P-3543

MAHATMA GANDHI
KCT
CENTRAL LIBRARY

INFOTAINMENT AND COMFORT ENHANCEMENT IN VEHICLE USING LIN COMMUNICATION

By

A.SARANYA	Reg. No: 0710107087
R.G. SREETHI THANGAM	Reg. No: 0710107100

of

KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE - 641 049.
(An Autonomous Institution affiliated to Anna University of Technology, Coimbatore)

A PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements
for the award of the degree

of

BACHELOR OF ENGINEERING
in
ELECTRONICS AND COMMUNICATION

BONAFIDE CERTIFICATE

Certified that this project report entitled **"Infotainment and comfort enhancement in vehicle using LIN communication"** is the bonafide work of **A. Saranya** [Reg. no. 0710107087] and **R.G. Sreethi Thangam** [Reg. no. 0710107100] who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Kalaiselvi

Project Guide

Ms. A. Kalaiselvi

Head of the Department

Dr. Rajeswari Mariappan

The candidate with university Register no. 0710107087, 0710107100, is examined by us in the project viva-voce examination held on ...19/4/2011

19/4/11

Internal Examiner

External Examiner

✦pricol✦

pricol limited
plant 1 perianaickenpalayam coimbatore 641 020 ind
phone +91 422 4331100 fax +91 422 4331122
e mail factory@pricol.co.in website www.pricol.com

PL/TRG/PROJ/2011
10/03/2011

PROJECT COMPLETION CERTIFICATE
To Whomsoever It May Concern

We are pleased to issue this certificate in the process of operationalising our "Industry- Institute Interaction Synergy" drive.

Name	:	A. Saranya R G Sreethi Thangam
Institution	:	Kumaraguru College of Technology
Qualification	:	B.E (ECE)
Project Title	:	Infotainment and Comfort Enhancement in Vehicle using LIN Communication
Project Duration	:	December 2010 to February 2011
Performance / Conduct	:	Good

For Pricol Limited,
R. Divya
R. Divya
Sr. Officer - Training
Human Resources

ABSTRACT

This project involves development of a module to implement the required configuration at lower cost in the car one may hold now. New features and facilities are continually being introduced in new model cars. Buying a new car each time a new feature is introduced is highly impractical. Hence, through this project, we have made an approach to implement some of the features in the existing cars. LIN protocol is used to serve this purpose.

In our project we are going to implement the power window and audio control. This involves two slaves for cassette player control in steering and door control. The cassette player control in steering contains several switches to control player on/off, mode select, volume up and volume down and the door control contains switches for central lock, window lift and light on/off. These slaves are connected to a master controller that supports both LIN and CAN communication. The inputs given by the driver are sensed and the information is communicated to the master controller through a single wire LIN bus.

When this module is attached to the car, it is made to communicate with other existing modules through CAN bus. This can be implemented by programming the CAN bus.

ACKNOWLEDGEMENT

Owing deeply to the supreme, we extend our sincere thanks to GOD almighty, who has made things possible.

I express my sincere thanks to our beloved Director **Dr.J.Shanmugam**, Kumaraguru College of Technology, for his kind support and for providing necessary facilities to carry out the work.

We are grateful to **Dr. S. RAMACHANDRAN Ph.d.,** principal, for the facilities provided to complete this project work.

We wish to extend our gratitude to **Dr. RAJESWARI MARIAPPAN Ph.d.,** HOD, Electronics and Communication Engineering, for the constant support and encouragement.

It is our duty to thank our Project coordinator **Prof. K. KAVITHA, assistant professor,** Project guide **Ms. A.KALAISELVI, assistant professor**, Electronics and Communication Engineering, Kumaraguru College of Technology, with whose persistent help and timely motivation, this work has been made possible.

We wish to express our sincere thanks to **Mr.R.S.PRAKASH, assistant manager** and the members of **PRICOL LIMITED,** Periayanakanpalayam for their guidance in the project work. We thank our beloved parents for their constant support and blessings.

Finally we would like to thank the teaching and non teaching staff members of the department of Electronics and Communication Engineering for their suggestions and solutions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS		PAGE NO.
LIN	Local Interconnect Network	1
CAN	Controlled Area Network	3
ECU	Electronic Control Unit	3
EMI	Electro Magnetic Interference	5
CPU	Central Processing Unit	16
NRZ	Non Return to Zero	17
ICU	Interrupt Control Unit	17
SCK	Serial Clock	18
SPI	Serial Peripheral Interface	18
SCR	Serial Control Register	18
TDRE	Transmit Data Reception Register	19
TIE	Transmission Interrupt Enable	19
TIR	Transmission Interrupt Request	19
UPCL	Programmable Clear	19
SMR	Serial Mode Register	19
TXE	Transmission enable	19
ESCR	Extended Status Control Register	19
LBD	LIN Break Detection	19
RDR	Reception Data Register	20
RDRF	Receive Data Register Flag	20

SSR	Serial Status Register	20
ORE	Over Run Error	20
FRE	Frame Error	20
RIE	Reception Interrupt Enable	20
CRE	Clear Error	20
OTO	Ono To One	21
SOT	Serial Output	22
LBIE	LIN Break Interrupt Enable	23
BGR	Baud rate Generation Register	24

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 GENERAL VIEW:

Today technology has made our life simple and easy. In this project we have attempted to enhance comfort in vehicle using LIN communication.

The Local Interconnect Network (LIN) is well established as the solution of choice for low-cost in-vehicle networks, especially in body and comfort systems ranging from air conditioning to door-locks and mirror control. It is a serial communication protocol, suitable especially for automobile industry. It is a low-speed network in which instead of every air conditioning vent, electrical seat motor and sunroof being individually wired point-to-point to its own switches, only one wire can connect up to 16 separate points using "LIN". It can be used for every application which has no demand for high data rates.

In this project we had developed Cassette player control module and Door control module using LIN communication that can be attached to a car at low implementation cost, without involving any changes in the car's existing modules.

1.2 BASIC BLOCK DIAGRAM:

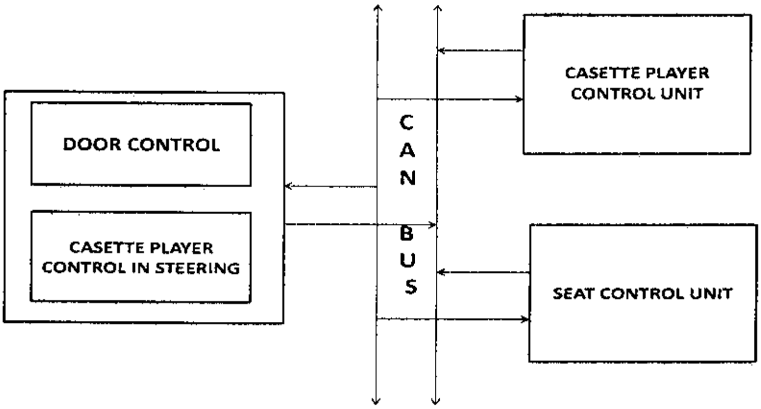

CASETTE PLAYER CONTROL UNIT
DOOR CONTROL
CASETTE PLAYER CONTROL IN STEERING
C A N B U S
SEAT CONTROL UNIT

FIG 1.1 OVERALL BLOCK DIAGRAM

CASETTE PLAYER CONTROL IN STEERING

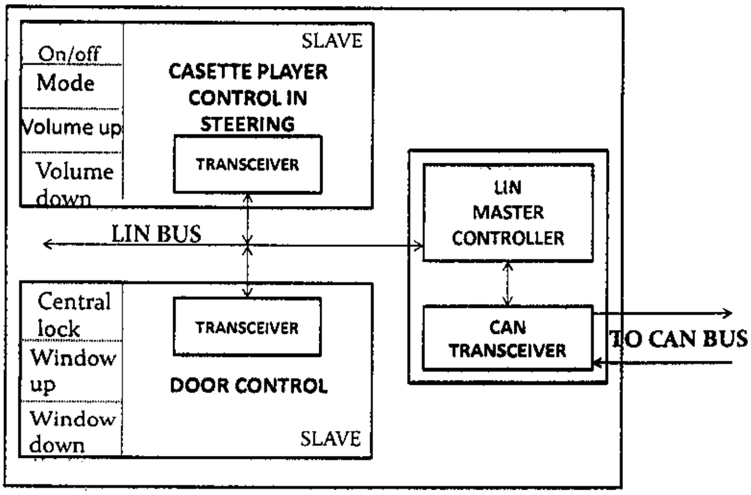


FIG 1.2 BLOCK DIAGRAM OF A SINGLE ELECTRONIC CONTROL UNIT

1.2.1. BLOCK DIAGRAM DESCRIPTION:

Overall block diagram description:

Figure 1.1 shows the overall block diagram. It consists of various electronic control units. Each unit is connected to the CAN bus. The transfer of information takes place through the CAN bus which is a two wire system.

Single electronic control unit description:

Figure1.2 shows the block diagram of a single electronic unit in which each slave is connected to the CAN network through the LIN bus.

Cassette player control unit:

Cassette player control in steering has four switches for volume up, volume down, mode select and player on/off.

Door control unit:

Door control has three switches for central lock, window up and window down.

These slave nodes are connected to a master controller that supports both LIN and CAN protocol through LIN network. The inputs given by the driver are sensed by the slave nodes and communicated to the master controller. Such information from each ECU are broadcasted to other ECU's through CAN network.

LIN DESCRIPTION

CHAPTER 2

LIN DESCRIPTION

2.1 GENERAL DESCRIPTION:

LIN (Local Interconnect Network) is a concept for low cost automotive networks, which complements the existing portfolio of automotive multiplex networks. LIN will be the enabling factor for the implementation of a hierarchical vehicle network in order to gain further quality enhancement and cost reduction of vehicles. The standardization will reduce the manifold of existing low-end multiplex solutions and will cut the cost of development, production, service, and logistics in vehicle electronics.

2.1.1. FEATURES AND POSSIBILITIES:

The LIN is a serial communications protocol which efficiently supports the control of electronic nodes in distributed automotive applications.

The main properties of the LIN bus are:

- Single master with multiple slave concept.
- Low cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software.
- Self synchronization without a quartz or ceramics resonator in the slave nodes.
- Deterministic signal transmission with signal propagation time computable in advance.
- Low cost single-wire implementation.
- Speed up to 20 kbps.

LIN provides a cost efficient bus communication where the bandwidth and versatility of CAN are not required. The specification of the line driver/receiver follows the ISO 9141 standard with some enhancements regarding the EMI behavior.

2.1.2. CONCEPT OF OPERATION:

Master and slave:

A LIN cluster consists of one master task and several slave tasks. A master node1 contains the master task as well as a slave task. All other nodes contain a slave task only. A sample LIN cluster with one master node and two slave nodes is depicted:

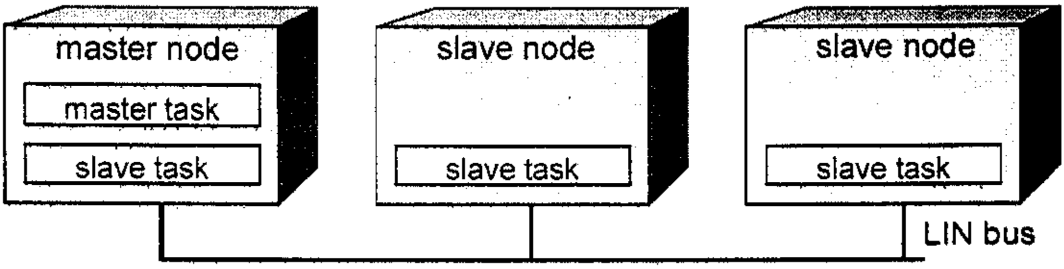


FIG 2.1 SAMPLE LIN CLUSTER

The master task decides when and which frame shall be transferred on the bus. The slave tasks provide the data transported by each frame. Both the master task and the slave task are parts of the Frame handler.

Frames:

A frame consists of a header (provided by the master task) and a response (provided by a slave task). The header consists of a break and sync pattern followed by an identifier. The identifier uniquely defines the purpose of the frame. The slave task appointed for poviding the response associated with the identifier transmits it, as depicted below. The response consists of a data field and a checksum field. The slave tasks interested in the data associated with the identifier receives the response, verifies the checksum and uses the data transported.

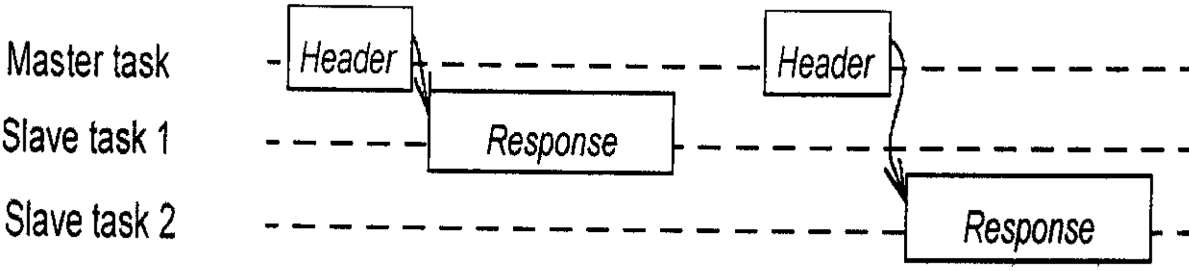


FIG 2.2 MASTER AND SLAVE TASKS

This results in the following desired features:

- **System flexibility:** Nodes can be added to the LIN cluster without requiring hardware or software changes in other slave nodes.
- **Message routing:** The content of a message is defined by the identifier2.
- **Multicast:** Any number of nodes can simultaneously receive and act upon a single frame.

2.1.3. DATA TRANSPORT:

Two types of data may be transported in a frame: signals or diagnostic messages.

1. Signals:

Signals are scalar values or byte arrays that are packed into the data field of a frame. A signal is always present at the same position of the data field for all frames with the same identifier.

2. Diagnostic messages:

Diagnostic messages are transported in frames with two reserved identifiers. The interpretation of the data field depends on the data field itself as well as the state of the communicating nodes.

2.1.4. SCHEDULE TABLE:

The master task (in the master node) transmits frame headers based on a schedule table. The schedule table specifies the identifiers for each header and the interval between the start of a frame and the start of the following frame. The master application may use different schedule tables and select among them.

2.1.5. FRAME STRUCTURE:

The structure of a frame is shown in figure below. The frame is constructed of a break followed by 4 to 11 byte fields.

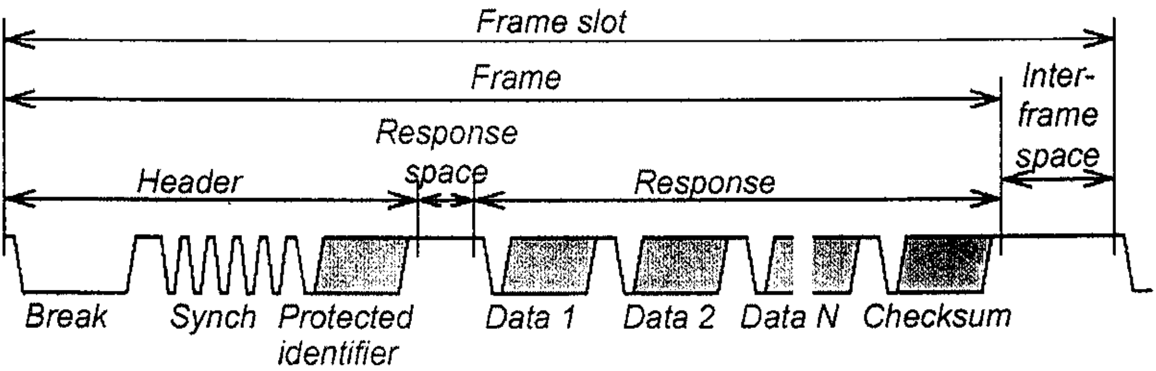


FIG 2.3 FRAME STRUCTURE

Each byte field is transmitted as a serial byte, as shown below. The LSB of the data is sent first and the MSB last. The start bit is encoded as a bit with value zero (dominant) and the stop bit is encoded as a bit with value one (recessive).

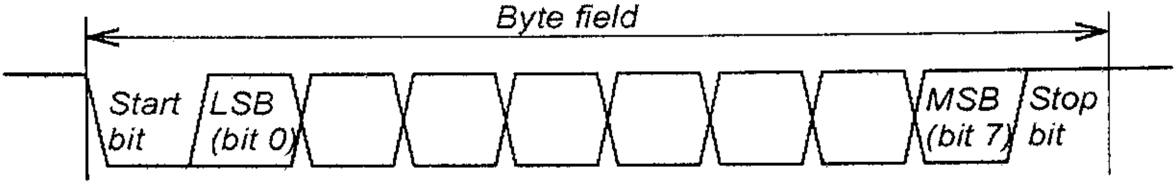


FIG 2.4 STRUCTURE OF A BYTE FIELD

Break:

The break symbol is used to signal the beginning of a new frame. A break is always generated by the master task and it shall be at least 13 bits of dominant value, including the start bit, followed by a break delimiter, as shown in Figure below. The break delimiter shall be at least one nominal bit time long. A slave node shall use a break detection threshold of 11 nominal bit times3.

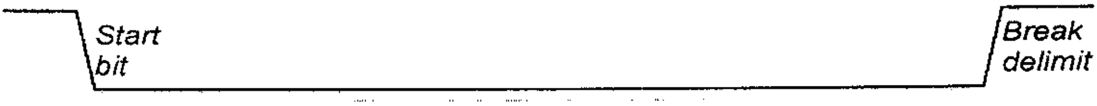


FIG 2.5 THE BREAK FIELD

Synch byte:

Synch is a byte field with the data value 0x55,

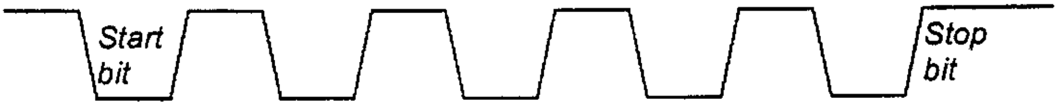


FIG 2.6 THE SYNCH BYTE FIELD

A slave task shall always be able to detect the break/synch symbol sequence, even if it expects a byte field (assuming the byte fields are separated from each other). If this happens, detection of the break/synch sequence shall abort the transfer in progress5 and processing of the new frame shall commence. Protected identifier consists of two sub-fields; the identifier and the identifier parity. Bit 0 to 5 is the identifier and bit 6 and 7 is the parity. Identifier Six bits are reserved for the identifier (ID), values in the range 0 to 63 can be used.

The identifiers are split in four categories:

- Values 0 to 59 (0x3b) are used for signal-carrying frames,
- 60 (0x3c) and 61 (0x3d) are used to carry diagnostic data,
- 62 (0x3e) is reserved for user-defined extensions,
- 63 (0x3f) is reserved for future protocol enhancements.

Data:

A frame carries between one and eight bytes of data. The number of bytes contained in a frame with a specific identifier shall be agreed by the publisher and all subscribers. A data byte is transmitted in a byte field, For data entities longer than one byte, the entity LSB is contained in the byte sent first and the entity MSB in the byte sent last (little-endian). The data fields are labeled data 1, data 2...up to maximum data 8.

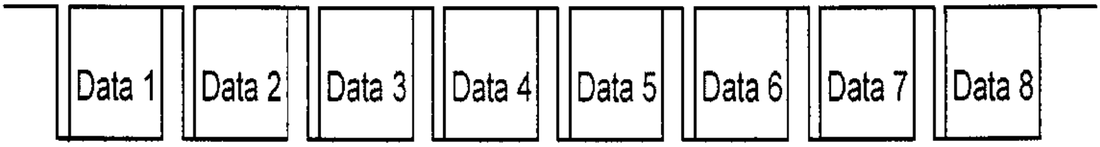


FIG 2.7 DATA BYTES IN A FRAME

Checksum:

The last field of a frame is the checksum. The checksum contains the inverted eight bit sum, with carry of all data bytes and the protected identifier. Checksum calculation over the data bytes only is called classic checksum and it is used for communication with LIN 1.3 slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and it is used for communication with LIN 2.0 slaves. Use of classic or enhanced checksum is managed by the master node and it is determined per frame identifier; classic in communication with LIN 1.3 slave nodes and enhanced in communication with LIN 2.0 slave nodes. Identifiers 60 (0x3c) to 63 (0x3f) shall always use classic checksum.

