PROJECT REPORT

ON

Development of a Microcontroller based local network for a Radio Telescope Health Monitoring System

BANGALORE  **UNIVERSITY**

Submitted in Partial fulfillment for award of the degree of

Bachelor of Engineering

In

Electronics & Communication

By

S. KRISHNAMURTHY

Register Number : 01GUEE2009

Under the guidance of

Internal Guides

Dr.K.S.Gurumurthy
Reader and Chairman
Dept. of Electronics and Computer Science Engg.
U.V.C.E., Bangalore –560001

Mr. B.K.Venugopal
Reader
Dept. of Electronics and Computer Science Engg.
U.V.C.E., Bangalore – 560001

External Guide

Prof. N. Udaya Shankar
Astronomy and Astrophysics Group
Raman Research Institute
Bangalore – 560080.

DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE ENGINEERING
UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING
BANGALORE – 560001

2003-2004

BANGALORE UNIVERSITY

Department of Electronics and Computer Science Engg.
University Visvesvaraya College of Engineering
K. R. Circle, Bangalore - 560001

CERTIFICATE

This is to certify that the project entitled ***"Development of a Microcontroller based local network for a Radio Telescope Health Monitoring System"*** is a bonafide work carried out by S. Krishnamurthy (01GUEE2009), student of final semester, having submitted the report in partial fulfillment for the award of Bachelor of Engineering degree in Electronics and Communication Engineering of Bangalore University, during the academic year 2003 – 2004 under our guidance.

Dr. K.S.Gurumurthy
Reader and Chairman
Dept. of Electronics and Computer Science Engineering
U.V.C.E., Bangalore – 560001

Mr. B.K.Venugopal
Reader
Dept. of Electronics and Computer Science Engineering
U.V.C.E., Bangalore - 560001

RAMAN RESEARCH INSTITUTE
C.V. Raman Avenue, Sadashivanagar, Bangalore - 560 080, India

July 10, 2004

Prof. N. Udaya Shankar
Astronomy & Astrophysics Dept.

Certificate

This is to certify that the project work entitled

"Development of a Microcontroller based Local Network for a Radio Telescope Health Monitoring System"

was carried out under my guidance by

S. Krishnamurthy (01GUEE2009)

student of the VIII semester, Bachelor of Engineering (Electronics & Communication Engg.), University Visvesvaraya College of Engineering (UVCE), Bangalore, in partial fulfillment of the requirement for the award of Bachelor's Degree in Electronics & Communication Engineering during the year 2003-04.

N. Udaya Shankar

Phone : Off. +91 (80) 361 0122 Fax : 361 0492 e-mail : root@rri.res.in

Acknowledgement

Very few occasions we get in our life to articulate gratitude towards the people who have shown light for the journey of success. Immense guidance and memorable advice were received towards the partial fulfillment of the project "Development of Microcontroller based local network for a Radio Telescope Health Monitoring System".

My sincere thanks to Dr. K. S. Gurumurthy and Mr. B.K. Venugopal, U.V.C.E., Bangalore for accepting me as their student and for all the guidance and cooperation extended to me during my stay.

My external guide Prof. N. Udaya Shankar, RRI, Bangalore, amidst of his very busy schedule accepted me as his student. I am very grateful to him for providing me an opportunity to do my project work under his guidance. He gave me both moral and technical support through out which helped me in carrying out the project. Dr. A. Krishnan, Visiting Scientist, RRI whose timely suggestion and guidance helped me to learn a lot about the project. Sincere admiration to them.

My special thanks to my friend Mr. Rishin P.V., RRI who helped me in selecting the project and extended his guidance in completing the project. My hearty thanks extend to my close friends at RRI Mr. Seshadri A.L., Ms. Aruna J., Mr. Gopalakrishna M.R. and Mr. Sabir for sharing their valuable knowledge.

My honourable thanks to my parents and my brother who gave me the basic foundation and support which enabled me to do my degree.

I am always proud of my loving wife R. Mamatha Bai for all the moral support given during critical times and also for sparing almost all evenings through out the course time.

S. Krishnamurthy

TABLE OF CONTENTS

Synopsis

List of Figures

List of Flow Charts

List of Tables

List of Circuit Diagrams

SYNOPSIS

One of the main requirements of any Radio Telescope system is a Health Monitoring System (HMS), which serves as a diagnostic tool for detecting and identifying system faults. The monitoring system measures and records several system parameters and informs the control system if any of them exceeds the preset permissible range.

In a Radio Telescope, different sub-systems are spatially apart. The two main tasks involved in Health Monitoring System are 1. Reliable sensor hardware which are connected to the parameters which vary and affects the performance of the Radio Telescope (so called the health monitoring) and 2. Reliable local network hardware that interconnects the sensor outputs and informs the observer's PC located in a centralized location. This PC takes the necessary precautionary measures pre-programmed by the observer.

The present project aims at the "Design and Development of a Microcontroller based Local Network for a Radio Telescope HMS", for a new 12m radio telescope that is being built in the Raman Research Institute.

Among the different network topologies, bus topology is considered to be the most efficient and suitable topology for building the HMS local network. The control PC is considered to be the master controller, which sequentially addresses the sensors present, and commands them to acquire or transmit the parameter value. The health monitoring system acquires data from several sensors that are in the form of analog voltages. Hence, it is desirable to choose a microcontroller with an in-built ADC that can directly interface to the sensors. The sensors will be interconnected to the slave with necessary sensor electronics. The microcontroller based slave systems, which has intelligence to acquire the data from the PC, process the commands and sends back the relevant data to the master.

The communication from the master end, which is a PC, and to the slave end, which is basically a microcontroller, is through the standard RS232 serial port (UART). Due to the requirement of long distance, multi-drop and noise immune communication, RS485 based local network is chosen. As no standard protocol is available to meet the local network for the HMS, a custom built protocol named HMS protocol has been designed and implemented.

Chapter 1
Preamble

Parameters such as ambient temperature, wind speed and wind direction are called environmental parameters. Motor current, power supply voltages, brake current are called control system parameters.

The monitoring system continuously acquires these parameters and checks whether the parameter values are within allowable limits and corrective action is commanded depending on the nature of the fault.

1.2 Statement of the problem

A 12m Radio telescope is being built at RRI. A control system for this radio telescope is under development. To ensure satisfactory performance of the Radio Telescope, continuous monitoring of environmental and system parameters are essential. Since the Radio Telescope consists of several subsystems, which are spatially separated, one requires a local network. "Development of a Microcontroller based Local Network for a Radio Telescope Health Monitoring system" is the problem taken up for the study in this project.

1.3 Objective of the study

The objective of the study is to develop a Microcontroller based Local Network for a Health Monitoring system for the 12m Radio Telescope being built at RRI. The main objective is to achieve safe and satisfactory performance of the telescope even under adverse operating conditions by monitoring the system parameters and taking appropriate corrective measures.

1.4 Scope of the Study

a) Understanding a radio telescope control system and various sub-systems involved in it.
b) Identifying the different environmental and safety interlock parameters.
c) Understanding the microcontroller architecture and programming concepts.
d) Understanding the hardware interface between PC and a microcontroller.
e) Framing the communication protocol between the PC and the microcontroller

f) Understanding different network topologies.

g) Understanding RS232, 422 and RS485 serial communication bus.

1.5 Review of literature

This project involves different aspects like hardware, software, firmware (the hexcode generated and then loaded to the microcontroller), etc. To realize a project of this type, literatures related to the Microcontrollers, Electronic circuits, Printed Circuit Boards (PCBs), Serial Ports, RS-232, RS-422, Linux serial programming and Assembly language programming for the microcontrollers were studied. Some of them are listed below:

[1]. Kenneth J. Ayala, The 8051 Microcontroller Architecture, Programming and Applications, Penram International Publishing (India) Ltd, 1996.

One of the classic book we referred for understanding about the microcontrollers is This helps in the electronic design and assembly programming of the small special purpose digital controllers and systems.

[2]. ADuC832 Microcontroller manual from Analog Devices Inc.

This is the sole manual, which gave an in-depth description of the ADuC832 architecture, instruction set, and timing diagrams. This is very much essential for writing the software as well as for designing the hardware related to the chip.

[3]. Fredrick M.Cady, Microcontrollers and Microcomputers, Oxford University Press, 1997

This book gives the knowledge on principles of software and hardware engineering, which involves introduction to computer architecture, assembly language programming, serial input & output system, ADC and DAC. It also talks about RS232, RS422 and RS485 bus.

[4]. Jan Axelson, The Microcontroller Idea Book Circuits, Programs, & Applications Penram International Publishing (India) Ltd, 1977

This book is a hands-on guide to designing, building and testing microcontroller based devices. Even though this book talks more about 8052-basic microcontroller, the general hardware concepts to design any microcontroller based circuits are very well explained and it is a kind of guide to any beginner and also to the one who has hands-on experience.

[5]. Jan Axelson, Serial Port Complete, Programming and Circuits for RS-232 and RS-485 links and networks, Penram International Publishing (India) Ltd.

This is a guide to designing, programming, installing and troubleshooting computer links, including networks of multiple computers. Most of the links described used one of two serial interfaces popularly known as RS-232 and RS-485.

Websites referred

http://www.bb-elec.com/

http://www.maxim-ic.com/

http://www.lammertbies.nl/comm/index.html

1.6 Methodology

A 12m radio telescope is being built at RRI. Control system for the same is under development. For the safe and satisfactory performance of the telescope it is mandatory to have a health monitoring system and a local network for communication, which monitors and informs the control system about various critical parameters.

In the 12m telescope, the control system related tasks are carried out by a PC (called the control PC). The control PC runs many control system processes with priorities assigned and cannot be tied up for continuous health monitoring processes. Hence it is desirable to have hardware with built-in intelligence, which measures several control parameters and environmental parameters and transfer the data to the PC over a local network. Microcontrollers have become the de facto standard in industry for these

kinds of applications and they are very robust and economical. Hence, it has been decided to build a microcontroller based intelligent hardware which can take care of the monitoring process independent of the PC. The microcontroller from Analog devices ADuC832 has been chosen for this purpose, since it combines the power of a 8052 microcontroller core with data acquisition capabilities. The ADuC832 is a complete smart transducer front end, integrating a high performance self-calibrating multichannel 12-bit ADC, dual 12-bit DACs, and programmable 8-bit MCU on a single chip.

The local network system is developed based on RS422/485 multidrop bus topology which links the different sensor points (hitherto called slave systems) to the control PC (hitherto called the Master controller). RS422/485 network has been chosen for this purpose since it has been proven in industry for it's noise immunity and long distance communication capability. The locations of the sensors and the approximate distances are mentioned in chapter 3. The data acquisition need not be of high speed since the variations in the parameters are slow speed.

1.7 Limitations

The 12m radio telescope is still under construction. The mount or supporting structure for the dish is still under fabrication. Hence the various sensors and interlocking systems are not procured. The testing of the system has been carried out using simulated sensor inputs with assumed standard input ranges.

System parameters should also include several receiver system parameters such as L.O. frequency, L.O. lock indicator, RF output level, etc., However the present project discusses only the needs of control system parameters. The communication protocol designed allows a maximum of 2^5 slave systems to be connected. This can easily cater to the maximum requirement of any radio telescope monitoring system. However the testing of the protocol has been carried out using two slave systems only.

Even though the RS422/485 network can cater to a maximum cable length of 3000feet, the testing of the network has been carried out with a cable length of 12m.

1.8 Report Organisation

The whole project report is organised into the following six chapters:

Chapter 1 – Preamble describes general introduction of the project, the literatures reviewed during the project, the methodology used.

Chapter 2 – System Design Analysis describes the control system of a radio telescope, why health monitoring system is required? The requirement of the local network and the communication protocols for health monitoring system are discussed.

Chapter 3 – Hardware requirement specifications describes the overall hardware built for the project and methodology used to approach the local network.

Chapter 4 – Hardware design and implementation describes the hardware, software and firmware for implementing the local network and the custom built protocol for radio telescope monitoring.

Chapter 5 – System configuration and testing describes about how the overall hardware is interconnected and the software part is implemented to build the protocol.

Chapter 6 – Conclusion and scope for future work describes the summary of work done, challenges encountered, limitations and future plans.

Chapter 2

System Design Analysis

2. System Design Analysis

2.1 Control system for a radio telescope

One of the major requirements of any radio telescope system is to be able to point the telescope accurately to the required source of interest and continuously track it. A radio telescope control system, as the name suggests controls the motion of the antenna so as to achieve tracking of a celestial source of interest.

2.1.1 A Radio telescope control system

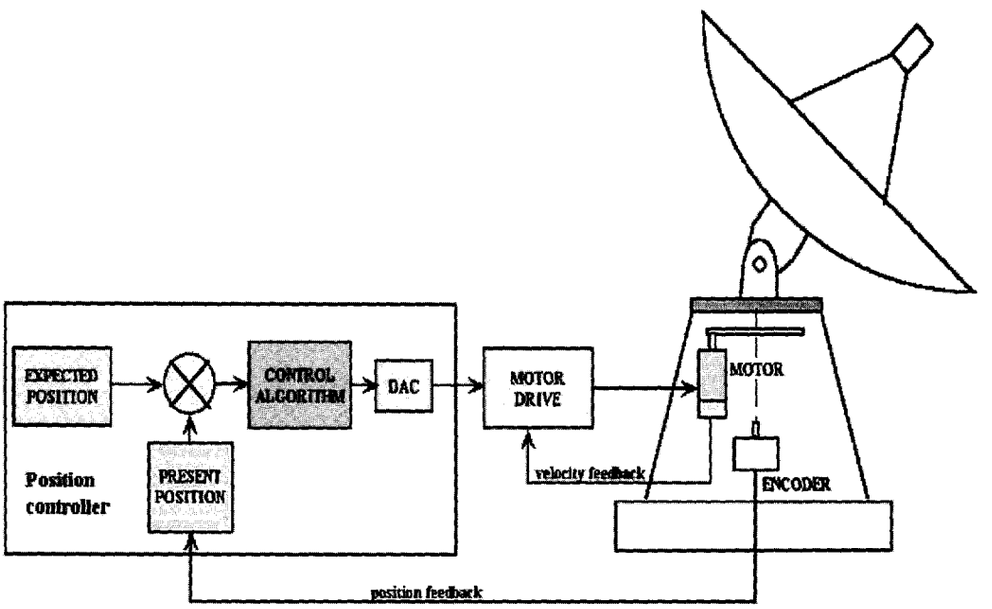


Fig.1 A Radio telescope control system

A Radio Telescope control system consists of

1. Position encoders which gives information regarding the present position of the telescope,

2. A PC which calculates the expected position of the telescope at each instant, a controller which compares it with the feedback received from the position

encoders and issues a velocity command to the motor drives depending on the error in position.

3. Motor drives which rotate the motors at the commanded speed issued by the position controller independent of the load and

4. Motors, which steers the telescope to the expected position in the sky.

The above components are integrated into three basic control system loops- 1.The position loop, 2.The velocity loop and 3. The current loop.

At any instant, the aim is to position the telescope continuously so that it tracks some particular source in the sky with the required accuracy. For continuous position correction, the speed required to reach the source position at the correct instant of time need to be calculated and the telescope has to be moved at that speed. Now, if we want to move the telescope continuously at the required speed, the motor has to develop the required torque at any instant. Since torque produced by the motor is proportional to the current flowing through the motor, we need to continuously correct the current at any instant.

2.1.2 The 12m Radio telescope control system

The 12m radio telescope control system consists of two redundant control paths - one using a hardware controller called Programmable Multi Axis Controller (PMAC) and the other using a Linux based PC.

The purpose of Linux based control system being built in-house is to provide redundancy in the control system and to provide an inexpensive substitute for commercially available hardware controllers. Being built in-house this also provides flexibility in implementation of different control system methodologies and architectures.

The present project aims to design and develop a local network for monitoring several control system parameters and also many environmental parameters.

2.2 The Health monitoring system

One of the main requirements of any Radio Telescope system is a health monitoring system, which serves as a diagnostic tool for detecting and identifying system faults. The monitoring system measures and records several system parameters and informs the control system if any of it exceeds the preset permissible range. The control system utilizes the monitored data and takes appropriate action so as to ensure a safe and healthy functioning of the telescope. Fig.2: shows the overall block diagram of the control system.

2.2.1 Monitoring System parameters

The satisfactory performance of a radio telescope necessitates several parameters to be monitored continuously. These parameters can be broadly classified into two main categories. 1. Environmental parameters 2. Control system parameters.

Environmental parameters:

Drastic variation in environmental conditions over and above the predefined design limits can seriously degrade the performance of a telescope and can even challenge its safety.

The mechanical structure design of any radio telescope assumes a maximum limit to the wind drag forces, which it can sustain. Generally the radio telescopes are designed for observations up to a maximum wind speed called the operational wind speed, to be capable of moving and getting parked up to a maximum wind speed called the maneuverable wind speed and lastly to sustain the loads due to a maximum wind speed called the survival wind speed. Hence monitoring the parameters such as wind speed and direction are of prime importance in any radio telescope.

Another environmental factor that requires monitoring in any radio telescope is the ambient temperature.

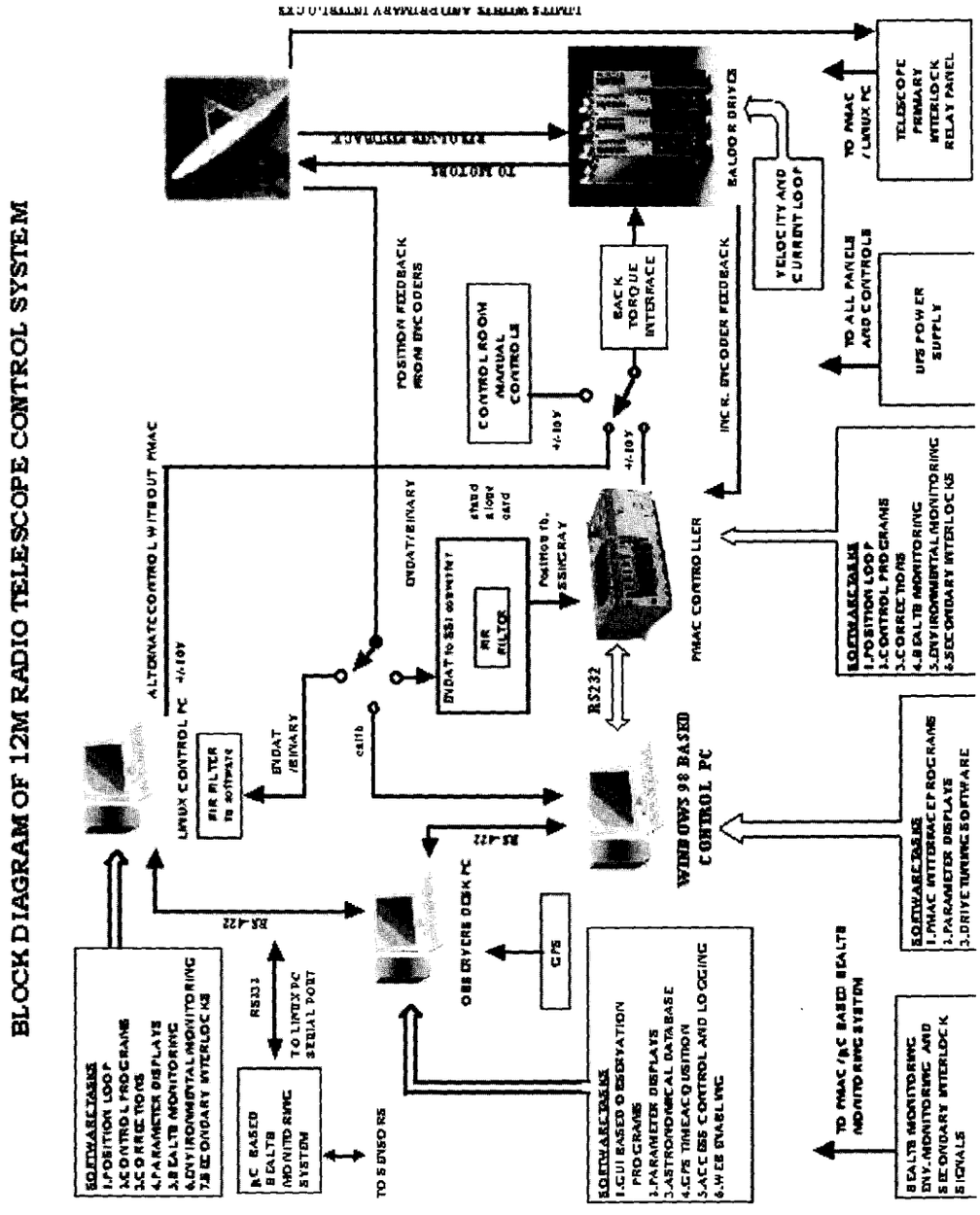


Fig.2: Block diagram of the 12m Radio Telescope control system

Control system parameters:

The control system consists of several electrical components like motors, encoders drives, whose operation needs to be continuously monitored by measuring motor currents, temperature of the regenerative resistors connected to the drives, display of the encoder readings at the remote locations. Apart from this, the control system will also incorporate several safety interlocks like limit switches, cable wrap monitors, UPS monitors, single phasing detectors that needs continuous monitoring.

2.3 Need for Local Network

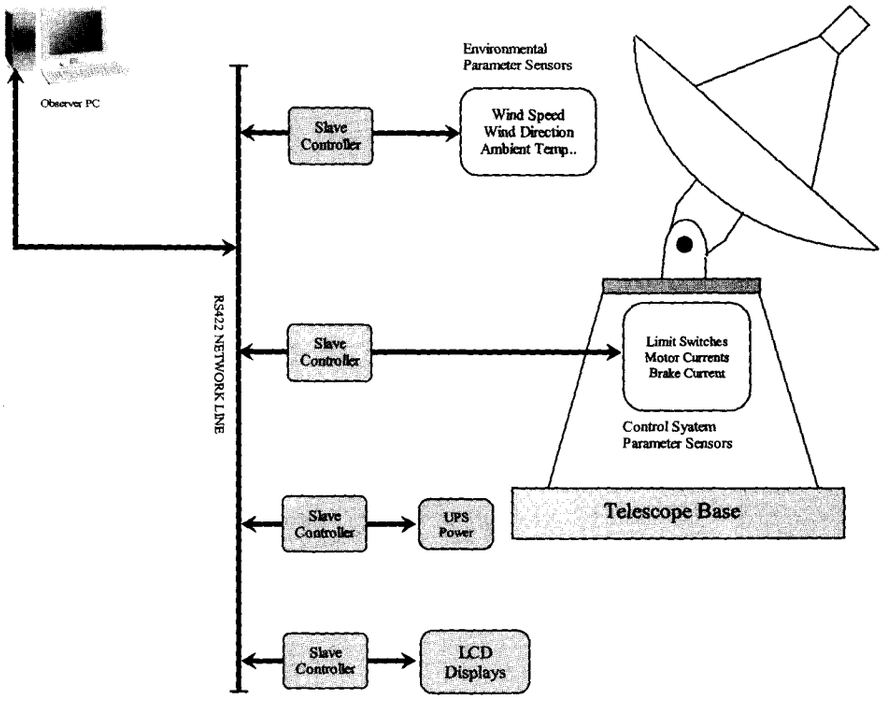


Fig.3: An overview of local network for health monitoring system

In the health monitoring system under consideration, the different parameters to be monitored are measured using sensors located as shown in the figure 3. Since the radio telescope system is a distributed system, these monitoring points are distributed over a wide area around the telescope. The approximate distances from the PC to the monitoring points are given the chapter-3, table-1. In order to process the data

acquired from these monitoring points, it is necessary to transfer it to a centralized unit. Hence the need for a reliable local network arises. Fig.3 shows an overview of local network for health monitoring system

2.4 Network Topologies

Network topologies describe the ways in which computers and peripherals (nodes) are connected together in a network. There are basically 4 ways in which a network can be organized. These are bus, star and ring topologies.

Bus Network

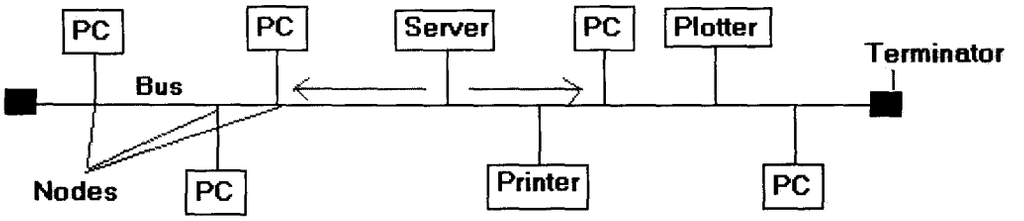


Fig.4: Bus network topology

In the Bus Network, messages are sent in both directions from a single point and are read by the node (computer or peripheral on the network) identified by the code with the message. Most Local Area Networks (LANs) are Bus Networks because the network will continue to function even if one computer/peripheral is down. The purpose of the terminators at either end of the network is to stop the signal being reflected back.

Star Network

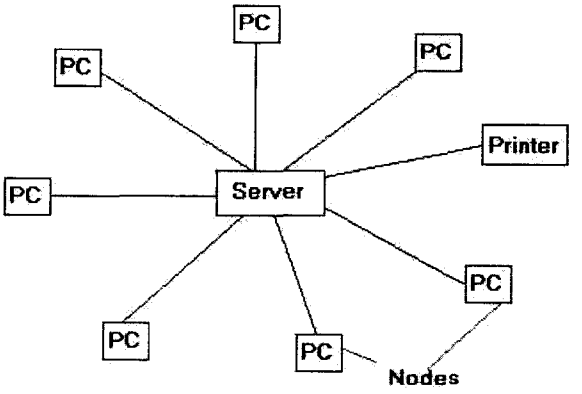


Fig.5: Star Network

In a Star Network, all the nodes (PCs, printers and other shared peripherals) are connected to a central server. The advantage of Star Networks is that one node that is not working properly will not affect the rest of the network. It is very easy to add and remove nodes. It can be more expensive because it uses more cabling than other topologies. If the central server goes down, then no one can use the network.

Ring Network

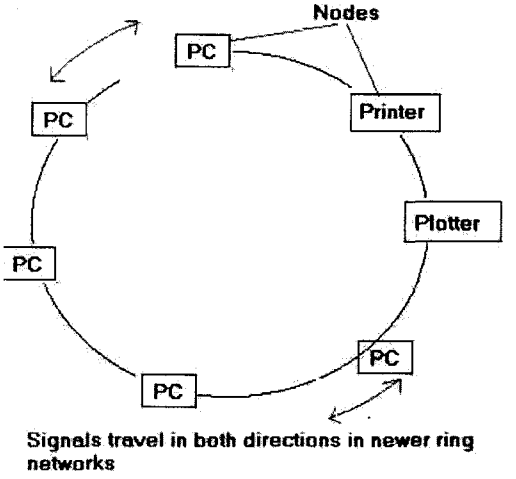


Fig.6: Ring network

All the nodes in a Ring Network are connected in a closed circle of cable. Messages that are transmitted travel around the ring until they reach the computer that they are

addressed to, the signal being refreshed by each node. There may or may not be a fileserver. The advantage of ring networks is that they can be larger than bus or star because each node regenerates the signal. A disadvantage is that the network goes down if one node is inoperable. Data clashes can also occur if two machines send messages at the same time. Tokens or electronic signals that travel around the ring were invented to solve this problem. In a Token Ring Network, a computer can only send a message when the token is with it at the time.

