Impact Factor (SJIF): 3.632

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 3, Issue 3, March-2016

Low cost implementation of local Interconnect Network (LIN) node on CPLD -A design approach (R & D paper)

Ms. Rita Aggarwal (Asst. Prof),
E & C Department , Babaria Institute of Technology,
Vadodara-Mumbai NH #8, Varnama, Vadodara,India

***Abstract*:** Electronics has become a driving force of innovation in today's vehicle industry. Communication networks in automobiles have been introduced with advancement in technology. In last few years number of protocols have been designed with a focus on cost optimization and aimed for use as a simple multiplexed electrical system. One of these protocol is local interconnect protocol, which has a potential to become a standard in the automotive industries in the near future. This paper presents a simple design of a lin node, which has been tested on cool runner –II cpld.

***Key words*:** *multiplexed, nodes, automotive, LIN, CPLD*

I.Introduction:

Almost every part of the vehicle today interacts with some kind of electronic unit, from climate control, anti spin systems and throttle control to switches, lights and sensors. The number of electronic control units (ECU) and particularly the wiring in vehicles in present vehicles are at a level where one has to take the cost and space requirement into serious consideration when adding new functionalities Not just the cost and space for wiring is becoming a problem in modern vehicles but also the increasing size of model program (required for automation) that the new functionalities enables. Different models require different wiring at production time and difficulties arise when number of versions to be handled becomes too large. To tackle this problem and to have a common standard sub bus, car manufacturers in Europe have formed a consortium to define a new communication standard for automotive sector. This new bus is known as LIN bus.

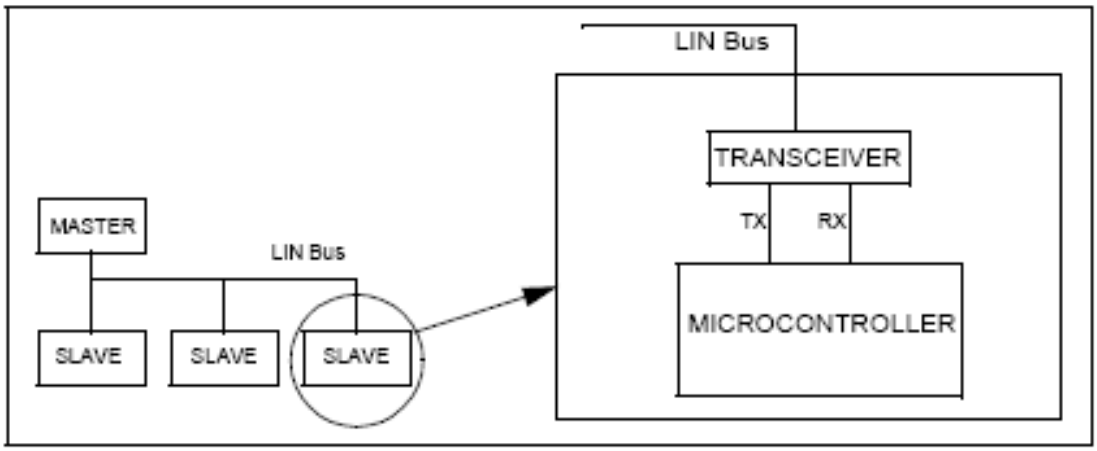
II.Features of LIN

LIN is a serial communication protocol which efficiently supports the control of mechatronic nodes in distributed automotive applications. The main properties of the LIN bus are:

1. Single master with multiple slave concept.
2. Low cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software or as pure state machine.
3. Self synchronization without a quartz or ceramic resonator in the slave nodes.
4. Low cost single wire implementation.
5. Speed up to 20 Kbits/s.
6. Signal based application interaction.
7. Predictable behaviors.

III. Concept of Master and Slave:

The master initiates all data transfers on the bus. Thus no arbitration scheme is necessary and contention for bus access is not an issue. This leads to deterministic traffic behavior and guarantees the latency times calculated for the specified frames transmitted on the LIN-network.



I. Figure1. Lin Bus Topology

IV. LIN message frame

The LIN message frame consists of a header and a response part. The header has a fixed length while the response part consists of 0 to 8 bytes of data. The inter-frame-response time is the time it takes for a slave to respond to a request (i.e. to a ID) from the master and it may vary among the nodes on the network since it depends on the hardware and software implementation in each node. At the end of the response part a checksum, which is calculated for the data part, is attached. The header is broken up into three fields: the SYNC-break, SYNC-field and the identifier- (ID) field.