Frame slots:

Each scheduled frame allocates a slot on the bus. The duration of a slot must be long enough to carry the frame even in the worst case. The nominal value for transmission of a frame exactly matches the number of bits sent, i.e. no response space, no byte spaces and no inter-frame space.

Therefore:

$$T_{Header\_Nominal} = 34 * T_{Bit} \quad (2.1)$$

$$T_{Response\_Nominal} = 10 * (N_{Data} + 1) * T_{Bit} \quad (2.2)$$

$$T_{Frame\_Nominal} = T_{Header\_Nominal} + T_{Response\_Nominal} \quad (2.3)$$

Where T_{Bit} is the nominal time required to transmit a bit, as defined in LIN Physical Layer.

The maximum space between the bytes is an additional 40% duration compared to the nominal transmission time. The additional duration is split between the frame header (the master task) and the frame response (a slave task).

These yields:

$$T_{Header\_Maximum} = 1.4 * T_{Header\_Nominal} \quad (2.4)$$

$$T_{Response\_ Maximum} = 1.4 * T_{Response\_Nominal} \quad (2.5)$$

$$T_{Frame\_ Maximum} = T_{Header\_Maximum} + T_{Response\_ Maximum} \quad (2.6)$$

Each frame slot shall be longer than or equal to TFrame_Maximum for the frame specified.

P-3543
MAHATMA GANDHI
KCT
CBE-6
CENTRAL LIBRARY

2.1.6. FRAME TYPES:

The frame type refers to the pre-conditions that shall be valid to transmit the frame. There are three types of frames namely:

- 1. Unconditional frames,
- 2. Sporadic frames and
- 3. Event triggered frames.

All bits not used/defined in a frame shall be recessive (ones).

Event triggered frame:

The purpose of an event triggered frame is to increase the responsitivity of the LIN cluster without assigning too much of the bus bandwidth to the polling of multiple slave nodes with seldom occurring events.

Event triggered frames carry the data field of one or more unconditional frames and the identifier of an event triggered frame shall be in the range 0 to 59 (0x3b). The first data byte of the carried unconditional frame shall be equal to its protected identifier. This implies that, at maximum, seven bytes of signals can be carried.

If more than one unconditional frame is associated with one event triggered frame which is the normal case) they shall all be of equal length, use the same checksum model (i.e. Mixing LIN 1.3 and LIN 2.0 frames is incorrect) and, furthermore, they shall all be published by different slave tasks.

The header of an event triggered frame is normally transmitted (the conditions are explained below) when a frame slot allocated to the event triggered frame is processed. The publisher of an associated unconditional frame shall only provide the response to the header if one of the signals carried by its frame is updated.

If none of the slave tasks respond to the header, the rest of the frame slot is silent and the header is ignored.

If more than one slave task responds to the header in the same frame slot, a collision will occur. The master has to resolve the collision by requesting all associated unconditional frames before requesting the event-triggered frame again.

If one of the colliding slave nodes withdraws without corrupting the transfer, the master will not detect this. A slave must therefore retry sending its response until successful, or the response would be lost.

All subscribers of the event triggered frame shall receive the frame and use its data (if the checksum is validated) as if the associated unconditional frame was received.

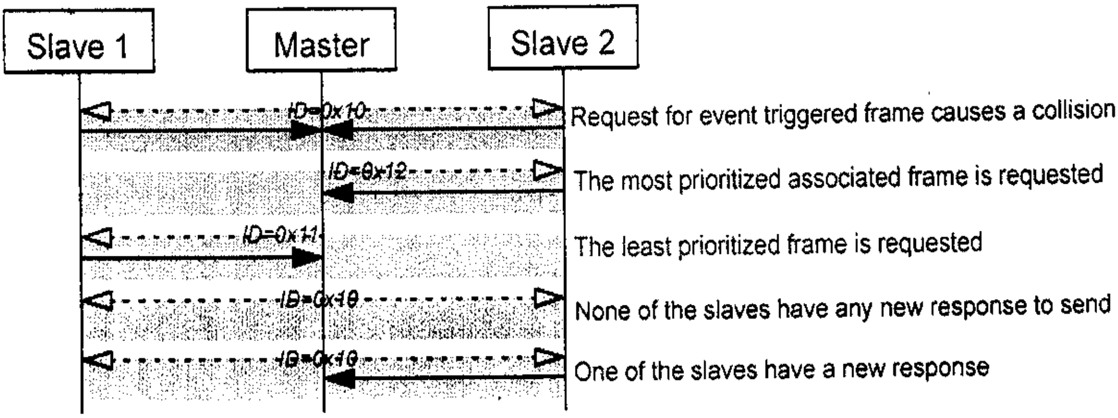

Slave 1
Master
Slave 2
ID=0x10
Request for event triggered frame causes a collision
ID=0x12
The most prioritized associated frame is requested
ID=0x11
The least prioritized frame is requested
ID=0x10
None of the slaves have any new response to send
ID=0x10
One of the slaves have a new response

FIG 2.8 ID 0x10 is an event triggered frame associated with the unconditional frames 0x11 and 0x12. Between each of the five frame slots in the figure, other frames may be transferred.

2.1.7. TASK BEHAVIOR MODEL:

The behavior model is based on the master task/slave task concept. It is not necessary to implement a master node with three independent state machines or a slave node with two independent state machines; they may very well be merged into one block per node.

Master task state machine:

The master task is responsible for generating correct headers, i.e. deciding which frame shall be sent and for maintaining the correct timing between frames, all according to the schedule table.

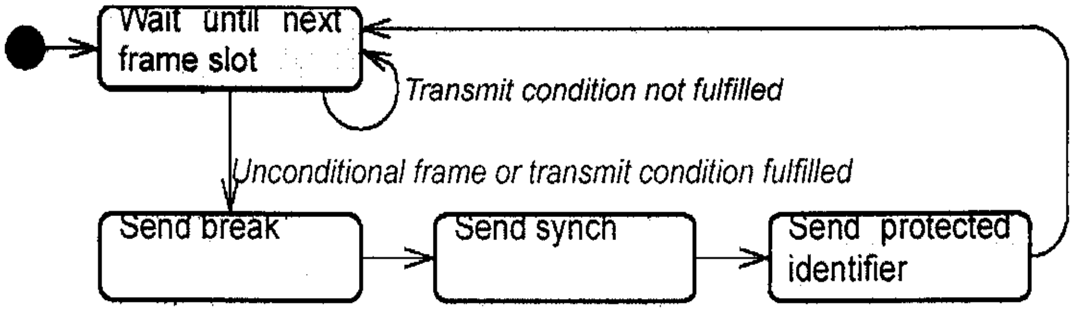

Wait until next frame slot
Transmit condition not fulfilled
Unconditional frame or transmit condition fulfilled
Send break
Send synch
Send protected identifier

FIG 2.9 COMPLETE STATE MACHINE FOR THE MASTER TASK

Slave task state machine:

The slave task is responsible for transmitting the frame response when it is the publisher and for receiving the frame response when it is a subscriber. The slave task is modelled with two state machines:

- Break and synch detector
- Frame processor

Break and synch detector:

A slave task is required to be synchronized at the beginning of the protected identifier field of a frame. It must stay synchronized within the required bit-rate tolerance throughout the remainder of the frame. For this purpose every frame starts with a sequence starting with break field followed by a synch byte field. This sequence is unique in the whole LIN communication and provides enough information for any slave task to detect the beginning of a new frame and be synchronized at the start of the identifier field.

Frame processor:

Frame processing consists of two states, Dormant and Active. Active contains five sub-states. As soon as Break and Synch is signaled the Active state is entered in the Receive Identifier sub-state. This implies that processing of one frame will be aborted by the detection of a new break and synch sequence. A mismatch between read back and sent data shall be detected not later than after completion of the byte field containing the mismatch. When a mismatch is detected, the transmission shall be aborted.

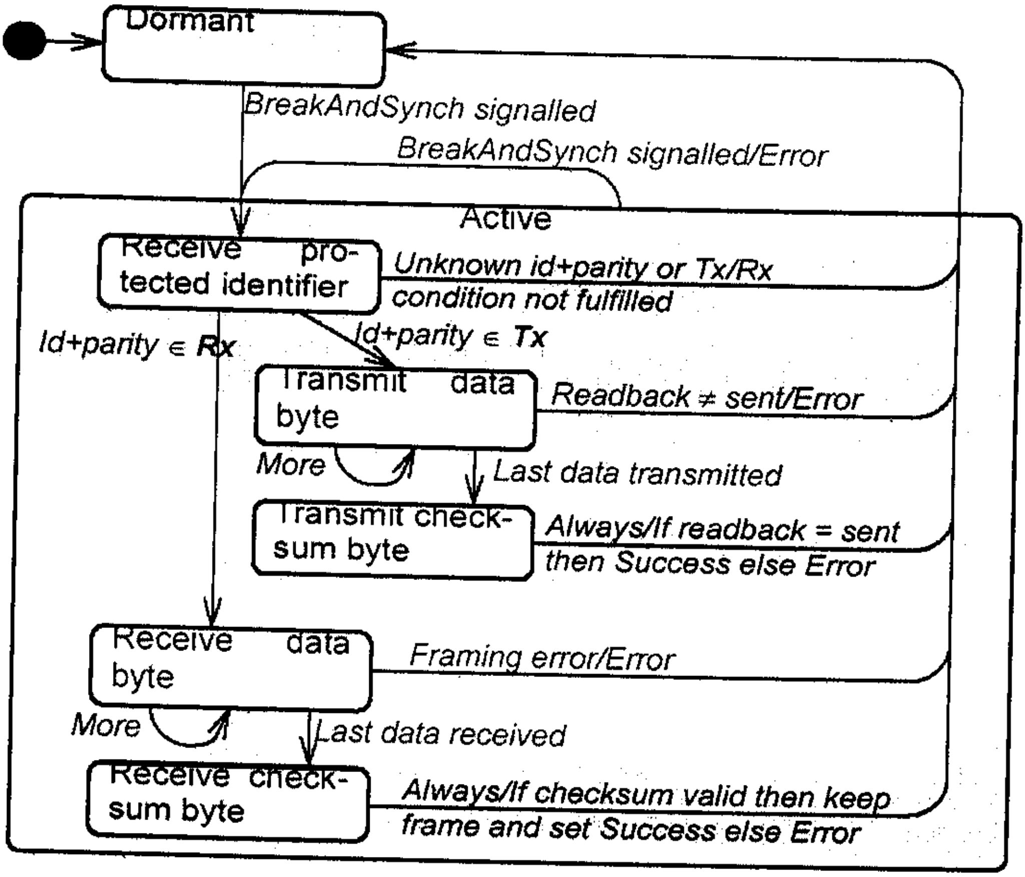


FIG 2.10 FRAME PROCESSOR STATE MACHINE

2.2 LIN OPERATION

2.2.1. OVERVIEW:

The LIN (Local Interconnect Network) is a general-purpose serial data communication interface for performing synchronous or asynchronous communication (start-stop synchronization with external devices. LIN-UART provides bidirectional communication function (normal mode), master-slave communication function (multiprocessor mode in master/slave systems) and special features for LIN bus system.

2.2.2. FUNCTIONS OF LIN:

It is a general purpose serial data communication interface for transmitting serial data to and receiving data from another CPU and peripheral devices. It has the functions as follows

Data buffer	Full duplicate double-buffer
Serial input	Perform over sampling 5 times and determine the received value by majority decision of sampling value (asynchronous mode only)
Transfer mode	• Clock synchronous (selecting start/stop synchronous or start/stop bit) • Clock asynchronous (start/stop bit can be used)
Baud rate	• Dedicated baud rate generator (the baud rate is consisted of 15-bit reload counter) • An external clock can be inputted and also be adjusted by reload counter
Data length	• 7 bit (other than synchronous or LIN mode)

	• 8 bit
Signaling	NRZ
Start bit timing	Synchronization to the falling edge of the start bit in the asynchronous mode
Detection of receive error	• Framing • Overrun • Parity
Interrupt request	• Receive interrupt (receive termination, detection of receive error, LIN sync break detection) • Transmit interrupt (transmit data empty) • Interrupt request to ICU (LIN synchronous field detection :LSYN)
Master/slave type communication function (multiprocessor mode)	This function enables communication between 1 master and 'n' slave
Synchronous mode	Master or slave function
LIN bus option	• Master device operation • Slave device operation • LIN sync break detection • LIN sync break generation.

TABLE 2.1 FUNCTION OF LIN

The LIN operates in 4 different modes, which are determined by the MDO and the MD1 bits of the LIN serial mode register (SMR).

2.2.3. LIN OPERATION MODES:

Operation mode	Data length		Synchronous method	Stop bit length	Data bit format
	without parity	with parity			
0-normal mode	7 bit or 8 bit		Asynchronous	1 bit Or 2 bits	LSB first MSB first
1-multiprocessor mode	7 or 8 bits+1	-	Asynchronous		
2-normal mode	8 bit		Synchronous	None,1 bit,2 bits	
3-LIN mode	8 bits	-	Asynchronous	1 bit	LSB first

TABLE 2.2 LIN OPERATION MODES

2.2.4. NOTES ON USING LIN-UART:

- Enabling operations- TXE-transmit enable
  RXE-receive enable } both enabled before transmission in SCR (manual)
- Communication mode setting- when LIN-UART communication not operating-set :TX/RX data-not guaranteed

- Transmit interrupt enabling timing

  Transmission data empty flag TDRE=1 (initial value)

  not manual

  TIR generated (TX interrupt request) - This happens only

  when TIE=1(manual)

  there must be no time delay in generation of TX interrupt request to avoid an immediate interrupt
- Changing operation setting

  If operation setting is changed (Eg: start/stop bit added or withdrawn), LIN-UART must be reset

  UPCL=1(in SMR)- programmable clear bit

  It doesn't change the registers but makes software to run from starting address.

  But even though after SMR is set, UPCL must be set
- Using LIN functions: mode 3

  8 bit, no parity, 1 stop bit, LSB first
- LIN slave settings: Baud rate must be set to slave, to ensure that at least 13 bits of LIN synch break is detected.
- Software compatibility- NOT
- Bus idle function- can be used
- AD bit (SCR – address/data type select bit)
- Software reset of LIN-UART : when TXE=0 execute UPCL=1
- Synch break detection: serial input=0 for 11 bit time

  LBR=1(ESCR reg)

  It enters into synch break detection mode, it recognizes synch break (LDB=1) and wait for synch field

2.2.5. INTERRUPTS OF LIN-UART:

Reception and transmission interrupt:

An interrupt request is generated for the following causes

- Receive data is set in the RDR, or a reception error occurs
- Transmitted data is transferred from the TDR to transmit to shift register and transmission is started
- LIN synch break is detected

The extended intelligent I/O service (EI^{2}OS) is available for these interrupts

Reception and transmission/ capture	Interrupt request flag bit	Flag register	Interrupt cause	Interrupt cause enable bit	Clearing interrupt request flag
Reception	RDRF	SSR	Receive data is written to RDR	SSR:RIE	Receive data is read
	ORE	SSR	Overrun error		“1” is written to reception error flag clear bit SCR:CRE
	FRE	SSR	Framing error		
	LBD	ESCR	LIN synch break detection	ESCR:LBIE	ESCR:LBD=”0”

Transmission	TDRE	SSR	Transmit register is empty	SSR:TIE	Writing transmit data

TABLE 2.3 INTERRUPTS OF LIN

Overrun error:

RDRF=1 & RDR was not read by the next serial data is received (ORE=1)

Framing error:

Stop bit reception error.

2.2.6. BAUD RATE SELECTION:

i. Dedicated baud rate generator (reload counter)
ii. I/P external clock to baud rate generator (reload counter)
iii. External clock (directly use SCKn pin input clock)

Among these we use dedicated baud rate generator in mode 3. It has 2 reload counters for transmission and reception serial clock. Baud rate selected through BGR. The reload counter divides the interrupt clock by set value. It is selected when SMR: EXT=0, OTO=0

2.2.7. LIN MASTER/SLAVE TYPE COMMUNICATION FUNCTION:

SCRn, SMRn

+	*	+	+	*	O	$	$	I	I	O	$	O	O	$	$

SSRn, TDRn/RDRn

*	$	$	$	$	+	$	$								

Set conversion data (data writing)

Retain reception data (during reading)

ESCRx, ECCRx

LBIE	$	$	$	$	$	O	O	/	$	*	*	*	/	$	$

TABLE 2.4 LIN MASTER/SLAVE TYPE COMMUNICATION FUNCTION

$- used bit 0/1

*- unused bit

I- set to "1"

o- set to "0"

+- bit automatically set correctly

2.2.8. OPERATION WITH LIN FUNCTION:

Lin master:

LBR='1'→break generate→ SOTn written with LIN synch break→then 55_H (synch field) written to TDR→while all these happens TDRE="0". After synchronization is transmitted TDRE="1"

Lin slave:

Slave has to sync to the master's baud rate

RXE=0   LBIE=1 → reception interrupt will be generated

LBD=1

When "0" is written to LBD, reception interrupt request flag is cleared.

2.2.10. DATA FRAME:

Cassette player control: 2 bytes

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
			VOLUME UP/DOWN				
ON/OFF	<<	MODE	[illegible]				

Door control: 1 byte

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
[illegible]	[illegible]		WINDOW UP/DOWN				