2.5 Communication Channels

A communication channel is a pathway over which information can be conveyed. It can be a physical wire that directly connects communicating devices or it can be other radiated energy sources that has no obvious physical presence like radio waves, laser light, Infrared etc. Information sent through a communication channel has a source from which the information originates, and a destination to which the information is delivered. Although information originates from a single source, there may be more than one destination, depending upon how many receive stations are linked to the channel and how much energy the transmitted signal possesses.

In a digital communication channel, the information is represented by individual data bits, which may be encapsulated into multibit message units. A byte, which consists of eight bits, is an example of a message unit that may be conveyed through a digital communications channel. A collection of bytes may itself be grouped into a frame or other higher-level message unit. Such multiple levels of encapsulation facilitate the handling of messages in a complex data communications network. Any communications channel has a direction associated with it.

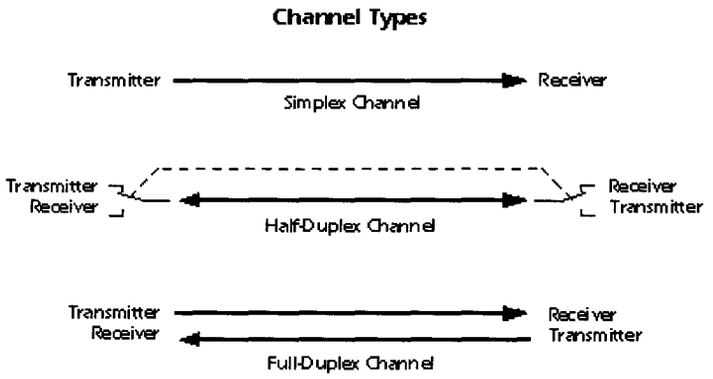


Fig.7: Communication channel types

The message source is the transmitter, and the destination is the receiver. A channel whose direction of transmission is unchanging is referred to as a simplex channel. For example, a radio station is a simplex channel because it always transmits the signal to its listeners and never allows them to transmit back.

A half-duplex channel is a single physical channel in which the direction may be reversed. Messages may flow in two directions, but never at the same time, in a half-duplex system. In a telephone call, one party speaks while the other listens. After a pause, the other party speaks and the first party listens. Speaking simultaneously results in garbled sound that cannot be understood.

A full-duplex channel allows simultaneous message exchange in both directions. It really consists of two simplex channels, a forward channel and a reverse channel, linking the same points. The transmission rate of the reverse channel may be slower if it is used only for flow control of the forward channel.

2.6 Protocols for data communication

Data Communication *protocols* define the manner in which peer processes communicate between computer and the hardware devices. The protocols give the rules for such things as the passing of messages, the exact formats of the messages and how to handle error conditions.

If two computers are communicating and they both follow the protocol(s) properly, the exchange is successful, regardless of what types of machines they are and what operating systems are running on the machines. As long as the machines have software that can manage the protocol, communication is possible. Essentially, therefore, a computer protocol is a set of rules that coordinates the exchange of information.

Data Communication, like any data transfer, requires coordination between the sender and receiver. For example, when to start the transmission and when to end it, when one particular bit or byte ends and another begins, when the receiver's capacity has been exceeded, and so on. Basically there are two communication methods (1) Serial communication (2) Parallel communication.

2.6.1 Serial communication

Serial is a very common protocol for device communication that is standard on almost every PC. It is not to be confused with Universal Serial Bus (USB). Most computers include two RS-232 based serial ports. Serial is also a common communication protocol that is used by many devices for instrumentation; numerous GPIB-compatible devices also come with an RS-232 port. Furthermore, serial communication can be used for data acquisition in conjunction with a remote sampling device.

The serial port sends and receives bytes of information one bit at a time. Although this is slower than parallel communication, which allows the transmission of an entire byte at once, it is simpler and can be used over longer distances. For example, the IEEE 488 specifications for parallel communication state that the cabling between equipment can be no more than 20 meters total, with no more than 2 meters between any two devices; serial, however, can extend as much as 1200 meters.

Typically, serial is used to transmit ASCII data. Communication is completed using 3 transmission lines: (1) Ground, (2) Transmit, and (3) Receive. Since serial is

asynchronous, the port is able to transmit data on one line while receiving data on another. Other lines are available for handshaking, but are not required. The important serial characteristics are baud rate, data bits, stop bits, and parity. For two ports to communicate, these parameters must match:

Baud rate: a speed measurement for communication. It indicates the number of bit transfers per second. For example, 300 baud is 300 bits per second. When we refer to a clock cycle we mean the baud rate. For example, if the protocol calls for a 4800 baud rate, then the clock is running at 4800Hz. This means that the serial port is sampling the data line at 4800Hz. Common baud rates for telephone lines are 14400, 28800, and 33600. Baud rates greater than these are possible, but these rates reduce the distance by which devices can be separated. These high baud rates are used for device communication where the devices are located together, as is typically the case with GPIB devices.

Data bits: a measurement of the actual data bits in a transmission. When the computer sends a packet of information, the amount of actual data may not be full 8 bits. Standard values for the data packets are 5, 7, and 8 bits. Which setting you choose depends on what information you are transferring. For example, standard ASCII has values from 0 to 127 (7 bits). Extended ASCII uses 0 to 255 (8 bits). If the data being transferred is simple text (standard ASCII), then sending 7 bits of data per packet is sufficient for communication. A packet refers to a single byte transfer, including start/stop bits, data bits, and parity. Since the number of actual bits depend on the protocol selected, the term packet is used to cover all instances.

Stop bits: used to signal the end of communication for a single packet. Typical values are 1, 1.5, and 2 bits. Since the data is clocked across the lines and each device has its own clock, it is possible for the two devices to become slightly out of sync. Therefore, the stop bits not only indicate the end of transmission but also give the computers some room for error in the clock speeds. The more bits that are used for stop bits, the greater the lenience in synchronizing the different clocks, but the slower the data transmission rate.

Parity: a simple form of error checking that is used in serial communication. There are four types of parity: even, odd, marked, and spaced. The option of using no parity is

also available. For even and odd parity, the serial port sets the parity bit (the last bit after the data bits) to a value to ensure that the transmission has an even or odd number of logic high bits. For example, if the data is 011, then for even parity, the parity bit is 0 to keep the number of logic-high bits even. If the parity is odd, then the parity bit is 1, resulting in 3 logic-high bits. Marked and spaced parity does not actually check the data bits, but simply sets the parity bit high for marked parity or low for spaced parity. This allows the receiving device to know the state of a bit to enable the device to determine if noise is corrupting the data or if the transmitting and receiving device clocks are out of sync.

The advantages of using serial communication when compared with parallel communication for the local network of HMS are as follows.

The serial port cable can be longer than a parallel port cable. The serial port transmits '1' as voltage from –5 to -12V and '0' as voltage from +5 to +12 V, while parallel port transmits '1' as voltage of 5 volts and '0' as voltage of 0 volts. At the same time the receiver of the serial port receives '1' as voltage from -3 to -25 V and '0' as voltage from +3 to +25 V. Thus serial port can have maximal swing up to 50 volts, while parallel port has maximal swing of 5 volts. Thus the losses in the cable when transmitting data using serial port are less substantial than losses when transmitting data using parallel port.

The number of wires needed when transmitting data serially is less than when the transmission is parallel. If the external device has to be installed at a great distance from the computer, the cable with three wires is much cheaper than the cable with 19 or 25 wires if the transmission is parallel. Still there are interface creation expenses for every receiver/transmitter.

Another proof of serial port universality is microcontrollers. Many of them have inbuilt SCI (Serial Communications Interfaces), used for communication with other devices. In this case serial interface reduces the number of outputs on the chip. Usually only 2 outputs are used: Transmit Data (TXD) and Receive Data (RXD).

Serial Communication protocols

Most digital messages are vastly longer than just a few bits. Because it is neither practical nor economic to transfer all bits of a long message simultaneously, the message is broken into smaller parts and transmitted sequentially. Bit-serial transmission conveys a message one bit at a time through a channel. Each bit represents a part of the message. The individual bits are then reassembled at the destination to compose the message. In general, one channel will pass only one bit at a time. Thus, bit-serial transmission is necessary in data communications if only a single channel is available. Bit-serial transmission is normally just called serial transmission and is the chosen communications method in many computer peripherals.

Byte-serial transmission conveys eight bits at a time through eight parallel channels. Although the raw transfer rate is eight times faster than in bit-serial transmission, eight channels are needed, and the cost may be as much as eight times higher to transmit the message. When distances are short, it may nonetheless be both feasible and economic to use parallel channels in return for high data rates.

2.6.2 Asynchronous Communications

For the computer to understand the serial data coming into it, it needs some way to determine where one character ends and the next begins. In this we deal exclusively with *asynchronous* serial data. In asynchronous mode the serial data line stays in the mark (1) state until a character is transmitted. A *start* bit preceeds each character and is followed immediately by each bit in the character, an optional parity bit, and one or more *stop* bits. The start bit is always a space (0) and tells the computer that new serial data is available. Data can be sent or received at any time, thus the name asynchronous.

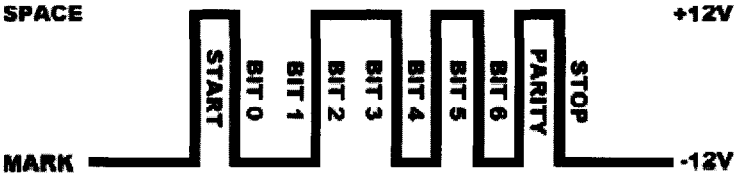


Fig.8: Asynchronous Data Transmission

2.6.3 Synchronous Communications

Unlike asynchronous data, synchronous data appears as a constant stream of bits. To read the data on the line, the computer must provide or receive a common bit clock so that both the sender and receiver are synchronized. Each protocol defines certain bit sequences to represent the beginning and end of a data packet. Each also defines a bit sequence that is used when there is no data. These bit sequences allow the computer see the beginning of a data packet.

Because synchronous protocols do not use per–character synchronization bits they typically provide at least a 25% improvement in performance over asynchronous communications and are suitable for remote networking and configurations with more than two serial interfaces.

Chapter 3

Hardware Requirement Specifications

3. Hardware Requirement Specifications

Hardware requirement specifications for health monitoring involves development of a local network with associated bus topology, bus converters, slave systems, which act as intelligent nodes and interface electronics to sensors. Table 1 gives the list of sensors required for HMS.

Sl. No.	Parameter to be monitored	Approximate Distance*	No of units
1	Wind speed	50m	2
2	Wind direction	50m	1
3	Limit switches (Elevation)	45m	4
4	Limit switches (Azimuth)	15m	4
5	Cable wrap monitor	15m	1
6	Temperature of regenerative resistors	5m	4
7	Ambient temperature	5m	1
8	UPS	5m	4
9	Single phase detector	5m	1
10	Motor currents 3-phase	3m	3×4
11	Brake current	3m	1×4

Table 1: Sensors required for HMS

*The proposed telescope will be located at a distance of 10m from the control room in which the control PC will be located. The approximate distance of the monitoring points referred in Table 1 is from the control PC (refer Figure 3).

Sl no's 1-3 are located on/near the telescope dish.

Sl no's 4, 5 are located inside the telescope conical structure.

Sl no's 6-11 are located near/inside the control room.

3.1 Specifications for local network

The local network requires communication interfaces to different sensors. The specifications for building the local network interface are to meet characteristics like

Long distance communication (a maximum length of 60 meters measured from the control PC to the last monitoring point with reference to the Table 1).

Multidrop facility (a minimum of 9 slaves are required to monitor the different parameters as mentioned in the Table 1. Even though the number of parameters shown in the table is more than 9, the slave system can accept multiple sensors connected to its port. [Refer "Microcontroller features" in Chapter 4 of this report]).

Noise immunity

Serial Communication (the serial port of the control PC has been chosen to build the local network for HMS. Different communication protocols like ETHERNET, CAN bus are available which require a hardware controller and a device driver. This will increase the hardware complexity).

To establish reliable data communication with far away sub systems, signals need to be transmitted and received without loss of information. Hence the signals need to be transmitted over balanced differential line using twisted pair cable, which is immune to cross talk and noise. The required hardware with sufficient intelligence has to be designed to interface different sensors. The hardware units will read the parameters and transfer the data to a centralized unit.

3.2 Local Network Layers

The network layers used in this project can be broadly classified into four layers. They are:

1. Physical layer.
2. Data link layer.
3. Transport layer.
4. Application layer.

1. Physical layer – This layer specifies electrical and mechanical details of the communication like signal type, the levels, the wire types and the connectors used. Since our requirement is for long distance noise immune data communication, RS485 standard has been selected. Further discussion discusses about RS485 data transmission.

EIA Standard RS-485 Data Transmission

RS-422/RS-485 involves sending an inverted or out of phase copy of the signal simultaneously on a second wire. This is called a balanced transmission. Any outside electrical noise adds coherently to both signal copies. The receiver electrically subtracts the two signals to reproduce the original signal. The advantage in the subtraction is that only the intended signal gets reproduced since they are out-of-phase. The in-phase noise on the two wires is also subtracted from each other to produce a net zero noise component in the reproduced signal. This noise immunity allows the RS-422/RS-485 interface to transmit digital signals at faster rates over longer distances than the RS-232/ SDI-12 interface. The RS-232/SDI-12 interface does not use balanced transmission and is therefore susceptible to noise interference, which considerably limits the transmission distance and speed.

The RS-485 Standard permits a balanced transmission line to be shared in a party line or multidrop mode. As many as 32 driver/receiver pairs can share a multidrop network. This limitation arises from the maximum output current the drivers can source taking into consideration the input impedances of the slave inputs and the termination resistors used. The number of multidrops can be increased to 128 by making use of higher-end chips whose input impedances are increased by a factor of 4. The range of the common mode voltage that the driver and receiver can tolerate is +12 to -7 volts. Since the driver can be disconnected or tristated from the line, it must withstand this common mode voltage range while in tristate condition. Some RS-422 drivers, even with tristate capability, will not withstand the full voltage range of -7 to +12 volts.

Fig.9 shows a typical two-wire multidrop network. It should be noted that the transmission line is terminated on both ends of the line but not at drop points in the

middle of the line. Termination should only be used with high data rates and where ever a long wire runs. The signal ground line is also recommended in an RS-485 system to keep the common mode voltage that the receiver must accept within the -7 to +12 volt range.

An RS-485 network can also be connected in a four-wire mode as shown in Fig.10. Note that four data wires and an additional signal ground wire is used in a "four-wire" connection. In a four-wire network it is necessary that one node be a master node and all others be slaves. The network is connected so that the master node communicates to all slave nodes. All slave nodes communicate only with the master node. Since the slave nodes never listen to another slave response to the master, a slave node cannot reply incorrectly to another slave node. A typical RS485 4-wire multidrop network has been chosen for building the local network for HMS.

2. Data Link Layer – This layer defines the formats of messages, how data is to be addressed, and error detection/correction techniques. In HMS local network, byte-based format is used to communicate between the master and the slave. There are future plans to extend the byte-based protocol into frame-based protocol with the error detection / correction using parity check. This is to increase the reliability and robustness of the local network. The master will send address, command and the arguments associated with the commands sequentially to the network.

3. Transport Layer - The transport layer defines how complete data files are to be handled over a network.

4. Application Layer – This layer defines the end-user standards for generating and/or interpreting communicated data in its final form. In other words, the actual computer programs using the communicated data.

The complete file handling and Graphic User Interface (GUI) corresponds to the transport layer and the application layer is under development. The proposed application layer implementation involves writing the master controller code in C language under LINUX platform. This project aims for a prototype development of building local network for HMS. The data collected from the monitoring points are not stored as files for analyses.

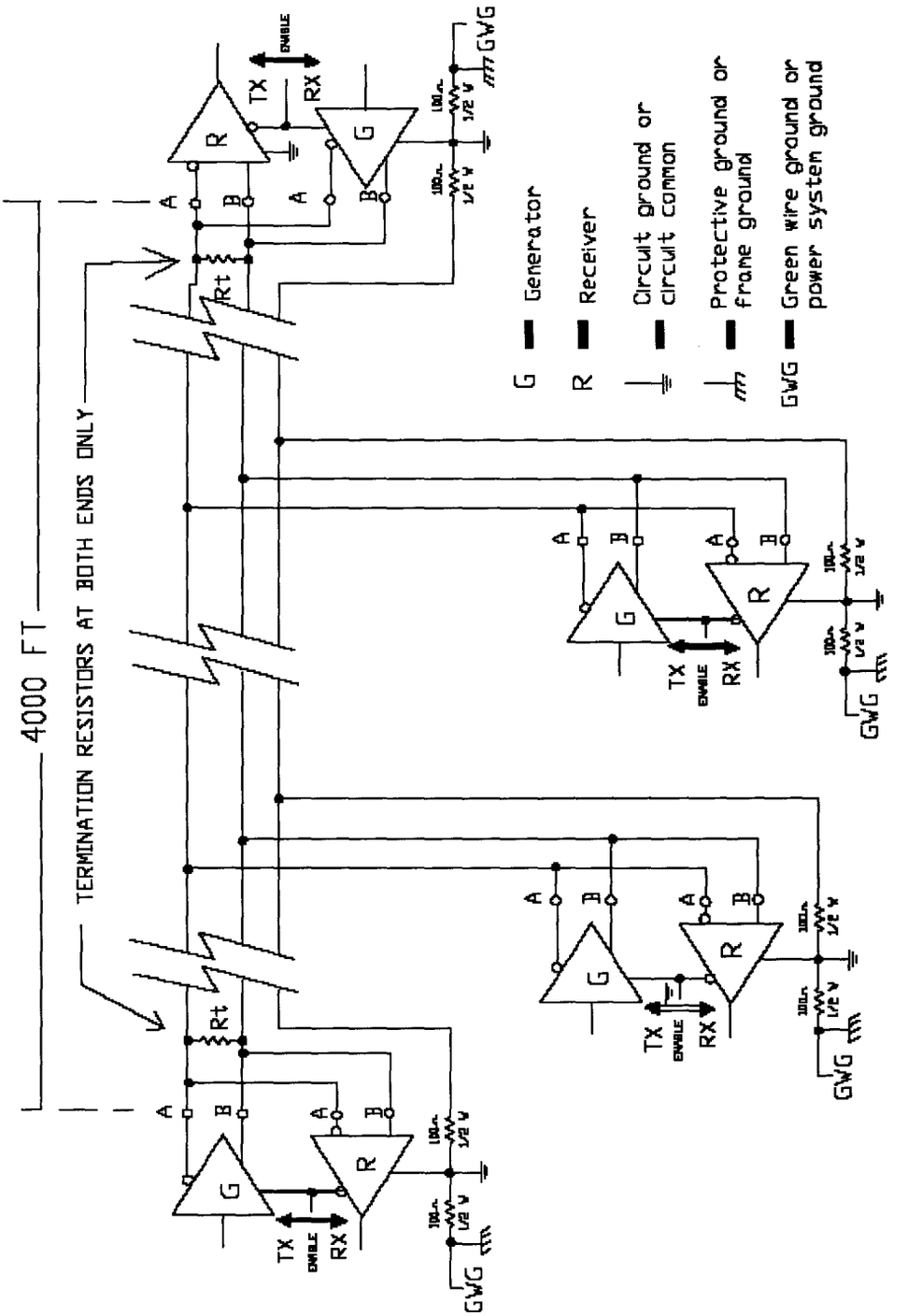


Fig.9: Typical RS-485 Two Wire Multidrop Network

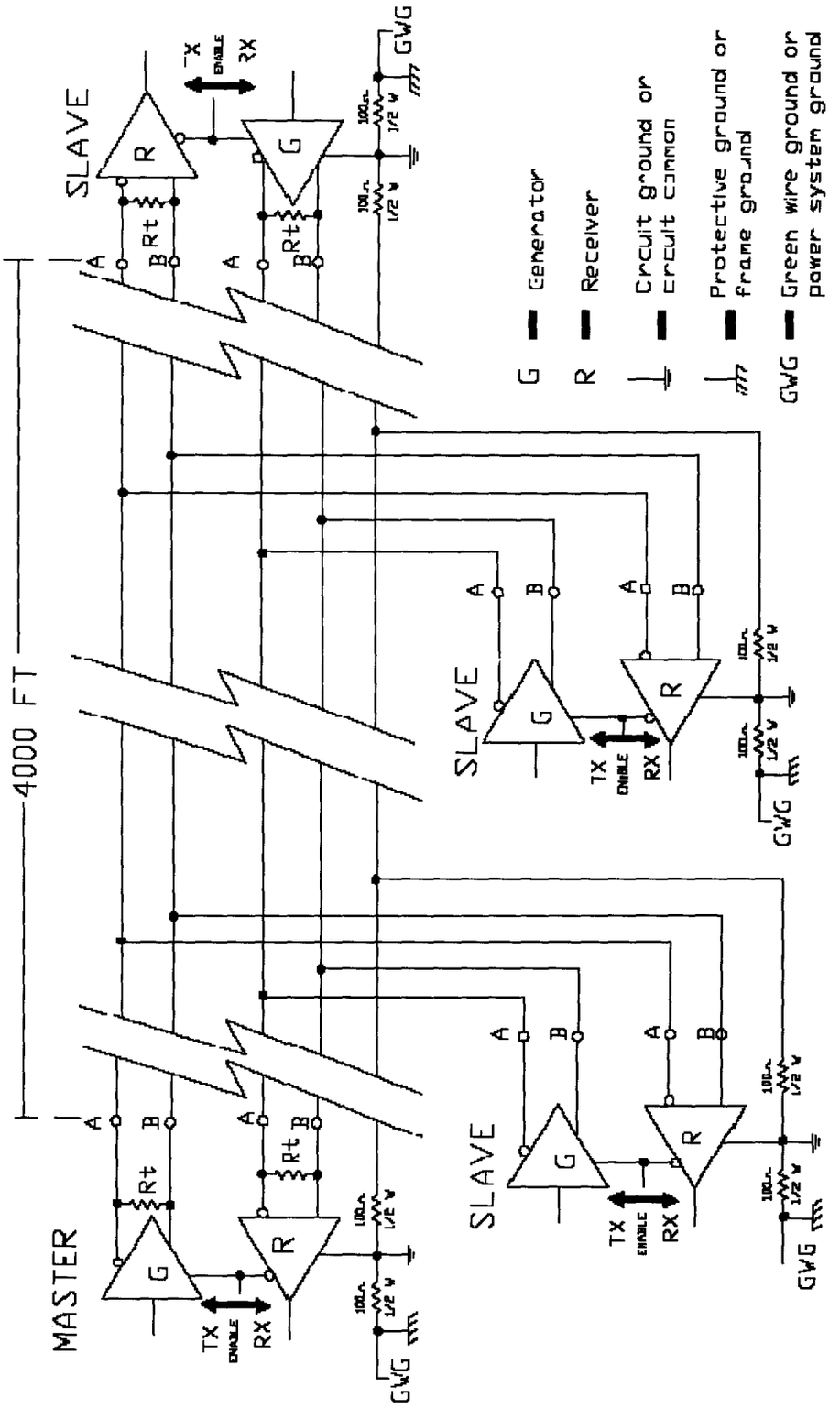


Fig.10: Typical RS-485 Four Wire Multidrop Network

3.3 Bus Topology

After the analysis of different network topologies, a bus topology is considered to be the most efficient and suitable for the HMS local network.

A bus network uses a multi-drop transmission medium. All nodes on the network share a common bus and thus share communication. This allows only one device to transmit at a time. A distributed access protocol determines which station is to transmit. The bus topology is passive. In other words, the nodes on the bus simply 'listen' for a signal; they are not responsible for moving the signal along. In the bus topology, failure of one of the station does not affect others. It is also a good compromise over the other two topologies (discussed earlier) as it allows relatively high rate of data transmission and easy to implement and extend.

Another significant aspect in considering bus topology is the fact that the radio telescope system is a distributed system spread over a wide area. The number of interconnecting cables, which already exist in the system, tends out to be quite large. This become very critical when the cables need to be drawn trough regions like cable wraps.

Fig.11 depicts the bus topology arrangement for the local network. As discussed earlier RS232 cannot be used to build long distance, multidrop bus network. Since the PC interface is always RS232, a need arises to convert the transmission line into RS485 level. Hence the converter hardware (RS232 – RS485 converter board) is designed.

A hardware which has the facility of programming and the communication with the PC is required for building the slave. Hence a microcontroller based intelligent slave is designed and developed for the local network. The slave system collects the data from the sensors and transmits it over the bus. The microcontroller in the slave system should be able to perform ADC (Analog to digital conversion) and even some purposes DAC (digital to Analog conversion). Since the data output from sensors could be in different formats and levels, interface electronics is required in between the sensor and the slave node in most cases.