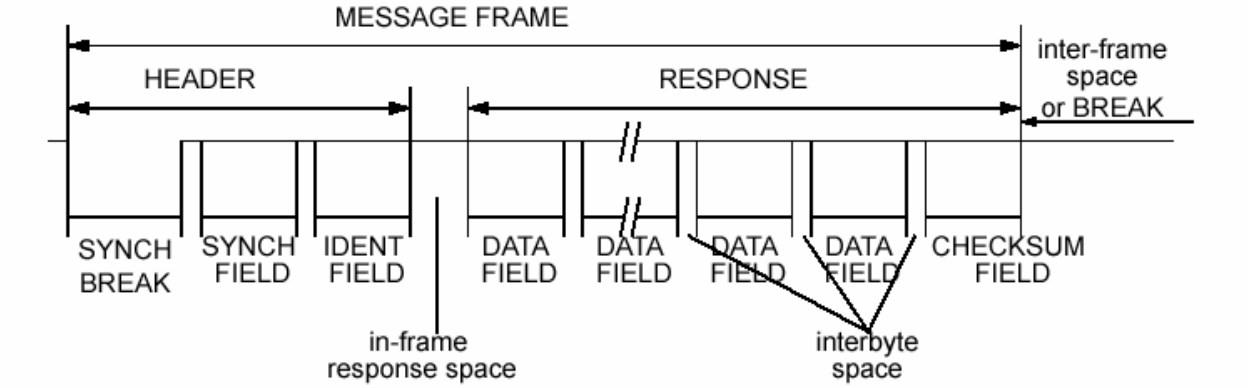


Figure 2: LIN message Frame

(i) Synchronization break:

The first part of a LIN-message is the SYNC-break, which consists of at least 13 bits of zeroes. The break is needed in order for the slaves to detect that a message is transmitted on the bus. According to the specification [LIN02] the slaves are allowed to have a clock frequency (i.e. baud rate) that differs with 15% to the masters. With this in mind, for a unsynchronized slave node with a clock frequency that is 15% slower than the master´s clock to detect a break, it has to sample at least 10 bits as zeroes since 9 zeroes can still be found in a ordinary UART-byte. For the master this would mean that 10 x 1.15 _ 12 bits would be sufficient to send in order for the slaves to detect the break. However, the specification states that a minimum of 13 bits should be sent.

(ii) Synchronization field

Since the master always initiates a transmission by sending a header, the slaves are able to synchronize their clocks every time a new message is received. This makes it unnecessary to implement expensive resonators or oscillators on the slave nodes and only one, by comparison, accurate resonator is necessary in the master as a time reference. The slave synchronizes its clock by measuring the time taken from the first falling edge of the ID-field (i.e. the falling edge of the start bit) to the fifth falling edge i.e. bit 7 of the SYNC-byte and divides it by 8 in order to get the bit time or baud rate of the master

(iii) Identifier field

In the last part of the message header the ID- field is placed. The ID- field denotes the content of the data part of the message and is protected with two parity bits. In the LIN specification rev.1.2, two of the bits in the ID-field are used for specifying the length of the data part of the message and the numbers of bytes allowed were 2, 4 or 8. In the current revision this is no longer the case and 0 to 8 bytes of data can be used. The slave nodes on the network are addressed by the ID- field. The nodes do not have physical addresses.

(iv) Data Field:

The data frame contains from two to eight Data Fields containing eight bits of data each. Transmission is done with LSB first. The data fields are written by the responding Slave task. Since there is no bus arbitration, only one slave task should be allowed to respond to each identifier. All other Slave tasks are limited to reading the response and act accordingly.

(v) Checksum Field:

The last field in the Message Frame is the Checksum Field. This byte contains the inverted modulo-256 sum of all data bytes (data frame not including the identifier). This sum is calculated by doing an "ADD with carry" on all data bytes and then inverting the answer. The properties of the inverted modulo-256 sum are such that if this number is added to the sum of all data bytes the result will be "0xFF".

V. *Design of a LIN Module*:

To design a complete LIN interaction network the first step is to design a single Lin (master/slave) controller which can be connected with similar controllers to initiate communication between them. The basic inner architecture s of a master and slave units are the same.
Each node should have the capability to transmit, receive, synchronize etc. Basic blocks of the node are as follows:

Configuration Registers: This block handles the bulk of CPU interface and mediates the setting and clearing of various status and control registers.

