| SYM | DESCRIPTION | DATE | E.E. | M.E. | Q.A. |
|---|---|---|---|---|---|
| A | ORIGINAL | 9/20/88 | VF | LAG | ---- |
| B | REVISED | 11/17/88 | VF | LAG | ---- |
| C | REVISED | 10/30/89 | VF | LAG | ---- |
| D | REVISED | 8/17/92 | CLY | BJD | RMM |
| E | REVISED - FORMAT CHANGE | 10/04/94 | ---- | BJD | ---- |
| F | REVISED | 5/14/97 | TPC | BJD | JT |
| G | REVISED | 8/17/00 | RDW | BJD | JT |
| H | REVISED | 10/24/00 | RDW | BJD | JT |

| | |
|---|---|
| DRAWN | DATE |
| CHECKED | DATE |
| ENGINEER | DATE |
| APPROVED - CHIEF ENG. M.E. L.A. Gorman | DATE 9/23/88 |
| APPROVED - CHIEF ENG. E.E. V. Formanek | DATE 9/23/88 |
| APPROVED - Q.A. R. Kinzie | DATE 9/23/88 |
| APPROVED TEST | DATE |

MCL INC A MITEQ COMPANY

501 South Woodcreek Road
Bolingbrook, Illinois 60440-4999

# SPECIFICATION FOR SAbus INTERFACE

40A1715

# TABLE OF CONTENTS

## 1.0 INTRODUCTION

### 1.1 Description

In recent years, more satellite earth station owners are requiring remote monitor and control capability to effectively manage their systems. The most common approach has been to provide individual contact-closure interfaces from each piece of equipment to a central control panel.

The SAbus is an interface designed explicitly as an economical method of providing complex remote monitor and control capability in satellite earth station equipment. Although it requires a limited degree of “intelligence” in each earth station component, this provides comprehensive monitor and control capabilities and a degree of flexibility not available in customized system solutions.

### 1.2 Applicable Documents

| | |
|---|---|
| EIA-422B | Electrical Characteristics of Balanced Voltage Digital Interface Circuits |
| EIA-232F | Interface between Data Terminal Equipment & Data Circuit-Terminating Equipment Employing... |
| EIA-423A | Electrical Characteristics of Unbalanced Voltage Digital Interface Circuits |
| EIA-449 | General Purpose 37-Position & 9-Position Interface... |

## 2.0 ELECTRICAL SPECIFICATIONS

Electrically, the SAbus is compatible with RS-422, which is an interface specification that is meant to eventually replace RS-232 as the industry standard for computer interconnection. RS-422 is a unipolar, balanced, 5-volt serial interface designed to connect equipment which must exchange data over considerable distances with high-noise immunity and high speed. Standard IC drivers and receivers are available for RS-422 that convert to and from TTL logic levels. The SAbus subset of RS-422 allows up to 64 devices to be connected in parallel with up to 4,000 feet between any master and groups of slaves.

## 3.0 PHYSICAL SPECIFICATIONS

The physical implementation of the SAbus interface takes the form of a single 9-pin “D” connector located on the rear panel of a compatible device. This connector and its wiring is compatible with EIA RS-449, which is the mechanical specification for RS-422/423-compatible equipment. The 9-pin connector chosen for the SAbus connector is described as the secondary interface in RS-449 and has ONLY the four data lines and circuit, common and shield. No hardware handshaking is used in the SAbus protocol, so all the control lines specified for the standard 37-pin connector are not needed. An SAbus compatible device that is only capable of operating as a slave has a female connector, whereas masters have male connectors. All SAbus devices can operate in electrical parallel with only a single 9-conductor cable required to connect all devices controlled by a master. **Figure 1** illustrates the connection of a master and multiple slaves.

## 4.0 SAbus PROTOCOL

The SAbus interface is a multi-drop, balanced line, asynchronous, full-duplex communications link designed to interconnect equipment for remote control and switching applications. Products that are SAbus compatible can be linked together over a parallel-connected 4-wire circuit without regard to their particular function.

Each SAbus configuration can have one master and up to 63 slave devices. Each slave device is internally configured to respond to a unique address. A master could be a protection switch, earth station controller, or any micro or mini-computer that is electrically and operationally compatible with the SAbus. Since the electrical specifications are very similar to RS-422 and RS-449, virtually any computer that meets these standards is capable of controlling remote devices over the SAbus with proper programming.