TABLE 2.5 DATA FRAMES

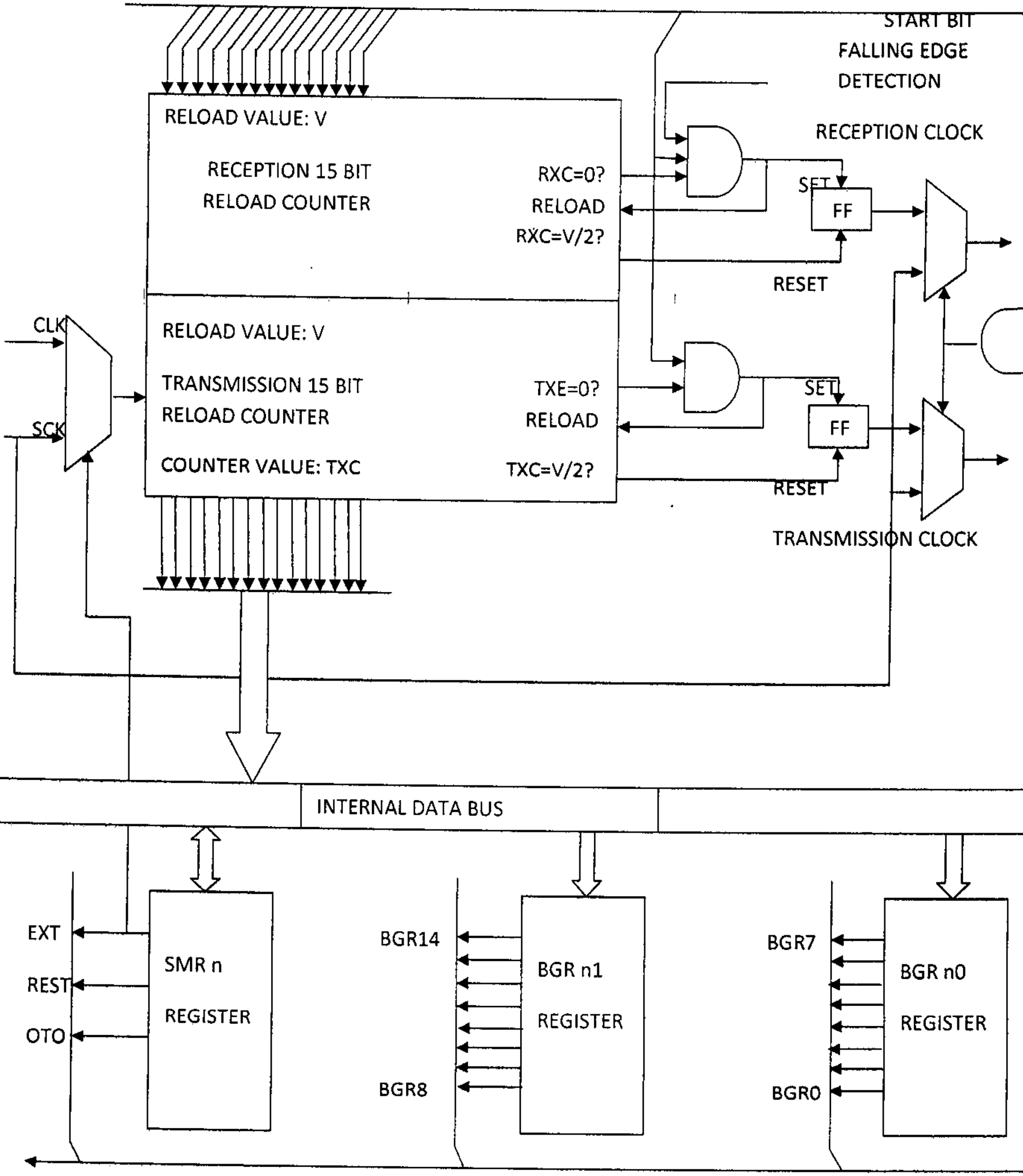


FIG 2.11 BAUD RATE SELECTION CICIRCUIT OF LIN

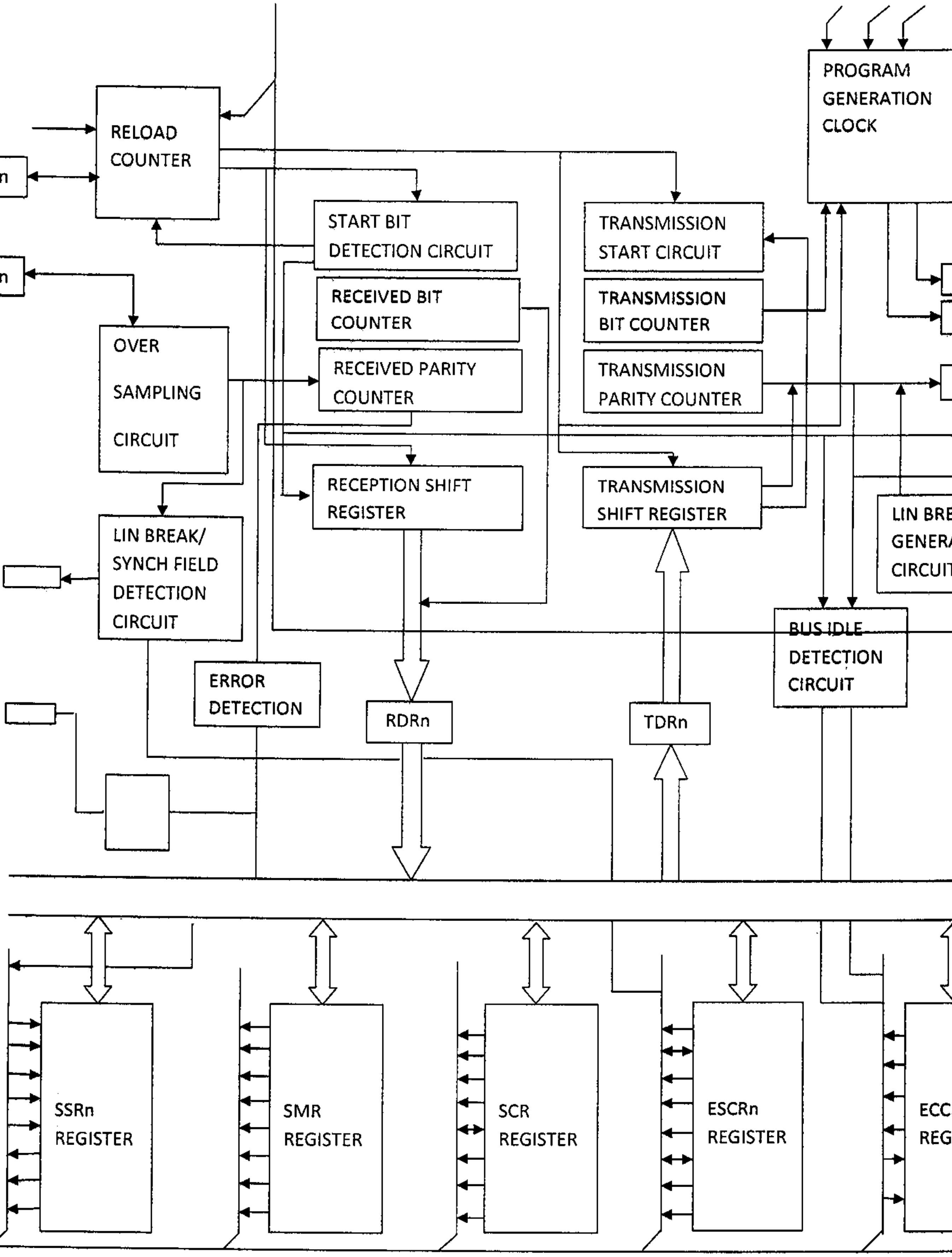


FIG 2.12 LIN BLOCK DIAGRAM

2.3 LIN REGISTERS

1. SCR: Serial Control Register

Specifies parity, selects stop bit and data lengths, selects frame data format in mode 1, and clears Reception error flag, enable/disable TX/RX.

Initial value: 0000 0000

Bit	15	14	13	12	11	10	9	8
	PEN	P	SBL	CL	AD	CRE	RXE	TXE
	R/W	R/W	R/W	R/W	R/W	W	R/W	R/W

TABLE 2.6 SERIAL CONTROL REGISTER

TXE: Transmission Enable bit

0 – Disable 1 – Enable

RXE: Reception Enable bit

0 – Disable 1 – Enable

CRE: Clear Reception Error bit

0 – No effect 1 – Clear error

AD: Address/ Data

0 – Data frame 1 – Address frame

CL: Data length

0 – 7 bit 1 – 8 bit

SBL: Stop Bit Length

0 – 1 bit   1 – 2 bit

P: Parity select

0 – Even parity   1 – Odd parity

PEN: Parity Enable Bit

0 – w/o parity   1 – with parity

SCR is set to (12) H

2. SMR: SERIAL MODE REGISTER

Selects baud rate, mode of operation and whether to enable / disable output of serial data & clock to the corresponding pin.

Initial value: 0000 0000

Bit	7	6	5	4	3	2	1	0
	MD1	MD0	OTO	EXT	REST	UPCL	SCKE	SOE
	R/W	R/W	R/W	R/W	W	W	R/W	R/W

TABLE 2.7 SERIAL MODE REGISTER

SOE: Serial Data o/p Enable bit

0 – General purpose   1 – For LIN

SCKE: Serial Clock o/p Enable bit

0 – General purpose   1 – Serial Clock o/p

UPCL: Programmable clear bit

0 – No effect 1 – Reset LIN

REST: Reload counter restart bit

0 – No effect 1 – Reset

EXT: External serial clock source select bit

0 – Reload counter used 1 – External clock source used

OTO: one-to-one external clk. i/p enable bit

0 – Reload counter used 1 – External clk. Used as it is

MD1	MD0	
0	0	mode 0 (Asyn. Normal)
0	1	mode 1 (Asyn. Multiprocessor)
1	0	mode 2 (Synchronous)
1	1	mode 3 (Asyn. LIN)

SMR is set to (C9) H

3. SSR: SERIAL STATUS REGISTER

Bit 15	14	13	12	11	10	9	8
PE	OVE	FRE	RDRE	TDRE	BDS	RIE	TIE
R	R	R	R	R	R/W	R/W	R/W

TABLE 2.8 SERIAL STATUS REGISTER

OVE: Over-run Error flag

0 – Empty 1 - Exists

FRE: Framing Error flag

0 – Empty   1 - Exists

RDRE: Receive Data Register Empty flag

0 – Empty   1 – Data exists

TDRE: Transmit Data Register Empty flag

0 – Data exists   1 – Empty

BDS: 0 – LSB first   1 – MSB first

RIE: Receive Interrupt Enable bit

0 – Disable   1 – Enable

TIE: Transmission Interrupt Enable bit

0 – Disable   1 – Enable

SSR is set to (0A) H

4. BGR: BAUD RATE GENERATION REGISTER

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0

TABLE 2.9 BAUD RATE GENERATION REGISTER

Baud Rate: 19200 bps

External clock frequency: 4 MHz

Machine clock: 4*4= 16 MHz

Reload value V= (machine clock/baud rate)-1

$$= ((16*10^{\wedge}(6))/19200)-1$$

$$= (340)\ H$$

5. ESCR: EXTENDED STATUS CONTROL REGISTER

Initial value: 0000 0100

Bit 15	14	13	12	10	9	8	7
LBIE	LBD	LBL1	LBL0	SOPE	SIOP	CCO	SCES
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TABLE 2.10 EXTENDED STATUS CONTROL REGISTER

LBIE: LIN Synch. Break detection Interrupt Enable bit

0 – Disable 1 – Enable

LBD: LIN Break Detecting flag bit

0 – Not detected 1 – Detected

LBL1, LBL0: LIN break length select bits

0	0	13 bit length
0	1	14 bit length
1	0	15 bit length
1	1	16 bit length

SOPE: Serial Output Pin direct access Enable bit

0 - Serial o/p access disabled 1 - Enabled

SIOP: Serial I/O Pin direct access setting bit

0 – Fixes “SOT” to 0 1 - Fixes “SOT” to 1

CCO: Continuous Clock Output enable bit (Mode2)

SCES: Sampling Clock Edge Selection bit (Mode 2)

ESCR is set to (0C) H

6. ECCR: EXTENDED COMMUNICATION CONTROL REGISTER

Initial value: 0000 00xx

Bit 7	6	5	4	3	2	1	0
Unused	LBR	MS	SCDE	SSM	unused	RB1	TB1
	W	R/W	R/W	R/W		R	R

TABLE 2.11 EXTENDED COMMUNICATION CONTROL REGISTER

LBR: LIN synchronous break generating bit

MS, SCDE, SSM: Used only in mode 2

RB1: Reception bus idle detection flag

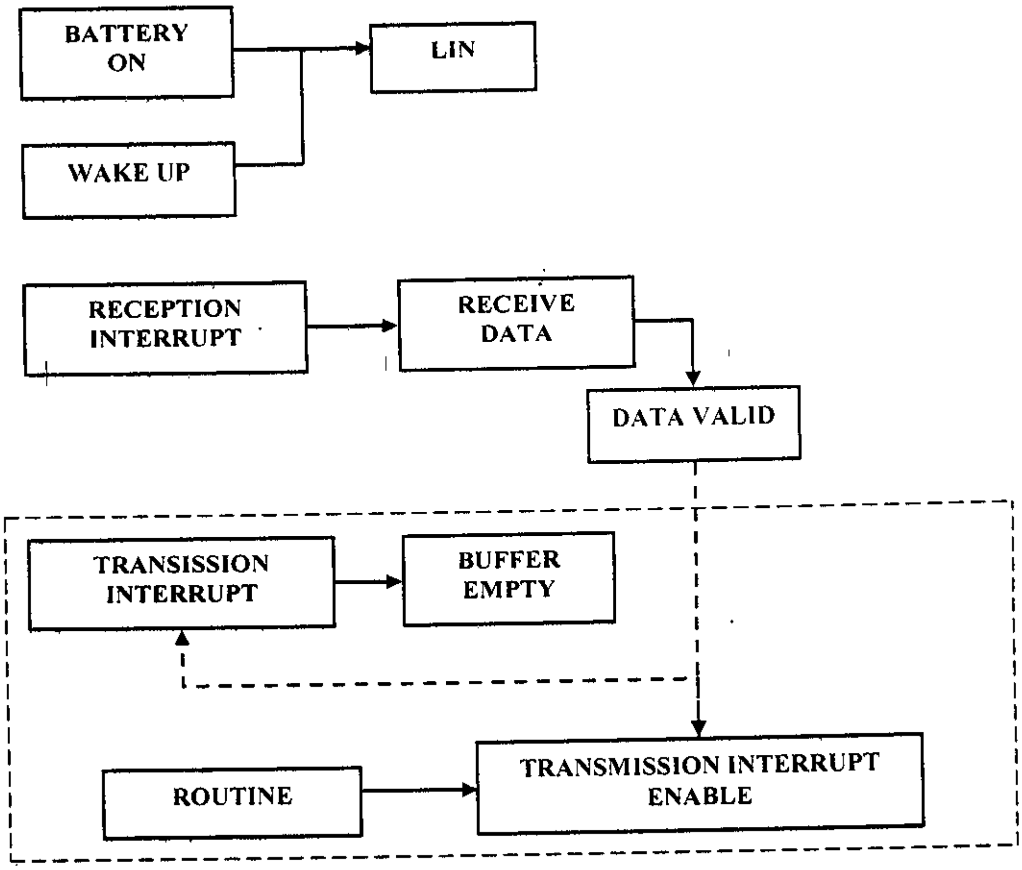
0 – Receiving 1 – No reception

TB1: transmission bus idle detection flag

0 – Transmitting 1 - No transmission

ECCR is set to (00) H

2.4 LIN SUB-ROUTINE



2.4.1. DESCRIPTION:

Initialization:

Whenever the battery is ON or a wakeup signal is received, the initial register configurations are made.

Status storage:

The slave continuously monitors the status of the switches and stores the current status in the data frames.

Reception interrupt:

Whenever the master sends the header information, a reception interrupt occurs. The header is received by each slave unit. If the ID in the header matches with the slave ID, that corresponding slave responds by sending the data i.e., the switch status in the response field.

Transmission interrupt:

When the data is loaded into the buffer, a transmission interrupt is generated and the information is transmitted to the master.

SCHEMATIC OVERVIEW

CHAPTER 3

SCHEMATIC OVERVIEW

3.1 BLOCK DIAGRAM

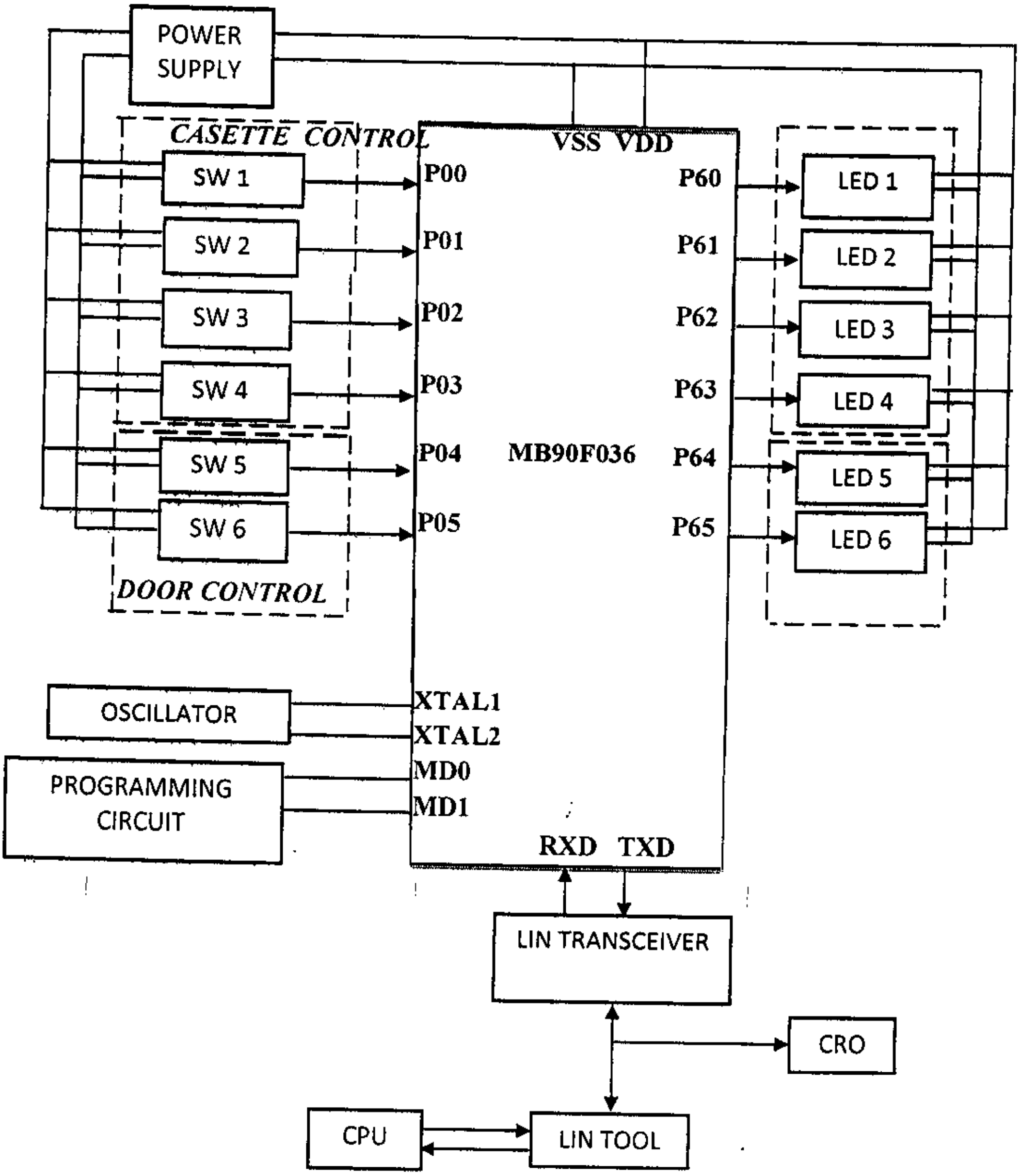


FIG 3.1 SCHEMATIC DIAGRAM

3.2 BLOCK DIAGRAM DESCRIPTION:

3.2.1. Microcontroller:

The Fujitsu MB90F036 controller is a 120 pin IC. Port pins P00 throug P06 are used as input port pins for switch interfacing circuits of cassette playe control and door control. Port pins P60 through P66 are used as output pins fo LED interfacing circuits of cassette player control and door control.

3.2.2. Power supply:

The power supply circuit converts AC supply to 5V DC. Each interfacin circuits and the controller is connected to power supply unit through V_{ss} and V_{d} pins.

3.2.3. Crystal oscillator:

A crystal oscillator circuit is connected to XTAL1 and XTAL2 pins of the controller. It provides 4 MHz frequency which is converted into 16 MHz by the PLL circuit in the microcontroller.

3.2.4. Lin transceiver:

The RXD and TXD pins of the controller are connected to LIN transceiver. This circuit is used to convert the double wire system from the microcontroller into single wire required for LIN protocol.

3.2.5. Logical operation:

The cassette player control consists of four switch circuits and four LED circuits; door control has three switch circuits and three LED circuits. In each switch circuits when supply is given, output will be high. When the switch is pressed, the output becomes low. The output of each switch circuit is given to the input port of the controller. The corresponding output port pin is given as the input to LED circuit. The LED circuit is programmed in such a way that when the output of the switch circuit is high, the input to LED circuit is high and LED glows.

Every time when switch is pressed, the corresponding bit in the data frame gets incremented or decremented according to their specific function and is stored. Header is transmitted from CPU through RS232 cable to LIN tool. It passes the signal from two wire system to single wire system and transmits to LIN transceiver. The LIN transceiver passes the signal from LIN wire to RXD pin of controller. When the header is received, the slave responds with the data frame in TXD pin which is then passed on through LIN wire. This LIN wire can be read through CRO.

CIRCUIT DIAGRAM

CHAPTER 4
CIRCUIT DIAGRAM

4.1 POWER SUPPLY:

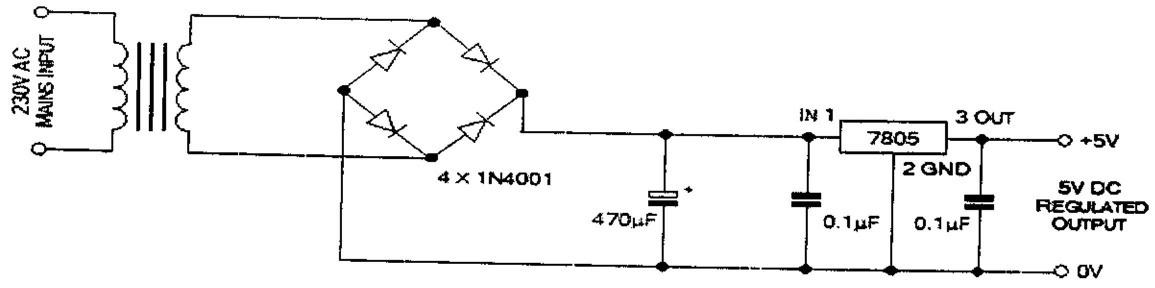


FIG 4.1 POWER SUPPLY

The power supply unit is used to convert the 230V AC supply into 5V DC supply. The step-down transformer step down the voltage level, the bridge rectifier is used to convert the AC into DC. The smoothing capacitor is used to remove the ripples in the DC. The regulator is used to give the required 5V.