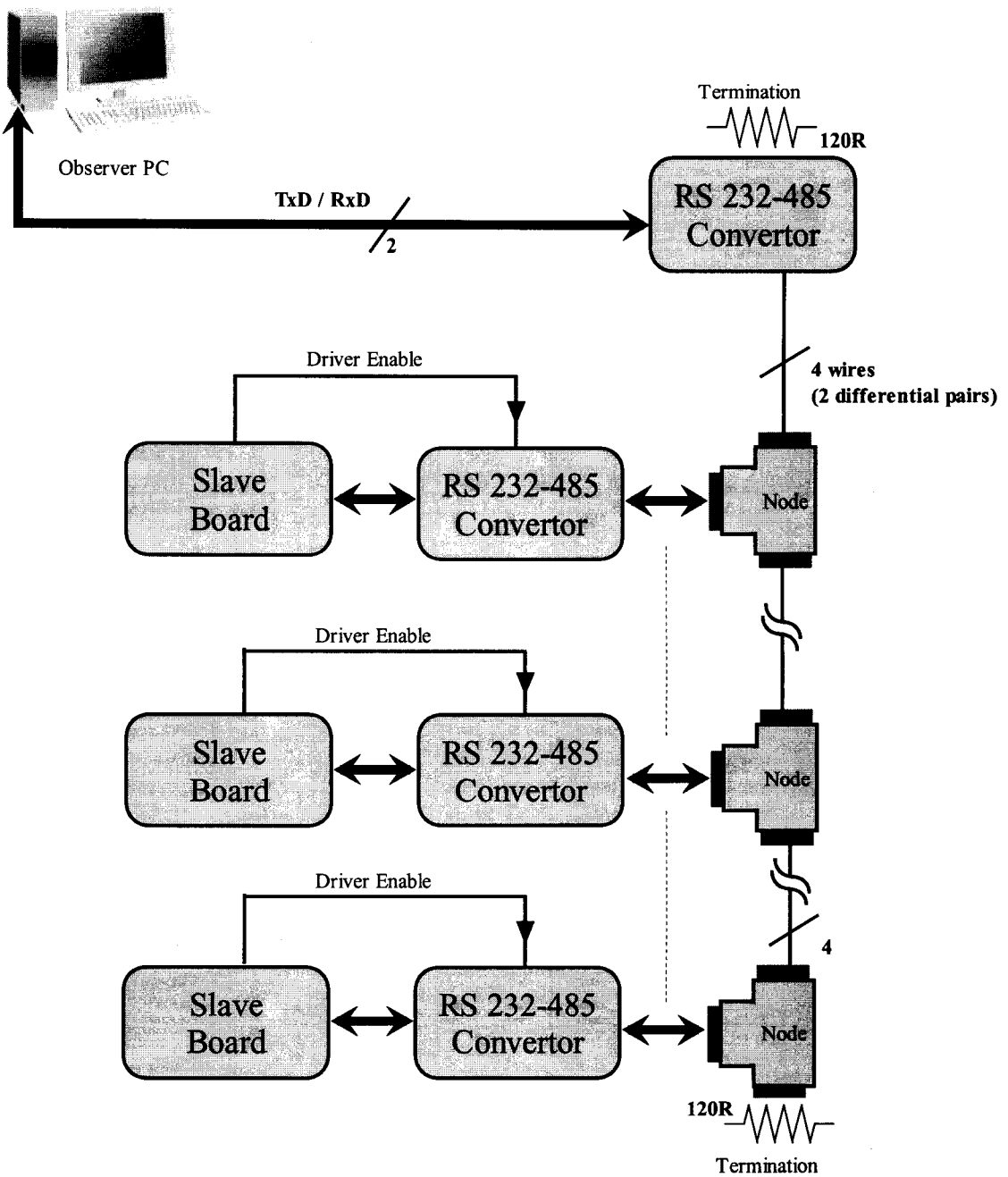


Fig.11:Bus topology arrangement for the local network

Chapter 4

Hardware Design and Implementation

4. Hardware Design and Implementation

4.1 Introduction

The hardware that needs to be designed for the local network involves several subsystems like bus level converters, microcontroller based slave systems to act as intelligent nodes and interface electronics for connection to a variety of sensors with different output levels. In this chapter we describe the design of these hardware components.

Knowing the structure of a character frame and the meaning of baud rate, as it applies to RS-232, the maximum transmission rate in characters per second, for a given communication setting can be calculated. This rate is just the baud rate divided by the bits per frame. In this case, since we are working on byte-based protocol, the transmission takes place at 9600 baud, hence the character per second will be 9600 / 8 = 1200. The 9600 baud is fixed as the crystal oscillator and timer chosen in the microcontroller.

4.2 RS232 – RS485 Converters

The communication at the master end, which is a PC, and at the slave end, which is basically a microcontroller, is through the standard RS232 serial port (UART). Due to the requirement of long distance noise immune communication, we are going for a RS 485 based local network for connecting the master with the slaves. Hence, a converter board, which translates the unbalanced single ended RS232 bus to a balanced differential RS485 bus, is designed. For the purpose of connecting the different slave nodes to the bus, appropriate T-taps has also been designed.

The converter card block diagram is shown in the figure 12. It contains a RS232 level converter followed by a RS485 transceiver. As the name suggests this is a bi-directional converter card. The facility is made to interface this card to the standard computer serial port (9 pin D male) directly. For debugging purposes, provision to loop back the RS232 side (checking internally the RS232 level converter) and the

RS485 side (checking internally the RS485 transceiver) has been included. In the present network hardware there is a master controller followed by multiple slave controllers on the same bus. When the master controller transmits to the slaves, there is no problem of bus contention or continuous short. But when the slave systems desire to transit there is a need to prevent bus contention between multiple slave outputs. A driver enable signal is brought out from corresponding slave microcontroller card to the converter board, which can be used to disable the transceivers driver line. Whenever the slave wants to communicate with the master it enables the driver line, transmits the bit pattern and immediately disables it. There are jumper settings in the transceiver card which facilitate to receive the driver enable signal from the slave or to continuously enable the driver / receiver if required. If this is not taken care, then the bus contention and the clash will occur which may damage the hardware and the reliable data communication is not possible.

MAX232

A simple way to translate from TTL levels to RS232 levels is to use one of the many chips designed for this purpose. Maxim Semiconductor was the first to offer RS-232 interface chips that require only a +5V power supply.

The MAX232 includes two drivers that convert TTL inputs to RS232 outputs and two receivers that accept RS232 inputs and translate them to TTL compatible outputs. The drivers and receivers also invert the signals. The chip contains two charge pump voltage converters that act as tiny unregulated power supplies that enable loaded RS232 outputs of +/- 7V or better. Four external capacitors store energy for the supplies. The recommended value for the capacitors is 1uf for MAX232C (Appendix 1).

The SP491 is a low power differential line driver/receiver meeting RS-485 and RS-422 standards with a maximum data rate of 5Mbps. It addition it also has driver and receiver tri-state enable lines.

The transmitter and receiver lines are always enabled for the first transceiver card, which interfaces to the PC side. The driver enable lines for the successive transceiver cards which are used to connect the slave systems will be controlled by the respective slaves.

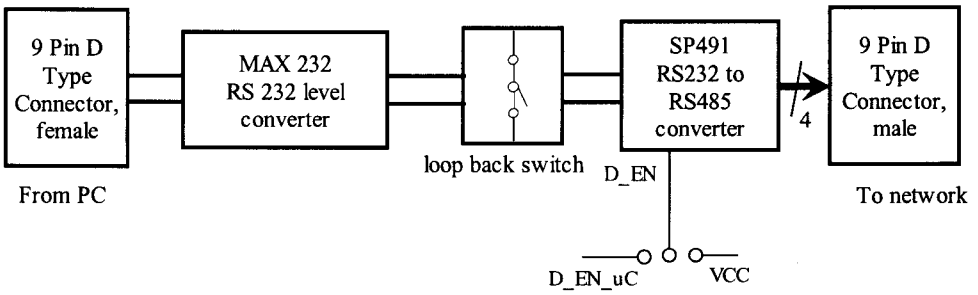


Fig. 12: Block diagram of RS232 to RS485 converter card

4.3 Design of Slave micro controller board

4.3.1 Choice of microcontroller.

As the complexity of the discrete digital logic and the FPGAs are more in designing and developing this kind of instrument. Also the FPGAs don't have the ADC's built-in. To ease the design and development complexity, the micrcontroller selection was made.

The health monitoring system involves acquisition of data from several sensors that give output as analog voltages. Hence it is desirable to choose a microcontroller with inbuilt ADC facility that can directly interface to the sensors. The ADuC832, which has been chosen for the system incorporates a 12 bit 8channel, multiplexed ADC.

Apart from this, the microcontroller uses a standard 8051 core, which has been the defacto industry standard for the last few decades.

4.3.2 Microcontroller features

The ADuC832 is a complete smart transducer front end, integrating a high performance self-calibrating multichannel 12-bit ADC, dual 12-bit DACs, and

programmable 8-bit MCU on a single chip. The device operates from a 32 kHz crystal with an on-chip PLL generating a high frequency clock of 16.77 MHz. This clock is, in turn, routed through a programmable clock divider from which the MCU core clock operating frequency is generated. The microcontroller core is an 8052 and therefore 8051 instruction set compatible with 12 core clock periods per machine cycle. 62 kBytes of nonvolatile Flash/EE Program memory are provided on-chip. 4 kBytes of nonvolatile Flash/EE data memory, 256 bytes RAM, and 2 kBytes of extended RAM are also integrated on-chip. The ADuC832 also incorporates additional analog functionality with two 12-bit DACs, power supply monitor, and a band gap reference. On-chip digital peripherals include two 16-bit-DACs, dual output 16-bit PWM, watchdog timer, time interval counter, three timers/counters, Timer 3 for baud rate generation, and serial I/O ports (SPI, I2C, and UART) On-chip factory firmware supports in-circuit serial download and debug modes (via UART) as well as single-pin emulation mode via the *EA* pin. Figure 13 shows the internal block diagram of AduC832 chip.

Applications

Optical Networking—Laser Power Control

Base Station Systems

Precision Instrumentation, Smart Sensors

Transient Capture Systems

DAS and Communications Systems

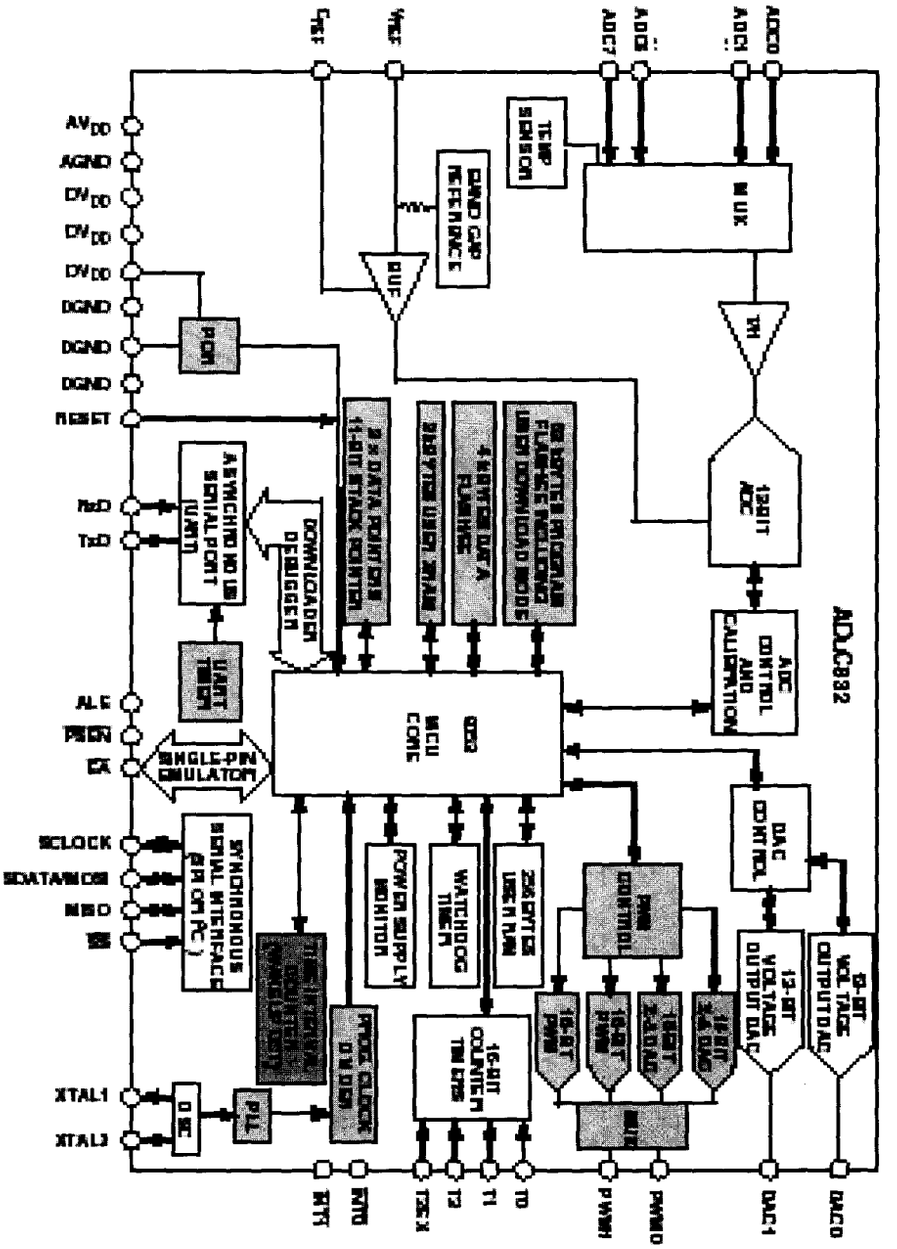


Fig. 13: Internal block diagram of AduC832 chip

General Overview

The ADC conversion block incorporates a fast, 8-channel, 12-bit, single supply A/D converter. This block provides the user with multi-channel mux, track/hold, on-chip reference, calibration features and A/D converter. All components in this block are easily configured via a 3-register SFR interface. The A/D converter consists of a conventional successive approximation converter based around a capacitor DAC. The converter accepts an analog input range of 0 to +VREF. A high precision, low drift and factory calibrated 2.5 V reference is provided on-chip. The internal reference may be overdriven via the external VREF pin. This external reference can be in the range 2.3 V to AVDD. Single step or continuous conversion modes can be initiated in software or alternatively by applying a convert signal to the an external pin. Timer 2 can also be configured to generate a repetitive trigger for ADC conversions. The ADC may be configured to operate in a DMA Mode whereby the ADC block continuously converts and captures samples to an external RAM space without any interaction from the MCU core. This automatic capture facility can extend through a 16 MByte external Data Memory space.

The ADuC812 is shipped with factory programmed calibration coefficients, which are automatically downloaded to the ADC on power-up ensuring optimum ADC performance. The ADC core contains internal Offset and Gain calibration registers. A software calibration routine is provided to allow the user to overwrite the factory programmed calibration coefficients if required, thus minimizing the impact of endpoint errors in the user's target system.

A voltage output from an On-Chip band-gap reference proportional to absolute temperature can also be routed through the front end ADC multiplexer (effectively a 9th ADC channel input) facilitating a temperature sensor implementation.

ADC Transfer Function

The analog input range for the ADC is 0 V to VREF. For this range, the designed code transitions occur midway between successive integer LSB values (i.e., 1/2 LSB, 3/2 LSBs, 5/2 LSBs . . . FS –3/2 LSBs). The output coding is straight binary with 1 LSB = FS/4096 or 2.5 V/4096 = 0.61 mV when VREF = 2.5 V. The ideal input/output transfer characteristic for the 0 to VREF range is shown in Figure 14.

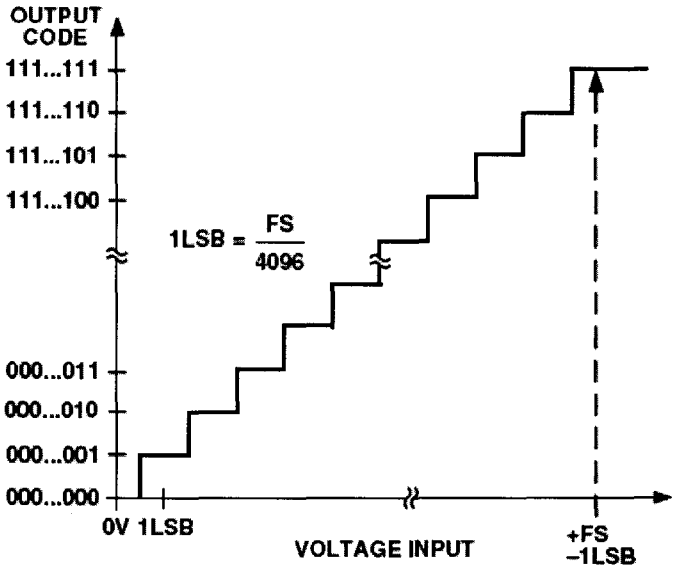
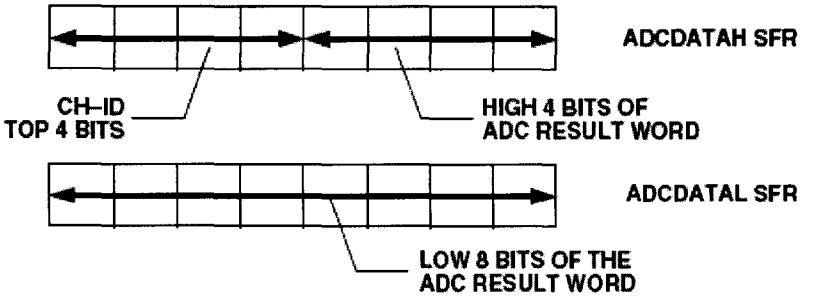


Fig. 14: ADC Transfer Function

Typical Operation

Once configured via the ADCCON 1–3 SFRs (shown on the following page) the ADC will convert the analog input and provide an ADC 12-bit result word in the ADCDATAH/L SFRs. The top 4 bits of the ADCDATAH SFR will be written with the channel selection bits to identify the channel result. The format of the ADC 12-bit result word is shown in the following Figure.



4.3.3 Board Design

The slave system is built using an embedded system, which involves a powerful microcontroller, which will trim down the burden on the PC by collecting data from the sensors independently, and passing it on when requested. Otherwise PC performing all the monitoring jobs will adversely affect the other important and high priority real time tasks in the control system. The microcontroller ADuC832 from Analog devices is used for this purpose. A patch board has been developed for the monitoring system, which contains a RS232 level converter and a Microcontroller.

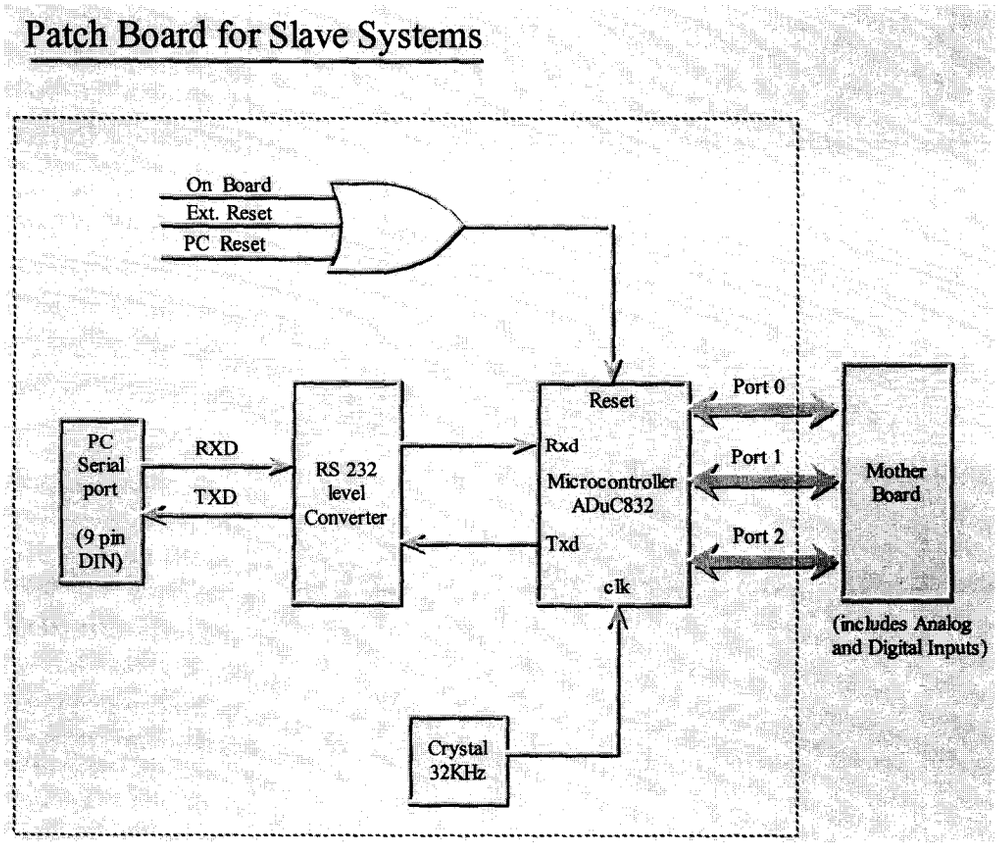


Fig.15: Block diagram of patch board for slave system

This board can be directly plugged into any standard PC serial port for in-circuit programming of the chip as well as to the custom made monitoring motherboard (not PC mother board) for continuous data acquisition. The idea behind this development is to prevent disturbing the hardware setup of the sensors situated at different locations, during program updation. Hence assembly code can be downloaded into this, just by de-latching the patch board and plugged in to the PC serial port directly to down load the HEX code of the assembly program. This facility will not disturb hardware setup of the sensor and motherboard. Figure 15 shows the block diagram of patch board used in the slave system.

4.4 Protocol Implementation

4.4.1 Introduction

Any two systems connected in a network need an efficient protocol to communicate each other. If the protocol is not proper then the communication between the two devices may not be proper and leads to malfunctioning. In the monitoring system as discussed earlier, the slave systems are on a network, controlled by the master PC called Control PC.

4.4.2 HMS Protocol

As there is no standard protocol available to meet the local network for the HMS, a custom built protocol named as HMS protocol has been designed. The HMS protocol is byte-based protocol without the parity check. This protocol allows a maximum of 2^5 (32) slave systems. This can easily cater to requirements for most of the radio telescope monitoring systems.

The address of each slave is already saved in its non-volatile memory. Whenever the control PC has to acquire data from a particular sensor connected to a slave system, it will send the address of that slave system. The slave system will identify its address and will send back an acknowledgement. The remaining slave systems will not respond, since the address sent and the address stored does not match. Once the acknowledgement from the slave system is received, the control PC will send a command followed by the argument(s). The slave system receives the command and takes required action.

In order to identify whether the bit patterns sent by the PC is an address or a command or an argument from the PC or the data sent back by the microcontroller to PC, a prefix system is used along with the sent bit pattern. Any data sent either by the PC or by the uC will be an 8-bit pattern with the three LSB bits representing the prefix. The prefixes used for the different bit pattern sent into the network are shown in the Table 2.

Sl. No.	Bit Pattern	Decoded as
1	XXXXX 000	Slave Address
2	XXXXX 001	Command to Slave
3	XXXXX 010	Argument not last
4	XXXXX 011	NOT USED - INVALID
5	XXXXX 100	RESERVED
6	XXXXX 101	RESERVED
7	XXXXX 110	Argument last
8	XXXXX 111	NOT USED - INVALID
9	XXXXX X11	Data from Slave to PC

Table 2: Prefix used in the protocol design

In the bit pattern shown in the table.1, X represents the bit value 0 or 1. The communication between the master and the slave is a byte-based protocol and uses asynchronous serial communication (UART-RS232). The identification of START and STOP bit is taken care in the serial communication. Figure 8 shows the asynchronous data transmission. Hence, a byte is taken into account and the protocol is designed. The pattern identification is as follows:

The LSB 3 bits containing '000' will be decoded as an address, where rest of the 5 bits gives the slave address. The slave can be addressed for 2^5 (32) combinations (00000 000 to 11111 000). LSB '001' is a prefix for a command identification to slave. The slave after receiving the command will act appropriately. A total of 32 commands can be processed. Any arguments send from the PC to the uC will have a prefix as '010'.Which denotes there are more arguments to follow. A prefix '110' identifies the end of argument list. This way multiple arguments can be sent and processed in the slave system.

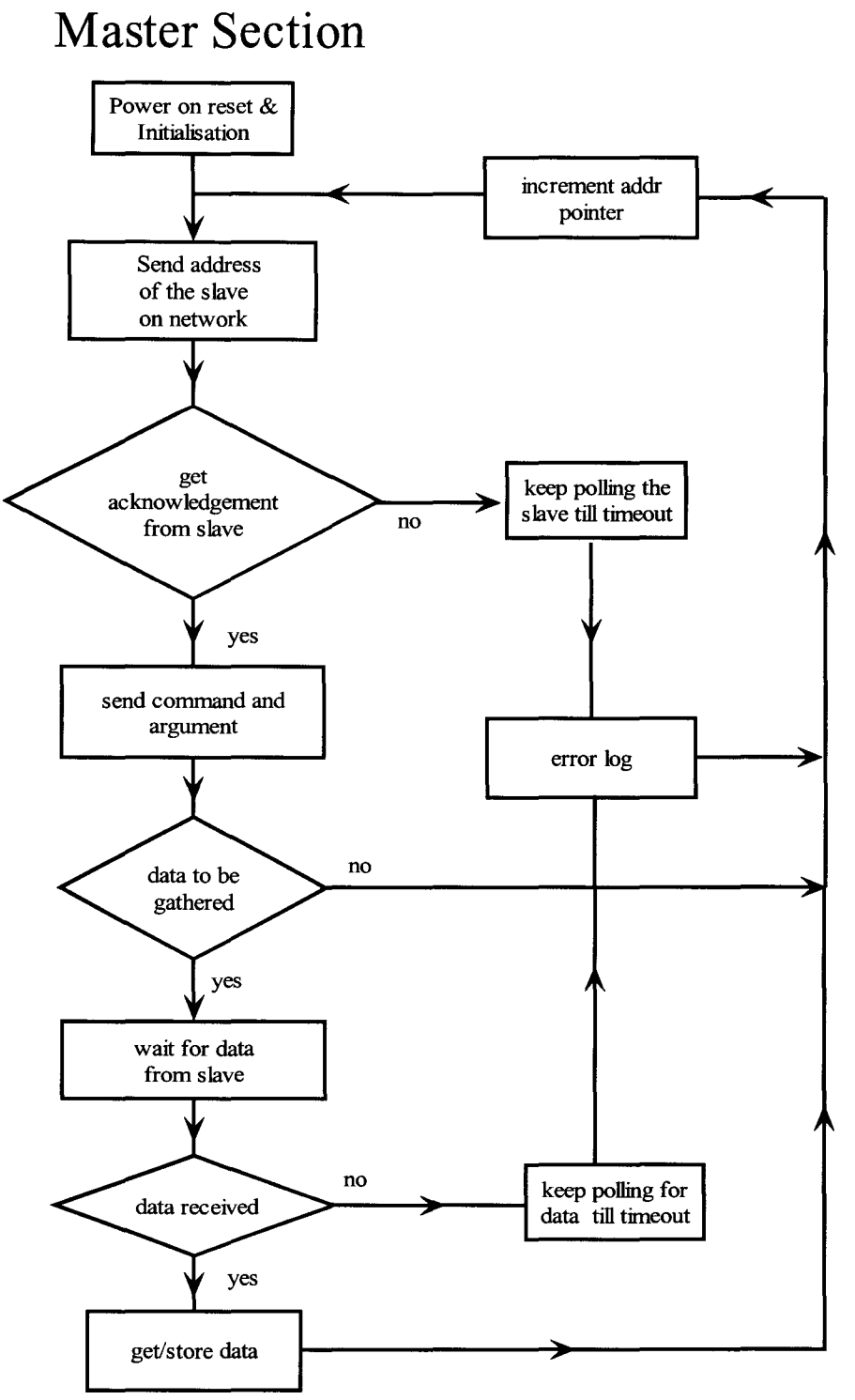
The prefix for sending the data from slave to PC will be of LSB two bits ' 11 '. The prefix in this case is restricted to two bits '11' due to the following reason.

The ADC output from the uC is of 2 bytes, out of which 12 bits are the ADC value and the remaining 4 bits represent the channel information. Since the uC sends ADC data to masters only in response to the master command, which already has the channel information, it is not required to send the channel information again along with the ADC bits.