## 5.0 DATA FORMAT

SAbus data format supports industry’s standard asynchronous ASCII format with one start bit, seven data bits, even parity, and one stop bit (see **Figure 6**). The ASCII control character subset 00-1F (hex) is reserved for message control. The printable ASCII characters 20-7F (hex) are used for address, command and data characters. The standard bus data rate via direct connect (up to 4,000 feet) is 9,600 BAUD, the data rate for devices connect to a master via modem is 1,200 BAUD.

## 6.0 MESSAGE PROTOCOL

Message format and protocol over the SAbus is derivative of IBM's binary synchronous communications protocol (BISYNC). The master station sends a command over the bus to all remote stations. The station, whose address is contained in the second byte of the command message, carries out the requested commands and then replies with a response message containing its own address and status information relating to its present condition. A remote station only sends a response following a command containing its unique address from the master. This prevents bus contention caused by more than one remote device communicating over the SAbus at the same time.

A remote device ignores all commands that contain parity or checksum errors, protocol errors, a wrong address, or message overrun errors (see paragraph 7.2 and **Figure 5**). A remote device replies with a not-acknowledged (NAK) character if it receives an invalid command or data.

### 6.1 Message Format

Command message (see **Figure 4**) begins with Start-of-text byte, STX, followed by a remote address, a command byte and multiple data bytes. The End-of-text byte, ETX, is sent following the last data byte, and the message is terminated by a checksum character. Response messages are identical to command messages in format with the exception of the ACK (Acknowledge) or NAK (Not Acknowledge) character at the start of the message instead of STX. **Figure 4** illustrates the format of the command and response messages. A command or reply message may have a variable length up to a maximum of 132 bytes, including delimiters and checksum. Although most currently implemented SAbus devices require no, or very few, data bytes, the capability for long messages is built into the protocol so that future applications requiring the transfer of large amounts of data can be accommodated. SAbus devices should observe the length of all messages, predefined by their communication protocol, and NAK messages longer than permitted.

### 6.2 Message Delimiters

A command message begins with STX (02 hex), the ASCII Start-of-text control character. A message-acknowledged reply begins with ACK (06 hex), the ASCII Acknowledge control character and a message-not-acknowledged reply begins with NAK (15 hex), the ASCII Not Acknowledge control character. All messages end with the ETX (03 hex), the ASCII End-of-text control character, followed by the checksum byte.

### 6.3 Address Character

The device address must be a valid ASCII printable character between "0" and "o", or 30 through 6F in hex; thus, 64 SAbus addresses are possible. Address 0 (ASCII 30) is reserved as an "all call" address and will cause all devices on the bus to execute a command without generating a reply.

## 6.4 Command Character

The command character (CMD) immediately follows the device address and specifies one of a possible 80 different commands for a particular device. Values from 30 to 7F (hex) are allowed. Commands may be completely device dependent with the exception of command 30 (hex) which must cause a device to return its six character device type and command 31 (hex) which is a status poll.

## 6.5 Command and Reply Data

A command or device reply may contain from 0 to 128 data characters and is restricted only to printable ASCII characters 20-7F (hex).

## 6.6 Check Character

The last character of any SAbus message is the check character (CHK). This character is simply the bit-by-bit exclusive OR of all characters in the message starting with STX character through the ETX character. This forms a Longitudinal Redundancy parity check over the entire message.

## 6.7 Message Timing

Different devices will require varying times to execute commands from a master. A receiver, for example, may be instructed to change frequency and may require up to a second for the synthesizer to lock. This should not, however, prevent it from immediately acknowledging the command. The NAK or ACK reply does not signify that a function has actually taken place, but only that the message was received and understood. A status reply should indicate if a device is executing a time-consuming function.

An MCL remote device must begin responding to a command within 10 milliseconds after receiving the last character of that command. Typical response time for an MCL remote device is approximately 5 milliseconds. See **Figure 7**. The top trace is the remote device receiving the command message and the bottom trace is the remote response to that message. All bytes in a response must be continuous with no undefined characters in between each character of the message. The master device must leave at least one character time between the end of a slave device response and the next master device transmission on the bus. If a remote device does not respond within 150 milliseconds the master-controller should attempt to re-establish communication by re-polling this device twice. **Figure 5** shows a remote device SAbus state table and timing requirements.