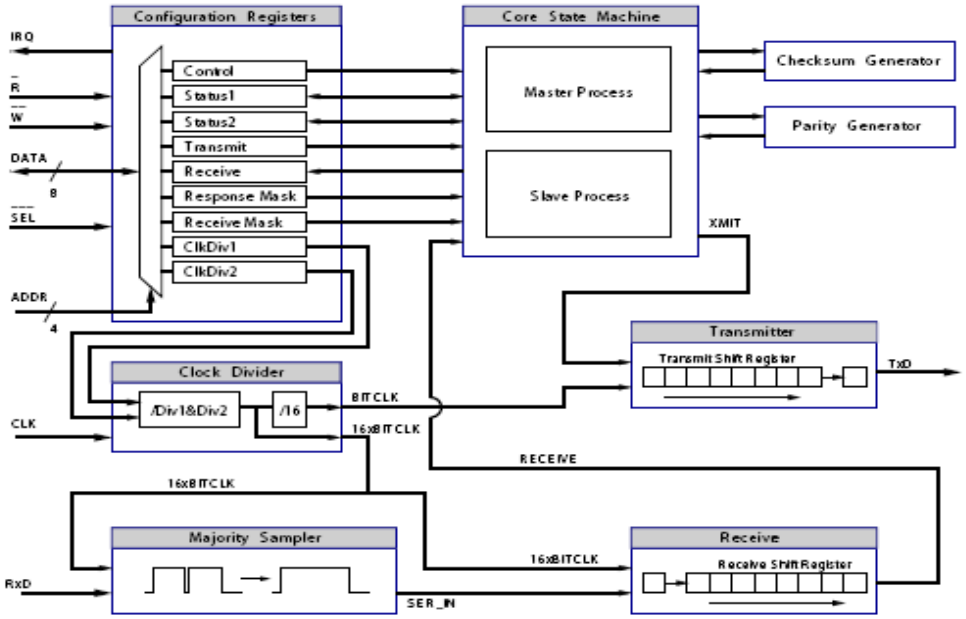
Core State Machine module: It handles all bit timing operations, set/resets various status registers and interrupts the CPU when appropriate.

Clock Divider: The clock divider divides the system clock down to internal clock

Receiver: It is implemented as a simple serial in parallel out shift register.

Transmitter: It is implemented as a parallel in serial out shift register with a strobe to load a new input.

Majority Sampler: It samples the incoming bit stream at 16times the current bit rate.



9

Figure 3: Block diagram of a LIN controller

VI. *Impementation:*

Xilinx Synthesis tool (VHDL) and ModelSim Simulator can be used for implementation of the controller. Xilinx ISE Design Tool is the ideal downloadable solution for FPGA and CPLD design offering HDL synthesis and simulation, implementation, device fitting, and JTAG programming.

VII *Simulation Results*:

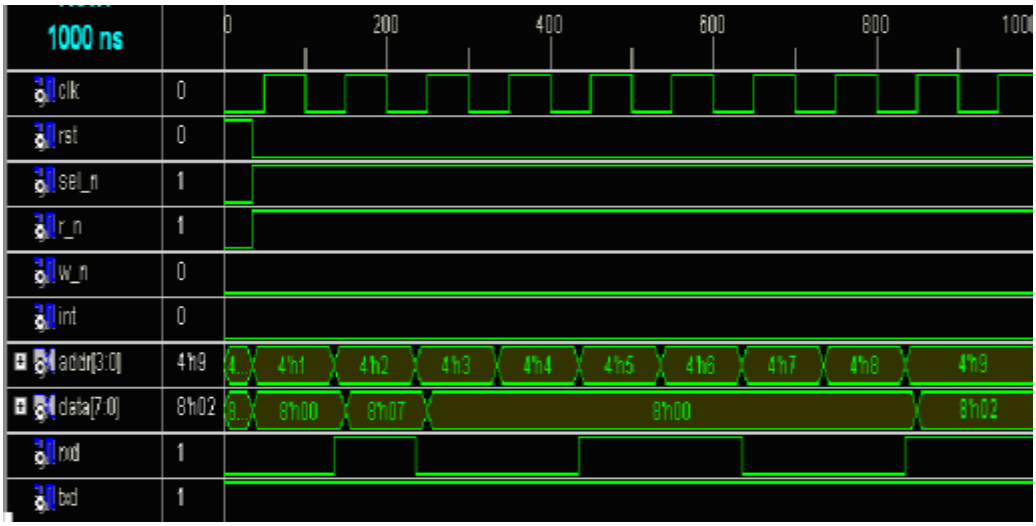


Figure 4: Master/Slave Read Cycle

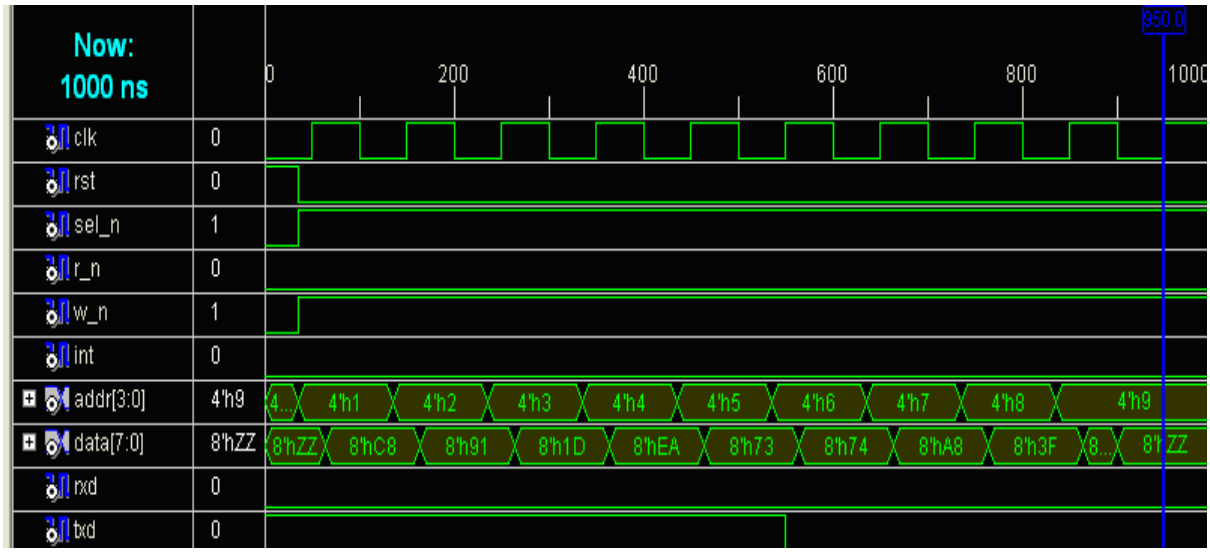


Figure 5: Master/Slave Write Cycle

FINAL CPLD REPORT

VII CONCLUSION

The LIN protocol in Master/Slave mode is implemented on cool runner-II cpld and results were found satisfactory. The physical connection between LIN network members is done by LIN transceivers.
A LIN network can consist of one Master Control Unit and up to sixteen Slave Control Units. The LIN node has sleep mode feature which reduces internal activity, with additional function such as passive bus drivers with the purpose of reducing system power consumption. The LIN module was developed using the "top down" approach and VHDL for reducing designing time. Functional simulation verified that the controller is internally consistent; that is, a network composed of these controllers can communicate with each other and each node correctly raises error conditions. Physical testing verified that the implementation worked correctly in a single node LIN network and also verified that the configuration and status registers functioned correctly.

References:

[1] Implementing a LIN Controller on a CoolRunner-II CPLD" (v1.1) April 3, 2007, www.Xilinx.com
[2] "LIN specification package, Revision 2.1,Nov24,2006"LIN Consortium,2006
[3] Local Interconnect Network (LIN) Demonstration – Motorola Semiconductor, Application Note – AN2103, January 2002
[4] J.Will Speks, Motorola GmbH, Munich Antal Rajnak, Volcano Communication Technologies, *LIN – protocol, development tools, and software inerfaces for local interconnect network in vehicles.*
[5] Christopher A. Lupini, *Mutiplex bus progression* Delphi Delco Electronics System
[6] Roth, H. Charles *Digital system design using VHDL* , PWS Publishing Copmany, 1998
[7] *Philips smart power solitions*, Philips Semiconductors, February 2002