4.2 LIN TRANSCEIVER:

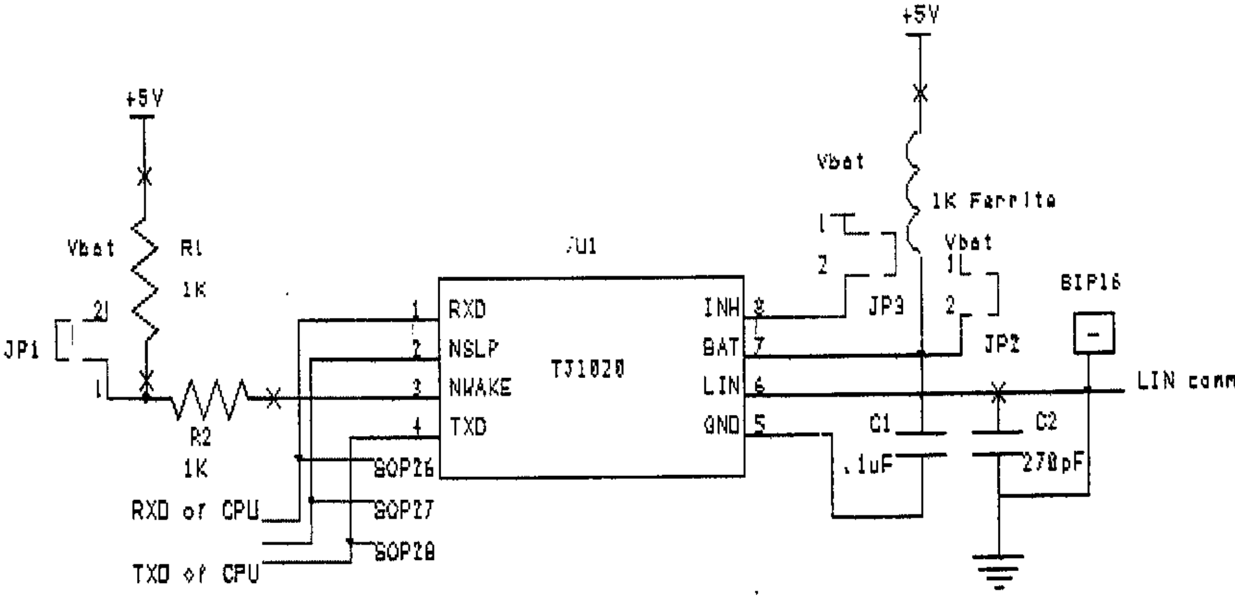


FIG 4.2 LIN TRANSCEIVER

The LIN transceiver is used to convert the double wire system into single wire system. The TJA1020 is the interface between the LIN master/slave protocol controller and the physical bus in a Local Interconnect Network (LIN). The transmit data stream of the protocol controller at the TXD input is converted by the LIN transceiver into a bus Signal. The receiver detects the data stream at the LIN bus input pin and transfers it via pin RXD to the microcontroller.

4.3 LED AND SWITCH INTERFACE:

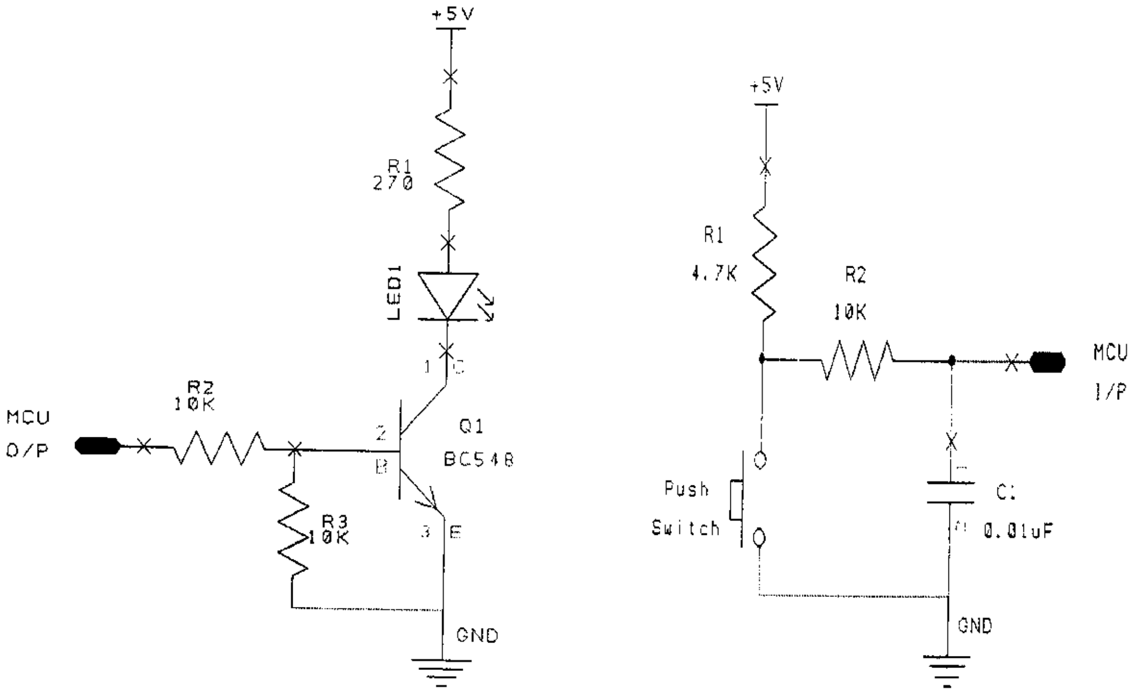


FIG 4.3 LED AND SWITCH INTERFACE

The port 6 pin of the microcontroller is given as the input to the base of the transistor of the LED interface. When the input from the controller goes low, the LED turns ON. The output of the switch circuit is connected to port 0 of the controller. When the switch is pressed, the output goes low.

4.4 CRYSTAL OSCILLATOR:

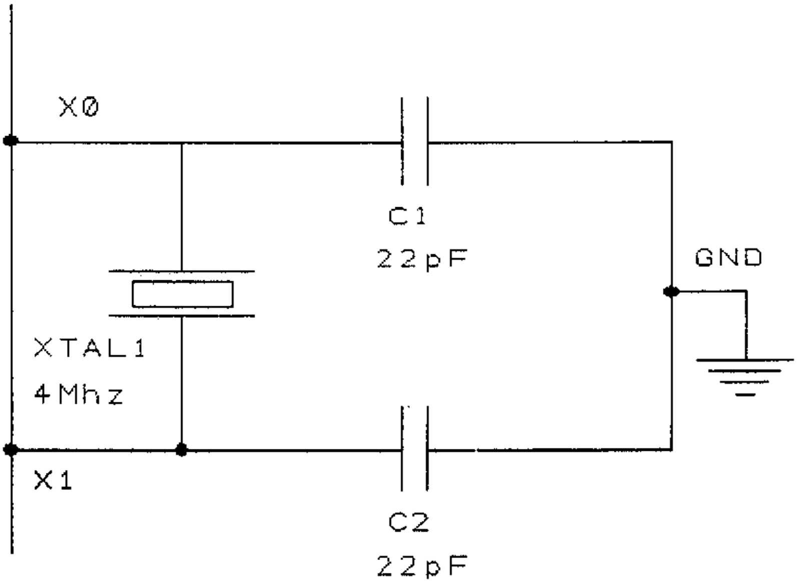


FIG 4.4 CRYSTAL OSCILLATOR

The crystal oscillator is used to provide 4 MHz frequency for the controller operation. This frequency is converted into the required 16 MHz by the PLL circuit which is in-built in the microcontroller.

4.5 PROGRAMMING CIRCUIT:

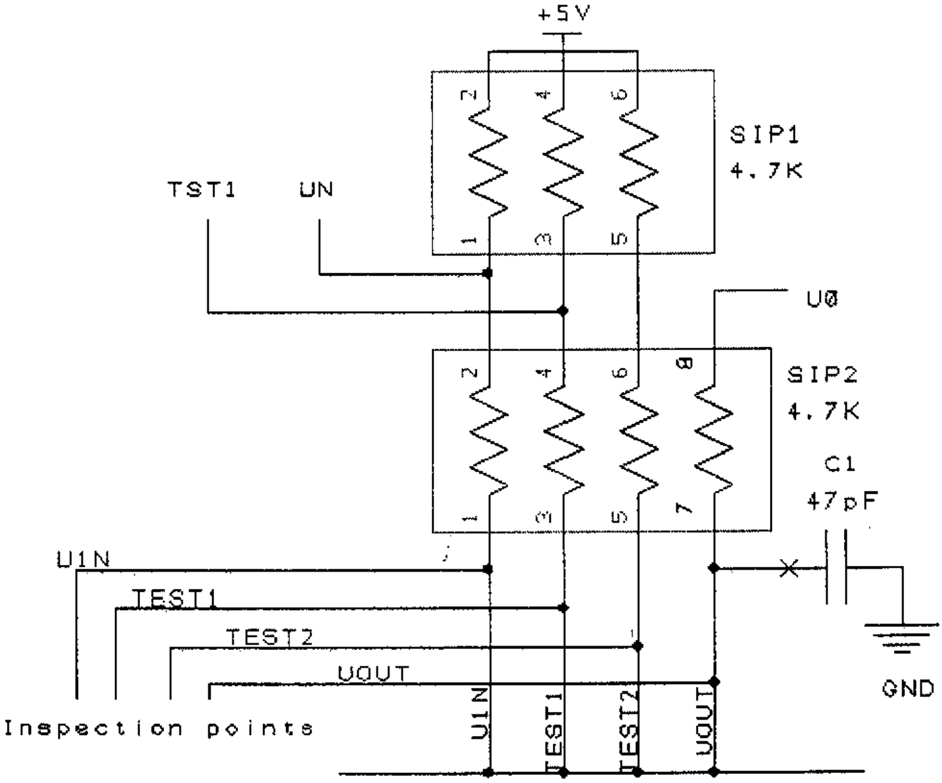


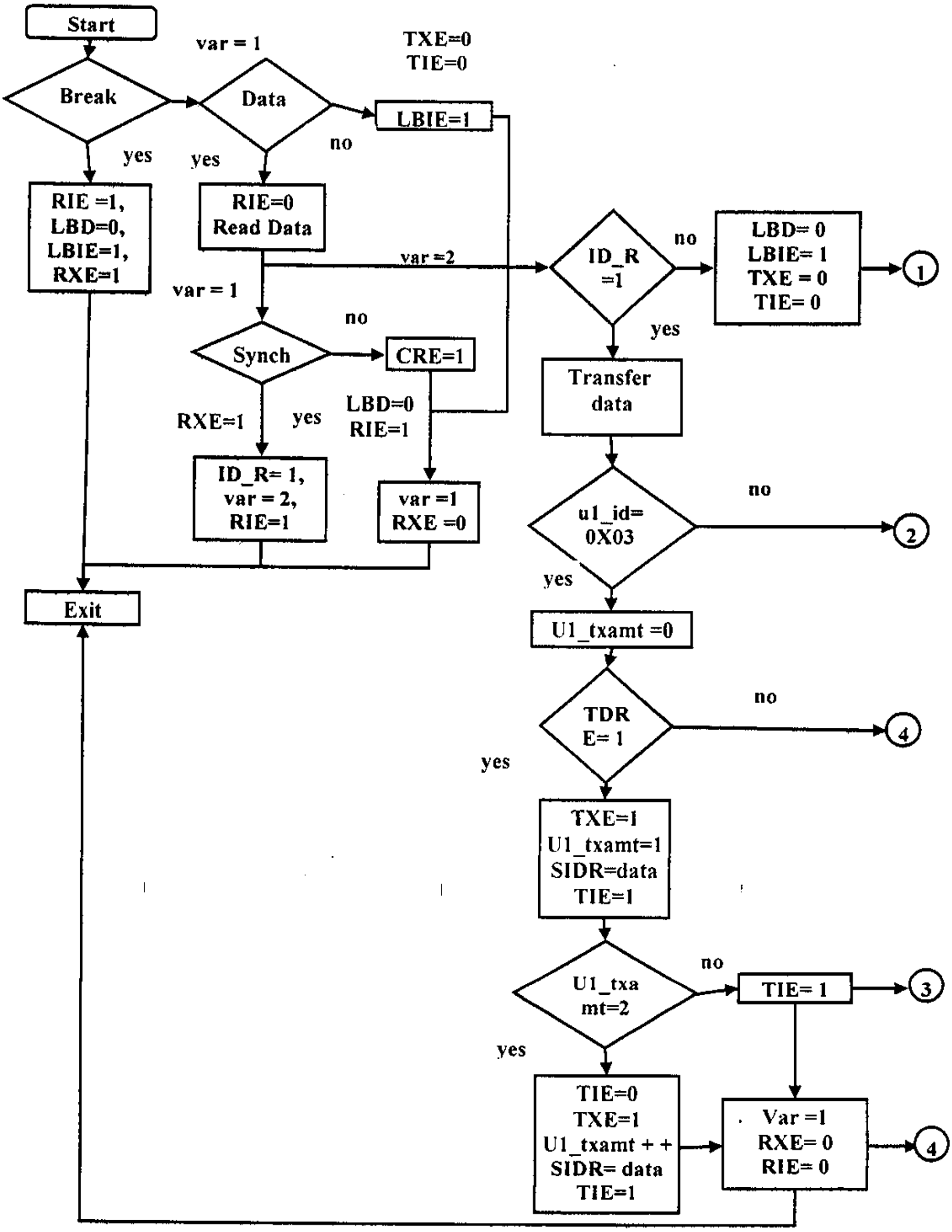
FIG 4.5 PROGRAMMING CIRCUIT

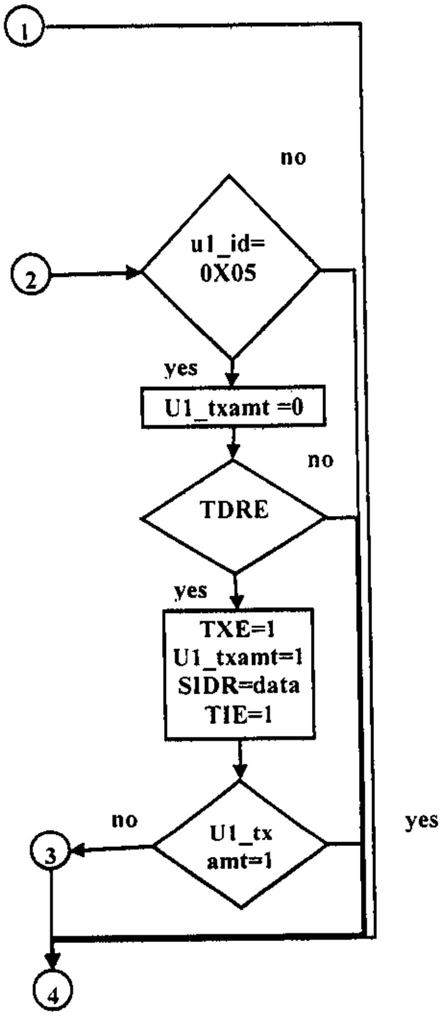
FLOW CHART

CHAPTER 5

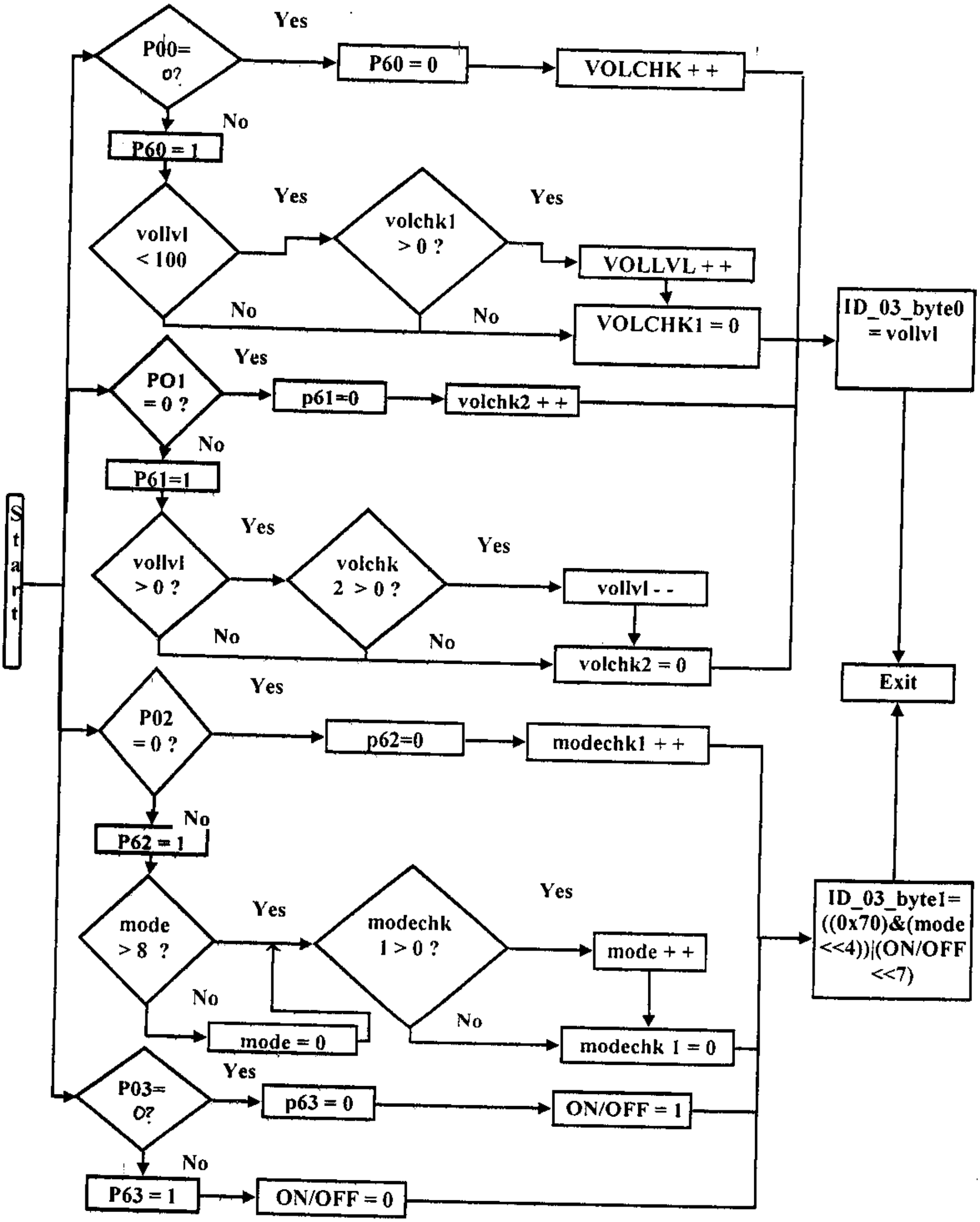
FLOW CHART

5.1 TRANSMISSION AND RECEPTION




1
no
2
u1_id=
0X05
yes
U1_txamt =0
no
TDRE
yes
TXE=1
U1_txamt=1
SIDR=data
TIE=1
no
3
U1_tx
amt=1
yes
4

