UFRMCON<FMOD> is cleared and unavailable in any other mode. Command is identical to setting USTAT1<UTXBRK> ON.

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
-	-	-	-	-	-	-	UTXBRK
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 0 **UTXBRK**： Sync Interval Character Transmission
1 = Whichever the state of transmitter is, UxTX pin is driven to low level(sync interval transmission – Start bit is tailed with 12 "0", followed by 1 stop bit)
0 = Disable or complete the transmission of sync interval characters

- Interval character transmission include 1 start bit, followed by 12 “0” bits and 1 stop bit. When the transmit offset register is loaded with data, interval character is transmitted by just setting UTXBRK and UTXEN bit ON. UTRXREG register must be falsely written before starting the transmission of interval character. Please note that the value of data written in UTRXREG register will be neglected in transmission of interval character. Proper sequence is started by sending all “0” in writing operation. After stop bit is transmitted, hardware will automatically reset UTXBRK bit. By doing so, user application program can preload the next byte to be transmitted into transmit FIFO following the interval character(generally sync character in Standard LIN).
- Before UTXBRK is set ON, user application program must firstly wait for the transmitter is changed to be idle(TRMT = 1). UTXBRK will cover all other activities of transmitter. If user application program clears TXBRK bit prior to the completion of sequence, it may lead to the accidental module behavior. Transmission interval character will not generate transmit interrupt.
- This function enables user to transmit LIN frame header when FMOD=0. This will not limit the user to handle LIN operation only when FMOD=2.

7.3 EUART Control Register Set(Buffer)

Name(Buffer)	Addr.	Length (Byte)	Description	Reset Value
UBRG	0x2A	2	Bus BaudRate scaler(16-bit integer, 2 bytes)	xx
U5BAUDCON	0x2B	10	5 BaudRate bus initialization control	xx
UPULSCON	0x2C	16	User-defined initialization pulse control	xx
UPULSBIT	0x2D	16	16 user-defined pulse bits setting	xx
SIMUCON	0x2E	12	Specifically used for ECU simulation control	xx
LIDCON	0x2F	64	LIN1.x LIN2.x ID Control Table, write use CMD_WR_LB	xx
LACKB0	0x30	8	LIN slave automatic response, data transmission buffer 0	xx
LACKB1	0x31	8	LIN slave automatic response, data transmission buffer 1	xx
LACKB2	0x32	8	LIN slave automatic response, data transmission buffer 2	xx
LACKB3	0x33	8	LIN slave automatic response, data transmission buffer 3	xx
UTXB	0x3F	262	UART message transmit buffer, write use CMD_WR_LB	xx
URXB		262*2	FIFO message receive buffer, FIFO depth is 2 frames	xx

7.3.1 UBRG：UART BaudRate Scaler

R/W-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BRG<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BRG<15:8>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: When applied, it is more convenient if this buffer can be seen as a 16-bit integer to handle <Bits 15: 0>

BRG<15:0>: BaudRate scaler bits

7.3.2 U5BAUDCON：5 BaudRate Initialization Control Register Set

Control register set consists of the following 10 bytes

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BYTE 0	ABAUD	INVAD	KW2EN	KW2T	ENL	RLEN<2:0>		
1	W1MAX<7:0> *20ms							
2	Not Used							
3	W2MAX<7:0> * 1ms							
4	Not Used							
5	Not Used							
6	Not Used							
7	W4MAX<7:0> * 1ms							
8	W4SEND<7:0> * 1ms							
9	USERKW<7:0>							

RLEN<2:0>： After the transmission of 5 BaudRate Trigger Addr. is completed, the required number of bytes to be continuously received. (OBD2 = 3 bytes)

ENL： Whether the transmission of BaudRate Trigger Addr. is simultaneously sent to L line

1 = 5 BaudRate trigger simultaneously outputs to L line

0 = Not output to L line

KW2T： Reversed KW2 byte transmission method(ISO9141-2, in initialization of ISO14230 definition, when tester receives KEYWORD2, tester is required to send a reversed KW2 to ECU)

1 = Return the reversed KW2 to the bus(OBD2-defined)

0 = Transmit to the bus in USERKW byte value(user-defined value)

KW2EN： Whether the reversed KW2 byte is transmitted or not

1 = Transmit (OBD2-defined)

0 = Not Transmit(when the RLEN-defined last byte is received, the module regards the initialization as completed)

INVAD： Whether the module receives the reversed initialization address byte

1 = When the module completely transmits the reversed KW2, receive the reversed and initialization address returned from ECU

0 = Not receive, initialization process is directly completed

ABAUD：Automatic BaudRate

1 = In the process of initialization, BaudRate is automatically detected,

0 = In the process of initialization, BaudRate is not detected

W1MAX：After the module transmits 5 BaudRate Trigger Addr., waiting time of ECU returning 0x55 sync character. If the time exceeds W1MAX*20ms but ECU doesn't return 0x55 byte, the module returns INIT_NONE_SYNC error information. The initialization process is aborted.

W2MAX：After ECU returned 0x55 sync byte is received. If the time exceeds W2MAX*1ms, if ECU doesn't return KeyWord1 or KeyWord2 byte, the module returns INIT_NONE_KW error information. The initialization process is aborted.

W4MAX：After the module completes the transmission of reversed KeyWord2 byte. If the time exceeds W4MAX*1ms, ECU doesn't return the reversed initialization address byte, the module returns INIT_NONE_INVADDR error information. The initialization process is aborted.

W4SEND：Time delay of the module transmitting the reversed KeyWord2 byte. Time value: W4SEND*1ms.

USERKW：When KW2T=0, user-defined transmit value, in replacement of KW2 reversed byte.

7.3.3 UPULSCON：User-defined Pulse Initialization Control Register Set

Control register set consists of 16 bytes, and the level of logic bits is controlled by UPULSBIT.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BYTE 0	T64US	ENL	-	-	BITCNT<3:0>			
1	TIMER1—Bit 1 Time							
2	TIMER2—Bit 2 Time							
3	TIMER3—Bit 3 Time							
4	TIMER4—Bit 4 Time							
5	TIMER5—Bit 5 Time							
6	TIMER6—Bit 6 Time							
7	TIMER7—Bit 7 Time							
8	TIMER8—Bit 8 Time							
9	TIMER9—Bit 9 Time							
10	TIMER10—Bit 10 Time							
11	TIMER11—Bit 11 Time							
12	TIMER12—Bit 12 Time							
13	TIMER13—Bit 13 Time							
14	TIMER14—Bit 14 Time							
15	TIMER15—Bit 15 Time							

BITCNT<3:0>：The number of user-defined pulse bits is no more than 15 bits, no bit output is available when set Clear.

ENL：When the user-defined pulse bits are transmitted, whether simultaneously transmitted to L line

1 = User-defined pulse bit, simultaneously output to L line

0 = Not output to L line

T64US：Bit Time TIMERn (n=0-15) Precision

1 = Bit time as TIMERn * 6.4us

0 = Bit time as TIMERn * 1ms

7.3.4 UPULSBIT：Logic Level of User-defined Pulse Initialization Bit

Control register set consists of the following 16 bytes, and when operated, the actual length is UPULSCON<BITCNT> set value.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BYTE 0	-	-	-	-	-	-	-	B0
1	-	-	-	-	-	-	-	B1
2	-	-	-	-	-	-	-	B2
3	-	-	-	-	-	-	-	B3
4	-	-	-	-	-	-	-	B4
5	-	-	-	-	-	-	-	B5
6	-	-	-	-	-	-	-	B6
7	-	-	-	-	-	-	-	B7
8	-	-	-	-	-	-	-	B8
9	-	-	-	-	-	-	-	B9
10	-	-	-	-	-	-	-	B10
11	-	-	-	-	-	-	-	B11
12	-	-	-	-	-	-	-	B12
13	-	-	-	-	-	-	-	B13
14	-	-	-	-	-	-	-	B14
15	-	-	-	-	-	-	-	B15

Bn (n=1-14): n Bit Logic Level in User-defined Initialization
1 = Logic high level
0 = Logic low level

B0: This bit is neglected, Bit 0 logic level in user-defined initialization is always cleared

B15: This bit is neglected. 15 bits are allowed to the maximum in user-defined initialization. The last one is an invalid bit.

7.3.5 SIMUCON：Simulation ECU Control

- This register set consists of 12 bytes, which is used to set the state when ET7190 enters ECU simulation.
- EUART can simulate the state of ISO9141-2/ISO14230 slow initialization, ISO14230 fast initializationor KW1281 initialization. After EUART module enters the simulation state, it waits for the correct triggering of requester, after the correct trigger signal is received, ET7190 will store the trigger information into EUART FIFO receive buffer URXB. And enter into the normal mode. User can read the received trigger information for the subsequent simulation ECU handling process.
- If the trigger information from requester errors, ECU always keeps the simulation state. Unless user cancels it.
- Its function is to make easy for user to compile program to simulate ECu diagnosis of K line operation and ECU output any data, and for the diagnosis and development.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BYTE 0	SMMOD Valid value as 0, 1, 2, any other value is invalid							
1	SMADDR The simulated 5 BaudRate ECU Trigger Addr.							
2	SMTTOL In quick initialization, the allowed error of low level time, up to ±255*6.4us							
3	SMLTIME In quick initialization, low level time required for simulation, 1ms/unit							
4	SMKW1 Keyword1							
5	SMKW2 Keyword2							
6	SMW1SEND, 1ms/unit							
7	SMW2SEND, 1ms/unit							
8	SMW3SEND, 1ms/unit							
9	SMW4SEND, 1ms/unit							
10	SMW4RMAX, 1ms/unit							
11	This Byte is not used							

SMMOD：Simulation ECU Mode of Operation

0 = Simulate ECU to wait for ISO9141-2 or ISO14230 slow initialization
1 = Simulate ECU to wait for ISO14230 quick initialization
2 = Simulate ECU to wait for KW1281 slow initialization
x = Any other value, unavailable

SMADDR：Slow 5 BaudRate Initialization Address 0x00-0xFF

When ECU enters in the simulation state of 5 BaudRate slow initialization(SMMOD=0/2), EUART module waits for the requester's 5 BaudRate trigger signal, trigger address must equal to SMADDR set value.
This value is neglected when SMMOD=1

SMTTOL：Low Level Time Allowed Error in Fast initialization

- When ECU enters in the simulation state of quick initialization(SMMOD=1), EUART module waits for requester's low level trigger signal, the time of requester's low level signal must be within the range of SMLTIME±SMTTOL before the module confirms this trigger signal as one of qualifying signals.
- This value is neglected when SMMOD=0/2
- SMTTOL time is SMTTOL* 6.4us, generally set as 78, 6.4us*78=499us. Compared to ±0.5ms.

SMLTIME：Low Level Standard Time of Quick Initialization(ms)

OBD2 is generally set as 25ms, SMLTIME=25. In simulation, high level time is neglected.
The actual effect of ISO-defined high level time is to give a time delay, allowing a process of preparation when ECU receives the low level signal, ET7190 preparation time is in us, so high level time is neglected.

SMKW1：The module transmitted Keyword1 in slow initialization, and this value is neglected in quick initialization.

SMKW2：The module transmitted Keyword2 in slow initialization, and this value is neglected in quick initialization.

SMW1SEND：In slow initialization. When the module receives the valid trigger address, after the time delay(0-255ms), 0x55 sync byte is transmitted. (Refer to ISO9141-2 W1)

SMW2SEND：In slow initialization. When the module receives the valid trigger address, after the time delay(0-255ms), SMKW1 Keyword1 is transmitted. (Refer to ISO9141-2 W2)

SMW3SEND：In slow initialization. After the module transmits SMKW1, after the time delay(0-255ms), SMKW2 Keyword2 is transmitted. (Refer to ISO9141-2 W3)
SMW4SEND：In slow initialization. After the module receives the reversed KW2, after the time delay(0-255ms), the module transmits the reversed SMADDR value to K bus to complete the trigger process.(Refer to ISO9141-2 W4 Sender)
SMW4RMAX：In slow initialization. After the module transmits SMKW2, it is required to receive the reversed KW2 Keyword, the module must receive the reversed value within SMW4RMAX set time, if exceed this time, no signal trigger abort will be received. (Refer to ISO9141-2 W4 Receiver)

Note: For more details, please refer to 8.9 under Chapter 8, Simulate ECU Related

7.3.6 LIDCON：LIN ID Control Table

- When EUART operates in(FMOD=2) LIN mode, EUART module handles the timing of frame transmission and frame reception, user only needs to define the relevant ID function and handle process and task in application program. It is key to understanding ID Control Table.
- LIDCON is 64 bytes long. Each ID is defined by ID function of the corresponding register.
- When EUART operates, master data frame and slave data frame can be simultaneously processed, when it is used, bus conflict must be avoided to ensure only a master operated in the bus. (Master

LIDCONi (i=0-63) Control Byte Form

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
IDEN	TRM	LENM		CSM	ARE	ARP	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **IDEN**：ID Enable
1 = Enable, the module processes ID=i data frame depending on the current LIDCONi defined form
0 = Disable, the module will not receive or respond ID=i data frame
bit 6 **TRM**：ID Mode(Translater or Receiver Mode)
When in master mode,
1 = Current ID receives the respond data required from slave, and this frame is slave task
0 = Current ID frame is master task, and all in-frame data are transmitted from master

When in slave mode,
1 = This frame is slave task, and the module responds to master

0 = Slave module only receives this task(if IDEN=1, receive is done whether for master task or slave task)

bit 5.4 **LENC<1:0>**：Current Frame Data Length Code

11 = Current ID frame data length is 8 bytes
10 = Current ID frame data length is 4 bytes
01 = Current ID frame data length is 2 bytes
00 = Current ID frame data length is random(generally used in extended identifier ID=0x3E/0x3F)

bit 3 **CSM**：Check Sum Mode

1 = LIN2.x check mode, check sum contains ID byte
0 = LIN1.x check mode, check sum doesn't contain ID byte

bit 2 **ARE**：Auto Respond Enable (only in Slave Mode) When the current ID frame header is received, if TRM=1, the module will directly use LACKBn buffer(n=ARP) as slave respond data for the transmission.

1 = Enable, responded from LACKBn
0 = Disable, the module stores the current ID into receive buffer, user transmits the respond slave task.

bit 1:0 **ARP<1:0>:** Current ID Auto Respond Position

11 = Use LACKB3 buffer as slave task respond data
10 = Use LACKB2 buffer as slave task respond data
01 = Use LACKB1 buffer as slave task respond data
00 = Use LACKB0 buffer as slave task respond data

Note: LIDCON Control Table is generally invariable after the system setting according to ID task property. No conflict shall arise between setting master and multiple slaves, and arrangement is rationally made according to the application.

7.3.7 LACKBn：LIN Auto Respond Data Buffer n=0-3 (LIN Acknowlege Buffer)

It is used for the data buffer of slave task auto respond, up to 8 bytes, the valid length is determined by LIDCON LENC(must not be 0), and the frame LIDCON ARP determines which buffer is used.
4 buffers(LACKB0, LACKB1, LACKB2, LACKB3) are available for user setting, when users applies these 4 buffers to process the slave tasks requiring response, data is written in these 4 buffers at any time respectively. When the module receives the slave task frame header, and ID is qualified, the module directly uses the buffer data as response.

Note: For more details, please refer to 8.6 under Chapter 8, LIN1.x /LIN2.x Application

7.4 EUART Time Control Register

When EUART is operated in frame mode(FMOD=1/2/3/4/5), in which 5 time registers are used for the reception and transmission of bus data frame, the setting of time register is very important, because it directly determines whether the complete data frame is correctly transmitted and received.

7.4.1 UTXP1T：Transmit Interbyte Space Time(Write Only)

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0-255 ms							

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- When FMOD=0/5/6/7: Not used, (Note FMOD=5, this time is not used in SCI full duplex transmission, when ET7190 is operated in this mode, interbyte space is always 0ms.)
- When FMOD=2: LIN mode, this time is In-Frame Response space when the module transmits master task or automatically responds to slave task, interbyte space when ET7190 transmits data is always 0ms.
- When FMOD=1/3/4, if there is no each byte check in data frame, in transmission of multiple frames, this time is the interbyte space time in continuous data transmission, which can be set as 0-255ms. When the data frame contains each byte check, this time is the next byte in module delay of UTXP1T after the check byte is received by sender. For the development of OBD2 tester, the time is generally set as 5ms.

7.4.2 UFREET：Bus Release Time(Write Only)

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0-255 ms							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- When FMOD=0/5/6/7: Not used
- When FMOD=2: In LIN Mode, UFREET must be set Clear, message frame is processed in LIN Mode according to the data length, when the message ends, master can directly start a new task with no need of time delay.
- When FMOD=1/3/4, the active module(e.g.: diagnostic tester) UFREET is generally set as the minimum P3 time(generally as 55ms). When ET7190 is used as passive module, it is generally set as the minimum P2 time(25ms).

  After user transmits UTXB and writes one frame data, when the previous data transmission or reception is completed, ET7190 needs to wait for the bus release time defined under protocol before the transmission of new frame is started. If UFREET is set Clear, after the last frame data is completed, the module will directly starts the transmission of new data frame(if UFTXSTAT<TXFB> is set ON). UFREET time must be precisely set as per ISO14230/ISO9141-2/KW1281/SAE J1708 or other protocols. The universality of device can be enhanced.
- When FMOD=1/3/4, there is a special case, if when UPRICON<J1708E> is set ON and J1708 is enabled, UFREET must be set ON. ET7190 starts the bus transmission as per UPRICON<PRO> Priority Time. For any protocol other than J1708, UPRICON<J1708E> must be set Clear.

7.4.3 URXP1T：Allowed Maximum Interbyte Space Time in Reception(Write Only)

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0-255 ms							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

In reception, if one byte is received, ET7190 module must receive the next byte within URXP1T time, if the time exceeds URXP1T time, the module receives no data, ET7190 considers the time as overflowed and frame as completed.

- When FMOD=0/6: Not Used
- When FMOD=1：ISO14230:

1) Data frame contains length byte. When EUART module is in bus-idle, wait for reception, EUART starts to receive frame after one byte is received, and the completion of data reception is determined according to in-frame length byte, when all data are completed and the reception of CS byte is completed, frame ends, data frame is loaded into FIFO URXB receive buffer, the bus is back to idle state, wait for the reception of next frame data.
2) In this process, interbyte space time transmitted by sender must be less than URXP1T time, in order to prevent the receiver EUART receive time from overflowing, in case of time overflow, this module regards this frame transmission as completed, data frame is loaded in FIFO URXB receive buffer, bus is back to idle state to wait for the next frame data. In this case, the received frame is generally an error frame.
3) In ISO14230 mode, URXP1T is generally set as 20ms-24ms

- When FMOD=2: LIN Mode
  1) All frames must comply with length standard(LIDCON-defined), no URXP1T time overflow shall arise from the normal process of transmission and reception. URXP1T time setting must be longer than the maximum respond space time(UTXP1T).
  2) If there is no slave respond in slave task, UPXP1T time will overflow. UERRF<NORP> error flag is set ON.
  3) If frame is not complted in reception of frame(partial data is received). In case of URXP1T timeout, the module reception aborts, incomplete data frame is loaded in FIFO URXB receive buffer. Meanwhile, UERRF<LENE> length error flag is set ON.

- When FMOD=3/4/5:
  1) In idle state, the bus is in a state of reception after the reception of data, if no more data is received over URXP1T time, EUART deems it as the completion of frame, the received data is

transmitted from the receiver to FIFO receive buffer URXB, whether enable length is controlled or not, no error flag will be generated.

2) No length byte contains in ISO9141-2 CARB defined data frame, frame end is identified by time overflow. Generally, URXP1T=20ms-24ms can be defined in CARB. UFREET=55ms UTXP1T=5ms

- FMOD=7 Logic State Capture

  If no level logic change is generated in the bus over URXP1T time, the gathered data is transported from receiver to FIFO receive buffer URXB.

7.4.4 UTXVT0：In Frame Transmission, Maximum Waiting Time for Check Byte(Write Only)

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0-255 ms							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- When FMOD=01/2/5/6/7: Not Used
- When FMOD=3/4:

  When each byte is checked in enable transmit data frame, after one byte is sent from the sender, the transmitter will wait for receiver to return the check byte, if no check byte is received within UTXVTO time, the transport aborts. UERRF<VERR> check error flag is set ON.

7.4.5 URXVT1: In Frame Reception, Time Delay for the Transmission of Check Byte(Write Only)

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0-255 ms							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as “0”
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

- When FMOD=01//2/5/6/7: Unavailable
- When FMOD=3/4

  When each byte is checked in enable receive data frame, after one byte is received by the receiver, check byte will be transported to the sender, and URXVT1 time is the time delay of transport check byte.

7.5 EUART Transmit Buffer UTXB Structure

EUART transmit buffer is 262 bytes long, data transmission of EUART bus is started by writing in UTXB transmit buffer. When it is written in transmit buffer, it must be processed in UFTXSTAT<TXBF> with flag bit as Cleared, if the transmit buffer is full, no writng is valid. Data structure of transmit buffer is presented below: Transmit method of Byte1 Byte2 control frame, Byte2-262 is actually the data to be transmitted to the bus, the length is determined by L<8:0>.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Byte 1	L7	L6	L5	L4	L3	L2	L1	L0
Byte 2	SACK	INIT<2:0>			-	-	-	L8
Byte 3-262	Data0-Data259 Data with up to 260 bytes							

L<8:0>： The valid length of transmit data frame, valid value is in 0-260. L<8:0> must not be greater than 260.

If L=0, and INIT=1, 2 or 3, bus initialization is only done. No transmission of data frame.

If L is not cleared, INIT=0, then data frame with the length of L is transmitted to the bus.

If L is not cleared, and INIT=1, 2 or 3, after the bus initialization, data frame with the length of L is transmitted to the bus.

When INIT=2, after the completion of initialization, time delay of transmitting message is UFREET

When INIT=1/3, after the completion of pulse, transmission is immediately started.

INIT<2:0>： The bus initialization is done before the transmission of data frame, only operated when FMOD=1 or 3, which is neglected in any other mode.

000 = Only normally transmit data, with no initialization before the transmission

001 = Fast initialization is done before the transmission of data frame

010 = 5 BaudRate initialization is done before the transmission of data frame, as per the control register set U5BAUDCON.

011 = User-defined initialization is done before the transmission of data frame, as per the control register set UPULSCON.

100-111：Unavailable

SACK： Only operated in LIN Mode when MOD=2, which is neglected in any other mode.

1 = User's respond data for LIN slave task, SACK is set ON only when the request for this module is received in master, as followed by the transmission of slave task data.

0 = Start the normal master task transmission. For ET7190 EUART module, when operated in FMOD=2, it is LIN mode, ET7190 hardware doesn't differentiate master from slave. When user starts the transmission of master task, LIN module deems this module as mater. So in the self-constructed LIN network, user must only ensure that only one module transmits master task. Other modules receive or respond to slave. More details of use will be discussed in LIN Chapter.

Data[260]： Data up to 260 bytes requied to transmit. 260 bytes is suitable to transmit the longest data frame under ISO14230 defined 4-byte protocol header.Header(3)+Len(1)+Data(255)+CS(1)=260 bytes.

7.6 EUART Receive Buffer URXB Structure

7.6.1 URXB Data Structure

Receive buffer structure is basically identical to that of transmit buffer, but the defiend form is slightly different. EUART receive buffer is 262 bytes long, EUART has 2 FIFO receive buffers in 262 bytes long(there is one temporary message buffer in the receiver, user cannot directly access to it, actually equivalent to 3 frames buffer). By reading URXB and FIFO receive buffer data, when data is empty, INTF<URX> bit is automatically cleared.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Byte 1	L7	L6	L5	L4	L3	L2	L1	L0
Byte 2	SACK	TYPE<2:0>			CSERR	-	-	L8
Byte 3-262	Data0-Data259 Data with up to 260 bytes							

L<8:0>： The Valid Length of Received Data Frame.

CSERR： Data Frame Check and Flag. (If there is no CS check in protocol, CSERR is always cleared)

1 = Current frame check and Checksum error

0 = Current frame check and Checksum correct

TYPE<2:0>： Received Data Frame Type.

000 =Received data is normal data frame

001=Not Used

010=Received data is the information of 5 BaudRate initialization

011=Not Used

100=Not Used

101 =Not Used

110 = Not Used

111 = The received data is the trigger signal in simulation of ECU, when L<8:0> is always cleared with no Data actual data.

SACK： Only operated in LIN Mode when MOD=2, which is always cleared in any other mode.

1 = LIN required user respond slave task. When L<8:0> length is 1, the received data is the user responded ID identifier(0-63 excludes odd/even parity check bits). User must immediately transmit slave respond data.

0 = The received data is normal data frame, which contains ID+Data Byte+CS.

Data[260]： Data up to 260 bytes requied to transmit. 260 bytes is suitable to receive the longest data frame under ISO14230 defined 4-byte protocol header. Header(3)+Len(1)+Data(255)+CS(1)=260 bytes.

7.6.2 Data Structure of Slow Initialization Information

When user does 5 BaudRate initialization, whether the initialization succeeds or not, EUART will load one-frame initialization information to FIFO receive buffer URXB, when the data frame **TYPE<2:0> =** 010(data type). User can process according to the received information in initialization. The content of receive buffer is listed below:

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Byte 1	L7	L6	L5	L4	L3	L2	L1	L0
Byte 2	0	TYPE<2:0>=010			0	-	-	L8
Byte 3	INIT_ERRFLAG, error flag in initialization							
Byte 4	INV_ADD reversed trigger address							
Byte 5,Byte6	BRG(BaudRate scaler value 2 bytes)							

Byte 7	SYNC(0x55, if BaudRate is automatically detected in initialization, SYNC=0)
Byte 8	KeyWord1 Protocol Keyword 1
Byte 9	KeyWord2 Protocol Keyword 2

L<8:0>：The valid length of the received data frame. Its length may be 4-7 bytes

Byte 3：**INIT_ERRFLAG** Initialization Result Flag

0x00 =Initialization normally completed(INIT_NORMAL), when Byte4-9 is the valid byte
0x01 =Error, no sync byte 0x55 is received(INIT_NONE_SYNC)
0x02 = Error, no Keyword1 or 2 is received(INIT_NONE_KW)
0x03 =Error, error arises when the reversed KW2 is transmitted(INIT_ERR_INVKW2)
0x04 = Error, no reversed trigger address returned from ECU is received(INIT_NONE_INVADDR)
0x05=Error, the reversed trigger address error returned from ECU is received(INIT_ERR_INVADDR)
0x06 = Error, the unknown data is received(INIT_RECE_ILLGAL)
0x0F = Module or bus error(INIT_ERROR_BUS)
Other = Undefined

Byte 4：**INV_ADD** If there is no error of initialization, ECU responded trigger address reversed value

Byte 5/6：**BRG, return** UBRG value of EUART module, in slow initialization, if enable auto BaudRate detection, BaudRate may be changed, BRG value is the actual UBRG value when EUART communicates, user can directly determine the bus BaudRate according to this value, with no need to read UBRG BaudRate scaler register value.

Byte 7：SYNC Sync byte 0x55, if enable auto BaudRate detection in initialization, SYNC=0

Byte 8：**KEYWORD1** ECU responded protocol KeyWord1

Byte 9：**KEYWORD2** ECU responded protocol KeyWord2, user can determine the type of current protocol according to the keyword.

7.7 EUART BaudRate Setting

EUART module contains a special 16-bit BaudRate generator(BRG). UBRG register controls the cycle of one free-running 16-bit timer. It is very easy to set BaudRate, just firstly set UMODE<BRGH> to select high-speed BaudRate or low-speed BaudRate, then set the scaler register UBRG.

A. When **<UMODE>BRGH** = 0, the equation of calculating BaudRate is expressed as:

$$\text{Baudrate} = \frac{10MHZ}{4*(UBRG+1)} \qquad \text{UBRG} = \frac{10MHZ}{4 * \text{Baudrate}} - 1$$

Maximum BaudRate can be configured as: 10M/4 = 2.5 MHZ
Minimum BaudRate can be configured as: 10MHZ/(4 * 65536) = 38.1 HZ.
*UBRG=(10,000,000/(4*10400)) - 1 = 239*

B. When **<UMODE>BRGH** = 1, the equation of calculating BaudRate is expressed as:

Maximum BaudRate can be configured as: 10M/1 = 10 MHZ
Minimum BaudRate can be configured as: 10MHZ/65536 = 152.28 HZ。

e.g.: 10,400 bps UBRG calculation:
UBRG=(10,000,000 / 10400) - 1 = 960

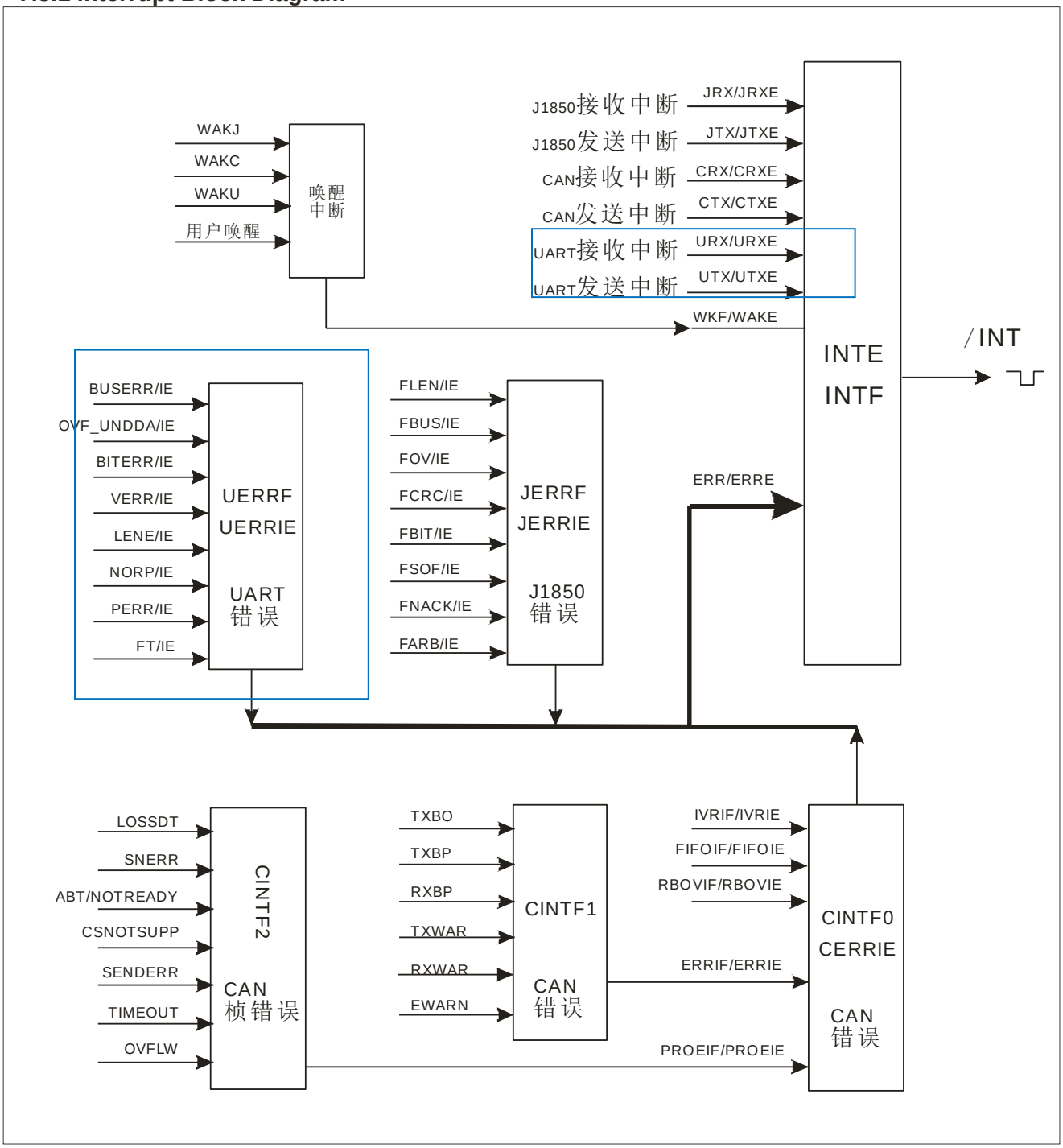
- UBRG will be reset(cleared) when a new value is written to UBRG register set. This ensures that BRG can generate new BaudRate without waiting for the timer overflow.

- K/LIN/SCI/SAE J1708 or other bus sets limit for maximum BaudRate, user must not exceed the BaudRate defined by physical bus.