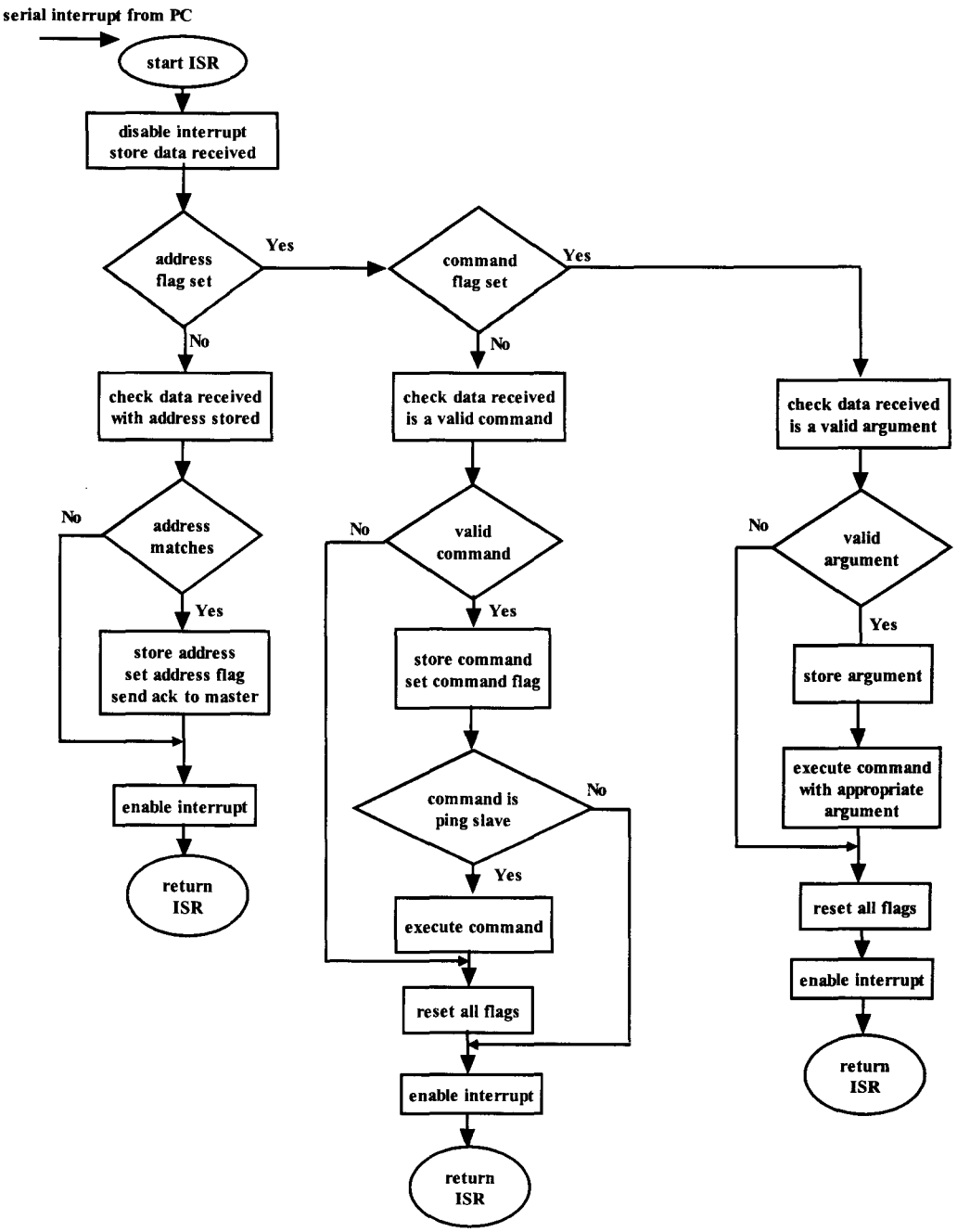
Using a three bit prefix even in this case will require the data to be send as three bytes which will have a tremendous impact on the data transfer speeds. The prefixes '111' and '011' are not used and are invalid in HMS protocol since the third LSB of ADC value can be '0' or '1'. The prefixes '100' and '101' are reserved for future expansion.

4.4.3 Protocol Implementation

The assembly language programs are written to implement the protocol in the microcontroller. Appendix 3 shows these programs. The programs are simulated using the Keil software and the HEX code of the same is programmed in the chip using Windows Serial Download software. The communication with this protocol is checked using Windows Hyper Terminal which supports hexadecimal input and output. Flow chart 1 shows the proposed protocol implementation in the master PC which will be implemented using C language under LINUX platform. Flow chart 2 shows the implementation which is executed using assembly language microcontroller program.



Flow Chart 1: Proposed protocol implementation in master PC



Flow Chart 2: Protocol implementation in slave system

A brief list of address patterns, commands, arguments and some of the functions developed in software are mentioned below. Refer appendix for the software programs in assembly.

Slave address bit pattern (for 5 slaves):

Slave number	Address	HEX Code
1	0000 1 000	08H
2	0001 0 000	10H
3	0001 1 000	18H
4	0010 0 000	20H
5	0010 1 000	28H

Sl.No	Command	Argument
1	ping_slave	
2	read_port_byte	<port_no>
3	read_port_bit	<port_no> <port_bit>
4	write_port_byte	<port_no>
5	write_port_bit	<port_no> <port_bit>
6	read_adc	<channel_no.>
7	scan_adc	<channel_no.> <no._of_scan>
8	scan_slave	
9	reset_slave	<slave_no>
10	reset_slave_all	

Table 3: List of Commands

Sl.no.	Command	Bit pattern B	HEX
1	ping_slave	0000 1 001 B	09H
2	read_port_byte	0001 0 001 B	11H
3	read_port_bit	0001 1 001 B	19H
4	write_port_byte	0010 0 001 B	21H
5	write_port_bit	0010 1 001 B	29H
6	read_adc	0011 0 001 B	31H
7	scan_adc	0011 1 001 B	39H
8	scan_slave	0100 0 001 B	41H
9	reset_slave	0100 1 001 B	49H
10	reset_slave_all	0101 0 001 B	51H

Table 4: Command bit pattern

Sl.no.	Command	Argument	ARG. _PATTERN
1	ping_slave		
2	read_port_byte	<port_no>	< xxxxx 110 >
3	read_port_bit	<port_no><port_bit>	< xxxxx 010 > < xxxxx 110 >
4	write_port_byte	<port_no>	< xxxxx 110 >
5	write_port_bit	<port_no> <port_bit>	< xxxxx 010 > < xxxxx 110 >
6	read_adc	<channel_no.>	< xxxxx 110 >
7	scan_adc	<channel_no.> <no._of_scan>	< xxxxx 010 > < xxxxx 110 >
8	scan_slave		
9	reset_slave	<slave_no>	<xxxxx 110 >
10	reset_slave_all		

Table 5: Argument bit pattern

Brief Description about the commands

All these commands are issued from the master computer, which is running the Linux operating system.

ping_slave

This command is used to check whether a particular slave Microcontroller system is functioning properly and the communication link to that system is working or not. If the slave system is healthy, it will respond to this command by sending its address five times.

read_port_byte

This command is used to read a byte from the slave system. This command reads one byte of data. This is the command used for reading the data from the microcontroller slave systems.

read_port_bit

This command is used to read any one bit of data from the slave system. This command helps to read certain configuration bits of the slave system.

write_port_byte

This command is used to write one byte of information to the slave system.

write_port_bit

This command is to write any one bit of data to the slave system.

read_adc

This command is used to read the ADC channel data from the slave microcontroller system.

scan_adc

This command is used to scan through all the ADC channels in the slave system. This gives the snapshot of all the ADC channels.

scan_slave

This command is used to read all the data at one shot from one slave microcontroller system.

reset_slave

This command is used to reset a particular slave system.

reset_slave_all

This command is used to reset all the slaves in setup. This command is used by the master computer.

Chapter 5

Testing and Conclusion

5. Testing and Conclusion

Any system needs to be integrated and configured in an efficient way to obtain the desired results. The local network for the 12m radio telescope health monitoring system developed in this project was configured for testing as shown in Figure 16. A length of 4 meter wire was used to connect the observer PC serial output to RS232 – RS485 converter. This was in turn connected via a 3 meter long cable to a node which is a 3 port device. One of the outputs of this is a continuation of the RS485 bus and the other output provides a connection to the microcontroller slave board. However, this slave board accepts a RS232 signal and hence the output of the node is fed to the slave card via a RS232-RS485 converter. For testing we used a 2 drop with microcontroller slave cards. The slave cards were fed with the output of a 1.5V battery connected through a potentiometer. This enables us to feed a variable voltage as a substitute for the output of the sensor. Since the 12m radio telescope is under construction, there are no actual sensors available to interconnect with the slave systems. A battery of 1.5V was connected to a potentiometer for testing.

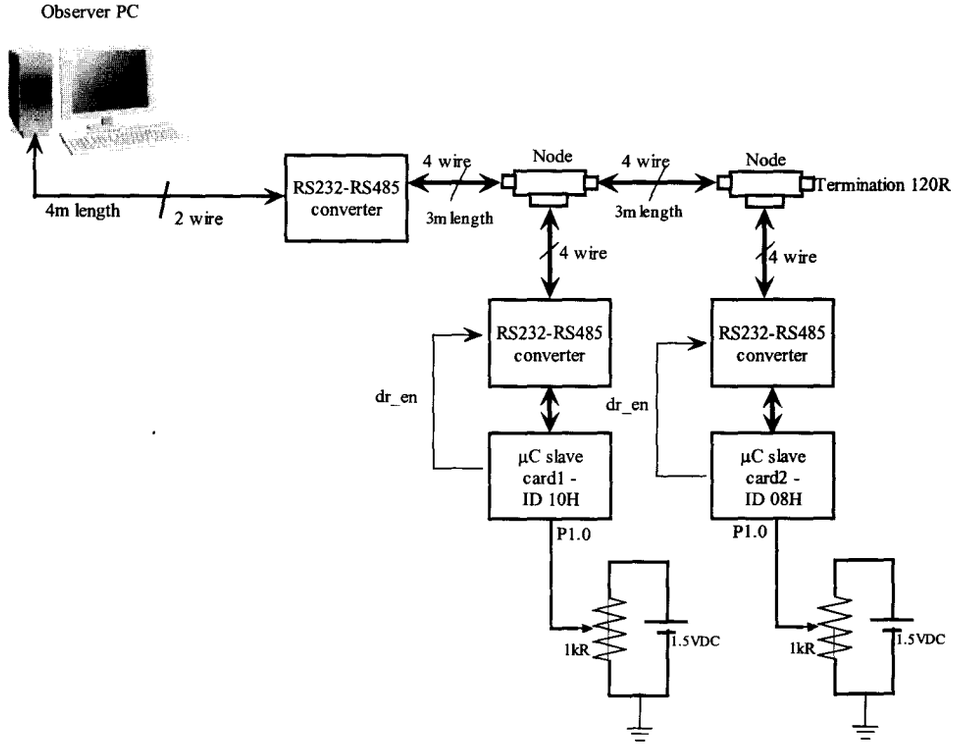


Fig.16: Test setup for local network

5.1 Testing of system hardware and software

The slave patch board as discussed in the chapter 4 (Figure 15) contains a microcontroller which has a facility of in-system programming. The assembly code for implementing the protocol for the HMS local network is simulated using Keil software where the simulation of the program has been executed. The Keil software creates a HEX equivalent data file of assembly code. This HEX code will be downloaded to the microcontroller using the Windows Serial Downloader (WSD), Version 6.4 which is a freely downloadable software from Analog Devices to program AduC832 microcontroller. The procedure to download the HEX code is as follows: The microcontroller card can be either plugged into the serial port of the PC or over the local network by enabling the PSEN pin. Figure 22 shows the configuration to be made to create the HEX file. To download this HEX data file into the microcontroller, reset the microcontroller, open the WSD software, the crystal frequency has to be set as watch crystal since the crystal used in the microcontroller is of 32kHz which will be converted to 16.77 MHz inside the chip using built-in PLL. Figure 17, 18 and 19 shows the signals that are used to configure WSD. The figures are self explanatory. Once the programming is complete PSEN pin can be removed from the board.

The local network is tested using the hyper-terminal which supports the hexadecimal data transmit / receiver facility. The sequence of sending the appropriate address, command followed by the argument associated with it will be executed through this hyper-terminal. All the commands (refer Chapter 4 – Table 4 and 5) are executed and tested. Figure 20, 21, and 22 demonstrates the communication between the master and the slave.

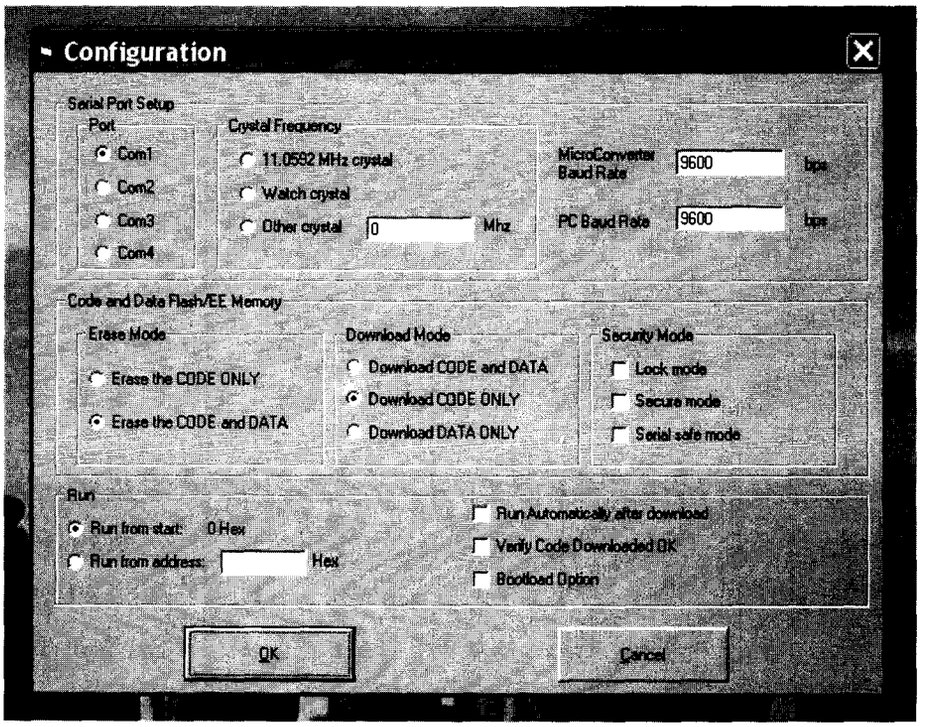


Figure 17: WSD configuration screen

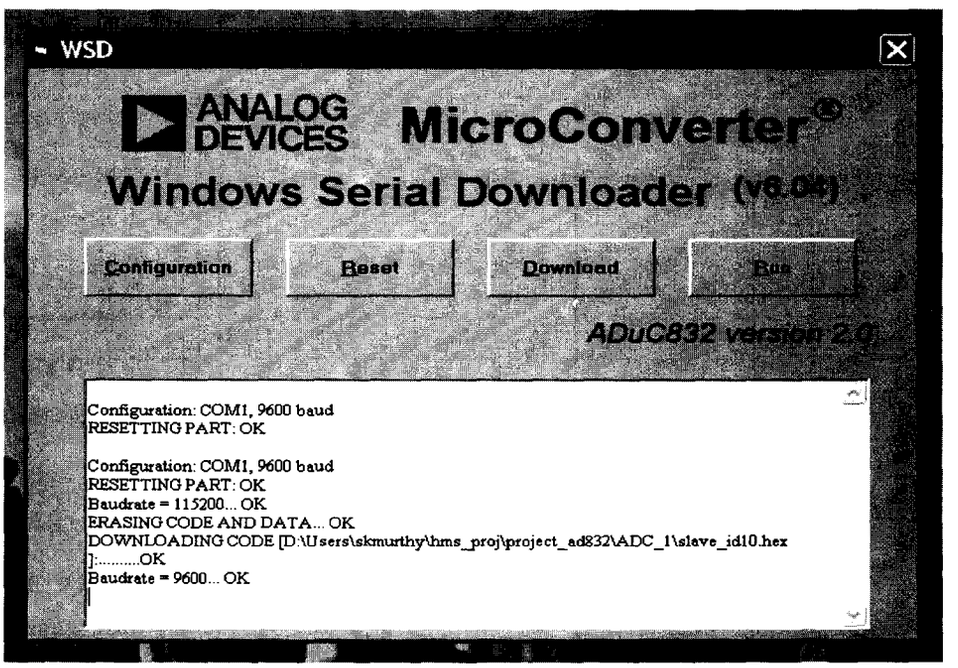


Figure 18: Screen showing WSD executed in-system programming of HEX code

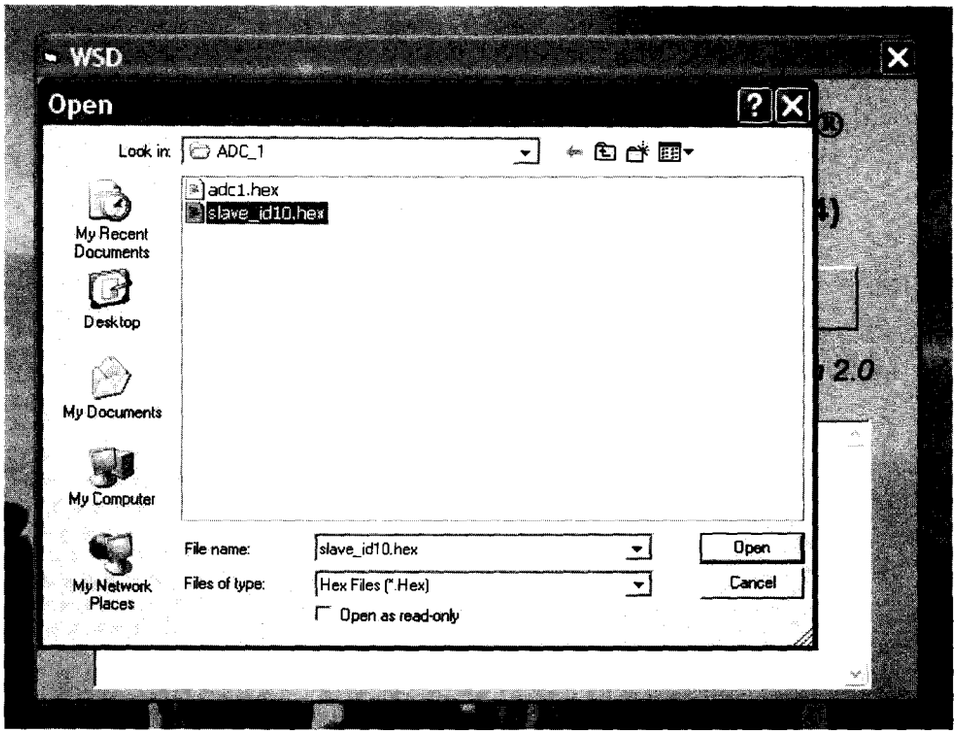


Figure 19: Selecting the appropriate HEX data file

Figure 20 shows the configuration of HEX Com tool hyperterminal. It indicates that the Com1 port is selected with the Baud rate of 9600 with no parity with one stop bit. The data transmission is of one byte.

Figure 21 shows that in the "Transmit HEX Field" 10H [00010000B] is sent to the slave which is the address of the slave ID (Figure 16) In the "Receive HEX Field" 13H [00010011B] is received which is nothing but the acknowledgement sent back from that particular slave to master. The acknowledgement received from the slave is 13H since the last two LSB is padded with 11 as per the design of protocol.

Figure 22 shows that a command ping_slave is sent (refer Section 4.4). The command pattern for which is 09H. On receiving this command, the slave will retransmit the address by padding 11 at two LSB to 10H, 5 times which is shown in second and third line of "Receive HEX Field".

Figure 20: Selecting the Baud rate for local network using RS232 HEX COM tool – hyperterminal

Figure 21: Example of sending the slave ID addressed 10H and receiving 13H as acknowledgement

Figure 22: Demonstrating the ping_slave command – 13 is echoed back five times form the slave

This was followed by a scan_adc command which enables the microcontroller card to acquire the data from the sensor and then transmits it to the master. The voltage fed to the ADC was measure with a 4 ½ digit multimeter. This was compared with the code received at the PC to demonstrate the functioning of the ADC and communication port. This was repeated for several values of the input voltage

5.2 Challenges encountered

As no standard protocol was available, the complete protocol development was carried out in-house. This was a very interesting design and its completion was indeed an exciting experience. The required network bus structure, which was also designed, developed, starting from the PCB layout design to card making, component assembling, and tested for communication without any bus contention. This indeed gave me a feeling of some achievement. However there were many limitations.

5.3 Limitations

This protocol is a byte-based protocol. It does not support the frame-based protocols. Here in this system the parity check has not been incorporated, since the 12m dish is not complete, one did not have all the set up required to test 10 drops with all its real time sensors.

5.4 Future Plans

This work can be expanded with more number of subsystems. Each subsystem can take different data and do a bit of processing before sending the information back to the master computer. Each subsystem can be made to have their own self test routines once powered (power on self test – POST). If any subsystem is having problems, then it can communicate the same to the master computer.

This system can be made to have a frame based protocol with provision for the parity check. This will help in making the network reliable.

The central PC can be replaced by a Master microcontroller system in the second phase. This will reduce the load on the computer. The master microcontroller can take care of acquiring sensor data and sending commands to the slave systems. It can occasionally interrupt the computer when required and in a burst mode, communicate the essence of all the previous readings and also interrupt the computer when it notices a serious fault in the system.

Appendix 1

MicroConverter®, 12-Bit ADCs and DACs with Embedded 62 kBytes Flash MCU

ADuC832

FEATURES

ANALOG I/O
- 8-Channel, 247 kSPS 12-Bit ADC
  - DC Performance: ±1 LSB INL
  - AC Performance: 71 dB SNR
- DMA Controller for High Speed ADC-to-RAM Capture
- 2 12-Bit (Monotonic) Voltage Output DACs
- Dual Output PWM/Σ-Δ DACs
- On-Chip Temperature Sensor Function ±3°C
- On-Chip Voltage Reference

Memory
- 62 kBytes On-Chip Flash/EE Program Memory
- 4 kBytes On-Chip Flash/EE Data Memory
- Flash/EE, 100 Yr Retention, 100 kCycles Endurance
- 2304 Bytes On-Chip Data RAM

8051-Based Core
- 8051 Compatible Instruction Set (16 MHz Max)
- 32 kHz Ext Crystal, On-Chip Programmable PLL
- 12 Interrupt Sources, 2 Priority Levels
- Dual Data Pointer
- Extended 11-Bit Stack Pointer

On-Chip Peripherals
- Time Interval Counter (TIC)
- UART, I²C®, and SPI® Serial I/O
- Watchdog Timer (WDT), Power Supply Monitor (PSM)

Power
- Specified for 3 V and 5 V Operation
- Normal, Idle, and Power-Down Modes
- Power-Down: 25 μA @ 3 V with Wake-Up cct Running

APPLICATIONS

Optical Networking—Laser Power Control
Base Station Systems
Precision Instrumentation, Smart Sensors
Transient Capture Systems
DAS and Communications Systems

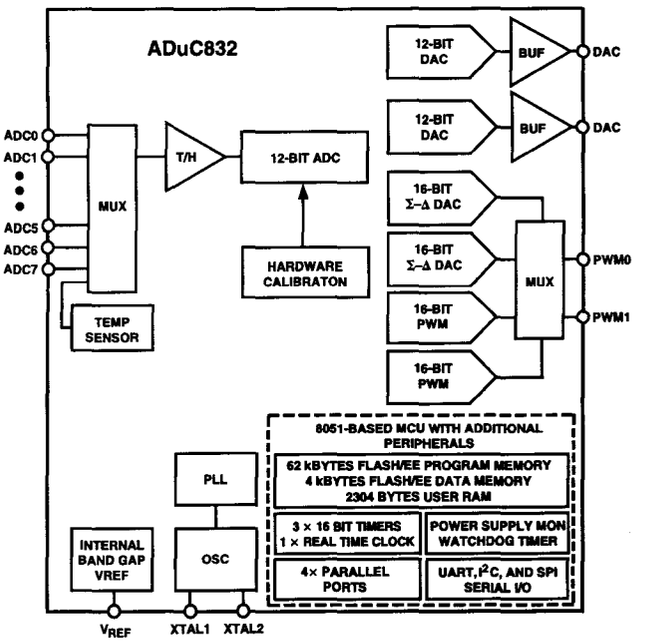
Upgrade to ADuC812 Systems. Runs from 32 kHz External Crystal with On-Chip PLL.

Also Available: ADuC831 Pin Compatible Upgrade to Existing ADuC812 Systems that Require Additional Code or Data Memory. Runs from 1 MHz–16 MHz External Crystal.

MicroConverter is a registered trademark and QuickStart is a trademark of Analog Devices, Inc.
SPI is a registered trademark of Motorola, Inc.
I²C is a registered trademark of Philips Corporation.

REV. 0

FUNCTIONAL BLOCK DIAGRAM



GENERAL DESCRIPTION

The ADuC832 is a complete smart transducer front end, integrating a high performance self-calibrating multichannel 12-bit ADC, dual 12-bit DACs, and programmable 8-bit MCU on a single chip.

The device operates from a 32 kHz crystal with an on-chip PLL generating a high frequency clock of 16.77 MHz. This clock is, in turn, routed through a programmable clock divider from which the MCU core clock operating frequency is generated. The microcontroller core is an 8052 and therefore 8051 instruction set compatible with 12 core clock periods per machine cycle. 62 kBytes of nonvolatile Flash/EE program memory are provided on-chip. 4 kBytes of nonvolatile Flash/EE data memory, 256 bytes RAM, and 2 kBytes of extended RAM are also integrated on-chip.

The ADuC832 also incorporates additional analog functionality with two 12-bit DACs, power supply monitor, and a band gap reference. On-chip digital peripherals include two 16-bit Σ-Δ DACs, dual output 16-bit PWM, watchdog timer, time interval counter, three timers/counters, Timer 3 for baud rate generation, and serial I/O ports (SPI, I²C, and UART)

On-chip factory firmware supports in-circuit serial download and debug modes (via UART) as well as single-pin emulation mode via the $\overline{EA}$ pin. The ADuC832 is supported by QuickStart™ and QuickStart Plus development systems featuring low cost software and hardware development tools. A functional block diagram of the ADuC832 is shown above with a more detailed block diagram shown in Figure 1.

The part is specified for 3 V and 5 V operation over the extended industrial temperature range and is available in a 52-lead plastic quad flatpack package and a 56-lead chip scale package.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781/329-4700 **www.analog.com**
Fax: 781/326-8703

ABSOLUTE MAXIMUM RATINGS*

(T_A = 25°C, unless otherwise noted.)