Baud rates available vary from model to model, therefore, may depend on the design of a particular MCL model number.

At least a 10-bit time delay must be inserted between command messages in order to “wake up” a remote device. Once the device is awakened by data on the bus, it looks for STX followed by its address. If it does not see its own address, it ignores the rest of the message by going to “sleep” and remains in such state until the serial data lines idle for at least 10-bit times or approximately 10 milliseconds.

**NOTE**

MCL models designed before 1990 do not meet all of these timing requirements.

## 6.8 SAbus Command Restrictions

All SAbus-compatible devices must respond to a command “0”, 30 (hex), with 6 data bytes of ASCII characters in the following form:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ACK | ADDR | 30 | D1 | D2 | D3 | D4 | D5 | D6 | EXT | CHSUM |

where D1-D4 are four ASCII characters representing the model number and D5-D6 are two ASCII characters representing a software version number.

If more than one command is required to obtain status information of a device’s functions that can cause setting of a change bit, then the device must implement a clear change bit command and this must be the only command which causes the change bit to be cleared. If several commands have to be executed in order to set all the information that can cause a change bit to be set, then multiple change bits may be used to reduce the bus traffic.

Whenever possible, SAbus numeric data should be sent encoded as ASCII data characters, and only in cases where it cannot be avoided, numeric data should be sent in Binary or BCD packed format. Status bits in data bytes, i.e. change bits, alarm bits, etc., should occupy no more than four bits in the low-order nibble. The high-order nibble should be set to 03 to guarantee that the byte will contain a printable ASCII character.

## 7.0 SLAVE DEVICE STATE DIAGRAM

### 7.1 Introduction

General Description. The Slave State diagram (see **Figure 5**) presents the required protocol implementation at the Slave device that guarantees the proper transfer and processing of communication messages sent by a Master-controller over SAbus.

State Diagram Notation. Each state that a slave device can assume is represented graphically as a circle. A single-digit number is used within the circle to identify the state.

All permissible transitions between states are represented graphically by arrows between them. Each transition is qualified by a condition that must be true in order for the transition to occur. The device will remain in its current state if conditions which qualify transitions leading to other states are false or conditions that qualify pseudo-transitions are true. A pseudo-transition is a transition that occurs within the same state and is represented graphically by arrows leaving from and arriving at the same state. **Table 1** describes mnemonics used to identify transitions in the state diagram.

**Table 1** - Mnemonics

| MNEMONICS | DESCRIPTION |
|---|---|
| STX | Start-of-Text ASCII control character, is used as a header in SAbus command message to identify the beginning of a new message. |
| ETX | End-of-Text ASCII control character, used as a termination character in SAbus messages to identify the end of data. |
| Checksum | LRC byte (Longitudinal Redundancy Check) is a last byte in the SAbus message data block. The value of LRC byte is the exclusive OR of all message bytes including the STX and the ETX bytes and is used to detect errors during transmission of data. |

## 7.2 States Description

State 1 (Device Idle State). In State 1, a device is ready to receive a new message, and therefore, must complete any previous message reception. A device always powers on in State 1.

A device will exit State 1 and enter State 2 (Device Addressed State) only if STX byte is received.

State 2 (Device Addressed State). In State 2, a device is waiting to receive the address byte, the second byte of SAbus command message.

A device will exit State 2 and enter:

1) State 3 (Device Data State) if received address byte equals a device’s address.
2) State 1 (Device Idle State) if received address byte does not equal a device’s address.
3) State 2 (remain in current state) if STX byte is received, which may be the beginning of a new message data block.

State 3 (Device Data State). In State 3, a device is engaged in receiving the command and associated data bytes sent by a master-controller.

A device will exit State 3 and enter:

1) State 4 (Device Date Error State) if ETX byte is received signifying the end of data in the message.
2) State 1 (Device Idle State) if invalid command, or data character, or incorrect number of data bytes is received.

States 4 (Device Data Error State). In State 4, a device is waiting to receive a Checksum byte which tests the transmitted message for errors.

A device will exit State 4 and enter:

1) State 5 (Command Execute State) if a Checksum byte is true — received LRC value of Checksum byte equals to LRC value computed by a device during message reception.
2) State 1 (Device Idle State) if a Checksum byte is false — received LRC value of Checksum byte does not equal to LRC value computed by a device during message reception.

State 5 (Command Execute State). In State 5, a device, having completed a reception of SAbus message, executes a device’s function specified by a command byte. A device will send an appropriate response message to a master-controller within 10 milliseconds after receiving the last character of the message.

A device will always exit State 5 and enter Device Idle State, State 1.

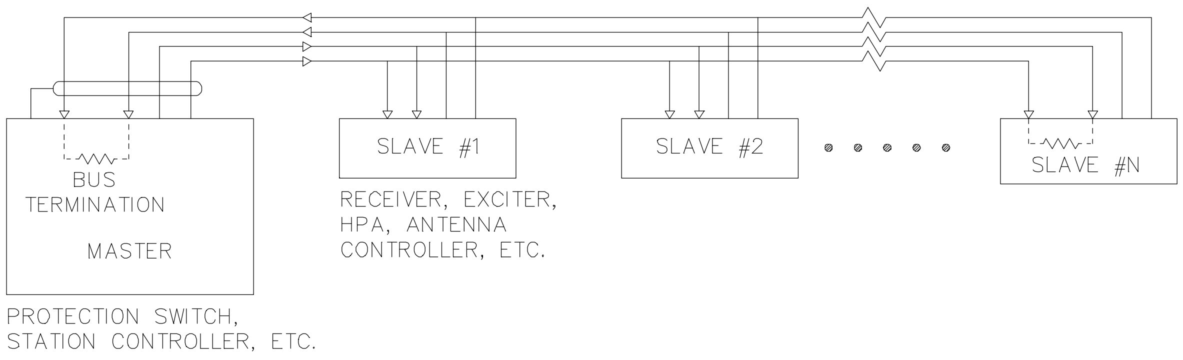
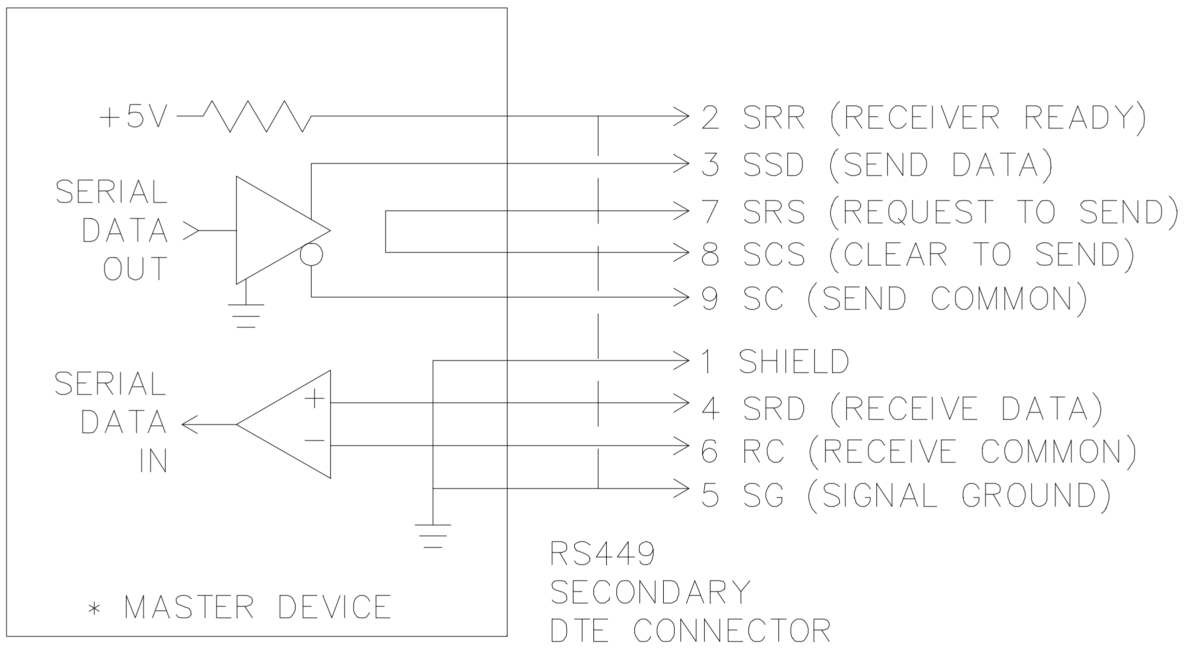


**Figure 1** - SAbus Master and Slaves



* MASTER DEVICE MAY BE A PROTECTION SWITCH, STATION CONTROLLER OR ESSENTIALLY ANY GENERAL PURPOSE MINI OR MICRO-COMPUTER.

**Figure 2** - SAbus Master Device Connections

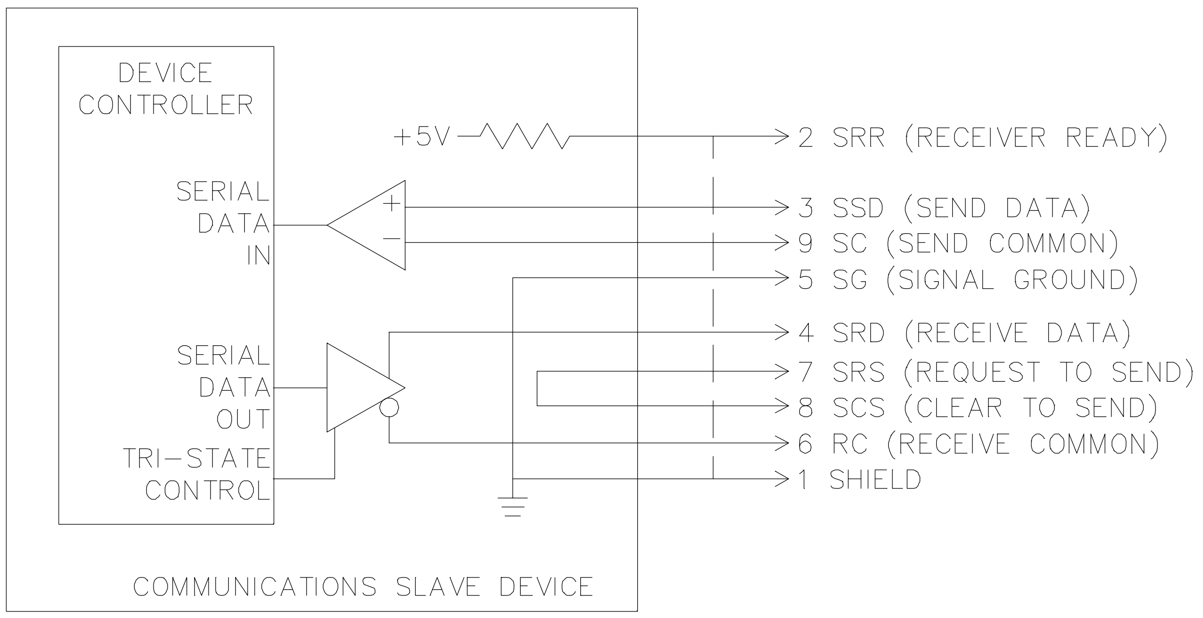


**Figure 3 -** SAbus Slave Device Connections

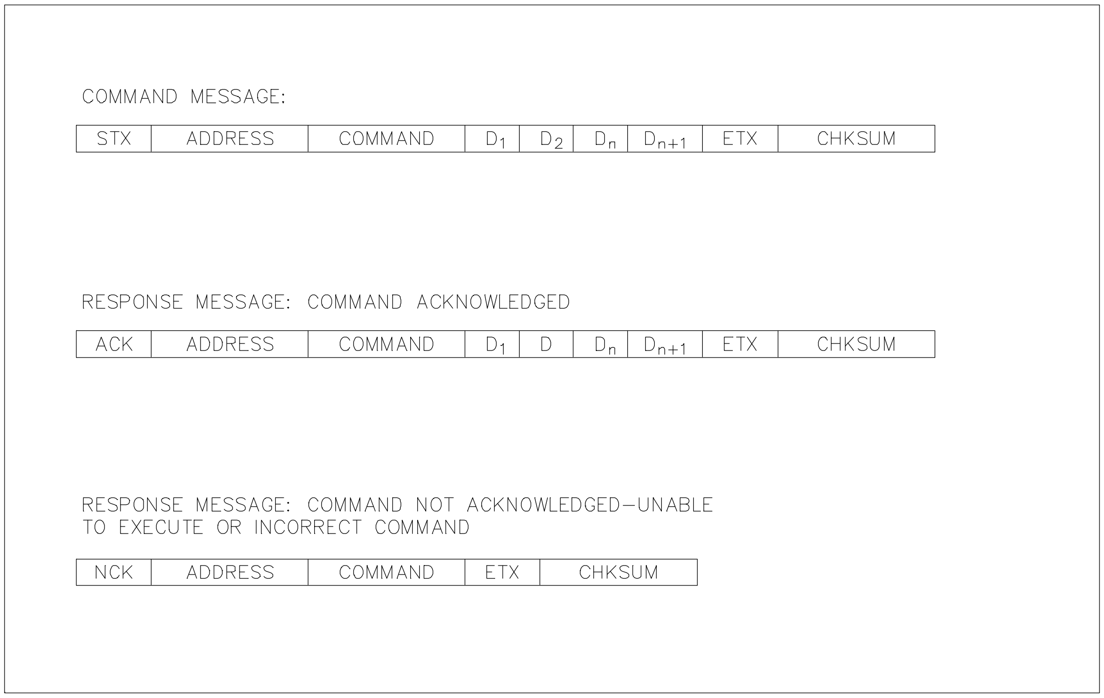

COMMAND MESSAGE:
STX | ADDRESS | COMMAND | $D_1$ | $D_2$ | $D_n$ | $D_{n+1}$ | ETX | CHKSUM
RESPONSE MESSAGE: COMMAND ACKNOWLEDGED
ACK | ADDRESS | COMMAND | $D_1$ | D | $D_n$ | $D_{n+1}$ | ETX | CHKSUM
RESPONSE MESSAGE: COMMAND NOT ACKNOWLEDGED—UNABLE TO EXECUTE OR INCORRECT COMMAND
NCK | ADDRESS | COMMAND | ETX | CHKSUM

**Figure 4** - SAbus Message Format

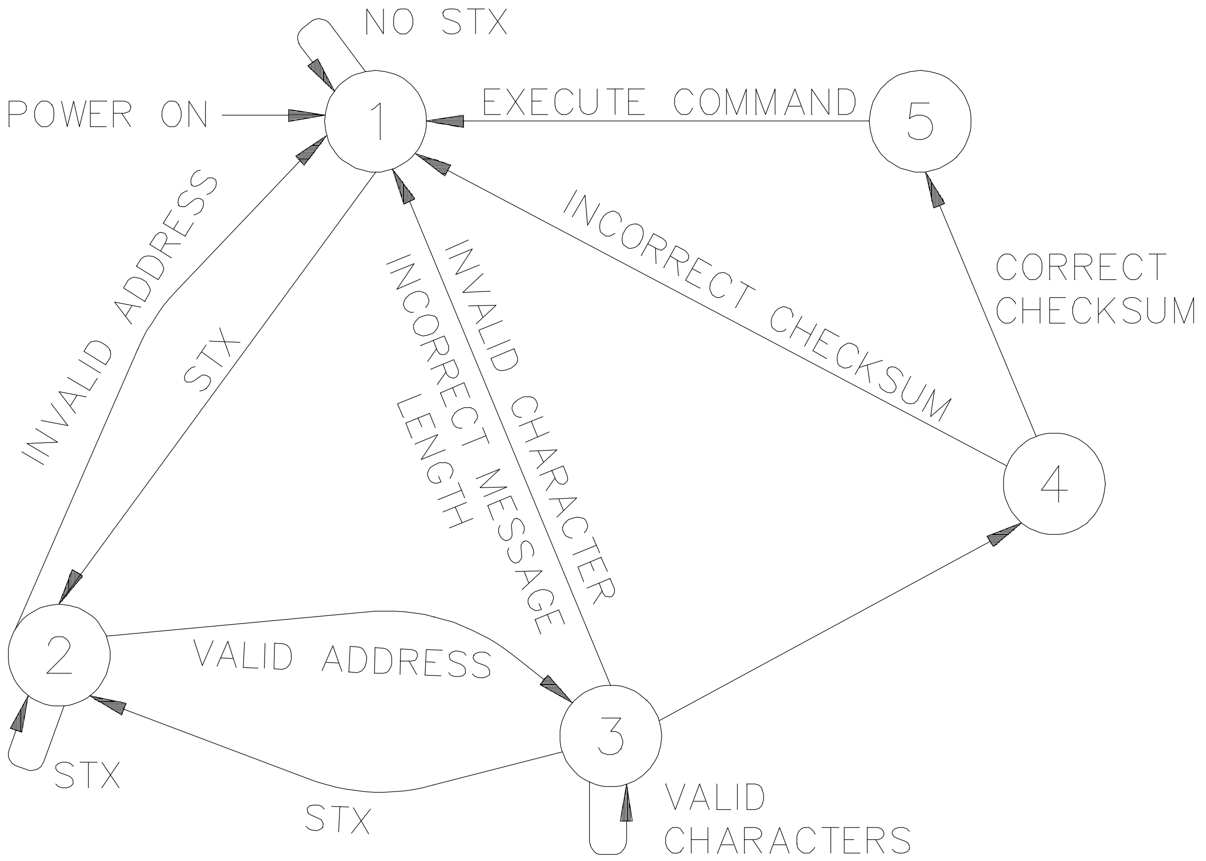