- The selection of BRGH can use 1 or 0 according to different BaudRates, and smaller BaudRate error might be obtained. BaudRate error is generally no greater than 3%.

7.8 Interrupts Management

7.8.1 Interrupt Block Diagram



7.8.2 Transmit Buffer Interrupt

User must read INTF to clear CTX transmit buffer interrupt.

When UFRMCON<FMOD>= 1/2/3/4/5,

Transmit interrupt(INTF<UTX>) will be generated after the completion of message transmission from message buffer UTXB.

When UFRMCON<FMOD>=0,

In EUART operation character mode, EUART has 4-character-deep FIFO transmit buffer, one transmit offset register(UTSR register, user has no access).

User loads data into UTRXREG register(FIFO). New data will not be loaded into UTSR register before the stop bit transmission of the previous data loading. Once the stop bit transmission is completed, new data(if any) in UTRXREG register will be loaded into UTSR. Interrupt flag INTF<UTX> is set ON when data is loaded into UTSR, if interrupt allows INTE<UTXE> as ON, hardware interrupt will be generated.

7.8.3 Receive Buffer Interrupt

When UFRMCON<FMOD>=1/2/3/4/5/7,

When the message is successfully received and loaded in one receive buffer URXB(FIFO), INTF<URX> bit in the module will be set ON, and buffer receive interrupt is cleared by reading null FIFO receive buffer URXB.

When UFRMCON<FMOD>=0,

In EUART operation character mode, EUART has 4-character-deep FIFO receive buffer, one receive offset register(UTSR register, user has no access). Data is received at URX pin, after the stop bit at UxRX pin is sampled, data received in URSR is transported to receive FIFO(if UTRXREG is empty).

If there is any data in FIFO receive buffer(UTRXREG), INTF<URX> is set ON, user clears INTF<CRX> receive interrupt by reading null UTRXREG register.

7.8.4 Error Handling/Interrupt

UART receiver hardware has one 4-character-deep FIFO receive data buffer. UTRXREG is one register of memory mapping, which can provide the access to FIFO output. Before the buffer overflow, 4-character data can be received and transported to FIFO, and data is shifted to URSR register from Character 5.

7.8.4.1 Error Handling in Character Mode

When UFRCON<FMOD> = 0, EUART is processed by character mode.

- If FIFO is full(4 characters), and Character 5 has been fully received in URSR register, the overflow error bit USTAT0<OERR> and UERRF<OVF> will be set ON. Character in URSR will be held, but the data transmission to FIFO will be disabled provided that USTAT0<OERR> bit is set ON. User application program must use software to clear USTAT0<OERR> bit to enable the data transmission.

- If the received data is required to save before data overflow, user application program must firstly read all 5 characters, and then clear OERR bit. If these 5 characters can be discarded,

user application program just needs to clear USTAT0<OERR> bit. This can effectively reset the receive FIFO, when all prior received data will get lost.

- If the stop bit is detected as low logic level, then frame error bit USTAT0<FERR> and UERRF<BITERR> will be set ON.

- If the data byte(i.e.: current character) at the top of buffer is detected with odd/even parity check error, then odd/even parity check error bit USTAT0<PERR> will be set ON. For example, if odd/even parity check is set as even parity check, but the total number of "1" in the data is detected as odd number, odd/even parity check error will be generated.

- If any of these errors(OVF, BITERR and PERR) occurs, interrupt will be generated. User application program is requied to enable the corresponding interrupt before the bits in enable control register UERRIE can be generated.

In character mode, reception and transmission are processed by reading and writing UTRXREG, OVF, BITERR and BITERR of UERRF in error register is associated with USTAT0<OERR\PERR\FERR>, the state of error flag of other bits in UERRF always keeps 0 in character mode.

7.8.4.2 Error Handling in Frame Mode

When UFRCON<FMOD> =1/2/3/4/5/7, UART frame handling mode

- **BITERR/PERR** error: EUART data processing method in frame mode: UTRXREG must not be directly read or written, user's reading and writing are done by UTXB transmit message frame buffer and URXB receive message frame buffer.

  In message reception, character USTAT0<OERR> overflow error will not be generated, because the module directly loads the received data from URSR into the temporary message buffer. If the character error USTAT0<PERR/FERR> is generated, error flag bit UERRF<PERR> is associated with USTAT0<PERR>, and UERRF<BITERR> is associated with USTAT0<FERR>. User needn't attend to PERR/FERR/OERR error flag in USTAT0.

  In message transmission, the module loads the data in UTXB message buffer one by one into UTSR transmit offset register to start the character transmission. If one character data transmitted from UART K bus is in discrepancy with the returned data, UERRF<BITERR> error flag is also set ON. (This case will not occur in SCI bus, because two different physical lines are used for SCI bus reception and transmission respectively, the self-transmitted data will not be received)

In LIN Mode, if the odd/even parity check bit P1 P0 in the received ID identifier is found with error, then UERRF<PERR> error flag is also set ON.

- **Frame Overflow Error(OVF/UNDDA):** In message reception, the module loads the data in URSR receive offset register into the temporary message buffer(the same length with URXB, 262 characters) in UART receiver, when the frame ends, data is automatically loaded from the temporary message buffer into FIFO message receive buffer URXB, if URXB buffer is full(2-frame data), then UERRF<OVF> is set ON. Frame overflow error will be generated and temporary buffer message will be discarded. User cannot differentiate because the same flag bit is used for the reception of unknown data and frame overflow error.

-
- **Received Unknown Data Error<OVF/UNDDA>:** K line features the half-duplex bi-directional transmission, if frame transmission is in process(frame transmission is not completed), when the unknown data transmitted from the receiver is received, then UERRF<UNDDA> error flag is set ON. If interrupt is allowed, hardware interrupt output will be generated.

- **Frame Length Shortage Error<LENE>:** In frame mode, if the frame contains length mark(ISO14230/LIN), or enables length mark(user-defined frame mode FMODE=3), if the receive frame length is in shortage, then UERRF<LENE> error flag will be set ON. If interrupt is allowed, hardware interrupt output will be generated.

- **Frame Each Byte Check Error<VERR>:** When the receiver and sender is enabled to check each byte, if the data transmitted from sender is in discrepancy with the data returned from receiver, then UERRF<VERR> error flag will be set ON. If interrupt is allowed, hardware interrupt output will be generated.

- **Bus Error BUSERR:** If no data is returned after UART K bus sends one-character data, then UERRF<BITERR> error flag will be also set ON.
  After FT action, if the external drive chip SI9241 is in error state, not reset, UERRF<BUSERR> bus error will be generated in starting the transmission of data frame.

- **Failure Protection Action FT:** /FT generally connects to SI9241A failure protection output(ODC), when K line is short-circuit to ground or power, failure protection acts, /FT level lowers, when UTX output signal will not be transmitte to K bus, user needs to give a high level to /CS pin to reset.

Chapter 8 Enhanced EUART Module Related Applications

UART-based K line network is commonly used in low-speed vehicle communication network because each node is at a cheap price and there is a low physical requirement for hardware, and ISO14230（KWP2000）/ISO9141-2/LIN are the vehicle protocols in extensive use. And there are a lot of extended functions concerning manufactuer. User can refer to the following applications.

8.1 ISO14230(KWP2000) Application

8.1.1 ISO14230 Message Format

There are 4 message formats, and the transmission format is determined according to Byte 1 Fmt.

Description	Header				Data bytes	Checksum
1. Sing header, with no additional length byte	Fmt	-	-	-	Data[Byte 1-63]	CS
2. Single header, with additional length byte	Fmt	Len	-	-	Data[Byte 1-255]	CS
3. Three byte header, with no additional length byte	Fmt	Tgt	Src	-	Data[Byte 1-63]	CS
4. Three byte header, with additional length byte	Fmt	Tgt	Src	Len	Data[Byte 1-255]	CS
	Byte 1-4				Variable data length	1

Fmt: Form byte, Tgt: Target Addr. Src: Source Addr. Len: Length Byte CAS: CheckSum

Definition of Fmt Byte(Format Byte)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
A1	A0	L<4:0>					

A1:A0：Message Type

00 = No address information exists in message

01 = Exception mode(CARB), special mode, CARB form. ISO9141-2 OBD2 Fmt=0x68/0x48 specifically for diagnosis

10 = Message with the information of Physical Addr.

11 = Message with the information of Functional Addr.

L<5:0>：6-bit Length Information

L=0 Message requires an additional length byte Len, where the data length in frame can be 1-255 bytes.

L=1-63 Message has no additional length byte, where the data length in frame is L<5:0> value.

ISO14230-2 Data Link layer Note: For more details, please refer to ISO14230-2 Data Link layer

8.1.2 Configuration Register/Register Set

8.1.2.1 Configuration Bus Initialization Register

There are two ways of ISO14230 initialization, fast initializationor 5 Baud slow initialization, with the following registers configured.
User transmits the first request message, which way of initialization is selected:

INITLT=25	Low level time of fast initialization25ms
INITHT=25	High level time of fast initialization25ms
U5BAUDCON . RLEN=3	Firstly receive 3 bytes in initialization
U5BAUDCON .ENL=0	ISO14230 generally doesn't use L line
U5BAUDCON .KW2T=1	Return the reversed Keyword2 to the bus line
U5BAUDCON .INVAD=1	Receive the reversed Physical Addr.
U5BAUDCON .ABAUD=1	Auto BaudRate detection enable
U5BAUDCON .W1MAX=50	Maximum waiting time of receiving sync character=50*20=1000ms
U5BAUDCON .W2MAX=100	Maximum waiting time of receiving keyword=100ms
U5BAUDCON .W4MAX=100	Maximum waiting time of receiving ECU returned reversed trigger address=100ms
U5BAUDCON .W4SEND=30	Time delay of the module transmitting the reversed KW2 byte=30ms
U5BAUDCON .USERKW=0x00	Irrelevant
INITADDR=0x33	For the trigger address in 5 Baud transmission, 0x33 is the Functional Addr. specifically used for OBD2. If for any other ECU, this address generally is one value corresponding to one ECU, e.g.: ABS module is generally 0x28

8.1.2.2 Configuration Message Related Time Register

UTXP1T=5	When data is transmitted by tester, the minimum interbyte space P4MIN=5ms If for special use, UTXP1T can be set OMS, accelerating the time of message transmission. But the general application must be set above 5MS.
UFREET=55	Bus release time 55ms
URXP1T=20	Allowed maximum interbyte space in reception 20ms

8.1.2.3 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x00	Select UTX1/URX1 pin
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps In slow initialization enable auto BaudRate, this value may vary
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit

USTAT0=0x00 USTAT1=0x00	State bit reset

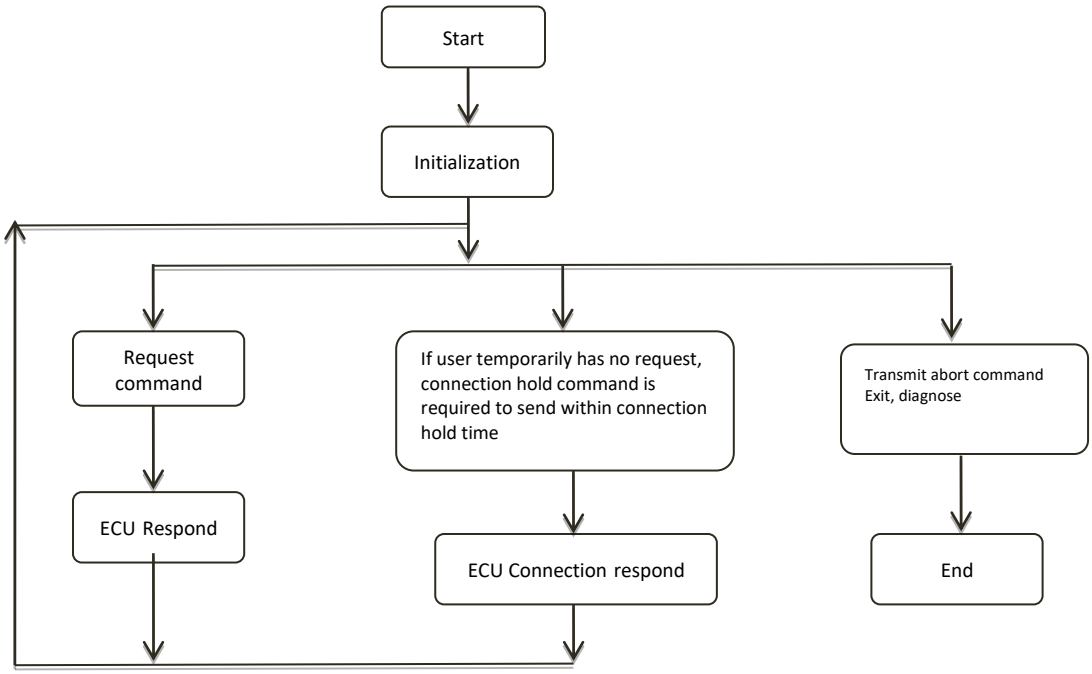
8.1.2.4 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UPRICON=0x00	Disable J1708 priority
UFRMCON=0x81	FMOD=1 Select ISO14230 protocol ON=1 Start EUART module

After EUART is started, bus message transmission can be started. The module will receive data if there is any data in the bus.

8.1.3 K-line Diagnostic Process

ISO14230/ISO9141-2/KW1281 is basically the same in K-line diagnostic process.



8.1.4 Transmit/Receive Message

8.1.4.1 Define C Structure of One Message Recepive/Transmit Buffer

Refer to EUART Receive/Transmit Buffer in 7.5 , 7.6

```
struct
{
    union                                          Frame information byte corresponds to one C union
    {
        struct{
         unsigned short int FLen   : 9;            Frame valid data length 9 bits
         unsigned short int        : 2;            2 bits not used
         unsigned short int CSERR  : 1;            Frame check and flag in reception, this bit is neglected in
                                                   transmission
         unsigned short int INIT   : 3;            Initialization mode
         unsigned short int SACK   : 1;            In LIN Mode, slave response attribute, ISO1230 not used, kept
                                                   as 0
        };
         unsigned short int  FrameInfo;            Frame information byte
    };
    unsigned char      Data[260];                  Transmit/receive the content of data frame with up to 260 bytes

}EUART_REPORT;                                     EUART message structure
```

Notes:

Data length is 260 bytes, frame information is 2 bytes, this structure is defined as one variable that occupies the memory of 262 bytes, this is designed as per ISO14230 maximum standard, after one block of memory is applied, user just needs to set up the memory data, and directly write the data in memory into EUART module UTXB transmit buffer to start the transmission.

A block of 262-byte memory is applied in WINDOWS, which can be negligible for computer. However, if applied in microcontroller system, enough memory must be equipped for the implementation of full function. But sometimes user may only use the simple function, e.g.: if OBD2 diagnosis is only made, Data[8]+Frame Information 2 byte can be defined, with the memory of total 10 bytes, and the data frame in excess of 8 bytes can be neglected.(OBD2 command and respond data flow only contains 8 bytes(including CS)).

8.1.4.2 ISO14230 and ECU Communication Setup(Fast Initialization):

ISO14230 Fast initialization Setup Process

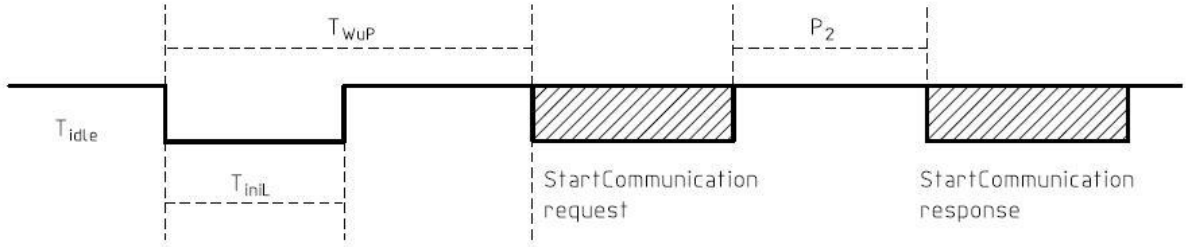


Figure 11 — Fast initialization

Table 10 — Timing values for fast initialization

Parameter		min value, ms	max value, ms
T_{iniL}	25 ± 1 ms	24 ms	26 ms
T_{WuP}	50 ± 1 ms	49 ms	51 ms

ISO14230 Fast Initialization Setup Operation

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.1.2 and start EUART Set Register(Set) in 8.1.2	When user doesn't send bus transmitted data, only prepares the module
2. Send the first message of ISO14230 connection setup(fast initialization)： Message content of communication startup: 81 xx F1 81 <CS>, CS must be loaded after user's calculation. Actual operation is:	Message command：0x81（Start Communication Request） ECU Physical Addr.：xx Tester Physical Addr. F1, Command length is 1 byte, Fmt<L>=1. Fmt A1:A0= 10, total length of data frame is 5 bytes
UTxPacket .FrameInfo=0x0000; UTxPacket .INIT = 0b001; UTxPacket .Flen = 5 ; UTxPacket .Data[0] = 0x81; UTxPacket .Data[1] = <XX>; UTxPacket .Data[2] = 0xF1; UTxPacket .Data[3] = 0x81; UTxPacket .Data[4] = <CS>;	Frame information byte is cleared Firstly process fast initialization(Time INITLT/INITHT) Length of data frame=5, message transmitted after 25MS low+25MS high time Fmt Tgt=ECU Physical Addr. Src SID Command =0x81 Check and CS, (Loaded after user calculates)
3. Write data packet into UTXB module transmit buffer, with total length of 7 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10 SpiWriteLongBuff(UTXB , 7 , (unsigned char *) &UTxPacket);	Start transmission 3. Write 7 bytes into UTXB, if bus released, immediately start the transmission. 4. Because INIT=001, in transmission, firstly start the fast initialization pulse(25MS low electric pulse

	25ms high level), start the transmission after completion: 81 xx F1 81 <CS> message.
4. User waits for ECU response User application software, if within P2MAX(maximum 5S) time, no ECU response is received, repeat Step 3 for reinitialization. If no response after 3 repeats, it indicates that connection fails.	ECU responds to 0x81 after the reception of initialization message. ECU respond time is P2(ISO14230 defined), because ET7190 is always in a state of reception after being started, ET7190 can receive any ECU response.
If data is normally received(StartCommunication Repond from ECU) 83 F1 xx C1 EF 8F <CS> Data read to URxPacket for data processing	For the response to 0x81, ECU respond format is: 83 F1 xx C1 <KeyWord1> <(KeyWord2> <CS> C1=0x81+0x40 xx ECU Physical Addr., EF 8F ISO14230 protocol defined keyword KeyWord2 must be 0x8F in ISO14230
5. Other RTS after normal communication setup Generally SID=0x10(Start Diagnosis Request) is firstly sent e.g.：82 xx F1 10 80 <CS> = Request 82 F1 xx 50 80 <CS> = ECU respond	User can set the message format according to the received keyword. Refer to ISO14230-2 for the definition of KEYWORD1/KEYWORD2 SID=0x10 Note: ISO14230 defines all requests(REQUEST) of BIT6=0 in SID, ECU respond of BIT6=1 in SID;

8.1.4.3 ISO14230 and ECU Communication Setup(Slow Initialization):

Slow initialization process is given below;

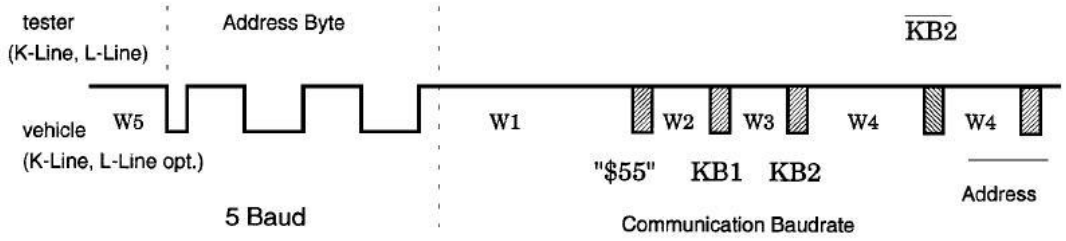


Figure 10 — 5 Baud initialization

able 9 shows timing values for 5 Baud initialization. These are fixed values. They cannot be changed by the ccessCommunicationParameter service.

Table 9 — Timing values for 5 Baud initialization

Timing parameters	Values ms		Description
	min.	max.	
W1	60	300	Time from end of the address byte to start of synchronization pattern.
W2	5	20	Time from end of the synchronization pattern to the start of key byte 1.
W3	0	20	Time between key byte 1 and key byte 2.
W4	25	50	Time between key byte 2 (from the ECU) and its inversion from the tester. Also the time from the inverted key byte 2 from the tester and the inverted address from the ECU.
W5	300	—	Time before the tester starts to transmit the address byte.

ISO14230 5 Baud Initialization Setup Operation

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.1.2 and start EUART Set Register(Set) in 8.1.2 INITADDR is ECU Physical Addr., INITLT/INITHT are irrelevant	At that time, user doesn't send bus transmit data, just prepare the module
2. In ISO14230 slow initialization, connection setup is completed in the process of initialization(Different from fast initialization, use SID=0x81 request in connection setup). So message is not transmitted in normal case of initialization. Set UTxPacket.Flen=0 Actual operation is：	
UTxPacket .FrameInfo=0x0000; UTxPacket .INIT = 0b010;	Frame information byte is cleared In 5 Baud initialization, the initialization process is

UTxPacket .Flen = 0 ;	U5BAUDCON setting Data frame length=0, only initialization
3. Write data packet into UTXB module transmit buffer, with total length of 2 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10 SpiWriteLongBuff(UTXB , 2 , (unsigned char *) &UTxPacket);	Start transmission(Because of no data frame, only write 2 bytes information character) 3. Write 2 bytes into UTXB, if bus released, immediately start the transmission. 4. Because INIT=010, in transmission, start 5 Baud initialization.
4. User waits for ECU response After user starts the slow initialization, the content of one initialization information will be received, see 7.6.2 for the data structure. If **INIT_ERRFLAG** is not cleared, it indicates the initialization bus error, repeat Step 3 for reinitialization. If no initialization is available after repeating for 3 times, it indicates connection failure.	
5. Normally received initialization information Read the initialization information(in URXB) to URxPacket receive buffer Refer to 7.6.2, set according to KW1/KW2 keyword(If required) 6. Other RTS after normal communication setup Generally SID=0x10(Start Diagnosis Request) is firstly sent e.g.: 82 xx F1 10 80 <CS> = Request 82 F1 xx 50 80 <CS> = ECU respond	Note: In 5 Baud initialization, if enable ABUAD, then UBRG value may be changed, user can obtain the actual bus BaudRate according to BYTE5/BYTE6 in the initialization information Subsequent diagnostic process of fast/slow initialization is identical.

8.2 ISO9141-2(USERMOD) Application

8.2.1 ISO9141-2 Message Format

ISO9141-2 message format is actually a special ISO14230 defined format, A1:A0=0b01(CARB format) in Fmt byte, with the transmission format listed below:

Description	Header			Data bytes	Checksum
Three bytes header, with no length byte	Fmt=0x68/0x48	Tgt	Src	[1-255 bytes]	CS
	3 bytes			Variable data length	1

Fmt: Format Byte, Tgt: Target Addr. Src: Source Addr. CS: Checksum

- ISO9141-2 Because of no length byte, frame end is determined by the transmitted time of data frame in message reception, for ET7190 handling method, when one frame reception is started, if the next byte is not received within URXP1T time, it is deemed as frame completion.

- Data Frame Length: OBD2 is defined with up to 7 bytes, but it is common for over 7 bytes applied in vehicle manufacturer. So data frame with longer byte is enabled to process when we do the

programming.

- ISO9141-2 initialization process is the same with ISO14230 slow 5 Baud initialization process, only the definition of keyword value is different.
  ISO9141-2 Keyword is fixed as KW1=KW2=0x08 or 0x94; (Refer to ISO9141-2)
  ISO9141-2 No fast initialization mode.

8.2.2 Configuration Register/Register Set

8.2.2.1 Configuration Bus Initialization Register

ISO9141-2 only has 5 Baud slow initialization, with the following registers configured.

INITLT=25	Irrelevant
INITHT=25	Irrelevant
U5BAUDCON Configuration is the same with ISO14230	
U5BAUDCON . RLEN=3	Firstly receive 3 bytes in initialization
U5BAUDCON .ENL=1	ISO14230 generally doesn't use L line
U5BAUDCON .KW2T=1	Return the reversed Keyword2 to the bus line
U5BAUDCON .INVAD=1	Receive the reversed address
U5BAUDCON .ABAUD=1	Auto BaudRate detection enable
U5BAUDCON .W1MAX=50	Maximum waiting time of receiving sync character=50*20=1000ms
U5BAUDCON .W2MAX=100	Maximum waiting time of receiving keyword=100ms
U5BAUDCON .W4MAX=100	Maximum waiting time of receiving ECU returned reversed trigger address=100ms
U5BAUDCON .W4SEND=30	Time delay of the module transmitting the reversed KW2 byte=30ms
U5BAUDCON .USERKW=0x00	Irrelevant
INITADDR=0x33	For the trigger address in 5 Baud transmission, 0x33 is the Functional Addr. specifically used for OBD2. If for any other ECU, this address generally is one value corresponding to one ECU, e.g.: ABS module is generally 0x28

8.2.2.2 Configuration Message Related Time Register

The same with ISO14230

UTXP1T=5	When data is transmitted by tester, the minimum interbyte space P4MIN=5ms If for special use, UTXP1T can be set OMS, accelerating the time of message transmission. But the general application must be set above 5MS.
UFREET=55	Bus release time 55ms
URXP1T=20	Allowed maximum interbyte space in reception 20ms

8.2.2.3 Configuration EUART Bit Definition and BaudRate

The same with ISO14230

UPINSEL=0x00	Select UTX1/URX1 pin
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps In slow initialization enable auto BaudRate, this value may vary
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00 USTAT1=0x00	State bit reset

8.2.2.4 Configuration Length Control and Check Control(User-defined mode must be configured)

UFRMV =0x80 UFLENCON=0	Only enable checksum ENCS=1 Disable frame length, CSM=0(ISO14230 checksum)
UFRMV =0x80 UFLENCON=0	Only enable checksum ENCS=1 Disable frame length, CSM=0(ISO14230 checksum)

8.2.2.5 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UPRICON=0x00	Disable J1708 Priority
UFRMCON=0x83	FMOD=3 Select<User-defined Mode> Protocol ON=1 Start EUART Module

Bus message transmission is started after EUART startup. The module will receive data if there is any data in the bus.

8.2.3 Transmit/Receive Message

The process is identical to that of ISO14230 slow initialization, only the data frame format and data definition are different.

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.2.2 and start EUART	At that time, user doesn't send bus transmit data, just prepare the module
Set Register(Set) in 8.2.2 INITADDR is ECU Physical Addr., INITLT/INITHT are irrelevant	In OBD2, INITADDR=0x33(Functional Addr.)
2. Connection In ISO9141-2 slow initialization, connection setup is completed in the process of initialization. So message is not transmitted in normal case of initialization. Set UTxPacket.Flen=0。 Actual operation is：	Also after the initialization, send one frame request message e.g.： 68 xx F1 01 00 <CS>
UTxPacket .FrameInfo=0x0000; UTxPacket .INIT = 0b010; UTxPacket .Flen = 0 ;	Frame information byte is cleared In 5 Baud initialization, the initialization process is U5BAUDCON setting Data frame length=0, only initialization
3. Write data packet into UTXB module transmit buffer, with total length of 2 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10	Start transmission(Because of no data frame, only write 2 bytes information character)
SpiWriteLongBuff(UTXB , 2 , (unsigned char *) &UTxPacket);	5. Write 2 bytes in UTXB, if bus released, immediately start the transmission. 6. Because INIT=010, in transmission, start 5 Baud initialization.
4. User waits for initialization completion After user starts the slow initialization, the content of one initialization information will be received, see 7.6.2 for the data structure. If **INIT_ERRFLAG** is not cleared, it indicates the initialization bus error, repeat Step 3 for reinitialization. If no initialization is available after repeating for 3 times, it indicates connection failure.	
5. Normally received initialization information Read the initialization information(in URXB) to URxPacket receive buffer Refer to 7.6.2, set according to KW1/KW2 keyword(If required) 6. Other RTS after normal communication setup In OBD2, generally send 01 00(MOD=1 PID=0) e.g.： 68 6A F1 10 80 <CS> = Request 48 6B xx AA BB CC DD <CS> = ECU respond	Note: In 5 Baud initialization, if enable ABUAD, then UBRG value may be changed, user can obtain the actual bus BaudRate according to BYTE5/BYTE6 in the initialization information Subsequent diagnostic process of fast/slow initialization is identical. Tester request ECU response, xx is Physical Addr.

8.3 KWP1281(USERMOD) Application

KW1281 is one of BOSCH diagnostic communication protocols, which has been extensively applied in vehicle. The difference between KW1281 and ISO9141 configuration is that KW1281 requires the configuration of frame length and interframe reversed check of each byte.

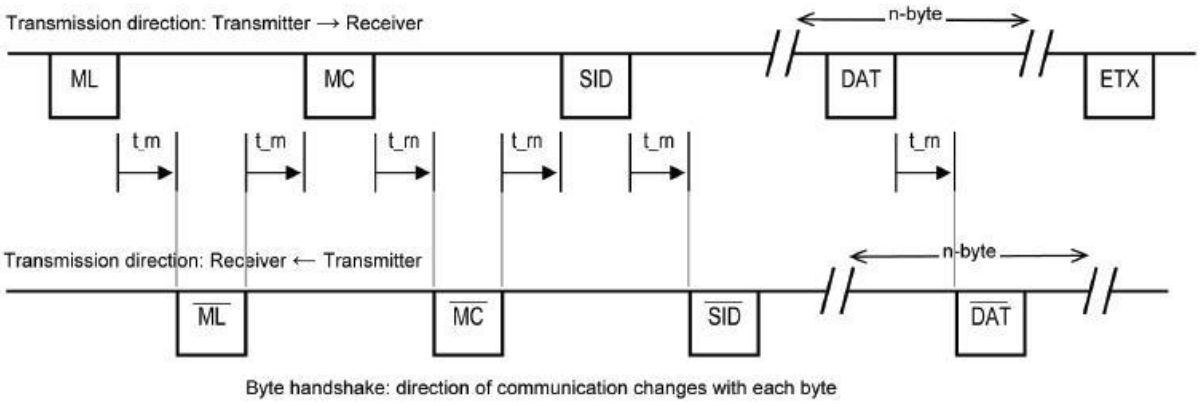
8.3.1 KW1281 Message Format

Message format is listed below:

ML	MC	SID	DAT	ETX=(0x03)
1 byte	1 byte	1 byte	n bytes	1 byte

- ML：Message Length, byte value= n+3.
- MC：Message counter, MC plus 1 for each transmission of one-frame data from 0.
- SID：Service identification.
- DAT：n bytes data, maximum value ET7190 allowed 252 bytes(ML=255, 0xFF).
- ETX：Frame end flag, the value is always 0x03.

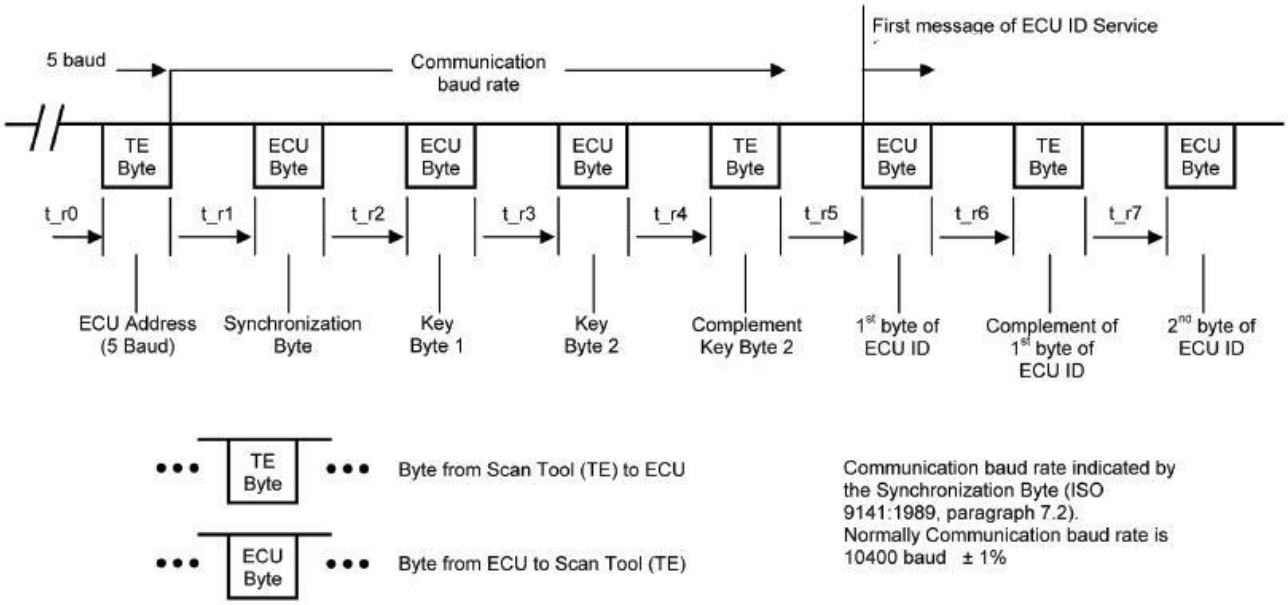
Message/Frame Transmission Process:



- In message transmission, the reversed value of each byte is required from receiver for check, and the last ETX is not checked.
- T_rn is the time delay of receiver returning the check byte.
  Tester is generally set as 2ms, and ECU T_rn is set as 1ms.

