ELM630
LIN Monitor

Description

LIN or 'Local Interconnect Network' is a low cost and relatively simple networking system that is used predominantly in the automotive world. Recently, it has been gaining in popularity, with proposals to use it in major appliances as well. For more information, visit the LIN web site (http://www.lin-subbus.de/).

The ELM630 is a monitoring device designed for troubleshooting LIN bus systems. It is capable of continually monitoring a LIN network, translating the LIN messages to standard ASCII characters, and re-transmitting them to an RS232 system (personal computer or PDA) for display and possibly analysis. The baud rate measurements, data formatting, synchronizing, and checksum calculations are all done for you by the ELM630.

The LIN specification has recently been updated to revision 2.0, incorporating several improvements. The ELM630 has been updated to be compatible with this new specification, as well as the previous ones.

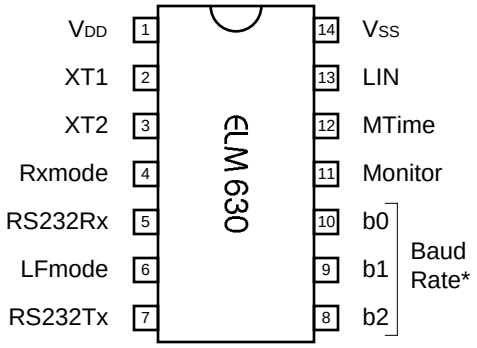
Applications

- LIN logic probes
- Diagnostic PC interfaces
- Instruction triggered (breakpoint) devices
- Educational or training devices

Features

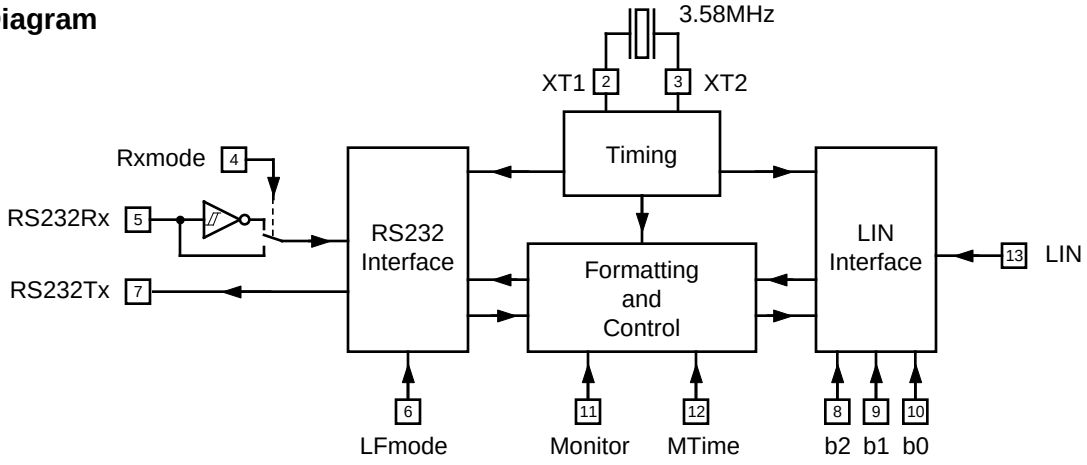
- Low power CMOS design
- Crystal-controlled for accuracy
- Standard ASCII character output
- Special power-on monitor mode
- Autobaud from 1200bps to 19200bps
- High speed (57600 baud) RS232 interface
- Works with LIN1.x and LIN2.0

Connection Diagram
PDIP and SOIC
(top view)



*see Table 1 on Page 3

Block Diagram



Pin Descriptions

V_{DD} (pin 1)

This pin is the positive supply pin, and should always be the most positive point in the circuit. Internal circuitry connected to this pin is used to provide power-on reset of the microprocessor, so an external reset signal is not required. Refer to the Electrical Characteristics section for further information.

XT1 (pin 2) and XT2 (pin 3)

A 3.579545MHz NTSC television colourburst crystal is connected between these two pins. Crystal loading capacitors (typically 27pF) will also be connected from each of these pins to the circuit common (Vss).

Rxmode (pin 4)

This input is used to control the inverting of the signal at the RS232Rx input (pin 5), allowing some flexibility as to how the RS232 is connected to it. Many experimenters will prefer to use only a single resistor between the RS232 interface and pin 5 to minimize costs. In that case, pin 4 need only be tied to common (V_{SS}), and the internal logic will invert the polarity of the signal for you.

Other users may prefer to use one of the standard (inverting) interface circuits such as the MAX232 series, or the SN75189/MC1489 type of IC. In these cases, the internal inversion is not required, and should be disabled by connecting the Rxmode input to a high level (V_{DD}). The interface IC can then be directly connected to the RS232Rx input (pin 5).

RS232Rx (pin 5)

A computer's RS232 transmit signal is connected to this pin, either through a resistor or through an interface IC, as discussed under Rxmode (pin 4). If a resistor is used, care should be taken in selecting its size in order to limit currents into the protecting diodes, while minimizing degradation of the input signal (due to the interaction between the resistor and stray circuit capacitance). A value of 47KΩ is typically chosen for this resistor.

LFmode (pin 6)

This input is used to select the default linefeed mode after a power-up or system reset. If it is at a low level, then the RS232 output from the ELM630 will be terminated by a carriage return character only, and no linefeed character. If it is at a high level, then all messages sent by the ELM630 will end with both a carriage return and a linefeed character. This behaviour can also be modified by issuing an AT L0 or AT L1 command (see the AT Commands section for more information).

RS232Tx (pin 7)

This is the RS232 transmit, or data output, pin. While at rest (no data is being sent), this pin will output a high level (V_{DD}), which is compatible with most interface ICs. It has sufficient current drive to allow interfacing using only a transistor if desired. See the Example Applications section for more details.

Baud Rate (pins 8, 9, and 10)

These pins are used to set the baud rate that is to be used when monitoring the LIN system. This is only the initial rate used after a power-up or reset, and does not affect rates that are subsequently set using the AT SB command. The following chart shows the possible input values and the resulting baud rate:

Pin Descriptions (continued)

b2	b1	b0	SB	Baud Rate
L	L	L	0	auto
L	L	H	1	1200
L	H	L	2	2400
L	H	H	3	4800
H	L	L	4	9600
H	L	H	5	19200
H	H	L	6	auto
H	H	H	7	auto

Table 1. Baud Rate Settings

Note that the SB column in this table refers to the value that would be used with an AT SB command to set the baud rate.

Monitor (pin 11)

This input is used to effectively force an AT MA command upon power-up, without any RS232 input being required. If this pin is found to be at a high level during a power-up (or reset), the ELM630 will display the ID string, but instead of issuing a prompt character it will immediately execute an AT MA command, reporting on all LIN bus activity.

MTime (pin 12)

This input sets the default measurement time period for monitoring the input in order to determine the LIN baud rate. If low, a time of 0.5 seconds will be used, and if high, 10 seconds is used. This is only the power-up time period – it can be changed at any time using the AT SM command, which is discussed later. Note that this time period is only used when in the automatic baud rate mode. If a specific baud rate has been selected, or if a previous rate measurement successfully determined a baud rate, there is no need to make more baud measurements, and this setting will not be relevant.

LIN (pin 13)

This is the active high LIN signal input. The signal from the LIN bus is inverted and buffered, then presented to this pin. Note that this input is limited to voltages from V_{SS} to V_{DD}, so it can not be directly connected to the LIN bus. (See the Example Applications section for a typical interface circuit.)

V_{SS} (pin 14)

Circuit common is connected to this pin. This is the most negative point in the circuit.

Ordering Information

These integrated circuits are available in either the 300 mil plastic DIP format, or in the 150 mil SOIC surface mount type of package. To order, add the appropriate suffix to the part number:

300 mil Plastic DIP.....ELM630P

150 mil SOIC..... ELM630SM

Absolute Maximum Ratings

Storage Temperature..... -65°C to +150°C

Ambient Temperature with
Power Applied.....-40°C to +85°C

Voltage on V_{DD} with respect to V_{SS}..... 0 to +7.0V

Voltage on any other pin with
respect to V_{SS}..... -0.6V to (V_{DD} + 0.6V)

Note:

Stresses beyond those listed here will likely damage the device. These values are given as a design guideline only. The ability to operate to these levels is neither inferred nor recommended.

Electrical Characteristics

All values are for operation at 25°C and a 5V supply, unless otherwise noted. For further information, refer to note 1 below.

Characteristic	Minimum	Typical	Maximum	Units	Conditions
Supply voltage, V_{DD}	4.5	5.0	5.5	V	
V_{DD} rate of rise	0.05			V/ms	see note 2
Average supply current, I_{DD}		0.8	1.4	mA	
Input low voltage	V_{SS}		0.15 x V_{DD}	V	
Input high voltage	0.85 x V_{DD}		V_{DD}	V	
Output low voltage			0.6	V	Current (sink) = 8.7mA
Output high voltage	V_{DD} - 0.7			V	Current (source) = 5.4mA
RS232Rx pin input current	-0.5		+0.5	mA	see note 3
RS232 baud rate		57600		baud	see note 4

Notes:

1. This integrated circuit is produced with a Microchip Technology Inc.'s PIC16C505 as the core embedded microcontroller. For further device specifications, and possibly clarification of those given, please refer to the appropriate Microchip documentation (available at http://www.microchip.com/).
2. This spec must be met in order to ensure that a correct power on reset occurs. It is quite easily achieved using most common types of supplies, but may be violated if one uses a slowly varying supply voltage, as may be obtained through direct connection to solar cells, or some charge pump circuits.
3. This specification represents the current flowing through the internal protection diodes when the voltage connected to the RS232Rx input (through a current limiting resistance) is greater than V_{DD} or less than V_{SS}. Currents quoted are the maximum that should be allowed to flow continuously.
4. Nominal data transfer rate when the recommended 3.58 MHz crystal is used as the frequency reference. Data is transferred to and from the ELM630 with 8 data bits, no parity, and 1 stop bit (8 N 1).

Communicating with the ELM630

The ELM630 relies on a standard RS232 serial connection to communicate with the user. The data rate is not adjustable, and is set at 57600 baud, with 8 data bits, no parity bit, 1 stop bit, and no handshaking (often referred to as 57600 8N1). All responses from the IC are terminated with a single carriage return character and, optionally, a linefeed character. Make sure your software is configured properly for this connection and for the linefeed mode that you have chosen. No special software is required to 'talk' to the IC – a standard terminal program is all that is needed.

Once it has been properly connected and powered, the ELM630 will send the message:

```
ELM630 v2.0

>
```

In addition to identifying the version of this IC, receiving this string is a good way to confirm that the computer connections and terminal software settings are correct. The '>' character displayed above is the ELM630's prompt character, which shows that the device is in its idle state, ready to receive characters on the RS232 port.

All messages that are sent to the ELM630 must begin with the character 'A' followed by the character 'T', and must be terminated with a carriage return character. No action is taken – commands are not checked for errors, nor are they acted upon – until this terminating carriage return is received. The one exception is when a command is interrupted for some reason, and no carriage return appears. In this case, an internal timer will automatically abort the incomplete message after about 15 seconds, and the ELM630 will print a single question mark to show that the input was not understood (and was not acted upon).

Messages that are not understood by the ELM630 (syntax errors) will always be signalled by this same single question mark ('?'). When this occurs, it is usually due to a spelling mistake, so you often only need to repeat the input, typing more carefully.

Occasionally, errors occur if the ELM630 is busy processing LIN messages when an RS232 command message begins. In these cases, the first character of the RS232 command message will always be missed by the IC, so the remaining characters will appear to be incorrect. One should always interrupt the monitoring process with a single character (it doesn't matter which one, as it will be ignored), then wait for the prompt character ('>') to appear before sending any more. This ensures that the ELM630 is ready to receive commands.

For convenience, the ELM630 has been designed to ignore spaces and control characters in the input, so if you prefer to add spaces or tabs, etc. to improve readability, then go ahead. Also, the ELM630 is not case-sensitive, so 'ATZ' is equivalent to 'atz', and to 'AtZ', which may be helpful in some situations.

AT Commands

The ELM630 is controlled with short commands that all begin with the two characters 'AT' (which is short for ATtention). These two characters serve no purpose other than to add validity to the characters that follow. Modem manufacturers have used this same technique for years, and it has become customary to refer to commands that begin with these characters as 'AT Commands'.

The ELM630 accepts several different AT commands, but only one at a time (it cannot process multiple commands on one line, as modems can). Each command is executed only upon the receipt of a terminating carriage return character. Several commands do not have a visible response (AT D for example), so completion of such commands will be acknowledged by the printing of the characters 'OK'.

Monitoring of the LIN bus can generally begin without requiring the use of any AT commands, as the factory default settings are appropriate for most situations. Occasionally, however, the user may wish to customize settings, such as turning the character echo off, and in these cases AT commands must be used.

To perform the desired AT command, simply send the characters AT followed by the appropriate characters from the following list. For example, to turn character echoing off, simply send AT E0 followed by a return character. To turn it back on, send AT E1.

The following is a summary of the commands that are recognized by the current version of the ELM630.

AT Commands (continued)

Remember that they are not case-sensitive, they can have spaces or tab characters embedded as you wish, and that for the on/off type commands, the '0' character is the number zero:

BR [display the Baud Rate]

This command asks the IC for the LIN baud rate that it is currently operating at. If the IC is set for auto and no rate has yet been determined, the reply will be 'AUTO'. Otherwise, it will be one of the allowed rates (1200, 2400, 4800, 9600 or 19200).

D [set all to Defaults]

This command resets all of the options to their default values as are set after a power-up, or manual reset command. This includes the baud rate, echo, linefeed, and the measurement period. Note that if the Monitor pin was at a high level during power-up, the IC will immediately enter the Monitor All (AT MA) mode after this command is issued.

E0 and **E1** [Echo off(0) or on(1)]

These commands control whether or not characters received on the RS232 port are re-transmitted (or echoed) back to the host computer. To reduce traffic on the RS232 bus, users may wish to turn echoing off by issuing AT E0. The default is E1 (echo on).

I [Identify yourself]

Issuing this command causes the chip to identify itself, by printing the startup product ID string (this is currently 'ELM630 v2.0'). Software can use this to determine exactly which integrated circuit it is talking to, without resorting to resetting the entire IC.

L0 and **L1** [Linefeeds off(0) or on(1)]

The option of transmitting a linefeed character after each carriage return character is controlled by these commands. If AT L1 is issued, linefeed generation will be turned on, and AT L0 will turn it off. Users may wish to have this option on if using a terminal program, but off if using a custom interface (as the extra characters serve no real purpose in that case). The default setting is determined by the logic level at the LFmode pin on powerup or reset: if it is a '1' or high level, then the default is L1, otherwise it is L0.

MA [Monitor All]

This command causes the IC to immediately begin monitoring the LIN bus for messages, displaying all that it finds. It can be interrupted at any time by activity on the RS232 receive input.

MR hh [Monitor for a Response to hh]

This is a special version of the AT MA command. Only identifier/command bytes matching the hex characters hh will be displayed, and all others will be ignored. This can be useful if trying to track down a troublesome switch problem or for triggering another action on a specific input.

SB n [Set the Baud rate to n]

Issuing this command causes the chip to configure itself for operation at baud rate n, where n is a number from 0-7. The possible values for n, and the corresponding baud rates are shown under the SB column in Table 1 on page 3. (For example, issue AT SB 5 to monitor a 19200bps LIN bus).

SM hh [Set the Measurement period to hh]

This command adjusts the time allowed for a baud rate measurement to that specified by hh, in 100 msec increments. A value of 00 is special, and causes the default period to be used, while values of 01 to FF provide times of 0.1 sec to 25.5 sec.

Z [reset all]

This command causes the chip to perform a complete reset, as if the power was cycled off and then on again. All settings are returned to their default values, and if the Monitor input is at a low level, the chip will be put in the idle state, waiting for characters to arrive on the RS232 bus. Note that the level at the Rxmode input is measured again at this time, so the RS232Rx input may be affected, if the level at pin 4 has changed since the initial power-up.

The LIN Standard

The cost and the weight of wiring harnesses in automobiles has been a concern to manufacturers for some time. To reduce both of these, manufacturers have begun adopting network bus structures to allow sharing of common information (and common copper). While several topologies (CAN, etc.) were developed for the high-speed information requirements, these systems were a definite overkill for some applications, particularly when interfacing with humans. Seeing this need, the LIN Consortium was formed, and the Local Interconnect Network standard was developed.

The LIN standard uses a bit serial interface that is in most respects identical to that used for personal computers. This keeps costs to a minimum as almost 'off the shelf' parts can be employed, and the learning time for developers is quite short, as the techniques are familiar. One main difference between the RS232 and the LIN protocols is that the LIN system requires a synchronizing signal.

The LIN synchronizing signal consists of a period of at least 13 consecutive bit times that are all in the active ('0') state. (This would normally never occur in an RS232 system as a start bit and 8 data bits could only give a maximum of 9 active bits.) The 13 bit long start signal (known as the 'Synch Break' signal) is always initiated by the bus master processor, to signal that a data transfer is about to follow. Once the Synch Break occurs, the message bytes are all sent in the same manner as for standard RS232.

Each message that is sent is split into two parts, the header and the response. The bus master always generates the header, while the response can be provided by either the master or by a slave processor on the bus (depending on whether the master is trying to send information or obtain it).

The first byte of the header is known as the 'Synch Byte', and it is always the byte value $55. That value was chosen because it creates a pattern of alternating '1's and '0's that can be used by the slave devices to perform an internal timing calibration. This allows inexpensive RC oscillators to be used for the slave processors, if desired.

Following the Synch Byte, the master will always send an Identifier Byte which describes the information which is required, or that which is to follow. One can think of it as the command byte.

The response field occurs after the ID byte, and will generally consist of two, four, or eight data bytes, followed by a single checksum byte. The standard does allow certain commands to initiate messages of arbitrary length however, so be aware of this if developing software to process the LIN data.

The following figure may prove helpful in visualizing a typical LIN message:

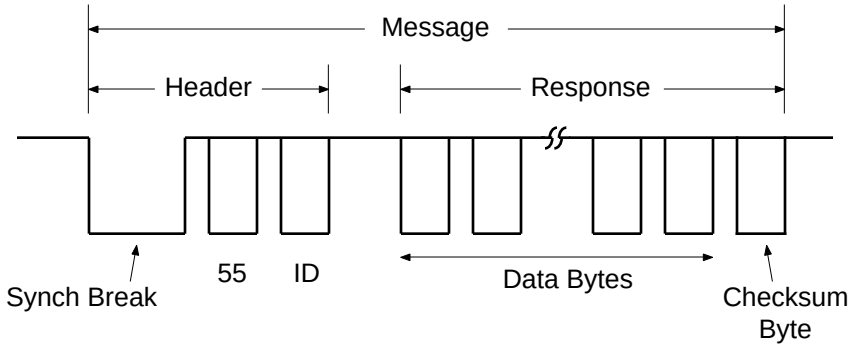


Figure 1. The LIN Message Structure

Measuring the Baud Rate

If the 'auto' baud rate mode is selected, the ELM630 will automatically configure itself for operation at whatever baud rate it determines is present on the LIN bus. It does this by monitoring the LIN data for the time period set by the AT SM command (or by the MTime pin), and calculating the baud rate from the bit patterns that were detected. This measurement will occur when the first monitor command is to be performed (either an AT MA or an AT MR).

Often, the bus will be asleep when a monitor request is first made by the user. In this case, if the ELM630 is in the auto mode, it will display the error message '[B]' to say that there has been a problem determining the baud rate. It does not stop measuring, however; it continues on, repeating the measurement routine until a reasonable baud rate has been found.

If the user should explicitly set the LIN baud rate, either by the AT SB command or by hard-wiring the b2, b1 and b0 pins, no baud rate measurement will be made. All bus monitoring will begin immediately after being requested, rather than being delayed by having to make a baud rate measurement. This could be an advantage, if the system's baud rate is known.

If the baud rate is not initially known, it is preferrable to use the auto baud mode rather than simply guessing at rates. The advantage to using the auto mode is that it allows the system to re-measure the baud rate should a synchronizing error occur. This allows automatic recovery for systems that perhaps have varying rates, or for a tester that is moved from system to system. This error recovery measurement is not performed automatically if a fixed baud rate has been set.

The measurement period can be adjusted using the AT SM hh command. This allows flexibility for systems that operate at very low baud rates, or have very infrequent messages. In these cases, a longer measurement period can be selected in order to ensure that enough samples appear to the ELM630 for reliable baud rate detection. Experimentation has shown that a value of about '3600 ÷ baudrate' seconds is often adequate, and will certainly make a good initial value for your testing. For example, a system that might be operating at 1200 baud would ideally be set for a measurement period of 3 seconds.

If unsure of the current baud rate, or of the rate after an automatic measurement, it can be requested by issuing the AT BR command. If the ELM630 is in the automatic mode and no baud rate has yet been determined, it will simply display 'AUTO'. After a rate has been measured, or if it has been set manually, AT BR will respond with the actual baud rate.

Monitoring the LIN Bus

Data bytes that appear on the LIN bus can assume values from 0 to 255. These cannot be displayed using a PC terminal program, however, since many of these values are not printable characters. In order to make them readable, the ELM630 re-formats every byte as a pair of hexadecimal digits, using standard ASCII characters for the digits. A typical request made of the ELM630 would appear as:

```
>AT MA
49: 6C 8F 04
```

The AT MA is the user's request to 'monitor all', while the 49: 6C 8F 04 is what the ELM630 found on the LIN bus. Note that the initial Synch Byte is always received, but is never displayed (it is always 55).

The identifier byte (49 in this example) always appears first, and is separated from the data bytes by a colon character (':'), while the pairs of hexadecimal digits following represent the data bytes that were received. The final pair of digits on each line is the checksum byte (04 in this case).

Should the checksum byte not match the value calculated internally by the ELM630, the error will be flagged by printing a single question mark at the end of the line. For an example, if the slave driver in this case was 'weak' (allowing an extra '1' to appear in the first data byte of this example), the output might typically look like:

```
>AT MA
49: 7C 8F 04?
```

The question mark at the end of the line alerts you to the fact that an error is present, but the position of the error can not be determined from this information - you will only be able to say that it is somewhere in the

Monitoring the LIN Bus (continued)

response. Another type of error that could occur is when a slave fails to respond to the master. In this case, you would typically see only the identifier, followed by a question mark:

49:?

The question mark gets printed in this case because 49 is not a valid checksum value (all bytes must add up to FF).

This would often be followed by a timeout, so the output could appear as:

49:?
[T]

As you experiment, you will likely find that timeouts are a very common occurence. They simply mean that there has been no activity for some time, often due to the system going into a low-power sleep mode.

This data is typical of what might be experienced in many systems. The current version of the LIN standard (2.0) does allow for the transmission of an arbitrary number of data bytes, however, so while the ELM630 has no limitations on what it can display, the user should be prepared for that possibility. This could involve simply allocating enough buffer space, processing the data faster than it arrives, or by simply ignoring lines that are longer than a predetermined length.

Error Messages

There are only a few error messages that the ELM630 will generate in response to data problems. They are kept short so that their sending does not interfere with the LIN data monitoring. Here is a brief description of each:

[B] [Baud rate error]

This error will be printed if the ELM630 attempts to make a baud rate measurement and fails. Failure may occur if the measured frequency is too high or too low, or if there was no activity on the bus (the devices are all 'sleeping'). Even though the error has occurred, the IC will continue to monitor the bus, and will carry on with the monitoring once a successful baud rate measurement has been made.

[S] [Sync error]

If the ELM630 is unable to synchronize to the LIN data stream, or if the received data bytes seem to be of incorrect length, a synchronizing error occurs. This might be for several reasons - if the device has been set to a rate that differs from the actual data rate; if the data stream is not of the LIN format, and sync breaks or bytes could not be found; or possibly if something occurred during data transfer and the stop bits did not occur when expected. The ELM630 will report this error, and immediately try to re-synchronize to the bus. If the ELM630 has been set to the 'auto' mode, a new baud rate measurement will also be initiated before the next attempt to synchronize.

[T] [Timeout error]

This message is printed if there have been no recognizable messages for some time (referred to as a 'Bus Idle Time-Out' in the LIN specification). The time period in the specification is actually 25000 bits, and the ELM630 sends the message after about 27500 bit periods. This timeout normally occurs when the LIN system has been put to sleep.

When a timeout occurs, the ELM630 will continue to monitor the bus, and will resume resending the LIN data when it appears again.

? [General error]

This is the standard response for a general error, often due to a misunderstood AT command received on the RS232 bus (usually due to a typing mistake). It can also occur if the checksum byte that was received was not correct for either a classic (LIN1.3) or an enhanced (LIN2.0) system.

Example Applications

The circuit of Figure 2 (on the next page) shows how the ELM630 might typically be used. While the LIN standard does not specify the type of connector that is to be used, it does specify that the physical interface will essentially be an enhanced implementation of the ISO 9141 standard currently used in automobiles. It provides a three wire connection, with power, data, and common.

The provision of a (nominal) 12V supply at the connector provides a convenient means to power the ELM630 circuit. This voltage is reduced to the required 5 volts by the 78L05 regulator shown, and filtered with the capacitors. An LED provides visual feedback that the supply is available.

The PNP transistor that is shown connected to the LIN input (pin 13) serves mainly to level-shift the input signal, and to invert it. A typical CMOS input will switch state at about half the supply level (2.5V) which might cause problems with noise in an automotive environment. By providing a transistor buffer, the input threshold is effectively raised to about 4V, increasing noise immunity, while adding amplification, signal clamping, and inversion. Note the use of the diode on the input, which prevents possibly damaging backfeeds from entering the circuit, protecting the transistor and the other components.

A very basic RS232 interface is shown connected to pins 5 and 7 of the ELM630. This circuit 'steals' power from the host computer in order to provide a full swing of the RS232 voltages without the need for a negative supply. The RS232 pin connections shown are for a 25 pin connector; if you are using a 9 pin, the connections would be 2(RxD), 5(SG) and 3(TxD).

RS232 data from the computer is directly connected to pin 5 of the IC through only a 47KΩ current limiting resistor. This resistor allows for voltage swings in excess of the supply levels while preventing damage to the ELM630. A single 100KΩ resistor is also shown in this circuit so that pin 5 is not left floating if the computer is disconnected. Note that pin 4 has been tied to V_{SS} in order to configure the receive circuit for this type of connection.

Transmission of RS232 data is via the single PNP transistor connected to pin 7. This transistor allows the output voltage to swing between +5V and the negative voltage stored on the 0.1µF capacitor (which is charged by the computer's TxD line). Although it is a simple connection, it is quite effective for this type of application.

Circuit timing is maintained by the crystal shown connected between pins 2 and 3. It is a common TV type that can be easily and inexpensively obtained. The 27pF crystal loading capacitors shown are only typical, so you may have to select other values depending on what is specified for the crystal you use.

Pins 8 to 12 have been tied to V_{SS} for this circuit, setting power-up options as follows. Pins 8, 9, and 10 are all '0' so the baud rate has been selected to be 'auto', allowing the IC to adapt to whatever baud rate appears. Pin 11 is also low, meaning that the 'Monitor All' mode will not be entered into on a power-up or reset – the chip will simply issue a prompt character and wait for instructions from the user. Finally, pin 12 is also tied to V_{SS}, so the default measurement period for determining the baud rate will be 0.5 seconds. This is often an adequate setting, even for the slowest (1200 bps) systems.

Another application for the ELM630 is in the hand-held "data snoop", or "LIN Logic Probe" that is shown in Figure 3. This circuit is very similar to that of Figure 2, except that in this case pin 11 has been tied to V_{DD}, causing the IC to immediately enter a "Monitor All" mode on power-up. Setting pins 8, 9 and 10 to low selects auto-baud operation, allowing this circuit to be used on virtually all LIN systems. Note that since there is no need to send commands to the ELM630 in this case, the RS232 interface has been simplified even further, eliminating the receive components. Power is from three button cells, which provide 4.5V to the circuit through a momentary action pushbutton switch. The 4.5V is at the lower level of recommended voltages, but is still within specified limits.

Hopefully this has provided enough information to get you started with this convenient device. By monitoring traffic on your automobile's seat control circuit, or your clothes dryer's information bus, one can learn a great deal. Certainly, you will be able to identify what is normal, then of course you will know when it is not…