AV_{DD} to DV_{DD} ..... –0.3 V to +0.3 V
AGND to DGND ..... –0.3 V to +0.3 V
DV_{DD} to DGND, AV_{DD} to AGND .....–0.3 V to +7 V
Digital Input Voltage to DGND .... –0.3 V to DV_{DD} + 0.3 V
Digital Output Voltage to DGND ... –0.3 V to DV_{DD} + 0.3 V
V_{REF} to AGND .....–0.3 V to AV_{DD} + 0.3 V
Analog Inputs to AGND .....–0.3 V to AV_{DD} + 0.3 V
Operating Temperature Range Industrial
ADuC832BS .....–40°C to +125°C
Operating Temperature Range Industrial
ADuC832BCP ..... –40°C to +85°C
Storage Temperature Range .....–65°C to +150°C
Junction Temperature ..... 150°C
θ_{JA} Thermal Impedance (ADuC832BS) ..... 90°C/W
θ_{JA} Thermal Impedance (ADuC832BCP) ..... 52°C/W
Lead Temperature, Soldering
Vapor Phase (60 sec) ..... 215°C
Infrared (15 sec) ..... 220°C

*Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

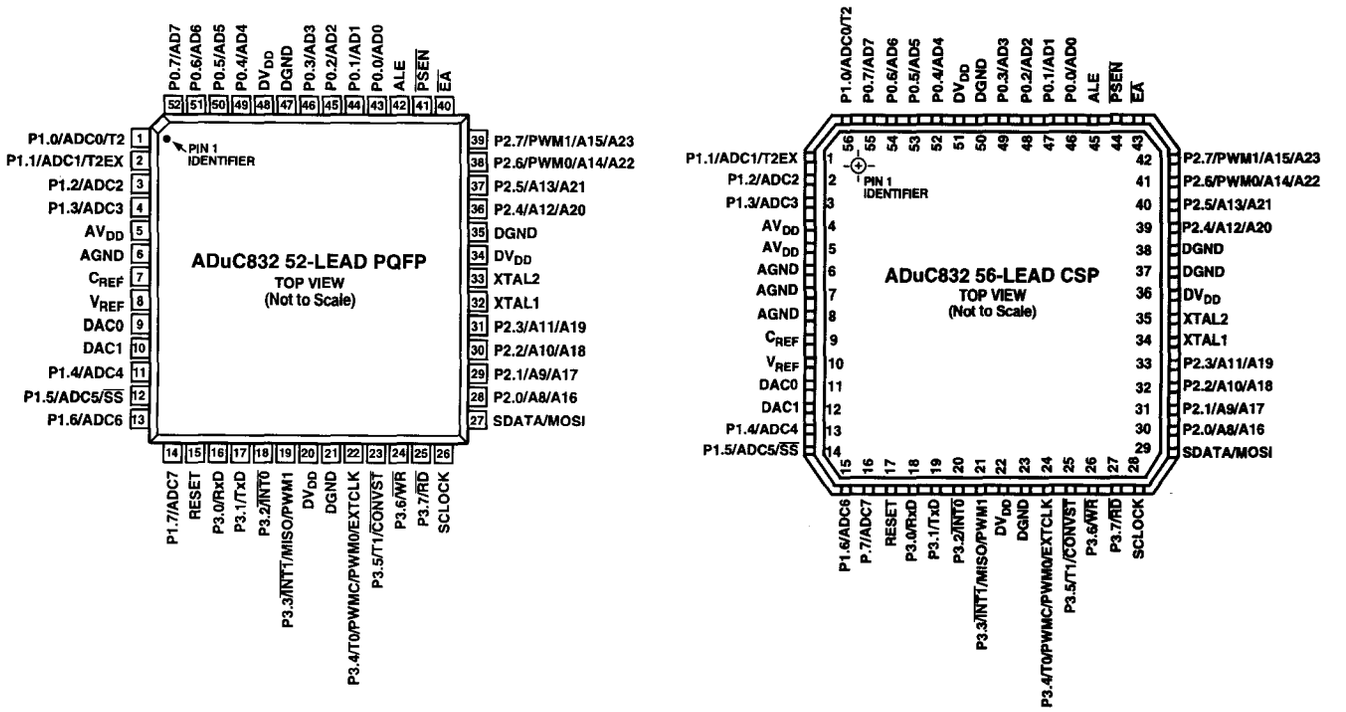
ORDERING GUIDE

Model	Temperature Range	Package Description	Package Option
ADuC832BS	–40°C to +125°C	52-Lead Plastic Quad Flatpack	S-52
ADuC832BCP	–40°C to +85°C	56-Lead Chip Scale Package	CP-56
EVAL-ADuC832QS		QuickStart Development System	
EVAL-ADuC832QSP		QuickStart Plus Development System	

CAUTION

ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily accumulate on the human body and test equipment and can discharge without detection. Although the ADuC832 features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

PIN CONFIGURATION



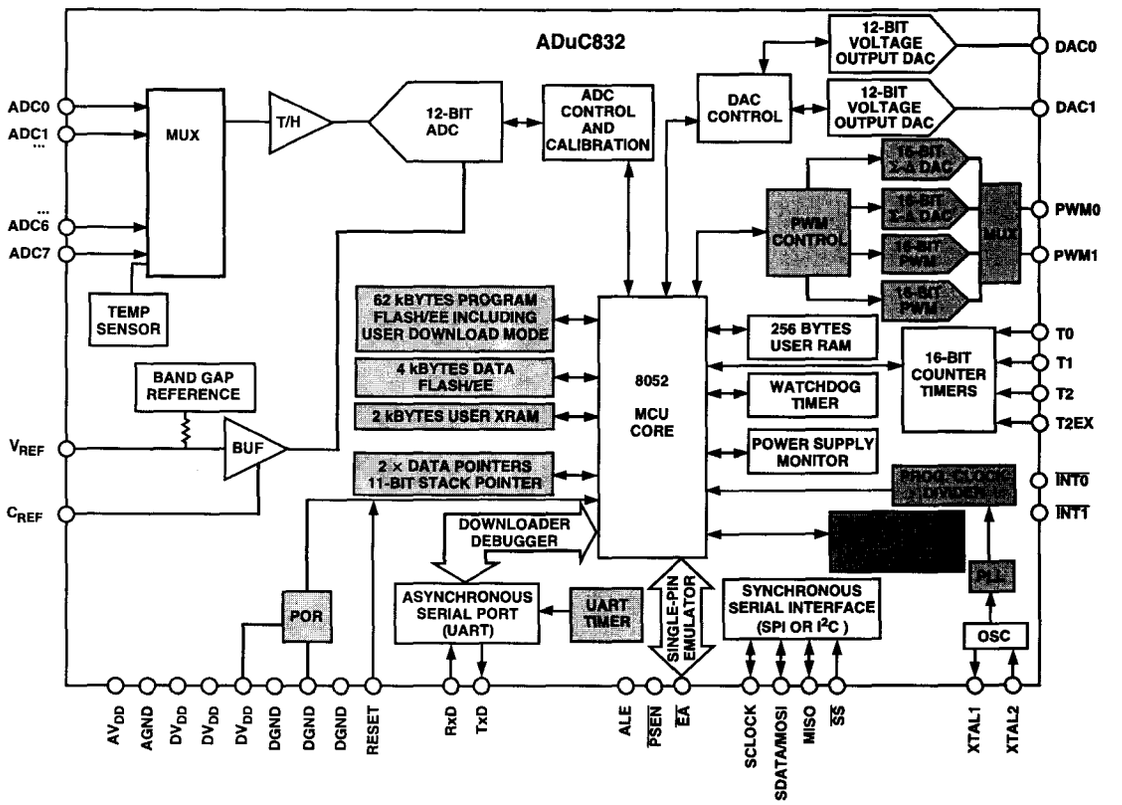


Figure 1. ADuC832 Block Diagram (Shaded Areas are Features Not Present on the ADuC812)

PIN FUNCTION DESCRIPTIONS

Mnemonic	Type	Function
DV_{DD}	P	Digital Positive Supply Voltage, 3 V or 5 V Nominal
AV_{DD}	P	Analog Positive Supply Voltage, 3 V or 5 V Nominal
C_{REF}	I/O	Decoupling Input for On-Chip Reference. Connect 0.1 µF between this pin and AGND.
V_{REF}	I/O	Reference Input/Output. This pin is connected to the internal reference through a series resistor and is the reference source for the analog-to-digital converter. The nominal internal reference voltage is 2.5 V, which appears at the pin. See ADC section on how to connect an external reference.
AGND	G	Analog Ground. Ground reference point for the analog circuitry.
P1.0–P1.7	I	Port 1 is an 8-bit input port only. Unlike other ports, Port 1 defaults to Analog Input mode. To configure any of these Port Pins as a digital input, write a "0" to the port bit. Port 1 pins are multifunction and share the following functionality.
ADC0–ADC7	I	Analog Inputs. Eight single-ended analog inputs. Channel selection is via ADCCON2 SFR.
T2	I	Timer 2 Digital Input. Input to Timer/Counter 2. When enabled, Counter 2 is incremented in response to a 1-to-0 transition of the T2 input.
T2EX	I	Digital Input. Capture/Reload trigger for Counter 2; also functions as an Up/Down control input for Counter 2.
$\overline{SS}$	I	Slave Select Input for the SPI Interface
SDATA	I/O	User Selectable, I^2C Compatible or SPI Data Input/Output Pin
SCLOCK	I/O	Serial Clock Pin for I^2C Compatible or SPI Serial Interface Clock
MOSI	I/O	SPI Master Output/Slave Input Data I/O Pin for SPI Interface
MISO	I/O	SPI Master Input/Slave Output Data I/O Pin for SPI Serial Interface
DAC0	O	Voltage Output from DAC0
DAC1	O	Voltage Output from DAC1
RESET	I	Digital Input. A high level on this pin for 24 master clock cycles while the oscillator is running resets the device.
P3.0–P3.7	I/O	Port 3 is a bidirectional port with internal pull-up resistors. Port 3 pins that have 1s written to them are pulled high by the internal pull-up resistors, and in that state can be used as inputs. As inputs, Port 3 pins being pulled externally low will source current because of the internal pull-up resistors. Port 3 pins also contain various secondary functions that are described below.
PWMC	I	PWM Clock Input
PWM0	O	PWM0 Voltage Output. PWM outputs can be configured to uses ports 2.6 and 2.7 or 3.4 and 3.3
PWM1	O	PWM1 Voltage Output. See CFG832 Register for further information.
RxD	I/O	Receiver Data Input (Asynchronous) or Data Input/Output (Synchronous) of Serial (UART) Port
TxD	O	Transmitter Data Output (Asynchronous) or Clock Output (Synchronous) of Serial (UART) Port
$\overline{INT0}$	I	Interrupt 0, programmable edge or level triggered Interrupt input, can be programmed to one of two priority levels. This pin can also be used as a gate control input to Timer 0.
$\overline{INT1}$	I	Interrupt 1, programmable edge or level triggered Interrupt input, can be programmed to one of two priority levels. This pin can also be used as a gate control input to Timer 1.
T0	I	Timer/Counter 0 Input
T1	I	Timer/Counter 1 Input
$\overline{CONVST}$	I	Active Low Convert Start Logic Input for the ADC Block when the External Convert Start Function is enabled. A low-to-high transition on this input puts the track-and-hold into its hold mode and starts conversion.
EXTCLK	I	Input for External Clock Signal; has to be enabled via CFG832 Register.
$\overline{WR}$	O	Write Control Signal, Logic Output. Latches the data byte from Port 0 into the external data memory.
$\overline{RD}$	O	Read Control Signal, Logic Output. Enables the external data memory to Port 0.
XTAL2	O	Output of the Inverting Oscillator Amplifier
XTAL1	I	Input to the Inverting Oscillator Amplifier
DGND	G	Digital Ground. Ground reference point for the digital circuitry.
P2.0–P2.7 (A8–A15) (A16–A23)	I/O	Port 2 is a bidirectional port with internal pull-up resistors. Port 2 pins that have 1s written to them are pulled high by the internal pull-up resistors, and in that state can be used as inputs. As inputs, Port 2 pins being pulled externally low will source current because of the internal pull-up resistors. Port 2 emits the high order address bytes during fetches from external program memory and middle and high order address bytes during accesses to the external 24-bit external data memory space.

PIN FUNCTION DESCRIPTIONS (continued)

Mnemonic	Type	Function
$\overline{\text{PSEN}}$	O	Program Store Enable, Logic Output. This output is a control signal that enables the external program memory to the bus during external fetch operations. It is active every six oscillator periods except during external data memory accesses. This pin remains high during internal program execution. $\overline{\text{PSEN}}$ can also be used to enable serial download mode when pulled low through a resistor on power-up or RESET.
ALE	O	Address Latch Enable, Logic Output. This output is used to latch the low byte (and page byte for 24-bit address space accesses) of the address into external memory during normal operation. It is activated every six oscillator periods except during an external data memory access.
$\overline{\text{EA}}$	I	External Access Enable, Logic Input. When held high, this input enables the device to fetch code from internal program memory locations 0000H to 1FFFH. When held low, this input enables the device to fetch all instructions from external program memory. This pin should not be left floating.
P0.7–P0.0 (A0–A7)	I/O	Port 0 is an 8-Bit Open-Drain Bidirectional I/O Port. Port 0 pins that have 1s written to them float and in that state can be used as high impedance inputs. Port 0 is also the multiplexed low order address and data bus during accesses to external program or data memory. In this application it uses strong internal pull-ups when emitting 1s.

TERMINOLOGY

ADC SPECIFICATIONS

Integral Nonlinearity
This is the maximum deviation of any code from a straight line passing through the endpoints of the ADC transfer function. The endpoints of the transfer function are zero scale, a point 1/2 LSB below the first code transition, and full scale, a point 1/2 LSB above the last code transition.

Differential Nonlinearity
This is the difference between the measured and the ideal 1 LSB change between any two adjacent codes in the ADC.

Offset Error
This is the deviation of the first code transition (0000 . . . 000) to (0000 . . . 001) from the ideal, i.e., +1/2 LSB.

Gain Error
This is the deviation of the last code transition from the ideal AIN voltage (Full Scale – 1.5 LSB) after the offset error has been adjusted out.

Signal to (Noise + Distortion) Ratio
This is the measured ratio of signal to (noise + distortion) at the output of the ADC. The signal is the rms amplitude of the fundamental. Noise is the rms sum of all nonfundamental signals up to half the sampling frequency ($f_S/2$), excluding dc. The ratio is dependent upon the number of quantization levels in the digitization process; the more levels, the smaller the quantization noise. The theoretical signal to (noise + distortion) ratio for an ideal N-bit converter with a sine wave input is given by:

$$\textit{Signal to(Noise + Distortion)} = (6.02N + 1.76)\ dB$$

Thus for a 12-bit converter, this is 74 dB.

Total Harmonic Distortion
Total Harmonic Distortion is the ratio of the rms sum of the harmonics to the fundamental.

DAC SPECIFICATIONS

Relative Accuracy
Relative accuracy or endpoint linearity is a measure of the maximum deviation from a straight line passing through the endpoints of the DAC transfer function. It is measured after adjusting for zero error and full-scale error.

Voltage Output Settling Time
This is the amount of time it takes for the output to settle to a specified level for a full-scale input change.

Digital-to-Analog Glitch Impulse
This is the amount of charge injected into the analog output when the inputs change state. It is specified as the area of the glitch in nV sec.

SPECIAL FUNCTION REGISTERS

All registers except the program counter and the four general-purpose register banks reside in the special function register (SFR) area. The SFR registers include control, configuration, and data registers that provide an interface between the CPU and other on-chip peripherals.

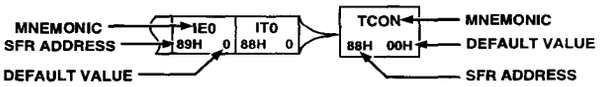
Figure 6 shows a full SFR memory map and SFR contents on Reset. Unoccupied SFR locations are shown dark-shaded in the figure below (NOT USED). Unoccupied locations in the SFR address space are not implemented i.e., no register exists at this location. If an unoccupied location is read, an unspecified value is returned. SFR locations reserved for on-chip testing are shown lighter shaded below (RESERVED) and should not be accessed by user software. Sixteen of the SFR locations are also bit addressable and denoted by '1' in the figure below, i.e., the bit addressable SFRs are those whose address ends in 0H or 8H.

ISPI FFH 0	WCOL FEH 0	SPE FDH 0	SPIM FCH 0	CPOL FBH 0	CPHA FAH 1	SPR1 F9H 0	SPR0 F8H 0	BITS
F7H 0	F6H 0	F5H 0	F4H 0	F3H 0	F2H 0	F1H 0	F0H 0	BITS
MDO EFH 0	MDE EEH 0	MCO EDH 0	MDI ECH 0	I2CM EBH 0	I2CRS EAH 0	I2CTX E9H 0	I2CI E8H 0	BITS
E7H 0	E6H 0	E5H 0	E4H 0	E3H 0	E2H 0	E1H 0	E0H 0	BITS
ADCI DFH 0	DMA DEH 0	CCONV DDH 0	SCONV DCH 0	CS3 DBH 0	CS2 DAH 0	CS1 D9H 0	CS0 D8H 0	BITS
CY D7H 0	AC D6H 0	F0 D5H 0	RS1 D4H 0	RS0 D3H 0	OV D2H 0	FI D1H 0	P D0H 0	BITS
TF2 CFH 0	EXF2 CEH 0	RCLK CDH 0	TCLK CCH 0	EXEN2 CBH 0	TR2 CAH 0	CNT2 C9H 0	CAP2 C8H 0	BITS
PRE3 C7H 0	PRE2 C6H 0	PRE1 C5H 0	PRE0 C4H 1	WDIR C3H 0	WDS C2H 0	WDE C1H 0	WDWR C0H 0	BITS
PSI BFH 0	PADC BEH 0	PT2 BDH 0	PS BCH 0	PT1 BBH 0	PX1 BAH 0	PT0 B9H 0	PX0 B8H 0	BITS
RD B7H 1	WR B6H 1	T1 B5H 1	T0 B4H 1	INT1 B3H 1	INT0 B2H 1	TxD B1H 1	RxD B0H 1	BITS
EA AFH 0	EADC AEH 0	ET2 ADH 0	ES ACH 0	ET1 ABH 0	EX1 AAH 0	ET0 A9H 0	EX0 A8H 0	BITS
A7H 1	A6H 1	A5H 1	A4H 1	A3H 1	A2H 1	A1H 1	A0H 1	BITS
SM0 9FH 0	SM1 9EH 0	SM2 9DH 0	REN 9CH 0	TB8 9BH 0	RB8 9AH 0	TI 99H 0	RI 98H 0	BITS
97H 1	96H 1	95H 1	94H 1	93H 1	92H 1	T2EX 91H 1	T2 90H 1	BITS
TF1 8FH 0	TR1 8EH 0	TF0 8DH 0	TR0 8CH 0	IE1 8BH 0	IT1 8AH 0	IE0 89H 0	IT0 88H 0	BITS
87H 1	86H 1	85H 1	84H 1	83H 1	82H 1	81H 1	80H 1	BITS

SPICON[1] F8H 04H	DAC0L F9H 00H	DAC0H FAH 00H	DAC1L FBH 00H	DAC1H FCH 00H	DACCON FDH 04H	RESERVED	RESERVED	
B[1] F0H 00H	ADCOFSL[3] F1H 00H	ADCOFSH[3] F2H 20H	ADCGAINL[3] F3H 00H	ADCGAINH[3] F4H 00H	ADCCON3 F5H 00H	RESERVED	SPIDAT F7H 00H	
I2CCON[1] E8H 00H	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	ADCCON1 EFH 00H	
ACC[1] E0H 00H	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	
ADCCON2[1] D8H 00H	ADCDATAL D9H 00H	ADCDATAH DAH 00H	RESERVED	RESERVED	RESERVED	RESERVED	PSMCON DFH DEH	
PSW[1] D0H 00H	RESERVED	DMAL D2H 00H	DMAH D3H 00H	DMAP D4H 00H	RESERVED	RESERVED	PLLCON D7H 53H	
T2CON[1] C8H 00H	RESERVED	RCAP2L CAH 00H	RCAP2H CBH 00H	TL2 CCH 00H	TH2 CDH 00H	RESERVED	RESERVED	
WDCON[1] C0H 10H	RESERVED	CHIPID C2H 2XH	RESERVED	RESERVED	RESERVED	EDARL C6H 00H	EDARH C7H 00H	
IP[1] B8H 00H	ECON B9H 00H	RESERVED	RESERVED	EDATA1 BCH 00H	EDATA2 BDH 00H	EDATA3 BEH 00H	EDATA4 BFH 00H	
P3[1] B0H FFH	PWM0L B1H 00H	PWM0H B2H 00H	PWM1L B3H 00H	PWM1H B4H 00H	RESERVED	RESERVED	SPH B7H 00H	
IE[1] A8H 00H	IEIP2 A9H A0H	RESERVED	RESERVED	RESERVED	RESERVED	PWMCON AEH 00H	CFG832 AFH 00H	
P2[1] A0H FFH	TIMECON A1H 00H	HTHSEC A2H 00H	SEC A3H 00H	MIN A4H 00H	HOUR A5H 00H	INTVAL A6H 00H	DPCON A7H 00H	
SCON[1] 98H 00H	SBUF 99H 00H	I2CDAT 9AH 00H	I2CADD 9BH 55H	NOT USED	T3FD 9DH 00H	T3CON 9EH 00H		
P1[1,2] 90H FFH	NOT USED	NOT USED	NOT USED	NOT USED				
TCON[1] 88H 00H	TMOD 89H 00H	TL0 8AH 00H	TL1 8BH 00H	TH0 8CH 00H	TH1 8DH 00H			
P0[1] 80H FFH	SP 81H 07H	DPL 82H 00H	DPH 83H 00H	DPP 84H 00H	RESERVED	RESERVED	PCON 87H 00H	

SFR MAP KEY:

THESE BITS ARE CONTAINED IN THIS BYTE.



NOTES

[1]SFRs WHOSE ADDRESS ENDS IN 0H OR 8H ARE BIT ADDRESSABLE.
[2]THE PRIMARY FUNCTION OF PORT1 IS AS AN ANALOG INPUT PORT; THEREFORE, TO ENABLE THE DIGITAL SECONDARY FUNCTIONS ON THESE PORT PINS, WRITE A "0" TO THE CORRESPONDING PORT 1 SFR BIT.
[3]CALIBRATION COEFFICIENTS ARE PRECONFIGURED ON POWER-UP TO FACTORY CALIBRATED VALUES.

Figure 6. Special Function Register Locations and Reset Values

ADC CIRCUIT INFORMATION

General Overview

The ADC conversion block incorporates a fast, 8-channel, 12-bit, single-supply ADC. This block provides the user with multichannel mux, track/hold, on-chip reference, calibration features, and ADC. All components in this block are easily configured via a 3-register SFR interface.

The ADC converter consists of a conventional successive-approximation converter based around a capacitor DAC. The converter accepts an analog input range of 0 to V_{REF}. A high precision, low drift, and factory calibrated 2.5 V reference is provided on-chip. An external reference can be connected as described later. This external reference can be in the range 1 V to AV_{DD}.

Single step or continuous conversion modes can be initiated in software or alternatively by applying a convert signal to an external pin. Timer 2 can also be configured to generate a repetitive trigger for ADC conversions. The ADC may be configured to operate in a DMA mode whereby the ADC block continuously converts and captures samples to an external RAM space without any interaction from the MCU core. This automatic capture facility can extend through a 16 MByte external data memory space.

The ADuC832 is shipped with factory programmed calibration coefficients that are automatically downloaded to the ADC on power-up, ensuring optimum ADC performance. The ADC core contains internal offset and gain calibration registers that can be hardware calibrated to minimize system errors.

A voltage output from an on-chip band gap reference proportional to absolute temperature can also be routed through the front end ADC multiplexer (effectively a ninth ADC channel input) facilitating a temperature sensor implementation.

ADC Transfer Function

The analog input range for the ADC is 0 V to V_{REF}. For this range, the designed code transitions occur midway between successive integer LSB values (i.e., 1/2 LSB, 3/2 LSBs, 5/2 LSBs . . . FS –3/2 LSBs). The output coding is straight binary with 1 LSB = FS/4096 or 2.5 V/4096 = 0.61 mV when V_{REF} = 2.5 V. The ideal input/output transfer characteristic for the 0 to V_{REF} range is shown in Figure 7.

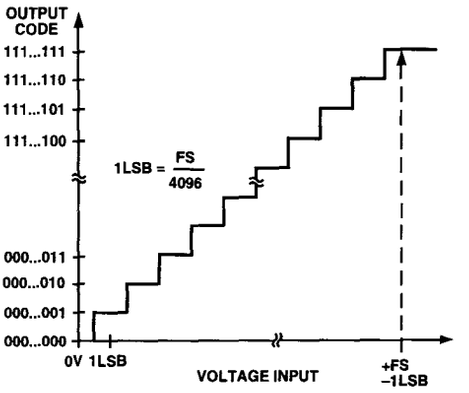


Figure 7. ADC Transfer Function

Typical Operation

Once configured via the ADCCON 1-3 SFRs, the ADC will convert the analog input and provide an ADC 12-bit result word in the ADCDATAH/L SFRs. The top four bits of the ADCDATAH SFR will be written with the channel selection bits so as to identify the channel result. The format of the ADC 12-bit result word is shown in Figure 8.

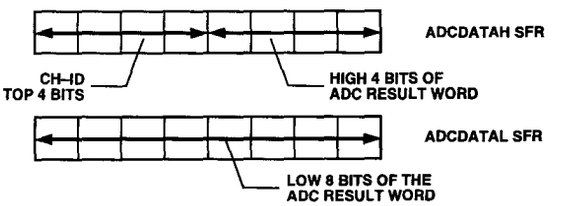


Figure 8. ADC Result Format

ADCCON1 – (ADC Control SFR #1)
The ADCCON1 register controls conversion and acquisition times, hardware conversion modes, and power-down modes as detailed below.

SFR Address: EFH
SFR Power-On Default Value: 00H
Bit Addressable: NO

Table III. ADCCON1 SFR Bit Designations

Bit	Name	Description
ADCCON1.7	MD1	The Mode bit selects the active operating mode of the ADC. Set by the user to power up the ADC. Cleared by the user to power down the ADC.
ADCCON1.6	EXT_REF	Set by the user to select an external reference. Cleared by the user to use the internal reference.
ADCCON1.5 ADCCON1.4	CK1 CK0	The ADC clock divide bits (CK1, CK0) select the divide ratio for the PLL master clock used to generate the ADC clock. To ensure correct ADC operation, the divider ratio must be chosen to reduce the ADC clock to 4.5 MHz and below. A typical ADC conversion will require 17 ADC clocks. The divider ratio is selected as follows: **CK1 CK0 MCLK Divider** 0 0 8 0 1 4 1 0 16 1 1 32
ADCCON1.3 ADCCON1.2	AQ1 AQ0	The ADC acquisition select bits (AQ1, AQ0) select the time provided for the input track-and-hold amplifier to acquire the input signal. An acquisition of three or more ADC clocks is recommended; clocks are selected as follows: **AQ1 AQ0 #ADC Clks** 0 0 1 0 1 2 1 0 3 1 1 4
ADCCON1.1	T2C	The Timer 2 conversion bit (T2C) is set by the user to enable the Timer 2 overflow bit be used as the ADC convert start trigger input.
ADCCON1.0	EXC	The external trigger enable bit (EXC) is set by the user to allow the external Pin P3.5 (CONVST) to be used as the active low convert start input. This input should be an active low pulse (minimum pulsewidth >100 ns) at the required sample rate.