8.3.2 KW1281 Initialization Process

Prior to the communication, tester(active module) requires an initialization process for communication setup, and the initialization method is similar to that of ISO9141-2, as shown below:



- Tester(TE) firstly transmits a 5 Baud trigger address(ECU Physical Addr.).
- ECU respond 3 bytes of Physical Addr.: Sync(0x55) bytes, and two key bytes, KB1 and KB2. In KW1281 protocol, KB1 is always 0x01, and KB2 is always 0x8A.
- After the reception of 3 bytes by tester, one reversed KB2 byte(0x75) is transmitted to ECU.
- ECU starts the transmit FF data of ECU ID message.
- According to the above process, after the initialization is started, if ET7190 successfully connects to ECU, one message with completed initialization and FF data of ECU ID message will be received in URXB.
- Tr0：Bus idle time>300ms

  Tr1：The time that ECU starts to respond to sync byte, at which BOSCH is also nonstandard, ECU in the early stage can be up to 1000ms, but the new standard requires ECU to be defined in 80ms-210m.

  Tr2：MIN: 5ms MAX: 20ms

  Tr3: MIN: 1ms MAX: 20ms

  Tr4: MIN: 25ms MAX: 50ms

  Tr5: MIN: 25ms MAX: 50ms

  Tr6: MIN: 1ms MAX: 20ms Generally as 2ms

  Tr7: MIN: 1ms MAX: 20ms Generally as 1ms

8.3.3 KW1281 Message

KW1281 message may be the one consisting of one or more frames, KW1281 is not defined under the network link layer, KW1281 multi-frame message defines the transmission and reception in the application layer, user needs to process the frame-to-frame transmission and multi-frame combination of message, and the reception process of common request is described below: (e.g.: Read ECU identity by SID=00)

03 0A 00 03	Tester sends one SID=00 request ML=3, MC=0x0A
0F 0B F6 B1 4A 44 39 32 30 38 32 36 20 20 20 03	ECU returns Frame 1 data of message ML=0F MC+1=0x0B
03 0C 09 03	Tester sends one SID=09 link hold frame
0F 0D F6 4B 4F 4D 42 49 2B 57 45 47 46 41 48 03	ECU returns Frame 2 of message, SID=F6
03 0E 09 03	Tester continues transmitting SID=09
0E 0F F6 52 53 50 20 56 44 4F 20 56 30 32 03	ECU returns Frame 3 of message, SID=F6
03 10 09 03	…
08 11 F6 00 28 04 00 32 03	Tester sends SID=09
03 12 09 03	ECU returns SID=09, indicating the message completion.
03 13 09 03	
…	…
03 14 09 03	Tester sends SID=09(If diagnosed below 1000ms. With no other task. Must send link hold)
03 15 09 03	ECU Link acknowledgement

For a request, if the multi-frame message responds, diagnosis must send SID=09 message request to receive the next frame, until ECU responds to SID=09 message, which indicates the completion of message.

8.3.4 Configuration Register/Register Set

8.3.4.1 Configuration Bus Initialization Register

KW1281 only has 5 Baud slow initialization, with the following registers configured.

U5BAUDCON Configuration	
U5BAUDCON . RLEN=3	Firstly receive 3 bytes in initialization
U5BAUDCON .ENL=0	Generally doesn't use L line, L line can be set ON if used.
U5BAUDCON .KW2T=1	Return the reversed Keyword2 to the bus line
U5BAUDCON .INVAD=0	In KW1281 initialization, no address returned from ECU,
U5BAUDCON .ABAUD=1	Auto BaudRate detection enable

U5BAUDCON .W1MAX=50	Maximum waiting time of receiving sync character=50*20=1000ms
U5BAUDCON .W2MAX=100	Maximum waiting time of receiving keyword=100ms
U5BAUDCON .W4MAX=100	Maximum waiting time of receiving ECU returned reversed trigger address=100ms
U5BAUDCON .W4SEND=30	Time delay of the module transmitting the reversed KW2 byte=30ms
U5BAUDCON .USERKW=0x00	Irrelevant
INITADDR=0x01 Odd parity check: The number of all bits as "1" in INITADDR is odd. E.g.: Instrument address is 0x17, 4 bits contains "1", so BIT7 must be 1. INITADDR=0x97, consequently INITADDR contains 5 bits(odd) in "1".	ECU initialization address is valued between 01 and 7F, 127 in total, the address is 7-bit data, Bit 8(highest bit) is the odd parity check bit of this address value. You must calculate the value of this bit, after forming a 8-bit byte with ECU 7-bit address, the initialization address of ECU INITADDR is set. The initialization address of common ECU module is described below: 0x01 Engine Electrical 0x02 Auto Transmission 0x15 Airbag SRS 0x03 ABS System 0x08 Air-conditioning System 0x17 Instrument and Anti-theft 0x46 Comfort System 0x41 Diesel Pump Electrical 0x44 Power-assisted Steering 0x22 All-wheel Drive 0x14 Suspension System 0x56 Radio Cassette Player 0x19 CAN Gateway 0x39 Right Light Control 0x11 Engine 11 0x21 Engine 21 0x28 Rear Air-conditioning Electrical 0x12 Electrically-controlled Clutch 0x13 Distance Control 0x0E Multimedia Player 0x24 Acceleration Slip Regulation 0x34 Body Leveling 0x75 Emergency Call 0x55 Headlamp Control 0x29 Left Light Control 0x45 Internal Monitor 0x16 Steering Wheel Electrical 0x26 Auto Roof 0x47 Audio System 0x36 Driver Seat 0x18 Auxiliary Heater 0x65 Tire Pressure Monitor 0x25 Electronic Alarm 0x76 Parking Assist 0x77 Telephone 0x09 Central Electrical 0x49 Auto Light Switch 0x2F Digital TV 0x35 Central Door Locking

8.3.4.2 Configuration Message Related Time Register

UTXP1T=5	Minimum interbyte space 5ms when the tester sends data
UFREET=55	Bus release time 55ms
URXP1T=20	Allowed maximum interbyte space 20ms in reception
UTXVT0=20	In transmission, the tester receives the allowed maximum check time 20ms returned from ECU

URXVT1=2	When ECU transmitted frame is received, time delay of transmit check byte 2ms

8.3.4.3 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x00	Select UTX1/URX1 pin Some module uses K2 line, then UPINSEL=0x05 (RXSEL=1，TXSEL=1)
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps In slow initialization enable auto BaudRate, this value may vary KW1281 BaudRate 1200/4800/9600/10400, generally enable auto BaudRate detect in initialization
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00 USTAT1=0x00	State bit reset

8.3.4.4 Configuration Length Control and Check Control(User-defined Mode must be Configured)

UFRMV = 0xE1	ENCS=0 Disable checksum OFFSET=1, Length byte(ML) + 1(OFFSET) is the actual total length of frame VEND=0, The last in-frame byte is not checked VFINV=1, In check, use the reversed value FTVEN=1, In transmission, enable each byte check FRVEN=1, In reception, enable each byte check
UFLENCON= 0x80	EN=1, Enable frame length, CSM=0 (Irrelevant, KW1281 no checksum, UFRMV.ENCS=0) LENP=0, In-frame Byte 0 is the length byte LRR=0, No right shift in calculating length
UFLAND = 0xFF	In calculating length, length byte(ML) doesn't mask any bit The calculation of actual length after configuration is: (ML>>LRR) & UFLAND + OFFSET

8.3.4.5 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt,

	URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UPRICON=0x00	Disable J1708 priority
UFRMCON=0x83	FMOD=3 Select<User-defined Mode> Protocol ON=1 Start EUART Module

The bus message transmission is started after EUART is started. The module will receive data if there is any data in the bus.

8.3.5 Transmit/Receive Message

This process is similar to that of ISO9141-2 slow initialization.

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.3.4 and start EUART Set Register(Set) in 8.3.4 INITADDR is ECU Physical Addr., INITLT/INITHT are irrelevant	At that time, user doesn't send bus transmit data, just prepare the module
2. In slow initialization, connection setup is completed in the process of initialization. So in initialization, after completed, ECU returns the multi-frame identification information(compared to SID=00 request), so must set UTxPacket.Flen=0.(Data frame must not be transmitted in initialization). Actual operation is：	
UTxPacket .FrameInfo=0x0000; UTxPacket .INIT = 0b010; UTxPacket .Flen = 0 ;	Frame information byte is cleared In 5 Baud initialization, the initialization process is U5BAUDCON setting Data frame length=0, only initialization
3. Write data packet into UTXB module transmit buffer, with total length of 2 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10	Start transmission(Because of no data frame, only write 2 bytes information character)
SpiWriteLongBuff(UTXB , 2 , (unsigned char *) &UTxPacket);	9. Write 2 bytes into UTXB, if bus released, immediately start the transmission. 10. Because INIT=010, in transmission, start 5 Baud initialization.
4. User waits for the initialization completion After user starts the slow initialization, the content of one initialization information will be received, see 7.6.2 for the data structure. If **INIT_ERRFLAG** is not cleared, it indicates the initialization bus error, repeat Step 3 for reinitialization. If no initialization is available after repeating for 3 times, it indicates connection failure. If initialized normally, ECU returns FF of identification information message, after FF is received, user must send SID=09 request, receive the next frame, until SID=09 response is received, indicating the message completion.	
5. Normally received initialization information Read the initialization information(in URXB) to URxPacket receive buffer Refer to 7.6.2, set according to KW1/KW2(01 8A) keyword(If	Note: In 5 Baud initialization, if enable ABUAD, then UBRG value may be changed, user can obtain the actual bus BaudRate according to BYTE5/BYTE6 in the initialization information

required) 6. Other RTS after normal communication setup	If tester temporarily requires no other instruction, SID=09 is required to be sent as link hold command within 1000ms(generally set as 500ms), or link will be disconnected. ECU communication is enabled after reinitialization

8.3.6 KW1281 Common Commands

Function	SID	Command Format	
Basic Setting	0x28	04 xx 28 <CH> 03	Byte 4 CH is channel number
Basic Setting(CHANNEL=0)	0x11	03 xx 11 03	Basic setting when the channel is 0
ECU Coding	0x10	07 xx 10 AA BB CC DD 03	AABBCCDD is the code and coding of service station
ECU Identification Information	0x00	03 xx 00 03	Read the multi-frame ECU identification information
Read Single Parameter	0x08	04 xx 08 <PARA> 03	Byte 4 is the parameter number 0-255
Read TroubleCode	0x07	03 xx 07 03	Multi-frame message may be returned
Clear TroubleCode	0x05	03 xx 05 03	
Read Parameter Block	0x29	04 xx 29 <CH> 03	Byte 4 CH is channel number
Read the Parameter Block with Channel Number as 0	0x12	04 xx 12 03	
Element Test	0x04	04 xx 04 00 03	
Link Holding	0x09	03 xx 09 03	
Negative Acknowledge	0x0A	ML MC 0A -- -- 03	In ECU respond, it represents request not supported length is related to request

8.4 SAE J1708(USERMOD) Application

J1708 is the RS-485-based SAE standard, which can be applied in agricultural vehicle, commercial vehicle and heavy machinery, as released and maintained by SAE(Society of Automotive Engineers). Primarily used for the serial communication between ECUs of heavy vehicle or between vehicle and computer. Connected to OSI model, J1708 mainly defines the physical layer, common higher-layer protocol operated above J1708 is SAEJ1587(TMC/SAE union microcontroller system on heavy vehicle for the exchange of electronic data) and SAEJ1922(used for the electrically-controlled transmission interface of medium-sized and heavy-sized road diesel vehicle).

This standard defines a 2-wire 18 gauge wire cable, the communication rate is 9600bit/s in operation, and the length can be up to 40m. 1 message can contain 21 characters, transmitter only enables the transmission of message with length up to 21 bytes when the engine or vehicle stops. Message starts from one message ID(MID), which ends at a checksum. Character takes in the form of common 8 data bits and 1 stop bits with no transmission of odd/even parity check format.

Hardware uses RS-485 transreceiver to connect to open collector operation(see wire diagram in 3.2.1), which pulls up and down via the separate data cable. Transmission is done by controlling the enable pin of transreceiver of driver. This method enables multiple devices to share the bus line with no need of a master node. Collision can be avoided by the monitor bus and priority setting, another node is ensured with no transmission of another higher-priority MID when one MID is transmitted.

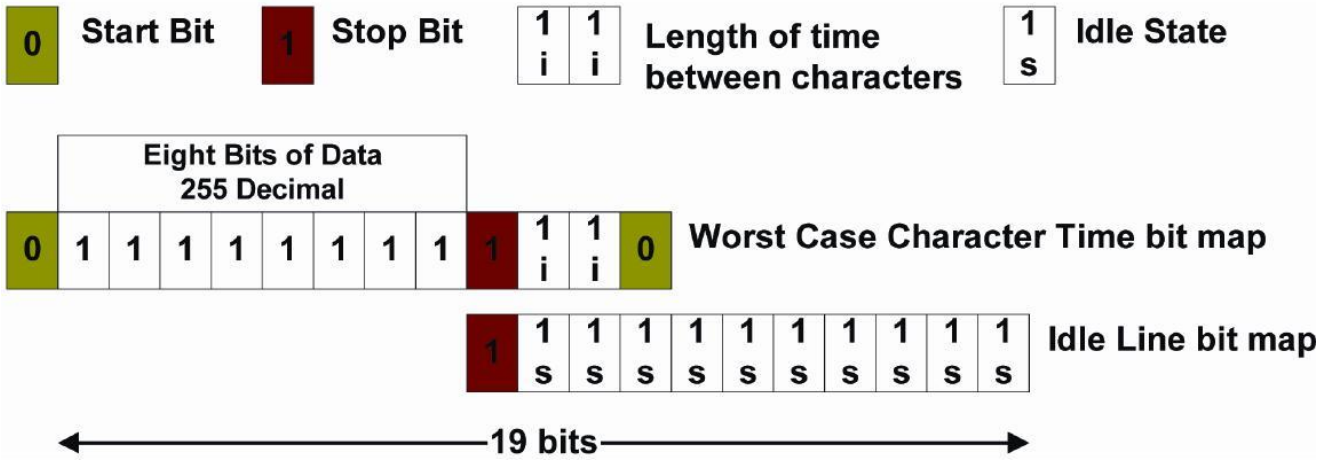
8.4.1 J1708 Message Format

Message Identification	Data Characters	Checksum
MID	n Data	CS

MID： Any value in 0-255
Data: Data, total length(including MID and CS) is generally in 2-21, it is allowed to be greater than 21 bytes when the engine is not operated, ET7190 maximum enable transmit or receive length is 260-byte message. Data byte can be none.
CS: Checksum, refer to J1708 text for algorithm. When ET7190 is configured, UFLENCON<CSM> bit is set ON.

8.4.2 J1708 Bit Timing

We only give a brief description, please refer to SAE J1708 text for more details.



- When sender transmits message, maximum allowed interbyte space is 2-bit time. If over 2 bits, the bus enters the idle state before Bit 3 appears the start bit.

- When the bus is in idle state, the transmission of next frame is started with the minimum 12-bit time. The node with the highest priority begins to send data(if multiple nodes are required to send data simultaneously, the node with the highest priority firstly obtains the bus), the priority determines the time of node starting the transmission.

- If two nodes with the same priority transmit at the same time, bus conflict may occur, if such conflict is detected, user must randomly change the priority for retransmission.

- The priority of node is in Priority 1-8, where 1 is the highest priority and 8 is the lowest.

- The calculation of node obtaining the bus time:
  T(Time) = (10 Bit Time) + (2 Bit Time) * P

  P is Priority(1-8), the highest priority is indicated when P=1, the obtained bus time is: 12=(10+2*1) Bit Time, the lowest priority is indicated when P=8, the obtained bus time is 26-bit Time.
- User needn't to concern SAE J170 bit time in using ET7190. ET7190 automatically handles bit time according to the user-defined priority, user only needs to write UTXB transmit message or read URXB receive message(as long as UTXB is empty, message can be written in UTXB buffer at any time, and the transmission is automatically started after the current module obtains the bus line).

8.4.3 Configuration Register/Register Set

J1708 doesn't define fast or slow initialization, so U5BAUDCON may not set. In transmission of message, the bus initialization prior to the transmission must not be enabled. Actually J1780 is a special example of FMOD=3(user-defined mode), if ET7190 requires, enable use such as the initialization method in ISO14230.

8.4.3.1 Configuration Message Related Time Register

In using J1708, the following related 3 time registers must be set Clear:

UTXP1T=0	Interbyte space in transmitting J1708 must be set 0ms
UFREET=0	When UPRICON<J1708E> is 1 enable J1708, the bus release time is determined by priority. UFREET must be set Clear
URXP1T=0	When UPRICON<J1708E> is 1 enable J1708, maximum allowed interbyte space in reception is 2-bit time, URXP1T must be set Clear,

8.4.3.2 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x0A	Select UTX3/URX3pin(if use) (RXSEL=2, TXSEL=2)
UBRG=259 (0x0103)	BaudRate scaler, 9600bps
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00	State bit reset
USTAT1=0x40	UTXINV=1; when MAX485 is used, TX polarity must be reversed, the state is 0 in idle state. Keep the bus in recessive state.

8.4.3.3 Configuration Length Control and Check Control(User-defined Mode must be Configured)

UFRMV = 0x08	ENCS=1 Enable checksum
UFLENCON= 0x08	EN=0, no frame length, CSM=1 (J1708 checksum mode)

8.4.3.4 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x0
UPRICON=0x80	Enable J1708 Priority(PRI is set as 0-7. Configured as required)
UFRMCON=0x83	FMOD=3 Select <User-defined Mode> Protocol ON=1 Start EUART module

After EUART is started, the bus message transmission can be started. The module will receive data if there is any data in the bus.

8.4.4 Transmit/Receive Message

Message transmission and reception, actually data transmission and reception are operated in the same way when FMOD is in any mode other than 0, message is transmitted when data is written in UTXB transmit buffer. When the module receives the message, URXB receive buffer data is read.

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.4.4 and start EUART	At that time, user doesn't send bus transmit data, just prepare the module
2. Write UTXB to directly transmit message	
UTxPacket .FrameInfo=0x0000; //UTxPacket .INIT = 0b000; UTxPacket .Flen = 5 ; UTxPacket .Data[0] = MID; UTxPacket .Data[1] = AA UTxPacket .Data[2] = BB; UTxPacket .Data[3] = CC; UTxPacket .Data[4] = <CS>;	Frame information byte cleared, with no initialization Valid data is 5 bytes long MID Data0 Data1 Data2 Chksum
3. Write data packet into UTXB module transmit buffer, with total length of 2 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10	
SpiWriteLongBuff(UTXB , 7 , (unsigned char *) &UTxPacket);	Write 7 bytes(Flen+2=7) to UTXB, if bus released, immediately start the initialization process.

4. Receive data
If the module receives data frame, INTF<URX> will be set ON.
User applies abortion or inquires flag bits, then read URXB to URxPacket receive buffer for processing.

8.5 HONDA(USERMOD) K-line Special Protocol

USERMOD(FMOD=3) implements the setting of HONDA special K-line diagnostic protocol function. (Non-universal OBD2 diagnosis, in K-line the standard OBD2 diagnosis HONDA generally uses SAEJ1979 application in ISO9141-2).
Like ISO9141-2/KW1281/J1708, this application is also one frame format in user-defined mode, theoretically ET7190 EUART module can implement all kinds of special frame formats. Different protocols are implemented by understanding the configuration of EUART control register.

8.5.1 HONDA Message Format

SYSTEM Identification	Len	Service ID	Data Characters	Checksum
SYS	LEN	SID	Data	CS
1 byte	1 byte	1 byte	n byte	1 byte

SYS：SYSTEM identification, this SYSTEM identification is a fixed value for communication with one module.

e.g.: Common value in request:
Engine：0x20/0x25 Safety Airbag：0x60 ABS：0xA0
Gearbox：0x40/0x45 ESP：0xC0

SYS is always 0x00 when ECU responds

LEN：Length byte, total length of one-frame data(including SYS/LEN/SID/DATA/CS all bytes), in 4~255,
SID：Command Service Identification
Data: Data, variable length, data byte can be none according to SID.
CS：Checksum, the same with J1708 checksum algorithm. In ET7190 configuration, UFLENCON<CSM> bit is set ON.

8.5.2 Enter System Initialization

Because there is no formal standard text released by HONDA, the analysis of various ECUs is done by the process below:

1. When some ECU enters the fast initialization, but the pulse time is not the common 25MS time, low level time is 70ms in initialization, high level time is around 160ms, but these two times may not be so precise in actual testing, it can be understood that tester only gives a low level wake-up signal to K line when the first command to enter system is transmitted.

   In user setting, set INITLT=70 INITHT=160, process the bus fast initialization when the first command is send.

2. When some ECU diagnoses, there is no initialization pulse signal, directly transmit the command to enter system.

3. In communication, the bus BaudRate is 9600 or 10400. It may vary because of different ex-factory years or different modules.
4. Bus Timing:
   Interbyte space in tester request, 1ms
   ECU respond time, 0-50ms
   ECU respond interbyte space, 0-2ms

8.5.3 Configuration Register/Register Set

8.5.3.1 Configuration Bus Initialization Register

User may select fast initialization or not when the first request message is sent:

INITLT=70	Low-level time in fast initialization 70ms
INITHT=160	High-level time in fast initialization 160ms

8.5.3.2 Configuration Message Related Time Register

UTXP1T= 1	Interbyte space 1ms when tester sends data
UFREET=55	Bus release time 55ms, value may be reduced to accelerate the request
URXP1T=10	Maximum interbyte byte 10ms in reception, (it must be greater than the interbyte space in ECU respond)

8.5.3.3 Configuration Length Control and Check Control(This is the key to Protocol Configuration)

According to the message format configuration length and checksum algorithm:

UFRMV = 0x04	ENCS=1 Enable checksum OFFSET=0, Length byte LEN is the actual total length of frame, subject to no change. VEND=0, Irrelevant VFINV=0, Irrelevant FTVEN=0, In transmission, disable each byte check FRVEN=0, In reception, disable each byte check
UFLENCON= 0x89	EN=1, Enable frame length, CSM= 1 (The same with J1708 check and algorithm) LENP= 1 , In-frame Byte 1 is the length byte LRR=0, No right shift in calculating the length
UFLAND = 0xFF	In calculating length, length byte(LEN) doesn't mask any bit The calculation of actual length after configuration is: (LEN>>LRR) & UFLAND + OFFSET Result value is LEN

8.5.3.4 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x00	Select UTX1/URX1 pin
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps / 9600bps
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00 USTAT1=0x00	State bit reset

8.5.3.5 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UPRICON=0x00	Disable J1708 priority
UFRMCON=0x83	FMOD=3 Select USERMOD protocol ON=1 Start EUART module

After EUART is started, the bus message transmission is started. The module will receive data if there is any data in the bus.

8.5.4 Transmit/Receive Message

Message transmission and reception are actually operated by the same method when FMOD is in any mode other than 0, and message is transmitted when data is written in UTXB transmit buffer. When the module receives message, URXB receive buffer data is read.
For example, SRS module is entered in the following process:
In entering system, one 70ms low level is firstly sent, then one 160ms high level is sent; then the command to enter sytem is sent; the command to enter engine is sent:
Request: 0xa0,0x05,0x70,0x06,0xe5
Respond: 00 09 02 03 01 02 03 00 EC

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.5.3 and start EUART	At that time, user doesn't send bus transmit data, just prepare the module

2. Write UTXB to directly transmit message	Access to the diagnosis requiring no initialization
UTxPacket .FrameInfo=0x0000; UTxPacket .INIT = 0b001; UTxPacket .Flen = 5 ; UTxPacket .Data[0] = 0xA0; UTxPacket .Data[1] = 0x05 UTxPacket .Data[2] = 0x70; UTxPacket .Data[3] = 0x06; UTxPacket .Data[4] = 0xE5;	Frame information byte cleared, with no initialization Before the transmission of this message, process the fast initialization. Access to the diagnosis requiring no initialization, INIT=0 Valid data is 5 bytes long SYS LEN SID Data Chksum
3. Write data packet into UTXB module transmit buffer, with total length of 2 bytes, run CMD_WR_LB byte command to write in long buffer, see Chapter 10	
SpiWriteLongBuff(UTXB , 7 , (unsigned char *) &UTxPacket);	Write 7 bytes(Flen+2=7) to UTXB, if bus released, immediately start the initialization process.
4. Receive data If the module receives data frame, INTF<URX> will be set ON. User applies abortion or inquires flag bits, then read URXB to URxPacket receive buffer for processing.	

8.6 LIN1.x / LIN2.x Application

8.6.1 About LIN

LIN(Local Interconnect Network) is a serial communication network at low cost, which is used for the distributed electronic system control of vehicle. LIN aims to provide the auxiliary functions for existing vehicle network(e.g.: CAN bus), so LIN bus is a auxiliary bus network. In an occasion with no need of CAN bus bandwidth and multifunctionality, e.g.: the communication between intelligent sensor and braking apparatus , and LIN bus can be used to slash the cost.
LIN technical standard also defines the development tool and application software interface in addition to basic protocol and application layer. LIN communication is based on UART data format, which applies the mode of single master controller/multiple slavers. Only one 12V signal bus and one unfixed time-based node sync clock line are used.
This serial communication mode at low cost and its development environment has been formulated into standard by LIN Association. LIN standardization will cut the cost of developing and applying OS for vehicle manufacturer and vendor.

LIN Milestone
1998.10, the tentative idea of LIN bus was firstly raised during the Vehicle Electronic Conference held in German Baden Baden
1999, LIN Alliance was established (founders include Audi, BWM, Chrysler, Motorola, BOSCH, Volkswagen and Volve)
2000, LIN Alliance began to accept the members
2001, the first vehicle using LIN bus was launched
2002, LIN Standard V.1.3 was released
2003, LIN standard V.2.0 was released
2004, LIN bus compliance test standard was released
2006, LIN Standard V.2.1(Current Edition) was released
2010, LIN Standard Package Specification Package Revision 2.2A(Latest Edition) was released

LIN Alliance
LIN Alliance was initially founded by Audi, BWM, Chrysler, Motorola, BOSCH, Volkswagen and Volve or other automakers and chip manufacturers, in an attempt to drive the development of LIN bus. Further, it is the body to release and manage LIN bus standard and formulate the compliance test standard and certification compliance test. Currently, this alliance is endeavored to promote LIN bus as ISO standard.

LIN Bus Features
Low cost: Based on the universal UART interface, almost all microcontrollers are equipped with LIN required hardware;
Few signal line is used to implement ISO9141 standard;
Communication rate is up to 20Kbit/s;
Sing master /multiple slavers mode requires no arbitration mechanism;
Slave node requires no crystal or ceramic oscillator to implement the self-synchronization, reducing the hardware cost of slaver;
Ensure the time delay of signal transport;
Add nodes to the network with no need of changing the hardware and software at LIN slave node;
Generally the number of nodes in LIN network is less than 12, with 64 identifiers in total;

LIN Communication Rule

A LIN network consists of one master node and one or more slave nodes, and all nodes carry a slave communication task. This communication task is divided into transmit task and receive task, and the master node also carries a master transmit task.

The communication in a LIN network is always launched from the master transmit task. Master controller transmits a start-of-message, which consists of sync break and sync byte information identifier. Accordingly, after the information identifier is accepted and filtered, one slave task is activated and the respond transmission of this message is started. This respond consists of 2/4/8 data bytes and a check code. Start-of-message and respond part constitutes a complete message frame.

How to correctly form LIN message frame? This message is formed as indicated by message identifier. This communication rule can be employed to exchange data via various methods: From master node to one or more slave nodes; from one slave node to master node or other slave nodes, communication signal can communicate beween the ndoes without going through the master node or all nodes from master node radio message to the network.

Timing of message frame is controlled by master

LIN Bus Application

Typical LIN bus application is the union assembly units in vehicle, e.g.: door, steering wheel, seat, air conditioner, light, humidity sensor, AC generator, etc. For these cost-sensitive units , LIN has those mechanical elements extensively applied, such as intelligent sensor, braker or photosensitive device. These elements can be easily connected to the vehicle network, which can be conveniently maintained and served. In LIN-enabled system, analog signal quantity is usually replaced by digital signal, which will optimize the bus performance. In the following vehicle electronic control system, perfect effects will be brought if LIN is implemented:

While LIN is initially designed for the vehicle electronic control system, LIN is also extensively applied to the bus of industrial automated sensor or consumer electronic products.

8.6.2 LIN Message Format

Header			Space	Data	Checksum
BREAK	SYNCH	ID	IFT	Data	CS
SYNCH BREAK	0x55	1byte		2/4/8 Bytes	1byte

Header: Header consist of 3 components, **BREAK**(sync break character), **SYNCH**(sync character) 0x55, and LIN-defined ID identifier. Header must be initiated by master, and slaver accepts or responds according to the header transmitted from master node. In network, header is not transmitted from slave node.

BREAK：Sync space character.

If the duration of dominant level is longer than the common dominant bit sequence defined in the protocol(0x00 field has 9 dominant bits), then this is a SYNCH BREAK FIELD. ET7190 transmission contains 1 start bit when the master starts task, followed by 12 “0 bit” and 1 stop bit(13 dominant bits in total).

SYNCH：Sync character, the value is always 0x55. Slave node permits the auto BaudRate detect according to sync character.

ID：LIN identifier, define the content and length of message, see the definition of ID Identifier Field.

IFT： In-frame response space time, a time delay exists between frame header and data field, this time is a must normally, and tends to be very important, because some times are required for the preparation of data transmission or reception after the node receives the frame header.

When ET7190 is used as the master node:

If one master task is transmitted(master node transmit header and data field), this time is defined by UTXP1T register. After the master transmits the header, the data field and check byte are transmitted in delay of UTXP1T.

If one slave task is transmitted(master only transmits the frame header, and the data field and check byte are responded from slaver), wait for the slaver's response after the master node transmits the frame header. The time that slaver starts to respond must be less than URXP1T time.

When ET7190 is used as the slaver node:

If the slaver task required for response is received(master node transmitting header is the respond task required at this node), if ET7190 is set as auto respond(respond data is one of LACKBn), the transmission of data field and check byte are started after ET7190 module delays UTXP1T time.

If non-auto respond is defined, ET7190 will transport header information(actually only the ID of an identifier) to URXB receive buffer after the frame header from master startup is received, user reads URXB and writes respond data into UTXB transmit buffer according to ID value, directly starting the response. At this time, IFT time totally depends on the time that user writes in UTXB. User needs to ensure, total time from the reception of ID to the writing of UTXB(this time is actually IFT respond break), must be less than URXP1T time defined by master node. Otherwise receiver generates the frame respond error. In this case, user can precisely control the responded IFT time in real-time system, but user is unable to precseily delay the time in WINDOWS non-real-time OS. It is actually an inappropriate application to use WINDOWS as slave node for response, unless other relevant nodes in the network permits the long enough IFT time setting(long frame slot time will affect the efficiency of bus transmission).

Data： Data field byte, as defined by ID. Its length can be set as 2 bytes, 4 bytes or 8 bytes. As determined by LENM bit in LIDCON Control Table.

CS： Checksum, LIN2. time checksum contains ID byte, LIN1.x Check Mode, checksum contains no ID byte, Check Mode, as determined by CSM bit in LIDCON Control Table.