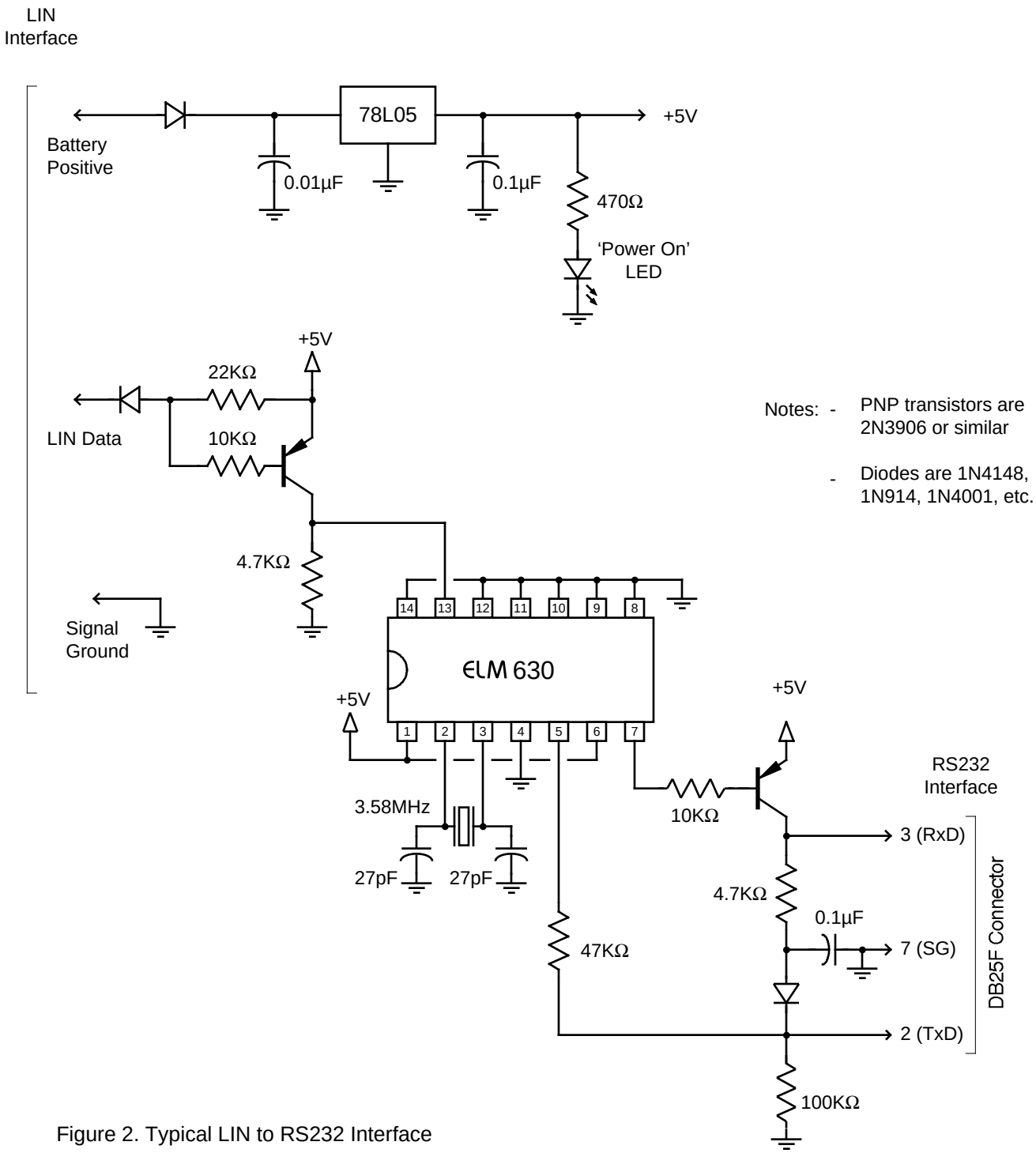


Figure 2. Typical LIN to RS232 Interface

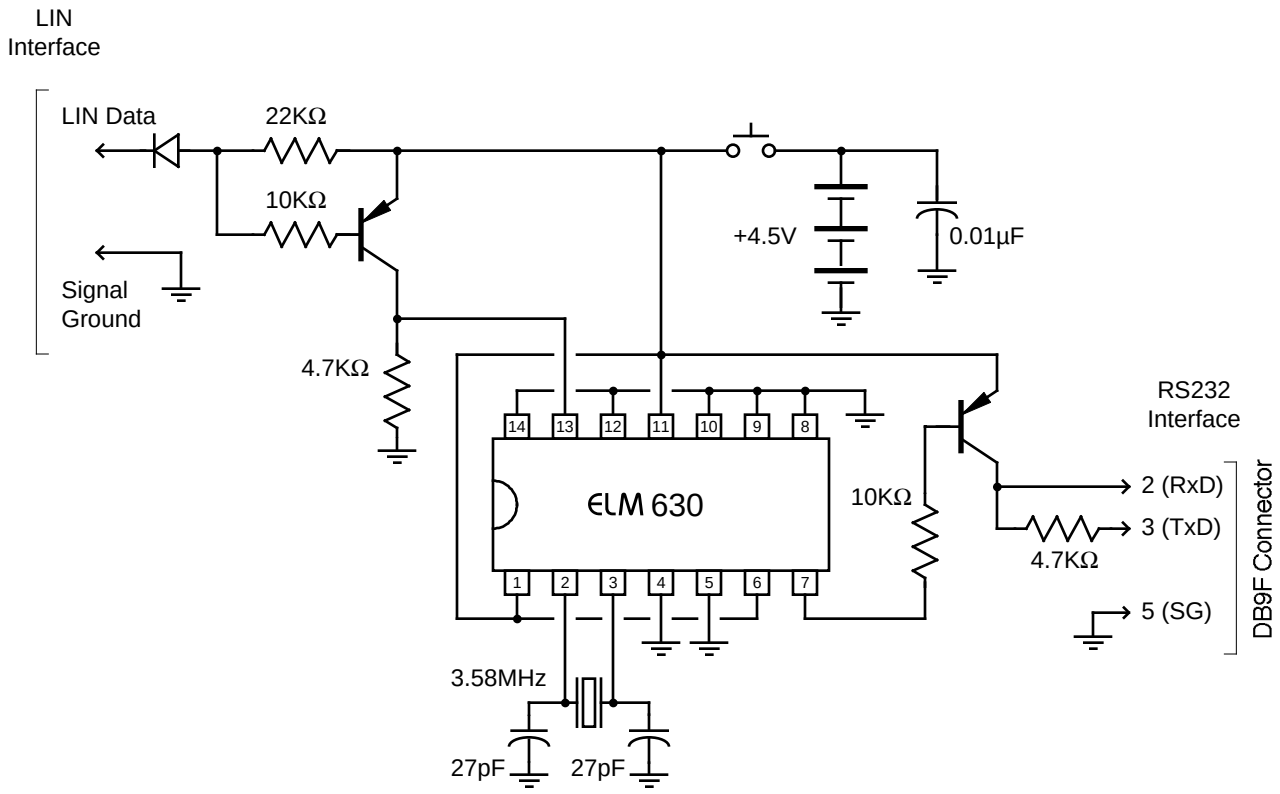


Figure 3. LIN Logic Probe