ADCCON2 – (ADC Control SFR #2)
The ADCCON2 register controls ADC channel selection and conversion modes as detailed below.

SFR Address: D8H
SFR Power-On Default Value: 00H
Bit Addressable: YES

Table IV. ADCCON2 SFR Bit Designations

Bit	Name	Description
ADCCON2.7	ADCI	The ADC interrupt bit (ADCI) is set by hardware at the end of a single ADC conversion cycle or at the end of a DMA block conversion. ADCI is cleared by hardware when the PC vectors to the ADC Interrupt Service Routine. Otherwise, the ADCI bit should be cleared by user code.
ADCCON2.6	DMA	The DMA mode enable bit (DMA) is set by the user to enable a preconfigured ADC DMA mode operation. A more detailed description of this mode is given in the ADC DMA Mode section. The DMA bit is automatically set to "0" at the end of a DMA cycle. Setting this bit causes the ALE output to cease, it will start again when DMA is started and will operate correctly after DMA is complete.
ADCCON2.5	CCONV	The continuous conversion bit (CCONV) is set by the user to initiate the ADC into a continuous mode of conversion. In this mode, the ADC starts converting based on the timing and channel configuration already set up in the ADCCON SFRs; the ADC automatically starts another conversion once a previous conversion has completed.
ADCCON2.4	SCONV	The single conversion bit (SCONV) is set to initiate a single conversion cycle. The SCONV bit is automatically reset to "0" on completion of the single conversion cycle.
ADCCON2.3 ADCCON2.2 ADCCON2.1 ADCCON2.0	CS3 CS2 CS1 CS0	The channel selection bits (CS3–0) allow the user to program the ADC channel selection under software control. When a conversion is initiated, the channel converted will be that pointed to by these channel selection bits. In DMA mode, the channel selection is derived from the channel ID written to the external memory.

CS3	CS2	CS1	CS0	CH#	
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	2	
0	0	1	1	3	
0	1	0	0	4	
0	1	0	1	5	
0	1	1	0	6	
0	1	1	1	7	
1	0	0	0	Temp Monitor	Requires minimum of 1 µs to acquire
1	0	0	1	DAC0	Only use with Internal DAC o/p buffer on
1	0	1	0	DAC1	Only use with Internal DAC o/p buffer on
1	0	1	1	AGND	
1	1	0	0	VREF	
1	1	1	1	DMA STOP	Place in XRAM location to finish DMA sequence, see the section ADC DMA Mode.

All other combinations reserved

Timers/Counters

The ADuC832 has three 16-bit Timer/Counters: Timer 0, Timer 1, and Timer 2. The Timer/Counter hardware has been included on-chip to relieve the processor core of the overhead inherent in implementing Timer/Counter functionality in software. Each Timer/Counter consists of two 8-bit registers THx and TLx (x = 0, 1 and 2). All three can be configured to operate either as timers or event counters.

In Timer function, the TLx register is incremented every machine cycle. Thus, one can think of it as counting machine cycles. Since a machine cycle consists of 12 core clock periods, the maximum count rate is 1/12 the core clock frequency.

In Counter function, the TLx register is incremented by a 1-to-0 transition at its corresponding external input pin, T0, T1, or T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since it takes two machine cycles (24 core clock periods) to recognize a 1-to-0 transition, the maximum count rate is 1/24 the core clock frequency. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it must be held for a minimum of one full machine cycle.

User configuration and control of all Timer operating modes is achieved via three SFRs:

TMOD, TCON	Control and configuration for Timers 0 and 1.
T2CON	Control and configuration for Timer 2.

TMOD **Timer/Counter 0 and 1 Mode Register**

SFR Address	89H
Power-On Default Value	00H
Bit Addressable	No

Table XX. TMOD SFR Bit Designations

Bit	Name	Description
7	Gate	Timer 1 Gating Control. Set by software to enable Timer/Counter 1 only while $\overline{INT1}$ pin is high and TR1 control bit is set. Cleared by software to enable Timer 1 whenever TR1 control bit is set.
6	C/T	Timer 1 Timer or Counter Select Bit. Set by software to select counter operation (input from T1 pin). Cleared by software to select timer operation (input from internal system clock).
5	M1	Timer 1 Mode Select Bit 1 (Used with M0 Bit).
4	M0	Timer 1 Mode Select Bit 0. M1 M0 0 0 TH1 operates as an 8-bit timer/counter. TL1 serves as 5-bit prescaler. 0 1 16-Bit Timer/Counter. TH1 and TL1 are cascaded; there is no prescaler. 1 0 8-Bit Auto-Reload Timer/Counter. TH1 holds a value that is to be reloaded into TL1 each time it overflows. 1 1 Timer/Counter 1 Stopped.
3	Gate	Timer 0 Gating Control. Set by software to enable timer/counter 0 only while $\overline{INT0}$ pin is high and TR0 control bit is set. Cleared by software to enable Timer 0 whenever TR0 control bit is set.
2	C/T	Timer 0 Timer or Counter Select Bit. Set by software to select counter operation (input from T0 pin). Cleared by software to select timer operation (input from internal system clock).
1	M1	Timer 0 Mode Select Bit 1.
0	M0	Timer 0 Mode Select Bit 0. M1 M0 0 0 TH0 operates as an 8-bit timer/counter. TL0 serves as a 5-bit prescaler. 0 1 16-Bit Timer/Counter. TH0 and TL0 are cascaded; there is no prescaler. 1 0 8-Bit Auto-Reload Timer/Counter. TH0 holds a value that is to be reloaded into TL0 each time it overflows. 1 1 TL0 is an 8-bit timer/counter controlled by the standard timer 0 control bits. TH0 is an 8-bit timer only, controlled by Timer 1 control bits.

T2CON	**Timer/Counter 2 Control Register**
SFR Address	C8H
Power-On Default Value	00H
Bit Addressable	Yes

Table XXII. T2CON SFR Bit Designations

Bit	Name	Description
7	TF2	Timer 2 Overflow Flag. Set by hardware on a Timer 2 overflow. TF2 will not be set when either RCLK = 1 or TCLK = 1. Cleared by user software.
6	EXF2	Timer 2 External Flag. Set by hardware when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. Cleared by user software.
5	RCLK	Receive Clock Enable Bit. Set by the user to enable the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. Cleared by the user to enable Timer 1 overflow to be used for the receive clock.
4	TCLK	Transmit Clock Enable Bit. Set by the user to enable the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. Cleared by the user to enable Timer 1 overflow to be used for the transmit clock.
3	EXEN2	Timer 2 External Enable Flag. Set by the user to enable a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. Cleared by the user for Timer 2 to ignore events at T2EX.
2	TR2	Timer 2 Start/Stop Control Bit. Set by the user to start Timer 2. Cleared by the user to stop Timer 2.
1	CNT2	Timer 2 Timer or Counter Function Select Bit. Set by the user to select counter function (input from external T2 pin). Cleared by the user to select timer function (input from on-chip core clock).
0	CAP2	Timer 2 Capture/Reload Select Bit. Set by the user to enable captures on negative transitions at T2EX if EXEN2 = 1. Cleared by the user to enable autoreloads with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to autoreload on Timer 2 overflow.

Timer/Counter 2 Data Registers

Timer/Counter 2 also has two pairs of 8-bit data registers associated with it. These are used as both timer data registers and timer capture/reload registers.

TH2 and TL2

Timer 2, data high byte and low byte.
SFR Address = CDH, CCH respectively.

RCAP2H and RCAP2L

Timer 2, Capture/Reload byte and low byte.
SFR Address = CBH, CAH respectively.

Timer/Counter Operation Modes

The following paragraphs describe the operating modes for Timer/Counter 2. The operating modes are selected by bits in the T2CON SFR as shown in Table XXIII.

Table XXIII. T2CON Operating Modes

RCLK (or) TCLK	CAP2	TR2	Mode
0	0	1	16-Bit Autoreload
0	1	1	16-Bit Capture
1	X	1	Baud Rate
X	X	0	OFF

16-Bit Autoreload Mode

In Autoreload mode, there are two options, which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then when Timer 2 rolls over it not only sets TF2 but also causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2L and RCAP2H, which are preset by software. If EXEN2 = 1, then Timer 2 still performs the above, but with the added feature that a 1-to-0 transition at external input T2EX will also trigger the 16-bit reload and set EXF2. The Autoreload mode is illustrated in Figure 49.

16-Bit Capture Mode

In the Capture mode, there are again two options, which are selected by bit EXEN2 in T2CON. If EXEN2 = 0, then Timer 2 is a 16-bit timer or counter that, upon overflowing, sets bit TF2, the Timer 2 overflow bit, which can be used to generate an interrupt. If EXEN2 = 1, then Timer 2 still performs the above, but a l-to-0 transition on external input T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set, and EXF2, like TF2, can generate an interrupt. The Capture mode is illustrated in Figure 50.

The baud rate generator mode is selected by RCLK = 1 and/or TCLK = 1.

In either case, if Timer 2 is being used to generate the baud rate, the TF2 interrupt flag will not occur. Therefore, Timer 2 interrupts will not occur so they do not have to be disabled. In this mode the EXF2 flag, however, can still cause interrupts and this can be used as a third external interrupt.

Baud rate generation will be described as part of the UART serial port operation in the following pages.

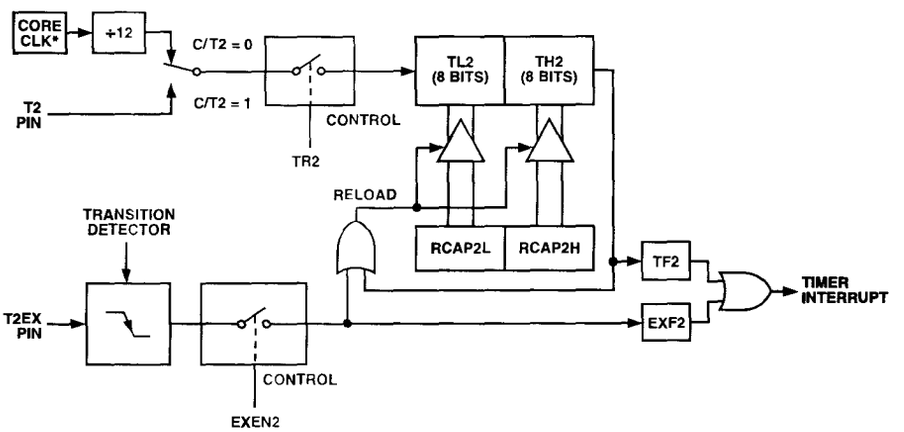


Figure 49. Timer/Counter 2, 16-Bit Autoreload Mode

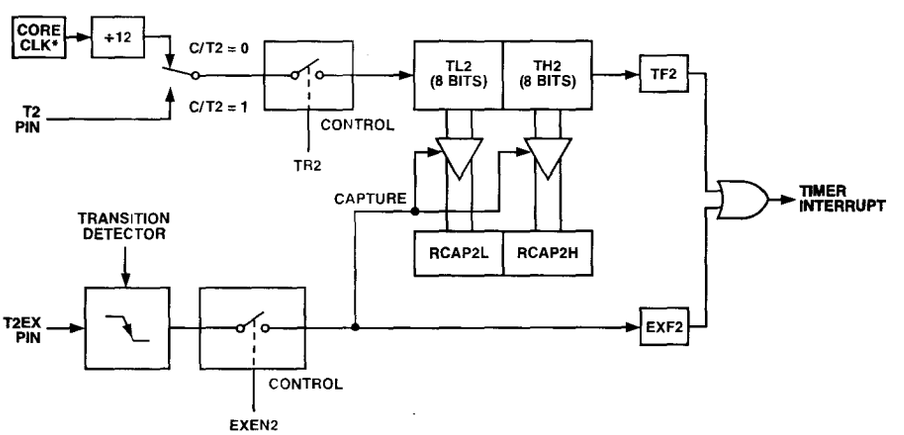


Figure 50. Timer/Counter 2, 16-Bit Capture Mode

Timer 1 Generated Baud Rates

When Timer 1 is used as the baud rate generator, the baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD as follows:

$$\textit{Modes 1 and 3 Baud Rate} = (2^{SMOD}/32) \times (\textit{Timer 1 Overflow Rate})$$

The Timer 1 interrupt should be disabled in this application. The timer itself can be configured for either timer or counter operation, and in any of its three running modes. In the most typical application, it is configured for timer operation in the Autoreload mode (high nibble of TMOD = 0010 binary). In that case, the baud rate is given by the formula:

$$\textit{Modes 1 and 3 Baud Rate} = (2^{SMOD}/32) \times (\textit{Core Clock}/(12 \times [256 - TH1]))$$

Table XXV shows some commonly used baud rates and how they might be calculated from a core clock frequency of 16.78 MHz and 2.0971 MHz. Generally speaking, a 5% error is tolerable using asynchronous (start/stop) communications.

Table XXV. Commonly Used Baud Rates, Timer 1

Ideal Baud	Core CLK (MHz)	SMOD Value	TH1-Reload Value	Actual Baud	% Error
9600	16.78	1	–9 (F9H)	9709	1.14
2400	16.78	1	–36 (DCH)	2427	1.14
1200	16.78	1	–73 (B7H)	1197	0.25
1200	2.10	0	–9 (F4H)	1213	1.14

Timer 2 Generated Baud Rates

Baud rates can also be generated using Timer 2. Using Timer 2 is similar to using Timer 1 in that the timer must overflow 16 times before a bit is transmitted/received. Because Timer 2 has a 16-bit Autoreload mode, a wider range of baud rates is possible using Timer 2.

$$\textit{Modes 1 and 3 Baud Rate} = (1/16) \times (\textit{Timer 2 Overflow Rate})$$

Therefore, when Timer 2 is used to generate baud rates, the timer increments every two clock cycles and not every core machine cycle as before. Thus, it increments six times faster than Timer 1, and therefore baud rates six times faster are possible. Because Timer 2 has 16-bit autoreload capability, very low baud rates are still possible.

Timer 2 is selected as the baud rate generator by setting the TCLK and/or RCLK in T2CON. The baud rates for transmit and receive can be simultaneously different. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode as shown in Figure 53.

In this case, the baud rate is given by the formula:

$$\textit{Modes 1 and 3 Baud Rate} = (\textit{Core Clk})/(32 \times [65536 - (RCAP2H, RCAP2L)])$$

Table XXVI shows some commonly used baud rates and how they might be calculated from a core clock frequency of 16.78 MHz and 2.10 MHz.

Table XXVI. Commonly Used Baud Rates, Timer 2

Ideal Baud	Core CLK (MHz)	RCAP2H Value	RCAP2L Value	Actual Baud	% Error
19200	16.78	–1 (FFH)	–27 (E5H)	19418	1.14
9600	16.78	–1 (FFH)	–55 (C9H)	9532	0.7
2400	16.78	–1 (FFH)	–218 (26H)	2405	0.21
1200	16.78	–2 (FEH)	–181 (4BH)	1199	0.02
9600	2.10	–1 (FFH)	–7 (FBH)	9362	2.4
2400	2.10	–1 (FFH)	–27 (ECH)	2427	1.14
1200	2.10	–1 (FFH)	–55 (C9H)	1191	0.7

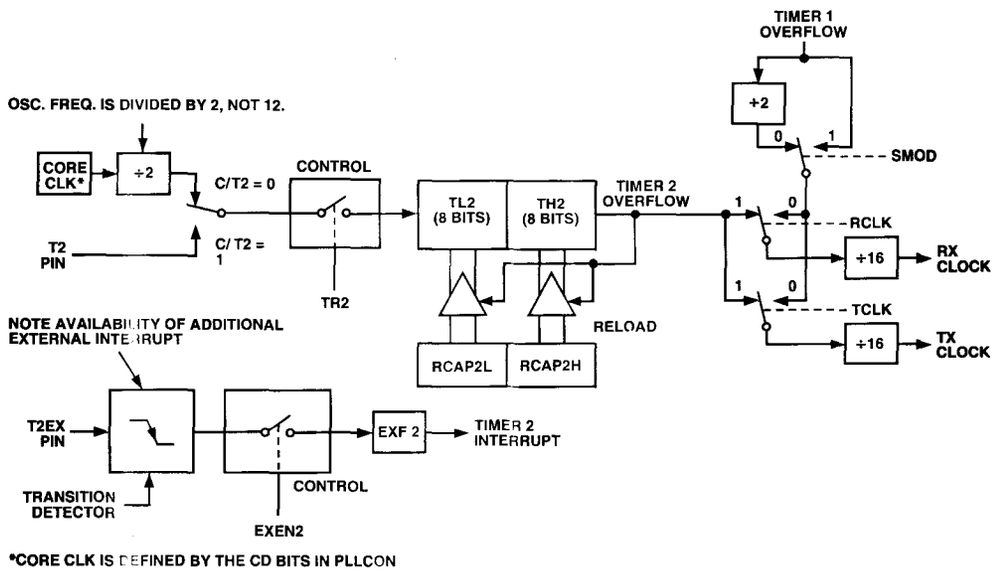


Figure 53. Timer 2, UART Baud Rates

Timer 3 Generated Baud Rates

The high integer dividers in a UART block mean that high speed baud rates are not always possible using some particular crystals. For example, using a 12 MHz crystal, a baud rate of 115200 is not possible. To address this problem, the ADuC832 has added a dedicated baud rate timer (Timer 3) specifically for generating highly accurate baud rates.

Timer 3 can be used instead of Timer 1 or Timer 2 for generating very accurate high speed UART baud rates including 115200 and 230400. Timer 3 also allows a much wider range of baud rates to be obtained. In fact, every desired bit rate from 12 bit/s to 393216 bit/s can be generated to within an error of ±0.8%. Timer 3 also frees up the other three timers, allowing them to be used for different applications. A block diagram of Timer 3 is shown in Figure 54.

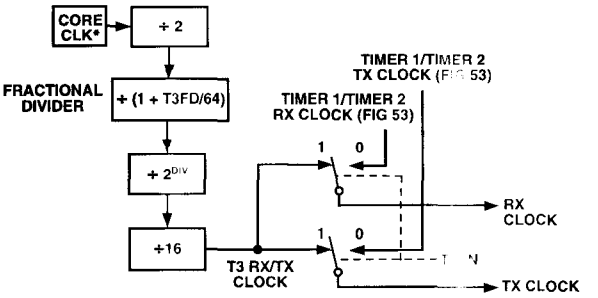


Figure 54. Timer 3, UART Baud Rates

Two SFRs (T3CON and T3FD) are used to control Timer 3. T3CON is the baud rate control SFR, allowing Timer 3 to be used to set up the UART baud rate, and setting up the binary divider (DIV).

Table XXVII. T3CON SFR Bit Designations

Bit	Name	Description
7	T3BAUDEN	T3UARTBAUD Enable Set to enable Timer 3 to generate the baud rate. When set PCON.7, T2CON.4 and T2CON.5 are ignored. Cleared to let the baud rate be generated as per a standard 8052.
6		-
5		-
4		-
3		-
2	DIV2	Binary Divider Factor
1	DIV1	DIV2 DIV1 DIV0 Bin Divider 0 0 0 1 0 0 1 1 0 1 0 1 0 1 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 1 1
0	DIV0	

The appropriate value to write to the DIV2-1-0 bits can be calculated using the following formula where f_{CORE} defined in PLLCON SFR:

Note: The DIV value must be rounded down.

$$DIV = \frac{\log\left(\frac{f_{CORE}}{32 \times Baud\ Rate}\right)}{\log(2)}$$

T3FD is the fractional divider ratio required to achieve the required baud rate. We can calculate the appropriate value for T3FD using the following formula:

Note: T3FD should be rounded to the nearest integer.

$$T3FD = \frac{2 \times f_{CORE}}{2^{DIV} \times Baud\ Rate}$$

Once the values for DIV and T3FD are calculated the actual baud rate can be calculated using the following formula:

$$Actual\ Baud\ Rate = \frac{2 \times f_{CORE}}{2^{DIV} \times (T3FD + 64)}$$

For example, to get a baud rate of 115200 while operating at 16.7 MHz

$$DIV = LOG\left(11059200 / \left(32 \times 115200\right)\right) / LOG2 = 1.58 = 1$$

$$T3FD = \left(2 \times 11059200\right) / \left(2^1 \times 115200\right) - 64 = 32 = 20H$$

therefore, the actual baud rate is 114912 bit/s.

Table XXVIII. Commonly Used Baud Rates Using Timer 3

Ideal Baud	CD	DIV	T3CON	T3FD	% Error
230400	0	1	81H	09H	0.25
115200	0	2	82H	09H	0.25
115200	1	1	81H	09H	0.25
115200	2	0	80H	09H	0.25
57600	0	3	83H	09H	0.25
57600	1	2	82H	09H	0.25
57600	2	1	81H	09H	0.25
57600	3	0	80H	09H	0.25
38400	0	3	83H	2DH	0.2
38400	1	2	82H	2DH	0.2
38400	2	1	81H	2DH	0.2
38400	3	0	80H	2DH	0.2
19200	0	4	84H	2DH	0.2
19200	1	3	83H	2DH	0.2
19200	2	2	82H	2DH	0.2
19200	3	1	81H	2DH	0.2
19200	4	0	80H	2DH	0.2
9600	0	5	85H	2DH	0.2
9600	1	4	84H	2DH	0.2
9600	2	3	83H	2DH	0.2
9600	3	2	82H	2DH	0.2
9600	4	1	81H	2DH	0.2
9600	5	0	80H	2DH	0.2

19-4323; Rev 10; 8/01

MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where ±12V is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers

Low-Power Modems

Interface Translation

Battery-Powered RS-232 Systems

Multidrop RS-232 Networks

Features

Superior to Bipolar

- **Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)**
- **Low-Power Receive Mode in Shutdown (MAX223/MAX242)**
- **Meet All EIA/TIA-232E and V.28 Specifications**
- **Multiple Drivers and Receivers**
- **3-State Driver and Receiver Outputs**
- **Open-Line Detection (MAX243)**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.

**Contact factory for dice specifications.*

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	0.1	No	—	120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✔	120	MAX241 and receivers active in shutdown
MAX225	+5	5/5	0	—	Yes	✔	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No	—	120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	—	No	—	120	No external caps
MAX233A	+5	2/2	0	—	No	—	200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	—	120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	—	Yes	—	120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	—	120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	—	120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	—	120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	—	120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	✔	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No	—	200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No	—	120	High slew rate
MAX245	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	—	Yes	✔	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	✔	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	1.0	Yes	✔	120	Available in quad flatpack package

ABSOLUTE MAXIMUM RATINGS—MAX223/MAX230–MAX241

V_{CC} .....-0.3V to +6V
V+ .....(V_{CC} - 0.3V) to +14V
V- .....+0.3V to -14V
Input Voltages
T_{IN} .....-0.3V to (V_{CC} + 0.3V)
R_{IN} .....±30V
Output Voltages
T_{OUT} .....(V+ + 0.3V) to (V- - 0.3V)
R_{OUT} .....-0.3V to (V_{CC} + 0.3V)
Short-Circuit Duration, T_{OUT} .....Continuous
Continuous Power Dissipation (T_A = +70°C)
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C)....800mW
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C)....842mW
20-Pin Plastic DIP (derate 11.11mW/°C above +70°C)....889mW
24-Pin Narrow Plastic DIP
(derate 13.33mW/°C above +70°C) 1.07W
24-Pin Plastic DIP (derate 9.09mW/°C above +70°C).....500mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C).....762mW
20-Pin Wide SO (derate 10 00mW/°C above +70°C).....800mW
24-Pin Wide SO (derate 11.76mW/°C above +70°C).....941mW
28-Pin Wide SO (derate 12.50mW/°C above +70°C) 1W
44-Pin Plastic FP (derate 11.11mW/°C above +70°C) 889mW
14-Pin CERDIP (derate 9.09mW/°C above +70°C).....727mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C).....800mW
20-Pin CERDIP (derate 11.11mW/°C above +70°C).....889mW
24-Pin Narrow CERDIP
(derate 12.50mW/°C above +70°C) 1W
24-Pin Sidebraze (derate 20.0mW/°C above +70°C).....1.6W
28-Pin SSOP (derate 9.52mW/°C above +70°C).....762mW
Operating Temperature Ranges
MAX2 _ _ C _ _ .....0°C to +70°C
MAX2 _ _ E _ _ .....-40°C to +85°C
MAX2 _ _ M _ _ .....-55°C to +125°C
Storage Temperature Range .....-65°C to +160°C
Lead Temperature (soldering, 10sec) .....+300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241

(MAX223/230/232/234/236/237/238/240/241, V_{CC} = +5V ±10; MAX233/MAX235, V_{CC} = 5V ±5%, C1–C4 = 1.0µF; MAX231/MAX239, V_{CC} = 5V ±10%; V+ = 7.5V to 13.2V; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