8.6.3 Message Frame Type

LIN has 6 types of message frame:

- Unconditional Frame: Must carry the data information

- Event Triggered Frame:

  Enhance LIN bus responsiveness, avoid polling on nodes for rare event, thereby wasting a lot of bandwidth identifiers: 0~59(0x3B) Event Triggered Frame must have an independent ID, which can be associated with multiple common frames and transmit frame header within the time slot of Event Triggered Frame, only there is signal updated in the associated unconditional frame before the first data byte equaling to identifier of frame respond is transmitted, namely respond can transport the data with up to 7 bytes. If there is no frame respond, header will be neglected and frame respond can be transmitted via multiple nodes, in caes of any conflict, switched to "Schedule Table of Conflict Resolving", before reswitched to the previous Schedule Table.

- Sporadic Frame: Represent to share one time slot, a set of unconditional frames only transmitted if required.

Purpose: Add the dynamic behavior and real-time into Schedule Table, without affecting the rest of deterministic identifiers in Schedule Table: The response of 0~59(0x3B) Sporadic Frame is only transmitted by master ndoe, if unconditional frame is required to send, then the transmission sequence is determined by the frame priority, if no unconditional frame is required to send, then the time slot must keep blank.

- Diagnostic Frame(Command Frame, Respond Frame)
  Diagnostic information or configuration information carrying 8 bytes, master node diagnostic request identifier is 0x3C(60). Slave node respond frame identifier is 0x3D(61).

- User-defined Frame: The identifier is 0x3E(62)
  Any user-defined information can be carried, with no limit of data length.
- Hold Frame: The identifier of Hold Frame is 0x3F(63). Not used in LIN2.x LIN1.x.

8.6.4 ID Identifier

ID identifier defines the function of node and length of message in the network, ID identifier function definition of the used node is the key to LIN network normal operation, when ID identifier function is defind, ID conflict of any two nodes is not allowed. ET7190 defines ID function by setting LIDCON control register set.

8.6.4.1 ID Identifier Bit Definition

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P1	P0	ID5	ID4	ID3	ID2	ID1	ID0

ID<5:0>： ID identifier has 6 bits, which is valued in 0-63, the identifier can be classified into 4 types:
The value of carrier wave frame, which is valued in 0-59(0x3B).
60(0x3C) and 61(0x3D) Used for the diagnosis of data
62(0x3E) Specifically used for the user-defined extension.
63(0x3F) Specifically used for the late revision of protocol.

P0： Odd Parity Check Bit

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$

P1： Even Parity Check Bit

$$P1 = \overline{ID1 \oplus ID3 \oplus ID4 \oplus ID5}$$

8.6.4.2 Data Frame Length Definition

ID5	ID4	NDATA(Data Length), Byte
0	0	2
0	1	2
1	0	4
1	1	8

- In the network, the defiend data length of message must be in line with the defined frame length as shown above.(ET7190 permits user to define the length in any way). In special application, the length of data byte can be any byte in no relevancy to identifier.
- Identifier has the same ID bit ID0 ..ID3, but with different length codes, ID4 ID5 can indicate the message with different functions.

8.6.5 Configuration Control Register/Set

8.6.5.1 Configuration Message Related Time Register

UTXP1T=5	IFT time 5MS, this parameter is set according to the handling speed of user module, less time can increase the utilization of bus. UTXP1T, URXP1T and UFREET determine the total LIN frame time slot.
UFREET=0	Bus release time is generally set as 0
URXP1T=8	This parameter is set according to the user module, less time can increase the utilization of bus. If in WINDOWS non-real-time OS, when used as slaver, this time must be set around 100MS, if in other real-time system, it can be set according to user handing speed.
UPRICON=0	J1708 priority control register must be disabled

8.6.5.2 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x00	Select UTX1/URX1pin
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps/9600bps，LIN Up to19200bps
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00	State bit reset
USTAT1=0x00	

8.6.5.3 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UFRMCON=0x82	FMOD=2 Select LIN protocol ON=1 Start EUART module Note: It is recommended to clear UFRMCON<LAB> bit, because ET7190 has the precise external clock. Auto BaudRate detection is not required.

8.6.6 Configuration LIDCON Control Table

Before the construction of LIN network system, user must wholly plan ID function, correctly configure LIDCON control table according to the required ID function, LIN master controls the system process via ID.

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
IDEN	TRM	LENM		CSM	ARE	ARP	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

8.6.6.1 LIDCON Control Table Setting Rule

- ID irrelevant to the module or unused ID, sets the corresponding LIDCONn<IDEN> bit as Clear, when IDEN is 0, the set value of Bit6..0 is irrelevant, generally all cleared, master is normally not allowed to launch IDEN=0 set as invalid ID task.

- TRM Bit Setting:

  Master:
  All as IDs of master task, TRM is set Clear. All as IDs of slaver task, TRM bit is set ON.

  Slaver:
  This slaver responded ID TRM bit is required to be set ON.

  ID TRM bit requiring no response is set Clear, in this case, slaver receives master task(data field is sent from master) or receive slaver task(data field is sent from another slaver).

- LENM Length Code Setting, normally set as per LIN standard protocol.
  ID：0x00-0x1F LENM=1 (ID5=0; ID4=0 or 1, data length is 2 bytes)
  ID：0x20-0x2F LENM=2(ID5=1; ID4=0, data length is 4 bytes)
  ID：0x30-0x3D LENM=3 (ID5=1; ID4=1, data length is 8 bytes)

ID： =62(0x3E) or 63(0x3F), LENM=0(This is special ID, data length can be random value)

Note: For the special application, length may not be set by standard, but LENM of IDs of all nodes in the network must be set identical.

- CSM Setting, LIN2.x allows for the simultaneous use of LIN1.x protocol in the same network. In this case, multiple ECUs are normally installed in the same vehicle due to different production years.

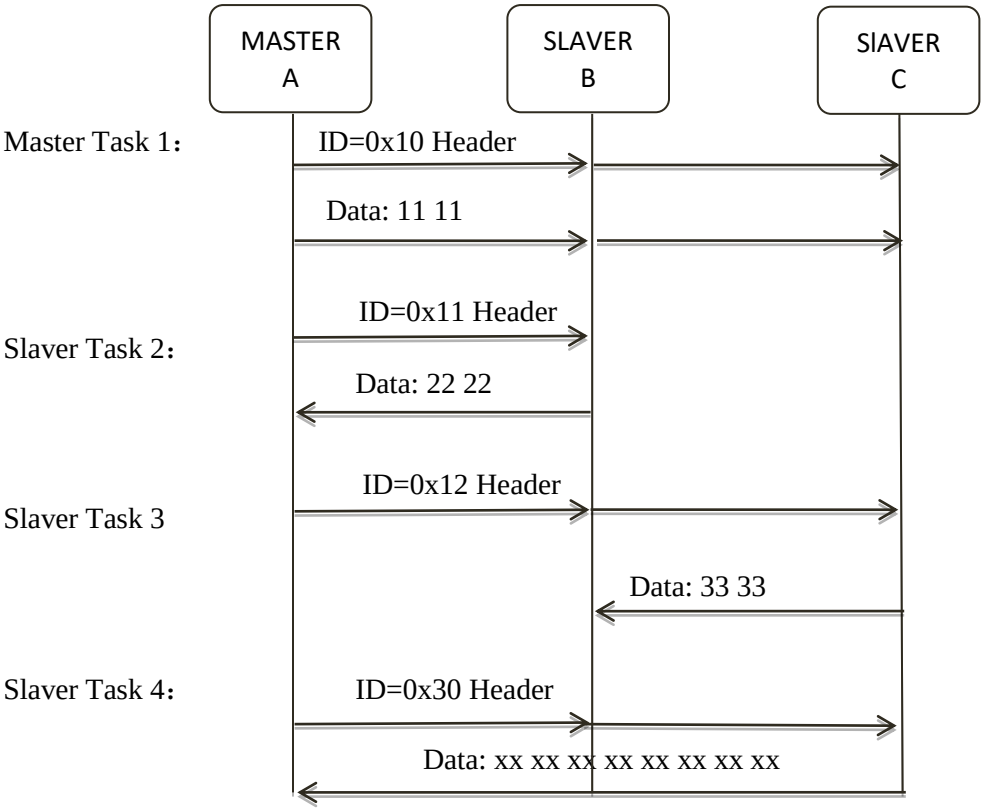
  For ID used as per LIN1.x protocol, CSM=0.
  For ID used as per LIN2.x protocol, CSM=1.

- AFE Setting, this bit is only relevant to slaver, and when TRM=1(ID requiring slaver response), only valid.
  AFE=1, when the module receives ID slaver task launched by master, slaver module directly uses the data response in LACKBn buffer(n=AFP).
  AFE=0, when the module receives ID slaver task launched by master, slaver module transports the current ID identifier to URXB receive buffer, user transmits the respond data according to ID(user needs to write data in UTXB transmit buffer)

- AFP Setting, select buffer, LACKB0, LACKB1, LACKB2, or LACKB3 when slaver automatically responds.
  There are 4 buffers, user just firstly set the data of buffer to be used, when slaver task ID requiring response is received, the module can automatically transmits the response. User is not required to make other operation.

8.6.6.2 How to Configure ID Control Table

e.g.: Application on all kinds of vehicle: Switching signal, sensor data, and action execution are indicated in the data transmission in LIN network, so we just illustrate an example by data transmission:



This example uses 4 IDs, ID function of each ID is described below:

- ID=0x10：Master task, maste sends data, Slaver B and Slaver C simultaneously receive, data length is 2 bytes.
- ID=0x11：Slaver task, Slaver B responds, master receives data from Slaver B. Slaver C neglects this task.
- ID=0x12：Slaver task, Slaver B, C simultaneously receives, C responds. This task is compared to transporting data from C to B and A.
- ID=0x30：Slaver task, Slaver B, C can simultaneously respond, master receives. Data length is 8 bytes. Bus error will be generated in case of simultaneous response. Or B, C might not respond, current frame will be neglected.
  This case is generally used for the transmission of Event Triggered Frame. B, C response depends on the state of module. In Schedule Table, this ID usually associates multiple unconditional frames.
- All headers are launched by master.

The configuration of LIDCON table at Master Node A can be described in C language as follows:

Int i;
For (i=0;i<64;i++)
{
  LIDCON[i]=0x08;
}
	LIDCON Control Table all cleared, assume that we only use LIN2.x protocol, CSM(Bit3) is set ON.

LIDCON[0x10].IDEN=1; LIDCON[0x11].IDEN=1; LIDCON[0x12].IDEN=1; LIDCON[0x30].IDEN=1;	Enable ID Identifier：0x10,0x11,0x12,0x30，
LIDCON[0x10].LENM=1; LIDCON[0x11].LENM=1; LIDCON[0x12].LENM=1; LIDCON[0x30].LENM=3;	Set length 0x10 ID Identifier frame data length is 2 bytes The same as above The same as above 0x30 ID Identifier frame data length is 8 bytes
LIDCON[0x11].TRM=1; LIDCON[0x12].TRM=1; LIDCON[0x30].TRM=1;	0x11, 0x12, 0x30 are slaver task TRM set ON 0x10 is master task, TRM=0; cleared as soon as it begins
Other ID is irrelevant, so LIDCON[id] all cleared	

The configuration of LIDCON table at Master Node B can be described in C language as follows:

Int i;
For (i=0;i<64;i++)
{
  LIDCON[i]=0x08;
}
	LIDCON Control Table all cleared, assume that we only use LIN2.x protocol, CSM(Bit3) is set ON.

LIDCON[0x10].IDEN=1; LIDCON[0x11].IDEN=1; LIDCON[0x12].IDEN=1; LIDCON[0x30].IDEN=1;	Enable ID identifier: 0x10, 0x11, 0x12, 0x30, other IDs are not used, so kept Clear. Node B also uses 4 IDs
LIDCON[0x10].LENM=1; LIDCON[0x11].LENM=1; LIDCON[0x12].LENM=1; LIDCON[0x30].LENM=3;	Set length(Must be the same with Master A) 0x10 ID Identifier frame data length is 2 bytes The same as above The same as above 0x30 ID Identifier frame data length is 8 bytes
LIDCON[0x10].TRM=0; LIDCON[0x11].TRM=1; LIDCON[0x12].TRM=0; LIDCON[0x30].TRM=1;	0x11, Node B needs to respond(Transport 0x22 0x22 to master) 0x12, Node B needn't to respond, respond data is transported from Node C. The module needs to respond, user determines whether to respond or not according to the state(Event Triggered), if user receives ID=0x30, no data is

	required to transmit, ET7190 module is back to idle state after a delay of URXPT1, until the next break field appears
Other ID is irrelevant, so LIDCON[id] all cleared	

The configuration of LIDCON table at Master Node C can be described in C language as follows:

Int i;
For (i=0;i<64;i++)
{
  LIDCON[i]=0x08;
}
	LIDCON Control Table all cleared, assume that we only use LIN2.x protocol, CSM(Bit3) is set ON.

LIDCON[0x10].IDEN=1; LIDCON[0x11].IDEN=0; LIDCON[0x12].IDEN=1; LIDCON[0x30].IDEN=1;	Enable ID identifier: 0x10, 0x12, 0x30, 0x11 is irrelevant to this module, so set ON Set length(Must be the same with Master A)
LIDCON[0x10].LENM=1; LIDCON[0x11].LENM=1; LIDCON[0x12].LENM=1; LIDCON[0x30].LENM=3;	0x10 ID Identifier frame data length is 2 bytes The same as above, 0x11 is irrelevant The same as above 0x30 ID Identifier frame data length is 8 bytes
LIDCON[0x11].TRM=0; LIDCON[0x12].TRM=1;	0x11, because IDEN=0, so irrelevant 0x12, respond data is sent from Node C.
LIDCON[0x30].TRM=1;	The module needs to respond, user determines whether to respond or not according to the state(Event Triggered), if user receives ID=0x30, no data is required to transmit, ET7190 module is back to idle state after a delay of URXPT1, until the next break field appears
Other ID is irrelevant, so LIDCON[id] all cleared	

8.6.7 Transmit and Receive as Master Node

When ET7190 configuration is started, slaver and master are not differentiated except the setting of LIDCON Configuration Table.

And each ET7190 node doesn't physically limit the active transmission of one message header. User must ensure no message transmitted if used as slaver node, slaver is only in passive respond or receive state, slaver node will not actively start a message transmission as per LIN protocol.

8.6.7.1 Configuration Start

1. Configure LIDCON as Node A as per 8.6.6
2. EUART controller is started as per 8.6.5

3. Configure another two nodes as per 8.6.6, Module B and Module C(Suppose we have connected three modules, A/B/C)

8.6.7.2 Node A: Transmission of Master Task(Header and Data Field Transmitted from Node A)

User's transmission can be simply compared to writing one-frame data in UTXB: (ID=0x10)

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.6.7.1 and start EUART	
2. Write UTXB to directly transmit message	Transmit master task header LIDCON[id].TRM=1 must be cleared

UTxPacket .FrameInfo=0x0000;
UTxPacket .Flen = 3 ;
 	
UTxPacket .Data[0] = 0x10; //ID
UTxPacket .Data[1] = 0x11
UTxPacket .Data[2] = 0x11;
	Frame information byte cleared, must be 0
	Valid data is 3 bytes long, no CS is contained in transmission, CS is automatically calculated by ET7190, CS calculation method is determined by LIDCON[id]>CSM Confirm if ID byte is contained.
	ID just requires BIT5:0, BIT7, BIT6 Automatically set by ET7190
	DATA0
	DATA1

- After the transmission is completed, interrupt flag INTF<UTX> is set ON. If interrupt is allowed, transmission completion interrupt will be generated.
- Flen length must be equal to ID defined data frame length+1. If in any length(LENM=0), then FLEN is actual data length+1;
- If the set Flen length is not enough, also transmitted in LENM-defined length when transmitted, UTXB data must be completely written.

8.6.7.3 Node A: Reception of Slaver Task

Master A requires to receive data from slaver, frame header must be sent from master(in this example, ID=0x11/0x12/0x30 are slaver taskes), wait for slaver response.

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Configure register parameters according to 8.6.7.1 and start EUART	
2. Write UTXB to directly transmit message	Transmit slaver task header LIDCON[id].TRM=1 must be cleared
UTxPacket .FrameInfo=0x0000; UTxPacket .Flen = 1 ; UTxPacket .Data[0] = 0x11; //0x12 or 0x13	Frame information byte cleared, must be 0 Valid data is 1 byte long, because LIDCON[0x11].TRM=1, set the current ID as salver task, actually only one ID data of the written longer data is valid. Only transmit frame ID is transmitted when ET7190 sends

3. Receive data	After master sends header, slaver begins to respond the data field. If the data length of slaver respond exceeds LENM set length, the excessive part will be discarded.
Read URXB data in URxPacket	At that time, the data length of received data buffer is the data field length+2. (The length of receive buffer contains ID+Data+CS)

This process of data reception may be along with the possible cases(Depending on ID definition)

- If the slaver responded length exceeds the master LENM defined length, and data in excess of such length will be discarded. (The module will not generate error interrupt, but the receive buffer CSERR will indicate error)

- In the received slaver task data buffer, length contains ID, data field, checksum. (No CS is contained when master task is transmitted)

- ID=0x11 case is an unconditional frame, Slaver B must precisely respond to data field, if Node B doesn' t respond, Master A will generate a NORP error, Node C will not receive this ID, because Node C IDEN=0, which is an invalid ID for Node C.

- ID=0x12 case is also an unconditional frame, Slaver C must precisely respond to data field, if Node C doesn't respond, Master A, Slaver B will generate a NORP error. When Node A and Node B simultaneously receive data frame, Node A can identify whether Node B correctly receives Node C's data by telling if data frame is correct, determining whether to start or cancel task.

- ID=0x13 caes is an Event Triggered Frame

  Slaver B/C may respond to data field, if Node B/C doesn't respond, Master A will generate a NORP error, when Node A's master user can process as non-event, Node B and Node C automatically return to idle state after URXP1T time, because user doesn't transmit respond data to the module.

  If only B or C responds, Node A will normally receive slaver task. In which, unresponded slaver node will automatically return to idle state after URXP1T time, and unresponded node will not receive any data frame at the same time.

  If B/C simultaneously respond, Node A/B/C will generate frame bit error or CSERR error(conflict), when master node user must start the relevant unconditional frame for data transmission.

  Slaver Node B/C will receive one ID slaver task request when master starts one ID=0x13 task, and user may give event respond according to the state of Node B/C.

8.6.7.4 Node A: Received Interframe Gap

If any other slaver node starts one master task in the network, if ID configuration enables, this Node A will be automatically processed as one slaver node. Actually this is illegal, LIN network permits no more than 2 masters, because ET7190 node doesn't physically differentiate master and slaver. When any node is in idle state, and SACK=0 data is written to UTXB, one task frame will be started.

8.6.8 Slaver Node Response and Reception

8.6.8.1 ET7190 Module Auto Respond

After Slaver Node B or C receives the slaver task request ID launched from Master A, if response is required, when the received ID LIDCON[id].TRM=1 and ARE=1, ET7190 module will automatically employ the data in auto respond buffer LACKBn to respond. ARP will select which buffer, LACKB0/1/2/3, to be used.

In this case, the module will not transmit the slaver task ID request to URXB. Not occupy user CPU time.

This function is too convenient for responding the data with no frequent change at node.

e.g.: As previously described, **ID=0x11 Node B slaver respond setting:**

1. Responded Initial data is written in LACKB0, e.g.: 0x22 0x 33 …, valid length is ID LENM set length. Up to 8 bytes. In auto respond, ID LENM may not be set Clear.
2. LIDCON[0x11].ARP=1; Select LACKB1 as ID=0x30 respond buffer
   LIDCON[0x11].ARE=1; Enable auto respond
   LIDCON[0x11].TRM=1; This node requires the responded slaver task
   LIDCON[0x11].IDEN=1; Enable ID
3. Node B can change data in LACKB1 at any time.
4. When Node B receives ID=0x11 slaver task request, Node B can automatically respond by data in LACKB1.

8.6.8.2 User Respond Data Transmitted by User

After Slaver Node B or C receives the slaver task request ID launched from Slaver A, if response is required, when the received ID LIDCON[id].TRM=1 and ARE=0, ET7190 module will transport ID request to URXB receive buffer. And INTF<URX> is set receive interrupt flag.

1. When SACK flag bit in the receive buffer is 1, indicating that this is one slaver task ID request, valid data is just an ID value(0-63), user writes respond data in UTXB according to the receive data.

2. The valid length of data(Flen) written in UTXB must equal to ID LENM defined length, no checksum is contained when data is written in UTXB. Checksum(CS) value is automatically informed by the module. SACK of data frame information word written in the transmit buffer UTXB must be 1, indicating that this is one LIN respond field data.

Note: Slaver node must ensure that SACK is always 1 when data is written in UTXB. If the information word SACK=0, if when the slaver is in respond state, this data will be discarded. If in idle state, when SACK=0 data is written, slaver will automatically changre to master, starting one task. This might lead to the network chaos.

3. After ID request is received, the time that data is written in UTXB must not exceed the receiver's URXP1T time. If exceeded, receiver will set NORP(no in-frame respond field) with error flag.

EUART_REPORT UTxPacket; EUART_REPORT URxPacket;	Define transmit buffer Define receive buffer
1. Slave node receives ID=0x11 request At this time, URxPacket Flen=1, SACK=1 Data[0] data is ID value	
2. Write UTXB for slaver respond	
UTxPacket .FrameInfo=0x0000; UTxPacket .Flen = 2 ; UTxPacket .SACK = 1 ;	Frame information byte cleared, must be 0 Valid data is 2 bytes long, SACK=1.Represent one respond field data
UTxPacket .Data[0] = 0x22; UTxPacket .Data[1] = 0x22	CS value is not required to write in, the module automatically fills in
3. Receive data	
SpiWriteLongBuff(UTXB , 4 , (unsigned char *) &UTxPacket);	Write 4 bytes(Flen+2=4) in UTXB, Node B will directly transport 2 bytes respond data(0x22, 0x22) and CS to bus.

8.6.8.3 Slaver Only Receives Data Frame

As illustrated, Node B enable Master Task 1(ID=0x10) and Slaver Task 3(ID=0x12), but this is because that the corresponding ID TRMs are 0, so data is only received. Frame respond will be transmitted. User applies the ID in the received data frame.

When SACK in the information word in data of receive buffer must be 0(this represents one-frame complete LIN data at that time), CSERR represents the checksum result, Flen length must be valid data length+2. (Including ID and CS)

8.7 SCI(J2610) Application

J2610 is a serial communication standard, data is transmitted via TX line at the port, data is received via RX line, the space between the modules is grounded, normally bus voltage is 5V, 20V(MAX:100MA) programming voltage is added to the transmit(TX bus) bus by tester in programming. (Use By DaumlerChrysler)

ET7190 can be connected as per 3.2.1 K-line Receive Diagram, when configured, UTX1 is set as the output pin of EUART, URX2 as the input pin of EUART, consequently K1 line is used as transmit line and K2 as receive bus. If programming function is required, programming voltage can be directly added to K1 line by adding the external triode circuit and controlling via P4:0 Port.

In K-line Receive Diagram under 3.2.1, communication voltage is 12V, which is also common in SCI actual application on vehicle. If 5V communication voltage is used, just change SI9241 Ⅴbat voltage as 5V.

8.7.1 Message Format and Length

SCI message format has no strict definition, generally subject to user-defined application. For data length, J2610 defines each frame no longer than 4096 bytes, ET7190 allows the transmitted or received frame length no greater than 256 bytes in FMOD=100/101 Mode, if 256-byte frame is required to be processed, user need to select FMOD=000 Mode(universal UART mode).

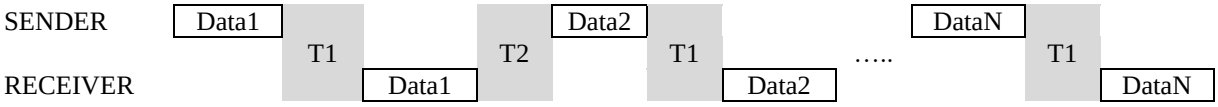
Byte No.	Data Value	Description
1	Data 1	Start of Message Data
2	Data 2	
.	.	
.	.	
.	.	
N	Data N	End of Message Data

8.7.2 SCI Message Transmission Mode

Frame can be transmitted by the following two methods in 8.7.2.1 and 8.7.2.2 when ET7190 transmits data: These two methods of transmission, reception and transmission of module can be separately set. And dynamic switch can be done in communication process.
For the specific application, user may enable frame length control.
Check and CS enable are only valid when FMOD=100.

8.7.2.1 Each Byte Requires Receiver to Delivery Back Check Byte Frame Mode(Only Available in FMOD=100 Half-duplex Mode)

If the enable last byte requires no check, then enable frame length must be controlled and correct length byte must be set, or operation anormaly may appear because the receiver doesn't know which byte is the last byte.



- T1 is the time of receiver delivering back the check byte, as set by URXVT1.
- T2 is the time delay of transmitting the next data after transmitter receives the check byte, as set by UTXP1T.
- The check byte value back-delivered by receiver may be the original value or reversed value, as set by UFRMV<VFINV> bit.

8.7.2.2 Continuous Direct Transmit Frame Mode

- Interbyte gap in T1 transmission, as set by UTXP1T.

8.7.3 Configuration of Half-duplex Mode

Half-duplex communication method is usually applied in vehicle. (When one module transmits data, the other module cannot transmit data)

8.7.3.1 Configuration Message Related Time Register

UTXP1T=1	Minimum interbyte gap when tester sends data 1ms At high speed, can be set as 0
UFREET=100	Bus release time 100ms, can be set as 1 at high speed
URXP1T=20	Allowed maximum interbyte gap in reception 20ms, can be set as 5 at high speed
UTXVT0=20	In transmission, allowed maximum time of tester receiving check time returned from ECU 20ms. Can be set as 5 at high speed
URXVT1=1	When ECU transmitted frame is received, time delay of transmitting check byte 1ms

8.7.3.2 Configuration EUART Bit Definition and BaudRate

UPINSEL=0x40	Select UTX1/URX2 pin (RXSEL=1，TXSEL=0), receive/transmit line are separate
UBRG=239 (0x00EF)	BaudRate scaler, 10400bps
UMODE=0x00	BRGH=0 Low-speed BaudRate PDSEL=0 8-bit data, no odd/even parity check STSEL=0 1 stop bit
USTAT0=0x00 USTAT1=0x00	State bit reset

8.7.3.3 Configuration Length Control and Check Control

UFRMV = 0x00 Note: Defined as required, after EUART is started, and this register value is changed, bus must be kept in idle state	ENCS=0, Disable checksum, disable length VEND=1, Last byte check in frame VFINV=0, In check, use the original value FTVEN=0, In transmission, 1 time enable each byte check, FRVEN=0, In reception, 1 time enable each byte check,
UFLENCON= 0x00 Set as required	EN=0, Enable frame length, CSM=0 (Checksum Mode) LENP=0, In-frame Byte 0 is length byte LRR=0, No right shift in calculating the length
UFLAND = 0xFF	In calculating the length, length byte(ML) must not mask any bit Calcualtion of actual length after configuration is: (ML>>LRR) & UFLAND + OFFSET, not used here, irrelevant

8.7.3.4 Start EUART

UERRIE=0x ??	Handle EUART error interrupt if needed If query mode is used or interrupt is not handled, set UERRIE=0x00
INTE=0x90	ERRE=1 Enable global error interrupt, URXE=1 Enable UART receive completion interrupt If the query method is used, set INTE=0x00
UPRICON=0x00	Disable J1708 Priority
UFRMCON=0x84	FMOD=100 Select <SCI Half-duplex Mode> ON=1 Start EUART mode

8.7.4 Configuration of Full-duplex Mode

In FMOD=101 full-duplex mode, the module is enabled to transmit and receive data. But in this mode, enable UFRMV<FTVEN> or <FRVEN> is invalid. Each byte check is not allowed in transmission and reception of data frame.
Other configuration is identical to the half-duplex mode.

8.7.5 Data Reception and Transmission

8.7.5.1 Transmit Data Frame

In the transmission of data frame, user just needs to write the data frame to be transmitted in UTXB transmit buffer. Note that in SCI Mode, frame information INIT in transmit buffer is invalid. In this mode, initialization will not be started.
Maximum valid data length to be written is 256b bytes, and the excessive part will be discarded.

8.7.5.2 Receive Data Frame

When the data frame is received, only need to read data in URXB receive buffer.

8.8 Used as Universal UART Module

When UFRMCON<FMOD>=0b000, UART Character Mode – This mode can be compared to one common UART serial communication port, each has 4-byte FIFO transmit and receive buffers, user must process all data and time of transmit byte in the application.

1. When the module transmits character, EUART has 4-byte-deep FIFO transmit buffers, and one transmit offset register(UTSR register, user has no access).
   User loads data into UTRXREG register(FIFO) to start the data transmission. Before the previous transmission of data loaded stop bit, new data will not be loaded in UTSR register. Once the stop bit transmission is completed, new data(if any) in UTRXREG register will be loaded in UTSR. When data is loaded in UTSR, transmit interrupt flag INTF<UTX> will be set ON, if interrupt allows INTE<UTXE> as 1, hardware interrupt will be generated.

2. When the module receives byte, EUART has 4-byte-deep FIFO receive buffer, and one receive offset register(URSR register, user has no access). After data is received at URX pin and stop bit at UxRX pin is sampled, data received in URSR will be transported to the receive FIFO(If UTRXREG is empty).

3. If there is any data in FIFO receive buffer(UTRXREG), INTF<URX> is set ON, user clears INTF<CRX> receive interrupt by reading empty UTRXREG register.

4. When EUART operates in Character Mode, any operation on UTXB transmit buffer will be invalid.

5. Error Handling in Character Mode
   - If FIFO is already full(4 characters), and Character 5 has been fully received in URSR register, then overflow error bit USTAT0<OERR> and UERRF<OVF> will be set ON. URSR word will be held, but data transported to receive FIFO will be disabled as long as USTAT0<OERR> bit is set ON. User application program must clear USTAT0<OERR> bit via software, in order to enable the continuous reception of data.
   - If the received data before overflow is required to be saved, user application program must read all 5 characters and clear OERR bit. If these 5 characters can be discarded, then user application program just needs to clear USTAT0<OERR> bit.
   - If the stop bit is detected as logical low level, then frame error bit USTAT0<FERR> and UERRF<BITERR> will be set ON.
   - If the data word at the top of buffer is detected(namely, current character) has any odd/even parity check error, then odd/even parity check error bit USTAT0<PERR> will be set ON. For example, if odd/even parity check is set as even parity check, but total number of "1" in data is detected as odd, odd/even parity check error will be generated. FERR, PERR bit and the corresponding character are buffered and must be read prior to the reading of data word.
   - If any of these errors(OVF, BITERR and PERR) are generated, interrupt will be generated. User application program must enable the corresponding interrupt enable control register UERRIE until the corresponding bit is generated.

Reception, transmission in Character Mode are processed by reading and writing UTRXREG, OVF, BITERR and PERR of UERRF in error register are associated with USTAT0<OERR\PERR\FERR>, error flag of other bits in UERRF is kept as Clear in Character Mode.

8.9 EUART Simulation ECU Receive Initialization Information

Prior to the communication, SO9141-2/ISO14230(KWP2000)/KW1281 or other protocols must go through the specially defined initialization process(this process doesn't conform to the message standard format of protocol) before the communication setup between two modules(generally between tester and ECU). Common K-line initialization methods are 5 BaudRate slow initialization and fast initialization. ET7190 specifically sets this simulation function of ECU initialization information for the simulation of ECU K-line function, in order to enable user to implement the simulation function of vehicle ECU operation by ET7190.

8.9.1 Description of Function

- In CAN, J1850, LIN or other protocols, there is no initialization process in compliance with the protocol message format, so user needn't make special setting if user intends to use ET7190 to simulate ECU CAN/J1850/LIN function.
- Simulation of ECU K-line Function: When the reversed study is made on the diagnositic protocol, it has a considerable effect. If the full-functional tester is to be developed, it is necessary to simulate ECU. When user develops the tester, user has to make the protocol analysis via monitor tester and data in ECU communication because vehicle manufacturer treats all application protocols as confidential. However, all data or states are not possible to monitor when tested on vehicle. In this concern, we can simulate ECU operation by programming and send any possible data to the tester, which in turn makes easy for the reversed protocol analysis via tester.

- Normally, when the tester communicates with ECU, tester is active module, ECU is passive module. So ET7190 simulation of ECU can be described as the transmission of respond data when user receives the request data frame from tester.

- The initialization information of receiving K line is only valid when FMOD=001(ISO14230) or FMOD=011(USERMOD), SIMU bit must not be set and K line initialization simulation is started in any other FMOD value. Otherwise unexpected error will be generated.