5.2 INTERFACING PROGRAM FLOW CHART



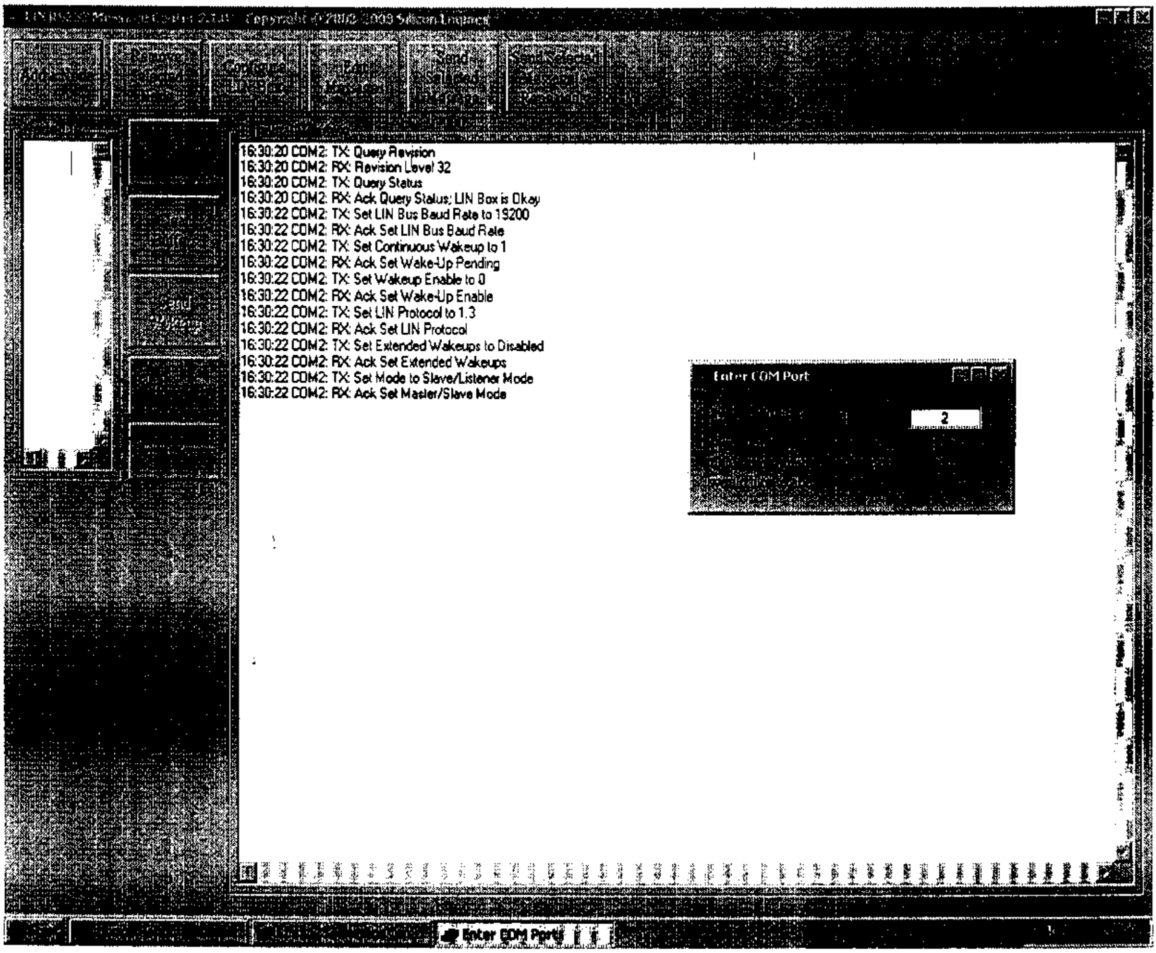
RESULTS

CHAPTER 6

RESULTS

6.1 LIN RS232 MESSAGE CENTER:

STEP 1: Including the COM port 2

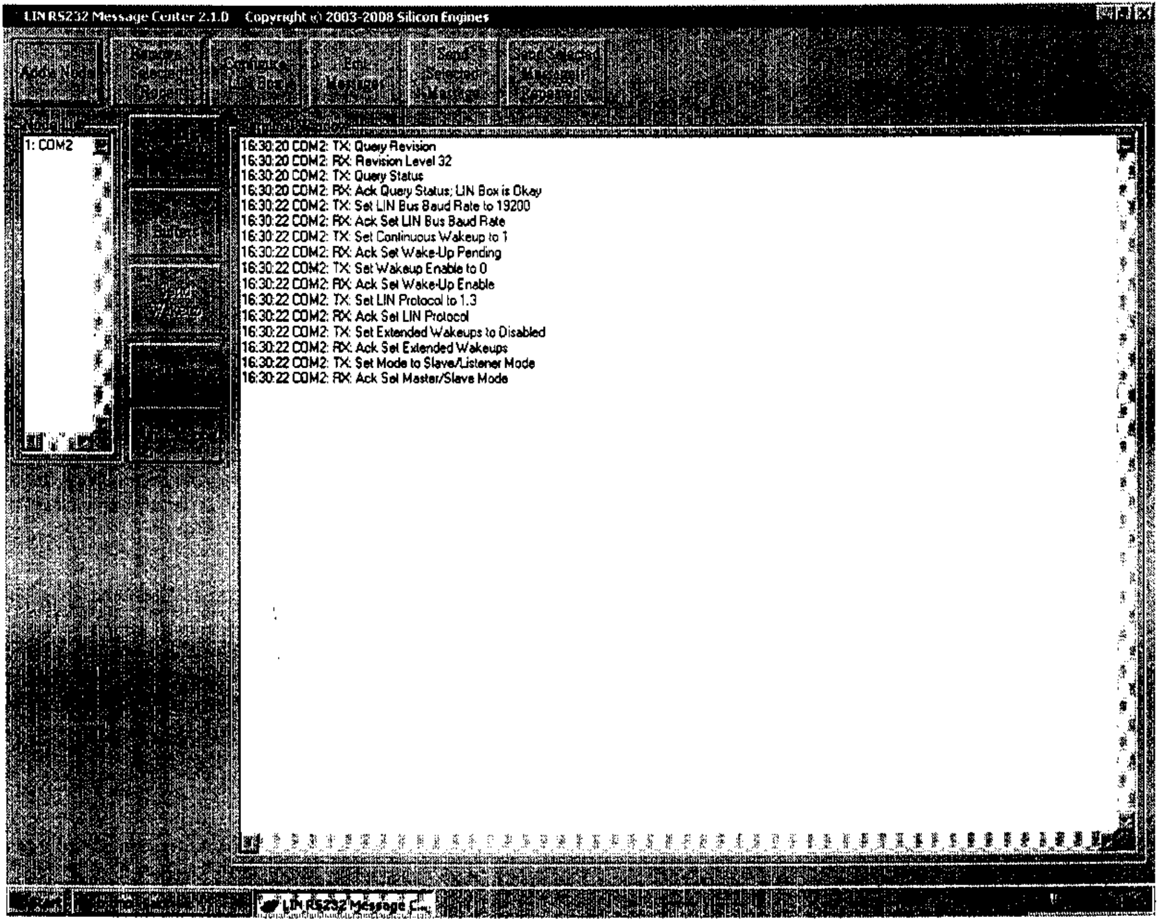


```
16:30:20 COM2: TX: Query Revision
16:30:20 COM2: RX: Revision Level 32
16:30:20 COM2: TX: Query Status
16:30:20 COM2: RX: Ack Query Status; LIN Box is Okay
16:30:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:30:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:30:22 COM2: TX: Set Continuous Wakeup to 1
16:30:22 COM2: RX: Ack Set Wake-Up Pending
16:30:22 COM2: TX: Set Wakeup Enable to 0
16:30:22 COM2: RX: Ack Set Wake-Up Enable
16:30:22 COM2: TX: Set LIN Protocol to 1.3
16:30:22 COM2: RX: Ack Set LIN Protocol
16:30:22 COM2: TX: Set Extended Wakeups to Disabled
16:30:22 COM2: RX: Ack Set Extended Wakeups
16:30:22 COM2: TX: Set Mode to Slave/Listener Mode
16:30:22 COM2: RX: Ack Set Master/Slave Mode
```

FIG 6.1 INCLUDING THE COM PORT

The COM port that is used for the communication between the CPU and the LIN tool is added.

COM 2 is now added:



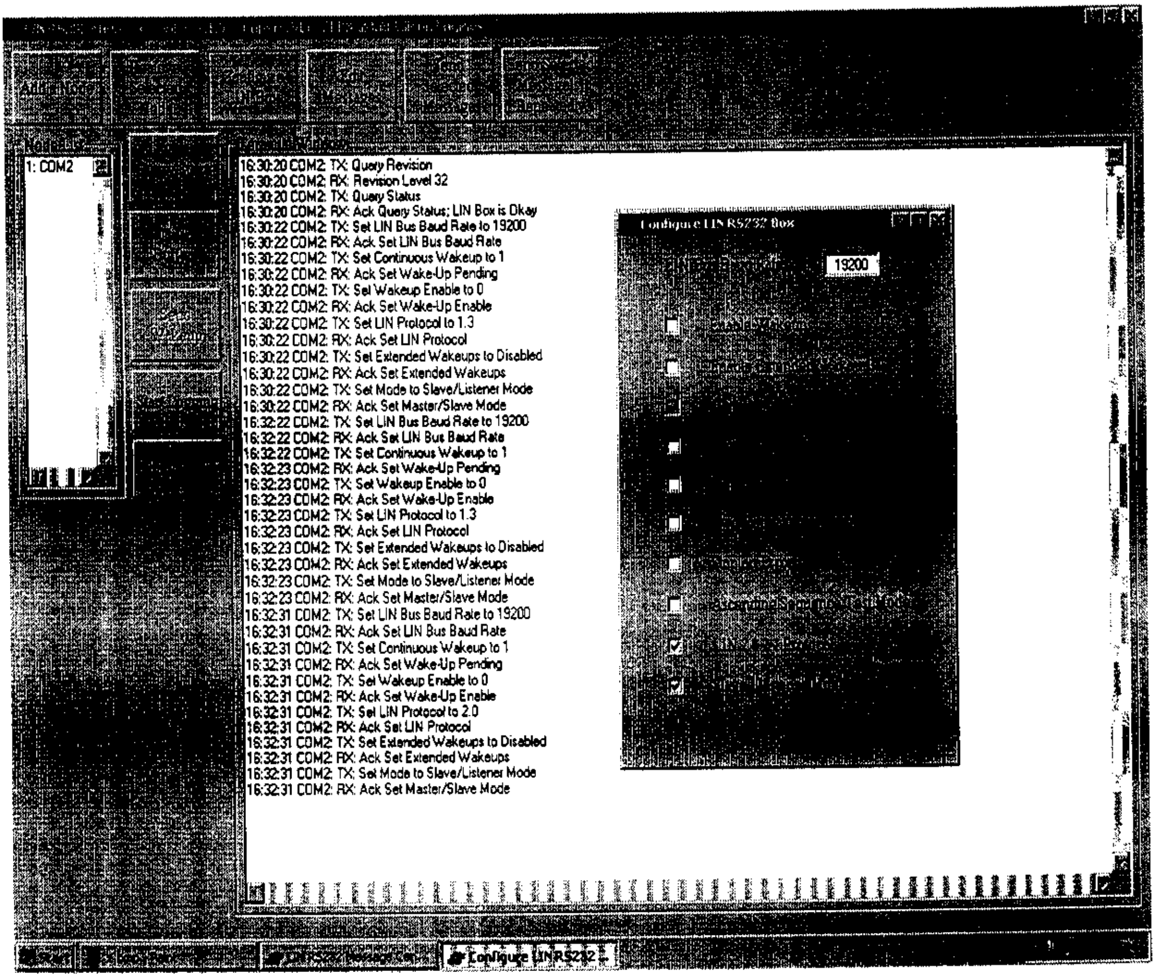
LIN RS232 Message Center 2.1.0 Copyright (c) 2003-2008 Silicon Engines
Add Node
1: COM2
LIN RS232 Message C...

```
16:30:20 COM2: TX: Query Revision
16:30:20 COM2: RX: Revision Level 32
16:30:20 COM2: TX: Query Status
16:30:20 COM2: RX: Ack Query Status; LIN Box is Okay
16:30:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:30:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:30:22 COM2: TX: Set Continuous Wakeup to 1
16:30:22 COM2: RX: Ack Set Wake-Up Pending
16:30:22 COM2: TX: Set Wakeup Enable to 0
16:30:22 COM2: RX: Ack Set Wake-Up Enable
16:30:22 COM2: TX: Set LIN Protocol to 1.3
16:30:22 COM2: RX: Ack Set LIN Protocol
16:30:22 COM2: TX: Set Extended Wakeups to Disabled
16:30:22 COM2: RX: Ack Set Extended Wakeups
16:30:22 COM2: TX: Set Mode to Slave/Listener Mode
16:30:22 COM2: RX: Ack Set Master/Slave Mode
```

FIG 6.2 MESSAGE CENTER WITH COM PORT ADDED

The added COM port 2 is shown in the Node List column.

STEP 2: Configuring LIN box



```
16:30:20 COM2: TX: Query Revision
16:30:20 COM2: RX: Revision Level 32
16:30:20 COM2: TX: Query Status
16:30:20 COM2: RX: Ack Query Status; LIN Box is Okay
16:30:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:30:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:30:22 COM2: TX: Set Continuous Wakeup to 1
16:30:22 COM2: RX: Ack Set Wake-Up Pending
16:30:22 COM2: TX: Set Wakeup Enable to 0
16:30:22 COM2: RX: Ack Set Wake-Up Enable
16:30:22 COM2: TX: Set LIN Protocol to 1.3
16:30:22 COM2: RX: Ack Set LIN Protocol
16:30:22 COM2: TX: Set Extended Wakeups to Disabled
16:30:22 COM2: RX: Ack Set Extended Wakeups
16:30:22 COM2: TX: Set Mode to Slave/Listener Mode
16:30:22 COM2: RX: Ack Set Master/Slave Mode
16:32:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:32:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:32:22 COM2: TX: Set Continuous Wakeup to 1
16:32:23 COM2: RX: Ack Set Wake-Up Pending
16:32:23 COM2: TX: Set Wakeup Enable to 0
16:32:23 COM2: RX: Ack Set Wake-Up Enable
16:32:23 COM2: TX: Set LIN Protocol to 1.3
16:32:23 COM2: RX: Ack Set LIN Protocol
16:32:23 COM2: TX: Set Extended Wakeups to Disabled
16:32:23 COM2: RX: Ack Set Extended Wakeups
16:32:23 COM2: TX: Set Mode to Slave/Listener Mode
16:32:23 COM2: RX: Ack Set Master/Slave Mode
16:32:31 COM2: TX: Set LIN Bus Baud Rate to 19200
16:32:31 COM2: RX: Ack Set LIN Bus Baud Rate
16:32:31 COM2: TX: Set Continuous Wakeup to 1
16:32:31 COM2: RX: Ack Set Wake-Up Pending
16:32:31 COM2: TX: Set Wakeup Enable to 0
16:32:31 COM2: RX: Ack Set Wake-Up Enable
16:32:31 COM2: TX: Set LIN Protocol to 2.0
16:32:31 COM2: RX: Ack Set LIN Protocol
16:32:31 COM2: TX: Set Extended Wakeups to Disabled
16:32:31 COM2: RX: Ack Set Extended Wakeups
16:32:31 COM2: TX: Set Mode to Slave/Listener Mode
16:32:31 COM2: RX: Ack Set Master/Slave Mode
```

FIG 6.3 CONFIGURING LIN BOX

The Lin tool is configured to operate with a baud rate of 19200 and using the LIN 2.2-2.1 protocol.

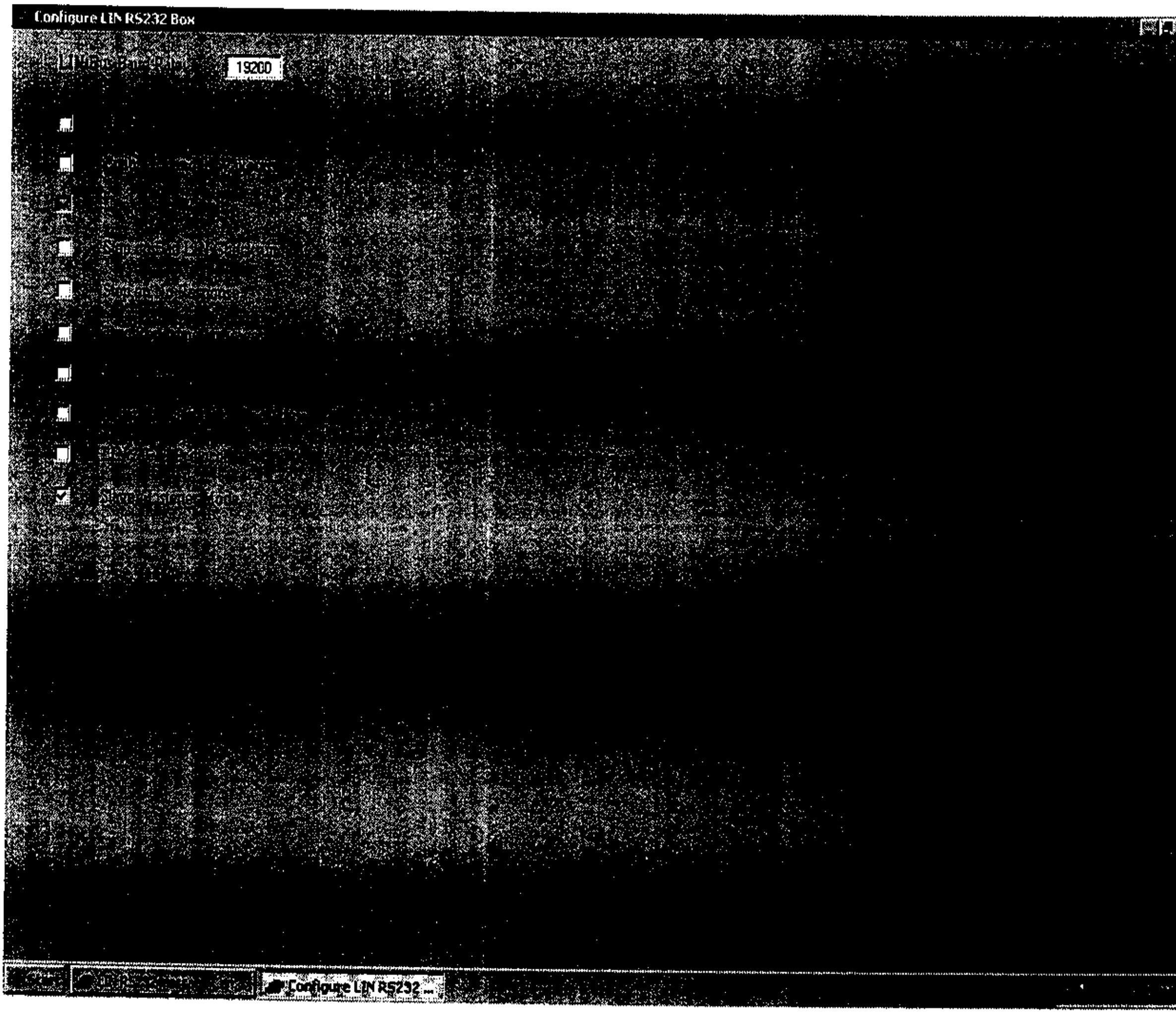
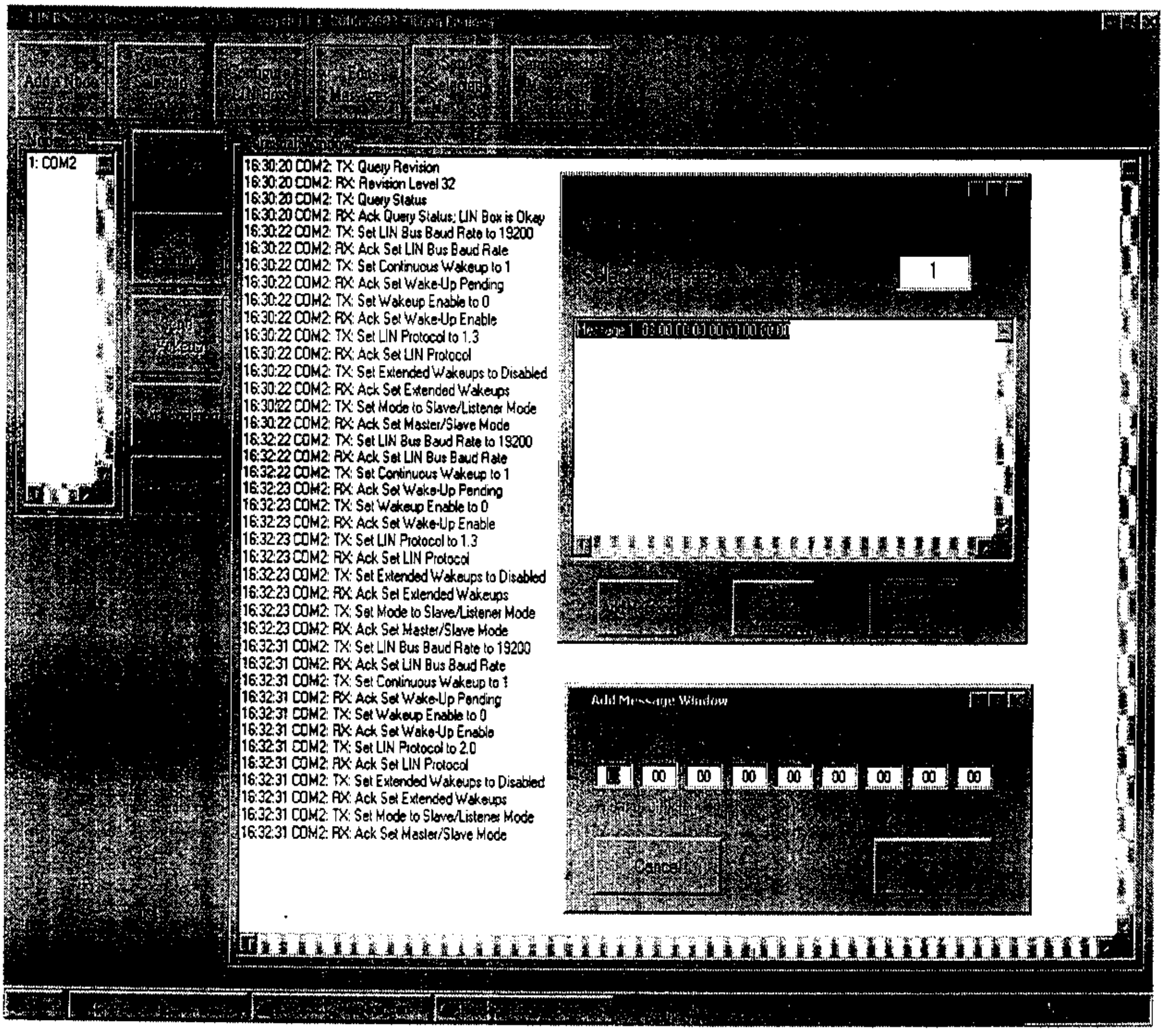