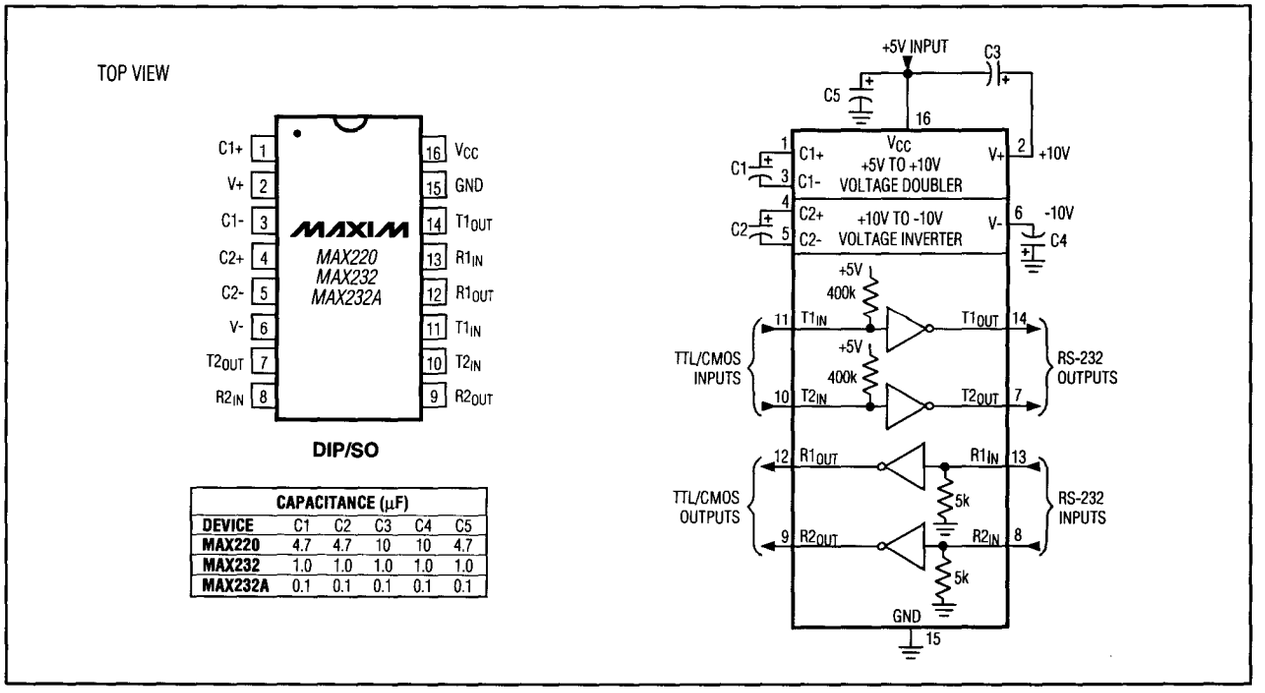
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to ground		±5.0	±7.3		V
V_{CC} Power-Supply Current	No load, T_A = +25°C	MAX232/233		5	10	mA
		MAX223/230/234–238/240/241		7	15	
		MAX231/239		0.4	1	
V+ Power-Supply Current		MAX231		1.8	5	mA
		MAX239		5	15	
Shutdown Supply Current	T_A = +25°C	MAX223		15	50	µA
		MAX230/235/236/240/241		1	10	
Input Logic Threshold Low	T_{IN}; EN, $\overline{SHDN}$ (MAX233); $\overline{EN}$, SHDN (MAX230/235–241)				0.8	V
Input Logic Threshold High	T_{IN}		2.0			V
	EN, $\overline{SHDN}$ (MAX223); $\overline{EN}$, SHDN (MAX230/235/236/240/241)		2.4			
Logic Pull-Up Current	T_{IN} = 0V			1.5	200	µA
Receiver Input Voltage Operating Range			-30		30	V

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ELECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241 (continued)

(MAX223/230/232/234/236/237/238/240/241, V_{CC} = +5V ±10; MAX233/MAX235, V_{CC} = 5V ±5%, C1–C4 = 1.0µF; MAX231/MAX239, V_{CC} = 5V ±10%; V+ = 7.5V to 13.2V; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 Input Threshold Low	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)	0.8	1.2		V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)	0.6	1.5		
RS-232 Input Threshold High	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)		1.7	2.4	V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)		1.5	2.4	
RS-232 Input Hysteresis	V_{CC} = 5V, no hysteresis in shutdown		0.2	0.5	1.0	V
RS-232 Input Resistance	T_A = +25°C, V_{CC} = 5V		3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 1.6mA (MAX231/232/233, I_{OUT} = 3.2mA)				0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1mA		3.5	V_{CC} - 0.4		V
TTL/CMOS Output Leakage Current	0V ≤ R_{OUT} ≤ V_{CC}; EN = 0V (MAX223); $\overline{EN}$ = V_{CC} (MAX235–241)			0.05	±10	µA
Receiver Output Enable Time	Normal operation	MAX223		600		ns
		MAX235/236/239/240/241		400		
Receiver Output Disable Time	Normal operation	MAX223		900		ns
		MAX235/236/239/240/241		250		
Propagation Delay	RS-232 IN to TTL/CMOS OUT, C_L = 150pF	Normal operation		0.5	10	µs
		$\overline{SHDN}$ = 0V (MAX223) t_{PHLS}		4	40	
		$\overline{SHDN}$ = 0V (MAX223) t_{PLHS}		6	40	
Transition Region Slew Rate	MAX223/MAX230/MAX234–241, T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V		3	5.1	30	V/µs
	MAX231/MAX232/MAX233, T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V			4	30	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V		300			Ω
Transmitter Output Short-Circuit Current				±10		mA



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

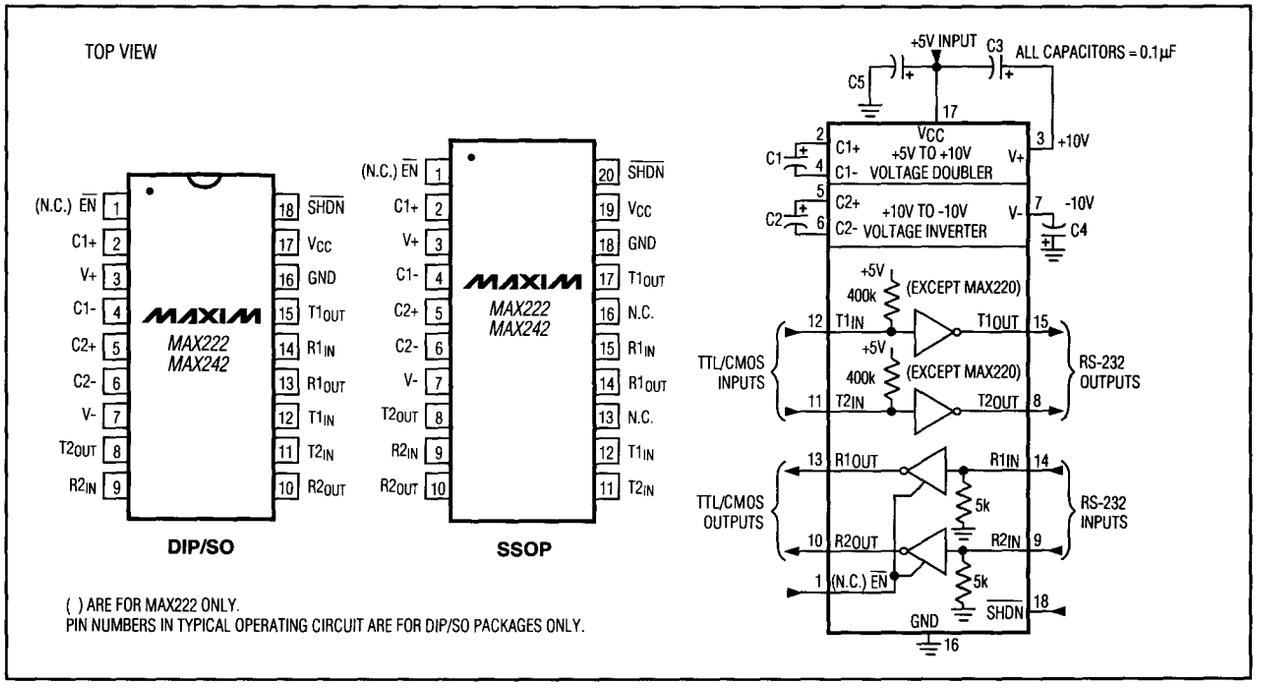
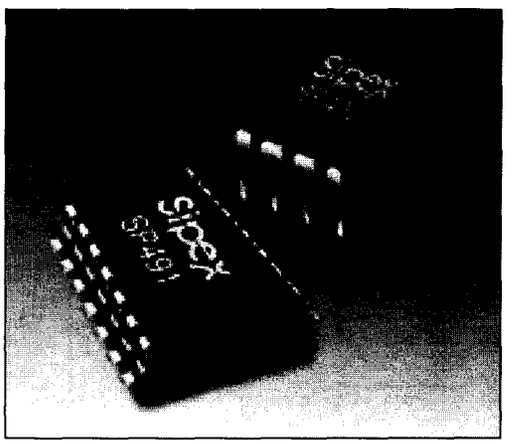


Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit

SP490/SP491

Full Duplex RS-485 Transceivers

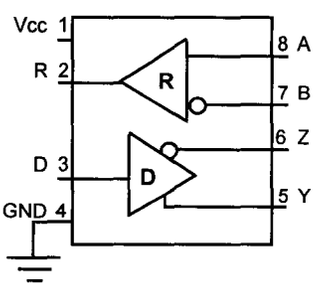
- +5V Only
- Low Power BiCMOS
- Driver/Receiver Enable (**SP491**)
- RS-485 and RS-422 Drivers/Receivers
- Pin Compatible with LTC490 and SN75179 (**SP490**)
- Pin Compatible with LTC491 and SN75180 (**SP491**)



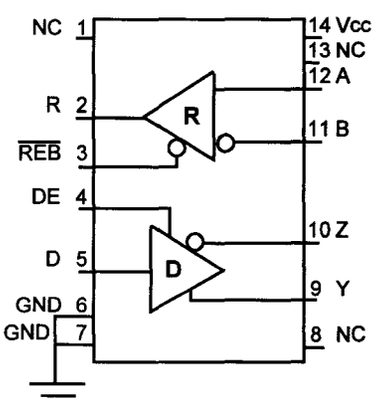
DESCRIPTION

The **SP490** is a low power differential line driver/receiver meeting RS-485 and RS-422 standards up to 5Mbps. The **SP491** is identical to the **SP490** with the addition of driver and receiver tri-state enable lines. Both products feature ±200mV receiver input sensitivity, over wide common mode range. The **SP490** is available in 8-pin plastic DIP and 8-pin NSOIC packages for operation over the commercial and industrial temperature ranges. The **SP491** is available in 14-pin DIP and 14-pin NSOIC packages for operation over the commercial and industrial temperature ranges.

BLOCK DIAGRAMS



SP490



SP491

ABSOLUTE MAXIMUM RATINGS

These are stress ratings only and functional operation of the device at these ratings or any other above those indicated in the operation sections of the specifications below is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

V_{CC} ..... +7V
Input Voltages
- Drivers ..... -0.5V to (V_{CC}+0.5V)
- Receivers ..... ±14V

Output Voltages
- Drivers ..... ±14V
- Receivers ..... -0.5V to (V_{CC}+0.5V)

Storage Temperature ..... -65°C to +150°
Power Dissipation 1000mW

ELECTRICAL CHARACTERISTICS

T_{MIN} to T_{MAX} and V_{CC} = 5V ± 5% unless otherwise noted.

PARAMETERS	MIN.	TYP.	MAX.	UNITS	CONDITIONS
SP490 DRIVER					
DC Characteristics					
Differential Output Voltage	GND		V_{CC}	Volts	Unloaded; R = ∞ ; *see figure 1*
Differential Output Voltage	2		V_{CC}	Volts	With Load; R = 50Ω; (RS422); *see figure 1*
Differential Output Voltage	1.5		V_{CC}	Volts	With Load; R = 27Ω; (RS485); *see figure 1*
Change in Magnitude of Driver Differential Output Voltage for Complimentary States			0.2	Volts	R = 27Ω or R = 50Ω; *see figure 1*
Driver Common-Mode Output Voltage			3	Volts	R = 27Ω or R = 50Ω; *see figure 1*
Input High Voltage	2.0			Volts	Applies to D
Input Low Voltage			0.8	Volts	Applies to D
Input Current			±10	µA	Applies to D
Driver Short-Circuit Current					
V_{OUT} = HIGH	35		250	mA	-7V ≤ V_O ≤ +12V
V_{OUT} = LOW	35		250	mA	-7V ≤ V_O ≤ +12V
SP490 DRIVER					
AC Characteristics					
Maximum Data Rate	5			Mbps	
Driver Input to Output		30	60	ns	t_{PLH}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
Driver Input to Output		30	60	ns	t_{PHL}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
Driver Skew		5		ns	*see figures 3 and 6*, t_{SKEW} = | t_{DPLH} - t_{DPHL} |
Driver Rise or Fall Time		15	40	ns	From 10% to 90%; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
SP490 RECEIVER					
DC Characteristics					
Differential Input Threshold	0.2		+0.2	Volts	-7V ≤ V_{CM} ≤ 12V
Input Hysteresis		70		mV	V_{CM} = 0V
Output Voltage High	3.5			Volts	I_O = -4mA, V_{ID} = +200mV
Output Voltage Low			0.4	Volts	I_O = +4mA, V_{ID} = -200mV
Input Resistance	12	15		kΩ	-7V ≤ V_{CM} ≤ 12V
Input Current (A, B); V_{IN} = 12V			±1.0	mA	V_{IN} = 12V
Input Current (A, B); V_{IN} = -7V			-0.8	mA	V_{IN} = -7V
Short-Circuit Current			85	mA	0V ≤ V_O ≤ V_{CC}

ELECTRICAL CHARACTERISTICS

T_{MIN} to T_{MAX} and V_{CC} = 5V ± 5% unless otherwise noted.

PARAMETERS	MIN.	TYP.	MAX.	UNITS	CONDITIONS
SP490 RECEIVER					
AC Characteristics					
Maximum Data Rate	5			Mbps	
Receiver Input to Output		90	150	ns	t_{PLH}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
Receiver Input to Output		90	150	ns	t_{PHL}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
Diff. Receiver Skew $\lvert t_{PLH}-t_{PHL}\rvert$		13		ns	R_{DIFF} = 54Ω; C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
POWER REQUIREMENTS					
Supply Voltage	+4.75		+5.25	Volts	
Supply Current		900		µA	
ENVIRONMENTAL AND MECHANICAL					
Operating Temperature					
Commercial (_C_)	0		+70	°C	
Industrial (_E_)	-40		+85	°C	
Storage Temperature	-65		+150	°C	
Package					
Plastic DIP (_S_)					
NSOIC (_N)					

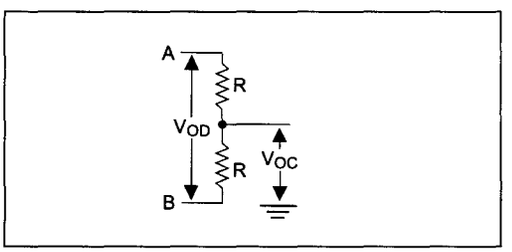


Figure 1. Driver DC Test Load Circuit

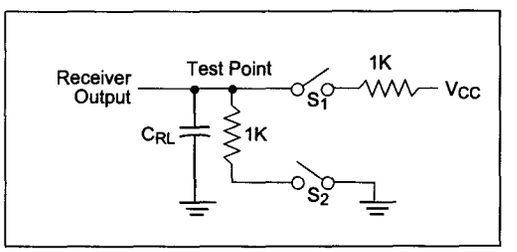


Figure 2. Receiver Timing Test Load Circuit

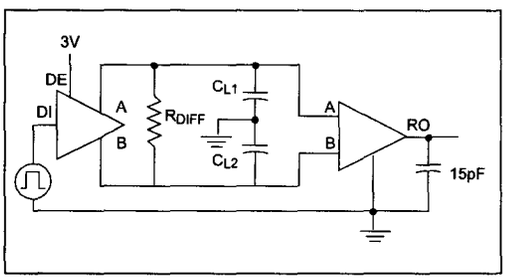


Figure 3. Driver/Receiver Timing Test Circuit

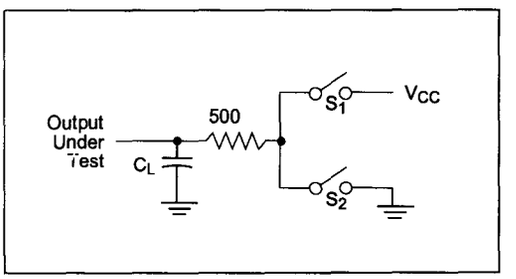


Figure 4. Driver Timing Test Load #2 Circuit

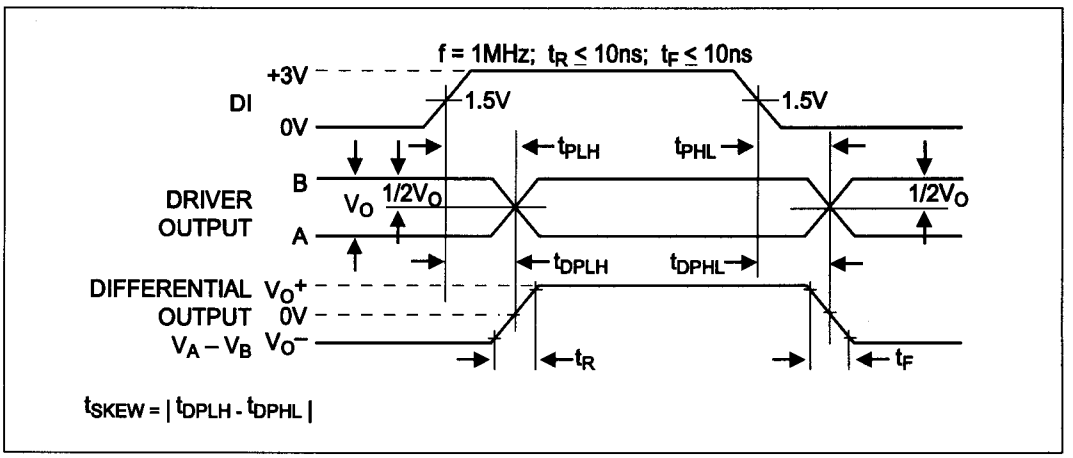


Figure 6. Driver Propagation Delays

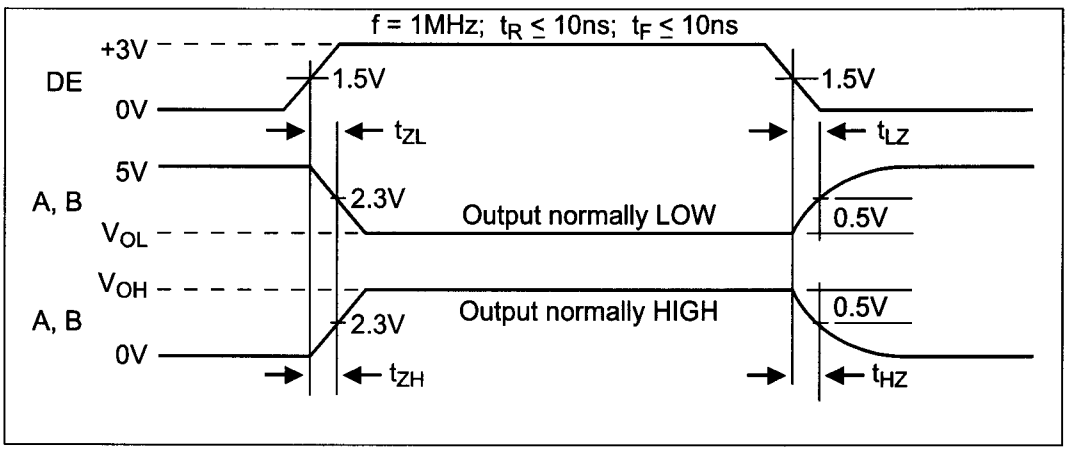


Figure 7. Driver Enable and Disable Times

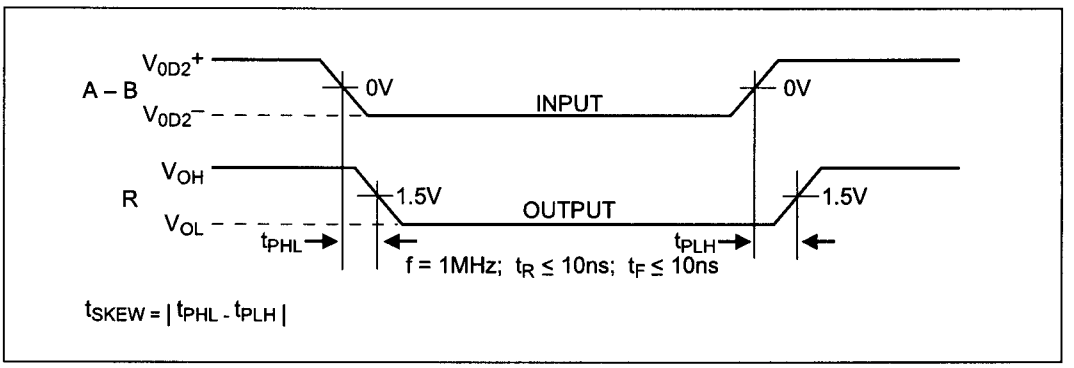


Figure 8. Receiver Propagation Delays

ABSOLUTE MAXIMUM RATINGS

These are stress ratings only and functional operation of the device at these ratings or any other above those indicated in the operation sections of the specifications below is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

V_{cc} ..... +7V
Input Voltages
- Logic ..... -0.5V to (V_{cc}+0.5V)
- Drivers ..... -0.5V to (V_{cc}+0.5V)
- Receivers ..... ±14V

Output Voltages
- Logic ..... -0.5V to (V_{cc}+0.5V)
- Drivers ..... ±14V
- Receivers ..... -0.5V to (V_{cc}+0.5V)

Storage Temperature ..... -65°C to +150
Power Dissipation 1000mW

ELECTRICAL CHARACTERISTICS

T_{MIN} to T_{MAX} and V_{cc} = 5V ± 5% unless otherwise noted.

PARAMETERS	MIN.	TYP.	MAX.	UNITS	CONDITIONS
SP491 DRIVER					
DC Characteristics					
Differential Output Voltage	GND		V_{CC}	Volts	Unloaded; R = ∞ ; *see figure 1*
Differential Output Voltage	2		V_{CC}	Volts	With Load; R = 50Ω; (RS422); *see figure 1*
Differential Output Voltage	1.5		V_{CC}	Volts	With Load; R = 27Ω; (RS485); *see figure 1*
Change in Magnitude of Driver Differential Output Voltage for Complimentary States			0.2	Volts	R = 27Ω or R = 50Ω; *see figure 1*
Driver Common-Mode Output Voltage			3	Volts	R = 27Ω or R = 50Ω; *see figure 1*
Input High Voltage	2.0			Volts	Applies to D, $\overline{REB}$, DE
Input Low Voltage			0.8	Volts	Applies to D, $\overline{REB}$, DE
Input Current			±10	µA	Applies to D, $\overline{REB}$, DE
Driver Short-Circuit Current					
V_{OUT} = HIGH	35		250	mA	-7V ≤ V_O ≤ 12V
V_{OUT} = LOW	35		250	mA	-7V ≤ V_O ≤ 12V
SP491 DRIVER					
AC Characteristics					
Maximum Data Rate	5			Mbps	$\overline{REB}$ = 5V, DE = 5V
Driver Input to Output	20	30	60	ns	t_{PLH}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
Driver Input to Output	20	30	60	ns	t_{PHL}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
Driver Skew		5	10	ns	*see figures 3 and 6,* t_{SKEW} = | t_{DPLH} - t_{DPHL} |
Driver Rise or Fall Time	3	15	40	ns	From 10% to 90%; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *see figures 3 and 6*
Driver Enable to Output High		40	70	ns	C_{L1} = C_{L2} = 100pF; *see figures 4 and 7;* S_2 closed
Driver Enable to Output Low		40	70	ns	C_{L1} = C_{L2} = 100pF; *see figures 4 and 7;* S_1 closed
Driver Disable Time from Low		40	70	ns	C_{L1} = C_{L2} = 100pF; *see figures 4 and 7;* S_1 closed
Driver Disable Time from High		40	70	ns	C_{L1} = C_{L2} = 100pF; *see figures 4 and 7;* S_2 closed

ELECTRICAL CHARACTERISTICS

T_{MIN} to T_{MAX} and V_{CC} = 5V ± 5% unless otherwise noted.

PARAMETERS	MIN.	TYP.	MAX.	UNITS	CONDITIONS
SP491 RECEIVER DC Characteristics					
Differential Input Threshold	-0.2		+0.2	Volts	-7V ≤ V_{CM} ≤ 12V
Input Hysteresis		70		mV	V_{CM} = 0V
Output Voltage High	3.5			Volts	I_O = -4mA, V_{ID} = +200mV
Output Voltage Low			0.4	Volts	I_O = +4mA, V_{ID} = -200mV
Three State (high impedance) Output Current			±1	µA	0.4V ≤ V_O ≤ 2.4V; $\overline{REB}$ = 5V
Input Resistance	12	15		kΩ	-7V ≤ V_{CM} ≤ 12V
Input Current (A, B); V_{IN} = 12V			±1.0	mA	DE = 0V, V_{CC} = 0V or 5.25V, V_{IN} = 12V
Input Current (A, B); V_{IN} = -7V			-0.8	mA	DE = 0V, V_{CC} = 0V or 5.25V, V_{IN} = -7V
Short-Circuit Current	7		85	mA	0V ≤ V_O ≤ V_{CC}
SP491 RECEIVER DC Characteristics					
Maximum Data Rate	5			Mbps	$\overline{REB}$ = 0V
Receiver Input to Output	60	90	150	ns	t_{PLH}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
Receiver Input to Output	60	90	150	ns	t_{PHL}; R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
Diff. Receiver Skew $\lvert t_{PLH}-t_{PHL} \rvert$		13		ns	R_{DIFF} = 54Ω; C_{L1} = C_{L2} = 100pF; *Figures 3 & 8*
Receiver Enable to Output Low		20	50	ns	C_{RL} = 15pF; *Figures 2 and 9;* S_1 closed
Receiver Enable to Output High		20	50	ns	C_{RL} = 15pF; *Figures 2 and 9;* S_2 closed
Receiver Disable from Low		20	50	ns	C_{RL} = 15pF; *Figures 2 and 9;* S_1 closed
Receiver Disable from High		20	50	ns	C_{RL} = 15pF; *Figures 2 and 9;* S_2 closed
POWER REQUIREMENTS					
Supply Voltage	+4.75		+5.25	Volts	
Supply Current		600		µA	$\overline{REB}$, D = 0V or V_{CC}; DE = V_{CC}
SP491 ENVIRONMENTAL AND MECHANICAL					
Operating Temperature					
Commercial (_C_)	0		+70	°C	
Industrial (_E_)	-40		+85	°C	
Storage Temperature	-65		+150	°C	
Package					
Plastic DIP (_S_)					
NSOIC (_N)					

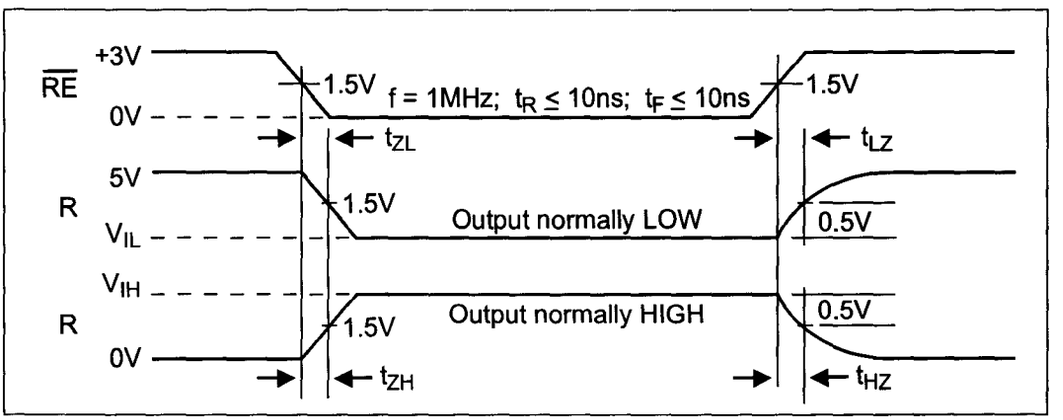


Figure 9. Receiver Enable and Disable Times

DESCRIPTION

The **SP490** and **SP491** are full-duplex differential transceivers that meet the requirements of RS-485 and RS-422. Fabricated with a **Sipex** proprietary BiCMOS process, both products require a fraction of the power of older bipolar designs.