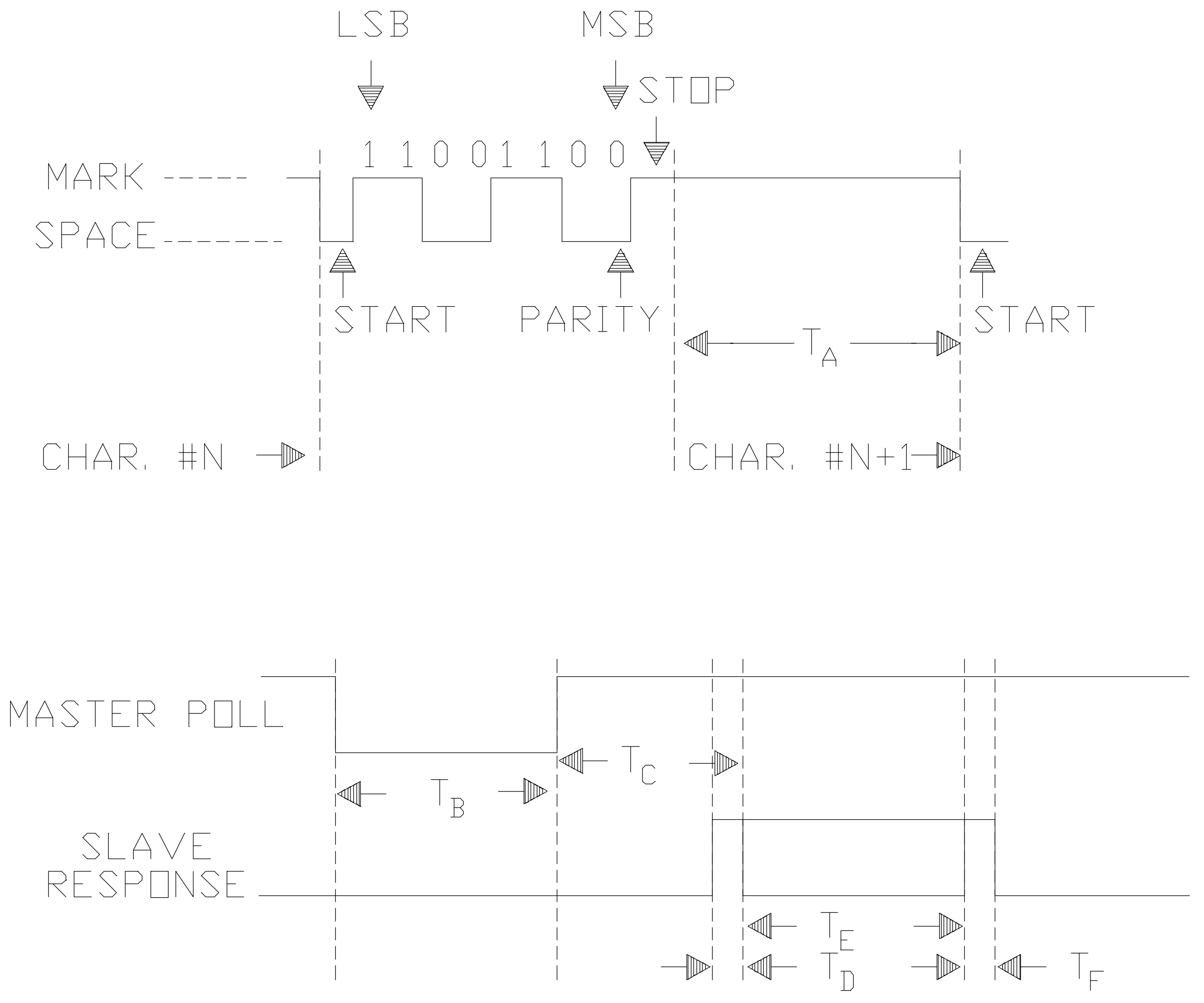