FIG 6.4 CONFIGURATION WINDOW

STEP 3: Add message to be transmitted



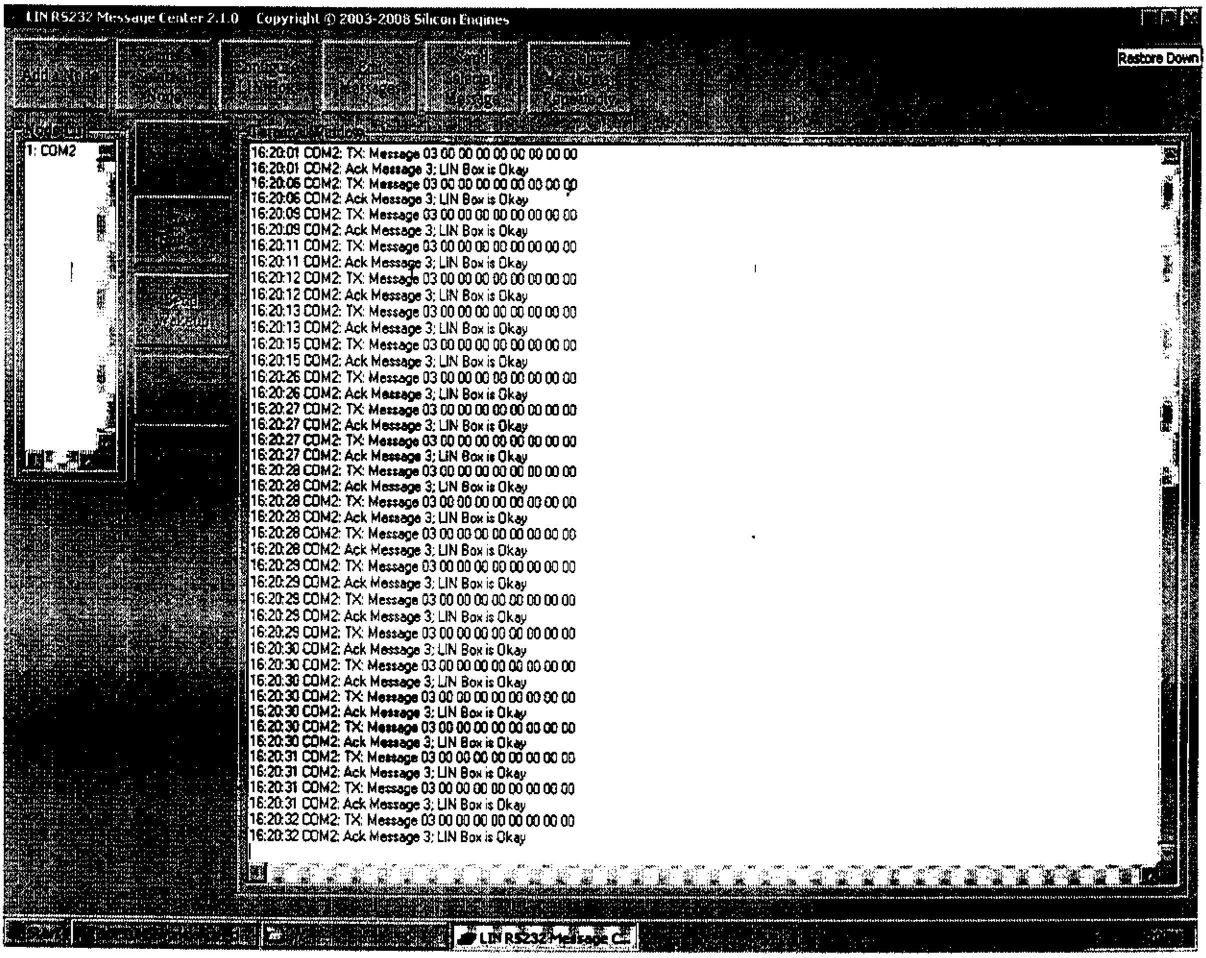
```
16:30:20 COM2: TX: Query Revision
16:30:20 COM2: RX: Revision Level 32
16:30:20 COM2: TX: Query Status
16:30:20 COM2: RX: Ack Query Status; LIN Box is Okay
16:30:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:30:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:30:22 COM2: TX: Set Continuous Wakeup to 1
16:30:22 COM2: RX: Ack Set Wake-Up Pending
16:30:22 COM2: TX: Set Wakeup Enable to 0
16:30:22 COM2: RX: Ack Set Wake-Up Enable
16:30:22 COM2: TX: Set LIN Protocol to 1.3
16:30:22 COM2: RX: Ack Set LIN Protocol
16:30:22 COM2: TX: Set Extended Wakeups to Disabled
16:30:22 COM2: RX: Ack Set Extended Wakeups
16:30:22 COM2: TX: Set Mode to Slave/Listener Mode
16:30:22 COM2: RX: Ack Set Master/Slave Mode
16:32:22 COM2: TX: Set LIN Bus Baud Rate to 19200
16:32:22 COM2: RX: Ack Set LIN Bus Baud Rate
16:32:22 COM2: TX: Set Continuous Wakeup to 1
16:32:23 COM2: RX: Ack Set Wake-Up Pending
16:32:23 COM2: TX: Set Wakeup Enable to 0
16:32:23 COM2: RX: Ack Set Wake-Up Enable
16:32:23 COM2: TX: Set LIN Protocol to 1.3
16:32:23 COM2: RX: Ack Set LIN Protocol
16:32:23 COM2: TX: Set Extended Wakeups to Disabled
16:32:23 COM2: RX: Ack Set Extended Wakeups
16:32:23 COM2: TX: Set Mode to Slave/Listener Mode
16:32:23 COM2: RX: Ack Set Master/Slave Mode
16:32:31 COM2: TX: Set LIN Bus Baud Rate to 19200
16:32:31 COM2: RX: Ack Set LIN Bus Baud Rate
16:32:31 COM2: TX: Set Continuous Wakeup to 1
16:32:31 COM2: RX: Ack Set Wake-Up Pending
16:32:31 COM2: TX: Set Wakeup Enable to 0
16:32:31 COM2: RX: Ack Set Wake-Up Enable
16:32:31 COM2: TX: Set LIN Protocol to 2.0
16:32:31 COM2: RX: Ack Set LIN Protocol
16:32:31 COM2: TX: Set Extended Wakeups to Disabled
16:32:31 COM2: RX: Ack Set Extended Wakeups
16:32:31 COM2: TX: Set Mode to Slave/Listener Mode
16:32:31 COM2: RX: Ack Set Master/Slave Mode
```

FIG 6.5 MESSAGE WINDOW

The message to be sent from the master to the slave unit i.e., header information with the corresponding ID of the slave is added.

P-3543
MAHATMA GANDHI
KCT
CBE
CENTRAL LIBRARY

STEP 4: Sending message(s) repeatedly


LIN RS232 Message Center 2.1.0 Copyright © 2003-2008 Silicon Engines
Restore Down
1: COM2
LIN RS232 Message C...

```
16:20:01 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:01 COM2: Ack Message 3; LIN Box is Okay
16:20:06 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:06 COM2: Ack Message 3; LIN Box is Okay
16:20:09 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:09 COM2: Ack Message 3; LIN Box is Okay
16:20:11 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:11 COM2: Ack Message 3; LIN Box is Okay
16:20:12 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:12 COM2: Ack Message 3; LIN Box is Okay
16:20:13 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:13 COM2: Ack Message 3; LIN Box is Okay
16:20:15 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:15 COM2: Ack Message 3; LIN Box is Okay
16:20:26 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:26 COM2: Ack Message 3; LIN Box is Okay
16:20:27 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:27 COM2: Ack Message 3; LIN Box is Okay
16:20:27 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:27 COM2: Ack Message 3; LIN Box is Okay
16:20:28 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:28 COM2: Ack Message 3; LIN Box is Okay
16:20:28 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:28 COM2: Ack Message 3; LIN Box is Okay
16:20:28 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:28 COM2: Ack Message 3; LIN Box is Okay
16:20:29 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:29 COM2: Ack Message 3; LIN Box is Okay
16:20:29 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:29 COM2: Ack Message 3; LIN Box is Okay
16:20:29 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:30 COM2: Ack Message 3; LIN Box is Okay
16:20:30 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:30 COM2: Ack Message 3; LIN Box is Okay
16:20:30 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:30 COM2: Ack Message 3; LIN Box is Okay
16:20:30 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:30 COM2: Ack Message 3; LIN Box is Okay
16:20:31 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:31 COM2: Ack Message 3; LIN Box is Okay
16:20:31 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:31 COM2: Ack Message 3; LIN Box is Okay
16:20:32 COM2: TX: Message 03 00 00 00 00 00 00 00 00
16:20:32 COM2: Ack Message 3; LIN Box is Okay
```

FIG 6.6 REPEATED MESSAGE SENDING

The messages from the master are sent continuously to the entire slave unit i.e., the microcontroller. The slave unit whose ID matches with the data in the ID field of the header, responds with the data.

6.2 OUTPUT

For cassette player control:

FIELD 1: BREAK (14 bit active low signal-680 us)

FIELD 2: SYNCHRONOUS (0x55(0101 0101))

FIELD 3: ID FIELD (0x03(0000 0011))

FIELD 4: DATA FIELD (byte 0: 0x0A (0000 1010), byte 1:0xB0 (1011 0000))

Each bit: 52 us

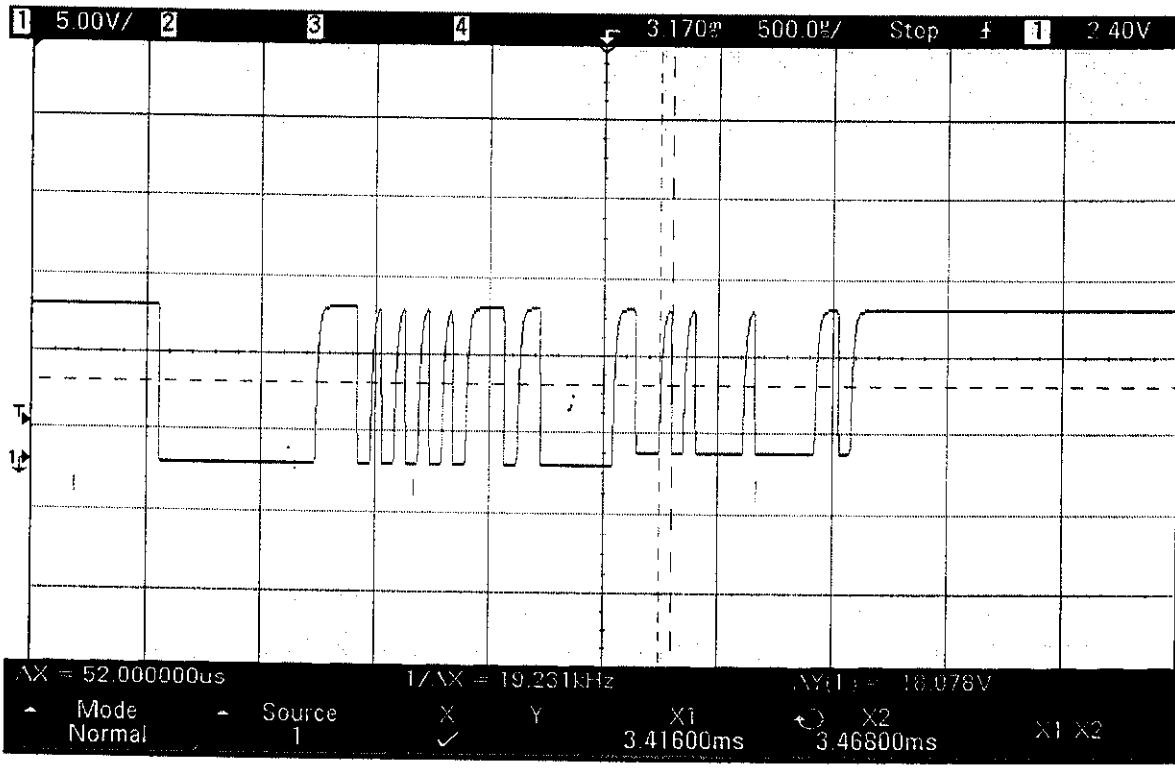


FIG 6.7 ID 3 SIGNAL

Entire frame with break field timing:

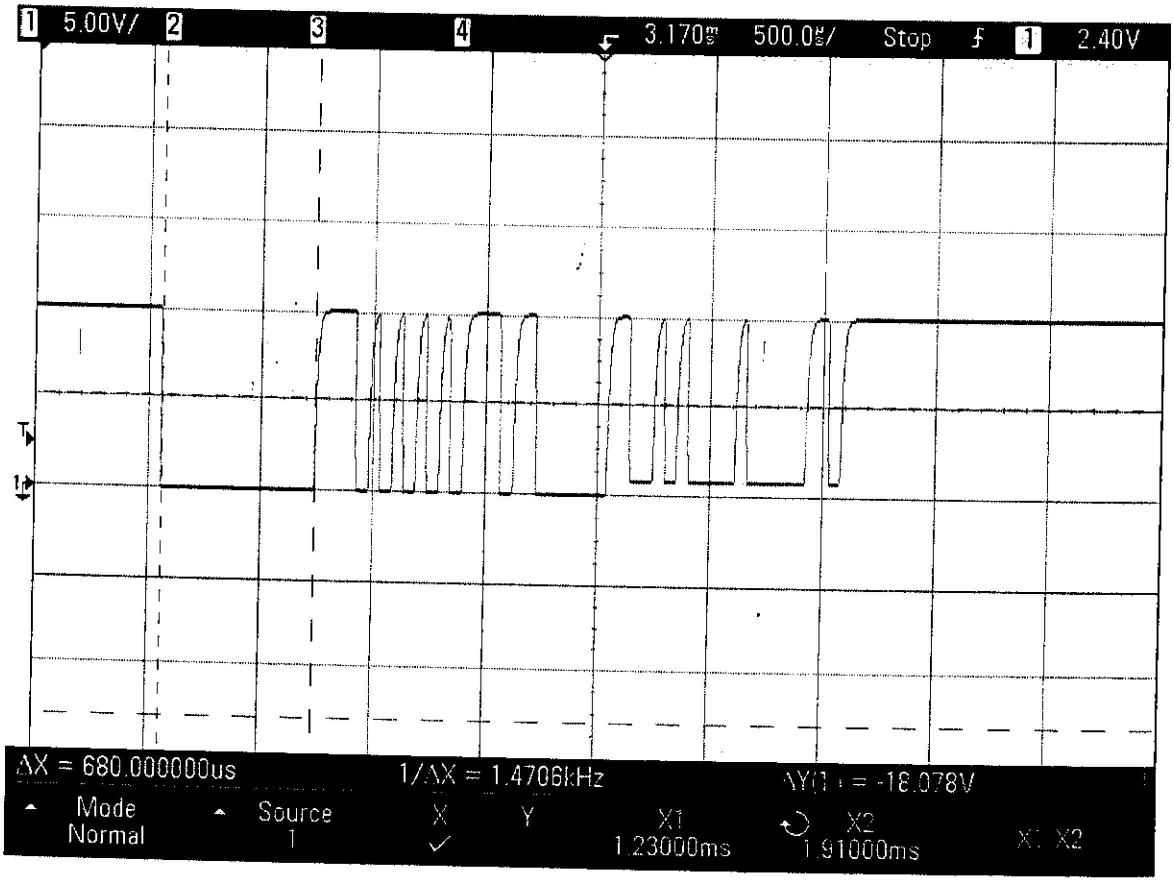
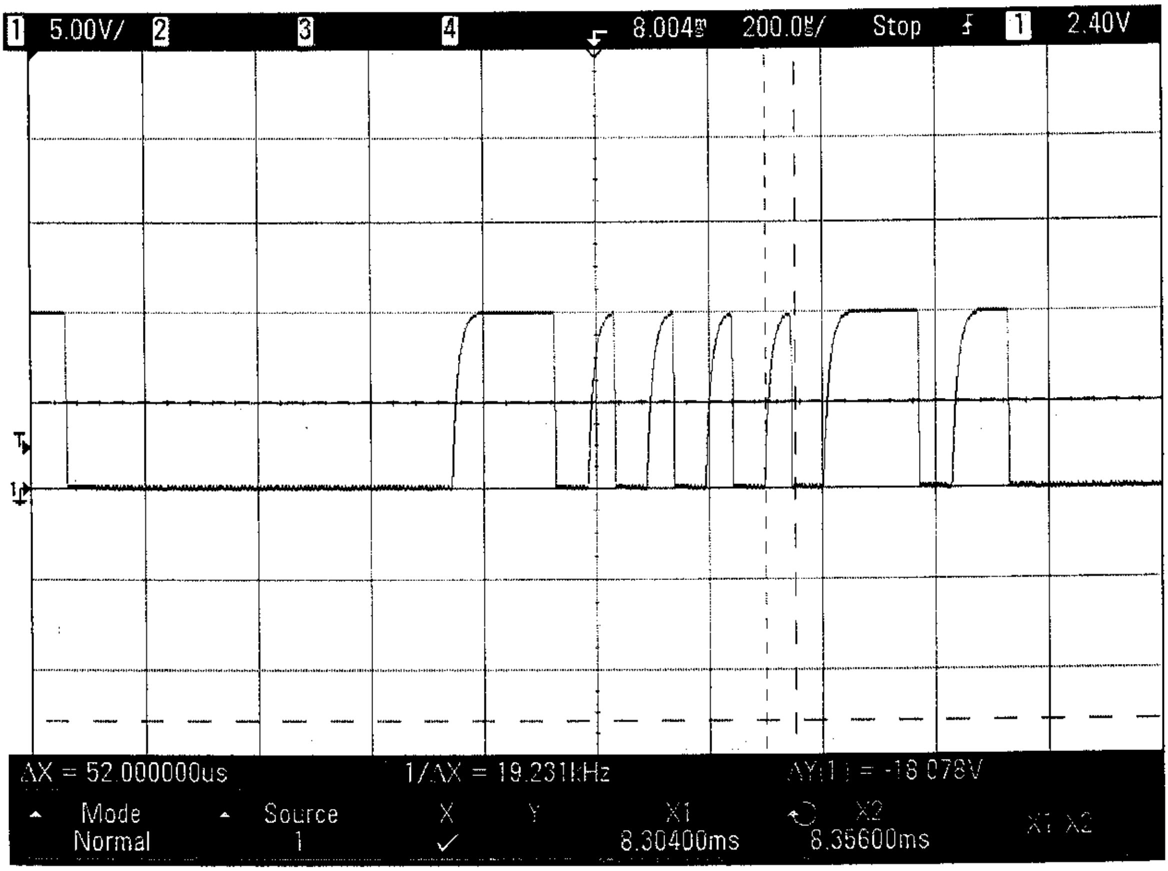


FIG 6.8 BREAK FIELD TIMING

For the baud rate of 19200, the bit time for the 11 bit long break field is 680us.



