Display - LIN Network Specification

Last Update:
7/8/2016

Prepared by:

Accelerated Systems Inc.
60 Northland Road
Waterloo, Ontario. N2V 2B8

Contents

1 Introduction

The LIN protocol specifies that there are 64 total frame identifiers. Two of these (IDs 62 and 63) are reserved for future protocol enhancements. Two more (IDs 60 and 61) are reserved for diagnostic and configuration data. This leaves 60 (IDs 0 to 59) for signal carrying frames.

Baud Rate: 19200 bps
LIN Checksum: Classic
Frame Data Length: Always 8 bytes. Unused bytes filled with 0.

1.1 Purpose

To define the LIN communications for the Display LIN node.

1.2 Scope

Define scheduled frames for normal operation.

Define frames for reading and writing to the extended Display objects.

1.3 Definitions, acronyms, and abbreviations

1.4 References

a) LIN Protocol Specification 2.2 (www.lin-subbus.org)

2 ASi Proprietary Protocol Extensions

Since this network is developed for a closed system specific to ASi eBlox vehicles there are 3 types of messages defined.

1) Scheduled 8-byte LIN responses predefined by ASi. These bytes are used for the real time requirements for a particular type of eBlox peripheral and summarized in Table 1

2) Secondary read/write mechanism to support addressable peripheral data objects.
The Object dictionary is summarized in Table 2.
The read/write addressing mechanism is described in Object Read/Write Frames (IDs 0x2C and 0x34)

3) Proprietary field upgradable boot loading protocol. A separate specification is available from ASi.

Table 1 Summary of Display LIN Frame IDs

Frame Identifier	Frame ID With Parity	Publisher Node	Subscriber Node(s)	Data Contained in Packet
0x1C	0x9C	Display	BAC	Display Firmware Revision as 16-bit data
0x1D	0xDD	POD	Display	POD Firmware Revision as 16-bit data
0x1E	0x5E	BMS	Display	BMS Firmware Revision as 16-bit data
0x1F	0x1F	BAC	Display	1. Display Configuration as 16-bit data (Reserved Display Parameter) 2. BAC Firmware Revision as 16-bit data 3. Features bit vector as 16-bit data
0x21	0x61	BAC	Display	1. Battery compensated state of charge (CSOC) in % as 8-bit data 2. Vehicle speed in km/Hr (VELO) in km/Hr q8 (256=1kmhr) as 16-bit data 3. Absolute battery power in % (0 to +100) as 8-bit data 4. Mode (assist level) 5. Headlight status (HDLT_STS) as 8-bit data 6. Range in km (RANGE) for 100% state of charge. 7. General command (COMMAND1) 8-bit data
0x22	0xE2	Pod	Display	1. State of Up button (1-bit data) 2. State of Mode button (1-bit data) 3. State of Down button (1-bit data) 4. Last assist level (used at startup to set assist level to last level used) 5. Headlight/Tail-Light/Backlight command 6. Walk or boost mode enable 7. Trip or Odometer select 8. LIN master communications disable
0x23	0xA3	BAC	Display	1. Fault code (FAULT1) as 16-bit data 2. Fault code (FAULT2) as 16-bit data 3. Fault code (WARNING) as 16-bit data 4. Trip odometer in km (BAC_TRIP)

Frame Identifier	Frame ID With Parity	Publisher Node	Subscriber Node(s)	Data Contained in Packet
0x24	0x64	Pod	Display	1. Vehicle odometer in km (ODO) in kmx100 as 16-bit data 2. Vehicle trip in km (TRIP) in kmx100 as 16-bit data 3. Vehicle average speed of whole trip in km/Hr (AVG_TRIP_VEL) in km 4. Select advanced LCD screen, displaying such things as Node firmware revs, wheel size, max speed setting. 5. Maximum vehicle speed in km/h (user selected from POD) 6. Wheel diameter in inches (user selected from POD)
0x25	0x25	BAC	Display	1. BAC language setting (LANGUAGE) a. 0 = Dutch-km b. 1 = German-km c. 2 = ENGLISH-km d. 3 = English- km e. 4 = English-mile 2. Motor Power Watts Low byte 3. Motor Power Watts Hight byte 4. LED brightness (Sets the brightness of the LEDs on the POD)
0x2C	0xEC	BAC	Display	Display Modbus read or write request
0x34	0xB4	Display	BAC	Display Modbus read or write reply

Table 2 Display Extended Object Dictionary

Address	Name	Read/Write Status	Units	Scale Factor	Description
0	Firmware Revision	Read Only	-	1000	Display Firmware Revision
1	LIN Checksum Error Count (Low 16 bits)	Read Only	-	1	A count of the total number of checksum errors since powered on.
2	LIN Checksum Error Count (High 16 bits)	Read Only	-	1	

3 LIN IDs and Data

3.1 Standard Scheduled Frames

Frame identifier 0x1C (Answered by the Display)

Startup frame, allows a network manager to read the firmware revision of the display;

1. Display Firmware Revision as 16-bit data

These would be transmitted by the Display after receiving a frame identifier of 0x1C as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
DISP_FW_L	DISP_FW_H	0	0	0	0	0	0

Frame identifier 0x1D (Answered by the POD)

Startup frame, allows the display to read the firmware revision of the POD;

1. POD Firmware Revision as 16-bit data

These would be transmitted by the POD after receiving a frame identifier of 0x1D as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
POD_FW_L	POD_FW_H	0	0	0	0	0	0

Frame identifier 0x1E (Answered by the BMS)

Startup frame, allows the display to read the firmware revision of the BMS;

1. BMS Firmware Revision as 16-bit data

These would be transmitted by the BMS after receiving a frame identifier of 0x1E as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
BMS_FW_L	BMS_FW_H	0	0	0	0	0	0

Frame identifier 0x1F (Answered by the BAC drive)

Startup frame, the display needs to know the following information from the BAC drive;

1. Display Configuration as 16-bit data (Reserved Display Parameter)
2. BAC Firmware Revision as 16-bit data
3. Features bit vector as 16-bit data

These would be transmitted by the BAC Drive after receiving a frame identifier of 0x1F as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
DISP_CON_L	DISP_CON_H	BAC_REV_L	BAC_REV_H	FEATURES_L	FEATURES_H	0	0

Frame identifier 0x21 (Answered by the BAC drive)

In normal operation the display needs to know the following information about the drive.

1. Battery compensated state of charge (CSOC) in % as 8-bit data
2. Vehicle speed in km/Hr (VELO) in km/Hr q8 (256=1kmhr) as 16-bit data
3. Absolute battery power in % (0 to +100) as 8-bit data
4. Mode (assist level)
5. Headlight status (HDLT_STS) as 8-bit data
6. Range in km (RANGE) for 100% state of charge.
7. General command (COMMAND1) 8-bit data (ie go to boot loader etc..)

These would be transmitted by the BAC controller after receiving a frame identifier of 0x21 as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
CSOC	VELO_L	VELO_H	BAT_PWR	MODE	HDLT_STS	RANGE	COMMAND1

COMMAND1:

0x00: No Command
0x01: Pod Boot load
0x02: Display Boot load
0x03: Reset Trips
0x04: Reset Odometer
0x05: BMS Boot load

Frame identifier 0x22 (Answered by the Pod)

In normal operation the BAC and the display need to know the following information about the pod.

1. State of Up button (1-bit data)
2. State of Mode button (1-bit data)
3. State of Down button (1-bit data)
4. Last assist level (used at startup to set assist level to last level used)
5. Headlight/Tail-Light/Backlight command
6. Boost or Walk mode enable
7. Trip or Odometer select
8. LIN master communications disable

These would be transmitted by the Pod after recieveing a frame identifier of 0x22 as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
SW_STATE	LAST_ASSIST_LEVEL	LIGHTS_CMD	BST_WLK	0	0	TRIP_ODO	COMMAND2

SW_STATE:

Bit 0: Up button state (0 = open, 1 = closed)
Bit 1: Mode button state (0 = open, 1 = closed)
Bit 2: Down button state (0 = open, 1 = closed)

LAST_ASSIST_LEVEL:

Pod will store assist level in eeprom, so that on power up, the BAC can see what the assist level was before the last power off, and restore that assist level on power up.

LIGHTS_CMD:
- 0x00: All lights are OFF
- 0x01: All lights are ON

BST_WLK:
- Bit 0: Walk mode enable (0 = off, 1 = enabled)
- Bit 1: Boost mode enable (0 = off, 1 = enabled)

TRIP_ODO:
- 0x00: Display odometer distance
- 0x01: Display trip distance

COMMAND2:
- 0x01: BAC Master communication disable (so that POD LIN transceiver can disable power supplies)

Frame identifier 0x23 (Answered by the BAC drive)

The display needs the following information to be able to display faults and warnings.

1. Fault code (FAULT1) as 16-bit data
2. Fault code (FAULT2) as 16-bit data
3. Fault code (WARNING) as 16-bit data
4. Trip in km (BAC_TRIP) since BAC power-up (starting at 0)

These would be transmitted by the BAC controller after receiving a frame identifier of 0x23 as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
FAULT1_L	FAULT1_H	FAULT2_L	FAULT2_H	WARNING_L	WARNING_H	BAC_TRIP_L	BAC_TRIP_H

Frame identifier 0x24 (Answered by the POD)

In normal operation the display needs to know the following information from the POD.

1. Vehicle odometer in km (ODO) in kmx100 as 16-bit data
2. Vehicle trip in km (TRIP) in kmx100 as 16-bit data
3. Vehicle average speed of whole trip in km/Hr (AVG_TRIP_VEL) in km
4. Select advanced LCD screen, displaying such things as Node firmware revs, wheel size, max speed setting.
    a. SCR_SEL = 0: Display main screen
    b. SCR_SEL = 2: Display advanced screen
    c. SCR_SEL = 3: Display the ‘wheel diameter’ configuration screen
    d. SCR_SEL = 4: Display the ‘maximum speed’ configuration screen
5. Maximum vehicle speed in km/h (user selected from POD)
6. Wheel diameter in inches (user selected from POD)

These would be transmitted by the POD after receiving a frame identifier of 0x24 as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
ODO_L	ODO _H	TRIP_L	TRIP_H	AVG_TRIP_VEL	SCR_SEL	MAX_SPEED	WHEEL_DIAM

Frame identifier 0x25 (Answered by the BAC drive)

The display needs this frame to configure the displayed language and to display motor power.

1. BAC language setting (LANGUAGE)
    a. 0 = Dutch-km
    b. 1 = German-km
    c. 2 = ENGLISH-km
    d. 3 = English- km
    e. 4 = English-mile
2. Motor Power Watts Low byte
3. Motor Power Watts Hight byte
4. LED brightness (Sets the brightness of the LEDs on the POD)

These would be transmitted by the BAC controller after receiving a frame identifier of 0x25 as follows:

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
LANGUAGE	MOT_PWR_LO	MOT_PWR_HI	LED_BRIGHT	0	0	0	0

3.2 Object Read/Write Frames

Details of what information is contained at each address are described in Table 2 Display Extended Object Dictionary

Frame identifier 0x2C (Answered by the Display)

A message that allows a Modbus Master to send Modbus read and Modbus write requests to the display, using LIN packets.

Modbus Read

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
0x03	Address_H	Address_L	Num_H	Num_L	0	0	0

Address_H: Start Address, High Byte
Address_L: Start Address, Low Byte
Num_H: Number of registers, High Byte
Num_L: Number of registers, Low Byte

Modbus Write

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
0x10	Address_H	Address_L	Num_H	Num_L	ByteCount	Data_H	Data_L

Address_H: Start Address, High Byte
Address_L: Start Address, Low Byte
Num_H: Number of registers, High Byte
Num_L: Number of registers, Low Byte
ByteCount: Number of bytes
Data_H: Data to write, High Byte
Data_L: Data to write, Low Byte

Frame identifier 0x34 (Answered by the Display)

A message that allows the Display to respond to the Modbus read or write message received under Frame ID 0x2C.

Modbus Read

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
0x03	ByteCount	Data_H	Data_L	0	0	0	0

ByteCount: Number of data bytes in this message
Data_H: Data High Byte
Data_L: Data Low Byte

Modbus Write

Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
0x10	Address_H	Address_L	Num_H	Num_L	0	0	0

Address_H: Start Address, High Byte
Address_L: Start Address, Low Byte
Num_H: Number of registers, High Byte
Num_L: Number of registers, Low Byte