**Figure 5** - Slave State Diagram



$T_A$ = TIME FROM THE END OF CHAR. #N TO START OF CHAR. #N+1

$T_B$ = LENGTH OF MASTER POLL

$T_C$ = SLAVE RESPONSE TIME

$T_D$ = LEADING TRISTATE ENABLE TIME (TO THE FIRST BIT OF THE FIRST CHAR.)

$T_E$ = LENGTH OF THE SLAVE RESPONSE

$T_F$ = TRAILING TRISTATE ENABLE TIME (FROM LAST BIT OF LAST CHAR.)

**Figure 6 -** Message Timing

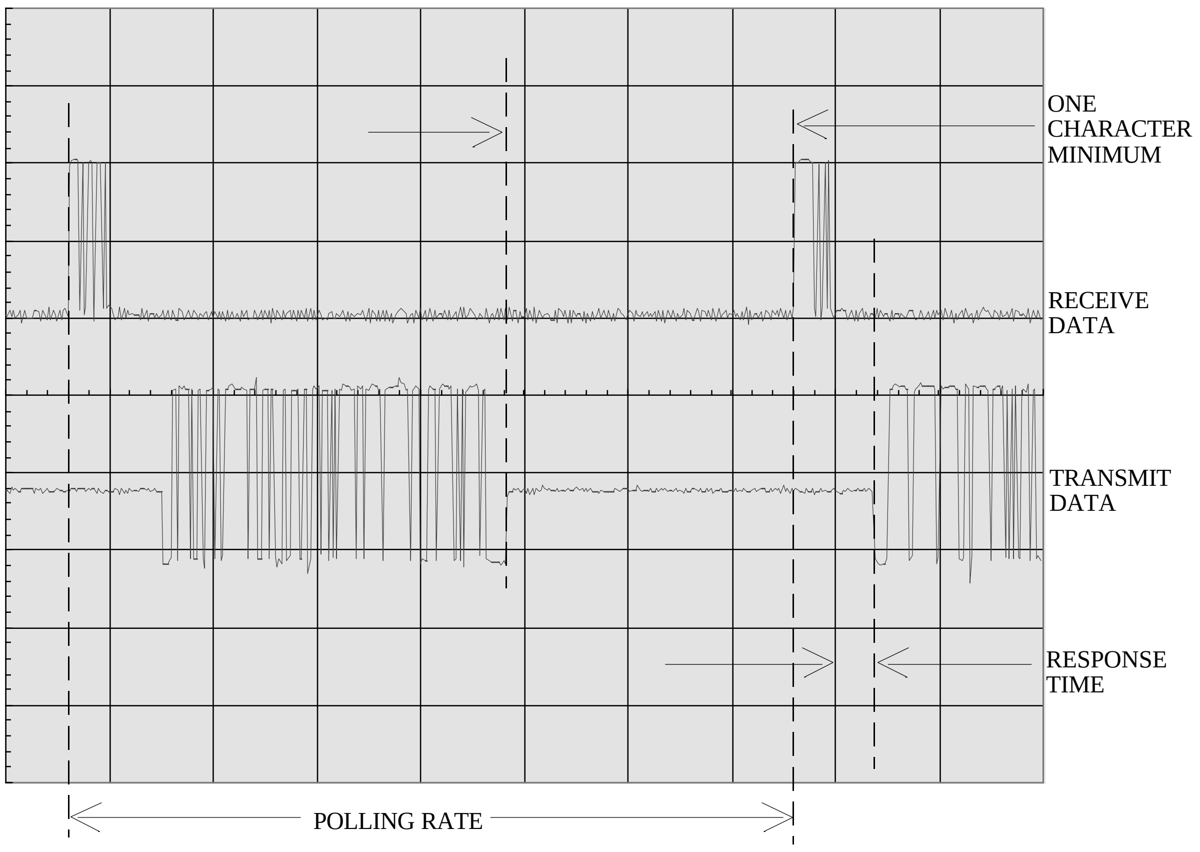


**Figure 7** - Protocol Timing