- Startup Method: When FMOD=1/ 3, when EUART controller is started, UFRMCON<SIMU> is set ON simultaneously.

8.9.2 The Process of Simulation Setting:

1. Set SIMUCON control register: set the trigger condition of initialization, only when the bus appears the trigger signal in compliance with SIMUCON setting, ET7190 changes from simulation state to normal operation state(FMOD defined state)
2. Set the bus time, time refers to the operation time of ECU, generally set interbyte space(UTXP1T) as 0 or 1ms, bus release time(UFREET) as 25ms, allowed maximum interbyte space in reception of frame is 20ms.
3. As per the applied protocol, set frame length and check control. This is identical to the setting in normal state of operation.
4. Set UART bit definition and bus BaudRate, and define the correct TX/RX pin.
5. Start protocol, (UFRMCON<ON> is set ON, FMOD protocol is set correctly, and SIMU bit is set ON)

8.9.3 Example 1: Simulate 5 BaudRate Initialization Bus(ISO9141-2)

1. Set SIMUCON

SIMUCON.SMMOD=SM_SLOW_INIT_ISO;	SM_SLOW_INIT_ISO=0, simulate slow initialization
SIMUCON.SMADDR=0x33;	In slow initialization, trigger address is set as 0x33
SIMUCON.SMTTOL=78;	Irrelevant
SIMUCON.SMLTime=25;	Irrelevant
SIMUCON.SMKW[0]=0x08;	When 0x33 trigger address is received, the sent KEY Byte 1
SIMUCON.SMKW[1]=0x08;	When 0x33 trigger address is received, the sent KEY Byte 2
SIMUCON.SMW1SEND=150;	Time delay of sending 0x55 sync byte, 150ms
SIMUCON.SMW2SEND=10;	Time delay of sending KB1 keyword byte, 10ms
SIMUCON.SMW3SEND=1;	Time delay of sending KB2 keyword byte, 1ms
SIMUCON.SMW4SEND=30;	After /KB2 reverse byte is received, time delay of sending reverse trigger address, 30ms
SIMUCON.SMW4RMAX=50;	Maximum waiting time of receiving /KB2

2. Set Time Register

UTXP1T=0；	Interbyte gap 0ms
UFREET=25；	Bus release time 25ms
URXP1T=20；	Maximum sender allowed interbyte gap in reception 20ms

3. Set Frame Format and BaudRate Bit Time, etc.

UFRMV=0;	No each byte check
UFRMV.ENCS=1;	Enable Checksum
UFLENCON=0;	No length byte
UFLAND=0xFF;	Irrelevant
PINSEL=0;	Pin select TX0/RX0
UBRG=239;	BaudRate 10400
UMODE=0;	Clear EUART Mode Control Register
USTAT0=0;	Clear State Register
USTAT1=0;	

4. Start Bus

UFRMCON=0xC3	Protocol FMOD=011(USERMOD) SIMU=1, Enter simulation ECU trigger state in startup ON=1, Start UART

- By the above setting, ET7190 EUART is in a state of waiting for the initialization process of ISO9141, firstly wait for 5 Baud trigger address as SMADDR(0x33) signal.
- When ET7190 receives 0x33 trigger signal, ET7190 starts the transmission of 0x55 sync byte and keyword byte SMKW0(0x08) and SMKW1(0x08). Then the reversed value(0xF7) of SMKW2 is received. When the reception is completed, the reversed value 0xCC of SMADDR(0x33) is transmitted. After the initialization, bus enters the state of normal operation.
- When the bus enters the state of normal operation, ET7190 automatically clears UFRMCON<SIMU> bit.
- When the bus enters the state of normal operation, ET7190 loads one INIT=0b111(7) into the receive buffer URXB, with the data length of 0. And URX is set with interrupt flag. The effect of this data only notifies the user of the reception of valid initialization signal.

  After INIT=7 is received, user must prepare for the reception of tester request. And respond to data according to the request.

- In the process of initialization, if ET7190 receives the signal in no compliance with SIMUCON setting, ET7190 will keep waiting for the initialization of ISO9141-2. And no error will be generated. User can clear UFRMCON abort simulation.
- In the state of bus simulation(UFRMCON SIMU bit is cleared), data frame must not be written in UTXB transmit buffer. ECU is unable to receive message frame when it waits for the process of realization.

8.9.4 Example 2: Simulate 5 Baud Initialization Bus(KW1281)

1. Set SIMUCON

SIMUCON.SMMOD=SM_SLOW_INIT_1281;	SM_SLOW_INIT_1281=2, simulate KW1281 slow initialization
SIMUCON.SMADDR=0x01;	Engine module address 0x01, simulation engine module
SIMUCON.SMTTOL=78;	Irrelevant
SIMUCON.SMLTime=25;	Irrelevant
SIMUCON.SMKW[0]=0x01;	Sent KEY Byte 1, KK1281 protocol keyword is 01 8A
SIMUCON.SMKW[1]=0x8A;	Sent KEY Byte 2
SIMUCON.SMW1SEND=150;	Time delay of sending 0x55 sync byte, 150ms
SIMUCON.SMW2SEND=10;	Time delay of sending KB1 keyword byte, 10ms
SIMUCON.SMW3SEND=1;	Time delay of sending KB2 keyword byte, 1ms
SIMUCON.SMW4SEND=30;	Irrelevant, KW1281 has no reverse address, ECU must directly start the transmission of FF information, (Transmitted by user in simulation)
SIMUCON.SMW4RMAX=50;	Maximum waiting time of receiving /KB2(=0x75)

2. Set Time Register

UTXP1T=0；	Interbyte space 0ms
UFREET=25；	Bus release time 25ms
URXP1T=20；	Allowed sender maximum interbyte space in reception 20ms
UTXVT0=20；	In sending, maximum time of allowed receiver returned check byte
UTXVT1=1;	In receiving, time delay of returned check byte 1ms

3. Set Frame Format and BaudRate Bit Time, etc.

UFRMV=0xE1;	Configure KW1281 protocol, check and length control (Each byte check, last byte not checked, length byte is Byte 0, total length of data frame is Byte 0 value +1)
UFLENCON=0x80;	
UFLAND=0xFF;	
PINSEL=0;	Pin select TX0/RX0
UBRG=239;	BaudRate 10400 (Can be set as other BaudRate)
UMODE=0;	Clear EUART Mode Control Register
USTAT0=0;	Clear State Register
USTAT1=0;	

4. Start Bus

UFRMCON=0xC3	Protocol FMOD=011(USERMOD) SIMU=1, Enter simulation ECU trigger state in startup ON=1, Start UART

- By the above setting, ET7190 EUART is in a state of waiting for the initialization process of KW1281, firstly wait for 5 Baud trigger address as SMADDR(0x01) signal.
- When ET7190 receives 0x01 trigger signal, ET7190 starts the transmission of 0x55 sync byte and keyword byte SMKW0(0x01) and SMKW1(0x8A). Then the reversed value(0x75) of SMKW2 is received. When the reception is completed, after the initialization, bus enters the state of normal operation. For the next step, user writes UTXB to transmit the first ECU information frame and responds to the request of tester.
- When the bus enters the state of normal operation, ET7190 automatically clears UFRMCON<SIMU> bit.
- When the bus enters the state of normal operation, ET7190 loads one INIT=0b111(7) into the receive buffer URXB, with the data length of 0. And URX is set with interrupt flag. The effect of this data only notifies the user of the reception of valid initialization signal.

After INIT=7 is received, user must prepare for the reception of tester request. And respond to data according to the request.

- In the process of initialization, if ET7190 receives the signal in no compliance with SIMUCON setting, ET7190 will keep waiting for the initialization of KW1281. And no error will be generated. User can clear UFRMCON abort simulation.
- In the state of bus simulation(UFRMCON SIMU bit is cleared), data frame must not be written in UTXB transmit buffer. ECU is unable to receive message frame when it waits for the process of realization.

8.9.5 Example 3: Simulate Fast Initialization Bus(ISO14230)

1. Set SIMUCON

SIMUCON.SMMOD=SM_FAST_INIT;	SM_SLOW_INIT_ISO=1, Simulate fast initialization
SIMUCON.SMADDR=0x33;	Irrelevant
SIMUCON.SMTTOL=78;	Low-level allowed offset error. 6.4us*78=499us
SIMUCON.SMLTime=25;	Low-level time 25ms, must not be less than 10ms
SIMUCON.SMKW[0]=0x08;	Irrelevant
SIMUCON.SMKW[1]=0x08;	Irrelevant
SIMUCON.SMW1SEND=150;	Irrelevant
SIMUCON.SMW2SEND=10;	Irrelevant
SIMUCON.SMW3SEND=1;	Irrelevant
SIMUCON.SMW4SEND=30;	Irrelevant
SIMUCON.SMW4RMAX=50;	Irrelevant

2. Set Time Register

UTXP1T=0；	Interbyte space 0ms
UFREET=25；	Bus release time 25ms
URXP1T=20；	Allowed sender maximum interbyte space in reception20ms

3. Set Frame Format and BaudRate Bit Time, etc.

UFRMV=0;	Irrelevant
UFRMV.ENCS=1;	Irrelevant
UFLENCON=0;	Irrelevant
UFLAND=0xFF;	Irrelevant
PINSEL=0;	Pin select TX0/RX0
UBRG=239;	BaudRate 10400
UMODE=0;	Clear EUART Mode Control Register
USTAT0=0;	Clear State Register
USTAT1=0;	

4. Start Bus

UFRMCON=0xC1	Protocol FMOD=001(ISO14230)

SIMU=1, Enter the simulation ECU trigger state in startup
ON=1, Start UART

- By the above setting, ET7190 EUART is in a state of waiting for 25ms low-level pulse of the fast initialization of ISO14230, when SMLTIME set 25ms(±499us=0.5ms) pulse is received, ET7190 enters the state of normal operation.
- When the bus enters the state of normal operation, ET7190 automatically clears UFRMCON<SIMU> bit.
- When the bus enters the state of normal operation, ET7190 loads one INIT=0b111(7) into the receive buffer URXB, with the data length of 0. And URX is set with interrupt flag. The effect of this data only notifies the user of the reception of valid initialization signal.

After INIT=7 is received, user must prepare for the reception of tester request. And respond to data according to the request.

- In the process of initialization, if ET7190 receives the signal in no compliance with SIMUCON setting(25ms low-level pulse), ET7190 will keep waiting for the correct initialization. And no error will be generated. User can clear UFRMCON abort simulation.
- In the state of bus simulation(UFRMCON SIMU bit is cleared), data frame must not be written in UTXB transmit buffer. ECU is unable to receive message frame when it waits for the process of realization.

8.10 Logic Analysis of K Line

EUART is started when FMOD=0b111, the function of EUART is to capture the time of level jump at RX pin, in case of level jump, the value of 32-bit time register is loaded into URXB buffer, the capture precision of time register is 1.6us. After such configuration, PC application software can implement the infinite deep storage and display logic waveform for K-line communication below 20KBPS BaudRate.
EUART Message Buffer Structure(see 7.6)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Byte 1	L7	L6	L5	L4	L3	L2	L1	L0
Byte 2	SACK	TYPE<2:0>			CSERR	-	-	L8
Byte 3-262	Data0-Data259 Up to 260 bytes(used for the storage of time value, 65 time at most)							

8.10.1 Mode of Operation

- After starting the capture, the first capture begins in RX falling edge.
- After starting, ET7190 has a 32-bit time register for the continuous work, with the clock cycle of 1.6us. Time register counts from 0 till the maximum 0xFFFF-FFFF. After (About 6872s), repeat counting from 0.
- The length of EUART receive buffer is 262 bytes, EUART has 2 262-byte FIFO receive buffers(receiver has one temporary message buffer, user has no direct access, actually equivalent to 3-frame buffer). In capture, 32-bit time value is loaded in the temporary message buffer, each URXB buffer can load in 65 32-bit time values to the maximum(260 bytes/4=65).
- In message buffer, Byte 1/2, L<8:0> represents the length of valid data in buffer, the length value is always the integer multiples of 4 in capture mode. Other bits in Byte 1/2 are always 0. Example:Suppose two level changes are captured in buffer, data length L<8:0>=8 bytes, four bytes, Data0/1/2/3 are the time of the first level change, and Data4/5/6/7 are the time of the second level change.
- Time register value, 32-bit integer storage in the buffer: Low byte is followed by high byte.

8.10.2 Transmit to Message Buffer

To increase the transmission efficiency of data, when RX logic level changes, time value is stored in EUART temporary message buffer, user cannot directly read it. In the following two cases, data is transmitted to URXB FIFO buffer for user to read.

1. When one or more level change times are acquired in the temporary message buffer, if the level doesn't change at RX pin over URXP1T set time, then the data in temporary message buffer is transmitted to URXB. And the receive buffer flag bit INTF<URX> is set, if hardware interrupt is allowed, hardware interrupt will be generated. (URXP1T time setting must be greater than one byte transmission time, which must not be set as 0.)
2. If the number of level changes acquired in the temporary message buffer reaches 65(260-byte length), temporary buffer is fully loaded, then data in the temporary message buffer is transmitted to URXB. And the receive buffer flag bit INTF<URX> is set, if hardware interrupt is allowed, hardware interrupt will be generated.
3. If data in the temporary message buffer is transmitted to URXB, FIFO is fully loaded(two FIFOs contain data, and user doesn't read), then data in the temporary message buffer gets lost. And the error flag UERRF<OVF> is set. If hardware interrupt is allowed, hardware interrupt will be generated.

8.10.3 Configure and Start Capture

1. Set Time Register

URXP1T=10；	10msLevel change timeout 10ms(Buffer data transmission time)

4. Start Bus Acquisition

UFRMCON=0x87	Protocol FMOD=111 ON=1, Start UART

8.11 Read Vehicle Trouble Flash Code

For the early vehicle diagnosis, the special diagnostic socket usually reads TroubleCodes via flash code(indicator mounted on the diagnosis header, TroubleCode information is output according to the flashes of indicator and time duration). User can judge the time and frequency of flashes as per K Line Logic Acquisition in 8.10. This is not further described here.

Chapter 9 Other Special Functions and Universal Register

9.1 Special Function Register

9.1.1 INTF Interrupt Flag Register(addr: 0x00)

R-0	R/C-0	R/C-0	R-0	R/C-0	R-0	R/C-0	R-0
ERR	WKF	UTX	URX	CTX	CRX	JTX	JRX
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ERR**：Global Error Interrupt Flag Bits
1 = Global error interrupt generated,
0 = Error not generated, when enable interrupt error dismisses, automatically cleared

bit 6 **WKF**：Hardware Wake-up Interrupt Flag
1 = Wake-up interrupt generated, this bit is set ON when the device is woken from Sleep Mode.
0 = Wake-up interrupt not generated. Automatically cleared after INTF register is read.

bit 5 **UTX**：EUART Transmit Completion Interrupt
1 = In Frame Mode, set ON when one frame transmission is completed. In Character Mode, set ON when one byte is transmitted.
0 = Interrupt not generated, automatically cleared after INTF register is read.

bit 4 **URX**：EUART Receive Completion Interrupt
1 = Frame mode, when URXB receive buffer has any data, set ON. In Character Mode, when UTRXREG FIFO receive buffer has any data, set ON.
0 = No data received. When URXB or UTRXREG is read empty, automatically set Clear.

bit 3 **CTX**：CAN Transmit Completion Interrupt
1 = Set ON when one frame data transmission is completed
0 = Transmit interrupt not generated, automatically cleared after INTF register is read.

bit 2 **CRX**：CAN Receive Completion Interrupt
1 = Set ON when the receive buffer CRXB has data
0 =Automatically cleared when the receive buffer has no data and CRXB is read empty

bit 1 **JTX**：J1850 Transmit Completion Interrupt
1 =Set ON when JTXB one frame data transmission is completed
0 =Transmit interrupt not generated. Automatically cleared after INTF register is read.

bit 0 **JRX**：J1850 Receive Completion Interrupt
1 =Set ON when the receive buffer JRXB has data
0 =Receive interrupt not generated. Automatically cleared when JRXB is read empty

9.1.2 INTIE Global Interrupt Enable Register(0x01)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ERRE	WKE	UTXE	URXE	CTXE	CRXE	JTXE	JRXE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ERRE**：Enable Global Error Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 6 **WKE**：Enable Hardware Wake-up Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 5 **UTXE**：Enable EUART Transmit Completion Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 4 **URXE**：Enable EUART Receive Completion Interrupt
1 = Enable generate hardware interrupt
0 = Disable interrupt.
bit 3 **CTXE**：Enable CAN Transmit Completion Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 2 **CRXE**：Enable CAN Receive Completion Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 1 **JTXE**：Enable J1850 Transmit Completion Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.
bit 0 **JRXE**：Enable J1850 Receive Completion Interrupt
1 = Enable generate hardware interrupt.
0 = Disable interrupt.

9.1.3 SLEEP Control Register(0x02)

R/W-0	R-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0
ON	-	-	-	-	WJE	WUE	WCE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

bit 7 **ON**：Device Enters Sleep
1 = Device enters sleep when 1 is written.
0 = Operation not generated.
bit 2 **WJE**：Enable J1850 Bus Wake-up

1 = When data is transmitted in J1850 bus, enable device wake-up from Sleep Mode when 1 is written.
In reading, if as 1, represent the last is J1850 bus wake-up device.
0 = Disable J1850 wake-up device from Sleep Mode when 0 is written.

bit 1 **WUE**：Enable EUART Bus Wake-up
1 = When data is transmitted in EUART bus, enable device wake-up from Sleep Mode when 1 is written.
In reading, if as 1, represent the last is EUART bus wake-up device.
0 = Disable EUART wake-up device from Sleep Mode when 0 is written.

bit 0 **WCE**：Enable CAN Bus Wake-up
1 = When data is transmitted in CAN bus, enable device wake-up from Sleep Mode when 1 is written.
In reading, if as 1, represent the last is CAN bus wake-up device.
0 = Disable CAN bus wake-up device from Sleep Mode when 0 is written.

9.1.4 ODC Open Drain Output Control Register(0x0B)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	ODCP5	ODCP4	ODCP3	ODCP2	ODCP1	ODCP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Each PORT can be separately configured as digital output or open-drain output. When the direction of port is configured as output port, if ODCPx bit is set ON, then current IO port pin is OD open-drain output port.
This open-drain features allows for using the external pull-up resistor, output voltage above 3.3V will be generated at the pin of required digital function.

9.1.5 UPR Port Input Pull-up Control Register(0x0C)

R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	UPR4	UPR3	UPR2	UPR1	UPR0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

PORT4…PROT0 When port is configured as digital input, if UPRx bit is set ON, then current IO port pin is the internal pull-up enable. The current typical value if internal pull-up to VDD is 250uA. PROT5 has no pull-up function.

UPRx (x=0-4)
1 = Pull-up at enable input level change pin
0 = Pull-up at disable input level change pin

9.1.6 PDIR Port Direction Control Register(0x0D)

R-0	R-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
-	-	DIRP5	DIRP4	DIRP3	DIRP2	DIRP1	DIRP0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Set the direction of PORT5…PROT0 port pin, if DIRPx bit is set ON, then current IO port pin is input. If DIRPx bit is set Clear, then current IO port pin outputs function.

DIRPx (x=0-5)

1 = IO is the input pin

0 = IO is the output pin

9.1.7 PORT Read Write Register(0x0E)

R-0	R-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-	-	P5	P4	P3	P2	P1	P0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Read or set the level at PORT pin.
When pin is configured as input, write bit is invalid. If configured as output, register bit operation command can be used to only change one pin level to set ON or Clear.

Px (x=0-5)

1 = Pin is in the high level

0 = Pin is in the low level

9.1.8 ADC Start Voltage Measure ADC Register(0x70)

The function of write operation on ADC register only starts one AD conversion and reads the voltage value at VM pin. Irrelevant to the written value. The converted value is loaded in ADCB register set. Conversion time is 3.2us.
Read register ADC value is invalid.

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
-	-	-	-	-	-	-	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

9.2 Special Function Register Set

9.2.1 DRVUBRG User UART BaudRate Scaler(Length 2 bytes) (addr:0x07)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BRG<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BRG<15:8>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: It is more convenient to treat this buffer as one 16-bit integer<Bits 15: 0> in application

When MOD=0, ET7190 communicates data with user via the internal UART, the initial BaudRate of communication is defined by B1/B0 pin, user may dynamically change the connected communication BaudRate when applied, BaudRate=10MHZ/(DRVUBRG+1), up to 10MHZ. Note that the allowed highest BaudRate of hardware circuit cannot be exceeded when BaudRate changes.

BRG<15:0>: BaudRate Scaler Bits

9.2.2 ADCB Measure Level Voltage ADC Register(10bit of 2 bytes)(addr:0x08)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADC<7:0>							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
-						ADC<9:8>	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE 1

Legend： C = Writable bit, but only write 0 to clear this bit R = Readable bit W = Writable bit U = Unimplemented bit, read as "0"
-n = Value at POR(Reset) 1 = Bit is set 0 = Bit is cleared x = Bit is Unknown

Note: It is more convenient to treat this buffer as one 10-bit integer<Bits 9: 0>in application

ADC<9:0>：

10-bit ADC result value, any value must be firstly written in ADC register to start AD conversion before reading. VM input is the highest level VDD(3.3V). When VDD is input, ADC value is 0x3FF. Actual measured level voltage must be calculated according to the external bleeder circuit.

9.2.3 ERRB Error Flag Register Map Register Set(6 bytes) (addr:0x3C)

This register set maps to the content of 5 error registers. Register set reads only and doesn't change content or clear in reading. Its function is to make easy for reading all error flags using one command when user simultaneously operates multiple modules in EUART/J1850/CAN.

	Registers
Byte0	UERRF Error Flag Register
Byte1	JERRF Error Flag Register
Byte2	CINTF0 CAN Error Flag Register 0
Byte3	CINTF1 CAN Error Flag Register 1
Byte4	CINTF2 Multi-frame Message Error Flag
Byte5	Not Used

9.3 Device ID

Device ID is read via command 0xFF, ID consists of 2 bytes, Byte 1 is 0xA1, Byte 2 higher 2 bits is 0b01. In which, Byte 2 lower 6 bits Bit5:0 is the version of device.

ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2
1	0	1	0	0	0	0	1
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE1

ID1	ID0	Ver5	Ver4	Ver3	Ver2	Ver1	Ver0
0	1	Device Ver<5:0>					
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

BYTE2

Chapter 10 SPI/UART User Interface

10.1 SPI

10.1.1 Instruction Summary

ET7190 design can be directly connected with the serial peripheral interface(SPI) of many microcontrollers. External data and instruction are transported to the device via MOSI pin. And data is transported in through the rising edge of SCK clock signal. ET7190 is transported out via MISO pin through SCK falling edge. SCK clock frequency must be equal to or smaller than 12MHZ. For any operation, the pin must be kept in low level. Table 10-1 lists instruction bytes of all operations. For more details of the input and output timing under the relevant SPI operation mode, please refer to Fig.10-2 to Fig.10-11.
When MOD pin is in high level, device operates in SPI mode, when MOD pin is in low level, device user interface operates in UART mode. If microcontroller is used, it is strongly recommended to use SPI operation mode.

Note: After the pin is set in low level, the first byte that ET7190 intends to receive is instruction/command byte. This means pin must pull up in high level, then reduce to low level, in order to employ another command.

Table 10-1: SPI Instruction Set

Instruction Name	Instruction Format	HEX	Description
Reset CMD_RESET	1 1 1 0 1 0 1 0	0xEA	Reset device
Read ID CMD_RDID	1 1 1 1 1 1 1 1	0xFF	Read device ID and version
Read Register CMD_RD_REG	0 0 A A A A A A	0x00	A(0-63) is register address, read the register value with address as A
Read Bufer (Read Register Set) CMD_RD_BUFF	0 1 A A A A A A	0x40	A(0-63) is buffer address, read the buffer with address as A, read biggest length is the length of buffer, return the first byte value as instruction byte.
Write Register CMD_WR_REG	1 0 0 0 0 0 0 0	0x80	Write register
Register Bit Operation (set/Set ON) CMD_SET_REG	1 0 0 0 0 0 0 1	0x81	Register ON bit operation
Register Bit Operation (clr/Set Clear) CMD_CLR_REG	1 0 0 0 0 0 1 0	0x82	Register Clear bit operation
Write Long Buffer One Byte CMD_WR_BUFF_BYTE	1 0 0 0 0 0 1 1	0x83	Only rewrite one byte in the long buffer
Fill Long Buffer CMD_FILL_BUFF	1 0 0 0 1 0 0 0	0x84	Fill the entire buffer with a fixed value
Write Buffer CMD_WR_BUF	1 0 1 0 0 0 0 1	0xA1	Continuously write data in the entire buffer(the buffer with length equal to or smaller than 16 bytes)

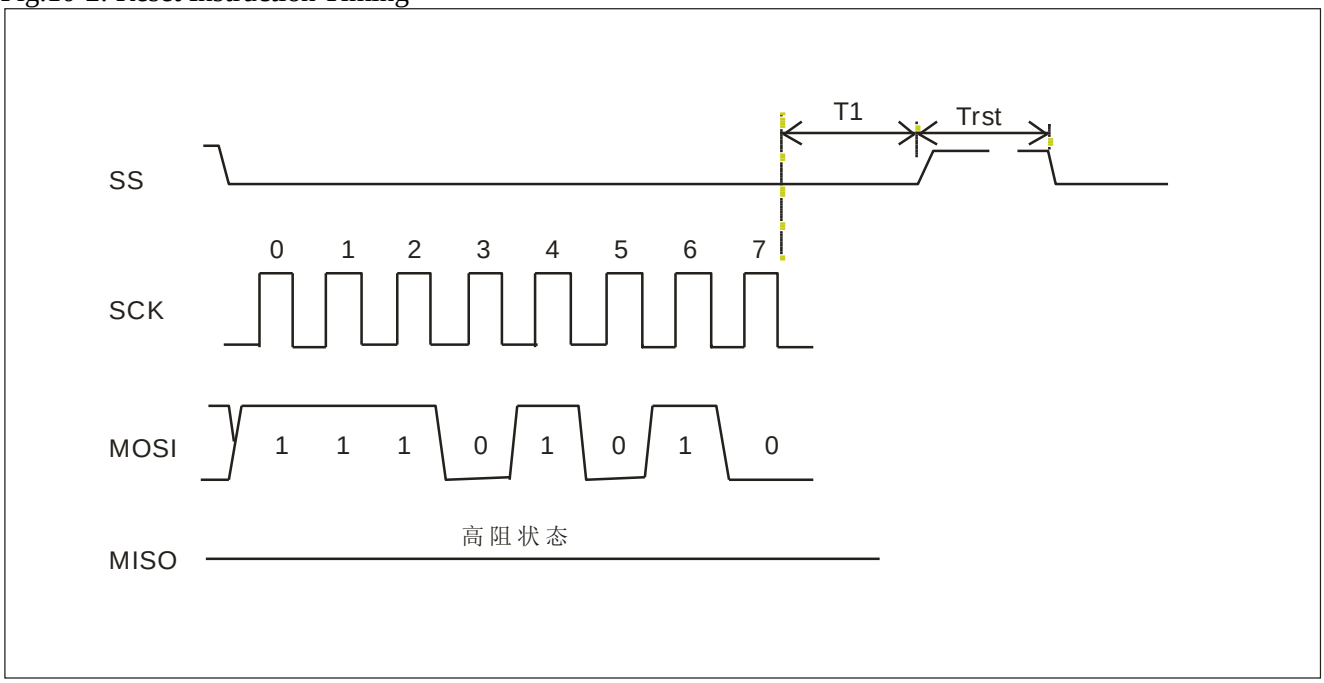
F			
Write Long Buffer CMD_WR_LB	1 0 1 0 1 0 0 0	0xA8	Continuously write data in the entire long buffer(the buffer with length bigger than 16 bytes)

Note: "Buffer" and "Register Set" are the same concept in this manual. They are represented by buffer in this chapter.

10.1.2 Reset Instruction

Reset instruction can reinitialize ET7190 internal register. This command is transmitted to ET7190 via SPI< its function is identical to RESET pin. Reset instruction is single-byte instruction. Firstly set the pin in low level to select the device, and transmit command byte, then pull up the pin in high level after the transmission is completed. It is strongly recommended to use the transmit reset instruction(or set RESET pin in low level) as one part of the device power-on initialization.

Fig.10-2: Reset Instruction Timing

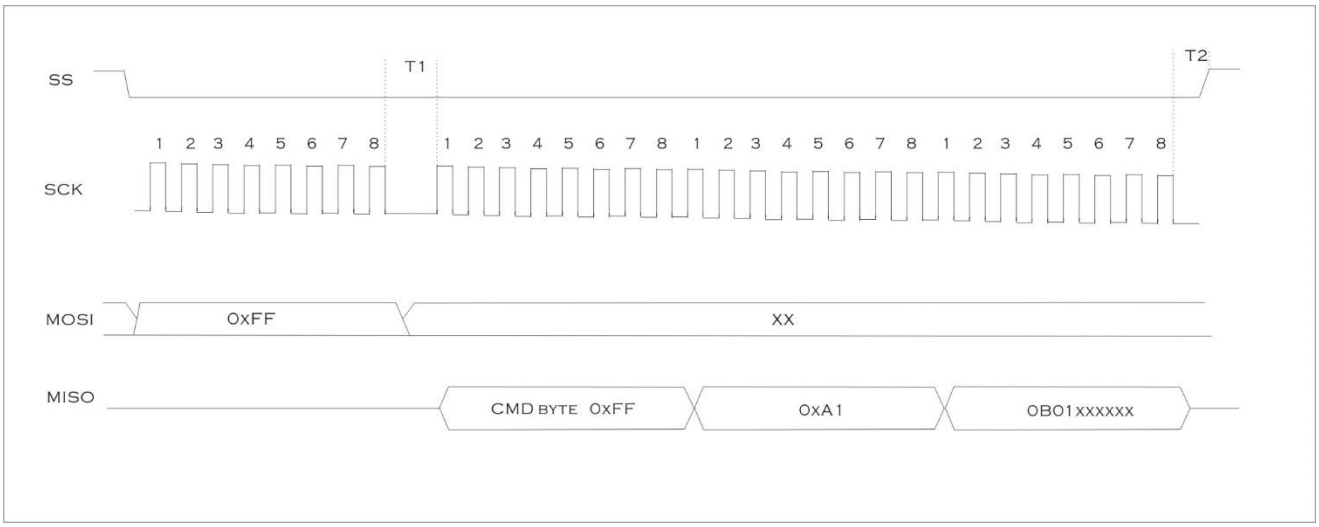


T1 is instruction delay time, refer to Table 10-2, minimum value is 10us. After the device receives reset instruction, enter the state of normal operation, require 20MS, after user resets the device, the next instruction is not sent until the minimum delay of 20MS.

10.1.3 Read Device ID

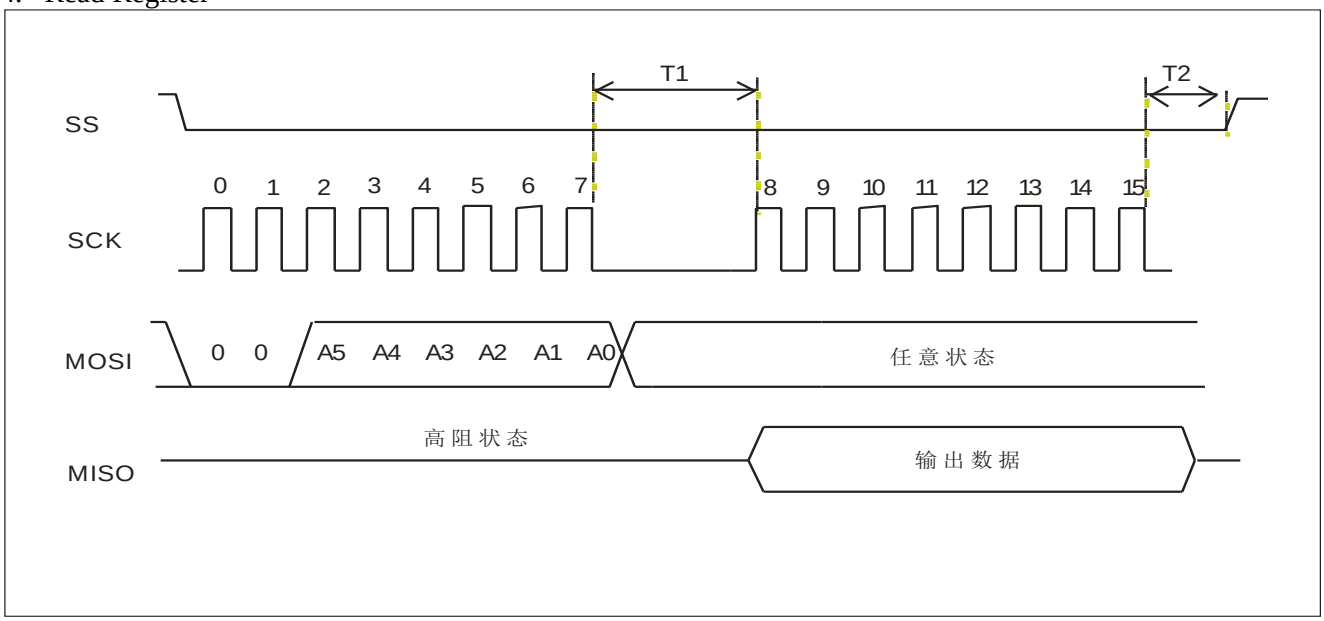
Set the SS pin in low level to start read instruction. After that, send read instruction 0xFF to ET7190. After read ID instruction is received, ET7190 will shift out 1 CMD byte and 2 bytes of ID data via MISO pin. Therefore, read out ID data through the continuous supply of clock pulse. Read operation can be ended by pulling up the SS pin level(see Fig.10-3), xxxxx 5 bits is the version of device.

Fig.10-3: Read Device ID Timing



10.1.4 Read Register

Set the pin in low level to start read instruction. After that, send read register instruction to ET7190 in sequence(including 6-bit address code A5-A0). After the read instruction is received, ET7190 will shift out the data in designated address register via SO pin. Read operation can be ended by pulling up pin level, see Fig.10-4. Read Register



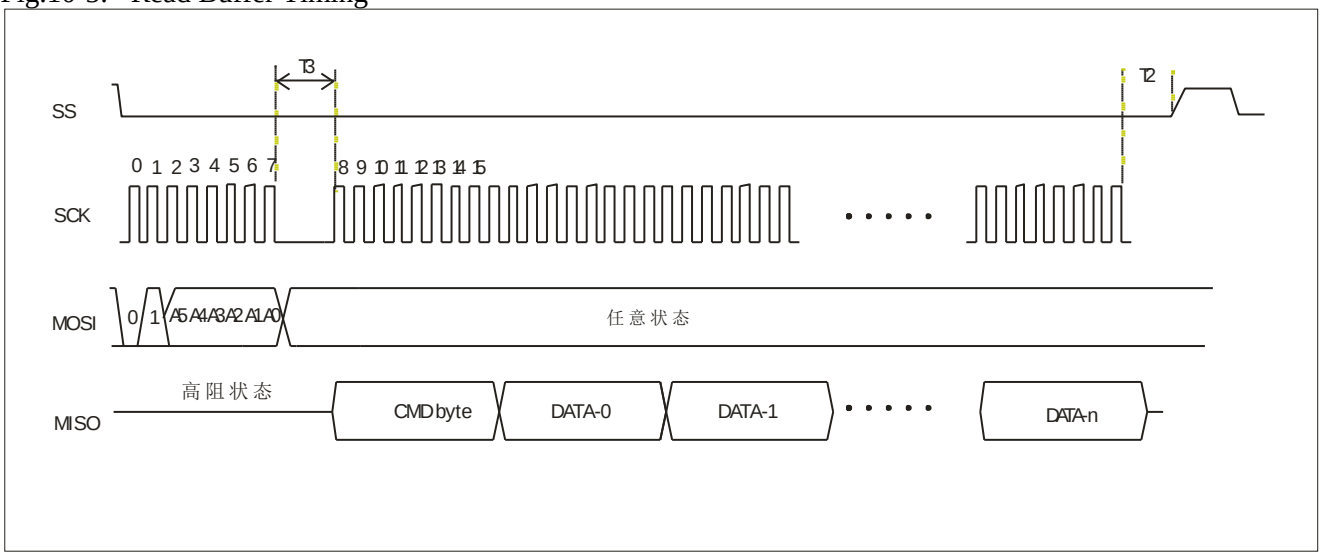
10.1.5 Read Buffer(Register Set)

Set the SS pin in low level to start read instruction. After that, send read register instruction to ET7190 in sequence(including 6-bit map address code A5-A0). After the read instruction and address code are received, ET7190 first shift-out byte is command byte value, if the value is not equal to the sent instruction byte(including address code) value, it represents that ET7190 doesn't prepare the shift data well. Later shift data may be error.

Buffer data is shifted out via MISO pin. After each data byte is shifted, the byte pointer in device will automatically plus 1 to point to the next buffer byte. Thus, continuous read operation can be processed on the next buffer by the continuous supply of clock pulse. A random number of data in the continuous buffer can be

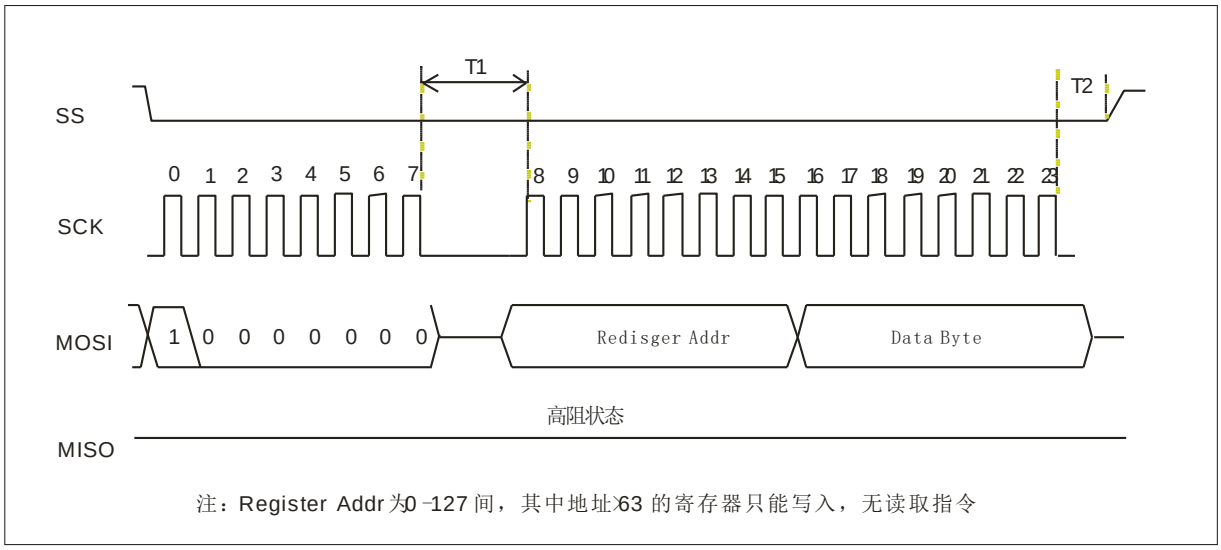
read in sequence by this method. Read operation can be ended at any time by pulling up the SS pin level(see Fig.10-5).
But the continuous read length must not exceed the length of this buffer, or data in excess of this length is read invalidly.
ET7190 buffer has less than 16 bytes(including 16 bytes) buffer and more than 16 bytes long buffer, the read instructions are identical, but note T3 time, when the common buffer(below 16 bytes) T3 is read as 10us, when long buffer is read, T3 is recommended as 50us. Refer to Table 10-2.
Note: Long buffer means the following buffers, which are: CFTXB(256) / UTXB(262) / LIDCON(64) / IDIFRCON(256) / HDFILTER(256), due to longer length of buffer, ET7190 is requied to prepare data and load SPI transmitter in reading, so longer T3 time(50us) is required.

Fig.10-5: Read Buffer Timing



10.1.6 Write Register

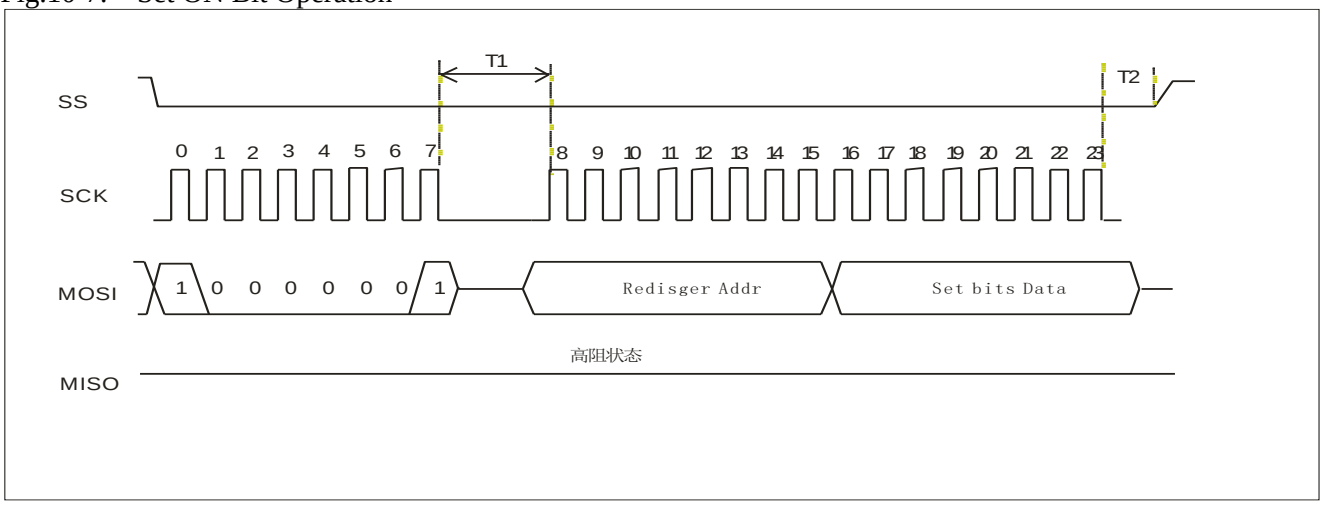
Set the SS pin in low level to start write operation. After that, send write register instruction 0x80, address code(Register Addr) and one byte data to ET7190 in sequence. Pin changes in high level, indicating the completion of write register. If the pin is pulled up to high level before the byte 8-bit data is loaded, this register will write the error data. For write operation timing, please refer to Fig.10-6.
Fig.10-6: Write Register



注：Register Addr 为0 -127 间，其中地址63 的寄存器只能写入，无读取指令

10.1.7 Write Register(Bit Operation Set ON)

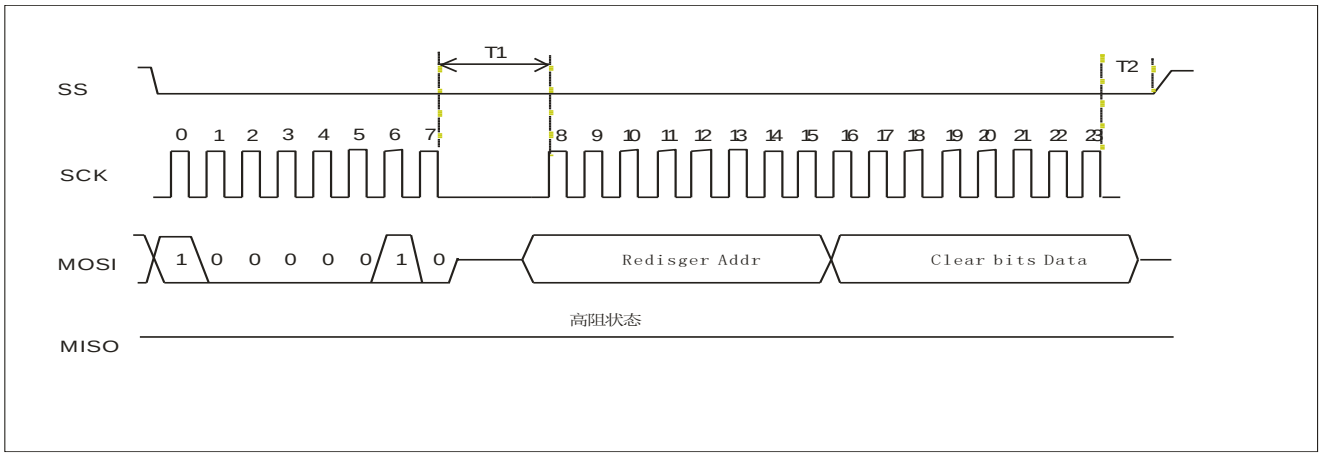
Function of Bit Operation Set ON Instruction: Set the register bit corresponding to ON bit as ON. Other bits are kept. It is only valid for the register with address in 0-63. Set the SS pin in low level to start write operation. After that, send write register bit instruction 0x81, address code and one byte data to ET7190 in sequence. Pin changes in high level, indicating the completion of write register. If the pin is pulled up to high level before the byte 8-bit data is loaded, this register will write the error data. For write operation timing, please refer to Fig.10-7. Set ON Bit Operation



10.1.8 Write Register(Bit Operation Clear)

Function of Bit Operation Set Clear Instruction: Set the register bit corresponding to Clear bit as Clear. Other bits are kept. It is only valid for the register with address in 0-63. Set the pin in low level to start write operation. After that, send write register bit instruction 0x82, register address code and one byte data to ET7190 in sequence. Pin changes in high level, indicating the completion of write register. If the SS pin is pulled up to high level before the byte 8-bit data is loaded, this register will write the error data.

Fig.10-8: Clear Bit Operation

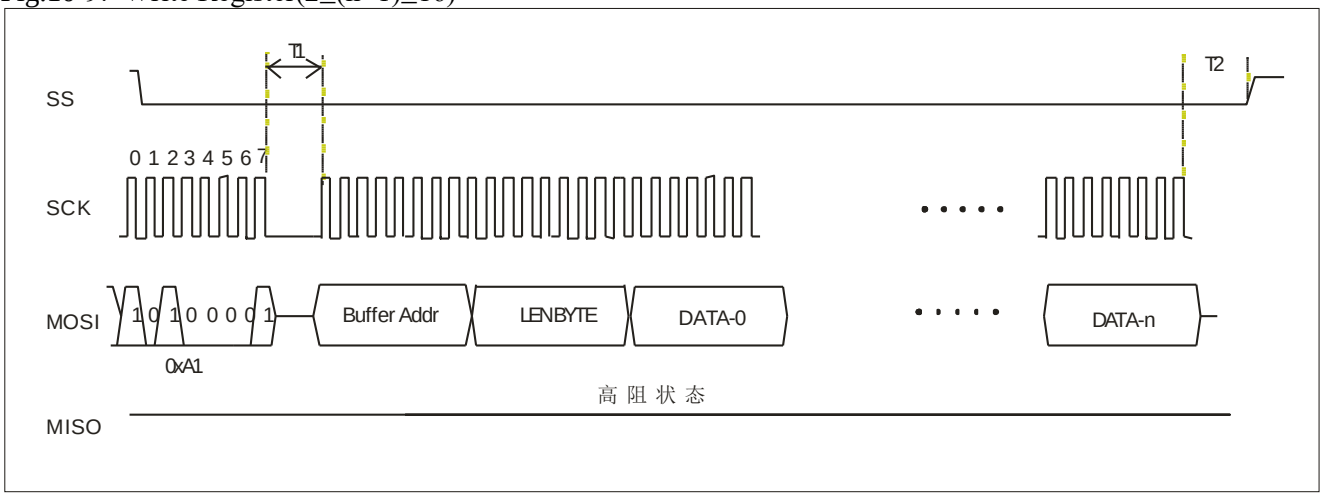


10.1.9 Write Buffer(Buffer Length≤16 Bytes)

Note: Only use A8 command to write in due to different instructions and different timings of the buffer with length greater than 16 bytes, see Write Long Buffer Content in 10.1.10. There are only 5 long buffers with length greater than 16 bytes, which are: CFTXB(256) / UTXB(262) / LIDCON(64) / IDIFRCON(256) / HDFILTER(256)

Set the SS pin in low level to start write operation. After that, ET7190 sends write instruction 0xA1, buffer map address code, buffer length Len Byte(2-16) and data up to 16 bytes in sequence. As long as the pin keeps in low level, the sequence write operation can be done to the continuous buffer by the continuous shift-in of data byte. The number of written data must be equal to the actual data length of buffer, if the written data is not enough, ET7190 will fill the unknown data, if written in excess of buffer length, the byte in excess of such length will be neglected.

Fig.10-9: Write Register(2≤(n+1)≤16)



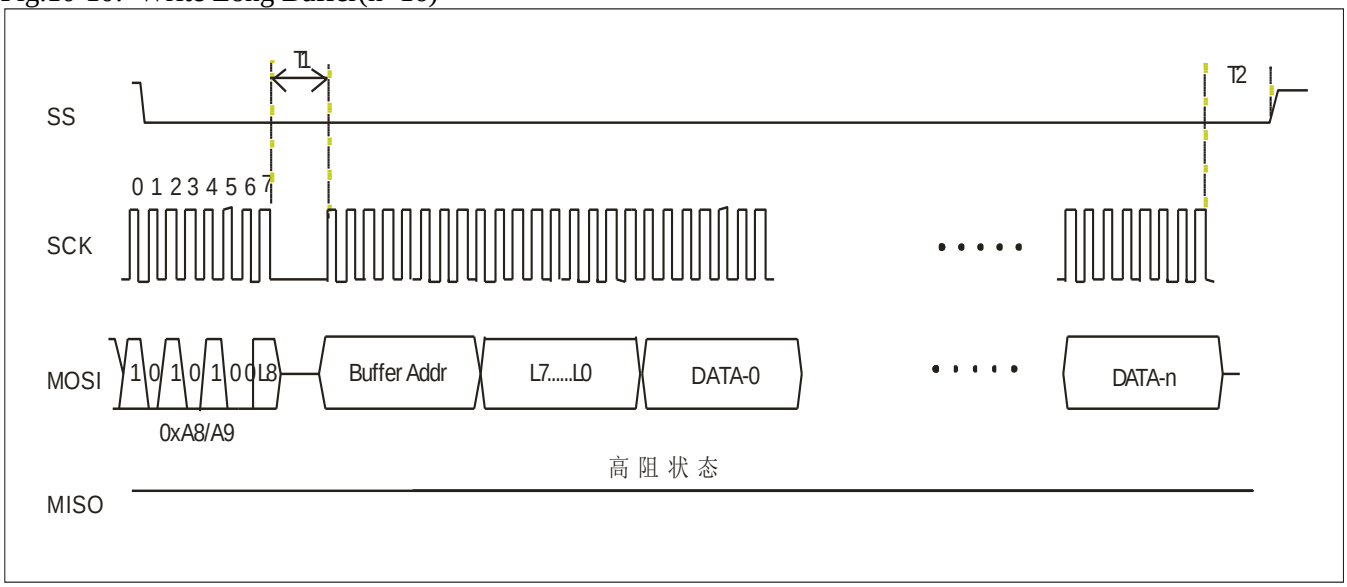
10.1.10 Write Long Buffer(Buffer Length>16 Bytes)

This command is only valid for 5 long buffers as follows:
CFTXB(256) / UTXB(262) / LIDCON(64) / IDIFRCON(256) / HDFILTER(256)
Set the SS pin in low level to start write operation. After that, ET7190 sends write instruction(0xA8+L8), buffer map address code, length byte lower 8 bit and continuous maximum 262 bytes data. As long as the pin keeps in low level, the sequence write operation can be done to the continuous buffer by the continuous shift-in of data byte. For the rising edge of SCK pin, data byte will start from Bit DATA0 to write in the buffer in

sequence. The number of written data must be equal to L<8:0>, if the written data is not enough, ET7190 will fill the unknown data, if written in excess of L<8:0>, the byte in excess of such length will be neglected.
L<8:0> must not exceed the actual length of buffer. The number of write buffer data is the value of L<8:0>. For most cases of CFTXB and UTXB transmit buffer, the transmitted data length will not use the entire buffer, so just write the data in actual length to buffer.
The effect of LIDCON / IDIFRCON / HDFILTER long buffer is to configure the relevant control unit, rarely modified in using, and generally this command is completely written in one time. If one byte value of buffer needs to be modified in using, 0x83 command can be used to change one byte value in the buffer.

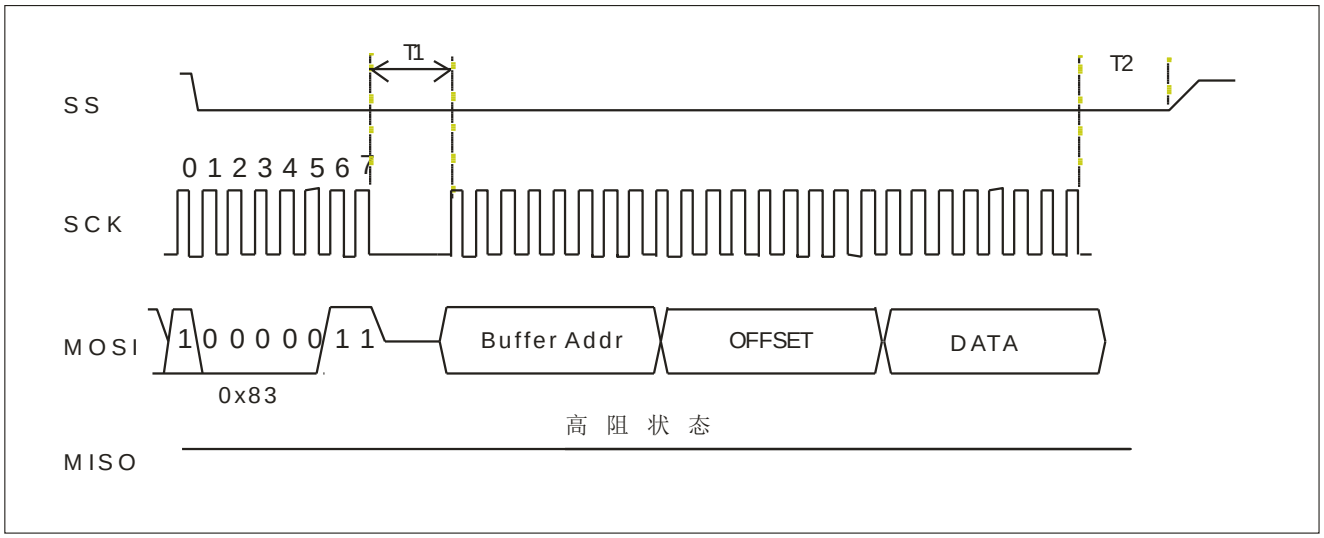
Note: For the fast setting method of 3 configuration buffers, LIDCON / IDIFRCON / HDFILTER, 0x84 Long Buffer Fill Command clear can be generally used to write the specific ID byte using 0x83 command.

Buffer Addr：Buffer map address, the value in 0-63.
Fig.10-10: Write Long Buffer(n>16)



10.1.11 Write One Byte in Long Buffer(0x83)

This command is only valid for 3 buffers, LIDCON / IDIFRCON / HDFILTER, the effect is to quickly change one byte of Configuration Table. UTXB, CFTXB is meaningless to change one byte in the transmit buffer.
Fig.10-11:

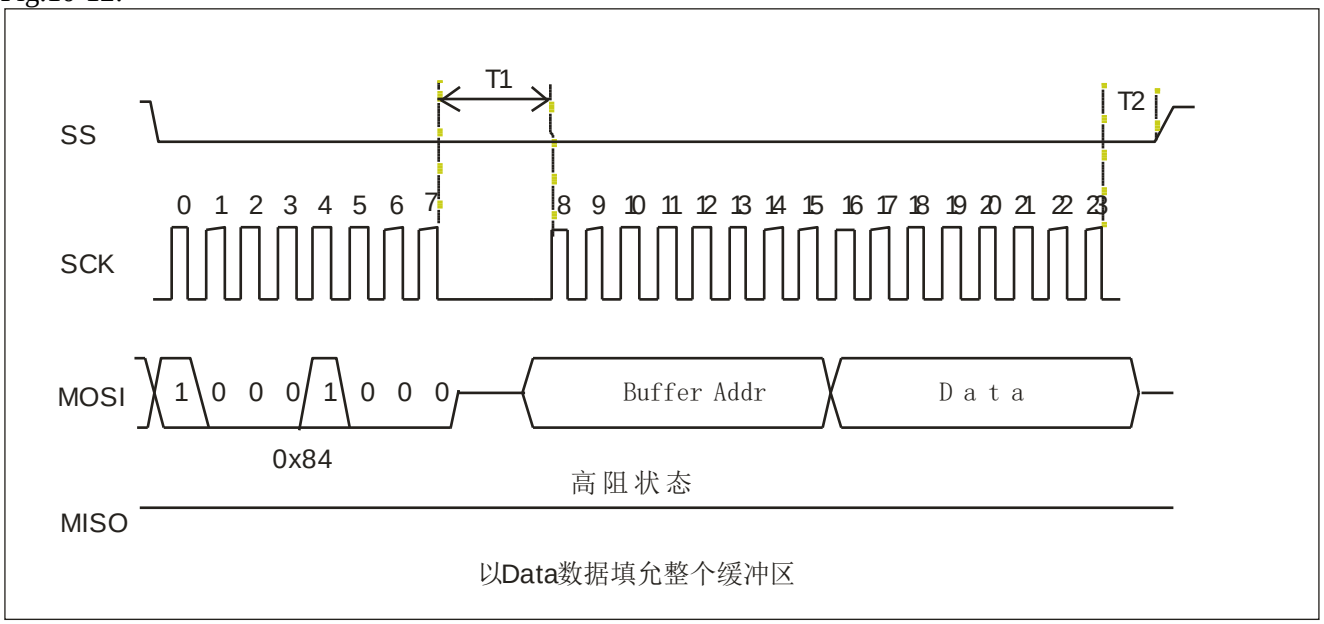


OFFSET is the address offset of byte written in the buffer(Start from 0, offset must not exceed the size of buffer). DATA is the written data. Buffer Addr is the buffer map address, which is only one of these three buffers, LIDCON / IDIFRCON / HDFILTER, and any other value is invalid.

10.1.12 Fill Long Buffer(0x84)

This command is only valid for 3 buffers, LIDCON / IDIFRCON / HDFILTER. The effect is to quickly clear or set Configuration Table.

Fig.10-12:



Buffer Addr is the buffer map address, which is only one of these three buffers, LIDCON / IDIFRCON / HDFILTER, and any other value is invalid.

10.1.13 Table of Time Parameters

Code	Description	Minimum	Typical	Maximum	Unit
T1	Instruction delay	1	10	-	us
T2	Data transmission	0	2	-	us

	completion delay				
T3	Read buffer instruction delay	10	10	-	us
	Read long buffer instruction delay	20	50	-	us
Trst	Device reset delay	15ms	20ms	-	ms
Tdly-1	Time slot between two commands: Pin becomes higher until the next pin becomes lower	2	10	-	us
Tdly-2	Write long buffer command delay: When write long buffer or Fill long buffer command is completed, pin becomes higher until the next pin becomes lower, command start time	20	50	-	us
Tdly-3	Time slot between two data bytes	0	0	∞	us

Note: The above time is not strictly tested.

10.2 UART Interface

When MOD pin is in high level, the device operates in SPI mode; when MOD pin is in low level, the device user interface operates in UART mode. In UART mode, ET7190 sends data to user via MISO/UTX pin, universal MOSI/URX receives user data, the data transmission format is 1 Start Bit+8 Digit Bit+1 Stop Bit. BaudRate is variable. The highest serial communication BaudRate is 10MHZ. When user sets BaudRate, it must not exceed the allowed highest BaudRate of user actual external hardware circuit.

10.2.1 BaudRate Setting

1. ET7190 reset initial BaudRate is determined by the potential at B1/B0 pin(B1/B0 pin builds in pull-up resistor, standoff=1), initial reset values are given below:

B1	B0	DRVUBRG	≈BaudRate bps
0	0	1040(0x28A0)	9600
1	0	259(0x0103)	38400
0	1	86(0x0056)	115200
1	1	38(0x0026)	256000

Table of Common BaudRates

DRVUBRG	≈BaudRate bps		DRVUBRG	≈BaudRate bps
1040 (0x28A0)	9,600		77 (0x004D)	128,000
520 (0x0208)	19,200		38 (0x0026)	256,000
390 (0x0186)	25,600		19 (0x0013)	500,000
346 (0x015A)	28,800		9 (0x0009)	1,000,000
259 (0x0103)	38,400		4 (0x0004)	2,000,000
232 (0x00E8)	43,000		3 (0x0003)	2,500,000
178 (0x00B2)	56,000		2 (0x0002)	3,333,333
173 (0x00AD)	57,600		1 (0x0001)	5,000,000
86 (0x0056)	115,200		0 (0x0000)	10,000,000

2. Equation of BaudRate Calculation;

$$\text{Baudrate} = \frac{10MHZ}{(DRVUBRG+1)} \qquad \text{DRVUBRG} = \frac{10MHZ}{\text{Baudrate}} - 1$$

DRVBRG is the value of 0x0000~0xFFFF

3. User may change the value of register set DRVUBRG(2 bytes) at any time in communication process, and use the new BaudRate.

10.2.2 Instruction and Format

User transmits the unified 4 bytes header and data items, as given below:

Instruction Function	4 Bytes Instruction(HEX)				Remark
	BYTE 1	BYTE2	BYTE3	BYTE 4	
Reset CMD_RESET	EA	00	00	00	Only send the next instruction after a delay of 20ms
Read ID CMD_RDID	FF	00	00	00	Controller delivers back 2 bytes ID value

Read Register	00	addr	00	00	Controller delivers back the 1 byte register value with address as addr(0~63)
Read Buffer (Read Register Set) CMD_RD_BUFF	40	addr	00	00	When the receiver accepts 4 bytes instruction, deliver back all data of buffer with map address as addr(0~63). (Except URXB, URXB transmitted length is the actually accepted data length)
Write Register CMD_WR_REG	80	addr	data	00	Write data in the register with address as addr(0~127)
Register Bit Operation (set/ ON) CMD_SET_REG	81	addr	data	00	Set ON some bits in register with address as addr(0~63). (data "1" bit is set ON, keep the "0" bit unchanged)
Register Bit Operation (clr /Clear) CMD_CLR_REG	82	addr	data	00	Set Clear some bits in register with address as addr(0~63). (data "0" bit is set Clear, keep the "1" bit unchanged)
One Byte in Write Long Buffer CMD_WR_BUFF_BYTE	83	addr	offset	data	Rewrite one byte in long buffer with map address as addr to data, the rewritten byte offset is offset(start from 0, this command is only valid for 3 buffers, LIDCON / IDIFRCON / HDFILTER)
Fill Long Buffer CMD_FILL_BUFF	84	addr	data	00	Fill the long buffer with map address as addr with all data values, this command is only valid for 3 buffers, LIDCON / IDIFRCON / HDFILTER 这
Write Buffer CMD_WR_BUFF	A1	addr	len	00	Write the buffer ≤16 bytes long, continuously transmit Len data to the controller after 4 bytes instruction is transmitted. Len must be equal to the length of buffer
Write Long Buffer CMD_WR_LB	A8	addr	Len0	Len1	Continuously transmit Len data to the controller after 4 bytes instruction is transmitted. Len≤Actual buffer length, this command is only valid for 5 buffers: CFTXB(256) / UTXB(262) / LIDCON(64) / IDIFRCON(256) / HDFILTER(256)

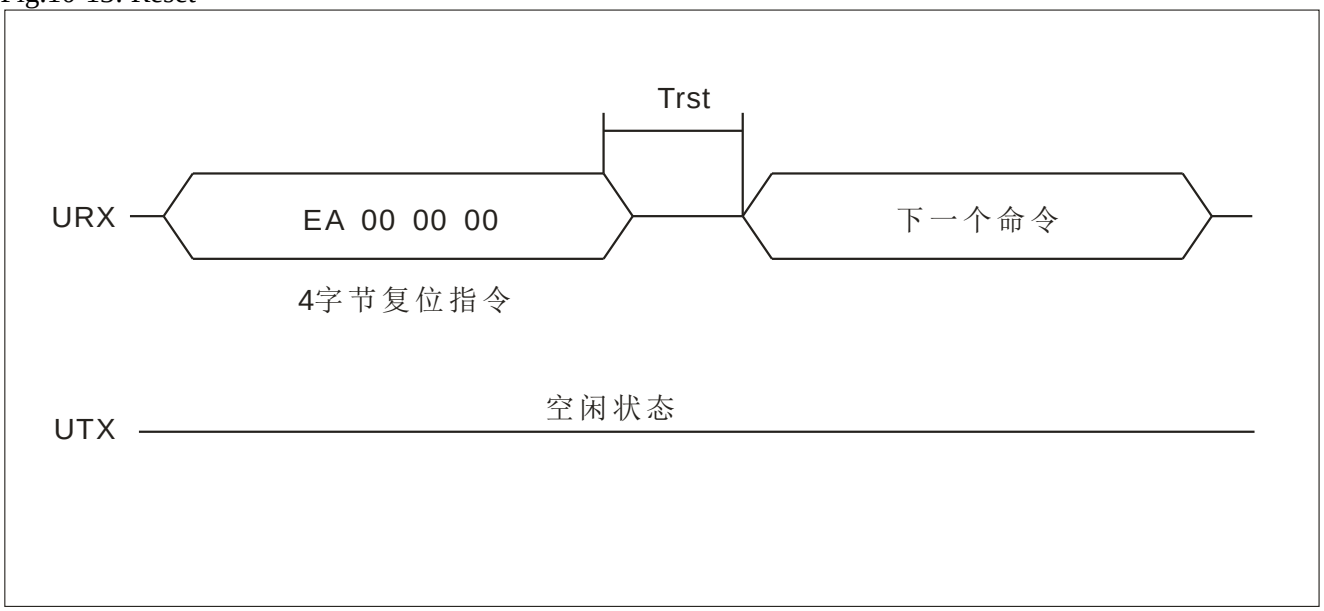
- Byte 1 is ET7180 command byte, and its value is identical to the command of SPI mode.
- After Byte 4 instruction is transmitted, for all read instructions, controller will deliver back one or more data. In the process of data back delivery, normally user must not send another new instruction. The next instruction is not sent until all data are received. If the controller doesn't transmit all data after new instruction has been sent(complete all 4 bytes), ET7190 controller aborts the original read instruction data delivered back from UTX pin, and directly executes the new instruction.
- For the write buffer0xA1(or write long buffer instruction 0xA8), after user sends 4 bytes instruction, ET7190 waits for the reception of user transmitted len data. (At that time, user must send len data, or ET7190 will keep waiting), and the number of transmitted data must not exceed len. (After ET7190 receies len data, it deems that this write buffer instruction is completed. Multiple transmit bytes that follow are considered as the new instruction bytes, which will lead to the chaos of user instruction. This UART mode is unlike SPI mode, which can abort at any time using pin).
- When ET7190 receives instruction, it must be 4 bytes, and instruction must be executed when ET7190 receives Byte 4.

- In case of chaotic instruction, user can send 1(Start Bit)+8(Data Bit)+1(Even Parity Check Bit)+1(Stop Bit) format even parity check 0 value data(when the even parity check value is also 0, which is equivalent to sending 9 “0” bit) at any time, enforce the device to enter the instruction state of receiving 4 bytes.(Actually after user sends this data, because there is no even parity check format when the device receivs, when the device receives data, the last stop bit is 0, with bit error generated, ET7190 returns to the original state of receiving iinstruction according to the error state, and neglects current error data)

10.2.3 Reset Instruction

Reset instruction can reinitialize ET7190 internal register. Its function is identical to RESET pin. In RS232 asynchronous serial communication setup with PC, user can use RTS/CTS pin at RS232 interface to go through level conversion in connection with ET7190 /RST pin, and control ET7190 device reset. After the reset, user must wait for the device reset after a delay of Trst(20ms).

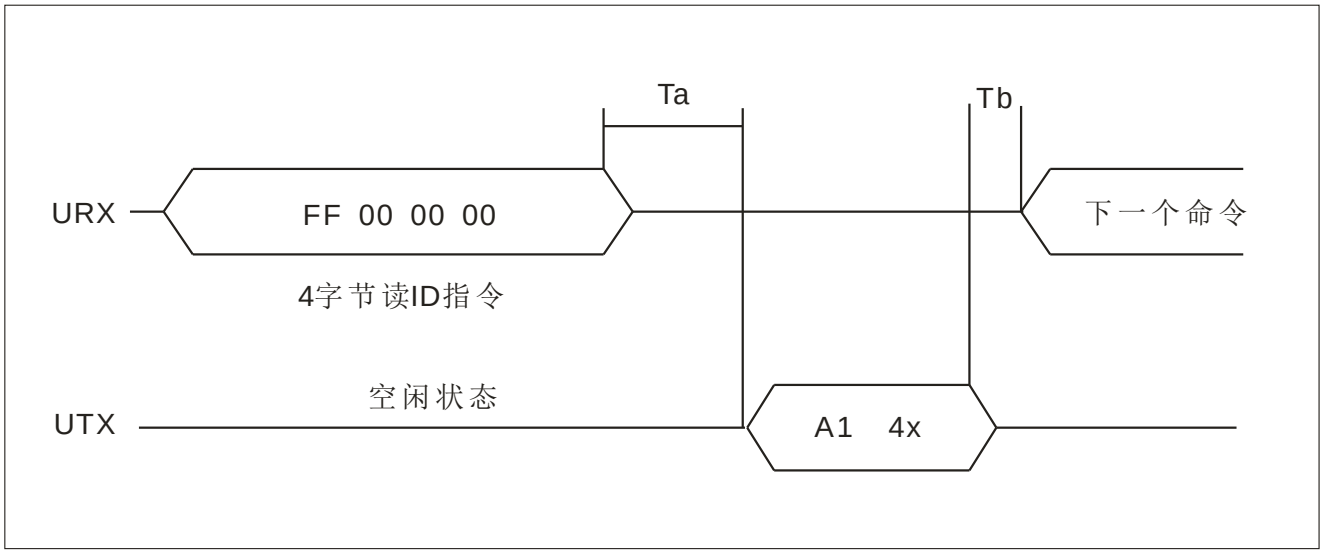
Fig.10-13: Reset



10.2.4 Read Device ID

After read ID instruction is transmitted, the device directly sends 2 bytes ID value via UTX pin.

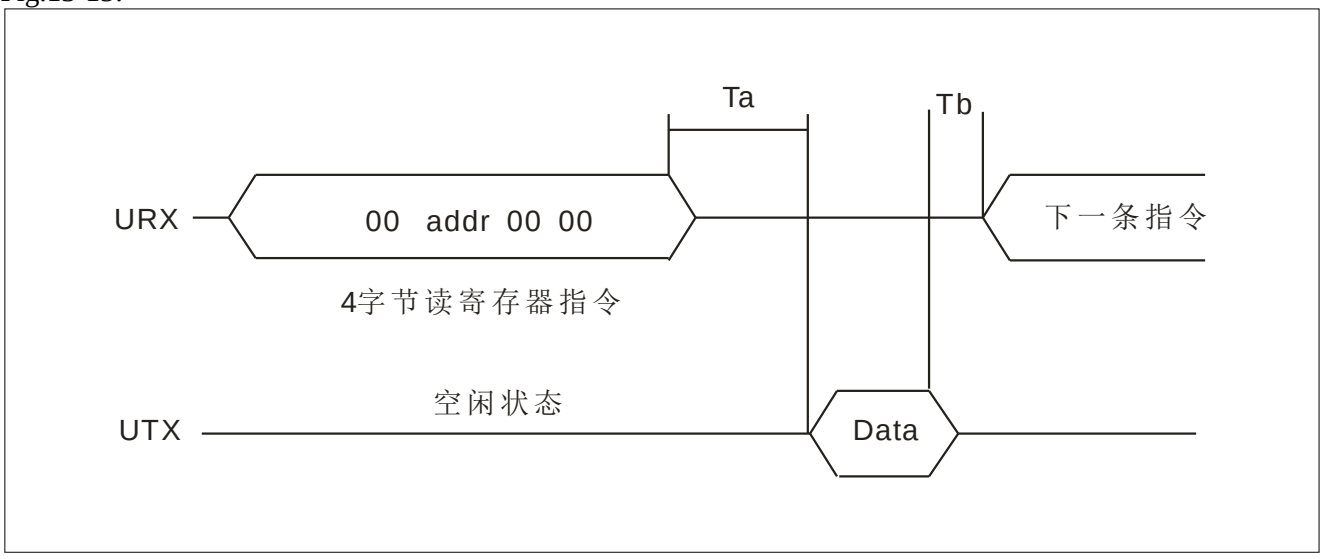
Fig.13-14: Read ID



10.2.5 Read Register

After the read register instruction is received, the device returns the one byte register value. Addr value is 0~63. Register with address >63 must not be read.

Fig.13-15:

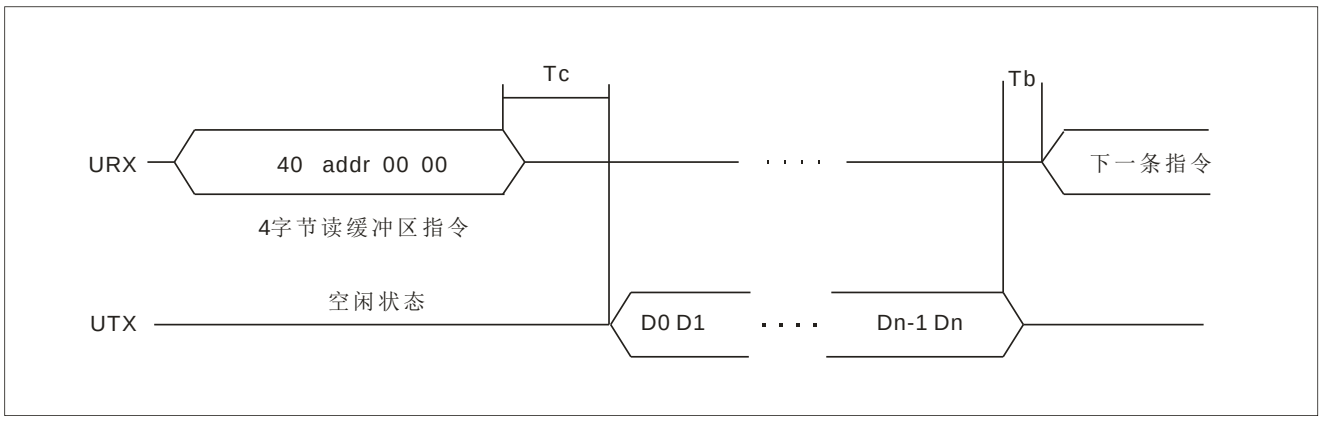


10.2.6 Read Buffer(Register Set)

After the read buffer instruction is received, the device returns the buffer length data(except URXB) corresponding to addr(0~63).

URXB is the receive buffer of EUART module, the length of returned data is 2 bytes information character+Actual length of the received data. Data length is contained in the information character. For more details, see 7.6.1 URXB Receive Buffer Structure. When user reads, the remainder data must be received according to the length of information character.

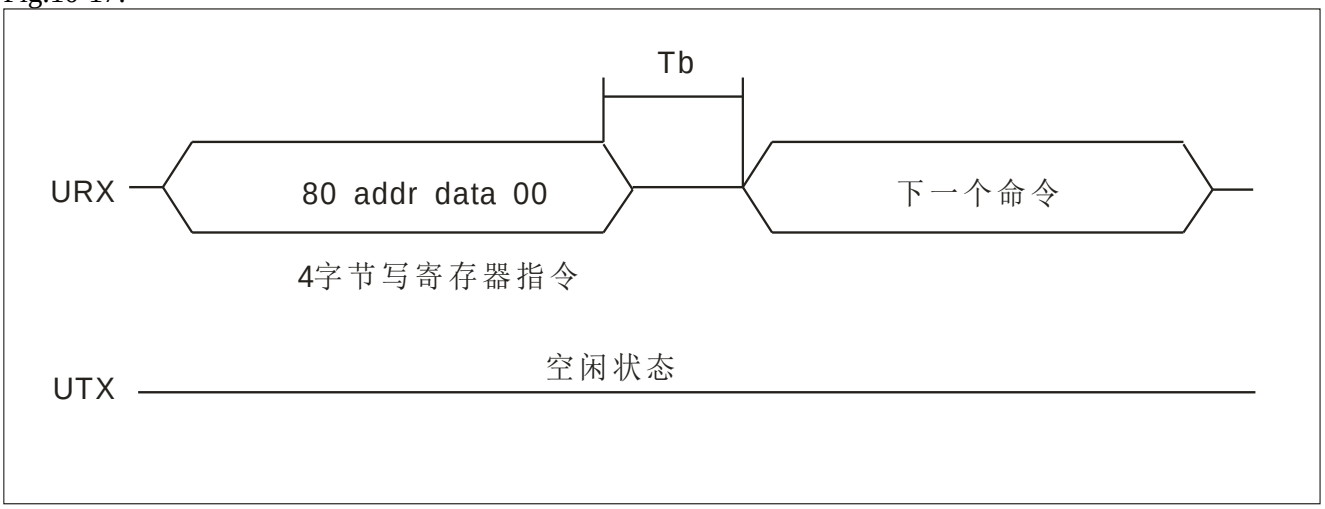
Fig.13-16:



10.2.7 Write Register

Write data in the register with address as addr(0~127), and the register with address 64-127 only has write instruction.

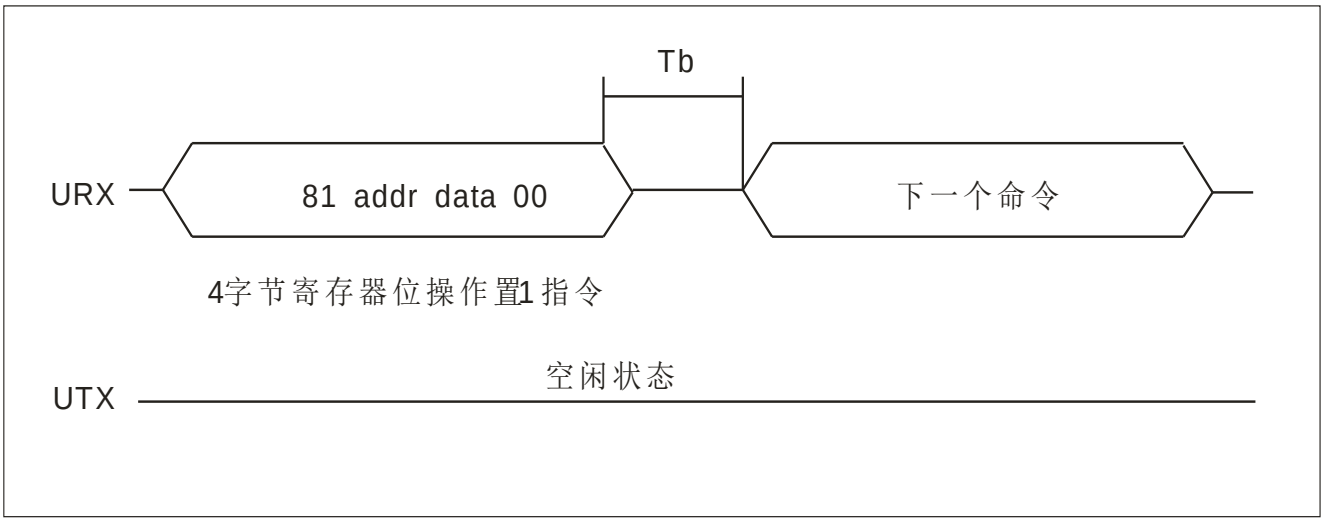
Fig.10-17:



10.2.8 Write Register(Bit Operation Set ON)

Set ON some bits of the register with address as addr(0~63). (data “1” bit is set ON, keep the “0” bit unchanged), and the register 64-127 has no bit operation instruction.

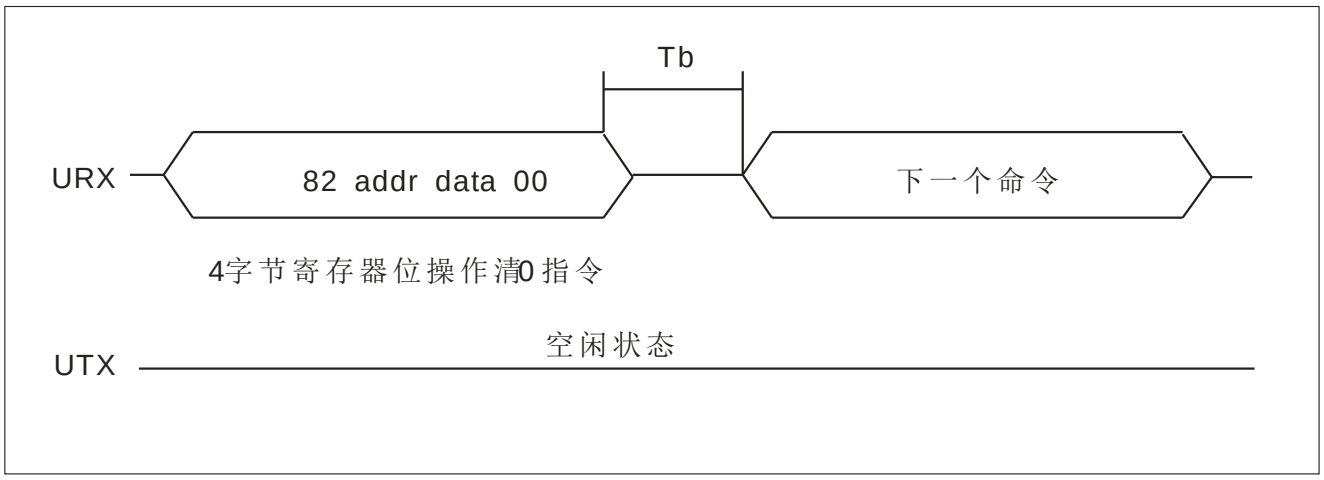
Fig.10-18



10.2.9 Write Register(Bit Operation Clear)

Set Clear some bits of the register with address as addr(0~63). (data “1” bit is set ON, keep the “0” bit unchanged), and the register 64-127 has no bit operation instruction.

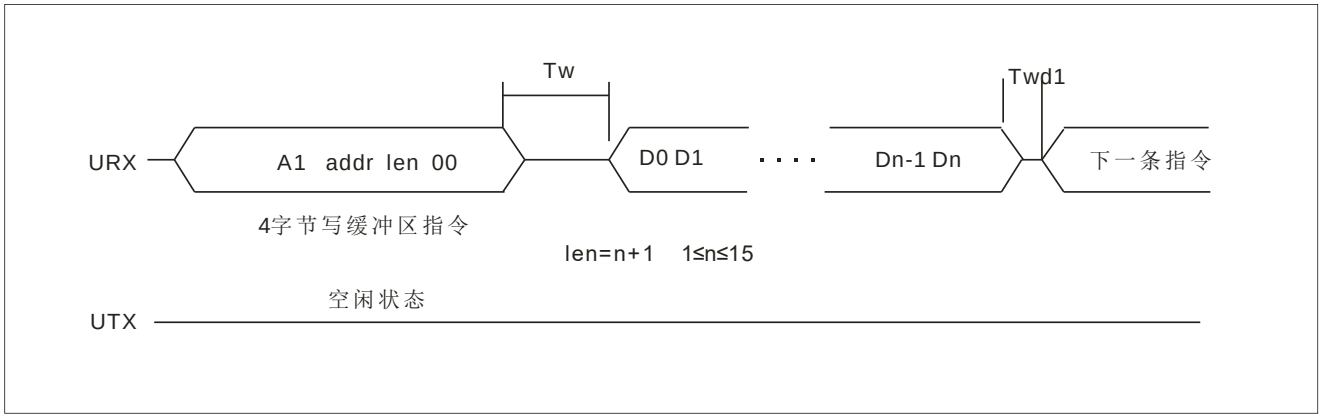
Fig.10-19:



10.2.10 Write Buffer(Buffer Length≤16 Bytes)

Write ≤16 bytes length buffer, after 4 bytes instruction is sent, delay Tw for the continuous transmission of Len data(D0~Dn) to the device. Len must be equal to the buffer length. This command is invalid for the long buffer over 16 bytes.

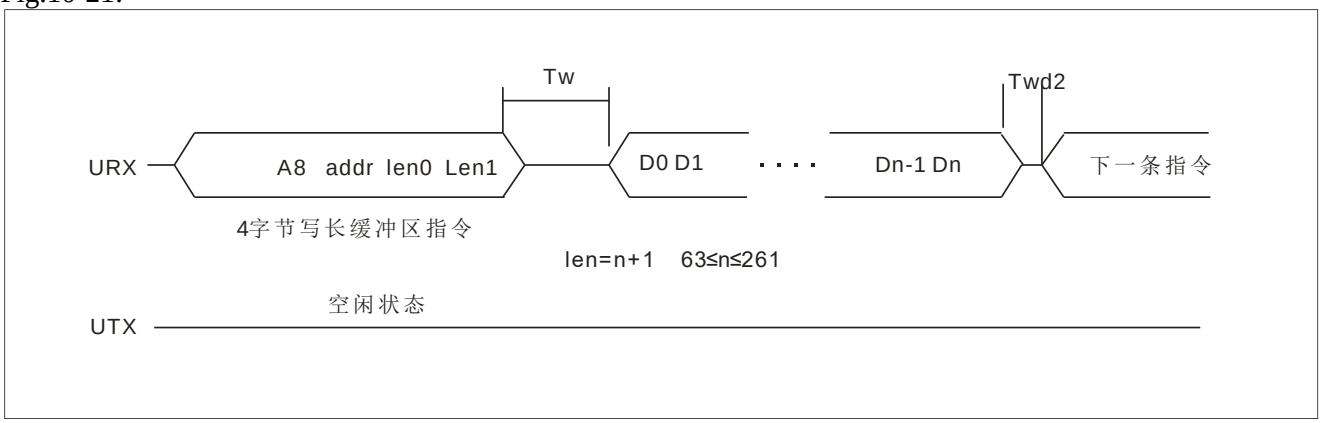
Fig.10-20:



10.2.11 Write Buffer(Buffer Length>16 Bytes)

After 4 bytes instruction is sent, delay Tw for the continuous transmission of Len1:Len0 data(D0~Dn) to the device. Len1:Len0 length≤Actual buffer length. Len1 is the length higher byte, and Len0 is the length lower byte. This command is only valid for the following 5 buffers: CFTXB(256) / UTXB(262) / LIDCON(64) / IDIFRCON(256) / HDFILTER(256).

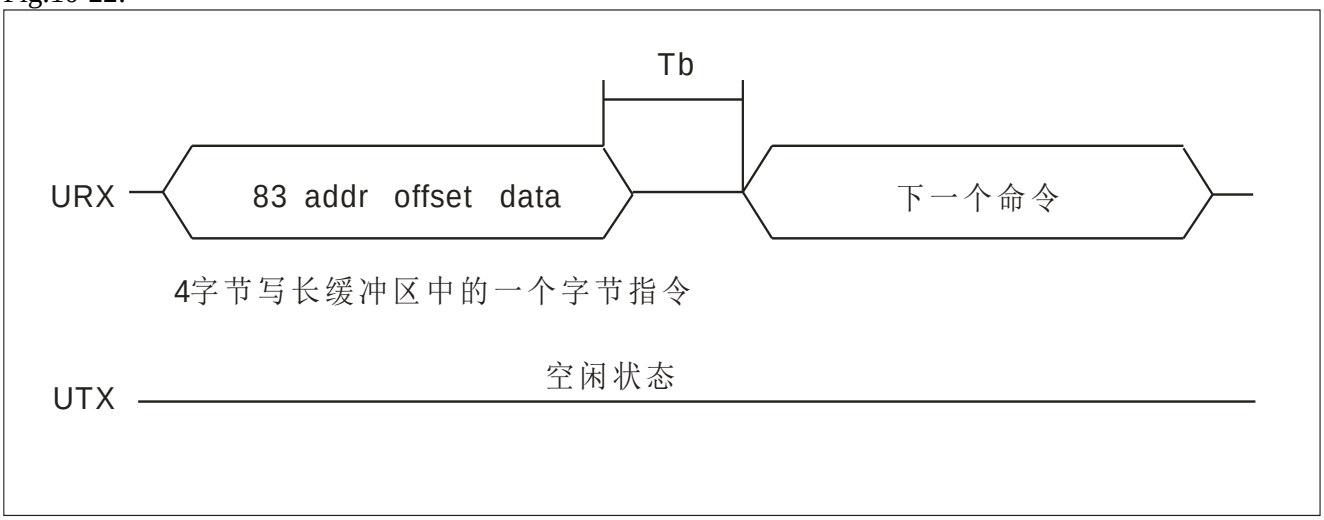
Fig.10-21:



10.2.12 Write Buffer One Byte(0x83)

Rewrite one byte value of long buffer with map address as addr to data, the offset of rewritten byte is offset(Start from 0, this command is only valid for these 3 buffers, LIDCON / IDIFRCON / HDFILTER)

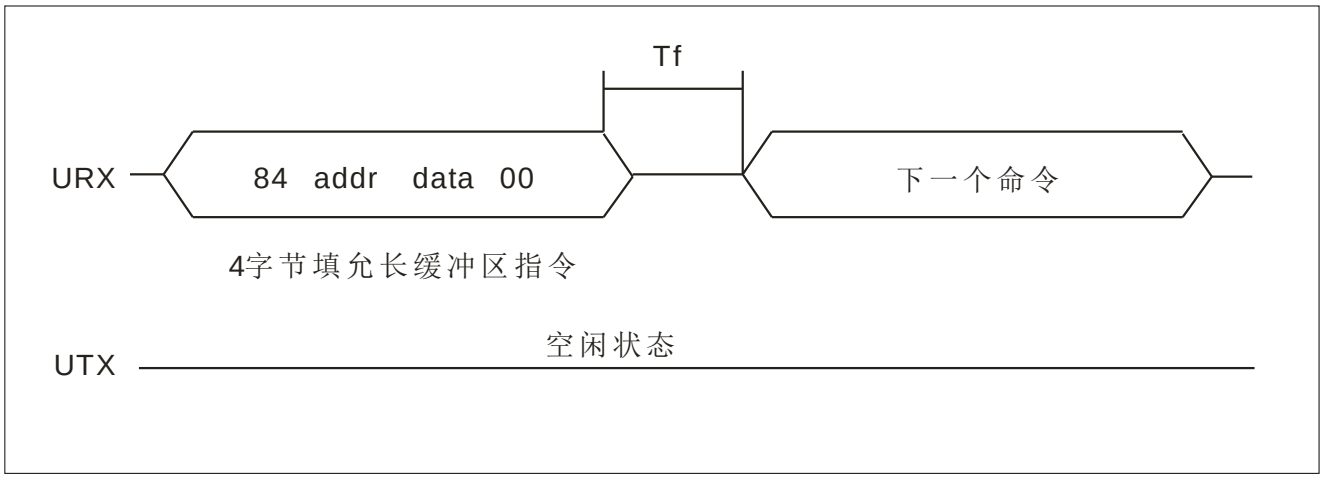
Fig.10-22:



10.2.13 Fill Long Buffer(0x84)

Fill long buffer with map address as addr with all data, this command is only valid for these 3 buffers, LIDCON / IDIFRCON / HDFILTER.

Fig.10-23:



10.2.14 Table of Time Parameters

Code	Description	Minimum	Typical	Maximum	Unit
Ta	In reading, time delay of device delivering back data	0	-	10	us
Tb	Space between two instructions	0	10	-	us
Tc	Read buffer(≤16 bytes), time delay of device delivering back data	0	-	10	us
	Read long buffer(>16 bytes)	2	-	50	us
Tf	Long buffer instruction delay	0	30	-	us
Trst	Delay after device reset	15ms	20ms	-	ms
Tw	In writing long buffer the delay between instruction and data	10	10	-	us
Twd1	In writing buffer the required instruction delay	10	10	-	us
Twd2	In writing long buffer the required instruction delay	50	50	-	us

Note: The above time is not strictly tested.

Chapter 11 Reset, Crystal Oscillator, IO Port, Sleep, Voltage Measurement

11.1 Device Reset

The device builds in power-on reset circuit, and the power-on reset time is 200MS.When the device uses /RST pin or uses reset command reset, user cannot transmit instruction to the device until the minimum delay of 20ms.
The device reset dircuit is as shown in 3.1.1 Main Device Diagram. R6 ≥ 240Ω will limit any current to flow in /RST, in order to avoid any /RST pin damage due to the electrostaticdischarge(ESD) or electrical overstress(EOS).
/RST pin provides the reset method using the external hardware trigger. Pull down this pin to generate the reset signal. The device has a noise filter in the internal route of /RST reset, which can be used to detect and filter the small interference pulse.

11.2 Crystal Oscillator

ET7190 device uses 16M crystal oscillator, as shown in 3.1.1 Main Device Diagram, the capacitance generally uses 15PF. In PCB design, crystal oscillator must be close to OSCI/OSCO pin as much as possible.

11.3 IO Port

Maximum 6 ports are available for use according to the enable functions of device. Separately configured as input or output. Used as universal I/O pin. The port has 4 operation registers, which are:

- PDIR Register: Set PORT5…PROT0 port pin direction.
- PORT Register: Port read/write
- UPR Register: Pull-up enable when port is configured as input port.
- ODC Register: Open-drain output enable when port is configured as output.

In operation of IO port, firstly write the defined port direction of PDIR register, then separately set UPR/ODC as input/output pull-up or open-drain(if required).
Read PORT value as pin level(output/input), write PORT has no effect on the input pin.

11.4 Sleep and Wake-up

Enable user control device to Sleep Mode, after entering Sleep Mode, the device clock source is closed, if IO port has no source current, the device current consumption will be reduced to the minimum(<50uA).
The device will be woken from Sleep Mode if:

- Generate any separately enabled WJE/WCE/WUE external bus activity.
- Device /RST pin reset
- When user interface sends any command to the device.

11.4.1 Enter Sleep Mode

When the device enters Sleep Mode, firstly close the unused module, then the device enters Sleep Mode.

- When writing(ON=1) in SLEEP register, the device immediately enters Sleep Mode. WJE/WCE/WUE bit in SLEEP register enables the wake-up in J1850/CAN/EUART bus activity. When all WJE/WCE/WUE are 0, all bus activities will not wake up the device, but active wake-up via user interface is only allowed. WJE/WCE/WUE can be set ON, enable multiple modules of simultaneous wake-up.
- If the device has no operation when SLEEP(ON=0) instruction is written.

- When the device enters Sleep Mode:

1. CAN module(if operated as required), firstsly enters Disable Mode(OPMODE=001), ensure the device entirely off. Incomplete bus transmit frame or receive frame may be generated if go to Sleep Mode when OPMODE=000 normal mode of operation, affecting the bus.

2. J1850 module must firstly write JCON0<ON>=0 to turn off the module, as long as WJE=1 when writing in SLEEP register, J1850 module can still wake up the device in off mode. If the device directly enters Sleep Mode when JCON0<ON>=1, incomplete bus transmit frame or receive frame may be generated.

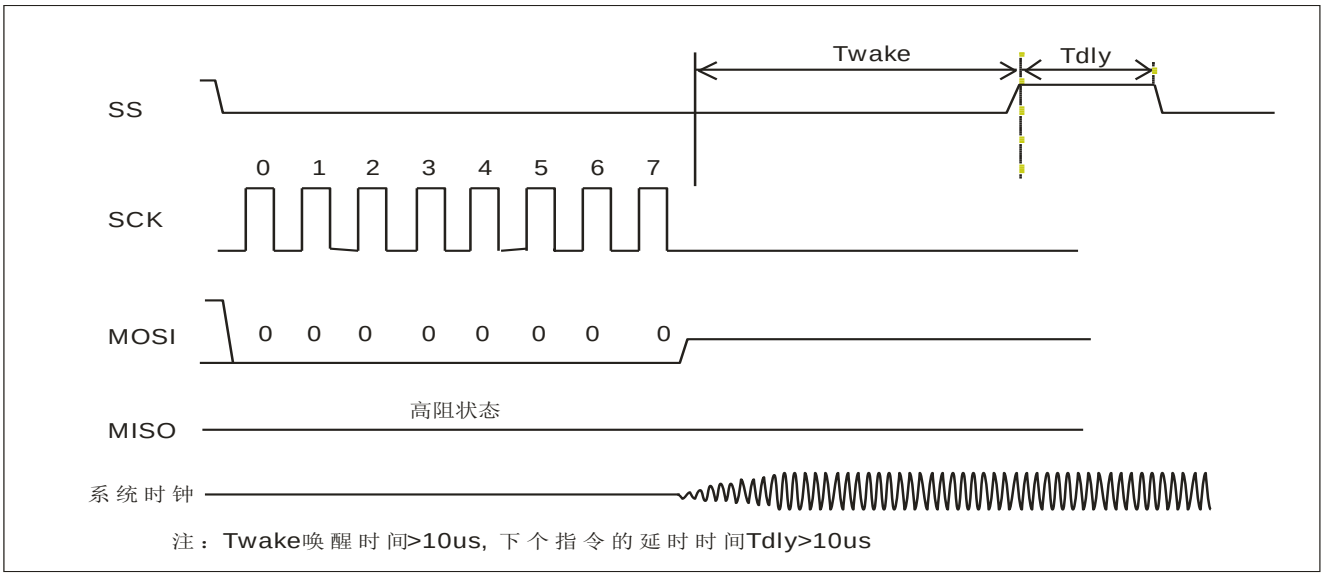
3. EUART module(in Frame Mode) must ensure data frame completely transmitted, transmit frame abortion will be generated if go to Sleep Mode in the process of transmit frame.

11.4.2 Device Wake-up

After the device is woken, if relevant module is off when entering Sleep Mode, relevant module must be started after wake-up, and make the device in normal mode of operation. The device can be woken via SPI/UART user interface at any time.

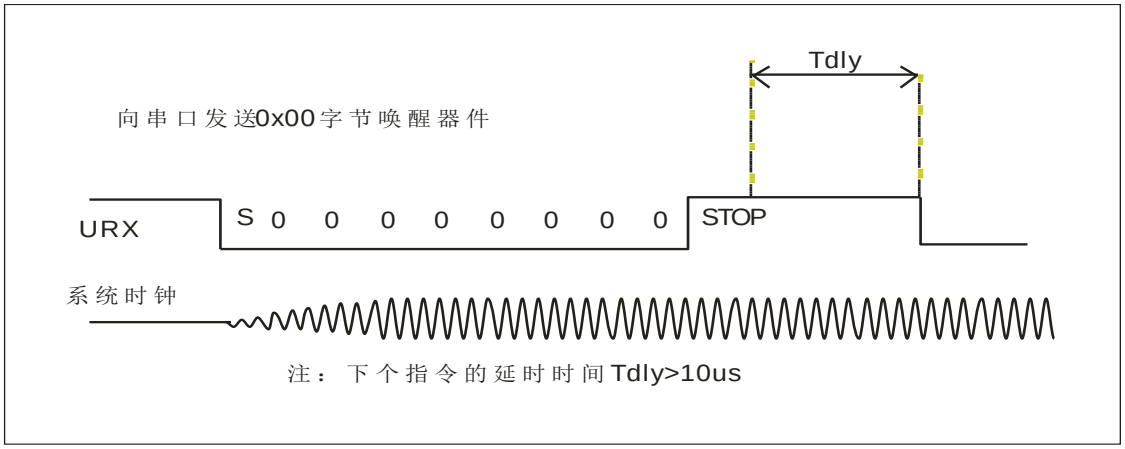
11.4.2.1 SPI User Interface Wake-up

When user uses SPI user interface to connect with the device, one 0x00 byte value can be sent to SPI interface to wake up the device. Wake-up timing is shown below: When the device receivs one byte from SPI, the device beings to wake up from Sleep Mode. After wake-up, INTF<WAF> interrupt flag is set ON, if interrupt is enabled, hardware interrupt will be generated.



11.4.2.2 UART User Interface Wake-up

When user uses UART user interface to connect with the device, one 0x00 byte value can be sent to UART interface to wake up the device. Wake-up timing is shown below(0x00 byte must be sent, UART wake-up condition is that URX pin>1bit time negative pulse receives instruction state in the first rising edge). After wake-up, INTF<WAF> interrupt flag is set ON, if interrupt is enabled, hardware interrupt will be generated.



11.4.2.3 CAN Module Wake-up

When CAN module enters Sleep Mode, firstly enter Disable Mode(OPMODE=001) or Configuration Mode(OPMODE=100), and enture the device entirely off. Incomplete bus transmit frame or receive frame may be generatd when entering Sleep Mode in OPMODE=000 normal mode of operation. When OPMODE=000 register bit WCE=1 is written, the device enters Sleep Mode and enable CAN bus to wake up the device.
After wake-up, WCE bit and intermediate flag INTF<WAF> bit are set ON. If interrupt is enabled, hardware interrupt will be generated.

When the module sleeps, the module will monitor if there is any activity in the receiving line. If the module is in Sleep Mode, the module will generate interrupt when bus activity is detected. Wake-up message activity will be lost due to the time delay caused by oscillator and device startup.

11.4.2.4 J1850 Module Wake-up

Before J1850 module sleeps, firstly write JCON0<ON>=0 to turn off J1850 module, as long as WJE=1 when writing in SLEEP register, the device enters Sleep Mode and enable J1850 module to wake up the device. When bus activity is found at the bus JRX pin, the device will be woken up.
After wake-up, WCE bit and intermediate flag INTF<WAF> bit are set ON. If interrupt is enabled, hardware interrupt will be generated.
When the module sleeps, the module will monitor if there is any activity in the receiving line. If the module is in Sleep Mode, the module will generate interrupt when bus activity is detected. Wake-up message activity will be lost due to the time delay caused by oscillator and device startup.

11.4.2.5 EUART Module Wake-up

When the device enters Sleep Mode, all clock sources of EUART module will be off and keep in logic 0 state. If the device enters Sleep Mode when EUART transmits or receives operation, this operation will be aborted, UART pin UTX1/2 or 3 is driven in default state.
If the device enables EUART bus wake-up(WUE=1 in Sleep), when start bit is detected at EUART receive(one of URX1/2/3, as set by UPINSEL)pin, the device will be woken from Sleep Mode. In this mode, WUE bit and intermediate flag INTF<WAF> bit are set ON. If interrupt is enabled, hardware interrupt will be generated. The module receives the normal EUART bus data until the first rising edge at URX1/2/3 pin.
When EUART starts, level change of protection circuit of /FT pin will also wake up the device.

Note: In writing WCE/WJE/WUE bit, it only represents command state as ON, automatically cleared when sleeping, if any activity of CAN/J1850/EUART bus wakes up the device, the corresponding WCE/WJE/WUE is reset ON. After wake-up, user can read SLEEP register value to determine which bus wakes up the device in the previous wake-up.

11.4.3 Special Application

External bus wake-up is triggered by the level change of bus input pin of the module, when CAN module or UART module CPINSEL/UPINSEL register selects the unused RX pin, the corresponding CRX/URX pin can be used as the external wake-up pin.

11.5 Voltage Measurement

Battery voltage measurement circuit is shown in 3.15 Diagram.
Equation of actual voltage calculation is:

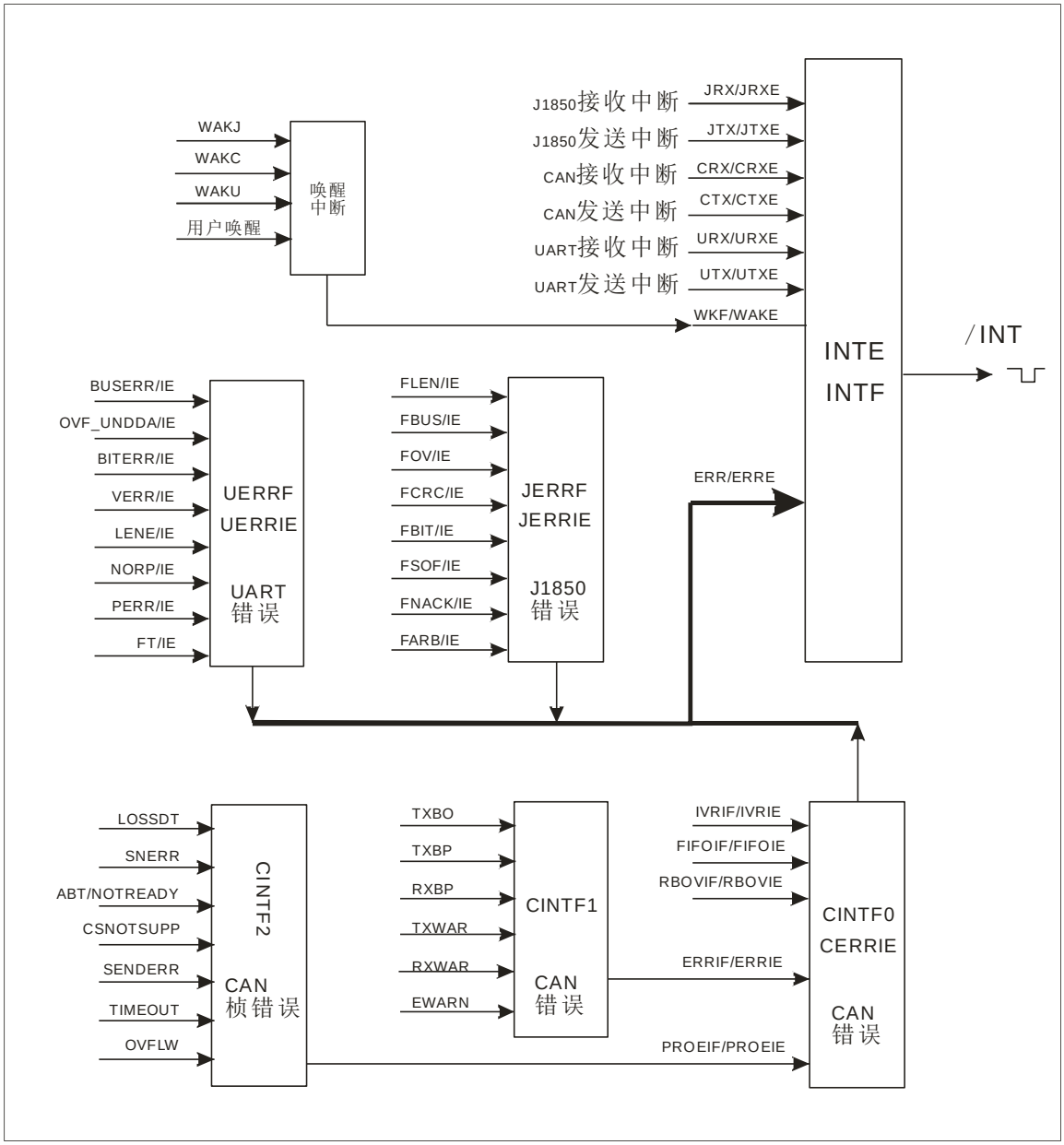
$$\text{Voltage} = \frac{(R12+R13)}{R13} \times \frac{3.3V}{0x400} \times (\text{ADCB Value})$$

In the process of reading voltage, firstly write any value in ADC register to start the voltage measurement 10-bit ADC, then read buffer ADCB value. ADCB value is 2 bytes, 10 bits are valid, maximum value is 0x3FF, ADCB value only changes once in case of AD conversion startup resulting from writing in ADC register.

Chapter 12 Interrupts

ET7190 has 8 main interrupt sources. INTE register contains the interrupt enable bit of enable each interrupt source. INTF register contains the interrupt flag bit of each interrupt source. When interrupt is generated, /INT pin will be pulled to low level by ET7190, and keep in low level state until clearing interrupt. Interrupt will not be cleared until the condition resulting in the corresponding interrupt is dismissed.
In which, INTF<ERR>/INTE<ERRE> error interrupt source is multi-level interrupt, and Bit INTF<ERR> will be set when the corresponding J1850/EUART/CAN module error flag occurs interrupt.

12.1 ET7190 Interrupt Management Example



12.2 Transmit Completion Interrupt

UTX/CTX/JTX in INTF register are EUART, CAN, J1850 transmit completion interrupt flag bit respectively, when the corresponding module completes the transmission, UTX/CTX/JTX bit is set ON. If the associated UTXE/CTXE/JTXE bit is set ON, the device will generate interrupt at /INT pin. When INTF register is read, UTX/CTX/JTX bit is automatically cleared.

12.3 Receive Interrupt

URX/CRX/JRX in INTF register are EUART, CAN, J1850 receive completion interrupt flag bit respectively, when the corresponding module completes the transmission, URX, CRX or JRX bit is set ON. If the associated UTXE/CTXE/JTXE bit in INTE register is set ON, the device will generate interrupt at /INT pin.

URX/CRX/JRX interrupt flag bit in INTF register is cleared by reading empty URXB, CRXB or JRXB receive buffer.

12.4 Error Interrupt

ERR bit in INTF register is the error interrupt flag bit, ERR bit is a global multi-level error interrupt flag. When ERR bit enables the interrupt error to dismiss, automatically cleared. INTE ERRE bit global error interrupt enable bit.

EUART manages error interrupt via two registers, UERRF/UERRIE. J1850 manages error interrupt via two registers, JEERF/JERRIE. CAN manages error interrupt via four registers, CINTF0/CINTF1/CINTF2/CERRIE.

12.5 Wake-up Interrupt

WAKF bit in INTF register is wake-up interrupt flag bit, when the devices wakes up from sleep, this bit is set ON. If INTE interrupt enables WAKE in register to be set ON, the device will generate interrupt at /INT pin.
If the device is woken by bus activity, the wake-up signal from which module can be determined by reading SLEEP register. WAKF flag bit is automatically cleared when INTF register is read.

Chapter 13 Summary of Register(Set)

13.1 Summary of Registers

Name	Map Addr.	Description	Bit ON Instruction	Bit Clear Instruction	Reset Value HEX
INTF	0x00	Interrupt Flag Register	×	×	00
INTE	0x01	Interrupt Enable Register	√	√	00
SLEEP	0x02	Sleep Control Register	√	×	00
-	0x03	Resevsed	-	-	-
JERRF	0x04	J1850 Error Register	×	√	00
JERRIE	0x05	J1850 Error Interrupt Flag Enable Register	√	√	00
IFR3B0LEN	0x06	IFR Type 3 Buffer 0 Data Length	×	√	00
IFR3B1LEN	0x07	IFR Type 3 Buffer 1 Data Length	×	√	00
JCON0	0x08	J1850 Control Register 0	√	√	02
JCON1	0x09	J1850 Control Register 1	√	√	7E
JSTAT	0x0A	J1850 State Register	√	×	00
ODC	0x0B	IO Open-drain	√	√	00
UPR	0x0C	IO Input Pull-up	√	√	00
PDIR	0x0D	IO Direction Control	√	√	3F
PORT	0x0E	IO Port Level	√	√	xx
-	0x0F	Reserved	-	-	-
CCTRL0	0x10	CAN Control Register 0, set CAN operation mode	×	×	44
CCTRL1	0x11	CAN Control Register 1	√	×	00
CCTRL2	0x12	CAN Control Register 2, control NSTB and SWEN pin	√	√	00
CPINSEL	0x13	CAN Select CAN1TX/RX or CAN2TX/RX Pin Set	√	√	00
CDNCNT	0x14	This register contains DeviceNet™ filter control bit	×	×	00
CINTF0	0x15	CAN Error Interrupt Flag Register 0	×	√	00
CINTF1	0x16	CAN Error Interrupt Flag Register 1	×	×	00
CINTE	0x17	CAN Error Interrupt Enable Register	√	√	00
CTERRCNT	0x18	CAN Transmit Error Counter	×	×	00
CRERRCNT	0x19	CAN Receive Error Counter	×	×	00
CCFG0	0x1A	CAN BaudRate Configuration Register 0	√	√	00
CCFG1	0x1B	CAN BaudRate Configuration Register 1	√	√	xx
CCFG2	0x1C	CAN BaudRate Configuration Register 2	√	√	xx
CFEN0	0x1D	CAN Receive Filter Enable Register 0	√	√	3F
CFEN1	0x1E	CAN Receive Filter Enable Register 1	√	√	00
CFMSKSEL0	0x1F	CAN Filter 3-0 Mask Select Register	√	√	xx
CFMSKSEL1	0x20	CAN Filter 7-4 Mask Select Register	√	√	xx
CFMSKSEL2	0x21	CAN Filter 11-8 Mask Select Register	√	√	xx
CFMSKSEL3	0x22	CAN Filter 15-12 Mask Select Register	√	√	xx
CTXPRI0	0x23	CAN Transmitter 3-0 Priority	√	√	00
CTXPRI1	0x24	CAN Transmitter 7-4 Priority	√	√	00

CTXREQ	0x25	CAN Message Transmit Request Register	√	×	00
CTXABT	0x26	CAN Message Abort Transmit Request Register	√	×	00
CRTREN	0x27	CAN Auto Remote Transmit Enable Register	√	√	00
CTXAUTOEN	0x28	After CAN writes in transmit buffer, automatically start the transmit register	√	√	FF
CPROCON	0x29	CAN Multi-frame Protocol Control	√	√	00
CINTF2	0x2A	CAN Multi-frame Protocol Transmit Error Flag Register	x	√	00
CCFDL	0x2B	CAN multi-frame transmission, data length of CFTXB buffer	√	√	xx
UFRMCON	0x2C	EUART Frame Mode Control Register	√	√	01
UPINSEL	0x2D	EUART Drive Pin Select	×	×	00
UFRMV	0x2E	EUART Frame Check Control Register	√	√	00
UFLENCON	0x2F	EUART Frame Length Control Register	√	√	00
UFLAND	0x30	“And Value” in EUART Frame Length Setting	√	√	FF
UFTXSTAT	0x31	State Register in EUART Frame Transmission	√	×	20
INITLT	0x32	Low Level Time of ISO14230 Fast Initialization	√	√	xx
INITHT	0x33	High Level Time of ISO14230 Fast Initialization	√	√	xx
INITADDR	0x34	5BUAD Slow Initialization and Trigger Address	√	√	xx
UMODE	0x35	EUART Control Register	√	√	xx
USTAT0	0x36	EUART State Register 0	√	√	10
USTAT1	0x37	EUART State Register 1	√	√	01
UTRXREG	0x38	EUART Single Byte Transmit/Receive Register	×	×	00
UBITADJ	0x39	Time Fine-tuning of Fast Initialization and Slow Initialization	√	√	00
UERRF	0x3A	EUART Error Register	×	√	00
UERRIE	0x3B	EUART Error Interrupt Enable Register	√	√	00
UPRICON	0x3C	J1708 Priority Control	√	√	00
-	0x3D	Reserved	-	-	-
-	0x3E	Reserved	-	-	-
-	0x3F	Reserved	-	-	-

The following register addresses are within the range of 0x40-0x7F, only write register instruction is available, bit operation clear and bit ON are not supported. No read instruction is read.

Name	Map Addr.	Description	Bit ON Instruction	Bit Clear Instruction	Reset Value HEX
ABAUD	0x40	EUART Enable BaudRate Measurement	-	-	0
UTXBRK	0x41	EUART Transmit Sync Break,	-	-	0
UTXP1T	0x42	Interbyte Space Time in EUART Frame Transmission	-	-	5
-	0x43	Reserved	-	-	-
UFREET	0x44	EUART Bus Release Time.	-	-	55
URXP1T	0x45	Allowed Maximum Interbyte Space in EUART Frame Reception	-	-	25

UTXVT0	0x46	Maximum Time of Receive Check Byte in EUART Frame Transmission	-	-	20
URXVT1	0x48	Time Delay of Transmit Check Byte in EUART Frame Reception	-	-	2
YZZ0	0x50	J1850 Corresponding to YZZCON Buffer Byte 0	-	-	00
YZZ1	0x51	J1850 Corresponding to YZZCON Buffer Byte 1	-	-	00
YZZ2	0x52	J1850 Corresponding to YZZCON Buffer Byte 2	-	-	00
YZZ3	0x53	J1850 Corresponding to YZZCON Buffer Byte 3	-	-	00
YZZ4	0x54	J1850 Corresponding to YZZCON Buffer Byte 4	-	-	00
YZZ5	0x55	J1850 Corresponding to YZZCON Buffer Byte 5	-	-	00
YZZ6	0x56	J1850 Corresponding to YZZCON Buffer Byte 6	-	-	00
YZZ7	0x57	J1850 Corresponding to YZZCON Buffer Byte 7	-	-	00
CN_BS	0x60	ISO15765 Time_N_BS =xx * 5ms	-	-	xx
CN_ST	0x61	ISO15765 Time_N_ST =xx * 100us	-	-	xx
CN_BR	0x62	ISO15765 Time_N_BR =xx * 100us	-	-	xx
CN_CR	0x63	ISO15765 Time_N_CR =xx * 5ms	-	-	xx
CTP_T1	0x64	TP2.0 T1= xx * 5ms Time-out used by this Control mode for received telegrams	-	-	xx
CTP_T3	0x65	TP2.0 T3= xx * 100us minimum time when sending between consecutive telegrams	-	-	xx
CTP_TE	0x66	TP2.0 TE = xx * 5ms Time-out for connection structure waiting for response	-	-	xx
CTP_T3A	0x67	TP2.0 T3 = xx * 1000us minimum time when sending between consecutive telegrams	-	-	xx
CJ1939_T2	0x68	J1939 T2= xx * 5ms Maximum waiting DT time when receiving multiple frames	-	-	xx
CJ1939_T3	0x69	J1939 T3= xx * 5ms Maximum waiting receiver CTS time when transmitting multiple frames	-	-	xx
CJ1939_TS	0x6A	J1939 TS= xx * 1ms Time slot of continuous transmission of DT	-	-	xx
CJ1939_TR	0x6B	J1939 TR= xx * 1m Delay of requiring the transmission of respond frame CTS/EOM when receiving multiple frames	-	-	xx
			-	-	
CTP_Tw	0x6F	TP2.0 Tw = xx * 5ms If Receiver is not ready, insert a delay of T_wait before sending the next Data telegram.	-	-	xx
ADC	0x70	Start AD conversion, do voltage measurement	-	-	xx

13.2 Summary of Register Set(Buffer)

In this data manual, **register set** is identical to **data buffer**, and the map address of buffer is unified, which are data storage devices for the storage of special function with over two bytes. Because some buffers control the operation of module, so we call it as the control register set. Some are simply used to store, send or receive data, so we name it as data buffer. Read or write operation is done via buffer instruction. One byte special function register is given in the said 13.1.

Register Set Name (Buffer)	Addr.	Number of Length Byte	Description	Remark/Reset Value
YZZCON	0x00	8	3 Byte Header IFR Control Register Set	00
IDIFRCON	0x01	256	IFR Control Register Set in Single Header or One Byte Header of Consilidated Mode, write and use long buffer write command CMD_WR_LB	00
HDFILTER	0x02	256	Receive Filter Control Register Set, write and use long buffer command CMD_WR_LB	07
IDBUFF	0x03	8	Responded ID Buffer in J1850 IFR type 1/2 Mode	xx
IFR3B0	0x04	10	Responded Data Buffer 0 in J1850 IFR Type 3 Mode	xx
IFR3B1	0x05	10	1 Responded Data Buffer 1 in J1850 IFR Type 3 Mode	xx
JTXB	0x06	16	J1850 Transmit Buffer	xx
DRVUBRG	0x07	2	UART User Interface BaudRate Scaler	Determined by B1B0
ADCB	0x08	2	ADC value, 10 bits	Read Only
CRXMASK0	0x09	4	CAN Receive Filter Mask Standard Identifier Register 0	xx
CRXMASK1	0x0A	4	CAN Receive Filter Mask Standard Identifier Register 1	xx
CRXMASK2	0x0B	4	CAN Receive Filter Mask Standard Identifier Register 2	xx
CRXF0	0x0C	4	CAN Receive Filter Identifier Register 0	xx
CRXF1	0x0D	4	CAN Receive Filter Identifier Register 1	xx
CRXF2	0x0E	4	CAN Receive Filter Identifier Register 2	xx
CRXF3	0x0F	4	CAN Receive Filter Identifier Register 3	xx
CRXF4	0x10	4	CAN Receive Filter Identifier Register 4	xx
CRXF5	0x11	4	CAN Receive Filter Identifier Register 5	xx
CRXF6	0x12	4	CAN Receive Filter Identifier Register 6	xx
CRXF7	0x13	4	CAN Receive Filter Identifier Register 7	xx
CRXF8	0x14	4	CAN Receive Filter Identifier Register 8	xx
CRXF9	0x15	4	CAN Receive Filter Identifier Register 9	xx
CRXF10	0x16	4	CAN Receive Filter Identifier Register 10	xx
CRXF11	0x17	4	CAN Receive Filter Identifier Register 11	xx
CRXF12	0x18	4	CAN Receive Filter Identifier Register 12	xx
CRXF13	0x19	4	CAN Receive Filter Identifier Register 13	xx
CRXF14	0x1A	4	CAN Receive Filter Identifier Register 14	xx
CRXF15	0x1B	4	CAN Receive Filter Identifier Register 15	xx
CFPNT0	0x1C	4	Filter **0-7** Buffer Pointer Register	0xFF FF FF FF

CFPNT1	0x1D	4	Filter **8-15** Buffer Pointer Register	0xFF FF FF FF
CTXB0	0x1E	16	CAN Transmit Buffer 0	xx
CTXB1	0x1F	16	CAN Transmit Buffer 1	xx
CTXB2	0X20	16	CAN Transmit Buffer 2	xx
CTXB3	0X21	16	CAN Transmit Buffer 3	xx
CTXB4	0X22	16	CAN Transmit Buffer 4	xx
CTXB5	0X23	16	CAN Transmit Buffer 5	xx
CTXB6	0X24	16	CAN Transmit Buffer 6	xx
CTXB7	0X25	16	CAN Transmit Buffer 7	xx
C15765TXFC	0X26	4	ISO15765 Multi-Frame Transmit Control Register Set	xx
C15765RXFC0	0X27	4	ISO15765 Receive Auto Respond FC Control Register Set 0	xx
C15765RXFC1	0X28	4	ISO15765 Receive Auto Respond FC Control Register Set 1	xx
CTP20CTRL	0X29	4	TP2.0 Multi-frame Transmit/Auto Respond Control	xx
UBRG	0x2A	2	Bus BaudRate Scaler(16-bit integer, 2 bytes)	xx
U5BAUDCON	0x2B	10	5 Baud Bus Initialization Control	xx
UPULSCON	0x2C	16	User-defined Initialization Pulse Control	xx
UPULSBIT	0x2D	16	16 User-defind Pulse Bit Setting	xx
SIMUCON	0x2E	12	ECU Simulation Control Special Use	xx
LIDCON	0x2F	64	LIN1.x LIN2.x ID Control, write use command CMD_WR_LB	xx
LACKB0	0x30	8	LIN Slaver Auto Respond, Data Transmit Buffer 0	xx
LACKB1	0x31	8	LIN Slaver Auto Respond, Data Transmit Buffer 1	xx
LACKB2	0x32	8	LIN Slaver Auto Respond, Data Transmit Buffer 2	xx
LACKB3	0x33	8	LIN Slaver Auto Respond, Data Transmit Buffer 3	xx
CJ1939CTRL	0x34	4	J1939 Multi-frame Transmit/Receive RTS/CTS Control	xx
-	0x35	-	Reserved	-
-	0x36	-	Reserved	-
-	0x37	-	Reserved	-
-	0x38	-	Reserved	-
-	0x39	-	Reserved	-
-	0x3A	-	Reserved	-
-	0x3B	-	Reserved	-
ERRB	0x3C	6	Error Flag Register Map Register Set	00
JRXB	0x3D	16*10	J1850 FIFO Receive Buffer, FIFO depth is 10 frames	xx
CRXB	0x3E	16*16	(Read Only) CAN Receive Buffer, FIFO depth is 16 frames*16bytes	xx
CFTXB		256	(Write Only) Multi-frame Transmit Buffer, length is 256 bytes, write use command CMD_WR_LB	xx
UTXB	0x3F	262	(Read Only) UART Message Transmit Buffer, write use command CMD_WR_LB	xx
URXB		262*2	(Read Only) FIFO Message Receive Buffer, FIFO depth is 2 frames	xx

Chapter 14 Electrical Characteristics

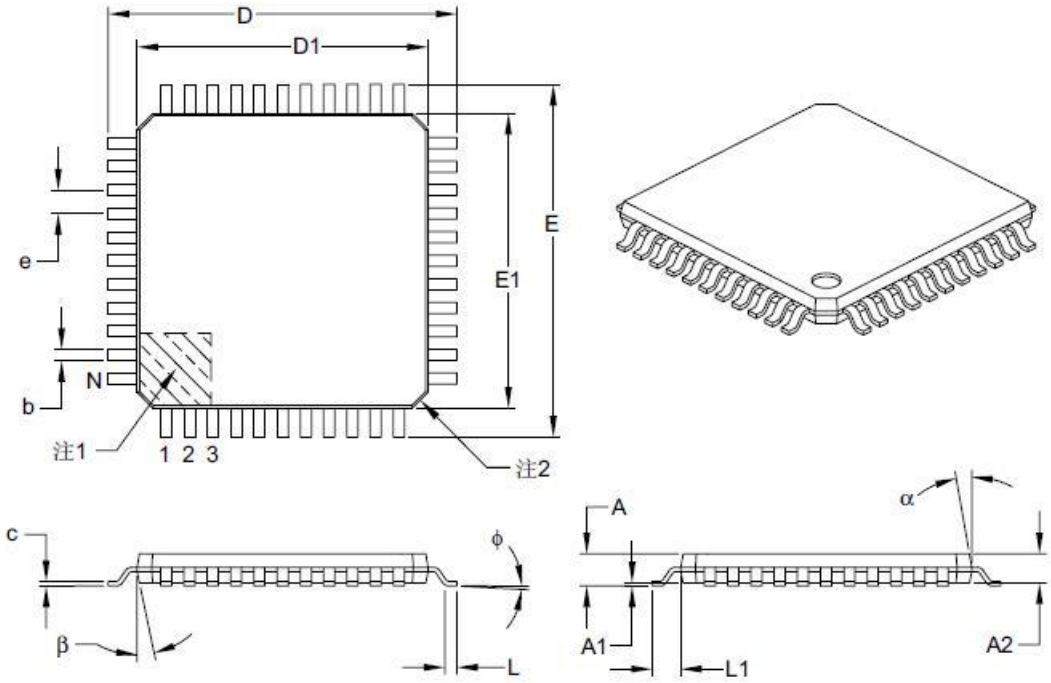
Table 14-1:

Ambient Temperature	-40°C - +85°C
Storage Temperature	-65°C - +150°C
VDD Pin to VSS Voltage	-0.3V - +4.0V(Absolute Maximum)
VDD Operation Voltage Range	3.0 V - 3.6V
VDD Rated Operation Current	45mA - 65mA (Excluding PORT IO Current)
Maximum Output Sink Current at any PORT Pin	4mA
Maximum Output Source Current at any PORT Pin	4mA
LED Pin Maximum Output Current	4mA

If the device operates under the maximum rated value for a long time, its reliability may be affected. We recommend not use this device under or above the maximum rated value as indicated in this standard.

Chapter 15 Packaging Information

15.1 44-Pins TQFP

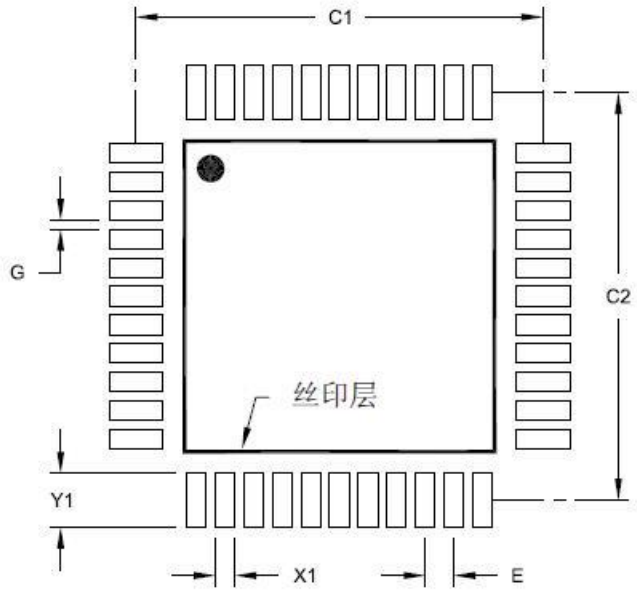


	单位	毫米		
	尺寸范围	最小	正常	最大
引脚数	N	44		
引脚间距	e	0.80 BSC		
总高度	A	–	–	1.20
塑模封装厚度	A2	0.95	1.00	1.05
悬空间隙	A1	0.05	–	0.15
底脚长度	L	0.45	0.60	0.75
引脚投影长度	L1	1.00 REF		
底脚倾斜角	ϕ	0°	3.5°	7°
总宽度	E	12.00 BSC		
总长度	D	12.00 BSC		
塑模封装宽度	E1	10.00 BSC		
塑模封装长度	D1	10.00 BSC		
引脚厚度	c	0.09	–	0.20
引脚宽度	b	0.30	0.37	0.45
塑模顶部锥度	α	11°	12°	13°
塑模底部锥度	β	11°	12°	13°

Note: 1. Pin 1 is located within the hatched area. 2. The slope at the chamfer is optional, but the size may vary.
3. BSC: Basic Dimension, theoretical precise value. REF: Reference Dimension

15.2 Recommended Pad Layout

Fig.15-2:



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.80 BSC	
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width	X1			0.55
Contact Pad Length	Y1			1.50
Distance Between Pads	G	0.25		