The RS-485 standard is ideal for multi-drop applications or for long-distance interfaces. RS-485 allows up to 32 drivers and 32 receivers to be connected to a data bus, making it an ideal choice for multi-drop applications. Since the cabling can be as long as 4,000 feet, RS-485 transceivers are equipped with a wide (-7V to +12V) common mode range to accommodate ground potential differences. Because RS-485 is a differential interface, data is virtually immune to noise in the transmission line.

Driver...

The drivers for both the **SP490** and **SP491** have differential outputs. The typical voltage output swing with no load will be 0 volts to +5 volts. With worst case loading of 54Ω across the differential outputs, the driver can maintain greater than 1.5V voltage levels.

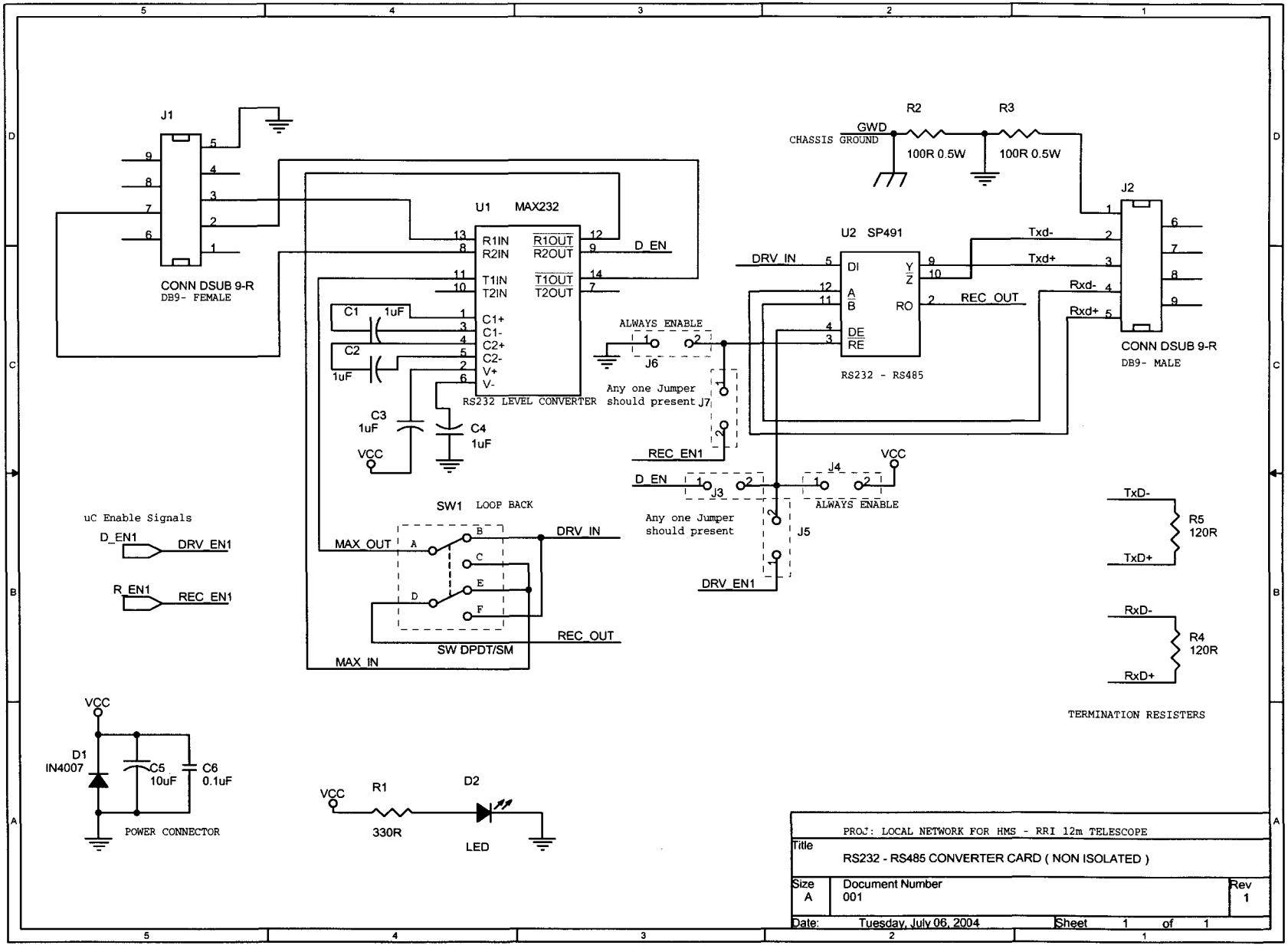
The driver of the **SP491** has a driver enable control line which is active high. A logic high on DE (pin 4) of the **SP491** will enable the differential driver outputs. A logic low on DE (pin 4) of the **SP491** will tri-state the driver outputs. The **SP490** does not have a driver enable.

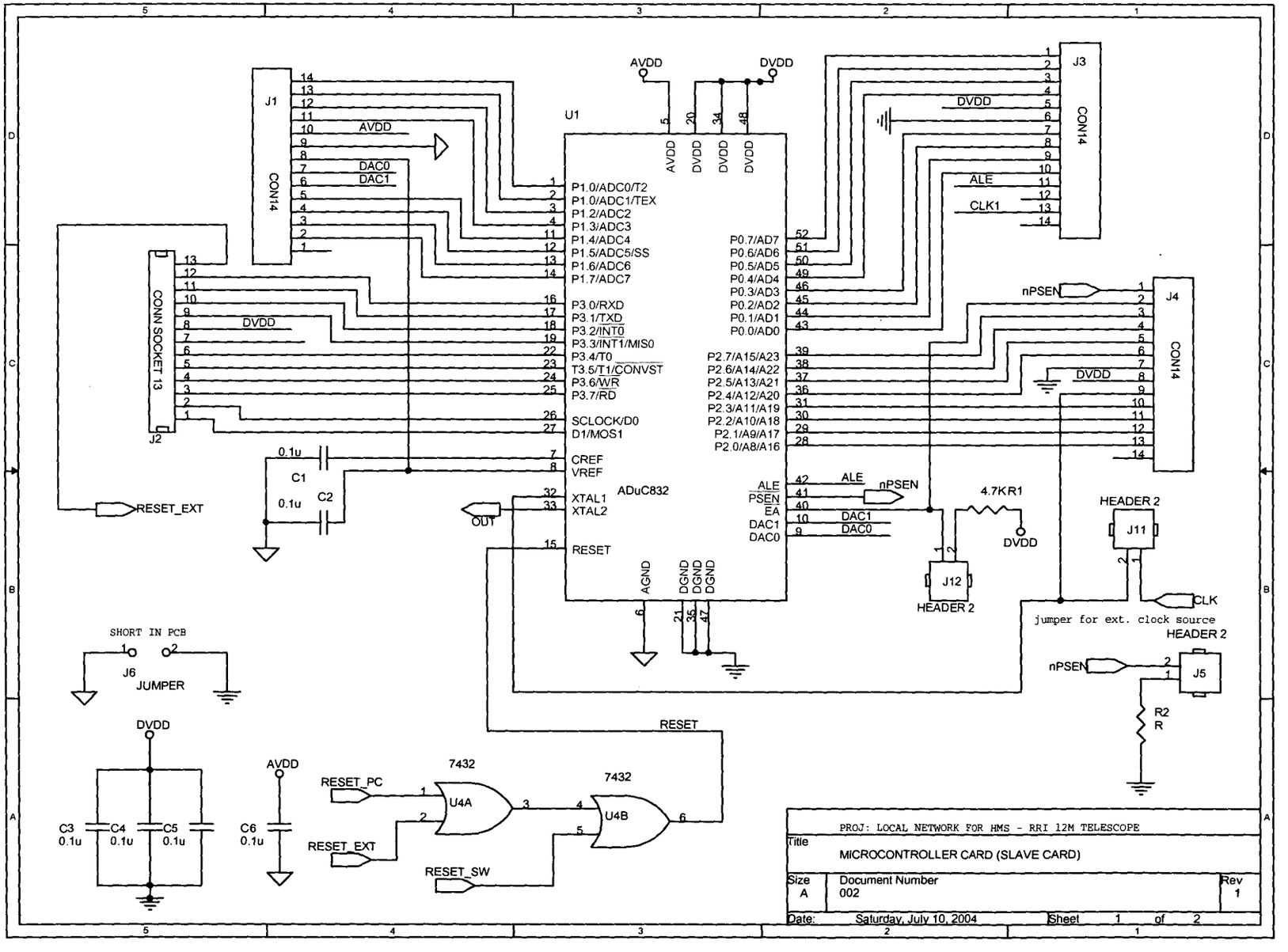
Receiver...

The receivers for both the **SP490** and **SP491** have differential inputs with an input sensitivity as low as ±200mV. Input impedance of the receivers is typically 15KΩ (12KΩ minimum). A wide common mode range of -7V to +12V allows for large ground potential differences between systems. The receivers for both the **SP490** and **SP491** are equipped with the fail-safe feature. Fail-safe guarantees that the receiver output will be in a high state when the input is left unconnected.

The receiver of the **SP491** has a receiver enable control line which is active low. A logic low on $\overline{REB}$ (pin 3) of the **SP491** will enable the differential receiver. A logic high on $\overline{REB}$ (pin 3) of the **SP491** will tri-state the receiver.

Appendix 2


J1
CONN DSUB 9-R
DB9- FEMALE
U1 MAX232
R1IN
R2IN
T1IN
T2IN
R1OUT
R2OUT
T1OUT
T2OUT
C1+
C1-
C2+
C2-
V+
V-
RS232 LEVEL CONVERTER
C1 1uF
C2 1uF
C3 1uF
C4 1uF
VCC
D_EN
R2
R3
GWD
CHASSIS GROUND
100R 0.5W
100R 0.5W
U2 SP491
DI
RO
DE
RE
DRV_IN
REC_OUT
RS232 - RS485
J2
CONN DSUB 9-R
DB9- MALE
Txd-
Txd+
Rxd-
Rxd+
ALWAYS ENABLE
J6
Any one Jumper should present J7
REC_EN1
D_EN
J3
J4
ALWAYS ENABLE
Any one Jumper should present
J5
DRV_EN1
SW1 LOOP BACK
MAX_OUT
DRV_IN
REC_OUT
SW DPDT/SM
MAX_IN
uC Enable Signals
D_EN1
DRV_EN1
R_EN1
REC_EN1
TxD-
TxD+
R5
120R
RxD-
RxD+
R4
120R
TERMINATION RESISTERS
D1
IN4007
C5
10uF
C6
0.1uF
POWER CONNECTOR
R1
330R
D2
LED
PROJ: LOCAL NETWORK FOR HMS - RRI 12m TELESCOPE
Title
RS232 - RS485 CONVERTER CARD (NON ISOLATED)
Size
A
Document Number
001
Rev
1
Date: Tuesday, July 06, 2004
Sheet 1 of 1

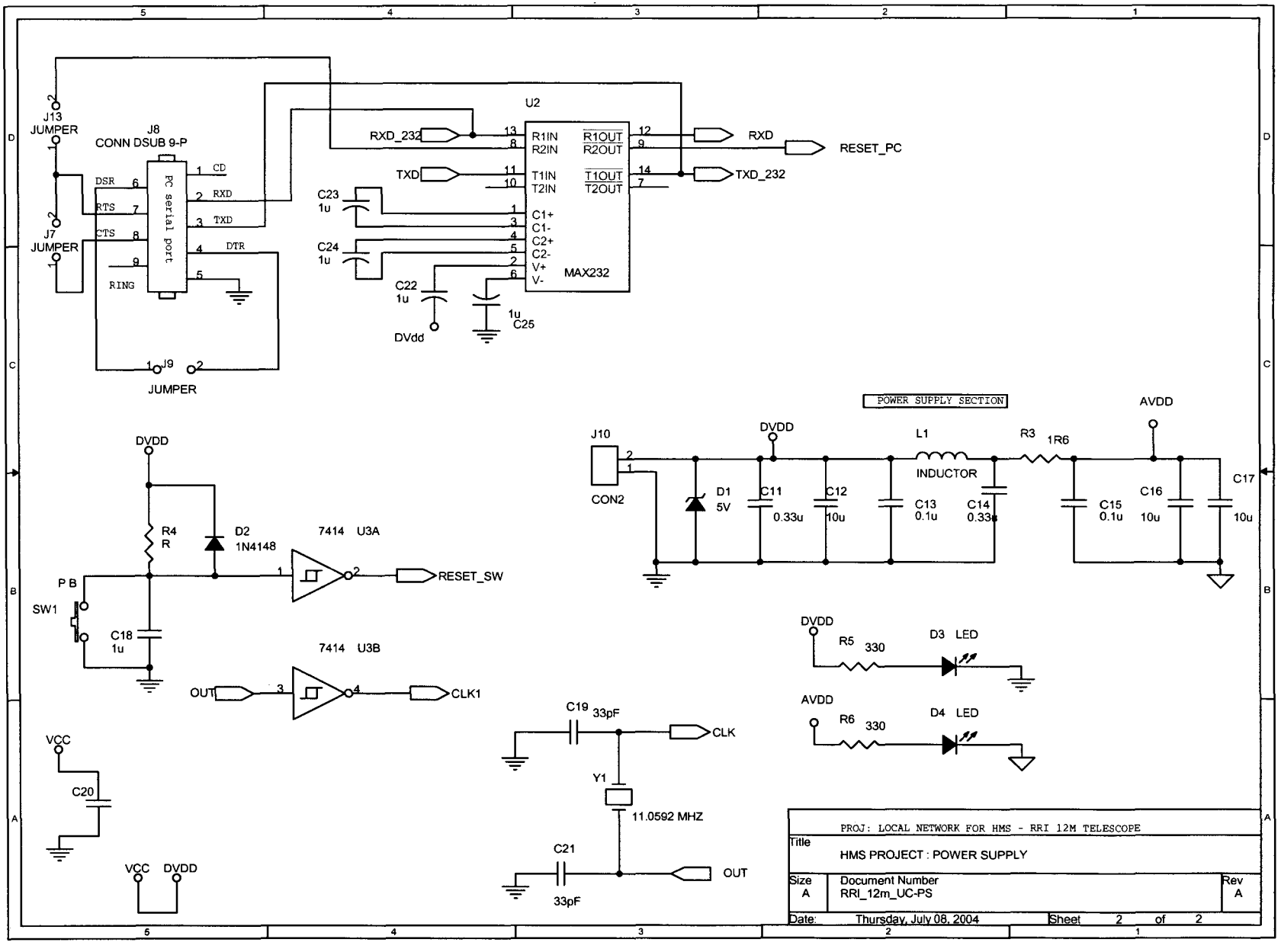


PROJ: LOCAL NETWORK FOR HMS - RRI 12M TELESCOPE

Title: MICROCONTROLLER CARD (SLAVE CARD)

Size	Document Number	Rev
A	002	1

Date: Saturday, July 10, 2004 Sheet 1 of 2


J13
JUMPER
J7
JUMPER
J8
CONN DSUB 9-P
DSR
RTS
CTS
RING
CD
RXD
TXD
DTR
PC serial port
J9
JUMPER
U2
RXD_232
TXD
R1IN
R2IN
T1IN
T2IN
R1OUT
R2OUT
T1OUT
T2OUT
C1+
C1-
C2+
C2-
V+
V-
MAX232
RXD
RESET_PC
TXD_232
C23
1u
C24
1u
C22
1u
C25
1u
DVdd
POWER SUPPLY SECTION
J10
CON2
DVDD
L1
INDUCTOR
R3
1R6
AVDD
D1
5V
C11
0.33u
C12
10u
C13
0.1u
C14
0.33u
C15
0.1u
C16
10u
C17
10u
DVDD
R4
R
D2
1N4148
7414 U3A
P B
SW1
C18
1u
RESET_SW
7414 U3B
OUT
CLK1
DVDD
R5
330
D3 LED
AVDD
R6
330
D4 LED
VCC
C20
VCC DVDD
C19 33pF
CLK
Y1
11.0592 MHZ
C21
33pF
OUT
PROJ: LOCAL NETWORK FOR HMS - RRI 12M TELESCOPE
Title
HMS PROJECT : POWER SUPPLY
Size
A
Document Number
RRI_12m_UC-PS
Rev
A
Date:
Thursday, July 08, 2004
Sheet
2
of
2

Appendix 3

Appendix 3

```
;-------------------------------------------------------------------------
;                  slave_id10.asm
; with the slave addres 0x10H
;-------------------------------------------------------------------------
;This program will receive address from PC, validates the address, if the
;address matches
;receives the command and arguments (if any) and takes appropriate action
;Refer protocol and hms chapter in report.
;-------------------------------------------------------------------------

;Setting SFR addresses.Refer page 17 of ADuC832 manual

RCAP2H EQU 0CBH
RCAP2L EQU 0CAH
TH2 EQU 0CDH
TL2 EQU 0CCH
T2CON EQU 0C8H
ADCCON1 EQU 0EFH
ADCCON2 EQU 0D8H
EADC EQU 0AEH
SCONV EQU 0DCH
ADCDATAH EQU 0DAH
ADCDATAL EQU 0D9H
CHAN    EQU     0               ; convert this ADC input channel..
;-------------------------------------------------------------
;Addresses of program flags
USER_FLAG EQU 20H
ADD_VALID EQU 00H       ;Set when valid address is received
CMD_VALID EQU 01H       ;Set when valid command is received
ARG_LAST  EQU 02H       ;Set when last argument is received
ARG_VALID EQU 03H
P0_SEL   EQU 04H        ;Set when PORT1 IS SELECTED
P1_SEL   EQU 05H        ;Set when PORT1 IS SELECTED
P2_SEL   EQU 06H        ;Set when PORT1 IS SELECTED
P3_SEL   EQU 07H        ;Set when PORT1 IS SELECTED
;-------------------------------------------------------------
;Address of the slave
TEMP    EQU R0
TEMP1   EQU R4
```

```
COUNT   EQU R6
SLAVE_ADD EQU 30H
ADD_RECD  EQU 31H
ADD_ACK   EQU 32H
CMD_RECD  EQU 33H
ARG_RECD  EQU 34H
;-----------------------------------------------------------------
ORG 0000h
      JMP     MAIN            ;jump to main program
;-----------------------------------------------------------------

ORG 0023H   ;Serial vector
    CALL SER_CALL
RETI
;-----------------------------------------------------------------

ORG 0033H   ;ADC Vector
      MOV IE, #00H
      MOV B,ADCDATAH
      MOV     A,ADCDATAL
      CALL    SEND_ADC_DATA
      MOV IE, #11010000B   ;enabling the interrupts.
RETI
;=================================================================
;                  MAIN PROGRAM
;=================================================================

ORG 050H
MAIN:
;Initialize all user flags to zero
    MOV USER_FLAG, #000H
    CLR P2.0
;Store slave address
    MOV SLAVE_ADD, #010H
;Set up UART
    MOV     RCAP2H,#0FFh      ; config UART for 9600 baud
    MOV     RCAP2L,#-7
    MOV     TH2,#0FFh
    MOV     TL2,#-7
    MOV     SCON,#52h
```

```
    MOV     T2CON,#34h
    MOV     ADCCON1,#080h   ; power up ADC
    MOV     ADCCON2,#CHAN   ; select channel to convert
;Interrupt wait loop
    MOV IE, #11010000B
AGAIN:
    CALL    DELAY           ;delay 100ms
    CALL    DELAY           ;delay 100ms
    JMP     AGAIN           ;repeat

;Serial ISR
SER_CALL:
      CALL RxD_ACC
      CALL ADDR_CHECK
RETI

SEND_ADC_DATA:
      CLR C
      RLC A
      JC SET_4
      CLR 0F7H
      JMP NEXT_B
   SET_4: SETB 0F7H
   NEXT_B: CLR C
      RLC A
      JC SET_5
      CLR 0F6H
      JMP NEXT_BI
   SET_5: SETB 0F6H
   NEXT_BI:
      ORL A, #00000011B
      CALL TXD_ACC
      MOV A, B
      SWAP A
      ORL A, #00000011B
      CALL TXD_ACC
   RET

;address check routine
ADDR_CHECK:
      JB ADD_VALID, PROC_COMMAND    ;If add already valid, jump to PROC_COMMAND
```

```
      CJNE A,SLAVE_ADD,return       ;Compare ACC with SLAVE_ADD, if not matched
return
        MOV ADD_RECD, A             ;Store received address in ADD_RECD
      SETB ADD_VALID          ;Set the ADD_VALID flag
      ORL A, #003H            ;Setting pattern 'xxxxxx11'
      MOV ADD_ACK, A          ;Storing ADD_ACK for future use with ping command
      LCALL TxD_ACC           ;Sending acknowledgement to PC
RET
;-------------------------------------------------------------------------
PROC_COMMAND:

      JB CMD_VALID,PROC_ARG   ;If command already checked move to arg process
                              ;routine
      MOV TEMP,A              ;Move second byte received from PC to TEMP
      ANL A,#007H             ;Checking whether pattern is 'xxxxx001'
      CJNE A,#001H,return     ;If not matching, byte is not a command, so return
      MOV CMD_RECD, TEMP      ;If matching, store the byte recived in TEMP to
CMD_RECD

      SETB CMD_VALID          ;Set the CMD_VALID flag
      MOV A,CMD_RECD          ;load ACC again with CMD_RECD
      XRL A,#009H             ;#009H is to check for ping_slave command
      JZ ping_slave           ;if acc is zero jump to ping_slave command
RET

;-----------------------------------------------------
PROC_ARG:
      MOV TEMP,A
      ANL A,#007H
      CJNE A,#006H,return
      MOV ARG_RECD,TEMP
      LCALL EXEC_COMMAND

CLR ADD_VALID
CLR CMD_VALID
CLR ARG_VALID

RET
;-------------------------------------------------------------------------
EXEC_COMMAND:
;XRL A,#0xxH --> here the oommand byte received which is in ACC is XOR'd  with
```

```
;predefined command patterns to jump to the appropriate command functions

      SETB P2.0
      MOV A,CMD_RECD
      XRL A,#011H
      JZ read_port_byte

      MOV A,CMD_RECD
      XRL A,#019H
      JZ read_port_byte

; For Read port bit is implemented in the C program in Control PC
; READ_PORT_BYTE will be called for read_port_bit and the required
; bit will be extacted.

      MOV A, CMD_RECD
      XRL A, #031H
      JZ read_adc
RET

;-------------------------------------------------------------------------
return:
RET
;-------------------------------------------------------------------------
;-------------------------------------------------------------------------
ping_slave:             ; ping slave procedure this sends the address to PC
                        ; 5 times.

MOV B,#005H             ; initialise the couter to 5
PING_LOOP:
      MOV A,ADD_ACK
      LCALL TxD_ACC
      DEC B
      MOV A,B
JNZ PING_LOOP
MOV USER_FLAG, #000H
RETI

;-------------------------------------------------------------------------
read_port_byte:         ; procedure to read port byte and send it to PC
```

```
        MOV A,ARG_RECD

        MOV ARG_RECD,A      ; Store acc content in ARG_RECD

        XRL A,#006H         ; #006H - '000 00 110' B arg_pattern for port-0
        SETB P0_SEL         ;
        JZ SELECT_PORT
        CLR P0_SEL

        MOV A,ARG_RECD
        SETB P1_SEL
        XRL A,#00EH
        JZ SELECT_PORT
        CLR P1_SEL

        MOV A,ARG_RECD
        SETB P2_SEL
        XRL A,#016H
        JZ SELECT_PORT
        CLR P2_SEL

        MOV A,ARG_RECD
        SETB P3_SEL
        XRL A,#01EH
        JZ SELECT_PORT
        CLR P3_SEL

RETI
;-------------------------------------------------------------------------

SELECT_PORT:

        JB P0_SEL,READ_P0
        JB P1_SEL,READ_P1
        JB P2_SEL,READ_P2
        JB P3_SEL,READ_P3

RET
;-------------------------------------------------------------------------
READ_P0:
        MOV A,#0ABh;P0
```

```
      MOV TEMP1,A       ;Register R4 is temp storage
      LCALL SEND_NIBBLE
RET

READ_P1:
      MOV A,#0BAh;P1
      MOV TEMP1,A       ;Register R4 is temp storage
      LCALL SEND_NIBBLE
RET

READ_P2:
      MOV A,#0FEh;P2
      MOV TEMP1,A       ;Register R4 is temp storage
      LCALL SEND_NIBBLE
RET

READ_P3:
      MOV A,#0EFh;P3
      MOV TEMP1,A       ;Register R4 is temp storage
      LCALL SEND_NIBBLE
RET

; ------------------ ADC Reading ----------------------------------------
read_adc:
   SETB SCONV
RET
; ------- end of read_adc -----------------------------------------------
;---------------------------------------------------------------------------

;----- begin of SEND_NIBBLE procedure -----------------------------------
SEND_NIBBLE:

      MOV COUNT,#004H               ; initialise the couter to 5
      LOOP:
            RL A
            DEC COUNT
      CJNE COUNT,#000H,LOOP

      ANL A,#0F0H
      ORL A,#003H
      LCALL TxD_ACC                 ; sends the lower nibble
```

```
      MOV A,TEMP1
      ANL A,#0F0H
      ORL A,#003H
      LCALL TxD_ACC           ; sends the higher nibble
;-------------------------------------------------------------------------
RET
;---- end of 'SEND_NIBBLE' procedure ------------------------------------

RxD_ACC:
      JNB RI, $
      MOV A, SBUF
      CLR RI
RET

TxD_ACC:
      SETB P2.0   ; To keep the enable the BUS  from the tristate
      CLR TI
      MOV SBUF, A
      JNB TI,$
      CLR TI
      CLR P2.0    ; To keep the BUS in the tristate
CALL DELAY
RET
;-------------------------------------------------------------------------
;return:
;RET
;-------------------------------------------------------------------------
              ; DELAY SUBROUTINE

DELAY:                          ; Delays by 100ms * A
            MOV   R1, #01h      ; Acc holds delay variable
 DLY0:      MOV   R2,#022h      ; Set up delay loop0
 DLY1:      MOV   R3,#0FFh      ; Set up delay loop1
            DJNZ  R3,$          ; Dec R3 & Jump here until R3 is 0
            DJNZ  R2,DLY1       ; Dec R2 & Jump DLY1 until R2 is 0
            DJNZ  R1,DLY0       ; Dec R1 & Jump DLY0 until R1 is 0
RET               ; Return from subroutine
;_________________________________________________________________________
END
; ------------------------------------------------ end of slave_id10.asm ------
```