ΔX = 52.000000us 1/ΔX = 19.231kHz ΔY(1) = -18.078V

Mode Normal

Source 1

X ✓

Y

X1 8.30400ms

X2 8.35600ms

X1 X2

FIG 6.9 BIT TIMING

For the baud rate of 19200, the bit time for each bit is 52us.

CONCLUSION

CHAPTER 7

CONCLUSION

This project finds a way to implement required features in required car without involving any changes in the existing module and at low implementation cost. Any new feature can be added into the existing car by this method. In our project, we have demonstrated the way to develop and establish cassette player control and door control.

This project finds difficulties with cars that doesn't support CAN communication.

REFERENCES

REFERENCES

1. LIN Specification Package, Revision 2.1, November 24, 2006. Introduction to the LIN specification package. LIN Consortium.

2. LIN Protocol Specification, Revision 2.1, November 24, 2006. Specifications for LIN data link layer. LIN Consortium.

WEBSITES:

1. http://www.atmel.com/literature

2. http://www.beyondlogic.org/serial/serial.html

3. http://www.designsparkPCB.net

4. https://www.embedded.com/story/OEG20020528s0057

5. https://www.epanorama.net/links/serialbus.html

6. http://www.kpitcummins.com

7. http://www.lin_subbus.org

8. http://lvr.com/serport.html

9. http://www.semiconductors.philip.com

10. http://www.serialata.org/

11. http://www.siliconengines.net

12. http://www.smbus.org

APPENDICES

APPENDIX 1

MICROCONTROLLER

Features: MB90F036

Models available	- FLASH product
CPU	- F^{2}MC-16LX CPU
Clock	- single/dual
System clock	- on-chip PLL clock, Minimum execution time = 31.25 (ns) (4MHz oscillation * 8)
Sub clock pins	- yes (X_{0A}, X_{1A})
ROM	- 128 kb
RAM	-8 kb
Low voltage/CPU	- yes
Package	- LQFP-120
Emulator	- Dedicated power supply

This microcontroller is a PROPRIETRY PRODUCT of Pricol.

APPENDIX 2

LIN/RS-232 CONVERTER, MODEL 9003:

Introduction:

Model 9003, a compact electronic device that allows a personal computer to connect to an automotive communications data link compatible with the LIN protocol.

Applications:

- **Development**: Facilitates development of an automotive ECU (electronic control unit) that supports a LIN data bus, by allowing a personal computer to act as the LIN bus master during software development.
- **Production**: Allows the LIN data link to serve as a port for testing the ECU, and for downloading programs, parameters, serial number, calibration data, etc.
- **Service**: Allows a personal computer to act as a tester or diagnostic analyzer.

Functions:

- **Level conversion**: Converts signals between LIN voltage levels and RS-232 (CCITT V.24) levels, for connection to a personal computer.
- **LIN 2.0-2.1, LIN 1.3**: Supports the newer LIN 2.0 and 2.1 protocol versions as well as LIN 1.3.
- **Synchronization**: The LIN/RS-232 Converter generates the 13-bit synchronization break, synchronization field, and identifier field at the

start of each LIN message frame. (Note: there is an alternate firmware version that creates a 19-bit break that can be used if that is required for compatibility with certain automotive ECUs. A personal computer working alone cannot readily generate these signals because they depart from standard asynchronous data format and require precise synchronization timing.

- **Speed buffering**: Interfaces to the PC at a fixed baud rate of 38,400 bps. Interfaces to the ECU at a programmable LIN data rate, from 1,000 to 20,000 bps.

- **Automatic keep-alive**: Optionally keeps the LIN bus awake by generating periodic wakeup signals as defined by the LIN specifications. Optionally keeps the LIN bus awake indefinitely using an automatic wake-up mechanism. Also allows extended wakeups to be sent to ensure ECU's are kept awake.

- **Duplexing**: Converts the half-duplex LIN line, to standard full-duplex RS-232 signals. Avoids the need for the PC to deal with echoed characters.

- **Error codes**: Provides a number of error codes to help diagnose problems with LIN protocol issues.

- **Signal indicators**: Provides seven two-color LEDs to show the states of all significant signal lines.

- **Pass-through mode**: Provides an optional mode that allows the LIN RS232 converter to be used strictly as a level converter at 9600 baud for use with Free scale AN 2295 for programming a LIN module. A firmware variant also exists for performing pass-through mode at 19,200 baud.

LIN/RS-232 converter is bus master:

The LIN/RS-232 Converter, together with a connected personal computer, acts as the LIN bus master. It initiates each message frame, which can be completed by a slave device, or the LIN/RS-232 Converter.

Half-duplex single-wire interface:

The LIN bus line is a single-wire interface. Data flow is half-duplex, in one direction only. Either the master ECU is talking, or a slave.

A direct result of the half-duplex nature of the LIN bus line is data echo. For example, whenever a LIN bus slave sends logic 0 to the LIN bus master on the LIN bus line, the LIN bus line goes low. The ECU's LIN bus line receiver detects this condition. This means that every byte that a LIN slave ECU sends to LIN master using its UART's TXD pin, is simultaneously echoed back to the ECU on its UART's RXD pin.

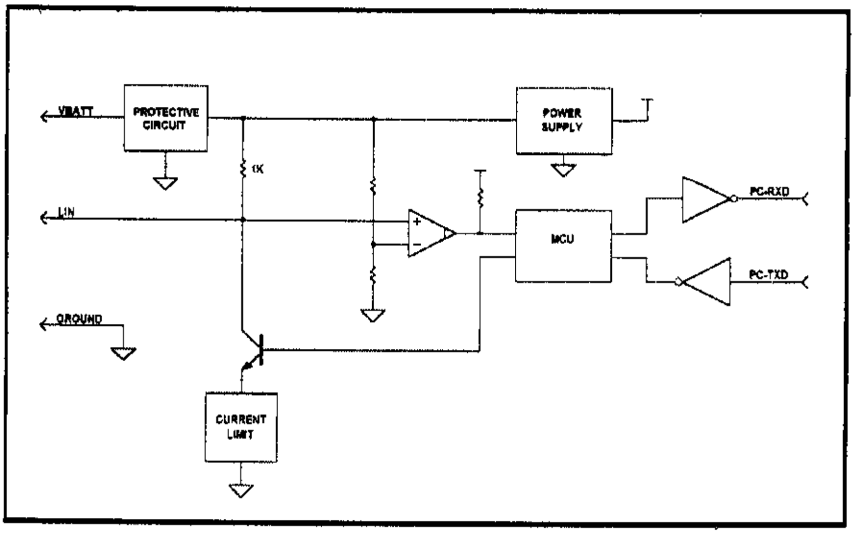
Lin bus line logic within lin/rs-232 converter:

The LIN/RS-232 Converter acts as the LIN bus master in the system.

The microcontroller (MCU) within the Converter does baud-rate conversion. It sends and receives data over the RS-232 TXD and RXD lines at 38,400 bps, and sends and receives data over the LIN bus line at the programmable LIN baud rate of 1,000 to 20,000 bps.

The MCU also converts half-duplex LIN bus line data to full-duplex RS-232. Specifically, when the PC sends a command to transmit a message, to the LIN/RS-232 Converter, the MCU forwards it onto the LIN bus line. Because the LIN bus line is half-duplex, the LIN bus line comparator will see this transmitted data. The MCU filters out echoed bytes to help simplify PC software. The MCU also checks that the echoed bit is correct, and if not, it will generate an error code.

The LIN/RS-232 Converter contains protective circuitry to assure that the maximum voltage on the LIN bus line does not exceed 40 VDC, even in the presence of higher-voltage transients on the VBATT line.



LIN BUS LINE LOGIC WITHIN LIN/RS-232 CONVERTER

LIN bus line electrical specifications:

PARAMETER	MINIMUM	TYPICAL	MAXIMUM	CONDITIONS/COMMENTS
OPERATING VOLTAGE RANGE	+8 VDC	+13.8 VDC	+18 VDC	
LIN/RS-232 CONVERTER LOAD RESISTOR	950 Ω	1,000 Ω	1,050 Ω	1 KΩ ±5%
MAXIMUM VOLTAGE PLACED ON LIN BUS BY LIN/RS-232 CONVERTER		34 VDC		VOLTAGE CLAMP WITHIN LIN/RS-232 CONVERTER POWER SUPPLY
MAXIMUM SINK CURRENT, OUTPUT LOW		58 mA		CURRENT-LIMITED OUTPUT TRANSISTOR IN LIN/RS-232 CONVERTER

LIN/RS-232 CONVERTER ELECTRICAL SPECIFICATIONS

Connectors:

The connector at the left of the LIN/RS-232 Converter is a type DB9M plug (9-pin male D subminiature). A DB9F (female DB9) socket plugs in here. Three signals are supported: VBATT, the LIN bus line, and ground.

PIN NO.	SYMBOL	SIGNAL	DESCRIPTION
1-2	V_B	VBATT	BATTERY POWER
3	**LIN**	LIN BUS	LIN BUS LINE
4		NC	*NO CONNECTION*
5		NC	*NO CONNECTION*
6-9	⏚	GROUND	POWER AND SIGNAL RETURN

LIN CONNECTOR PIN-OUTS

LIN cable to ECU:

A special cable connects from the ECU to the LIN/RS-232 Converter. At the LIN/RS-232 Converter side, use a DB9F connector. Connect VBATT to pin 1, the LIN bus line to pin 3, and ground to pin 6.

RS-232 connector:

The connector at the right of the LIN/RS-232 Converter is a DB9F type, designed for connection to an asynchronous serial communications port on a PC.

Connector polarities and signal names have been chosen for compatibility with the RS-232 (CCITT V.24) serial port of a recent model IBM®-compatible PC. PC-AT and later computer models typically provide a DB9M connector at their serial ports.

Typically the LIN/RS-232 Converter connects to the PC over a DB9 extender cable about 6 feet 2 M) long. A DB9M connector plugs into the LIN/RS-232 Converter at one end of the cable, and a DB9F connector plugs into the PC at the other end. This cable provides straight-through wiring—pin 1 connects to pin 1, pin 2 to pin 2, etc. A cable of this description is provided with each LIN/RS-232 Converter.

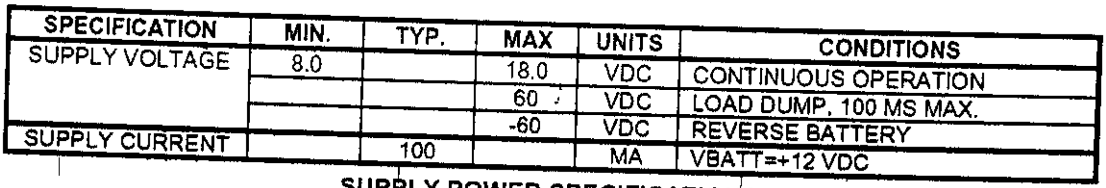
POWER REQUIREMENTS:

Connecting V_{BATT} and ground:

When working with an automotive ECU, the VBATT and GROUND lines are typically connected along with the LIN bus line—to a suitable connector on the ECU. Both the LIN/RS-232 Converter and the ECU are powered from the same source of power. The LIN/RS-232 Converter can also be powered by a DC power supply that connects to building AC power lines.

Input voltage and current:

The LIN/RS-232 Converter contains built-in power supply circuitry that generates needed power from VBATT and GND. The LIN/RS-232 Converter is compatible with 12-volt battery systems (8 to 18 VDC). The unit generates +5 VDC internally for digital logic and the LED indicators, as well as ±10 VDC for the RS-232 interface.



SPECIFICATION	MIN.	TYP.	MAX	UNITS	CONDITIONS
SUPPLY VOLTAGE	8.0		18.0	VDC	CONTINUOUS OPERATION
			60	VDC	LOAD DUMP, 100 MS MAX.
			-60	VDC	REVERSE BATTERY
SUPPLY CURRENT		100		MA	VBATT=+12 VDC

SUPPLY POWER SPECIFICATIONS

Load dump protection:

The LIN/RS-232 Converter contains circuitry for protection against automotive load dump transients up to the maximum levels shown above. These levels are adequate for most current vehicle designs. However, if higher transient levels are anticipated, measures should be taken to protect the LIN/RS-232 Converter. One method is to power the device from an AC line-powered DC power supply, rather than from the vehicle's battery.

Separate V_{BATT} sources:

If the ECU and the LIN/RS-232 Converter are powered from separate sources:

- **Grounds:** The ground of the LIN/RS-232 Converter must be connected to the grounds of the ECU and of both power sources.
- **V_{BATT}:** The V_{BATT} voltage provided to the LIN/RS-232 Converter should be within the ranges specified above, and within ±3 VDC of the V_{BATT} voltage provided to the ECU.

Reverse battery protection:

The LIN/RS-232 Converter is protected against inadvertent reverse battery connection. The unit will not operate properly with reversed power inputs, but will not be damaged, so long as the negative voltage is within the range.

LAMP FUNCTIONS AND SIGNAL FLOW:

Status lamp:

In the center of the top panel of the LIN/RS-232 Converter is a green LED lamp, marked **STAT**.

LED PATTERN	SIGNAL LINE CONDITION
STEADY GREEN	CONVERTER OPERATING NORMALLY
BLINKING	CONVERTER MCU ERROR

STATUS LAMP PATTERNS

At power-up, the MCU within the LIN/RS-232 Converter does a quick self-diagnostic test. A steady STAT lamp indicates that the MCU has passed the test.

Two-color lamps:

Seven two-color indicators on the top panel of the LIN/RS-232 Converter show the status of key RS-232 and LIN signal lines. Each lamp glows either green or red whenever the device is powered up.

LED COLOR	SIGNAL LINE CONDITION
GREEN	HIGH VOLTAGE LEVEL ON SIGNAL AT CONNECTOR
RED	LOW VOLTAGE LEVEL ON SIGNAL AT CONNECTOR

LED COLOR CODES

RS-232 signal lines:

Six LED indicators show the voltage levels on key RS-232 signal lines. The names of the RS-232 signals on the LIN/RS-232 Converter top panel are from the point of view of the connected personal computer.

CONN. PIN	SIGNAL NAME	DATA DIR.	LAMP COLOR	TYP. LEVEL	SIGNAL FUNCTION
1	DCD	CONV→PC	*(NONE)*	+10 V	DATA CARRIER DETECT, HIGH WHEN POWER IS ON
2	RXD	CONV→PC	GREEN	+10 V	START BIT; LOGIC 0 (SPACE)
			RED	-10 V	IDLE LINE; LOGIC 1 (MARK); STOP BIT
3	TXD	CONV←PC	GREEN	+10 V	START BIT; LOGIC 0 (SPACE)
			RED	-10 V	IDLE LINE; LOGIC 1 (MARK); STOP BIT
4	DTR	CONV←PC	GREEN	+10 V	*SIGNAL UNUSED BY LIN/RS-232 CONVERTER*
			RED	-10 V	*SIGNAL UNUSED BY LIN/RS-232 CONVERTER*
5	GND	---	*(NONE)*	0 V	SIGNAL COMMON
6	DSR	CONV→PC	GREEN	+10 V	LIN BUS AWAKE
			RED	-10 V	LIN BUS ASLEEP
7	RTS	CONV←PC	GREEN	+10 V	REQUEST TO SEND, FLOW CONTROL, PC READY TO RECEIVE FROM CONVERTER
			RED	-10 V	PC NOT READY TO RECEIVE FROM CONVERTER
8	CTS	CONV→PC	GREEN	+10 V	DATA SET READY, FLOW CONTROL, CONVERTER READY TO RECEIVE FROM PC
			RED	-10 V	CONVERTER NOT READY TO RECEIVE FROM PC
9	RI		(NONE)	0 V	RING INDICATOR, NOT USED FOR LIN COMMUNICATIONS , CONNECT ONLY TO PC-RI

RS-232 SIGNALS

RXD: Serial data from converter to PC

The RXD line leads to the RECEIVE DATA input of the PC's RS-232 serial port. Voltage levels are ±10 volts nominal. Data speed is fixed at 38,400 bps. Data format is 8N1—one start bit, eight data bits, no parity bit, and one stop bit.

TXD: Serial data from PC to converter

The TXD line originates from the TRANSMIT DATA output of the PC's RS-232 serial port. Voltage levels are typically ±10 volts nominal. Data speed is fixed at 38,400 bps. Data format is 8N1—one start bit, eight data bits, no parity bit, and one stop bit.

DTR: Unused

The DTR (Data Terminal Ready) signal is not used by the LIN/RS-232 Converter. The PC may turn it on or off without any effect on the LIN/RS-232 Converter.

DSR: LIN bus status indicator

The LIN/RS-232 Converter uses the DSR (DATA SET READY) line as a status signal to indicate whether the LIN bus is awake or sleep. When DSR is high, the LIN bus is awake. Communications are active, or the bus is being kept alive by wake-up signals.

RTS: PC flow control

The PC can optionally use the RTS (REQUEST TO SEND) line for flow control. The PC sets RTS high to indicate that it is prepared to receive RS-232 data from the LIN/RS-232 Converter. The PC sets RTS low to cause the LIN/RS-232 Converter to pause in sending data to the PC. The PC serial port should be configured for hardware flow control. RTS should be set ON except when desired to pause incoming data from the LIN/RS-232 Converter.

CTS: Converter flow control

The LIN/RS-232 Converter uses the CTS (REQUEST TO SEND) line for flow control. The LIN/RS-232 Converter sets CTS high to indicate that it is prepared to receive RS-232 data from the PC. The LIN/RS- 232 Converter sets CTS low to cause the PC to pause in sending data to the LIN/RS-232 Converter. The PC serial port should be configured for hardware flow control. The LIN/RS-232 Converter will set CTS ON except when desired to pause incoming data from the PC.

RI: Factory test only

The LIN/RS-232 Converter does not use the RI (RING INDICATOR) line during normal operations. This line is an input to the PC, typically used by an external modem to signal that there is an incoming call. The RI line is reserved for use by Silicon Engines for factory test purposes.

APPENDIX 3

TJA1020-LIN TRANSCEIVER
FEATURES:

General:

□Baud rate up to 20 Kbaud.

□Very low Electromagnetic Emission (EME)

□High Electromagnetic Immunity (EMI)

□Low slope mode for an even further reduction of EME

□Passive behavior in unpowered state.

□Input levels compatible with 3.3 and 5 V devices.

□Integrated termination resistor for Local Interconnect Network (LIN) slave applications.

□Wake-up source recognition (local or remote).

Low power management:

Very low current consumption in sleep mode with local and remote wake-up.

Protections:

□Transmit data (TXD) dominant time-out function.

□Bus terminal and battery pin protected against transients in the automotive environment (ISO7637).

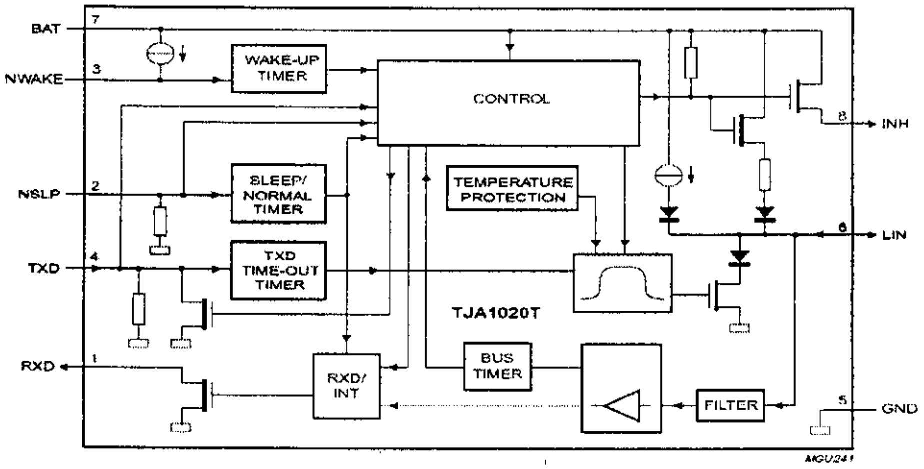
□Bus terminal short-circuit proof to battery and ground.

□Thermally protected.

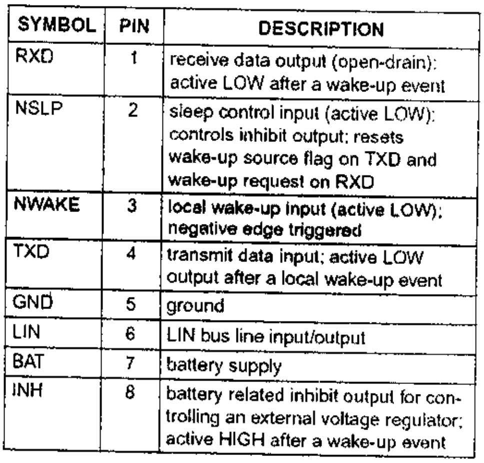
General description:

The TJA1020 is the interface between the LIN master/slave protocol controller and the physical bus in a Local Interconnect Network (LIN). It is primarily intended for in-vehicle sub-networks using baud rates from 2.4 up to 20 Kbaud. The transmit data stream of the protocol controller at the TXD input is converted by the LIN transceiver into a bus signal with controlled slew rate and wave shaping to minimize EME. The LIN bus output pin is pulled HIGH via an internal termination resistor. For a master application an external resistor in series with a diode should be connected between pin INH or pin BAT and pin LIN. The receiver detects the data stream at the LIN bus input pin and transfers it via pin RXD to the microcontroller. In normal transceiver operation the TJA1020 can be switched in the normal slope mode or the low slope mode. In the low slope mode the TJA1020 lengthens the rise and fall slopes of the LIN bus signal, thus further reducing the already very low emission in normal slope mode. In sleep mode the power consumption of the TJA1020 is very low, whereas in failure modes the power consumption is reduced to a minimum.

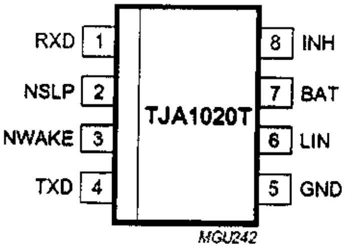
Block diagram:



PINNING



SYMBOL	PIN	DESCRIPTION
RXD	1	receive data output (open-drain): active LOW after a wake-up event
NSLP	2	sleep control input (active LOW): controls inhibit output; resets wake-up source flag on TXD and wake-up request on RXD
NWAKE	3	local wake-up input (active LOW); negative edge triggered
TXD	4	transmit data input; active LOW output after a local wake-up event
GND	5	ground
LIN	6	LIN bus line input/output
BAT	7	battery supply
INH	8	battery related inhibit output for controlling an external voltage regulator; active HIGH after a wake-up event



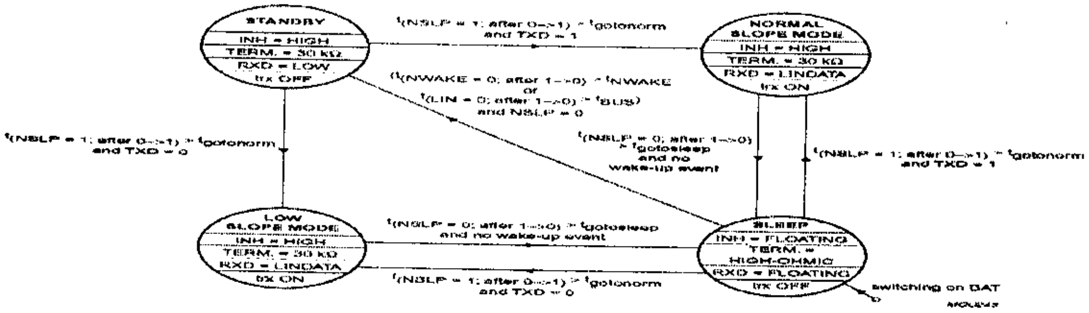
Functional description:

The TJA1020 is the interface between the LIN master/slave protocol controller and the physical bus in a Local Interconnect Network (LIN). The LIN transceiver is optimized for the maximum specified LIN transmission speed of 20 Kbaud providing optimum EMC performance due to wave shaping of the LIN output.

Operating modes:

The TJA1020 provides two modes of normal operation, one intermediate mode and one very low power mode.

State diagram:


STANDBY
INH = HIGH
TERM. = 30 kΩ
RXD = LOW
trx OFF
t(NSLP = 1; after 0->1) > tgotonorm and TXD = 1
NORMAL SLOPE MODE
INH = HIGH
TERM. = 30 kΩ
RXD = LINDATA
trx ON
t(NWAKE = 0; after 1->0) > tNWAKE or t(LIN = 0; after 1->0) > tBUS) and NSLP = 0
t(NSLP = 1; after 0->1) > tgotonorm and TXD = 0
t(NSLP = 0; after 1->0) > tgotosleep and no wake-up event
t(NSLP = 1; after 0->1) > tgotonorm and TXD = 1
LOW SLOPE MODE
INH = HIGH
TERM. = 30 kΩ
RXD = LINDATA
trx ON
t(NSLP = 0; after 1->0) > tgotosleep and no wake-up event
t(NSLP = 1; after 0->1) > tgotonorm and TXD = 0
SLEEP
INH = FLOATING
TERM. = HIGH-OHMIC
RXD = FLOATING
trx OFF
switching on BAT
MGU245

trx: transmitter.
TERM.: slave termination resistor, connected between pins LIN and BAT.

Operating modes:

MODE	NSLP	TXD (OUTPUT)	RXD	INH	TRANSMITTER	REMARKS
Sleep	0	weak pull-down	floating	floating	off	no wake-up request detected
Standby(1)	0	weak pull-down if remote wake-up; strong pull-down if local wake-up; note 2	LOW; note 3	HIGH	off	wake-up request detected; in this mode the microcontroller can read the wake-up source: remote or local wake-up
Normal slope mode	1	weak pull-down	HIGH: recessive state LOW: dominant state	HIGH	normal slope mode	notes 2, 3 and 4
Low slope mode	1	weak pull-down	HIGH: recessive state LOW: dominant state	HIGH	low slope mode	notes 2, 3 and 5

Sleep mode:

This mode is the most power saving mode of the TJA1020 and the default state after power-up (first battery supply). Despite its extreme low current consumption, the TJA1020 can still be waken up remotely via pin LIN, or waken up locally via pin NWAKE, or activated directly via pin NSLP. Filters at the inputs of the receiver (LIN), of pin NWAKE and of pin NSLP are preventing unwanted wake-up events due to automotive transients or EMI. All wake-up events have to be maintained for a certain time period (t_{BUS}, t_{NWAKE} and $_{tgotonorm}$).

The sleep mode is initiated by a falling edge on the pin NSLP driven by the microcontroller. After a filter time continuously driven sleep command (pin NSLP = LOW), pin INH becomes floating.

In sleep mode the internal slave termination between pins LIN and BAT is disabled to minimize the power dissipation in case pin LIN is short-circuited to ground. Only a weak pull-up between pins LIN and BAT is present. The sleep mode can be activated independently from the actual level on pin LIN or NWAKE.

Standby mode:

The standby mode is entered automatically whenever a local or remote wake-up occurs while the TJA1020 is in its sleep mode. These wake-up events activate pin INH and enable the slave termination resistor at the pin LIN. As a result of the HIGH condition on pin INH the voltage regulator and the microcontroller can be activated.

The standby mode is signaled by a LOW level on pin RXD which can be used as an interrupt for the microcontroller.

In the standby mode (pin NSLP is still LOW), the condition of pin TXD (weak pull-down or strong pull-down) indicates the wake-up source: weak pull-down for a remote wake-up request and strong pull-down for a local wake-up request.

Setting pin NSLP HIGH during standby mode results in the following events:

□An immediate reset of the wake-up source flag; thus releasing the possible strong pull-down at pin TXD before the actual mode change (after tgotonorm) is performed.

- A change into normal slope mode if the HIGH level on pin NSLP has been maintained for a certain time period (tgotonorm) and pin TXD is HIGH.
- A change into low slope mode if the HIGH level on pin NSLP has been maintained for a certain time period (tgotonorm) and pin TXD is pulled LOW by the application. In the event of a short-circuit to ground or an open-wire on pin TXD, the LIN output remains recessive (fail safe).

Normal slope mode:

In the normal slope mode the transceiver is able to transmit and receive data via the LIN bus line. The receiver detects the data stream at the LIN bus input pin and transfers it via pin RXD to the microcontroller. HIGH at a recessive level and LOW at a dominant level on the bus. The receiver has a supply voltage related threshold with hysteresis and an integrated filter to suppress bus line noise. The transmit data stream of the protocol controller at the TXD input is converted by the transmitter into a bus signal with controlled slew rate and wave shaping to minimize EME. The LIN bus output pin is pulled HIGH via an internal slave termination resistor. For a master application an external resistor in series with a diode should be connected between pin INH or BAT on one side and pin LIN on the other side.

The normal slope mode is entered by a HIGH level on pin TXD and a HIGH level on pin NSLP maintained for a certain time period (tgotonorm), coming from the sleep or standby mode.

The TJA1020 switches to sleep mode in case of a LOW level on pin NSLP, maintained during a certain time period (tgotosleep).

Low slope mode:

The only difference between the normal slope mode and the low slope mode is the transmitter behaviour. In the low slope mode the transmitter output stage drives the LIN bus line with lengthened rise and fall slopes. This will further reduce the already outstanding EME in the normal slope mode. The low slope mode is perfectly suited for applications where transmission speed is not critical. The mode selection is done by the LIN transceiver after a positive edge on pin NSLP, maintained for a certain time period ($_{tgotonorm}$). If pin TXD is LOW at that time, the low slope mode is entered; otherwise the normal mode is entered. The transition to the low slope mode will be executed during an open pin TXD (fail-safe), a short-circuit from pin TXD to ground (fail-safe) or an intended LOW level of pin TXD programmed by the microcontroller. The transmitter is enabled after a LOW-to-HIGH transition on pin TXD. In the event of a short-circuit to ground on pin TXD, the transmitter will be disabled.

Wake-up:

There are three ways to wake-up a TJA1020 which is in sleep mode:

1. Remote wake-up via a dominant bus state.

2. Local wake-up via a negative edge at pin NWAKE.

3. Mode change (pin NSLP is HIGH) from sleep mode to normal slope/low slope mode.

Remote and local wake-up:

A falling edge at pin NWAKE followed by a LOW level maintained for a certain time period (t_{NWAKE}) results in a local wake-up. The pin NWAKE provides an internal pull-up towards pin BAT. If, during power-up, pin NWAKE is LOW for a certain period of time (tNWAKE) this will also result in a local wake-up. A falling edge at pin LIN followed by a LOW level maintained for a certain time period (tBUS) and a rising edge at pin LIN respectively results in a remote wake-up. After a local or remote wake-up pin INH is activated (it goes HIGH) and the internal slave termination resistor is switched on. The wake-up request is indicated by a LOW active wake-up request signal on pin RXD to interrupt the microcontroller.

Wake-up source recognition:

The TJA1020 can distinguish between a local wake-up request on pin NWAKE and a remote wake-up request via a dominant bus state. The wake-up source flag is set in case the wake-up request was a local one.

The wake-up source can be read on pin TXD in the standby mode. If an external pull-up resistor on pin TXD to the power supply voltage of the microcontroller has been added a HIGH level indicates a remote wake-up request (weak pull-down at pin TXD) and a LOW level indicates a local wake-up request (strong pull-down at pin TXD; much stronger than the external pull-up resistor). The wake-up request flag (signaled on pin RXD) as well as the wake-up source flag (signaled on pin TXD) are reset immediately, if the microcontroller sets pin N_{SLP} HIGH.

TXD dominant time-out function:

A 'TXD Dominant Time-out' timer circuit prevents the bus line from being driven to a permanent dominant state (blocking all network communication) if pin TXD is forced permanently LOW by a hardware and/or software application failure. The timer is triggered by a negative edge on pin

TXD. If the duration of the LOW level on pin TXD exceeds the internal timer value (t_{dom}), the transmitter is disabled, driving the bus line into a recessive state. The timer is reset by a positive edge on pin TXD.

Fail-safe features:

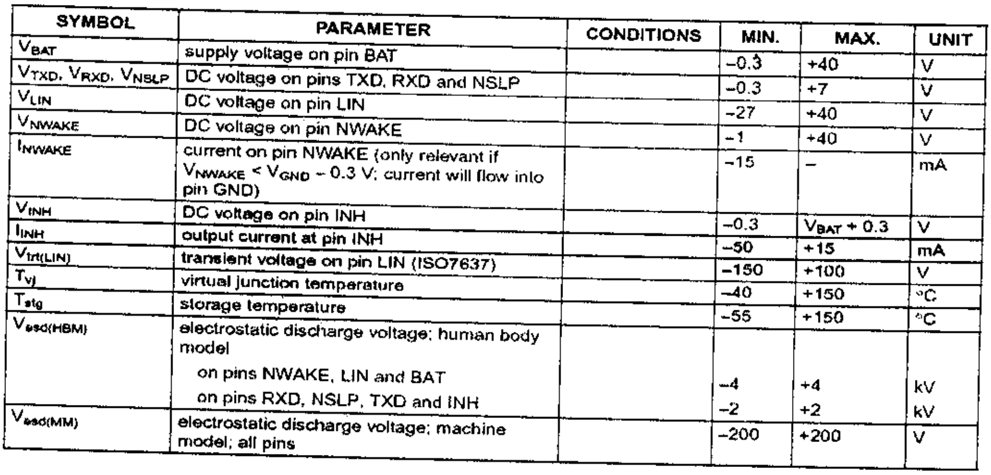
Pin TXD provides a pull-down to GND in order to force a predefined level on input pin TXD in case the pin TXD is unsupplied. Pin NSLP provides a pull-down to GND in order to force the transceiver into sleep mode in case the pin NSLP is unsupplied. Pin RXD is set floating in case of lost power supply on pin BAT.

A loss of power (pins BAT and GND) has no impact to the bus line and the microcontroller. There are no reverse currents from the bus. The LIN transceiver can be disconnected from the power supply without influencing the LIN bus.

The output driver at pin LIN is protected against over temperature conditions. If the junction temperature exceeds the shutdown junction temperature Tj (sd), the thermal protection circuit disables the output driver. The driver is enabled again if the junction temperature has been decreased below Tj(sd) and a recessive level is present at pin TXD.

Limiting values:

In accordance with the Absolute Maximum Rating System (IEC 60134); all voltages are referenced to pin GND.



SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V_{BAT}	supply voltage on pin BAT		−0.3	+40	V
V_{TXD}, V_{RXD}, V_{NSLP}	DC voltage on pins TXD, RXD and NSLP		−0.3	+7	V
V_{LIN}	DC voltage on pin LIN		−27	+40	V
V_{NWAKE}	DC voltage on pin NWAKE		−1	+40	V
I_{NWAKE}	current on pin NWAKE (only relevant if $V_{NWAKE} < V_{GND} - 0.3$ V; current will flow into pin GND)		−15	−	mA
V_{INH}	DC voltage on pin INH		−0.3	$V_{BAT} + 0.3$	V
I_{INH}	output current at pin INH		−50	+15	mA
$V_{trt(LIN)}$	transient voltage on pin LIN (ISO7637)		−150	+100	V
T_{vj}	virtual junction temperature		−40	+150	°C
T_{stg}	storage temperature		−55	+150	°C
$V_{esd(HBM)}$	electrostatic discharge voltage; human body model				
	on pins NWAKE, LIN and BAT		−4	+4	kV
	on pins RXD, NSLP, TXD and INH		−2	+2	kV
$V_{esd(MM)}$	electrostatic discharge voltage; machine model; all pins		−200	+200	V

Characteristics:

VBAT = 5 to 27 V; Tvj □40 to +150 □C; RL(LIN-BAT) = 500 □; all voltages are defined with respect to ground; positive currents flow into the IC; typical values are given at VBAT = 12 V.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UN
Supply						
I_{BAT}	supply current on pin BAT	sleep mode ($V_{LIN} = V_{BAT}$; $V_{NWAKE} = V_{BAT}$; $V_{TXD} = 0$ V; $V_{NSLP} = 0$ V)	1	3	8	μA
		standby mode; bus recessive ($V_{INH} = V_{BAT}$; $V_{LIN} = V_{BAT}$; $V_{NWAKE} = V_{BAT}$; $V_{TXD} = 0$ V; $V_{NSLP} = 0$ V)	100	400	1000	μA
		standby mode; bus dominant ($V_{BAT} = 12$ V; $V_{INH} = 12$ V; $V_{LIN} = 0$ V; $V_{NWAKE} = 12$ V; $V_{TXD} = 0$ V; $V_{NSLP} = 0$ V); note 3	300	900	2000	μA
		low slope mode; bus recessive ($V_{INH} = V_{BAT}$; $V_{LIN} = V_{BAT}$; $V_{NWAKE} = V_{BAT}$; $V_{TXD} = 5$ V; $V_{NSLP} = 5$ V)	100	400	1000	μA
		normal slope mode; bus recessive ($V_{INH} = V_{BAT}$; $V_{LIN} = V_{BAT}$; $V_{NWAKE} = V_{BAT}$; $V_{TXD} = 5$ V; $V_{NSLP} = 5$ V)	100	400	1000	μA
		low slope mode; bus dominant ($V_{BAT} = 12$ V; $V_{INH} = 12$ V; $V_{NWAKE} = 12$ V; $V_{TXD} = 0$ V; $V_{NSLP} = 5$ V); note 3	1	3.5	8	mA
		normal slope mode; bus dominant ($V_{BAT} = 12$ V; $V_{INH} = 12$ V; $V_{NWAKE} = 12$ V; $V_{TXD} = 0$ V; $V_{NSLP} = 5$ V); note 3	1	3.5	8	mA
Pin TXD						
V_{IH}	HIGH-level input voltage		2	–	7	V
V_{IL}	LOW-level input voltage		–0.3	–	+0.8	V
V_{hys}	TXD hysteresis voltage		0.03	–	0.5	V
R_{TXD}	TXD pull-down resistor	$V_{TXD} = 5$ V	125	350	800	kΩ

APPENDIX 4

RELAOD VALUEAND BAUD RATE FOR EACH CLOCK SPEED

	8Mhz		10Mhz		16Mhz		20Mhz		32Mhz	
BAUD RATE	RELOAD VALUE	Dev	RELOAD VALUE	Dev	RELOAD VALUE	Dev	RELOAD VALUE	Dev	RELOAD VALUE	De
4M	-	-	-	-	-	-	4	0	7	0
2M	-	-	4	0	7	0	9	0	15	0
1M	7	0	9	0	15	0	19	0	31	0
500000	15	0	19	0	31	0	39	0	63	0
460800	-	-	-	-	-	-	-	-	68	-0.1
250000	31	0	39	0	63	0	79	0	127	0
230400	-	-	-	-	-	-	-	-	138	0.0
15360	51	-0.16	64	-0.16	103	-0.16	129	-0.16	207	-0.1
25000	63	0	79	0	127	0	159	0	255	0
15200	68	-0.64	86	0.22	138	0.08	173	0.22	277	0.0
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	416	0.0
57600	138	0.08	173	0.22	277	0.08	346	-0.06	554	-0.0
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	832	-0.0
28800	277	0.08	346	<0.01	554	-0.01	693	-0.06	1110	<0.0
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1665	0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	3070	<0.0
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	3332	<0.0
7200	1110	<0.01	1388	<0.01	2221	<0.01	2777	<0.01	4443	<0.0
4800	1666	0.02	2082	-0.02	3332	<0.01	4166	<0.01	6666	<0.0
2400	3332	<0.01	4166	<0.01	6666	<0.01	8332	<0.01	13332	<0.0
1200	6666	<0.01	8334	0.02	133332	<0.01	16666	<0.01	26666	<0.0
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-

APPENDIX 5

PROJECT MODULE: