



CY233

Local Intelligent
Network Controller

Cybernetic Micro Systems, Inc. software products are copyrighted by and shall remain the property of Cybernetic Micro Systems, Inc. Duplication is subject to a license from Cybernetics. Cybernetic Micro Systems, Inc. reserves the right to improve design or performance characteristics. Cybernetic Micro Systems, Inc. assumes no responsibility for the use of any circuitry other than circuitry embodied in Cybernetic products. No other circuit patent licenses are implied.

Information furnished by Cybernetic Micro Systems, Inc. is believed to be accurate and reliable. However, no responsibility is assumed by Cybernetic Micro Systems, Inc. for its use, nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent right of Cybernetic Micro Systems, Inc. Further, Cybernetic Micro Systems, Inc. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation to notify any person or organization of such revision or changes; and Cybernetics assumes no responsibility for any errors which may appear in this document and makes no commitment to update the information contained herein.

The following are trademarks of Cybernetic Micro Systems, Inc:

Bin-ASCII
CYMPL
Analog-ASCII
ASCII-analog
Sim8048
Sim8051
Sim8096
Sim7000
dBUG88
dBUG/EGA
dICE-51
dICE-96

Copyright 1987 by CYBERNETIC MICRO SYSTEMS, INC.
All rights reserved; no part of this publication may
be reproduced without the prior written permission of

Cybernetic Micro Systems, Inc.
P.O. Box 3000 • San Gregorio CA 94074
Phone: 650-726-3000 • Fax: 650-726-3003
Web: <http://www.ControlChips.com>
Email: Info@ControlChips.com

Table of Contents

SECTION 1

Introduction to the CY233

Introduction to the CY233.....	1
Case I: Host Ring Network.....	3
Case II: Peer Ring Network.....	3
Guidelines for CY233 Use.....	4
Organization of this Manual.....	4
Component Level Interfacing.....	5
Logic Diagram.....	6
Design Simplicity.....	6
Production Economics.....	7
Flexibility of Networks.....	8
Demonstration System.....	10
CY233 Overview and Comparison to the CY232.....	11
Serial Ring Network.....	12

SECTION 2

PIN DESCRIPTION

Pin Functions.....	13
Pin Configuration.....	13
Pin Description.....	14
Description of Signals.....	15
Communication Modes.....	30
Independence of Operations.....	31

SECTION 3

THE SERIAL SIDE

Concept of Messages.....	32
Standards.....	34
RS-232 Specifics.....	35
RS-232 Polarities and Terms.....	36
CY233 Waveforms.....	37
RS-232 Connector Pins.....	38
RS-449	39
TTL Specifics.....	39
Other Standards.....	39

SECTION 4

THE PARALLEL SIDE

The Three Buses: Overview.....	41
Address Bus Specifics.....	41
Encoded Parallel Addressing.....	43
Decoded Parallel Addressing.....	43
Data Bus Specifics.....	45
Control Bus Specifics.....	46
Messages to Parallel Devices.....	56
Interface Example.....	57

SECTION 5
CHARACTERS AND MESSAGES

The Alphabet.....	58
ASCII Table.....	59
Start and Stop Bits.....	60
Parity.....	61
Character Length.....	62
Baud Rate.....	63
Commands,Address,Data,and Terminators.....	64
Networks,Masters,and Slaves.....	66
Echo Modes.....	67
Simple Network Configurations.....	69

SECTION 6
UART Mode

The CY233 as a UART.....	70
--------------------------	----

SECTION 7
CY233 Commands

The Command Set.....	72
Network Serial Commands.....	75
Parallel Commands.....	76
LAN Serial Commands.....	77

SECTION 8
Network Serial Command Details

Read (R) Command.....	78
Write (W) Command.....	79
Transfer (X) Command.....	79
Fill (M) Command.....	80
Display (N) Command.....	80
All Echo (J) Command.....	81
Enable Error (K) Command.....	81
Initialize (I) Command.....	82
Query (Q) Command.....	82
Timeout (T) Command.....	83
Periodic Delay (P) Command.....	84
Network Test (@) Command.....	85
Address Sense (S) Command.....	85
Token (U) Command.....	86
Reserved Commands.....	87

SECTION 9

Parallel Command Details

Initialize (I) Command.....89
All Echo (J) Command.....89
Parallel Error Enable (K) Command.....89
Handshake Timeout (T) Command.....89
Periodic Time Delay (P) Command.....90
Load Second Status Byte (L) Command.....90
Example of "Load Status Byte" (L) Command.....91
Sense Address (S) Command.....92
Example of Use of the "Sense" Command.....93
Token (U) Command.....94

SECTION 10

LAN Serial Command Details

Write (W) Command.....96
Read (R) Command.....97
Initialize (I) Command.....97
Query (Q) Command.....97
Handshake Timeout (T) Command.....98
Local Address Query (@) Command.....98
Parallel Pass (P) Command.....99
Token Timeout Details.....101

SECTION 11

Basic Networks

The CY233 NETWORK Mode.....102
Console to CY232.....104

SECTION 12

Point to Point Example

Communication Example.....109

SECTION 13

Advanced Serial Side Examples

Basic Configurations.....110
Bi-Directional Bus.....111
Dual Bus Network.....113
Ring Network.....114

SECTION 14
RING NETWORKS

An RS-232 Network.....	116
When to Echo.....	117
Ring Communications.....	119

SECTION 15
Local Area Networks

The Local Area Network Controller.....	124
Local Area Network Node Design.....	126
The Host Ring: A LAN with Host Computer as One Node.....	127
The Peer Ring: A CY233 Based LAN.....	129
"Who am I?".....	132

SECTION 16
Parallel Side Examples

Further Description of Parallel Data Transfers.....	134
Example Using an 8212 as a Latched Output Port.....	136
Example Using an 8212 as an Interrupting Source.....	138
Example of Connecting to an 8255.....	139
Example Using the CLK Output.....	140
Additional Ideas.....	141

SECTION 17
CYxxx Examples

CYxxx Family Members.....	143
CY360 Waveform Synthesizer.....	143
CY5xx Stepper Motor Controllers.....	144
CY600 Data Acquisition Controller.....	146
CY750 Programmable Controller.....	148

SECTION 18
Electrical Specifications

Electrical Conventions.....	149
Reset Circuitry.....	150
Clock Circuits.....	150

SECTION 19
Basic Language Driver for the CY233

Basic Language Driver for the CY233.....	151
--	-----

SECTION 20
Getting your CY233 Running

Checklist of Things to Do.....	153
--------------------------------	-----

Introduction to the CY233

You might think of the CY233 as an 'Intelligent UART'.

The primary function of a UART is to transfer information between two separate, asynchronous systems in time multiplexed or serial fashion. The 'classic' UART generally handles data flow between two systems, usually only one of which is intelligent. The CY233, an 'Intelligent UART', is designed to handle data flows between any number of systems, of which any or all may be intelligent.

Did you realize that UARTs are older than microcomputers? The Universal Asynchronous Receiver/Transmitter, or UART, was commercially available in the 1960s, whereas Intel introduced the first microprocessor, the 4004, in 1971. Any comparison of the evolution of microprocessors from the four bit 4004 in 1971 to the thirtytwo bit 80386 in 1987 with the evolution of UART technology during the same period, shows UARTs have not kept up.

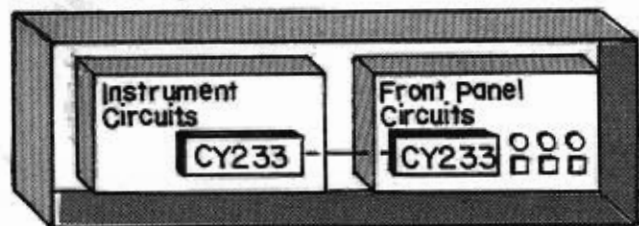
The CY233 allows you to keep up. Since RS-232 communications ports are standard on computers, the CY233 provides a means to connect any computer to any other computer. The CY233 also provides nice solutions to interfacing personal computers to TTL circuits, in fact, up to 2048 I/O lines can be accessed via one serial line from your computer.

Since old, 'dumb' UARTs are so universally applicable, the CY233 intelligent UARTs might be expected to find the same range of application. CY233 applications will be discussed in terms of the following classifications:

Intra-System:

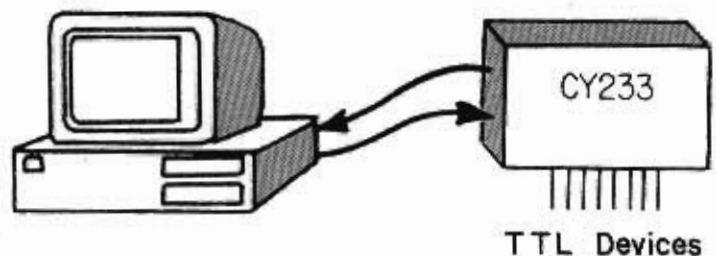
(components to components)

Complete System

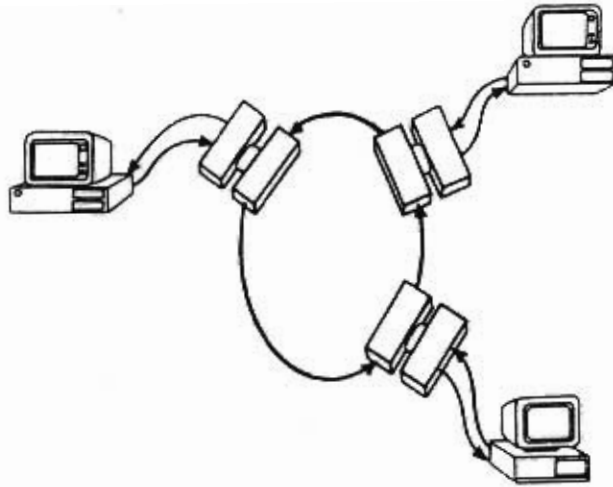


Extra-System:

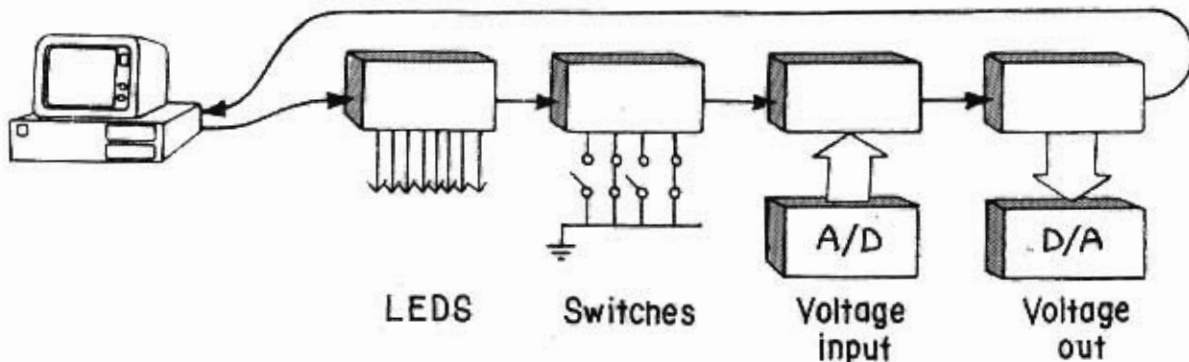
(system to components)



Inter-System:
(system to system)



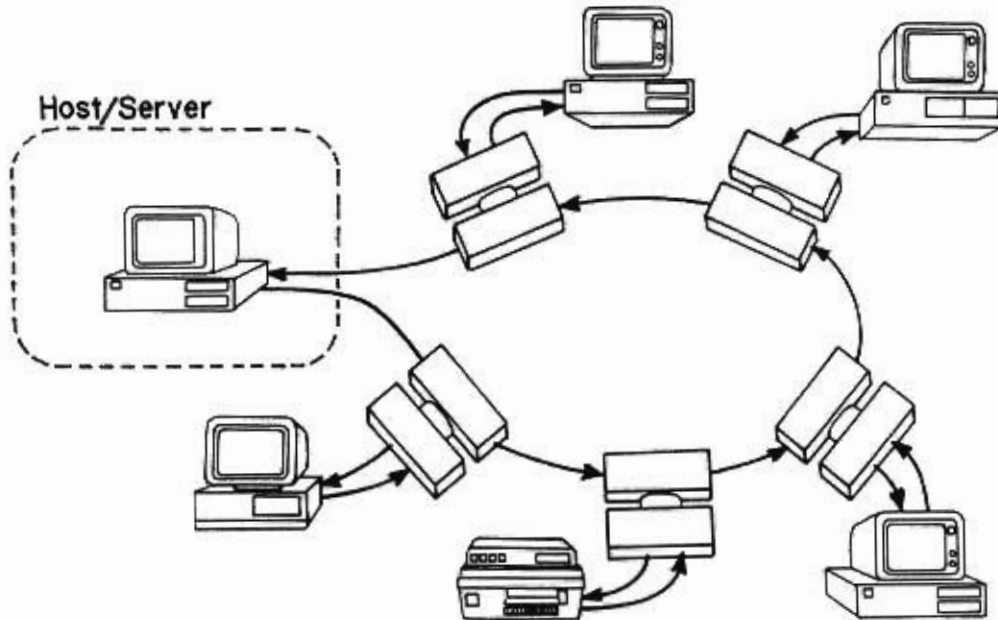
The CMOS CY233 evolved from Cybernetic Micro Systems' NMOS CY232 parallel/serial interface and network controller. Although the two devices are not pin compatible, all of the CY232 functions are included as a subset of the CY233. The CY233 is 20 times faster than the CY232, operating up to the standard rate of 38 K Baud with a non-standard maximum rate of 57.6 K Baud. The CY232 was designed to address applications in the Intra- and Extra-System categories described above. While the 'classic' UART interfaced an intelligent system to a dumb subsystem, the CY232 interfaced an intelligent system to up to 255 dumb subsystems. A typical CY232 application might have one intelligent 'host' communicating serially with many non-intelligent local parallel devices, such as switches, LEDs, A-to-Ds, D-to-As, etc.



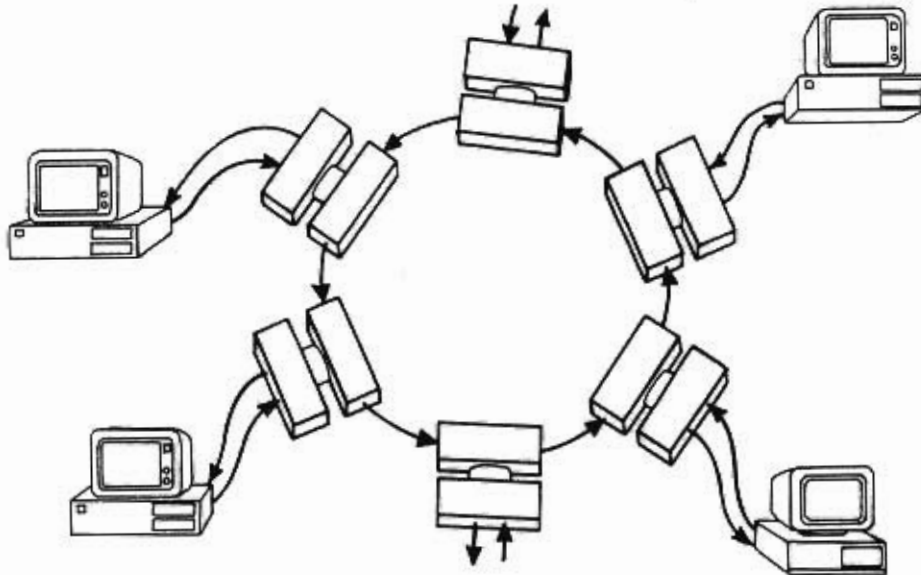
The CY233 retains these CY232 functions but also provides all functions necessary to support Inter-System applications such as Local Area Networks (LANs) including even sophisticated 'Token' passing support. While the CY233 can also address up to 255 local "non-intelligent" devices, it has been optimized for applications in which some or all of the CY233s on a ring network are intelligent devices such as micro computers. The primary support for such local intelligent devices comes in the form of new system commands and more complex parallel side transfers. In particular, local devices can sense their own identity and the identity of all other devices on the ring, and can set status bytes, read error status, and initiate a variety of operations.

Finally, the CY233 supports token passing as a means of managing ring traffic in a busy ring. The two types of ring network supported by the CY233 are shown below:

Case I: Host Ring Network – Host device is in the ring.



Case II: Peer Ring Network – All devices are on the ring.



The CY233 simplifies the communication of digital information between systems and between sections of a system. It provides an easy-to-use interface between standard RS-232 serial devices and parallel TTL compatible devices. Its whole reason for being is to save you time, effort, and money in the solution of information transfer problems.

Guidlines for CY233 Use

A major reason that Local Area Networks have not lived up to their performance lies in the mismatch of Ethernet type systems to the typical personal computer. While these systems support data transfer rates up to 10 MBits/sec, your computer can usually handle about 19.2 KBits/sec comfortably. Special hardware and wiring are required to support these high data rates, adding to the expense, but not solving the basic mismatch problem.

If personal computers can only sustain 19.2 KBaud or so, then why do people try to force 10 MBaud into them? For a variety of reasons, some having to do with the techno-economic evolution of the personal computer. Inter-system transfers began when computers were very expensive, and most user terminals were dumb. The networks were used mostly for file transfers between systems. Technological inertia often tends to keep evolving answers based on problems that have disappeared. This is not to say that file transfers are not important today, only that most computer users do not want files, they want answers. Do you really want a data base or do you want something extracted from the data base? Do you want to send a file to Fred, or do you want to simply open a window on Fred's screen and ask him a question? If you want to ship massive amounts of data, not information, around, then you might be better off investing in bigger pipelines. On the other hand, if you primarily want questions and answers instead of file transfers, then the CY233 will be able to handle your need.

Organization of this Manual

The organization of this manual is generally in terms of increasing complexity. The CY233 pin description is presented, and RS-232-C concepts and definitions are reviewed. Since the appearance of the CY233 depends on which side you hook to, one chapter deals with the serial side and another discusses the parallel side of the CY233. Parallel interfaces and handshakes are described in detail.

Most CY233 applications require a specific message format. These are discussed in section 5. Immediately following this discussion, the only non-message-based mode, the UART mode, is described.

Several sections are devoted to the description of network commands, with serial and parallel commands treated separately. A separate section also discusses CY233 LAN Network commands.

After all of these detailed preliminaries are complete, the basic networks topologies and considerations are treated in detail. The final sections treat networks in order of increasing complexity.

Component Level Interfacing

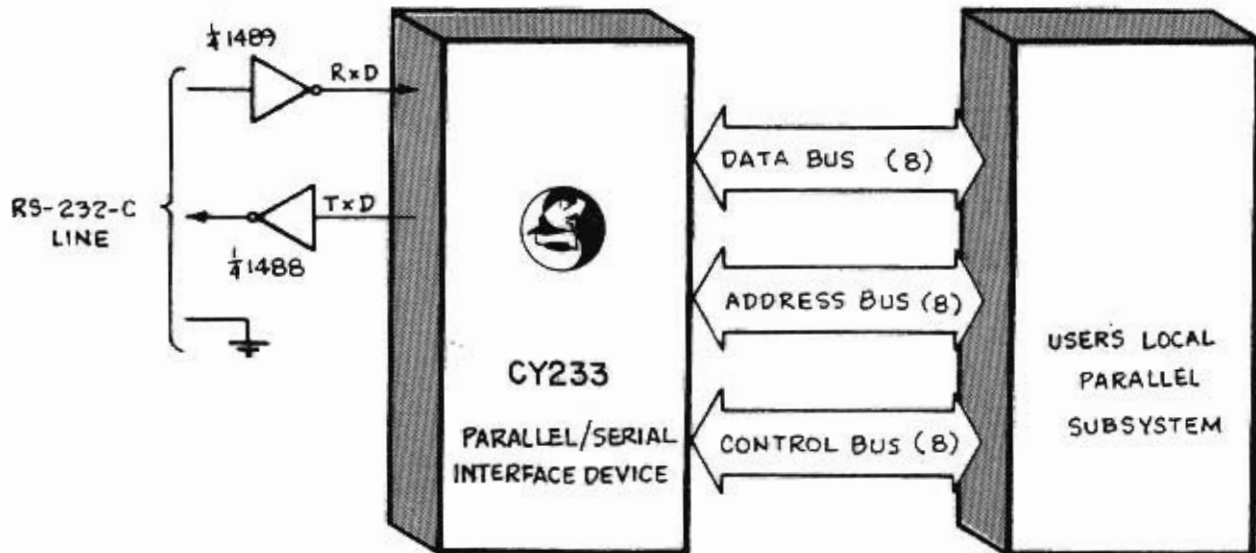


Figure 1.1 Block Diagram of Parallel Subsystem Connected to RS-232 line through CY233.

Information transfer problems come in various disguises. Sometimes they are obviously communications problems, where there is great distance between two systems, and RS-232 serial communication is called for.

Other times the "distance" is not physical, but logical. For example, the interface between a front panel with many switches and lamps, and a single board computer or microprocessor. The computer will normally have several RS-232 ports. One of these can read and control an extremely large number of front panel components, using CY233s. Instead of modifying the computer to add the requisite number of parallel ports, simply use the available RS-232 serial port and add CY233s as required on the front panel. Software then controls the front panel.

There are also problems that are not obviously communications oriented, but rather appear as "wiring" problems. Consider an example where a front panel must control an instrument with digital-to-analog converters (DACs), and other such electronics, but the panel is separated from the instrument by, say, 100 feet. If the number of parallel circuits required is substantial, say, 50, the cable and its connectors will be quite expensive to fabricate in production. As long as the control information doesn't change too often, an RS-232 serial link can prove to be much less expensive. Here there would be CY233s on both ends of the link, placing information on the line at the front panel and taking it off at the instrument.

Logic Diagram

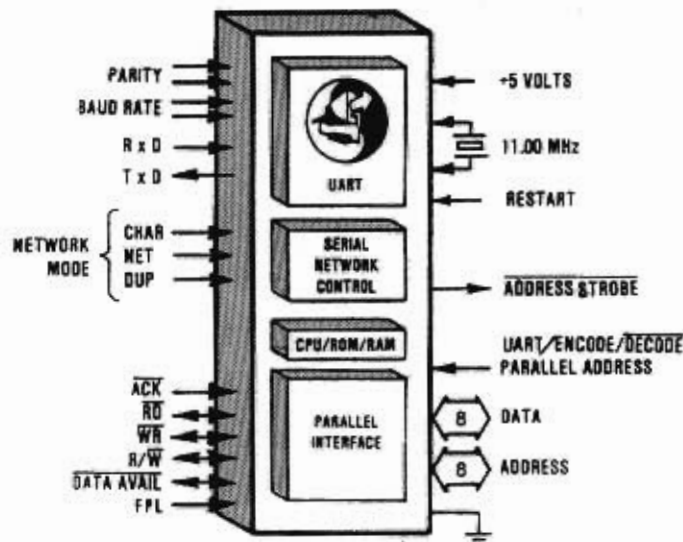


Figure 1.2 CY233 Logic Diagram.

Design Simplicity

The CY233 simplifies the design process in several ways. This allows you to concentrate on bigger issues: what is this system supposed to do and how is it doing it?

The CY233 replaces several LSI devices: a CPU, ROM, RAM, and a UART to interface to an RS-232 line, and multiplexers and decoders in the parallel interface. Since it replaces many devices, printed circuit layout is easier, and you can concentrate on other aspects of the problem.

A more subtle advantage of using the CY233 instead of your own UART circuit is that the CY233 works. Not that your UART will not eventually work, but these circuits can be difficult to debug. The CY233 makes things happen right the first time.

RS-232 brings other advantages to your design. It is a well known standard. Using a standard means there is a lot less explaining to do in describing your system in manuals and on data sheets. Along with the standard goes a level of confidence for the end user.

Personal Computer System for Debugging

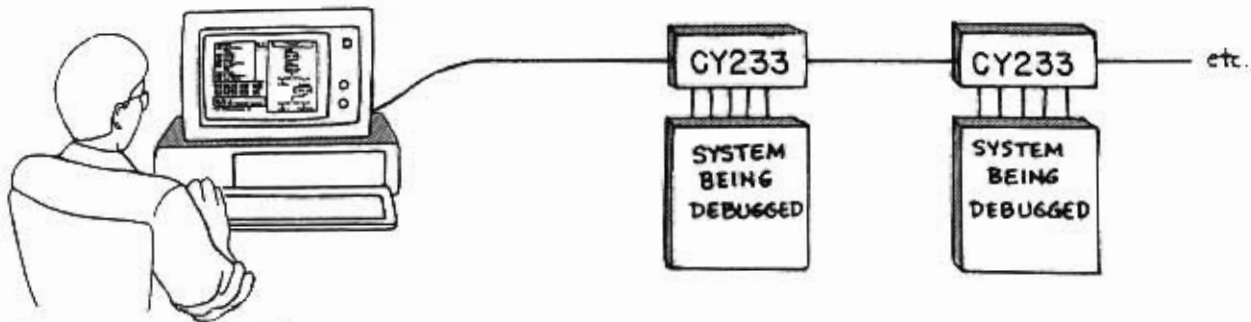


Figure 1.3 Using Personal Computer as Debug Tool.

RS-232 also makes your job as the designer easier, in that you can use RS-232 compatible terminals and printers to control and display what is happening in the system. Consider a case where you are using a CY233 to interface a computer to lights, switches, and some DACs. You can redirect your computer keyboard to your serial output, and debug the operation of the system directly from the keyboard. You can also use an RS-232 compatible printer, or a teleprinter, rather than a CRT, to generate hard copy of your debugging session. In addition, this can be used for production testing.

Also, most inexpensive personal computers, as well as larger systems, contain RS-232 ports as standard equipment. The relatively simple design of CY233 based interfaces, and the networking functions built into the device, allow you to connect many pieces of special equipment through these standard ports. The computer can now monitor and control this equipment without the need for special interface hardware or additional boards within the system. Everything is simply connected through the existing serial port.

Production Economics

The CY233 is an inexpensive device, particularly when you consider what it replaces in the way of other ICs. Beyond the direct replacement of ICs, it saves money by occupying less pc board space, and simplifying board checkout because it represents a distinct subsystem.

The concept of serial communications is economic in its own right, because it leads to a cleaner interface between subsystems. Think about the example in the first section, where a front panel was interfaced to a computer through a CY233, rather than through an added parallel interface to the computer. (This argument is equally valid whether you are designing a microprocessor based CPU, or are using a commercial single board computer or computer box.) What happens when somebody in Marketing wants to add an additional switch, or display? Murphy guarantees you have just run out of spare parallel lines... But, if you have a serial link, no hardware change is needed at the

computer end. At the front panel end, only the parallel side of the CY233 has to be changed; but that is necessary anyway, since this is where the new switch or lamp is actually added. So, only the front panel hardware and the computer software are changed.



Figure 1.4 Replace Expensive Wiring with CY233.

The final aspect of production economics is that, simplistically speaking, computer chips are cheaper than copper. Here we are thinking of the CY233 as a computer, which it actually is inside. When you realize that many cables, even relatively small ones, cost \$1/foot, and connectors often cost \$10/mated pair, it doesn't take much in the way of connectors and cables to cost more than a CY233, or two CY233s. And this is just the material cost. When you consider the advantages afforded by the CY233 in making changes, or expanding the system later, the economic payoff of the CY233 becomes very real.

Flexibility of Networks

The CY233 has one more important use. The applications suggested in the preceding sections could all have been done with conventional parallel TTL electronics, putting aside economics and convenience.

But the CY233 can also be used in networks. Networks are a different breed. The CY233 is unique in that it offers the user a wide range of network functions. In simple applications, these are not needed, and can be ignored. In more sophisticated problems, the CY233 is a powerful tool for implementing a network.

Networks can be thought of as "circuits" in which each of the "components" is a system. Furthermore, to be useful, the circuit must be flexible and allow components to be added or subtracted, without bothering the operation or setup of the other components. The best analogy is the telephone network. Phones are added and deleted each day, without bothering the hookup of the others.

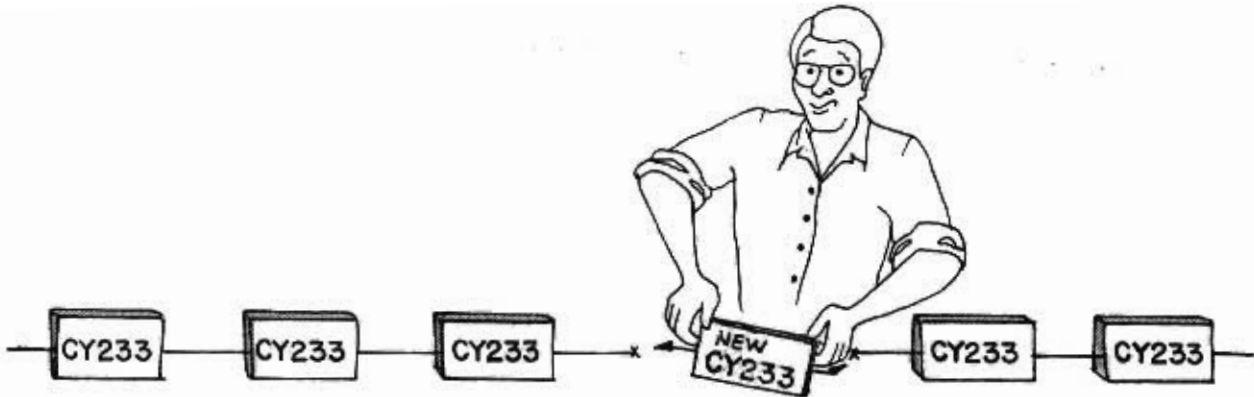


Figure 1.5 Patching CY233 into Network.

The feature of networks which separates them from ordinary electronic circuits is that each of the components has a lot of intelligence. In the telephone system, the central switching office is intelligent, and (presumably) the person using an individual phone is intelligent, in that he dials the right area code, if required, then the number, and then, again, if required, gives an extension. The CY233 has intelligence, and simplifies the implementation of networks.

Please do not confuse the CY233 with large scale networks such as Ethernet. These large scale networks use much higher bandwidths, do more, and cost much, much more. What the CY233 does is bring many advantages of networks to the low cost, low-to-moderate bandwidth RS-232 world.

The CY233 allows many RS-232 devices, itself and other RS-232 compatible devices, to be connected together on the same serial line. Addresses, similar to telephone numbers in concept, allow specific devices to talk to one another. Depending on how the addressing is set up, there are "private" calls or "conference" calls. Judicious choice of addressing allows devices to be added or deleted from the network without disturbing others.

One simple form of network, which is very useful, is the parallel bus. Compared to the conventional computer bus, which might be 8 or 16 bits wide, the RS-232 bus is 1 bit wide. Rather than have separate address and data buses, both are time multiplexed. Also, there is no separate control bus. In terms of conventional bus terminology, an RS-232 bus might be described as one bit wide, character multiplexed address and data, and with asynchronous serial characters.

There are many ways CY233s can be connected in networks. These are described in the latter portions of this manual. The fact that some networks are quite complicated should not deter the first time reader. The CY233 can profitably be used in traditional point-to-point serial communications. These are the

simplest and most obvious applications. It can also be used in simple networks, and it can be used in complex networks. The sophisticated applications can be ignored, with no penalty, if not needed. But they are always there when needed later.

Demonstration System

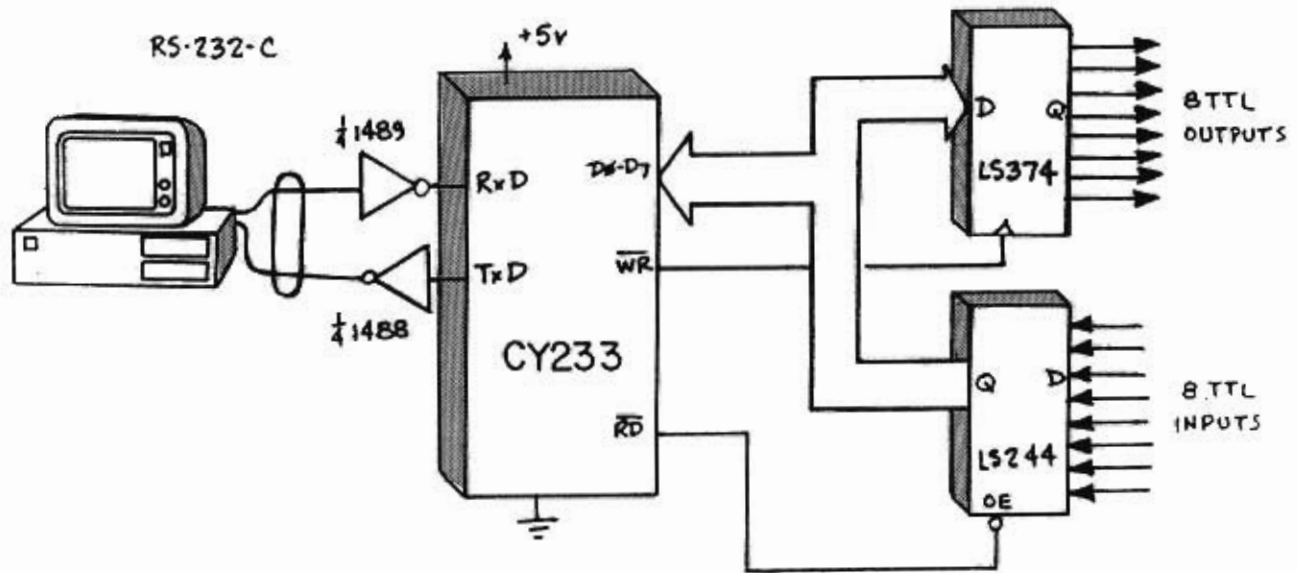


Figure 1.6 This simple system directly translates keyboard inputs to TTL outputs, and TTL inputs to alphanumeric displays.

CY233 Overview and Comparison to the CY232

This section summarizes the major features of the CY233, and compares this device to our older CY232 Network Controller. All features are discussed in detail in later sections of this manual.

The CY233 is an upgrade of our popular CY232 device, with similar functions, and much higher performance. The part takes advantage of newer CMOS technology to achieve the performance gains, at a substantial reduction in power over the CY232. One disadvantage of the newer technology is that the CY233 is not pin compatible with the CY232, so existing CY232 designs cannot be upgraded by simply swapping chips. Cybernetics offers an adaptor board that allows existing designs to upgrade to the CY233, at least in prototype versions, by switching the pins as required for a CY232 layout.

The basic operating modes and characteristics of the CY232 are preserved in the CY233, while adding several new features. The most obvious change is the improved performance. The device can now run to a standard baud rate of 19.2K baud with the standard crystal of 11.059 MHz, 38.4K baud with a non-standard crystal, and a non-standard rate of 57.6K baud with the 11 MHz crystal. There is also a special mode in which the CY233 will adapt its baud rate to that of the network when first powered up.

The character length and parity options have also been expanded, allowing the CY233 to be configured for most systems, without changes to the basic modes of the system itself.

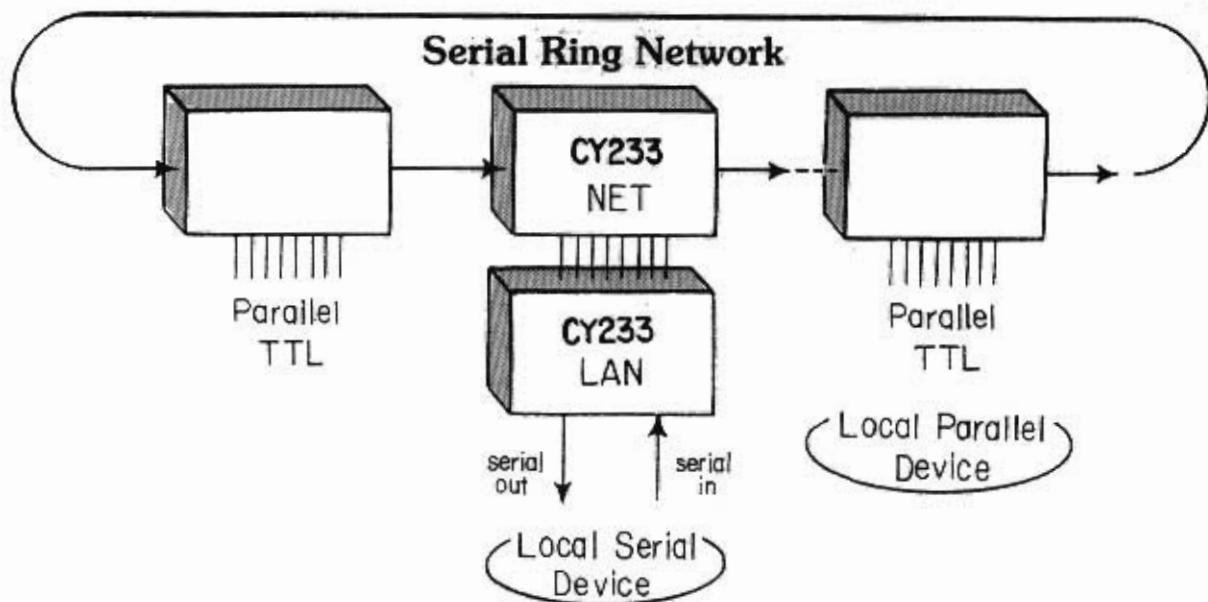
In addition to the R and W commands of the CY232, the CY233 implements several others. Some commands give more options to the way data are transferred between the serial and parallel sides of the device, while others add internal mode control functions to the CY233.

If you have an existing network of CY232s, but wish to add new nodes, using the CY233, this would be possible if only the R and W commands are used. The CY233 can take the place of a CY232 in these cases. Performance would be limited by the CY232 baud rate, but additional features of the parallel side interface might make this a useful upgrade.

Finally, the CY233 implements several basic operating modes. The Network communications mode of the CY232 is preserved, and we expect this to be the most popular use of the CY233. Here the CY233 behaves like a CY232, but with additional commands and higher performance. This mode allows you to connect and control up to 255 CY233s from one serial port, and gives any standard computer the ability to act as an intelligent controller for special external hardware modules.

A new basic operating mode is the UART mode, in which the CY233 acts as a single-chip UART. In this mode, data bytes are transferred between the serial and parallel sides of the chip, without any message structure involved. Baud rate and character format selections are the same as used in the Network mode, but there are no commands, device addresses, or messages. This mode is useful for connecting a serial port to some parallel device, such as an RS-232 port from a computer to a parallel printer. This mode would only support a single CY233 on one serial port, but makes the interface design between the serial port and parallel device very easy and compact. The single CY233 plus line drivers and receivers are the only required parts for such a design.

The last basic operating mode is called the Local Area Network (LAN) mode. In this mode, a pair of CY233s form a node in a network. The various nodes of the network are connected together by one of the CY233s, and this part operates like the CY233 in the standard Network mode. The other CY233 connects to a local serial device, such as a terminal or computer, and allows this serial device to operate in a network of other serial devices. The two CY233s of a node are connected back-to-back on their parallel sides, so the serial sides are available for connection to the network and the local serial device. This scheme allows serial devices to be connected in a Local Area Network, at minimal cost. Most other LAN systems cost many hundred to several thousand dollars per node. While the CY233 LAN does not have all the performance capabilities of an expensive LAN system, it can provide a functional, cost-effective solution in systems that require some communications, but do not justify a high performance LAN design.



Note that even though the LAN mode requires two CY233s, the pair provide the most cost effective Local Area Network mode available today!

Pin Functions

This chapter is concerned with defining the functions of the various pins on the CY233.

Pin Configuration

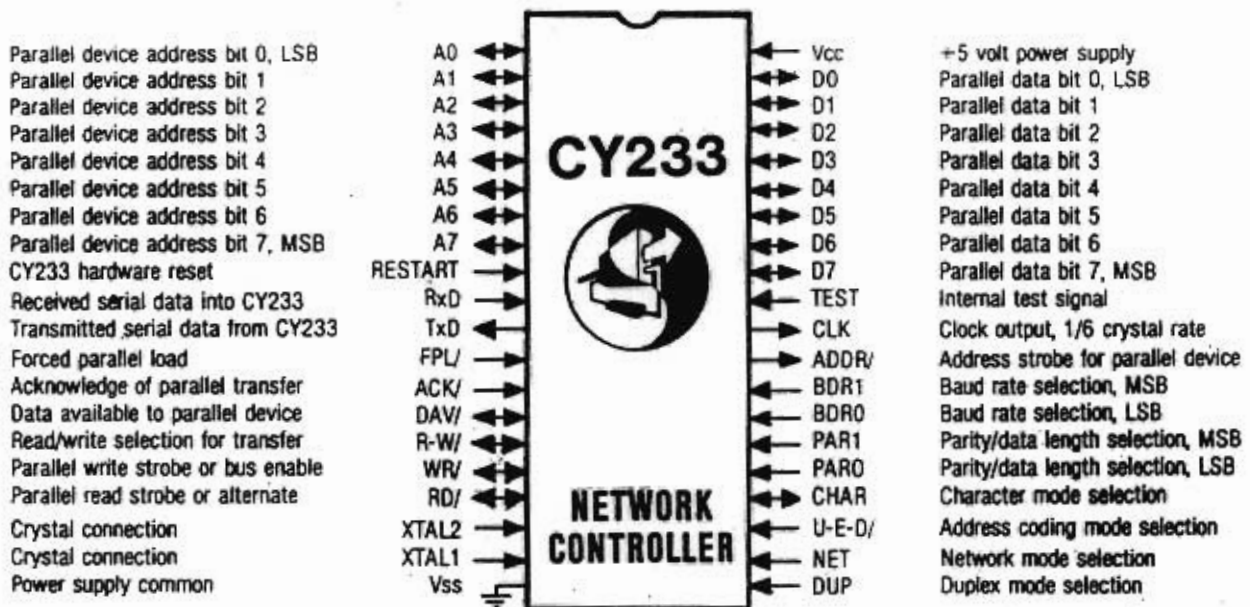


Figure 2.1 Pin Configuration

The pins will be defined in the table on the next page, and then explained in far greater detail on the following pages.

In some cases the control pins seem to allow more states than are normally encoded with control pins. This is because the CY233 allows these pins to be in three states (TTL 0, TTL 1, or left floating, which is called F), rather than the normal two (TTL 0 or 1). Also, many pins on the CY233 are bi-directional, assuming various functions in the different operating modes of the device. Although the number of options may seem confusing, within each mode, the pin functions are more restricted and consistent.

Table I

CY233 Pin Description

Pin	Mnemonic	Function
1	A0	Parallel device address bit 0, LSB
2	A1	Parallel device address bit 1
3	A2	Parallel device address bit 2
4	A3	Parallel device address bit 3
5	A4	Parallel device address bit 4
6	A5	Parallel device address bit 5
7	A6	Parallel device address bit 6
8	A7	Parallel device address bit 7, MSB
9	RESTART	CY233 hardware reset
10	RxD	Received serial data into CY233
11	TxD	Transmitted serial data from CY233
12	FPL/	Forced parallel load
13	ACK/	Acknowledge of parallel transfer
14	DAV/	Data available to parallel device
15	R-W/	Read/Write selection for transfer
16	WR/	Parallel write strobe or Bus Enable
17	RD/	Parallel read strobe or alternate
18	XTAL2	Crystal connection
19	XTAL1	Crystal connection
20	VSS	Power supply common
21	DUP	Duplex mode selection
22	NET	Network mode selection
23	U-E-D/	Address coding mode selection
24	CHAR	Character mode selection
25	PAR0	Parity/Data length selection, LSB
26	PAR1	Parity/Data length selection, MSB
27	BDR0	Baud rate selection, LSB
28	BDR1	Baud rate selection, MSB
29	ADDR/	Address strobe for parallel device
30	CLK	Clock output, 1/6 crystal rate
31	TEST	Internal test signal, connect to Vcc
32	D7	Parallel data bit 7, MSB
33	D6	Parallel data bit 6
34	D5	Parallel data bit 5
35	D4	Parallel data bit 4
36	D3	Parallel data bit 3
37	D2	Parallel data bit 2
38	D1	Parallel data bit 1
39	D0	Parallel data bit 0, LSB
40	VCC	+5 Volt power supply

Description of Signals

The following section describes the pin functions listed on the previous page. Brief remarks about options and selection decisions are included. They are discussed in much greater detail in later sections of this manual. For signal levels, an "F" indicates a floating signal, that can be driven both high and low by the CY233, a "1" indicates a TTL high that cannot be driven low by the CY233, and a "0" indicates a TTL low that cannot be driven high by the CY233. Also, "I" indicates an input only pin, "O" indicates an output only pin, and "I/O" indicates a bi-directional pin. Finally, a name followed by "/", such as RD/, indicates an active low signal.

Mnemonic	Pin #	Description
VCC (I)	40	+5 volt power supply.
VSS (I)	20	Power supply common, logic ground.
XTAL (I)	18,19	Crystal or external clock inputs, signals are not TTL, see Clock Circuits for design details. Standard device operating range is 3.5 to 12 MHz, with frequencies of 11.059 MHz (11 MHz ok) or 7.3728 MHz for standard baud rates.
CLK (O)	30	Crystal clock divided by six. Pulse width is at least 127 nsec.
RESTART (I)	9	Active high. The CY233 will reset all internal operating modes and test I/O line settings when this signal is pulsed. The signal must be active when power is applied. See Reset Circuits for details.
TEST (I)	31	This is an internal test signal that is not used for normal operation. It should be connected to Vcc or logic high.
D0-D7 (I/O)	32-39	Bi-directional data bus for parallel data transfers. These lines are open drain, and must have external 10K ohm pullup resistors connected to Vcc.

Mnemonic	Pin #	Description	continued
RD/ (I/O)	17	<p>Data bytes are transferred between the CY233 and the parallel devices on these lines. Timing and latching functions will differ with each type of operating mode, and is discussed in a later section. Two basic transfer functions are supported, a strobed mode, using RD/ and WR/ strobes, and a handshake mode, using DAV/, ACK/, R-W/, and Bus Enable.</p>	
		<p>Read Strobe/. Strobe for transferring data from the parallel device into the CY233.</p>	
		<p>F Strobed mode enabled</p>	
		<p>If this line is left floating, so the CY233 can drive it, the strobed data transfer mode is enabled. In this mode, the data will be read into the CY233 on the trailing (rising) edge of RD/. This signal may be used to enable a tri-state output driver to put data onto the data bus.</p>	
		<p>0 Strobed mode disabled, WR/ is BUSEN/</p>	
		<p>If RD/ is tied low, the strobed transfer mode is disabled, and WR/ assumes a separate Bus Enable function. A slower logical version of RD/ can be recreated in this mode, by combining DAV/ with R-W/.</p>	
		<p>The CY233 tests RD/ before each parallel transfer is attempted, so it is possible to dynamically switch between the strobed and non-strobed modes.</p>	
		<p>1 Special Local Area Network Mode</p>	
		<p>If RD/ is tied high, the CY233 goes into the Local Area Network (LAN) mode. In this mode, two CY233s are connected back-to-back on their parallel sides. One acts as a normal CY233, connecting to a network of other CY233s, and with all network operations available, while the other acts as a special LAN device.</p>	
		<p>The RD/ line is tested for the LAN mode only at Restart. If the line is not tied high at this time, the CY233 enters the UART or Network mode. These modes are further described in a later section.</p>	

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

WR/ (I/O)	16	Write Strobe/. When the strobed transfer mode is enabled (RD/ floating), this signal indicates that data output by the CY233 is valid. The CY233 drives the data bus just before generating the WR/ strobe signal. Data will be valid on both edges, and is generally latched into the external parallel device on the trailing (rising) edge of the strobe. WR/ is always an output from the CY233 in the strobed transfer mode.	
-----------	----	---	--

When the strobed transfer mode is disabled, this signal becomes the separate Bus Enable input signal, BUSEN/. This signal allows the parallel device to indicate when the CY233 may drive the data bus. If the CY233 has a data byte to transfer, it will generate DAV/, but will not drive the data bus unless BUSEN/ is low. If BUSEN/ is high, the CY233 will wait for BUSEN/ to go low before putting the data on the bus lines. Data will be removed again when BUSEN/ is brought back high. If BUSEN/ is left low, or tied low, data will stay latched on the data bus until another byte is written, or the CY233 performs a data read function. This latching action may extend beyond the duration of a serial message, and is independent of the DAV/ ACK/ handshake function. In this mode, a slower logical version of the WR/ strobe can be recreated by combining DAV/ with R-W/. The BUSEN/ function treats this signal as an input only for the non-strobed mode.

In addition, if enabled by the E (Error Enable) command, the WR/ signal indicates to the local parallel device that an error, including parity, buffer overflow, or timeout, has been detected by the CY233. When enabled, the WR/ line will be driven low, with the R-W/ line high, whenever the CY233 has detected an error condition, and when nothing else is pending at this CY233. Any received message, switch to master mode, or FPL mode operations will remove the error indication until the pending operation is complete.

When an error is indicated, the local parallel device may read an error status byte from the CY233, using a modified handshake protocol. This procedure is described in a later section.

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

Finally, the WR/ line has a special function in the LAN mode. Whenever the local node address should be applied to the Network CY233 of the LAN configuration, the LAN CY233 will drive the WR/ line low. This should enable a tri-state buffer that drives the address lines of both CY233s. When the WR/ line is high, the LAN CY233 will generate the address for the Network CY233, so the externally buffered node address should be tri-stated. The other functions of WR/ are disabled in the LAN mode.

DAV/ (I/O) 14

Data Available/. Handshake signal for controlling data transfers between the CY233 and a parallel device. Normally used in combination with ACK/ and R-W/. If the handshake transfer is not used, DAV/ and ACK/ should be tied together, giving the CY233 an automatic handshake for each transfer. This signal is generated in both the strobed and handshake transfer modes.

When the strobed mode is enabled (RD/ floating), DAV/ has both the data available function and the Bus Enable function combined. In this case, a floating DAV/ will automatically drive the data bus when the CY233 has data to transfer, and remove data from the data bus when DAV/ is removed. A slight modification of the Bus Enable function in this mode is that the CY233 will drive the data bus before generating DAV/ low, so the data will be valid on both edges of DAV/. In fact, DAV/ will be generated shortly after the WR/ strobe occurs. If DAV/ is tied low externally, the CY233 will latch and maintain data on the bus between write transfers, similar to holding Bus Enable low when the strobed transfer mode is disabled. Data will be maintained as long as DAV/ is low, or until the CY233 must perform a read transfer.

When the CY233 is to perform a read transfer in the strobed mode, DAV/ will go low to indicate that a read should occur. The R-W/ signal will be high, indicating a read operation. DAV/ going low signals the parallel device that it may drive the data bus, with data actually being read into the CY233 when the RD/ strobe occurs.

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

For a non-strobed write transfer, DAV/ has only the Data Available function, and WR/ assumes the Bus Enable function. In this mode, DAV/ indicates that the CY233 would like to transfer data to the parallel device, but the data bus is not driven until BUSEN/ is low. DAV/ is removed again when the parallel device acknowledges the transfer on ACK/.

For a non-strobed read transfer, DAV/ also goes low to indicate that the CY233 is requesting a read operation. The parallel device may drive the data bus as long as DAV/ is low. The CY233 will wait for ACK/, then perform the read shortly before bringing DAV/ high again. The BUSEN/ signal is ignored for this operation, and may be high or low during the transfer. If the CY233 had its data bus in the latched output mode, the bus will be changed to high impedance when the CY233 knows a read transfer should occur, before the DAV/ signal is generated.

ACK/ (I)	13	Acknowledge/. Handshake signal for controlling data transfers between the CY233 and a parallel device. This line is normally used in combination with DAV/ and R-W/.	
----------	----	--	--

Acknowledge/. Handshake signal for controlling data transfers between the CY233 and a parallel device. This line is normally used in combination with DAV/ and R-W/.

This line is first tested as part of the parallel address checking function. When the CY233 generates an address at the start of a parallel transfer, this line must be high to allow the transfer to continue. If ACK/ is low, any addresses for this CY233 will be treated as invalid. This test occurs after the CY233 has checked the address lines themselves for validity.

If the address qualifies as valid, the transfer continues, with the CY233 generating the R-W/ signal, driving the data bus, and bringing DAV/ low. Strobed write transfers will also generate the WR/ strobe. The CY233 then waits in this state until the parallel device drives ACK/ low, indicating that the write transfer has been accepted by the device. When ACK/ goes low, the CY233 brings DAV/ high, and removes the data from the bus, unless the latching function is enabled.

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

For read transfers, the CY233 clears the data bus before driving DAV/ low. It then waits for ACK/ to go low, indicating that the parallel device is driving the data lines for the transfer. The CY233 then reads the data, generating the RD/ strobe in the strobed mode, and brings DAV/ high again, indicating that the transfer is complete.

Note that the parallel device can control the duration of the data transfer by the timing of the ACK/ signal. However, the CY233 imposes a maximum time limit on the handshake and will continue with the transfer if the ACK/ signal is not received within the current time limit. The time limit is set to a default value, related to the CY233 baud rate, when the device is restarted. A special command message allows the network to override the default value or to disable the time out function.

If the parallel device only needs the RD/ and WR/ strobes, ACK/ should be tied to DAV/, which allows the CY233 to automatically take care of the handshake protocol. The ACK/ signal has the same function in both the strobed and handshake data transfers.

R-W/ (I/O) 15

Read Write/. This line normally indicates the direction of a data transfer initiated by the CY233. When it is high, the CY233 is performing a read, and when it is low, the CY233 is performing a write. The logical combination of this signal with DAV/ can recreate slower versions of RD/ and WR/ when separate strobes are needed, but the strobed transfer mode is disabled.

When the parallel device initiates a read operation by putting the CY233 in the master mode, or by using the forced parallel load (FPL/) signal, the CY233 will test the state of the R-W/ signal, which may be driven by the parallel device. If R-W/ is high, the CY233 generates a read message for the transfer, and if R-W/ is low, a write message is generated. This second option allows a parallel device to write data to another CY233 module in the network. The state of the R-W/ line is tested during the first data read, when DAV/ and ACK/ are both low.

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

FPL/ (I)	12	Forced Parallel Load/. This signal allows the parallel device to enter a sequence of data bytes to the CY233, without changing it to master mode, and without the CY233 sequencing through the addresses looking for a valid address. The address used in the resulting read or write message is the address being driven onto the address lines when the load request is detected.	
----------	----	---	--

Saving the address testing makes this operation faster than the master mode reads, and allows the parallel device to enter more than one data byte per resulting message.

The FPL mode is initiated by a falling edge transition on the FPL line. This edge will be sensed and recorded in the CY233 whenever it occurs, but the data transfers will only occur when the CY233 is idle, with no serial messages being transmitted or received. This is similar to the state from which the master mode is entered.

With the falling edge detected and the CY233 at idle, a data read will be attempted. The CY233 does not drive the address lines, but the ADDR/ strobe is generated as a signal to the external logic that the FPL address may be applied. A normal data read is then executed, with the regular handshake lines involved. During this first data read, the externally driven address will also be read, and the state of the R-W/ line will be tested. These events occur when both the DAV/ and ACK/ lines are low. A reply message is started from this data, with the R or W determined by the R-W/ line status, and the device address generated from that read by the CY233, unless the address read has all lines high (0FFh). In this case, the CY233 considers the parallel data to be a special parallel command, and takes the requested action. The possible commands are described in the section on CY233 Commands.

If the parallel device holds the FPL/ line low after the first data read, additional data bytes will be read, each with the normal handshake transfer. These bytes will become part of the same reply message started from the first data read.

Mnemonic

Pin #

Description

continued

There are two ways to end an FPL mode read operation. First, if the message terminator byte is read from the parallel device, the FPL mode will end, independent of the state of the FPL/ line. To continue reading bytes in the FPL mode, the parallel device must raise the FPL/ line, then lower it again, causing another falling edge to be detected. This will start a new FPL mode message. This mechanism does not work if the CY233 is in the ASCII HEX character mode. If the CY233 reads a 0Dh pattern in this mode, the data is sent as two hex characters, "0" and "D", and does not terminate the message. Only raising the FPL/ line, the second termination mechanism, will end a HEX mode FPL message.

The second way to end the FPL mode read is to raise the FPL/ line during a data byte transfer. If the last byte read was not the message terminator byte, the CY233 will automatically add a terminator, completing the message to the network. In HEX mode, the terminator is always added to the message, since all data is sent as two HEX data characters.

Note that single byte transfers may be signaled by simply strobing the FPL/ line, since the falling edge is recorded internally by the CY233. When the CY233 is idle, it will then read the data byte, and generate a message around it.

Multiple byte transfers require the parallel device to hold the FPL/ line low until all bytes have been read by the CY233. This mode is ended by reading a terminator or raising the FPL/ line, as discussed above. Note that slow parallel devices, or a large number of data byte transfers in one FPL message, could cause problems in a busy network, by switching a CY233 into an FPL mode read operation, then keeping it in that mode while a serial message is being received from the network. This will eventually overflow the CY233 serial receive buffers, and cause a loss of network data. FPL mode messages should be kept as short as possible in a busy network.

Mnemonic	Pin #	Description	continued
U-E-D/ (I)	23	<p>Uart Encoded Decoded/. Mode selection for parallel device addressing. The state of this line indicates how the CY233 will treat the address line values, as follows:</p> <p>1 Uncoded UART mode, addresses not used F Encoded Address, positive true binary 0 Decoded Address, negative true, 1 of 8</p> <p>The Uncoded UART mode is tested at Restart, and disables all special command and message features of the CY233. In this mode, it behaves as a simple UART, transferring data between the serial and parallel sides. However, the device does distinguish between HEX, and ASCII or Binary character modes, with two serial characters required for one parallel HEX transfer, but only one serial character required for one parallel ASCII or Binary transfer. Also, Master mode and FPL mode will work in the UART mode, with the CY233 generated serial responses containing data only, minus the normal command, address, and terminator portions of a full message.</p> <p>The Encoded and Decoded modes are tested each time the CY233 attempts to generate a new address, so these modes may be switched between messages.</p>	
A0-A7 (I/O)	1-8	<p>Address lines. These lines specify the current address to the parallel device.</p> <p>In the UART mode, the lines are not used, and data are transferred between the serial and parallel sides without any address protocol or message structure. This allows the CY233 to be used in a simple controller application where multiple devices and serial messages are not needed. Internally, the CY233 treats all data bytes as following a valid address, but the address is never generated on the address lines, or included in the messages. Also, message terminators are not needed and will not be generated by the CY233.</p> <p>In Encoded mode, the address lines represent a binary, positive true logic address. Valid addresses are 0 to 254. Address 255 should not be used, since the address lines are driven high between messages.</p>	

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

In Decoded mode, the address lines perform a negative true, linear select function, with addresses 0 to 7 valid. If received messages contain addresses greater than 7, they will be interpreted modulo 8, so address 2 and 10 generate the same address pattern. When no device is being addressed, all address lines will be high. When a message is received, or a parallel transfer is being initiated by the parallel device, one address line will be driven low, corresponding to one of the addresses 0 to 7, and may be used as a device select signal.

In the Encoded and Decoded modes, addresses are tested for validity before a parallel transfer is enabled. Two conditions must be satisfied for an address to be considered valid. First, the CY233 must be able to drive the address lines into the state for the specified address. This is checked by the CY233 writing the desired address to the address lines, generating the ADDR/ strobe, then reading back the actual state of the address lines. This state is compared to the desired address, and the first validity test passes if the two are the same.

Any address lines tied to 1 or 0 may override the pattern which the CY233 is attempting to generate, restricting the possible addresses that CY233 can generate, while any address lines left floating will always match the state the CY233 is attempting to generate.

The second test for address validity occurs after the address lines are checked. When an address pattern is found to be valid, the ACK/ line is also tested, and if this line is low the address is considered invalid, but if the line is high, the address is valid, and the parallel transfer is enabled. The ACK/ line is checked approximately 70 usec after the address lines are tested, allowing external logic some time to set the ACK/ line before the CY233 tests it.

The validity test is made for each message received or sent by the CY233, so external logic may dynamically change the set of valid addresses for any CY233.

Mnemonic	Pin #	Description	continued
ADDR/ (O)	29	Address Strobe/. This strobe is generated after the CY233 has written a new address to lines A0 to A7, but before the lines have been read to see if they compare to the desired address. The strobe may be used to trigger external logic, indicating when an externally applied address may be changed. This change could influence the validity test for the address being generated. If ADDR/ is not needed by an application, it may be left open.	
RxD (I)	10	Received Serial Data. This signal receives serial data bytes from the network into the CY233. This is a TTL level input, with a MARK condition, or a binary 1 value represented as a TTL high level, and a SPACE condition, or a binary 0 value represented as a TTL low level. The standard RS-232 voltages are NOT TTL compatible. For RS-232 interfaces, a MARK is a negative voltage, and a SPACE is a positive voltage. Connecting the CY233 into an RS-232 network requires external line drivers and receivers. The standard receiver circuits will translate the RS-232 voltage levels into the corresponding TTL signals expected by the CY233, including the logical inversion in going from RS-232 to TTL.	
TxD (O)	11	Transmitted Serial Data. This signal sends serial data bytes from the CY233 out to the network. This signal is also at TTL levels, and requires an external driver to translate from TTL to RS-232 voltage levels. Logic levels and timing are the same as for RxD	
BDR0, BDR1 (I)	27-28	Baud Rate Selection. These lines select the baud rate for RxD and TxD. The receiver and transmitter both run at the same rate. The baud rate is set at Restart. One set of baud rates is defined, with the standard operating frequency of 11.059 MHz used for most rates, as shown in the table below. The highest rate, 57600 baud, is not a standard rate at 11.059 MHz, and should	

Mnemonic

Pin #

Description

continued

only be used in CY233 networks that do not use a host system, such as the back-to-back wire saver mode. Note that 11.0 MHz will work in place of the 11.059 MHz crystal, since the error is only 0.6%.

If the CY233 crystal frequency is changed to 7.3728 MHz, the highest rate becomes a standard 38400 baud. Note that the other choices for baud rate will become non-standard rates with these crystals.

When the adaptive mode is chosen, the CY233 will adjust the baud rate to match that being used by the network. This allows a non-standard crystal frequency or a different baud rate to be used, as well as letting the host system dictate baud rates to the network. The internal counter resolution of the CY233 will limit the baud rates to the ranges already supported by the fixed rates of the CY233.

In the adaptive mode, the host system must send two carriage return codes to the CY233 when power is applied or when it is reset, allowing the CY233 to adjust the internal baud rate to match that of the characters received. If the Echo All or Echo Invalid modes are chosen, the CY233 will echo the two carriage returns after both have been received, so that all stations in a ring network may be initialized. The carriage returns must be sent before any normal network communications begins.

The following rates are available from the CY233, using internal hardware as the timing source. The column next to the baud rate is the clock divider used to get that rate.

BDR1,0	Rate	11.059 MHz	Rate	7.3728 MHz
F F		Self Adaptive		
F 1	1	57600	38400	
F 0	3	19200	12800 *	
1 F	6	9600	6400 *	
1 1	12	4800	3200 *	
1 0	24	2400	1600 *	
0 F	48	1200	800 *	
0 1	96	600	400 *	
0 0	192	300	200 *	

* These are non-standard rates at 7.3728 MHz

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

PAR0, PAR1 (I)	25-26	Parity and Data Length selection. These signals define the data parity and length, as shown below. These settings are valid for all character modes, but the Binary mode should use one of the options with 8 data bits to drive all data lines and receive the terminator, 0B3h. Parity and data length are set by the CY233 at Restart.	
-------------------	-------	---	--

When parity is specified, even if Mark or Space, the parity bit must be included in the received characters. While 7 data bits and marking parity is equivalent to 7 data bits, no parity, and two stop bits, it could not be used with 7 data bits, no parity, and one stop bit, since the CY233 expects both the parity bit (always marking) and the stop bit. A host system sending 7 data bits, no parity, and one stop bit, could start the next character before the CY233 was ready for it. Note the total character length is start bit + data bits + parity, if used, + stop bit.

PAR1,0	Parity	Data Length	Total Character Length
F F	Mark	7	10
F 1	Even	7	10
F 0	Odd	7	10
1 F	Space	7	10
1 1	None	8	10
1 0	Space	8	11
0 F	Mark	8	11
0 1	Even	8	11
0 0	Odd	8	11

CHAR (I/O)	24	Character. This signal selects the type of characters used by the serial side to represent the data bytes of the parallel transfer. This character mode is determined at Restart, and each mode is explained below.	
------------	----	---	--

Once the CY233 is operating, this signal becomes an output. The CY233 will attempt to drive it low while transmitting, except if the line is tied high, so the CY233 cannot drive it low. This signal may be used to enable tri-state type drivers, as used in the RS-449 applications, and allows multiple transmitters to share one line, so long as only one transmits at a time.

Mnemonic

Pin #

Description

continued

1 ASCII HEX character mode

Two ASCII characters are used to represent one 8 bit data value. The first character is the high nibble, and the second is the low nibble. Valid characters are 0 to 9 and A to F or a to f. One data byte transfer occurs for every two serial data characters. Invalid characters are ignored in the data portion of received messages, but will be echoed, if echo is enabled. The message format is <command, address, data, terminator>, with the address and data portions being represented by the two character HEX values, but the command and terminator being single ASCII characters. Address 0FFh is reserved, and should not be used. Multiple data values may be sent in one message. The terminator is <cr> (0Dh), and is not written when received.

0 Binary character mode

Each character represents one byte value, and a data transfer occurs for each data byte received. The message format is <address, data, terminator>. All messages are implied write messages, with no explicit commands. The address is represented by a single byte, and address 0B3h is reserved. Multiple data values may be sent in one message. The terminator is 0B3h, and is not written when received. If 0B3h is the first data value, it is not considered a terminator, and is transferred to the parallel side. Any 0B3h value after the first data byte will terminate the message. This mode should use an 8 bit data length serial character, otherwise the terminator cannot be received.

F ASCII Character mode

Each character represents one byte value, and a data transfer occurs for each data byte received. If an 8 bit data length is chosen, this mode can drive all 8 data lines, like the binary mode. Otherwise, the 8th bit will be zero when written and ignored when read. The message format is <command, address, data, terminator>. Commands are single ASCII letters, addresses are represented by two HEX characters, and data bytes may be any value except the terminator. The terminator is

Mnemonic	Pin #	Description	continued
----------	-------	-------------	-----------

<cr>, 0Dh, and is written when received, unless the command specifies not to write the terminator. Address 0FFh is reserved, and should not be used.

Note:

The current implementation of the CY233 will allow address 0FFh in messages, and will test address 0FFh in the Master mode, when sequencing through the possible addresses. However, address 0FFh should not be valid in general, because the CY233 drives the address lines to this state between messages. Also, future implementations may reserve address 0FFh for special functions.

DUP (I)	21	Duplex. This line selects the duplex modes of operation, and along with the NET line, determines the serial operating characteristics of the CY233. The mode is checked dynamically, while the CY233 is not busy with some other operation.
---------	----	---

NET (I)	22	Network. Along with DUP, this signal determines the serial operating protocol of the CY233. This line is also sampled when the CY233 is in a background mode, with no transfers pending, and may be used to switch the CY233 between modes. The most common switch is between the Master and Slave modes of a particular Echo selection, with the modes assigned to allow easy switching, since a Slave mode has a line floating, where a corresponding Master mode has a line zeroed. However, dynamic switching is supported between any of the possible modes.
---------	----	---

The combination of NET and DUP determine the operating mode and echo characteristics of the CY233. Two operating modes and four echo modes are possible, as shown below:

Type	NET	DUP	Echo	Mode
A	0	0	Invalid	Master
B	0	F	None	Master
C	0	1	All	Master
D	F	0	Invalid	Slave
E	F	F	None	Slave
F	F	1	All	Slave
G	1	0	Valid	Master
H	1	F	Valid	Slave
I	1	1	Valid	Slave

Communication Modes

As mentioned in the DUP and NET pin descriptions, the CY233 has two basic communications modes and four echo options. These modes and options are discussed in this section.

In the Slave modes, the CY233 waits for a serial message from the network before initiating a parallel data transfer. The network and controlling computer determine the communications flow and actions of the slave CY233. However, a parallel device can still force a read operation by using the FPL/ signal.

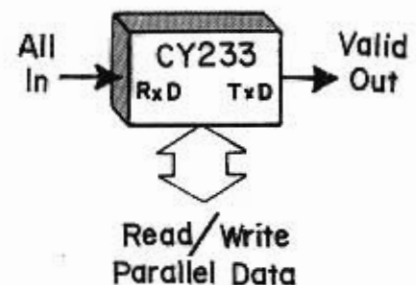
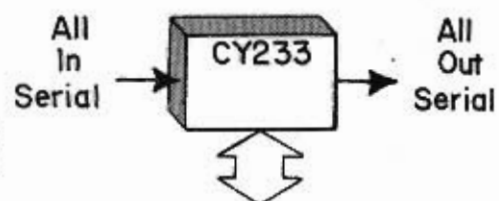
In the Master modes, the CY233 will request reads from the parallel devices when there is no network activity at this CY233. The reads occur from sequential valid addresses, with the CY233 going through a full valid address check before requesting the read operation. Note that for one valid address in the Encoded mode, the CY233 will attempt 255 address tests before coming to the valid address again. One data byte is sent per read operation, so all master mode read messages will contain one data byte. If the parallel device drives the R-W/ line low during a Master mode parallel read, the CY233 will generate a write message around the data just read. Otherwise, the CY233 generates a read message.

The four possible echo modes determine how the CY233 will retransmit characters that are received by the RxD signal. In all cases, echo is disabled between messages, keeping the CY233 from retransmitting random noise received on RxD. A special command may be used to enable echo between messages.

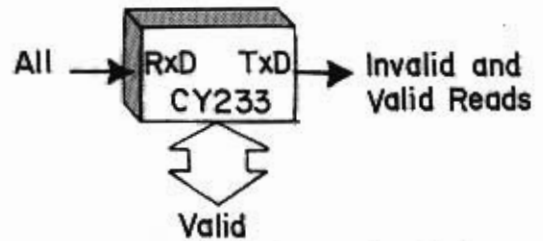
A valid message is one that occurs for a valid address at the CY233 in question. That is, the message contains an address to which the particular CY233 will respond. An invalid message is a properly formed message, but with an address that is not valid at this CY233. In most applications, each CY233 is assigned a unique address.

Echo All will echo valid and invalid messages, adding data characters when a read is requested.

Echo Valid will echo only messages that are valid for this CY233. Invalid messages will not be retransmitted.



Echo Invalid will echo only messages that are invalid for this CY233. Valid messages will cause a parallel transfer to occur, but will not be retransmitted, thus valid messages get off the network when they find the proper CY233. However, a valid Read message received from the network will retransmit the command letter and address, along with the data read and terminator. This feature preserves the full read message format, and keeps read messages from getting lost in a ring network using Echo Invalid. This feature is implemented for all commands that generate a local response, including Read, Display, and Query.



Echo None will not echo any messages, but valid messages will cause parallel data transfers. If a message causes a local response, such as a read message, the CY233 will only transmit the data and terminator, not the command letter and address.

In the UART mode, in which there are no addresses, all data characters are considered valid, and will be echoed for the Echo All and Echo Valid cases, but not for the Echo Invalid and Echo None cases.

Also note that ASCII HEX mode has a filter for hex data characters. Non-hex data characters will be echoed as required, but are not passed on to the parallel device. This feature may be used to embed control or format characters, such as spaces, in a message without passing them to the parallel device. The allowed hex characters are ASCII 0 to 9, A to F, and a to f. Both upper and lower case letters are accepted as valid hex values.

Independence of Operations

The CY233 parallel and serial operations are orthogonal as shown by the waveforms below. There is no necessary timing connection between operations occurring on the parallel side of the CY233 and those occurring on the serial side, although there may be certain precedence relations, i.e., some operations may be held up until others have occurred.

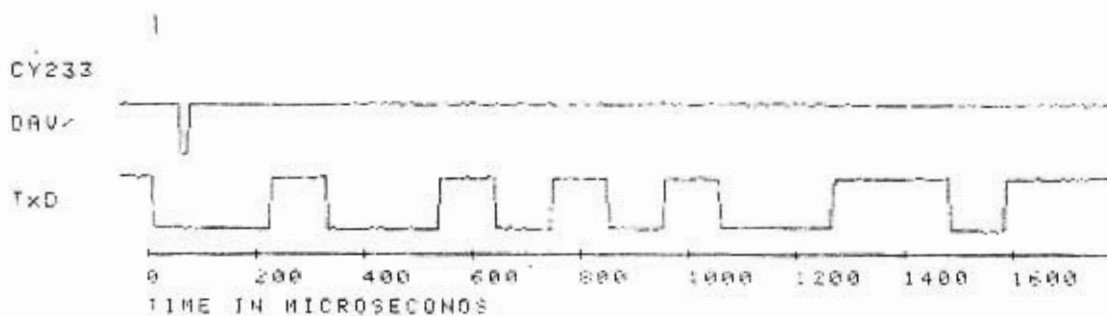


Figure 2.2 Illustrating the asynchronous nature of parallel and serial operations - parallel I/O is being performed in the middle of a serial transmission.

The CY233 is an intelligent interface between a serial network and a user parallel device. It is most convenient to separate the discussion of how the CY233 operates into these two sides: the serial and the parallel. The purpose of the CY233 is to transfer messages between them. A local parallel device may initiate a message and send it out to the serial network, or someone on the serial network may send a message to a parallel device.

Remember the purpose of the serial line is either

to minimize wiring over long distances, or
to have a standardized interface between equipment.

This section is an introduction to serial communication. Details on how to actually hookup a CY233 are left to later sections, which cover various networks and configurations. If the user understands the basics of serial communication and RS-232, the present chapter may be skipped.

Concept of Messages

We use the term message to represent the unit of information transfer from one device to another. The message is the actual unit, or block, of information.

Messages inherently have three components:

an address: (e.g., the telephone number)
data: (What I am saying)
control: (dial tone, ringing, Hello, Goodbye)

The variety of communication protocols arises from different ways of representing these components. Let us assume for the moment that all three have been encoded in TTL electrical signals. There are still fundamentally different ways of arranging the message transfer. For example, we can have

* all parallel: where all bits of all three components exist simultaneously on separate wires, one bit per wire, with a label on each wire identifying its function;

* component serial, bit parallel: where the message is divided into its major components which are presented serially in time, but the individual bits of each component are presented simultaneously, so components are labeled by time, but bits are labeled by wire;

* component serial, bit serial: where everything comes down a single wire, serially in time, and the timing of both the bits and the components is the labeling.

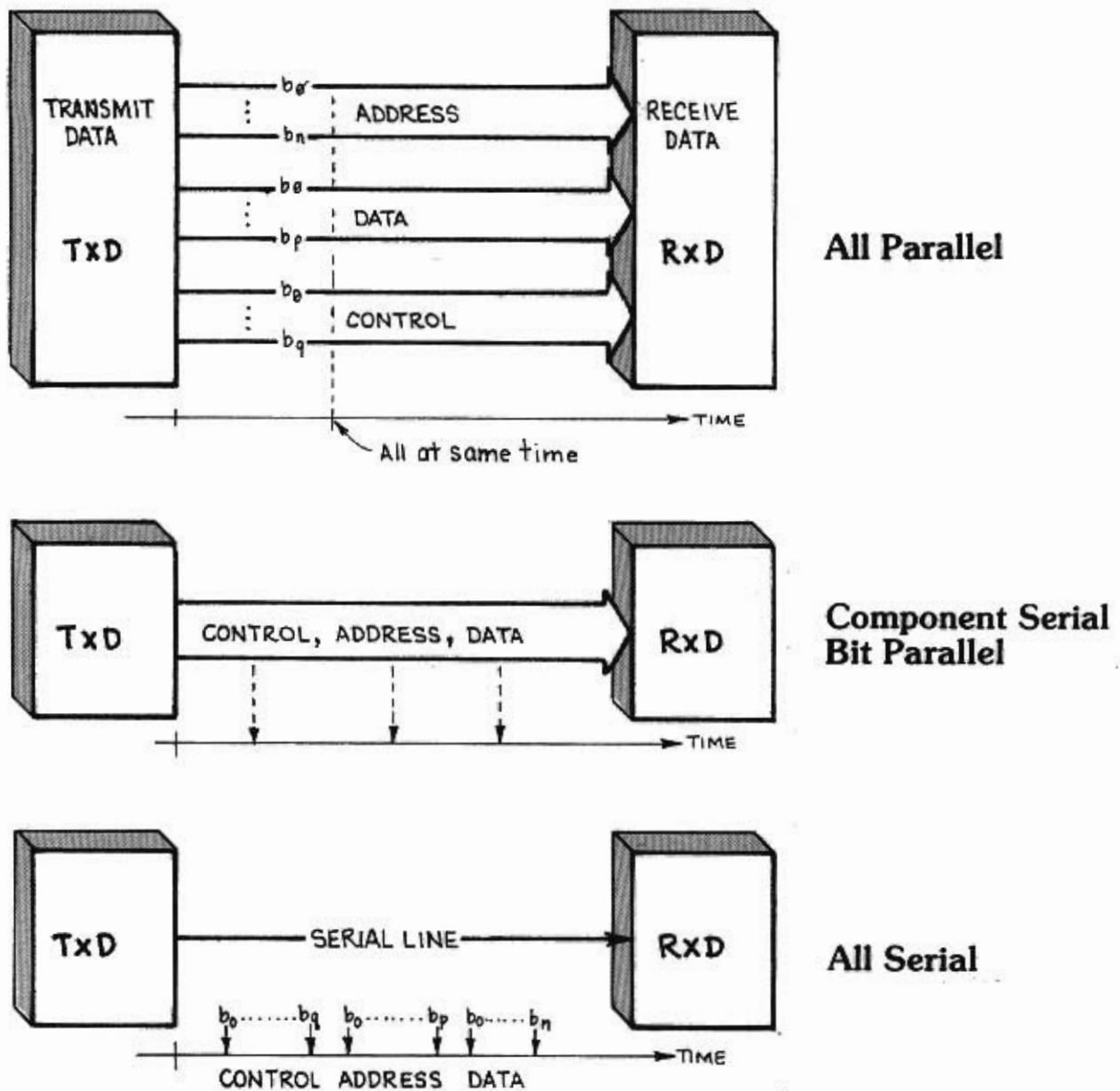


Figure 3.1 Different Message Transfer Schemes.

What we see is a spectrum of possibilities from all parallel, which is time efficient but hardware expensive, to all serial, which is hardware efficient but time expensive.

The serial side of the CY233 is concerned with this last choice, which is very hardware efficient. It may be too slow for some applications, but is sufficiently fast for a great number of others. In fact, with a maximum standard baud rate of 19.2K, the CY233 can sustain a transfer rate that is too fast for many host computer systems!

While RS-232 is an example of all serial (component serial, bit serial), standard microprocessor buses are examples of component serial, bit parallel, and a simple hookup of switches to lamps and relays is an all parallel circuit.

One more important item is how message information is encoded, i.e., what is the alphabet used? Two different alphabets are available within the CY233. The first is binary characters, which can, by themselves, represent one of up to 256 choices. The second is ASCII (American Standard Code for Information Interchange) characters, which are an extension of the English alphabet and Arabic number system we use every day. ASCII is particularly important because most CRT terminals, hard copy printers, etc., use it. This enables computers and these peripheral devices to be easily connected, since they use the same alphabet.

To summarize, messages are subdivided into components. Message components are encoded by characters. Characters are, in turn, made up of 7 or 8 data bits. When using a serial communications protocol,

messages are component serial, components are character serial, characters are bit serial.
--

Standards

Standards come in many different varieties. The purpose of this section is to separate apples from oranges.

Electrical standards say what voltages will represent a "0" or a "1" in a digital system. TTL is a common example. These standards may also say something about short circuit current, frequency range, etc.

Mechanical standards determine connectors and pinouts, and may also say something about operating temperatures, vibration or what have you. Microprocessor buses, for example, have a mechanical standard for card size and connector location.

Information interchange standards have to do with the alphabet used and in what order information is sent. ASCII is an example.

The important issue is what things are included in a descriptive term, such as RS-232, and what are not. RS-232 does not say anything about the alphabet, but does say something about electrical and mechanical standards.

RS-232

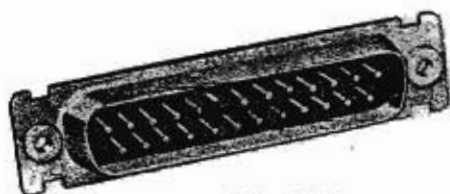
This standard is published by the Electronic Industries Association, Engineering Department, 2001 Eye Street NW, Washington, DC 20006. It is RS-232-C, "Interface between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange." The "-C" is the current revision. It covers voltage levels at the serial line interface, mechanical details about the connector, and pin designations. It does not cover how information is encoded.

RS-232 was originally intended to standardize connections between computer equipment and modems, which are devices for sending and receiving digital information over the telephone network. Therefore, RS-232 contains much detail about establishing and maintaining a satisfactory connection. This is also the reason for the terms "Data Terminal Equipment" (DTE) and "Data Communication Equipment" (DCE) in the standard. Think of DTE as a CRT terminal, and DCE as the modem which connects to the telephone line going to a computer. If no modem is used, then the computer might appear directly as DCE. Sometimes DCE is referred to as the "data set". All of this becomes important in defining connector pins. One has to separate DTE from DCE.

RS-232 establishes voltage and current levels, as an electrical standard. It establishes a pinout for the interface connector, but, interestingly, does not specify the connector itself. However, most computer devices use what is commonly called the "DB-25", which is miniature 25 pin connector. An S suffix is added for the socket, or female, version, and a P for the plug, or male, version. Normally (but there are a few infamous exceptions), the DB-25S is used on chassis equipment and the DB-25P on interconnecting cables. Strictly speaking, RS-232 talks about female connectors being associated with DCE (in a fixed position), and male connectors with extension cables on DTE. So while a manufacturer can choose not to use the DB-25, the pin functions are defined, by number, in RS-232.

Note that some popular personal computers have switched their serial connectors to a "DB-9" style, to eliminate the pins that are normally not used in the DB-25 for a single channel connection. These are not RS-232 ports in the strictest sense, since they do not have 25 pins, and use different pin numbers for specific functions. However, with special adaptor cables, they can be translated into a standard RS-232 connector, with 25 pins.

The voltage standards for RS-232 are bipolar, and are not directly TTL compatible. While these levels are not used elsewhere in common microprocessor circuits, there are inexpensive and readily available IC drivers which meet RS-232. For example, a common driver is the MC1488 and a common receiver is the MC1489. Also, newer circuits incorporate both drivers and receivers in the same chip, with some even running off a single 5 volt supply, and generating the RS-232 levels within the chip! Examples are the Maxim MAX232 and MAX233 devices.



DB-25P



DB-25S

Figure 3.2 DB-25P and DB-25S Illustrations.

The negative RS232 level is between -15V and -3V. The positive level is between +3V and +15V. Common RS232 IC drivers work at +/-12 volts, and their output falls within the allowed ranges.

(Notwithstanding the above, many receivers will function correctly with a negative voltage as positive as 0.4V, allowing TTL circuits to drive RS-232 inputs. This is not recommended, but is sometimes done.)

RS-232 Polarities and Terms

RS-232 Voltage

Description	Negative	Positive
Binary Data	1	0
Condition	Mark	Space
Function	Off	On

"Mark" and "space" are traditional terms which are still used in some situations; for example, mark (1) and space (0) are used to describe parity bits which are added to data characters as checks on the transmission. Notice data 1 is represented by the negative voltage, -15 to -3V, (negative true). Common IC drivers and receivers are inverters, so TTL data presented to the CY233, or any other UART (Universal Asynchronous Receiver Transmitter) is positive true.

In the following illustration of RS-232 waveforms, the start and stop bits are bits which precede and follow the actual data bits in order to separate characters. Notice they are of opposite polarity to one another. Start, Stop, and Parity bits will be discussed in greater detail in later sections.

CY233 Wave Forms

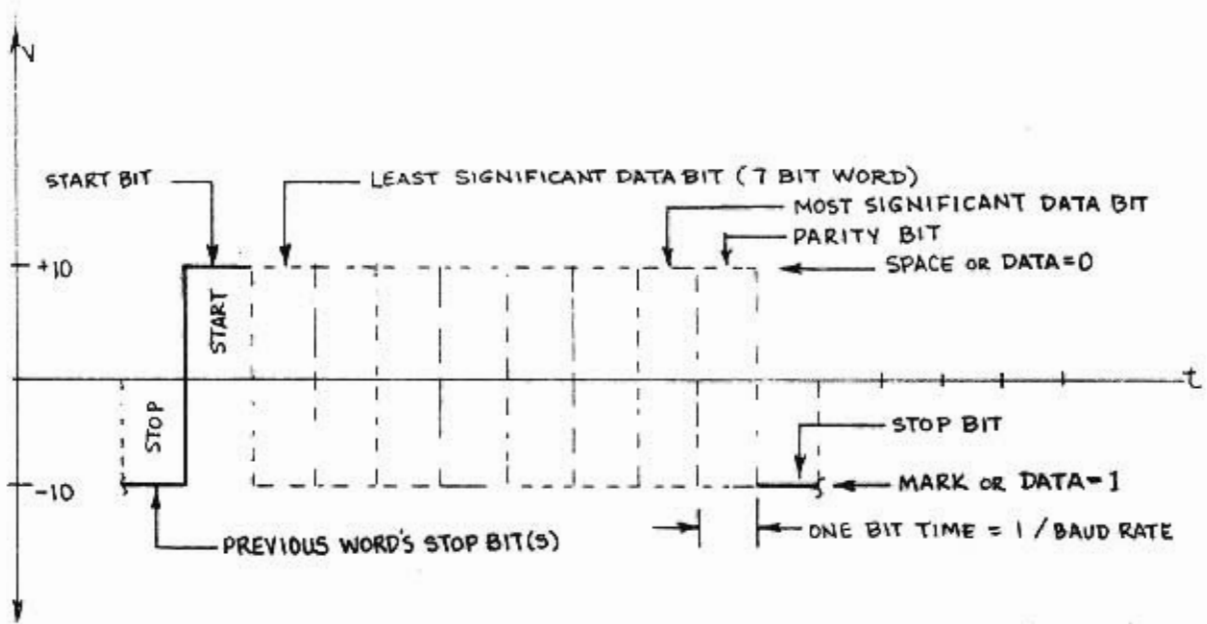
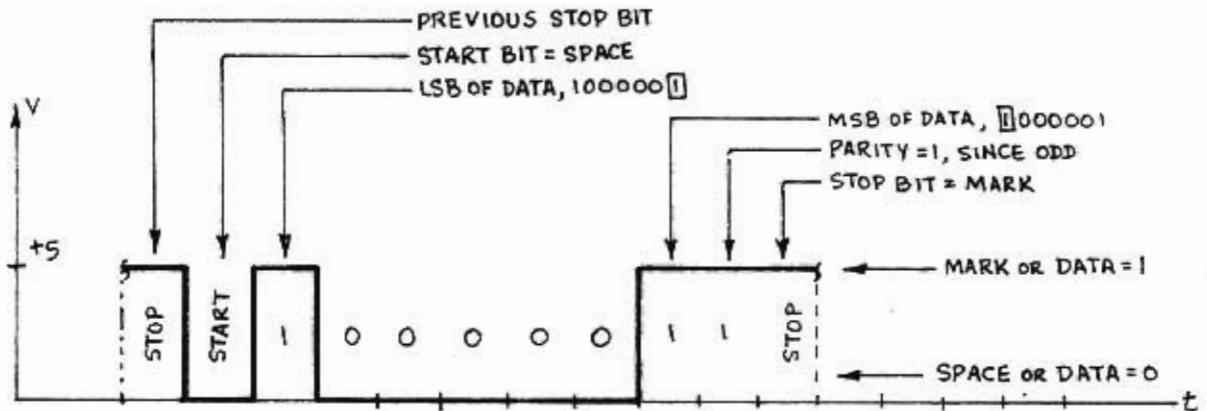


Figure 3.3 RS-232-C Waveforms. Notice there must be inverting driver and receiver between CY233 and RS-232-C Line. (Inverting drivers and receivers are standard.)



Example CY233 TxD Waveform for case of character = ASCII "A" (41 hex) in 7 bit word, odd parity format. Notice TTL levels and inversion from RS-232-C polarity.

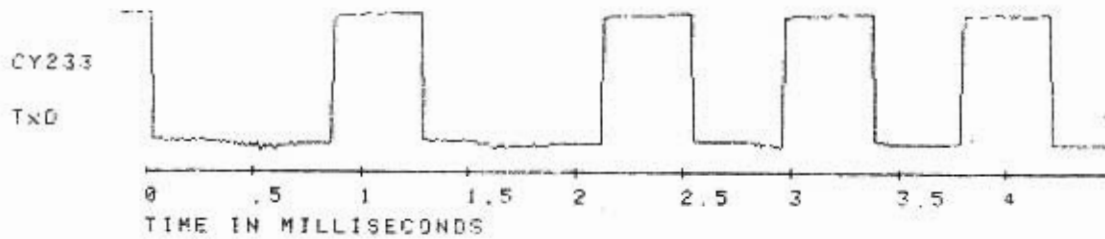


Figure 3.4 CY233 Serial Timing Diagram

RS-232 defines a 25 pin connector for the interface. Only 4 of the pins are of concern to the CY233. These are

RS-232 Connector Pins	
1	Protective ground
2	Transmitted data
3	Received data
7	Signal common

It is at this point that the question of who is the receiver and who is the transmitter becomes important. Since RS-232 is written for the above described data terminal equipment to data set interface, pin 2 carries information from the terminal to the data set and the serial line. Pin 3 brings information from the data set to the terminal. The thing to remember is that one device's transmission is another's reception. If you are connecting a CY233 to a CRT terminal, the terminal will output its data on RS232 pin 2 (TxD), which should ultimately connect to the CY233's pin 10, (RxD). If a DB-25 is used for the CY233, and a one-to-one cable is used between the terminal's DB25 and the CY233's DB-25, then pin 10 on the CY233 connects to pin 2 on the CY233's connector. Otherwise if pin 10 on the CY233 is connected to pin 3 on its DB-25, the cable must cross connect pins 2 and 3 as shown in Figure 3.5. (This is called a "null modem").

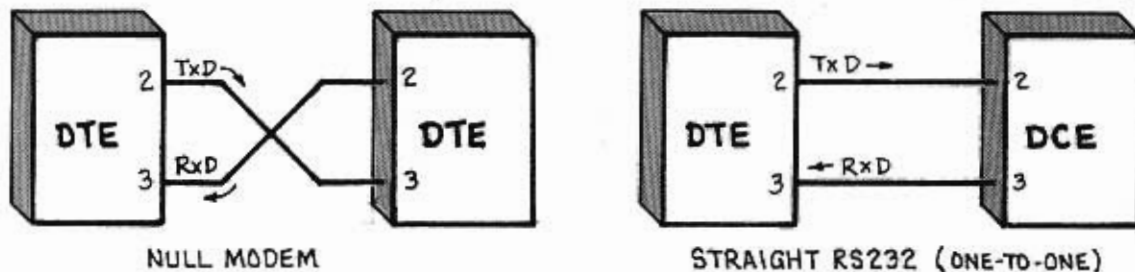


Figure 3.5 Two Ways to Hookup RS-232.

A final important issue is the physical length of the serial line. While RS232 recommends line lengths less than 50 feet, the actual standard allows longer lines if low capacitance cable is used so the total capacitance is less than 2500 picofarads. RS232 is written for baud rates up to 20,000 per second. In simple modulation schemes, like the ones used by the CY233, baud rate and bit rate are equivalent. It is readily apparent that the distortion effects of line capacitance are less important at lower baud rates, thus leading to the possibility of reliable communications over even longer lines. The real limitation then becomes line resistance. In these cases, it is necessary to know the circuit details of the line driver and receiver, to see what will actually work at the desired baud rate.

RS-449

RS449 is a newer standard which is intended to replace RS232 in situations requiring longer line lengths and higher bandwidths. There are two associated standards: RS422 for balanced transmission lines and RS423 for unbalanced transmission lines. RS422 and RS423 use voltage levels which are similar to TTL.

For our purposes, the primary significance of RS422 and RS423 is that IC drivers and receivers exist for these standards and allow the CY233 to be used over extremely long lines, when RS232 might be unsatisfactory. Since RS232 is an electrical and mechanical standard, RS449/422/423 are merely substitutes for specialized applications. The same choice of alphabets and protocols exists for these newer standards as for RS232.

Some devices will say they are RS423 compatible, and "backwards" compatible to RS232. This is because the voltage ranges of RS423 are a subset of RS232.

TTL

TTL is included in this section on serial lines because TTL can be used as the standard for serial ASCII communications using CY233s, and also standard UART's. The idea here is that, within a system, we can use the convenience of standard TTL ICs and voltages, and still have the advantages of minimum wiring or convenient interface to CPUs.

Other Standards

For extra long distances or highest noise immunity there are two additional possibilities. Current loop connections have been around for a long time, and circuits can readily be derived from the TTL levels of the CY233. A newer option is an optical link, using fiber optic cables. This provides the ultimate in isolation between systems, and is inherently serial in nature.

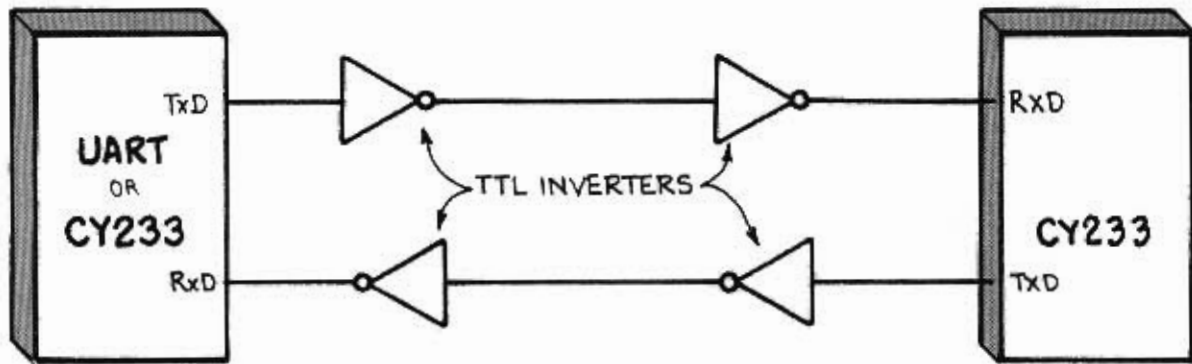


Figure 3.6 Serial Communications Using TTL

Since this is no longer RS-232, ordinary CRT terminals and printers cannot be used, unless they are patched in.

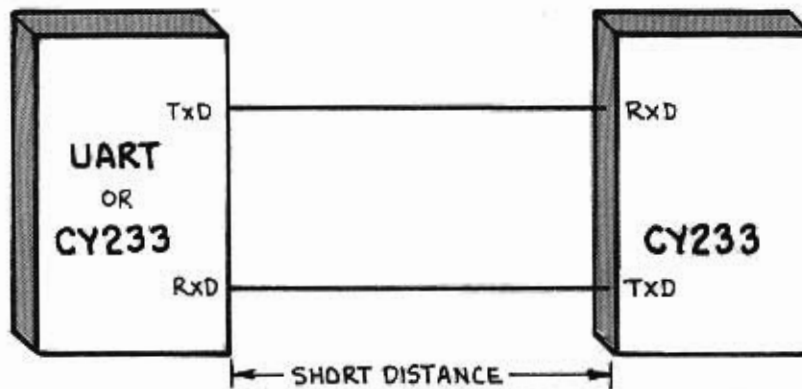


Figure 3.7 Direct Serial I/O between between MOS Chips.

This idea can be extended, for systems physically contained on a small printed circuit board, to have direct connections between MOS chips. This is limited to very small distances because of the limited electrical drive from MOS.

This section is concerned with connecting the user's circuitry to the CY233. The user's devices are called parallel to distinguish them from the serial, RS-232, side. In addition, they often will be parallel, as in switches or lamps. But they may also be LSI peripheral chips which the user wishes to connect to a serial line.

The Three Buses: Overview

The various CY233 pins concerned with the parallel side can be partitioned into three groups, each group being called a "bus":

- * address bus: which device do I wish to connect to
- * data bus: the actual information to transfer
- * control bus: the lines which time the transfer

Any type of parallel bus, including microprocessor buses and minicomputer backplanes, can be subdivided in this fashion; so the concept is quite general. In the language of the previous chapter, this side is all parallel.

The Address Bus

The address bus consists of 8 pins.

Mnemonic	Pin	Function
A0	1	Address bit 0, parallel device address LSB
A1	2	Address bit 1, parallel device address
A2	3	Address bit 2, parallel device address
A3	4	Address bit 3, parallel device address
A4	5	Address bit 4, parallel device address
A5	6	Address bit 5, parallel device address
A6	7	Address bit 6, parallel device address
A7	8	Address bit 7, parallel device address MSB

The address lines are both input and output. They are output in the sense they indicate the address of the parallel device the CY233 wishes to transfer information with. However, they are also input in the sense the user may program certain addresses to be skipped, or ignored, and thereby establish the set of addresses that are valid.

The user programs addresses by tying selected address lines high ("1") or low ("0"). The CY233 always tests the proposed address before making a transfer with the parallel device. This is done by it driving the address lines with the proposed address (i.e., writing the address out with latched drivers), and then subsequently reading the address present. If the user has bolted any address lines high or low, then the address read back may be different from that written out, since the CY233 output drivers can only supply a milliamp or so current. If the read back address is different from the written out address, then the address is considered invalid and no transfer takes place. If the proposed address is the same as what the user has bolted in place, then the read back address is still the same and the transfer takes place. Similarly, if the address lines were left free, so the CY233 can drive them both high and low, then the "read back" will match the "written out", and the address is considered valid.

Each time the CY233 writes out a proposed address, the user circuitry has a brief time (a few microseconds) to make a dynamic decision as whether to allow or disallow the transfer by driving the address lines. The CY233 generates a strobe on ADDR/ to indicate that it has written a new address on the address lines, then waits about two microseconds before reading the address back. (A better way of allowing or disallowing a transfer dynamically is to drive the ACK/ line, in which case the user has 70 microseconds to make a decision).

If the user wishes to drive an address line high dynamically, i.e., from active circuitry, he should be able to source 45 mA for a "1" and sink 1 mA for a "0".

The address represented on the address lines can be either encoded or decoded, as selected by the U-E-D/ line. Systems may dynamically switch between encoded and decoded addresses.

Encoded is the same as straight positive true binary, and addresses 0-255 decimal can be represented. In the encoded mode, for example, 0001 0000 would address device number 16.

Note that when an invalid address is encountered, the CY233 will take all address lines high (i.e., remove the last address). Valid addresses stay latched throughout the duration of a message.

Encoded Parallel Addressing

(positive true binary)

Any address lines tied high or low, and thus not free to toggle, limit the addresses the CY233 will consider.

All lines A0-A7 are free to toggle			Tie A2-A7 low, & the address set will be		
A7	A0	DEV	A7	A0	DEV
0000	0000	0	0000	0000	0
0000	0001	1	0000	0001	1
0000	0010	2	0000	0010	2
	:	:	0000	0011	3
1111	1111	255			

Decoded Parallel Addressing

(negative true linear select)

Any lines tied high will not be considered. For example:

All lines A0-A7 are free to toggle			Tie A2-A7 high & the address set will be		
A7	A0	DEV	A7	A0	DEV
1111	1111	NONE	1111	1110	0
1111	1110	0	1111	1101	1
1111	1101	1			
1111	1011	2			
1111	0111	3			
1110	1111	4			
1101	1111	5			
1011	1111	6			
0111	1111	7			

Decoded is also called 1-of-8, or linear select. When decoded, devices 0-7 (decimal) may be addressed, with negative true logic. In this mode, one of the eight address lines will be taken low as a device select, and the other seven will remain high. As an example of the decoded mode, address lines 1111 1110 will select device #0 and 1110 1111 will select device #4. These addresses are represented in messages as 0 and 4, respectively.

Messages received by a CY233 in the decoded mode are interpreted modulo 8. Thus, message addresses of 7 and 15 both address the same device, namely, #7.

As examples of programming addresses to be valid or invalid, if encoded address lines are programmed 0000 XXX1, only devices <1, 3, ..., 15> will be considered. If decoded address lines are programmed to 1111 XXX1, only devices <1, 2, and 3> will be considered.

Again, since the lines which are tied and those which are left floating may be changed by external logic, addressing can be dynamic. In particular, it is possible to implement "interrupt driven" systems, where data is sent only where the source commands it.

In general, address 255 should not be used. In the binary protocol, B3h (179 decimal) is used as the message terminator, and in all the protocols, the CY233 removes the current address at the end of a message, and drives the lines to FFh. In this

state, all parallel devices should be deselected. Current CY233s will support a message address of FFh, but this address may be reserved for special functions in future versions.

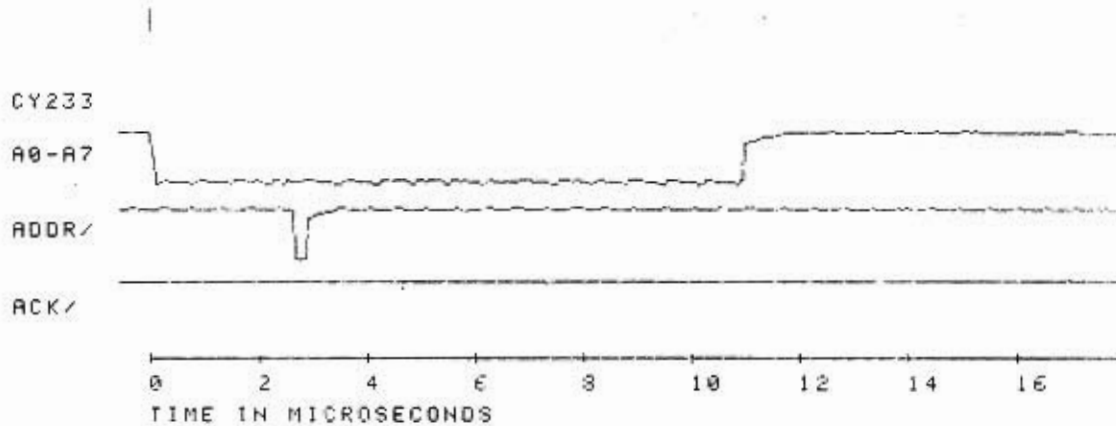


Figure 4.1a Local Address Test with address lines not matching proposed address (Invalid Address)

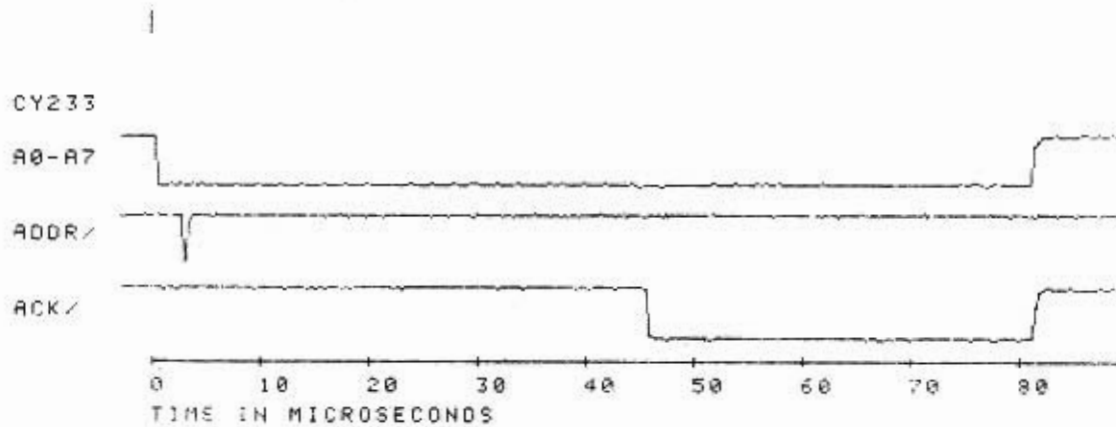


Figure 4.1b Local Address Test with ACK/ low (Invalid Address)

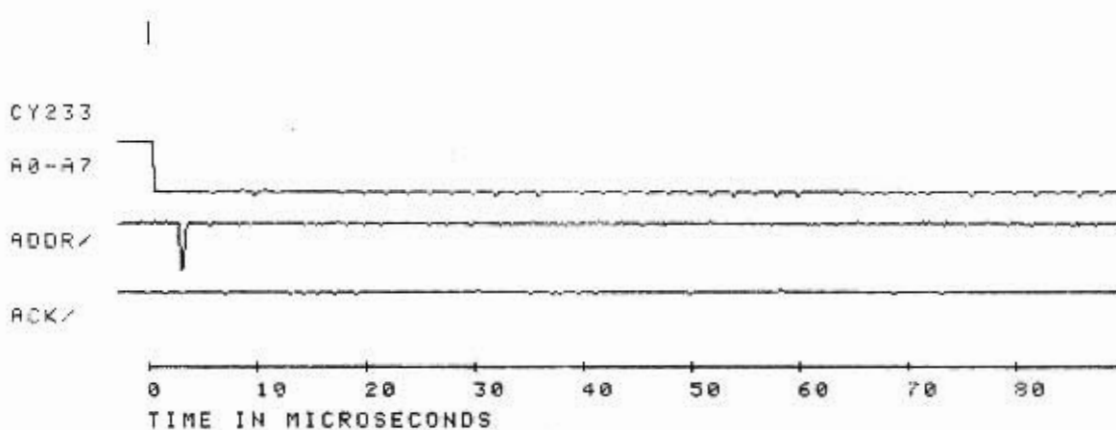


Figure 4.1c Local Address Test with address matching and ACK/ high (Valid Address)

The Data Bus

The data bus consists of 8 pins:

Mnemonic	Pin	Function
D0	39	Data bit 0, parallel data I/O LSB
D1	38	Data bit 1, parallel data I/O
D2	37	Data bit 2, parallel data I/O
D3	36	Data bit 3, parallel data I/O
D4	35	Data bit 4, parallel data I/O
D5	34	Data bit 5, parallel data I/O
D6	33	Data bit 6, parallel data I/O
D7	32	Data bit 7, parallel data I/O MSB

The data lines are both input and output, depending on the direction of information transfer between the parallel devices and the CY233. In contrast to the other CY233 pins, they have no "other" meanings, depending on how the user connects them. It is simply data, one way or the other. Note that the lines are open drain, and require external 10K ohm pullup resistors, connected to Vcc, when used as outputs.

The only choice you have about the data bus is whether it is "latched" or "multiplexed". Latched means the data remains present after the control strobes are complete. In this mode, for example, the bus could drive a display lamp. Multiplexed means the data is present only during the several microseconds or so duration of the control handshake. This is the mode necessary when many sources are connected to a local parallel bus. Using a multiplexed parallel bus requires that all destinations for data contain their own data holding latches. The latched mode is selected by tying the DAV/ control line low in the strobed transfer mode, or by tying the WR/ (BUSEN/) control line low in the non-strobed transfer mode. Otherwise, the CY233 multiplexes the data bus.

In the strobed transfer mode, data is written by the CY233 just before the WR/ control strobe goes from high to low. Data is read into CY233 just before the RD/ control strobe goes from low to high. In the non-strobed mode, the handshake lines control the data transfers, with timing similar to the strobed cases.

Continuing the discussion of latched versus multiplexed data bus operation, the data lines may be latched after a transfer, or they may be returned to a high impedance state, at the user's discretion. The function is controlled by the DAV/ line (acting as the BUSEN/ signal) in the strobed transfers, and the WR/ line (separate BUSEN/) in the non-strobed transfers. If the data bus is multiplexed, it returns to a high impedance after the ACK/ control line is pulled low. Irrespective of BUSEN/, a read leaves the data bus in a high impedance state. It will exhibit transient behavior in this transition from latched write to read, or multiplexed write to high impedance.

The Control Bus

The control bus consists of 8 pins:

Mnemonic Pin	Function	
FPL/	12	Forced parallel load from parallel device
ACK/	13	Acknowledge of transfer from parallel device
DAV/	14	Data available to parallel device
R-W/	15	Read/write selection for parallel device
WR/	16	Write strobe or separate BUSEN/
RD/	17	Read strobe or alternate mode selection
U-E-D/	23	Address coding for parallel device
ADDR/	29	Address strobe for parallel device

ACK/ is an active low input control line. First, it must be high during the address test, before the CY233 will take DAV/ low to strobe read or write to parallel devices. By pulling ACK/ low during the address check, the parallel device will prevent the initiation of any read or write operations. The internal logic treats this situation as if all addresses were invalid at this CY233. The user has approximately 70 microseconds after the new address, before ACK/ is checked for a high condition.

Once the address check passes, the CY233 will proceed with the data transfer. The DAV/ signal indicates that the CY233 has data for the parallel device, or is waiting for data from the parallel device. When DAV/ is low, the parallel device should pull ACK/ low to complete the transfer, indicating to the CY233 that the desired write or read has been completed.

In summary, when a parallel read or write is needed:

1. the address lines are checked for validity, with an ADDR/ strobe generated just before the test,
2. and then if the address is valid,
3. ACK/ is checked for 1, after a 70 usec delay. If ACK/ is high, the transfer is made; after DAV/ goes low, ACK/ is taken low by the parallel device, to acknowledge the transfer.

Notice ACK/ both allows the initiation of a parallel transfer, by being high, and then concludes the transfer, by being pulled low by the parallel device. After DAV/ goes low, the parallel device may extend the duration of the data transfer by its delaying taking ACK/ low. However, this extension is limited by an internal CY233 timer function. The time value default is set to one bit time longer than the serial character time at the chosen baud rate, except at rates greater than 9600 baud, where the 9600 baud time is used. A special command message may be used to modify the time out value. If timeout occurs, the CY233 completes the transaction, as if ACK/ = 0.

If the you desire the multiplexed bus mode, and need no dynamic inhibition nor extension of parallel transfers, simply tie ACK/ to DAV/, providing an automatic handshake.

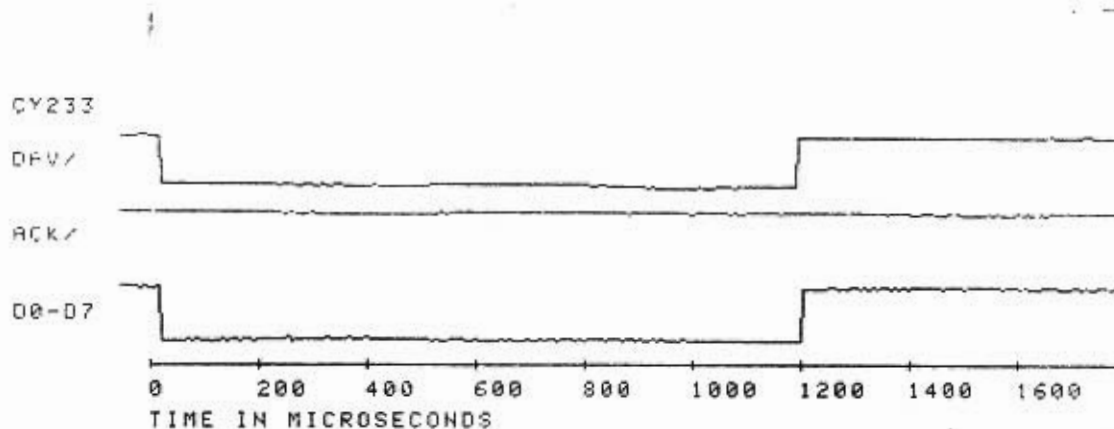


Figure 4.2 Timeout of write handshake

DAV/ is an active low control input/output line. It is a handshake or strobe signal for data transfer between the CY233 and parallel devices. When used as an output, DAV/ can serve as a strobe for an external latch during write operations. Write data is valid all during the time DAV/ is low. Again, being used as an output, during read operations DAV/ going low indicates the parallel device should place its data on the bus. The actual read takes place on the rising edge of RD/.

Alternatively, DAV/ may be used as the BUSEN/ (bus enable) input to control the data bus latching, in the strobed transfer mode. The latched mode is selected by pulling DAV/ low. DAV/ is sampled for this condition after each parallel device transfer sequence. The CY233 continues to monitor the state of DAV/ (as BUSEN/) while waiting for additional data to transfer, and will keep the output latched until DAV/ is raised. When DAV/ is used as an input, the RD/ and WR/ strobes may control the flow of information between the CY233 and the parallel device. Note that the ACK/ line must still be driven with a proper handshake signal in this mode, unless the CY233 timeout function is used. For latched output data, we recommend using the non-strobed transfer mode, where the BUSEN/ signal is separated from DAV/, allowing DAV/ to still be used for the handshake functions.

R-W/ is an active low control output line. It is an early indicator that a write operation is being contemplated by the CY233. If the next proposed transfer is a write, R-W/ will pull low prior to the new address. If the transfer is to be a read, it remains high. The line will maintain the proper state for the duration of a transfer message. At the end of the message, the R-W/ line will be driven high again.

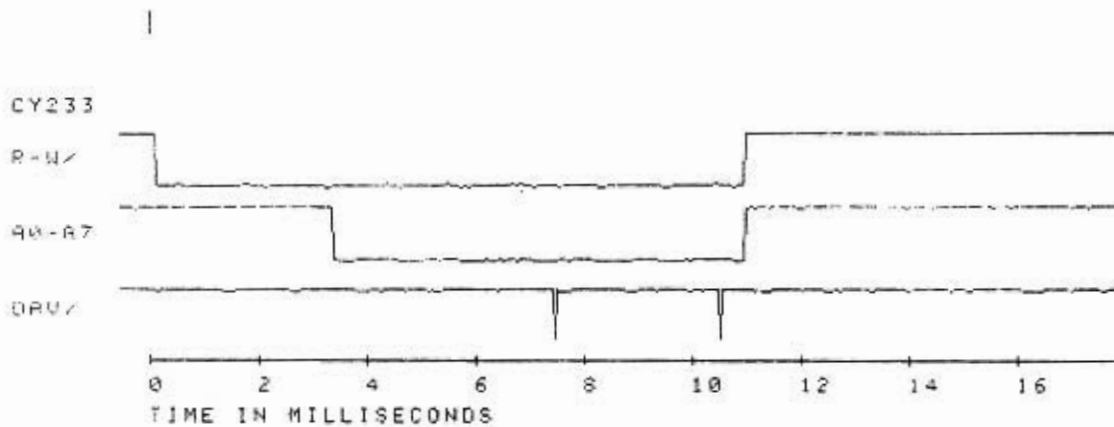


Figure 4.3 Timing for a complete parallel write message

When data reads are being initiated by the parallel device, either through the FPL/ line, or the Master mode, the R-W/ line is used as an input. If the line is high, the resulting message generated by the CY233 will be a read message, and if the line is low, the resulting message will be a write message. This feature allows intelligent parallel devices to generate write messages for other CY233s in a network, without requiring a host computer at the network level.

U-E-D/ determines the mode selection for parallel device addressing. If the UART mode is chosen, by tying U-E-D/ high, the CY233 operates in a mode that does not use the address functions at all. Data transfers occur without messages or address testing. This mode is checked when the CY233 is reset.

In the message based communications, the U-E-D/ pin is used to select between the encoded and decoded address formats. This pin is sampled before each new address check, so it may be changed dynamically, if desired. A high causes a positive true binary representation, and a low causes negative true, 1 of 8, device selection.

If the RD/ pin is left floating, the strobed data transfer mode is used, where RD/ and WR/ are strobe signals indicating exact timing of the data transfers. DAV/ assumes the BUSEN/ function in this mode, as explained previously, and may be used to latch data written by the CY233. In the strobed mode, data is read by the CY233 just before the trailing (rising) edge of RD/. This signal may be used to enable an addressed tri-state buffer to drive the CY233 data lines for the read function.

The WR/ line is also used during data transfers. In the strobed mode, WR/ indicates when the CY233 has write data for the parallel device. The strobe is generated just after the data is output on the bus, and before the DAV/ signal is generated. For devices that do not require a handshake, the WR/ strobe may be used to clock data into the parts.

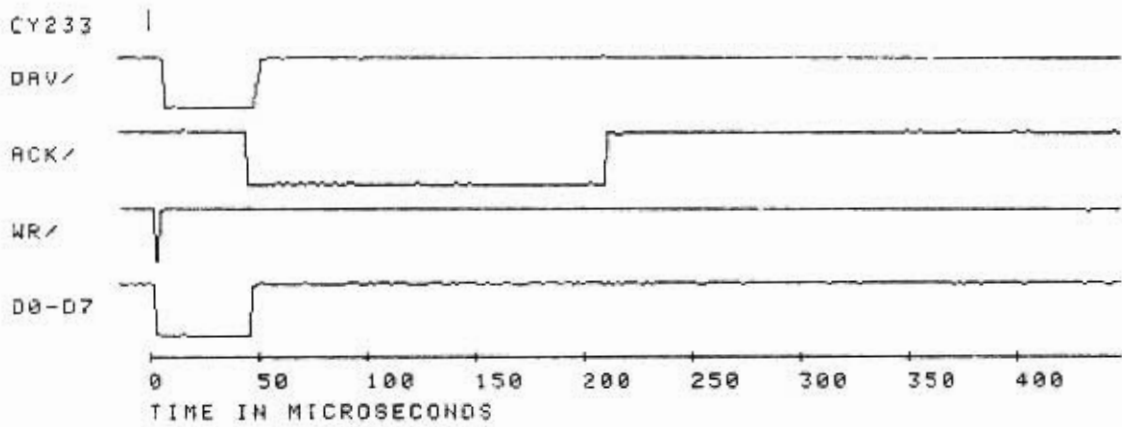


Figure 4.4 Strobed mode write transfer handshake

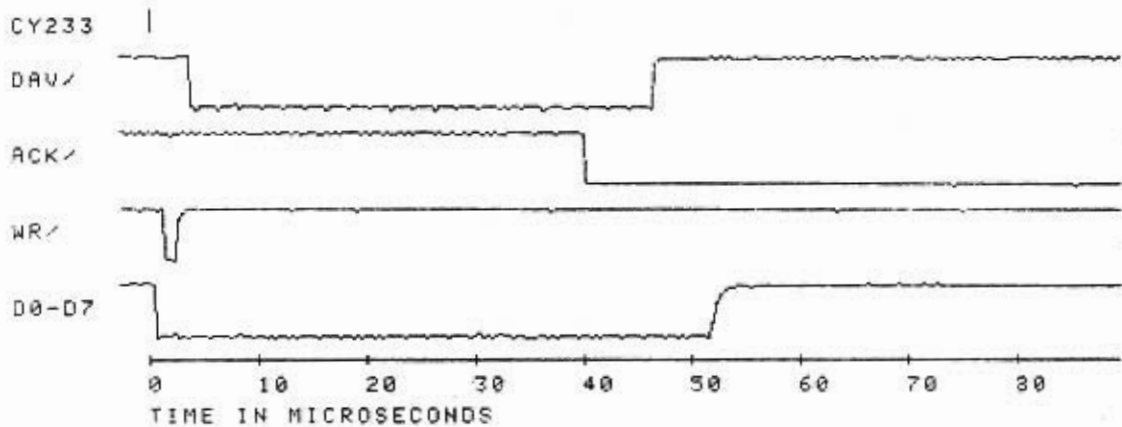


Figure 4.5 Strobed mode write timing

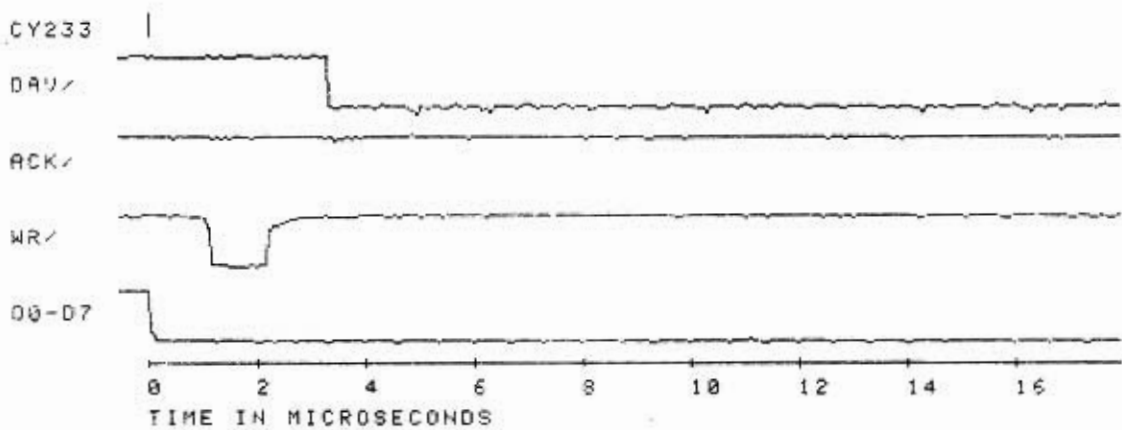


Figure 4.6 Strobed mode write timing details

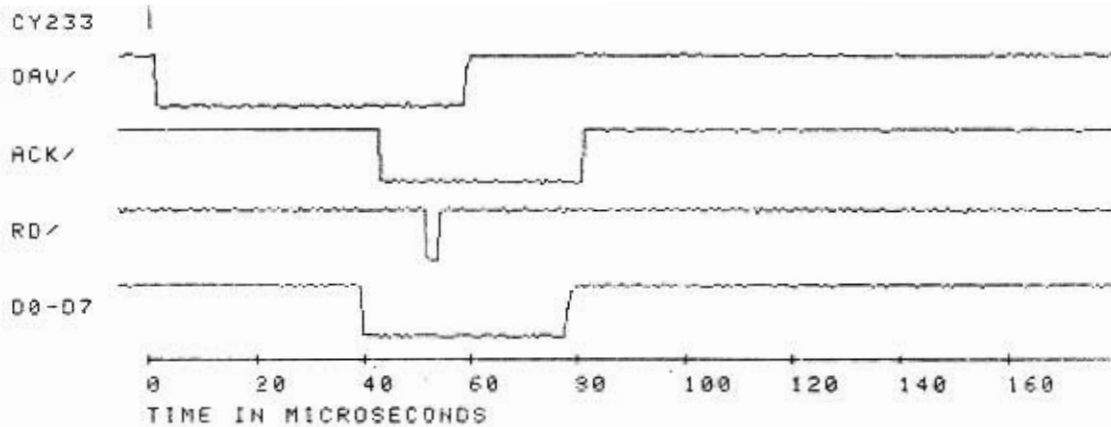


Figure 4.7 Strobed mode read transfer handshake

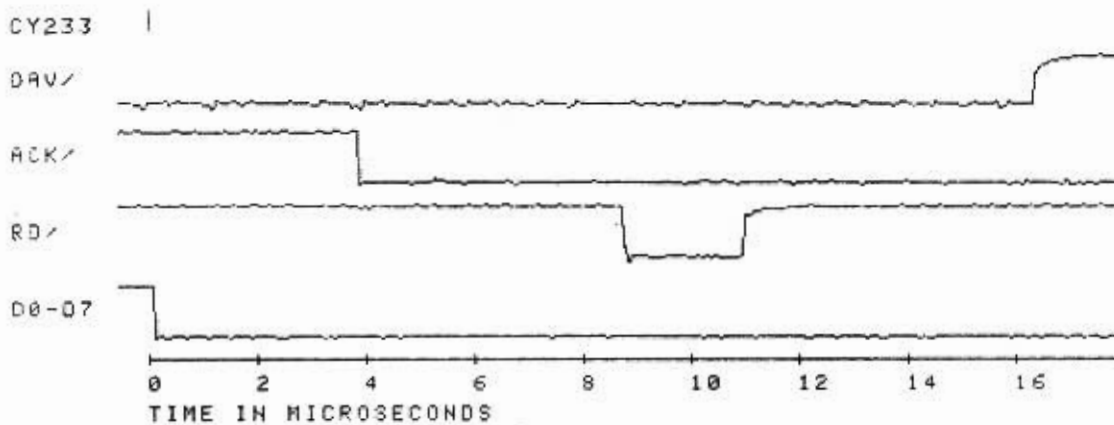


Figure 4.8 Strobed mode read timing details

When the RD/ line is tied low, the non-strobed transfer mode is used. In this mode, DAV/ and ACK/ have the handshake control functions, but RD/ and WR/ are not used as strobe signals. RD/ is tied low, so it does not perform a strobe function, and WR/ assumes the BUSEN/ function. This allows a parallel device to control when the CY233 may drive the data bus, independent of the handshake function. It also allows the CY233 to automatically perform the handshake between DAV/ and ACK/, while allowing data to be latched on the parallel side.

The state of the RD/ line is tested before each transfer is attempted, so you may dynamically switch between the strobed and non-strobed transfer modes.

If the non-strobed mode has been selected, by tying RD/ low, the WR/ line becomes a separate BUSEN/ signal. This is the second function for the WR/ line during data transfers. In this mode, it is an input signal to the CY233, that the parallel device uses to control the writing of the data bus. When the CY233 has data to send to the parallel device, the DAV/ line is brought low, but the data is not written to the bus. The CY233 then waits for the BUSEN/ signal to be low before actually writing the data. This could be used to prevent the CY233 from writing data arbitrarily onto a bus that was being shared between several parallel devices during local parallel transfers.

Alternatively, this separate BUSEN/ signal could be used to latch data without losing the handshake functions of the DAV/ signal. In fact, the non-strobed mode is preferred in this case. Note that slower versions of RD/ and WR/ strobes can be recreated in this mode, by combining the R-W/ and DAV/ signals.

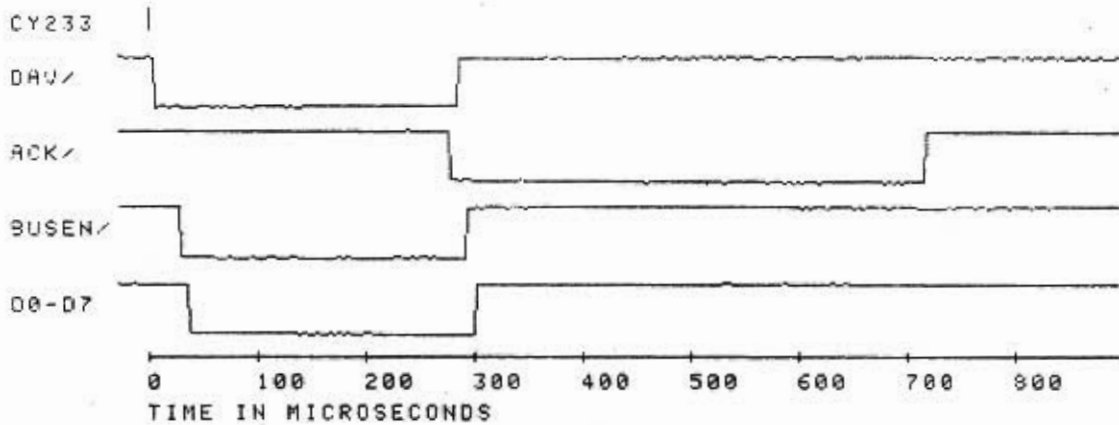


Figure 4.9 Non-strobed write transfer handshake

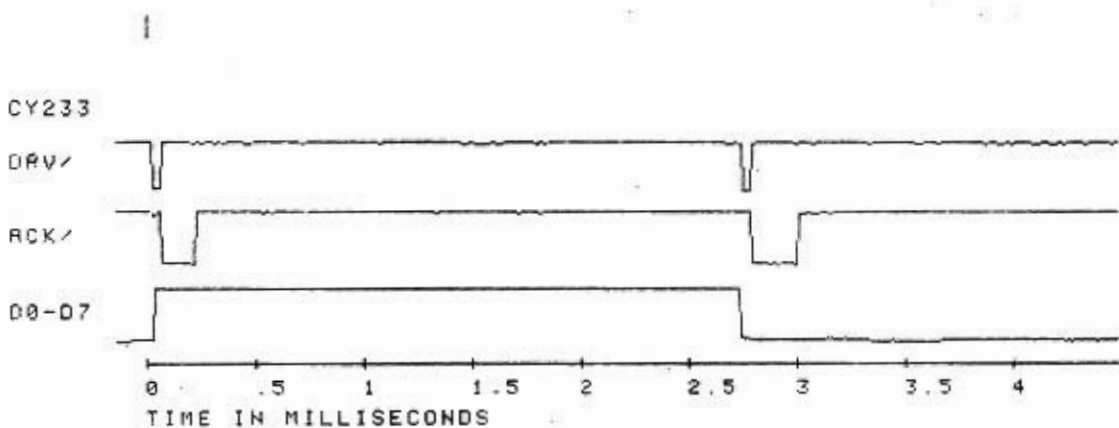


Figure 4.10 Non-strobed writes with latched bus, BUSEN/ low

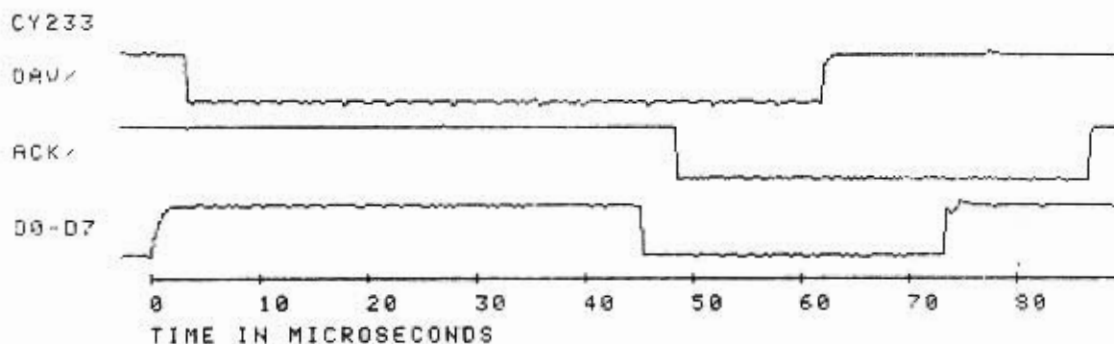


Figure 4.11 Non-strobed read starting from a latched bus

A special mode is selected when the RD/ line is tied high. This function is tested only when the CY233 is reset, and if RD/ is tied high, the CY233 goes into a special LAN mode. This mode is used for a network of all serial devices, and is explained in detail in the Local Area Networks Section of this manual.

Also, the WR/ line has some non-transfer functions. If enabled by the reception of an Error Enable command, the WR/ signal is used to indicate when the CY233 has detected an error condition, such as parity error, buffer overflow, and parallel device timeout. An intelligent parallel device may then request an error status byte from the CY233, to determine the nature of the error. This mode should only be used with intelligent parallel devices, since it requires a special handshake protocol, as described below. Since there are special requirements to support this mode, it is disabled when the CY233 is reset, and is only enabled by the reception of the Error Enable command.

When enabled, the WR/ line will be driven low, with the R-W/ line high, whenever the CY233 has detected an error condition, and when nothing else is pending at this CY233. Note that having the WR/ line low, while the R-W/ line is high, is a unique state, and can be used to generate an error status signal. This function should be used with the strobed transfer mode only, so the WR/ line is not also being used for the BUSEN/ function. Any received serial messages, switch to master mode, or FPL mode operations will remove the error indication until the pending operation is complete.

Once an error is indicated, the local parallel device may ask for the error status byte, by using a modified version of the CY233 handshake. In the modified handshake, the parallel device drives the ACK/ line low before the CY233 drives the DAV/ line. When the CY233 detects a falling edge of ACK/, AFTER indicating an error, it will output the parallel error status byte on the data bus and drive DAV/ low. The address lines will be left floating, R-W/ will still be high, and WR/ will still be low at this point.

The CY233 then waits for the parallel device to drive ACK/ high again, indicating that it has accepted the error status. The CY233 then drives DAV/ high again, restores the data bus to its previous state, and removes the WR/ error indication. Note that this is a backwards handshake, since the parallel device initiates the transfer, and the CY233 responds to the request.

Once the error status byte has been written to the data bus, the CY233 will enable the standard handshake timeout function while it waits for the ACK/ line to be high again. If the parallel device does not respond within the timeout period, the CY233 proceeds as if ACK/ were high again, but it generates a new timeout error, which will give another error indication!

The parallel error status byte has the following definition. A bit that is high indicates that error was detected. The parallel error status byte is cleared after it is read by the local parallel device.

DB0	Received Parity Error
DB1	Receive Buffer Overflow
DB2	Invalid Command Format
DB3	Parallel Write Handshake Timeout
DB4	Parallel Read Handshake Timeout
DB5	Parallel FPL Mode Handshake Timeout
DB6	Character Received was not Carriage Return in the Adaptive Baud Rate Setup
DB7	Reserved

A final use of the WR/ line is in the special LAN mode, where the other uses are disabled. In this mode, the WR/ line controls the application of the local node address to both CY233s, when this address should be used. Otherwise, the local address should be tri-stated, since the LAN CY233 will be generating a destination address. This function is explained in more detail in the section on the LAN mode.

The final control line is the Forced Parallel Load (FPL/) line, which allows the parallel device to initiate read operations from the CY233. There are two ways for the parallel device to request reads from the CY233. One way is to switch the CY233 into Master mode, where it will sequence through the possible addresses until a valid address is found, then read one byte from that address. This process continues while the CY233 is idle, with no messages received from the serial side. The Master mode is explained in greater detail in the section on Characters and Messages.

The second way for a parallel device to request reads is to use the FPL/ line. When the FPL/ line is pulled low, the CY233 will also attempt parallel reads when there is no other activity to perform. However, the local parallel device also supplies the address of the read, unlike the Master mode, in which the CY233 tests all possible addresses, and performs the read when a valid address is found. Saving the address testing makes the FPL/ read faster than the Master mode read, and allows the parallel device

to vary the address. This could be used by a parallel device to send messages to another device on the network. Note that by also controlling the R-W/ line, the parallel device controls the type of message generated by the CY233. Read messages would be used for general data from the local device to the network computer, while write messages could be destined for another parallel device connected to another CY233 in the network.

One externally applied address is reserved by the CY233 for special parallel commands, the address with all lines high (0FFh in encoded mode and no line low in decoded mode). When this address is used, the CY233 interprets the data from the parallel device to be a command to the CY233 rather than a message to another device on the network. Several parallel commands are supported. They are described in the section on CY233 Commands.

Another feature of the FPL/ read is the ability to load more than one byte per message. In the Master mode reads, the CY233 always reads one byte per valid address, then searches for the next valid address. In the FPL/ mode reads, the local parallel device controls the number of bytes read by the CY233.

An FPL/ read operation starts when the parallel device pulls the FPL/ line low. The CY233 will sense and remember this falling edge, even if busy with some other operation. Once the CY233 is idle, with no serial messages being transmitted or received, the FPL/ read will be performed.

The FPL/ read starts with the CY233 generating an ADDR/ strobe, indicating that the parallel device may drive the address lines with the desired FPL/ read address. The CY233 will leave the address lines floating at this time, so they may be driven by the external device. A normal data read is then executed, with the regular handshake lines involved, as described earlier in this section. During the first data read, the externally driven address will also be read, and the state of the R-W/ line will be tested. These events occur when both the DAV/ and ACK/ lines are low, an indication that valid data is available. A reply message will be started from the data read, with the message being a read message, starting with "R" if the R-W/ line is high, or a write message, starting with "W" if the R-W/ line is low. However, no message will be generated if the applied address has all address lines high. Instead, the CY233 will consider this a parallel command issued by the local parallel device to the CY233.

If the parallel device continues to hold the FPL/ line low after the first read operation, the CY233 will continue to perform reads, and the data will become part of the same message started with the first read. Also, parallel commands will require additional data characters, so the external device must continue to hold the FPL line low, insuring that the CY233 will read all bytes of the command.

An FPL/ read may be terminated in two normal fashions. First, if the message terminator value is read from the parallel device, the FPL/ read will end. To restart an FPL/ read after the terminator, the parallel device must raise the FPL/ line, then lower it again, causing the CY233 to detect a new falling edge on the FPL/ line. If the CY233 is in the ASCII HEX character mode, the parallel device cannot generate the terminator, so this mechanism cannot be used to end an FPL/ read. When the parallel device generates a 0Dh value, the CY233 will send it as two hex characters, ASCII "0" and ASCII "D", and will not consider it the terminator. In ASCII HEX mode, the FPL/ read can only be ended by the second method.

The second way to end an FPL/ read is to raise the FPL/ line during a data byte transfer. If the last byte read was not the message terminator, the CY233 will automatically add the terminator character, completing the serial message to the network. In ASCII HEX character mode, the terminator is always added, since it cannot be generated by the local device.

There is also one non-standard way to end an FPL/ mode read, by exceeding the handshake timeout limit while the CY233 is trying to read the next FPL/ value. This will also terminate any message in progress, and generate a timeout error status in the CY233. The FPL/ line must be raised and lowered again to restart the read operations in this case.

Since the CY233 remembers FPL/ requests, single byte reads may be signaled by simply strobing the FPL/ line. When the CY233 is idle, it will read the data byte and generate a serial message.

Multiple byte transfers require the parallel device to hold the FPL/ line low until all bytes have been read by the CY233. End this mode by reading the terminator or raising the FPL/ line, as explained above.

Note that slow parallel devices or long FPL/ messages, with many data bytes, could cause problems in busy networks. Once the CY233 starts an FPL/ read operation, the resulting message is not ended until indicated by the parallel device. If the network is sending serial messages through the CY233 at the same time, the CY233 must buffer the messages until the FPL/ read is finished. Since the internal CY233 buffer is limited, it will eventually overflow, and cause a loss of network data, if the local device keeps the CY233 in the FPL/ read operation for too long. FPL/ generated messages should be kept as short as possible in a busy network, to insure no loss of data.

Messages to Parallel Devices

All transfers between the CY233 and the user's parallel devices occur in transactions called "messages". The message consists of:

- *an address
- *actual data
- *appropriate control line movements
- *a terminator.

The address is that information presented on the A0-A7 lines, and the data is that information transferred over the D0-D7 lines. The terminator is not seen by the parallel device; it is something the serial side requires to separate one message from another.

Messages that transfer data are either "Read" or "Write" in nature. The direction is that perceived by the parallel device. Thus, a write moves data from the CY233 to the parallel device, and a read moves information from the parallel device to the CY233.

The exact serial message that causes a parallel transfer can take several different forms, as explained in the next section. The CY233 supports different types of "Read" and "Write" functions, but the parallel device is treated similarly for each, with the functions of the address, data, and control lines fixed in the selected operating mode.

Writes can transfer any number of bytes, until there is a terminator on the serial side. When this happens, the address is only checked once, at the beginning. Similarly, the ACK/ line is checked for "1" only before the first data byte. Once the transfer is allowed, it goes on. The only thing the parallel device can do is to stretch the individual writes, by delaying in pulling ACK/ low, with the maximum delay limited by the internal timeout value.

Reads may be initiated from a foreign source through the serial line, or they may be initiated locally in what is called the "Master" mode, or through use of the FPL/ line. The user can determine for each CY233 whether it is to be a "master" or a "slave", and not initiate local reads. The resulting information from a successful read is placed on the serial line in the form of a message to the rest of the network. In the Master or FPL/ cases, this message may be a read or write message, depending on the state of the R-W/ line when the first byte is read.

A Simple Interface Example

The schematic shows the details on the parallel side for an interface to eight LEDs (a Write to port) and eight switches (a Read from port). The user has selected the decoded, or linear select, addressing mode by tying pin 23 low. Since only A1 is free, the only valid address is 1, which has an Address Bus bit pattern 1111 1101. The same address is used for both ports.

Simple RD/ and WR/ strobes are satisfactory in this example. Therefore, the other handshake signals are ignored and DAV/ is tied to ACK/. The two ports are always available to the CY233.

At this point it is not known whether this CY233 is a master or a slave. This is not important to the parallel devices. Whenever the CY233 wants to read the switches, it pulses RD/ low which enables the three-state output of the 74LS244 buffer. Notice a closed switch is a TTL 0.

Whenever the CY233 wishes to write to the 74LS273 latch, it pulses WR/ low. Notice a TTL 0 turns the corresponding LED on.

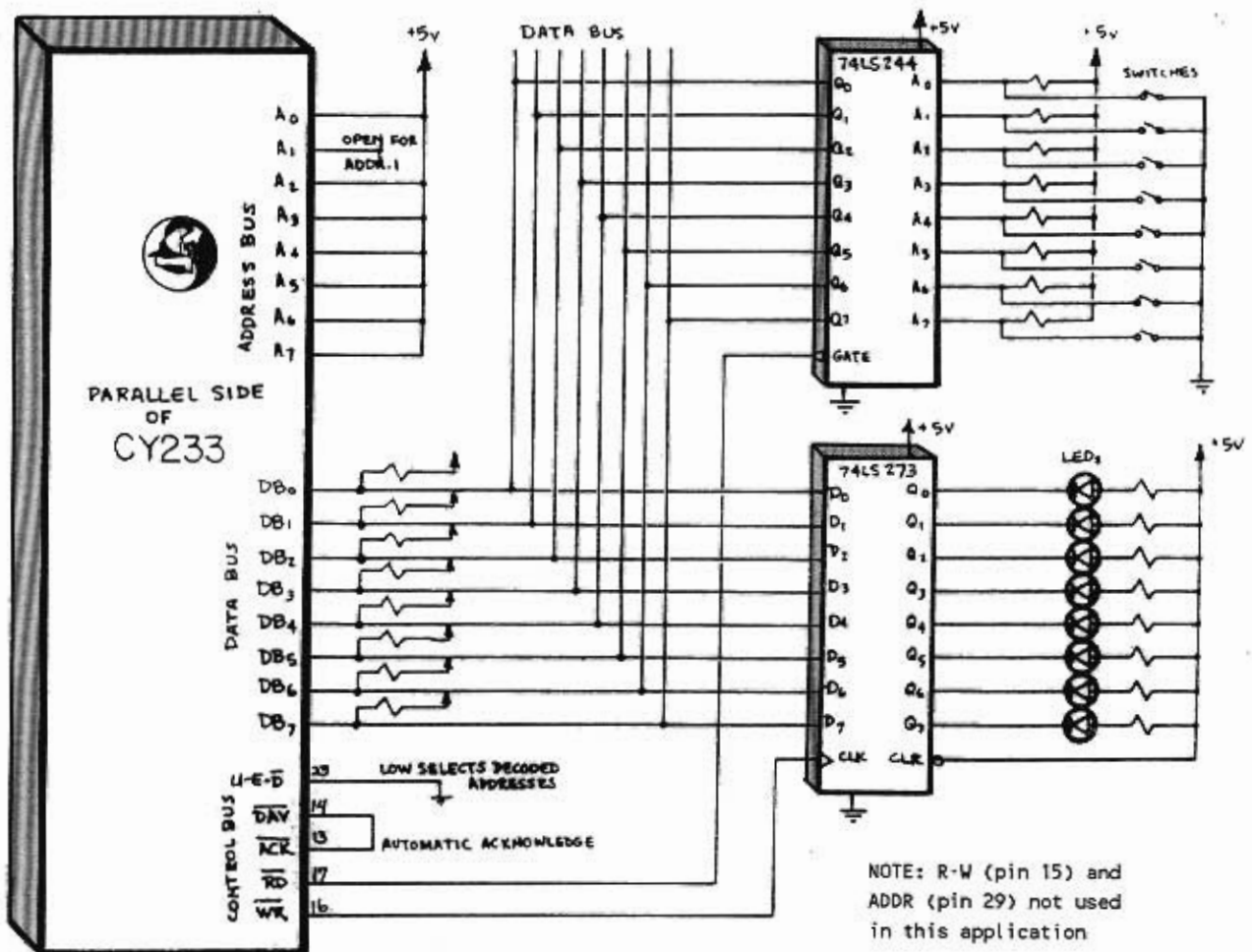


Figure 4.12 Example CY233 connection to switches and lamps

This section goes into the details of serial messages and how to set up the CY233 on the serial side. In this sense, it is analogous to the previous chapter on the parallel side. It is also an extension to section 3, which was an introduction to serial networks.

The Alphabet

Any form of communication requires an "alphabet" with which to build the message. The elements of an alphabet are called "characters". The CY233 has two different character sets available to the user. However, interpretation of a character's meaning depends on the application, and is not enforced by the CY233. For example, the "ASCII" set could be used to transfer "Binary" data if the character codes are properly understood by both the sender and receiver.

Binary is the first character set. It consists of the 256 integers $\langle 0, 1, \dots, 255 \rangle$. These characters are encoded by 8 bit binary words $\langle 00000000, 00000001, \dots, 11111111 \rangle$. The user can attach whatever meaning he desires to each of the characters in the alphabet.

ASCII is the second character set. ASCII is used by many computers and data communication devices to represent the ordinary alphabet, numbers and punctuation marks. In other words, the symbols found on a typewriter keyboard. It also contains special symbols called "control" characters. The ASCII character set used by the CY233 is the 128 element set shown in the table on the following page. ASCII characters may be encoded as 7 or 8 bit binary words in the CY233.

ASCII can be used in another way, to only represent numbers. The set of numbers represented could be the decimal set,

$\langle 0, 1, \dots, 9 \rangle$

but it is common to extend this set to base 16 by using the "ASCII Hex" set

$\langle 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, a, b, c, d, e, f \rangle$.

In order to represent numbers between 0 and 255, two ASCII Hex characters are combined to represent the upper and lower nibbles. Thus 0 becomes 00h, where the "h" indicates ASCII Hex, and 255 becomes FFh. Notice that binary requires one 8 bit character to represent the decimal number 255, while ASCII Hex requires two 7 bit characters, viz. FF. Also notice we could use 8 bit binary characters to transmit ASCII, or ASCII Hex, if the source and receiver both understand the intended meaning. In these cases, the eighth bit may be unnecessary.

Bits					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1				
b7	b6	b5	b4	b3	b2	b1	Column	Row	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	0	1	1	1	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	1	0	0	2	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	1	1	1	3	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	0	0	0	4	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	1	1	1	5	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	1	1	0	6	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	1	1	1	7	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	0	0	0	8	8	BS ←	CAN	(8	H	X	h	x
1	0	0	1	1	1	1	9	9	HT	EM)	9	I	Y	i	y
1	0	1	0	1	0	0	10	10	LF ↓	SUB	*	:	J	Z	j	z
1	0	1	1	1	1	1	11	11	VT ↑	ESC	+	;	K	[k	{
1	1	0	0	1	1	0	12	12	FF →	FS	,	<	L	\	l	
1	1	0	1	1	1	1	13	13	CR	GS	-	=	M]	m	}
1	1	1	0	1	1	0	14	14	SO	RS	.	>	N	^	n	~
1	1	1	1	1	1	1	15	15	SI	US	/	?	O	_	o	DEL RUB

Figure 5.1 ASCII Character Set

In summary, individual elements of the alphabet are characters, which may either be ASCII or binary. ASCII characters may be used in a special way, in pairs, to represent numeric data, by ASCII Hex.

Serial data is transmitted LSB first and MSB last. Thus, a "picture" of the binary character for decimal "2" would be (least significant bit is on the left and time progresses to the right):

```
0 1 0 0 0 0 0 0
```

If "2" is the Arabic numeral, ASCII for this is 32h (see the ASCII table) and the corresponding picture is the 7 bit character:

```
0 1 0 0 1 1 0
```

Finally, if we wish to represent the decimal number 2, which is the same as 02, in ASCII Hex, we would use two 7 bit characters, 30h ("0") and then 32h ("2"):

```
0 0 0 0 1 1 0      0 1 0 0 1 1 0
```

While this may sound complex to the uninitiated, the CY233 does all conversion between "real" data and the selected alphabet. The primary thing to remember is that in the 7 bit character modes, the eighth bit (D7) is ignored.

Start and Stop Bits

The CY233 uses asynchronous communication. This means a character may start any time after the previous one is complete. In order to separate characters, extra bits (really, extra time) are added before and after the data bits. The time before is called the "start" bit and the time after is called the "stop" bit. Each bit time is exactly the same as a data bit time, and is the period specified by the baud rate. A choice of either one or two stop bits is possible. The CY233 always uses one stop bit. This will not normally prove to be a limitation, as it is customary to use two stop bits only at 110 baud, which is not a CY233 baud rate.

The separation afforded by start and stop bits is that, unlike data bits which may either be "0" or "1", start bits are always the equivalent of "0" and stop bits are the equivalent of "1". The CY233, and other UART's, use this particular pattern to separate characters. This works as long as there is a specific "beginning" of the character stream. To understand this, assume there is a generator of serial characters which runs with no extra break between characters, other than the one stop and start bit. If this generator were applied to a CRT terminal while the terminal is off, and the terminal then turned on, the character being displayed could likely not be the character actually being transmitted, because the terminal would not be "synchronized"

with the generator. On the other hand, if the terminal were turned on first, it would synchronize on the first start bit and then remain forever synchronized, and display the correct character.

As an example, if we add start ("0") and stop ("1") bits to the ASCII character for "2" (32h), the resulting bit pattern is

0 0 1 0 0 1 1 0 1

To establish the type of characters to use, the CY233 samples the CHAR line at startup:

Mnemonic	Pin	Function
CHAR	24	Determines serial I/O character type:
	F	ASCII character set
	1	ASCII Hex (0...9,A...F,a...F)
	0	Binary

Parity

Parity is a means of checking on the fidelity of the serial transfer. A "parity bit" is an additional bit added to a character before transmission. The parity bit is checked on reception for correctness. An error which is detected is called a parity error.

There are two basic kinds of parity: even and odd. Even parity means there is an even number of 1's in the composite word - the original character plus the appended parity bit. (Start and stop bits do not count). For example, if the original character contains no 1's (all 0's) then the parity bit is 0. The second kind of parity is odd. Here there is an odd number of 1's in the composite word. The parity bit would be 1 in the above example, if odd parity were used.

Parity can detect single bit errors, but cannot detect two errors within the same word, since they would cancel. In practical systems however, (unless they go over the telephone network), errors are rare, and parity is a sufficient safeguard.

If parity is to be used, there is some advantage to odd parity, since it will detect an error of all 1's or all 0's in a 7 bit word.

If parity is not to be used, the parity bit may be set to 1, equivalent to an extra stop bit, also called "mark" parity, or it may be set to 0, called "space" parity, or it may be unused in the case of 8 bit data characters, and not included at all in the character data stream.

Note that the following are equivalent:

7 data bits, mark parity, 1 stop bit
7 data bits, no parity, 2 stop bits

As a final example, the bit pattern for the ASCII "2", with mark parity (including start and stop bits) is

0 0 1 0 0 1 1 0 1 1

With even parity, it is

0 0 1 0 0 1 1 0 1 1

With odd parity, it is

0 0 1 0 0 1 1 0 0 1

Character Length

The CY233 will work with two sizes for the data portions of a character, either 7 or 8 data bits. The 7 bit modes are common when ASCII characters are being used, since the ASCII set is defined in 128 codes, represented by 7 data bits. In the ASCII Character mode, with 7 bit characters, the CY233 will ignore the eighth data bit value when reading parallel characters, and will set bit D7 to zero when writing parallel characters. The CY233 will never read or write the parity bit. It is always added to the serial bit stream when transmitting, and removed when receiving.

For 8 bit data lengths, all parallel bits are used when reading or writing. Note that the ASCII set can be represented in an 8 bit mode, with the first 128 values, 0 to 7Fh, used for the ASCII set, and the additional values, 80h to FFh, used as required.

The ASCII Hex mode can use either a 7 bit or an 8 bit format to represent all 8 parallel data bits, since two serial ASCII characters are used to represent one 8 bit parallel data byte.

The Binary mode should use an 8 bit character format, although the 7 bit formats are allowed. However, if a 7 bit format is used, the CY233 can never receive the message terminator, B3h, since 8 bits are required to receive the terminator code in one binary character.

To establish parity and character length, the CY233 samples the PAR0 and PAR1 lines at startup. The total character length is the sum of start bit + data bits + parity, if used, + stop bit.

Mnemonic Pin Function

PAR1, PAR0 26,25 Determine serial I/O parity and data length:

Pins	Parity	Data Length	Total Length
F F	Mark	7	10
F 1	Even	7	10
F 0	Odd	7	10
1 F	Space	7	10
1 1	None	8	10
1 0	Space	8	11
0 F	Mark	8	11
0 1	Even	8	11
0 0	Odd	8	11

Baud Rate

The baud rate is the reciprocal of actual time allotted to each bit in the serial character. This time is crystal controlled in the CY233, and is therefore quite accurate. Two sets of baud rates are supported by the CY233. Standard rates from 300 to 19.2K baud use an 11.059 MHz crystal. However, if the crystal frequency is changed to 7.3728 MHz, a faster selection becomes a standard 38.4K baud. In addition, an adaptive mode allows the CY233 to adjust the baud rate to that of the network, and allows more freedom in the choice of operating crystal.

Most standard rates are defined when the CY233 operates with an 11.059 MHz crystal. The highest possible rate, 57600 baud, is not a standard rate, and should only be used in CY233 networks that do not use a host system, such as the back-to-back wire saver mode. Note that 11.0 MHz will work in place of the 11.059 MHz crystal, since the error is only 0.6%.

With a 7.3728 MHz crystal, the top rate becomes a standard 38400 baud, but all other rates become non-standard. Be sure your host system can handle this rate in a network, if it is chosen, since very few general systems can operate continuously at this data rate.

When the adaptive rate is chosen, the CY233 will adjust the baud rate to match that being used by the network. This allows the use of a non-standard crystal frequency or baud rate, as well as letting the host system dictate baud rates to the network.

In the adaptive mode, the host system must send two carriage return codes, 0Dh, to the CY233 when power is applied or when it is reset, allowing the CY233 to adjust the internal baud rate to match that of the characters received. If the Echo All or Echo Invalid modes are selected, the CY233 will echo the two carriage returns after both have been received, so that all nodes in a ring network may be initialized. The carriage returns must be sent before any normal network communications begins. Note that the adaptive mode requires a host system in the network, to dictate the operating rate. A parallel device connected to a CY233 cannot initiate a parallel read and message transfer until the operating baud rate has been established.

To establish the baud rate, the CY233 samples the BDR0 and BDR1 lines at startup:

Mnemonic	Pin	Function
BDR1, BDR0	28,27	Determine serial I/O baud rates:
	F F	Self Adaptive
	F 1	57600 / 38400
	F 0	19200
	1 F	9600
	1 1	4800
	1 0	2400
	0 F	1200
	0 1	600
	0 0	300

All rates are shown for an 11.059 MHz crystal, except 38400, which requires a 7.3728 MHz crystal.

As a timing example, if a character requires 1 start bit, 7 data bits, 1 parity bit, and 1 stop bit, for a total of 10 bits, at 2400 baud, it requires 4.16mS to transmit. This is 240 characters per second.

Commands, Addresses, Data & Terminators

Except in the UART mode, all serial communications with the CY233 occur in special messages. Serial messages conform to a definite but simple protocol, which is a natural extension of what you would type if you were talking to the device from a keyboard. Each message consists of a collection of characters from the chosen alphabet, and has four components.

Messages begin with a "Command" letter. Those messages that cause parallel data transfers with the parallel device are either Read or Write, as referenced to the parallel device. I write information to the parallel device and read information from it. The standard Read or Write messages, using the "R" and "W" command letters will be used as examples in this section, illustrating the nature of the serial messages. Other types of commands are discussed in the next section of this manual.

Next is the "Address" which selects specific parallel device(s). The address portion is used in testing for a valid or invalid address at a particular CY233.

Following the address is a "Data" section of the message. Messages can have any number of data bytes, depending on the nature of the transfer. In the ASCII Hex modes, two serial characters are used to represent one parallel data byte value.

Finally, there is a terminator to signify the end of the message.

Each of the three alphabets (character types) has slightly different rules for these four parts: command, address, data, and terminator. Part of the flexibility of the CY233 is that the user can choose the protocol to fit his needs.

For the ASCII character type (CHAR=F), each data byte is represented by one character. Either 7 or 8 bit characters may be used, with the D7 parallel bit value ignored when reading 7 bit values, and set to 0 when writing 7 bit values. Addresses are represented in ASCII Hex, and thus require two characters to represent the range <00...FF>. Data for Writes is any positive number (1 or more) of bytes, each requiring one character. Data for Reads is the same, except where there is only one byte. The terminator is <CR> (0Dh, or ^M="Control M"), and is also written to the parallel device in some cases. This is a very flexible operating mode, since all commands are supported in the ASCII character type.

For ASCII Hex (CHAR=1), each data byte is represented by two characters. Addresses are also represented in ASCII Hex, and thus require two characters to represent the range <00...FF>. Data for Writes is any positive number (1 or more) of bytes, each requiring two characters. Data for Reads is the same, except where there is only one byte. Only valid characters are passed on to the parallel device. The Terminator is <CR> (0Dh, or ^M=Control M), and is not written to the parallel device.

For binary (CHAR=0), each data byte should be represented by one 8 bit character. There are no explicit commands for this character type. Instead, any message on the serial line is considered to be a Write as far as the local parallel device receiving it is concerned. Binary mode messages contain only three components, starting with the address. Addresses are a single byte, and are thus the first character of a new message. Any address except 255 (FFh) can be used. Data is any positive number of bytes (1 or more) until the terminator. The terminator is 179=B3h, and is not written to the parallel device.

Since Write messages should always contain at least one data byte, the terminator character can be successfully transferred to the parallel device in binary mode if it is the first data byte in the message. If it is the second, or later, it is interpreted as the terminator of the serial message and is not transferred to the parallel device.

It is important to note that binary, for example, can be used to transfer ASCII information between a source and a receiver, as can ASCII Hex or ASCII Character be used to transfer binary information. The labels refer to how information is represented internally to the serial message, not what can, or will, be done with it by the parallel devices.

An example of an ASCII Hex message which causes a Write to location A2 (A2 hex = 162 decimal) of the data 4F (79 decimal) is

`WA24F<cr>` (6 characters)

Notice that <cr> is the single ASCII carriage return character, value 0D in hexadecimal notation. As the message terminator, it is sent as a single character, not as two Hex characters.

Now using hexadecimal notation for 8 bit binary data, the following example of a binary message causes a Write to location F0 (240 decimal) of the data 31 (49 decimal) and the data 02 (2 decimal):

`F0 31 02 B3` (4 characters)

An example of an ASCII character write message is:

`W20The Quick Brown Fox<cr>`

which has 23 characters (1 for "W", 2 for the hex address 20, 19 for the alphabetic message, and 1 for the <cr> terminator) and transfers

`The Quick Brown Fox<cr>`

to location 20 (32 decimal). Notice that the <cr> terminator is transferred in this case.

An example of an ASCII character read operation would be a read of the parallel device with hex address 20 (32 decimal). The requestor issues the command R and the two hex character address. Any CY233 which has this address as valid will respond with one byte of data and the <cr> terminator. The characters in the message, then, are

`R20x<cr>`

where the response by the requested device is, for example, "x"<cr>.

All serial read commands transfer one byte, which is two characters in ASCII Hex and one in ASCII Character mode. However, read messages may have more than one data byte if they originate from the parallel device, using the FPL/ line.

Networks, Masters, and Slaves

The CY233 can be used in many networks, which can be thought of as large scale circuits. Many CY233s may be communicating with one another or with a central microprocessor or computer. In order to control this activity, the CY233 has two more pins: DUP (Duplex) and NET (Network).

To establish the desired configuration, the CY233 samples DUP and NET while idle between messages. It is possible to dynamically switch the CY233 between any of the supported modes, however, it is most common to switch between the Master and Slave modes of a particular echo selection.

Mnemonic	Pin	Function
NET	22	Network protocol; Master/Slave
DUP	21	Duplex; Echo mode

Two concepts, MASTER/SLAVE and ECHO are used in the following description.

A CY233 is in the MASTER mode when it will initiate a transmission of data from a local parallel device onto the serial network, in the absence of any specific request by the network. In other words, on its own accord, it will read a parallel device and transmit this data onto the network. It will do this whenever the network is sufficiently free of transmissions, and the CY233 can find a local parallel device from which it can read data. If a CY233 is not in the MASTER mode, it is considered to be a SLAVE, and will not initiate messages on its own. However, a parallel device can still initiate a transfer in this mode, by using the FPL/ line.

A CY233 will ECHO messages it receives from the serial network under certain circumstances. There are the following possibilities:

ECHO ALL, echo any valid or invalid message received, perhaps adding data if the message is a read command,

ECHO NOTHING, although replies will be made to read commands,

ECHO VALID, which means that only messages containing a valid address for a local parallel device will be echoed, along with any required read reply, and,

ECHO INVALID, which means that only messages not containing a valid address for a local parallel device will be echoed. When a valid address occurs, any required reply will be made, however.

note: ECHO ALL will not echo any extraneous characters that might occur between messages. Echo is not enabled until after a valid command character and valid or invalid device address have been received. However, a special command may be used to enable echo between messages if this is required by your application.

The combinations of NET and DUP are shown in the following table.

Mode	NET	DUP	ECHO	Master/Slave	Useful for
A	0	0	Invalid	Master	Ring
B	0	F	None	Master	Point-to-point
C	0	1	All	Master	Ring
D	F	0	Invalid	Slave	Ring
E	F	F	None	Slave	Half-duplex ***
F	F	1	All	Slave	Console to CY233
G	1	0	Valid	Master	Console to CY233
H	1	F	Valid	Slave	Bus
I	1	1	Valid	Slave	Bus

*** = line turnaround

The modes are identified by letters for easy recognition in later descriptions. Although certain modes are useful with particular network configurations, any mode may be used so long as it satisfies the communications requirements of the application.

The communications modes have been chosen to allow easy switching between the Master and Slave versions of a particular echo selection. One of the two control lines, DUP or NET, is floating in the Slave mode, while it is low in the corresponding Master mode. An open collector buffer can be used to switch between the modes as required by the parallel device.

In the ASCII modes, the CY233 does not respond with any echoes until it has received a valid command letter, followed by two valid hex ("0" - "9", "A" - "F", or "a" - "f") address characters. Then it determines its echo mode and responds accordingly. Any invalid characters in the command or address result in a terminated message and no response. Nothing will happen until the next valid message.

ASCII hex mode also has a filter for hex data characters. Non hex data characters will be echoed, as required, but are not passed on to the parallel device. This feature may be used to embed control characters, such as spaces, in a message without passing them to the parallel device.

Specific examples of these configurations will be given in later applications chapters.

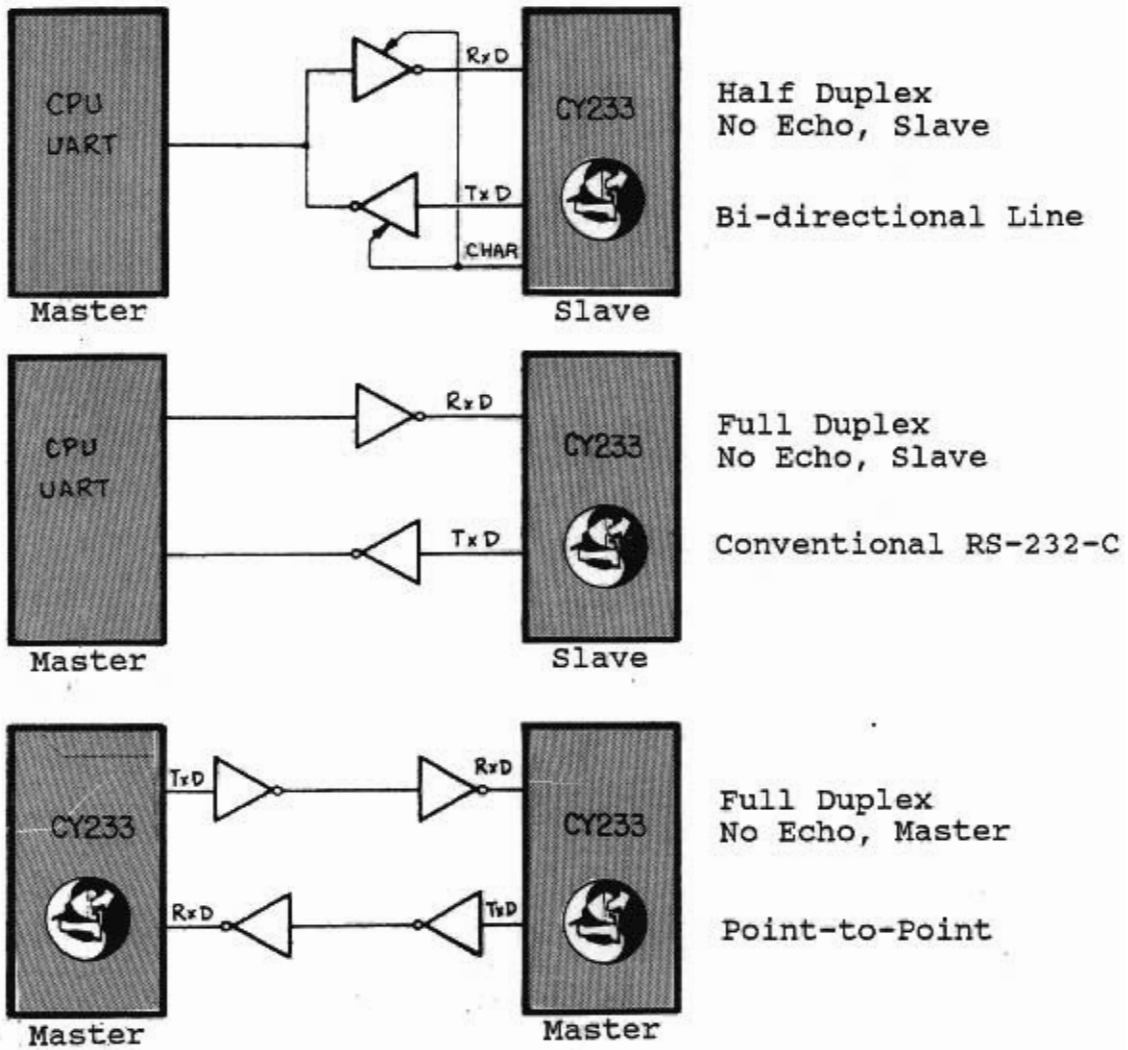


Figure 5.2 Simple Network Configurations.

The CY233 as a UART

The UART Mode is the simplest operating mode of the CY233. In this mode the CY233 passes data characters between the serial and parallel sides, without any message structures, and with limited communications protocols. This mode allows the CY233 to be used as a single-chip UART, without the network functions. Since there are no addresses in this mode, only one CY233 would be used per serial line, unless multiple CY233s were to receive the same data from the host.

The UART mode is chosen by tying the U-E-D/ line high, and is tested when the CY233 is reset. When the CY233 detects that it is in the UART mode, it bypasses the normal command and address portions of the message structure, and considers all serial communications to be data only.

Within the UART mode, the CY233 baud rate, parity, character format, character type, and echo modes may still be set by their respective pin functions.

For the ASCII and Binary character types, one serial character corresponds to one parallel side transfer. For every character received, the CY233 will write that character to the parallel side, and for every character read from the parallel side, the CY233 will send one serial character.

For the ASCII Hex character types, two serial characters correspond to one parallel transfer. The CY233 must receive two valid hex characters before it will write the represented value to the parallel side, and it will send two hex characters for every value read from the parallel side.

For the Echo All and Echo Valid modes, the CY233 will retransmit any serial character received. For the Echo Invalid and Echo None modes, the CY233 will not retransmit received characters. In these cases, it will only transmit data read from the parallel side.

When the CY233 receives serial characters, they will be written to the parallel side. The parallel write occurs with the normal write transfer handshake, using DAV/, ACK/, R-W/, RD/, and WR/. Both the strobed and non-strobed transfers may be used, and the data bus may be latched by holding DAV/ low in the strobed case, and holding WR/ (BUSEN/) low in the non-strobed case. There is no address test performed, and the address lines are not used. Also, a low ACK/ line does not mean an invalid CY233, since this is part of the address test. However, the CY233 will wait for the ACK/ line to be high before DAV/ is driven for the next transfer, within the default timeout limits of the handshake.

There are two ways for the CY233 to read parallel data in the UART mode. If the CY233 is in one of the Master modes, it will attempt a parallel read when there is nothing else to do. If the read is successful, the results will be transmitted serially. If the read times out, no transfer occurs, and nothing is transmitted. The read operation also uses the normal handshake protocols, with R-W/ high to indicate the read, and RD/ used in the strobed case. The CY233 may be dynamically switched between master and slave modes, as well as between various echo modes.

The second mechanism for reading parallel data is the use of the FPL/ line. When the line is held low, the CY233 will perform parallel reads and transmit the results. Multiple reads will be performed if the FPL/ line is held low, so long as the reads are successful. If a read times out with no parallel response, the read function will terminate until the FPL/ line is pulled low again. Otherwise, reads will proceed until the parallel device raises the FPL/ line again.

If serial data is received while the CY233 is reading parallel data due to the FPL/ function, the serial data will be saved in an internal buffer. The CY233 will not attempt to write parallel data in the middle of a series of FPL/ reads. Once the FPL/ line is released, the CY233 will start to write the received data. If there are a large number of FPL/ reads with no break, there is a danger of overflowing the CY233 receive buffer and losing received serial data.

Other CY233 functions are not implemented in the UART mode. Since there is no message structure, no CY233 commands are active, and only the default operating modes can be used. For example, it is not possible to modify the handshake timeout value used in parallel data transfers.

A popular application of the CY233 UART mode would be the connection of a parallel device to a computer that has only serial I/O ports. For example, the CY233 could connect a serial computer port to a parallel printer port. In this case, the message structure of the non-UART modes represents unnecessary overhead, since the computer simply wishes to send serial data to be printed on a parallel printer. This type of connection is illustrated in the following figure:

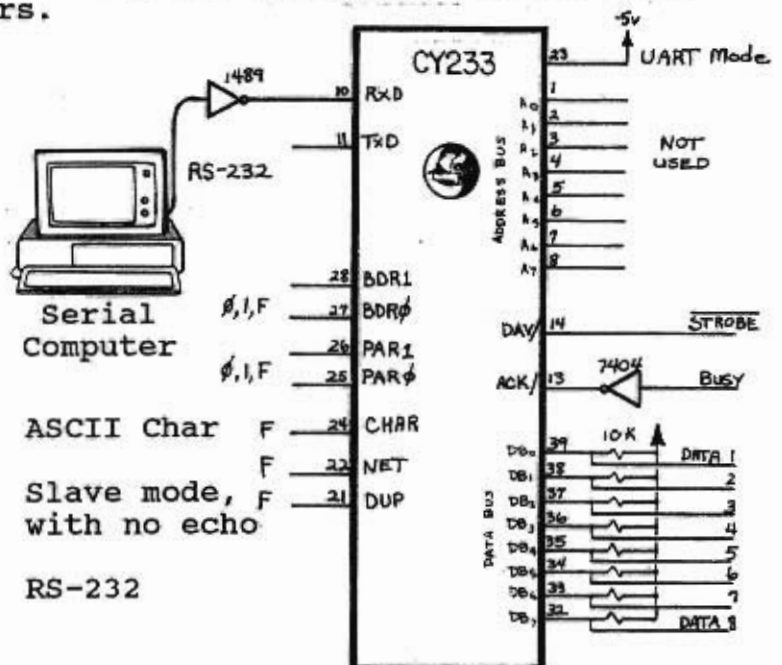


Figure 7.1 Serial computer to parallel printer example.

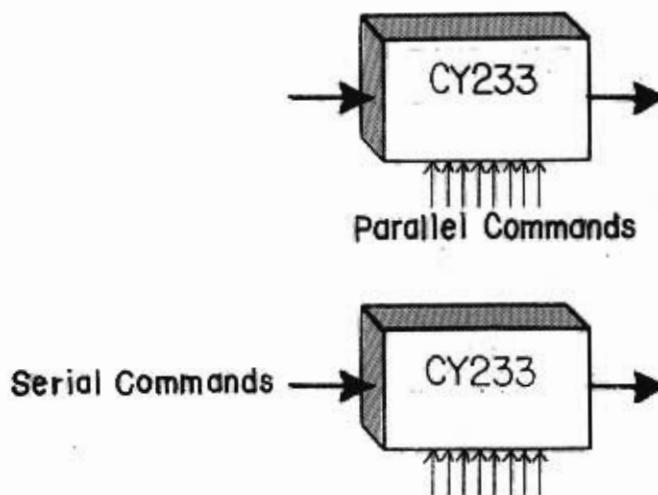
The Command Set

This section describes the commands used to implement various functions in the CY233. Each command is designated by a single ASCII command letter, and for serial commands, follows the general message structure described in the previous section. Each serial command is available to the ASCII Character and ASCII HEX character modes, but not to the Binary mode. The message format of the Binary mode is different from that of the ASCII modes, starting immediately with the address byte. All Binary mode serial messages are considered to be write messages. To read data in the Binary mode, the CY233 must be put in a master mode, or the FPL/ line must be used. Thus, none of the other serial commands are available to the Binary mode. However, the special parallel commands may be used in Binary mode.

The CY233 commands use the upper case ASCII letters, and reserve these letters for message headers when they start a message. The data portion of a message is not restricted, and any code may be used within the data message section, except the terminator character, which will end the message.

The various commands for the CY233 may be split into three major groups that depend on how the commands are issued to the CY233. The most common commands are the Network Serial Commands, that are sent and received as messages on the serial side of a CY233, functioning as part of a serial network. All message examples in this manual thus far have been examples of Network Serial Commands and messages. These commands will be used between the CY233s and host system of a standard CY233 network.

Network Serial Commands are the most common commands used by applications of the CY233, and will be described first in the following command description sections.



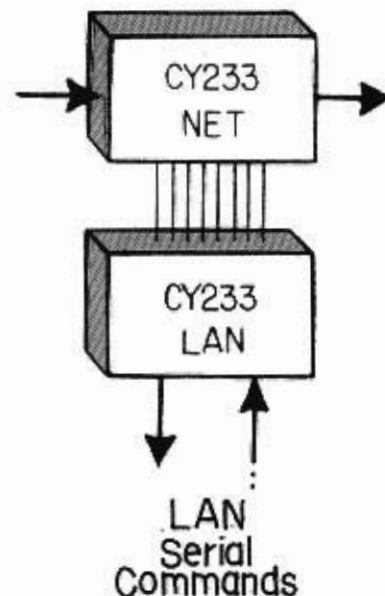
A second group of commands supported by the CY233 is the Parallel Commands. These commands are issued to the CY233 by a local parallel device connected to the CY233. This gives an intelligent parallel device some control over the operation of the local CY233 connected to that device. Use of these commands is strictly optional. They would not be needed or required in a system that has random logic connected to the parallel side of the CY233.

The CY233 will treat locally generated data as a Parallel Command if it was read by an FPL/ signal request with the applied address of all ones (OFFh). Any other address will result in the normal FPL read or write serial message to the network. Thus, address FF is reserved for FPL/ generated data, and signals that a command is being issued, rather than the normal FPL/ data function.

Most Parallel Commands control a local function of the CY233, such as the handshake timeout value. These commands have equivalent Network Serial Commands that perform the same functions, and result in some internal change to the CY233, without generating any serial or parallel data. This allows an intelligent parallel device to configure the CY233 as appropriate for the device, without burdening the network host computer with the details of what that particular device requires.

A few of the Parallel Commands perform functions that will generate special serial messages. These commands are used in special circumstances, and are described in detail in the following section on the Parallel Commands. Some are implemented in support of the special Local Area Network (LAN) mode of the CY233, and would probably not be used in a standard CY233 network.

The last group of CY233 commands is the LAN Serial Commands, which are serial commands sent to the LAN CY233 from the local serial system connected to a LAN node of a network. The node and network configurations are discussed in the section on Local Area Networks, later in this manual. The LAN CY233 is one of two CY233s that make up a LAN node, the other being the Network CY233. The LAN CY233 is connected to a local serial system that wishes to communicate with the network. When the LAN CY233 is operated in a message (non-UART) mode, the LAN Serial Commands are used to control communications between the local system and the network.



Only a few LAN Serial Commands differ from the Network Serial Commands, and these are discussed later in this section. Other LAN Serial Commands will be treated the same as their Network Serial counterparts.

The CY233 commands have been broken into their functional groups in the following tables. Each command is explained in detail in the following sections.

There are two classes of unused commands for the Network and LAN Serial Command groups, Undefined, and User Reserved. Undefined commands will cause an Invalid Command Format error if the CY233 receives any message that starts with one of the Undefined command letters. Such a message will be removed from the network, as the CY233 will ignore all characters of that message until a terminator is received. This insures that noise or communications errors are not spread throughout the network. Error bit 2 of the status bytes will be set to record this error.

User Reserved commands are treated the same as Undefined commands until the CY233 receives an All Echo Command (Jaa<cr>). After this, any User Reserved command is treated as a CY233 serial message for communications purposes, but is always echoed by the CY233. This allows a non-CY233 device on a network to have its own messages, independent of those for the CY233.

Network Serial Commands

@	Test Network Link, @aa<cr>
A	Undefined, Hex Character conflict
B	Undefined, Hex Character conflict
C	Undefined, Hex Character conflict
D	Undefined, Hex Character conflict
E	Undefined, Hex Character conflict
F	Undefined, Hex Character conflict
G	Reserved command letter
H	Undefined
I	Initialize CY233, Iaa<cr>
J	Enable All echo mode, Jaa<cr>
K	Enable parallel Error status, Kaa<cr>
L	Undefined
M	Fill consecutive locations, Maahh...hh<cr>
N	Display consecutive locations, Naacchh...hh<cr>
O	Reserved command letter
P	Set Periodic Master mode delay, Paatttt<cr>
Q	Query for serial error status, Qaass<cr>
R	Read from parallel device, Raad...d<cr>
S	Sense local address, Saabb...ll<cr>
T	Set handshake Timeout delay, Taatttt<cr>
U	Token message, Uaa<cr>
V	Undefined
W	Write to parallel device, Waad...d<cr>
X	Transfer to parallel device, Xaad...d<cr>
Y	User Reserved, ASCII 59h
Z	User Reserved, ASCII 5Ah
[User Reserved, ASCII 5Bh
\	User Reserved, ASCII 5Ch
]	User Reserved, ASCII 5Dh
^	User Reserved, ASCII 5Eh
_	User Reserved, ASCII 5Fh

Parallel Commands

G	Reserved command letter
I	Initialize CY233, I<cr>
J	Enable All echo mode, J<cr>
K	Enable parallel Error status, K<cr>
L	Load value of Status byte 2, Lvv<cr>
O	Reserved command letter
P	Set Periodic Master mode delay, Ptttt<cr>
S	Generate Sense Message, S<cr>
T	Set handshake Timeout delay, Ttttt<cr>
U	Generate Token message, Uaa<cr>

Only the commands listed above are available as Parallel commands. Any others will set the parallel status error bit 2, for an invalid parallel command format, then attempt to continue with a normal FPL initiated data message. This is similar to a normal FPL read operation, with an address other than 0FFh, and will result in a serial read or write message, depending on the level of the R-W/ line.

LAN Serial Commands

@	Return LAN Node Address, @aa<cr>
G	Reserved command letter
I	Initialize CY233, Iaa<cr>
O	Reserved command letter
P	Pass Parallel Command to Network CY233, PCmd<cr>
Q	Query for serial error status, Qaass<cr>
R	Generate Read Message to Network, Raad...d<cr>
T	Set LAN handshake Timeout delay, Taatttt<cr>
W	Generate Write Message to Network, Waad...d<cr>

Any command letters not listed above should not be used in a Local Area Network. All LAN commands not listed will be attempted as if they were received by a Network mode CY233. While many of these commands will not generate any error status, their operations do not make sense in a LAN environment, and their use will probably cause other data transfer errors.

In the following descriptions, command formats are shown with particular fields designated by various letter combinations. The command headers are all similar, consisting of the particular command letter (upper case ASCII), followed by a two character hex address, indicated by "aa". This is followed by the data portion of each command, and varies with the particular function involved. Commands end with the terminator character, a carriage return in the ASCII modes.

Read (R) Command

The read command, designated by "R", will read and transfer one byte of information, if the command is received from the serial side, or if initiated from a Master mode. However, if a local read is initiated by a parallel device, using the FPL/ line, multiple data bytes may be returned in the same message. Therefore read messages may have more than one data byte. The general formats are:

R aa <cr> as sent by the serial host, or

R aa d ... d <cr> reply in ASCII, and

R aa hh ... hh <cr> reply in Hex

The spaces are added for clarity, but are not actually part of a message. In ASCII Character mode, "d" represents a single data byte value, sent as a single character. In the Hex mode, "hh" represents a single byte value, sent as two Hex characters, with the most significant nibble sent first.

When the CY233 is operating in the Echo None modes, the "Raa" portion of the message will not be echoed by the part, for any Read commands received by the serial side. The reply for these messages will consist of the single data value and terminator portions only. Messages initiated locally in the Master mode, or by use of the FPL/ line, will always contain the "Raa" header, allowing the network host computer to identify the source of the data.

For the Echo Invalid cases, the CY233 treats the Read command specially, since a reply is expected when the command is received from the serial side. The CY233 for which the message is valid will echo the "Raa" header along with the reply and terminator. Since Echo Invalid is normally used in a Ring configuration, this keeps the reply from being lost at the next node in the ring, and insures that the complete message will be returned to the network host computer.

Write (W) Command

The write command, designated by "W", will transfer any number of bytes to the parallel device, and finishes when the terminator is received. The formats are:

W aa d ... d <cr>	in ASCII,
-------------------	-----------

W aa hh ... hh <cr>	in Hex, and
---------------------	-------------

a d ... d <B3>	in Binary
----------------	-----------

Note that Binary mode uses single bytes for all data, including the CY233 address that starts the message. An identical format is used to send data read in the Master mode or by the FPL/ line in Binary mode. If the first data byte of a Binary message has the value FFh, it is not considered the terminator, and will be written to the parallel device. In this case, a separate terminator must still be supplied. The message terminator byte is not written.

In the ASCII Character modes, the carriage return terminator is written to the parallel device. It is both the last value written, and the character that ends the message. In the ASCII Hex modes, the terminator is not written, since it is not part of the Hex alphabet. To write the value 0Dh to the parallel device, send the two hex characters "0" and "D".

When the parallel device pulls the R-W/ line low during Master mode or FPL/ reads, the resulting message will be a Write message instead of the default Read message. The formats are identical to those received from the serial side, and allow a parallel device to generate write messages for another parallel device in a CY233 network. This feature allows networks that do not contain host computers, but limits the types of messages that can be handled by the network. Only the "R" and "W" commands would be available in this case.

Transfer (X) Command

The transfer command, designated by "X", is similar to the write command, but in the ASCII character mode it will not write the terminator carriage return, unless it is the first data character. This allows users to operate the CY233 in the ASCII mode, but write messages to the parallel devices without always writing the carriage return terminator.

Also, operation of the transfer command is identical in ASCII and Hex modes, with one data byte written per character received. This is not the standard hex format, in which two serial characters represent one parallel byte value. However, it makes direct data transfers possible with a CY233 that is normally configured for the Hex character mode. If this function is not desired, simply do not use the X command in Hex mode CY233 networks. The command format is:

```
X aa d ... d <cr>
```

The command will write out the "d ... d" portion, as one byte per character received, with a normal handshake executed for each byte transfer. Note that the terminator is never written, unless it is the first data character, where it is treated as a data value instead of a terminator. In this case, the message must still include a separate terminator character.

Fill (M) Command

The CY233 has a special command for writing individual bytes to sequential addresses, known as the fill command, using the command letter "M". The message header contains the starting address, which must be valid for the command to continue. The CY233 writes the first data value to the starting address, then increments the address for each additional value received. These next sequential addresses are not tested for validity, the data transfer simply proceeds with a new address. The function ends when the message terminator is received. Message format is:

```
M aa hh ... hh <cr>
```

The fill command uses a hex format for the data portion of the message. This is true for both ASCII Character and ASCII Hex modes, and allows arbitrary data to be written to sequential addresses. For example, "M02215A7F<cr>" would write 21h to address 02, 5Ah to address 03, and 7Fh to address 04. The terminator is not written, as is standard with hex format data.

Display (N) Command

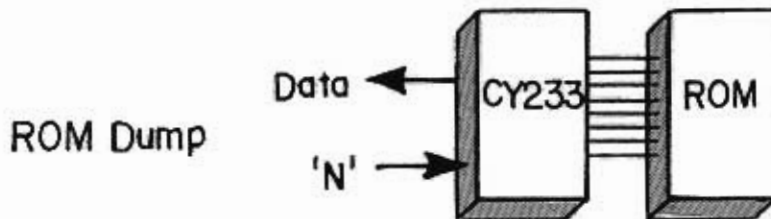
As a complement to the fill command, the CY233 also has a display command, designated by "N". This command will read data bytes from sequential addresses and send the results back in a single message. The display message format includes a count parameter after the address, to indicate how many bytes to read. The count is a two character hex value, similar to the address, and allows from 1 to 256 data bytes to be read. Format is:

```
N aa cc <cr>
```

as sent by the serial host, and

```
N aa cc hh ... hh <cr>
```

as sent by reply from the CY233



In the Echo None modes, the CY233 will not echo the "Naacc" portion of the reply. Data values are sent in the hex format for both ASCII and Hex modes, and there will be cc values of hh data in the reply, with each value coming from the next sequential address, starting at address aa. Only the starting address is tested for validity. If the cc value is zero, 256 values will be sent.

The CY233 implements several mode control commands that have no data portion to their messages. These commands enable a particular function that is specified by the command letter only. No data transfers take place, since these commands control some internal CY233 function. Note that a valid address must be received before the CY233 will act on the command, allowing these special commands to be used by only particular devices in a network of parts.

All Echo (J) Command

The first command enables the echoing of characters between CY233 messages, and support for the User Reserved command messages, which could allow the mixing of CY233s with other serial equipment in a single network. User Reserved messages will be echoed by the CY233, so it will not act on them. The echo will continue until the terminator character is sent. Also, any non-command characters, such as control codes or lower case ASCII characters will be echoed between messages. However, as soon as a valid command letter is received, the CY233 will intercept the data stream and treat it as a CY233 message. The command uses the letter "J", for all echo mode. Message format is:

```
J aa <cr>
```

This will enable the all echo function for the specified CY233. Once this mode is turned on, it can only be turned off by restarting the CY233.

Enable Error (K) Command

Another control command enables the parallel error status function, and is designated by "K", for error enable. When this command is received, the specified CY233 will indicate error status to the local parallel device (when errors occur), using the WR/ signal while the CY233 is idle. Message format is:

```
K aa <cr>
```


This function is enabled by command because it uses signals that are also involved in the normal parallel data transfer. If the parallel device is not intelligent enough to handle the error status indication and backwards handshake of this mode, confusion could occur between the CY233 and parallel device whenever some error was detected. Since the function is enabled by a serial command, the network and host computer must also know which parallel addresses are able to handle local error status. Note that the local device could also enable this function by loading the parallel error enable command into the CY233.

The error status byte values and special handshake mode are described in the previous section on The Parallel Side.

Initialize (I) Command

The last control command allows the network to restart a CY233 without pulsing the Restart line. The initialize command, designated by "I", is equivalent to generating a local, hardware restart. A valid address and terminator must be received before the CY233 will reset itself. Message format is:

```
I aa <cr>
```

This command turns off any special functions, such as enabled by all echo and error enable, and causes the CY233 to reconfigure itself. It will redetermine all power-on characteristics, including baud rate, parity, and character length. Note that for the adaptive baud rate function, the I command message must be followed by two additional carriage returns to properly reset the CY233 baud rate, prior to resuming normal operations.

Query (Q) Command

The query command, designated by "Q", will send internal status information about the CY233 out the serial side, and will not attempt a parallel data transfer. This is intended as an error reporting mechanism, with the status information indicating if parity, buffer overflow, badly formed messages, or parallel timing problems have occurred. The status flags will be cleared when they are read by this command. The serial error status byte reported by this command is separate from the parallel error status that can be read by a local parallel device. This allows both the network and the parallel device to query the CY233 for error status. When an error occurs, the status bits of both bytes are set by the CY233, and when a status byte is read, only that status byte is cleared, allowing the other of the serial and parallel sides to also receive the same error information.

The query response may send back one or two status byte values. The first value contains the error status flags described above, while the second byte can contain an optional value loaded by the parallel device. When the CY233 is restarted, the second status byte value is set to zero, and so long as it remains zero, the byte will not be sent by the query command. However, if an intelligent parallel device sets a non-zero value, using the parallel load (L) command, this value will also be sent. Use of the second status byte and interpretation of its values is not predefined, but left to the user application.

Command format is:

Q aa <cr>	as sent by the serial host, and
Q aa ss <cr>	as sent by reply from the CY233, or
Q aa ss ss <cr>	as reply from the CY233 with two values

Note that the "Qaa" header will not be sent in the Echo None case. Also, this command sends the "ss" serial error status as a hex (two character) value in both ASCII and Hex modes. The error bits of the first serial status byte are the same as those for the parallel status byte, and are repeated below. Note that Bit 7 is added as a status indicator, and will not be cleared when it is read. The bit is set when the parallel Sense command is loaded, and enables the passing of the U token message to the parallel side.

Bit 0	Received Parity Error
Bit 1	Receive Buffer Overflow
Bit 2	Invalid Command Format
Bit 3	Parallel Write Handshake Timeout
Bit 4	Parallel Read Handshake Timeout
Bit 5	Parallel FPL Mode Handshake Timeout
Bit 6	Char Received was not Carriage Return in the Adaptive Baud Rate Setup
Bit 7	Write U Token message enabled

Timeout (T) Command

A timeout command, designated by "T", will allow the network to define the timeout for the parallel handshake. When the CY233 is reset, the timeout value defaults to one bit time longer than the character time at the chosen baud rate. For baud rates faster than 9600, the 9600 baud times are used, giving a multi-character timeout for those rates.

If the local parallel device does not respond to a CY233 data transfer within this time period, an error is set, and the CY233 continues. If the CY233 is writing to the parallel device, the data transfer continues as if the parallel device had generated a proper handshake.

When the CY233 is trying to read from the parallel device, a timeout will abort the read and terminate the command in progress. No read data will be sent in this case. However, the message terminator is sent, indicating the end of the transfer.

The T command allows the network to redefine the timeout period. The message format is:

```
T aa tttt <cr>
```

The "tttt" portion represents a 16 bit time value, sent as 4 hex digits. Each increment corresponds to about 1.1 usec at 11 MHz, giving a timeout range of about 1 usec to 72 msec for the handshake. If the value is zero, the timeout function is disabled, and the parallel device must participate in the handshake for the process to continue. Note that a dead parallel device can hang up a CY233 network in this case.

Periodic Delay (P) Command

The CY233 has a second timing related command, designated by "P", for Periodic delay. This time function may be used when the CY233 is in a Master mode, and sets a time value for which the CY233 will delay between Master mode data reads.

The default is for this delay to be disabled, so the CY233 attempts Master mode reads as quickly as possible when it is in the Master mode.

If the delay is specified, the CY233 will delay for the designated time before attempting another Master mode read. The delay will occur between every successful Master mode read. However, after the delay has expired, the CY233 will sequence through any invalid addresses as quickly as possible, until another read is successful, and generates a Master mode read message. The CY233 will then delay for the designated time before making another Master mode read attempt. Message format for this command is:

```
P aa tttt <cr>
```

The format is similar to the timeout command, but here the time value represents the Master mode delay time. Each increment represents about 72 msec, giving a range of 72 msec to about 78 minutes to the delay. Note that a CY233 that is busy processing other network messages will not delay exactly as long as specified by this command, since other data transfers will effect the internal delay function. Also, if the delay value is zero, the periodic delay is turned off, and the CY233 reverts to the default case, performing Master mode reads as quickly as possible.

The periodic delay allows a network to set the "sampling" rate of the CY233, when connected to a parallel device that indicates a slowly varying function, such as temperature. The CY233 may be left in the Master mode, to automatically read the parallel value, without flooding the network with messages that contain the same data over and over. This operation would allow a host computer to monitor many such sensors, without generating read requests or dealing with a constant stream of redundant data values. It also makes the parallel interface design easier, since the parallel device is not required to switch the CY233 between Slave and Master modes when data values change.

Network Test (@) Command

When the CY233 receives a message beginning with "@", it is simply echoed back, with no local action. No address test is performed, and the echo function occurs in any echo mode, including the echo none case. Echoing will continue until a terminator character is sent. The purpose of this message is to provide a mechanism for testing ring networks. When the host computer that controls the ring sends the test command, it will be echoed from one CY233 to the next in the ring, until it goes all the way around, and back to the host. If the message comes back, the host knows that all nodes of the ring are alive, and all ring connections are operating. If the test message does not come back, a ring connection is broken, or one of the nodes is not operating. While any form for the test message is allowed, we recommend using:

```
@ FF <cr>
```

In a properly operating ring, the message will come back in the same format that it was sent.

Address Sense (S) Command

The sense command performs a special function in a CY233 ring network. The action taken by a particular CY233 depends on whether the first address of the sense message is valid at that CY233 or not. This command is provided as a mechanism for intelligent parallel devices to determine what addresses are present in a ring network. It is also supported for a LAN network, and allows serial computers connected to one node of the LAN to determine what other nodes are resident and functioning. The LAN aspects of this command will be discussed later in this section, and in the section on Local Area Networks.

When the sense message is received by the CY233 in serial form, the address is tested for validity, as with other serial messages. If the address is not valid at this CY233, the sense message is echoed until the message terminator is received. Before the CY233 echoes the terminator, it appends the local address of this CY233 to the end of the message.

This mechanism builds the sense message at each CY233 node of a ring network, eventually resulting in a message that contains all the addresses of CY233s in the ring. Every address is appended to the message in hex form, matching the address format of other messages for the CY233. The sense message is echoed in any mode, including the echo none case. General format is:

`S aa xx ... xx <cr>` as sent around the ring

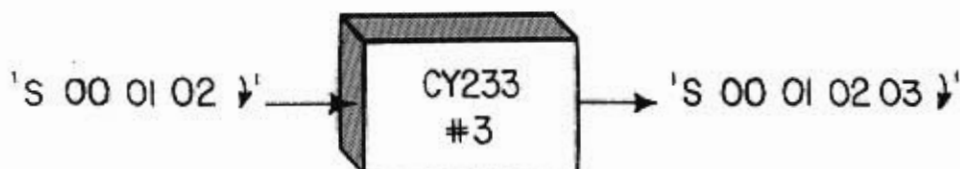
Each "xx" field will represent the address of one CY233, added to the message when it is echoed through that CY233.

When the sense message arrives at a CY233 for which the first address is valid, a different action takes place. At this point, the entire sense message is written to the parallel device, and the message is removed from the network. The message will not be echoed any further, even in the echo all or echo valid modes.

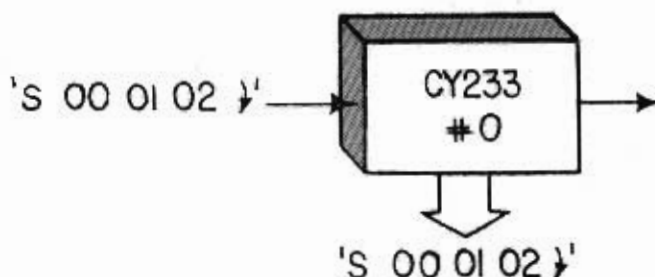
The parallel device will receive the entire message, including the command letter "S" and the first address field, which is the address of this local CY233, as follows:

`S aa xx ... xx <cr>` as written to a valid address

Note that the first address, "aa", is the address of the local CY233 connected to the parallel device, while the other addresses, "xx", are the addresses of other CY233s on the ring. The sense message is started by loading the parallel sense command into the local CY233, as explained in the next section on Parallel Commands.



If the first address does not match the CY233 address, then the CY233 simply appends it's own address to the message string.



Token (U) Command

When the CY233 receives a message beginning with the token command letter, "U", it will perform one of two functions. If the serial error status bit 7 is set at this CY233, the token

message will be written to the parallel device and removed from the network. No address testing is performed on the message, and it is written until the terminator is received and written.

Bit 7 of the serial error status byte is set when a parallel-sense command is loaded into the CY233. This implies that an intelligent parallel device is present that can deal with the addresses of other CY233s on the network. The token command is implemented to allow network communications to occur on a token passing basis.

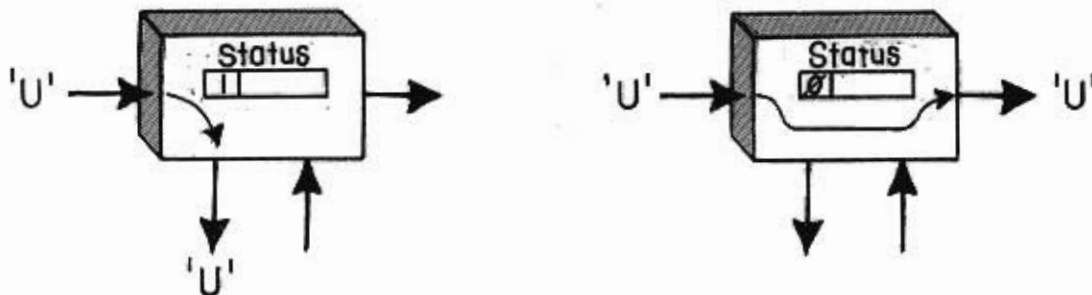
The use of this function is strictly optional. Communications on a network may proceed without any token scheme at all, and we expect that most CY233 networks will not use token messages. The token scheme is also very useful in LAN based communications, and will be discussed in detail in the Local Area Network sections.

Although the token message supports a general format, we recommend using:

U aa <cr>	for the Network Token Message, with
U aa <cr>	passed to the local device

Note that the entire message is passed to the local device, including the command letter "U".

If bit 7 of the serial error status byte has not been set for the CY233 receiving the token message, that CY233 does not pass the message to the parallel device. In this case, the token message is simply echoed to the next CY233 on the ring. This allows a ring to consist of CY233s with intelligent parallel devices connected, that can operate on a token basis, along with CY233 nodes that are not connected to intelligent parts. Those nodes with simple hardware just pass the token on to another node that can handle the message.

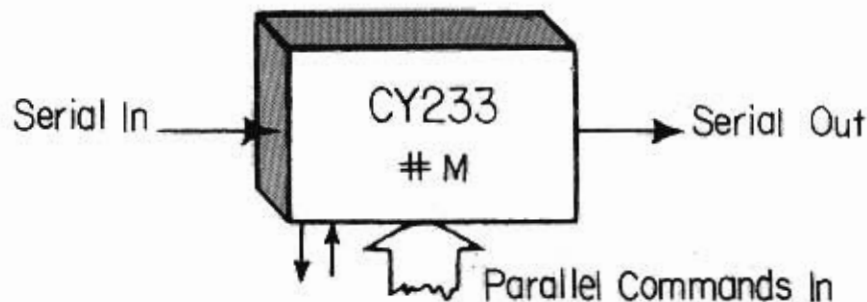


Illustrating the effect of status bit 7 on the token command, 'U'.

Reserved Commands

In addition to the commands already described, the command letters "G" and "O" are reserved for testing purposes. These letters may not be used to start normal command messages, and unspecified results will occur if these commands are tried.

In addition to the serial messages sent to the CY233 over the network, it can also accept parallel commands from the local device connected to the parallel side of the CY233. All parallel commands are read by using the FPL/ line to load the command bytes, with the parallel device supplying an external address of 0FFh, that is, with all address lines high. Any other address will treat the loaded data as part of a standard FPL/ read function, and generate a network read or write message, depending on the level of the R-W/ line.



When the CY233 detects that the parallel device is issuing a parallel command, the command letter is checked for validity, and the parallel command is performed if it is valid. Otherwise, the CY233 will treat the data as a standard FPL/ function, and generate a read or write message with address FF.

All parallel commands involve multiple bytes, so the parallel device must hold the FPL/ line low until the entire command has been read by the CY233. Once the command is entered, the CY233 performs the desired action, then continues with its normal operations.

Most of the parallel commands duplicate functions that are also available through serial messages, such as setting the handshake time out value. Others perform differently when issued from the parallel side. These will be discussed in detail in this section, while the commands that behave the same from both the serial and parallel sides will only be mentioned briefly. Reference the previous section on Network Serial Command Details for more information.

All commands have a format that is similar to the message structure of the serial side. However, the address field is not used, since the CY233 does not perform any address tests with the parallel commands. Thus, the parallel commands use a command letter, a data field (if needed), and a terminator. All bytes are entered using the ASCII code values for the characters. Parameters are entered as hex characters when needed, not as binary values. This will require two byte reads to load one 8 bit value into the CY233.

Initialize (I) Command

The parallel initialize command is equivalent to the serial version, and allows the parallel device to restart the local CY233. Command format is:

```
I <cr>
```

Note that addresses are not used within the structure of the parallel commands.

All Echo (J) Command

The parallel all echo command is equivalent to the serial version, allowing the parallel device to enable this mode on the local CY233. Command format is:

```
J <cr>
```

Parallel Error Enable (K) Command

The error enable command, that enables the CY233 to present any detected errors to the parallel device, can also be issued by the parallel device. The function is equivalent to the serial version. By using this command, the intelligent parallel device can tell the CY233 that it can handle the special handshakes of the parallel error status. Since this is a function between the CY233 and the local parallel device, having this command relieves the network host computer from knowing which nodes should have this function enabled. Command format is:

```
K <cr>
```

Handshake Timeout (T) Command

The local parallel device can also set the handshake timeout delay used by the CY233 in reading or writing with the device. When the CY233 is reset, this delay defaults to one bit time longer than the baud rate setting of the CY233. With this command, the local device can adjust the timeout to match its own characteristics more closely. The function of this command is equivalent to the serial version. Command format is:

```
T tttt <cr>
```

Note that no address is used, but the timeout value is still loaded as four hex digits. Internally, the CY233 will translate these digits into a 16 bit value. The parallel device must use the proper ASCII code value for each digit loaded into the CY233.

Periodic Time Delay (P) Command

The local device can also control the periodic delay used to set a "sampling rate" for the CY233 in the Master mode. This command is equivalent to the serial version, and has a format similar to the handshake timeout shown above:

```
P tttt <cr>
```

Load Second Status Byte (L) Command

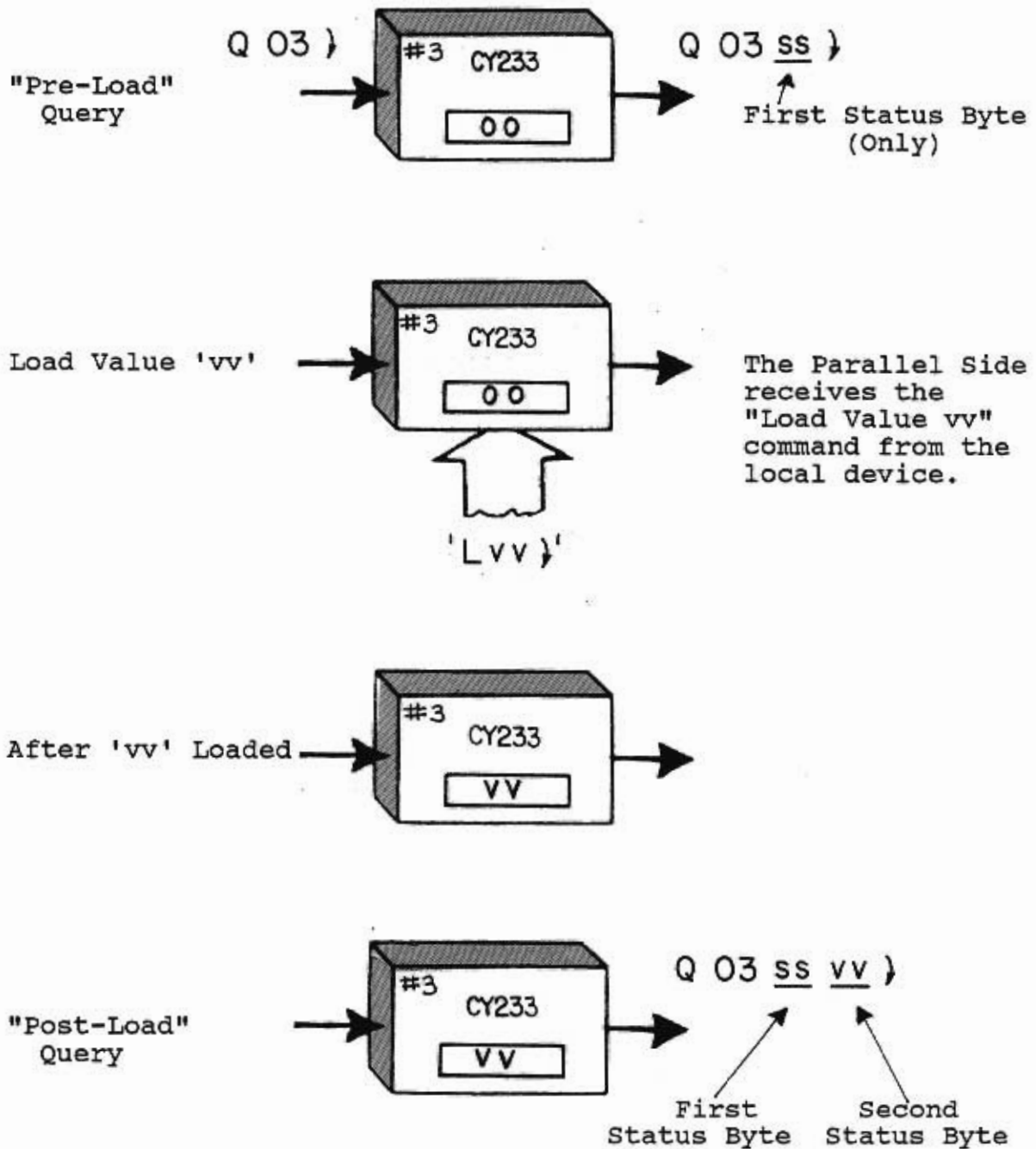
The parallel device has the opportunity to load a special value into the CY233, using the parallel load command, indicated by the command letter "L". This command does not have a serial equivalent. It is unique to the parallel side. If the value loaded is non-zero, the CY233 will send this value out as part of the serial query command. No special meaning has been assigned to this value, and it may be used as desired by the application. One possibility would be a means to identify the nature of the parallel device connected to the CY233. This would allow a serial host computer, that is controlling the network, to determine through the query command what types of parallel devices are connected in the network. Command format is:

```
L vv <cr>
```

The "vv" portion is a two digit hex value that is used as the value of the second status byte. When the value is not zero, the CY233 will send it after the serial error status byte of the query response. When the value is zero, only the error status byte is sent. The second status byte value is set to zero when the CY233 is reset.

Example of "Load Status Byte" (L) Command

Assume the local device at CY233 #3 wishes to Load a value into the second status register as shown below: We examine the response to a serial Query command before and after the Load.



Sense Address (S) Command

The parallel sense address command is related to the serial version. However, the exact function is different from the serial side. When a parallel device loads the parallel sense address command into the CY233, this starts the generation of a serial sense message. The first address of the message will be the local address of the CY233, so when the message is sent around the ring network, it will be removed by this CY233. The response will be to write the sense message back out to the local parallel device, with the addresses of all CY233s of the network appended to the message. Command format is:

<code>S <cr></code>	as issued by the parallel device
<code>S aa <cr></code>	as the start of the serial sense message
<code>S aa xx ... xx <cr></code>	as written back by the CY233

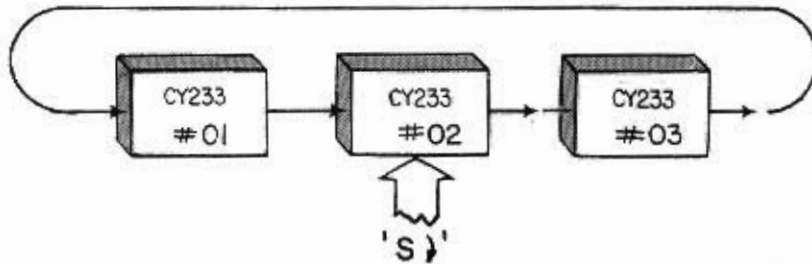
Note that the local parallel address must be reapplied to the CY233 as soon as the parallel sense command is loaded. Remember that the address must be 0FFh to load the command, but the CY233 will try to read the local address just after the command has been loaded, and the FPL/ line is high again. The "aa" address used in the message will be the address read by the CY233 from the address lines. The address will not be tested for validity, and the ACK/ line test is not performed. Only the value on the address lines will be used.

Once the message has gone all the way around the ring, and is received by the local CY233 (on its serial side), a normal address test is performed, and if the address "aa" is valid, the entire sense message, with all the other appended addresses, is written back to the parallel device. This allows the parallel device to know what other CY233 addresses are valid in the network. These addresses could then be the targets of Write data messages generated by the parallel device.

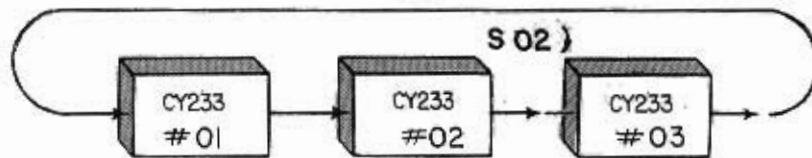
An additional effect of the parallel sense address command is to set bit 7 of the serial error status byte. This enables the CY233 to write any token (U) messages to the parallel device, rather than echoing them on to the next CY233 of the ring. This function is more useful in LAN configurations than in networks of CY233s with parallel devices. However, it may be used if desired.

Example of Use of the "Sense" Command

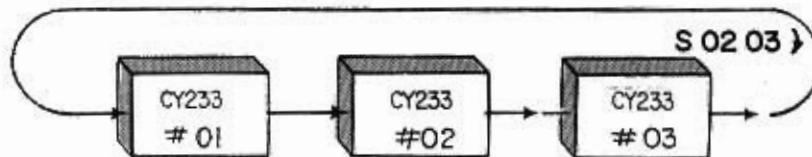
The 'Sense' command is provided as a means of sensing "who" is on the network. In normal use, the sense command is initiated by a parallel device, makes it's way around the ring, sensing the addresses of all ring CY233s, and "gets off" the ring at the same place it was inserted. The following example assumes devices 01, 02, and 03 are on the ring network. We will follow through the case in which device 02 sends a sense command to the ring.



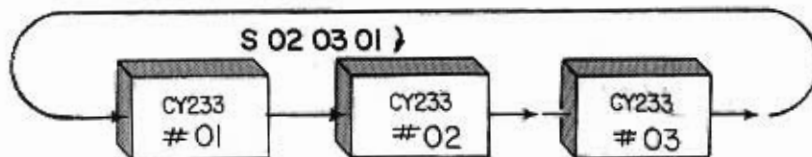
The local device at CY233 #02 issues the sense command, 'S)'.



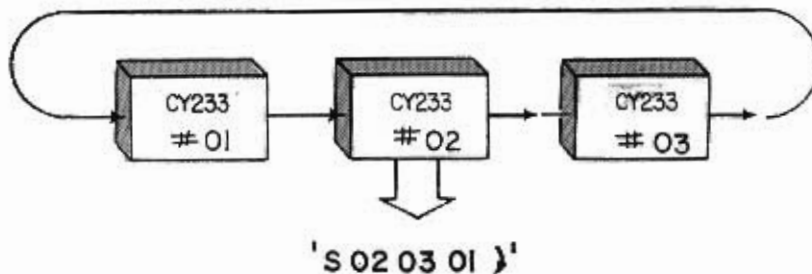
CY233 #02 appends it's own address and outputs the sense command serially.



CY233 #03 receives the command and appends it's address, then retransmits.



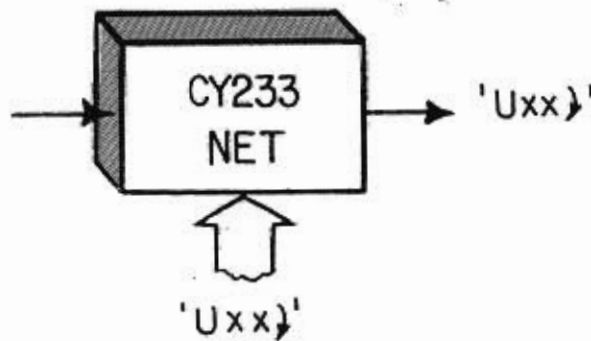
CY233 #01 receives the command and appends it's address.



CY233 #02 recognizes it's own address is first, and takes the sense command "off the ring", allowing the originating device to set up it's "directory" of ring devices.

Token (U) Command

The parallel token command is related to the serial version, but the function is different when it is loaded from the parallel side. When the token message is received on the serial side, the CY233 will write the message to the parallel device, if bit 7 of the serial error status byte is set. Otherwise, the token message is simply echoed to the next CY233. When a token message is loaded from the parallel side, the CY233 will generate a token message to the network.



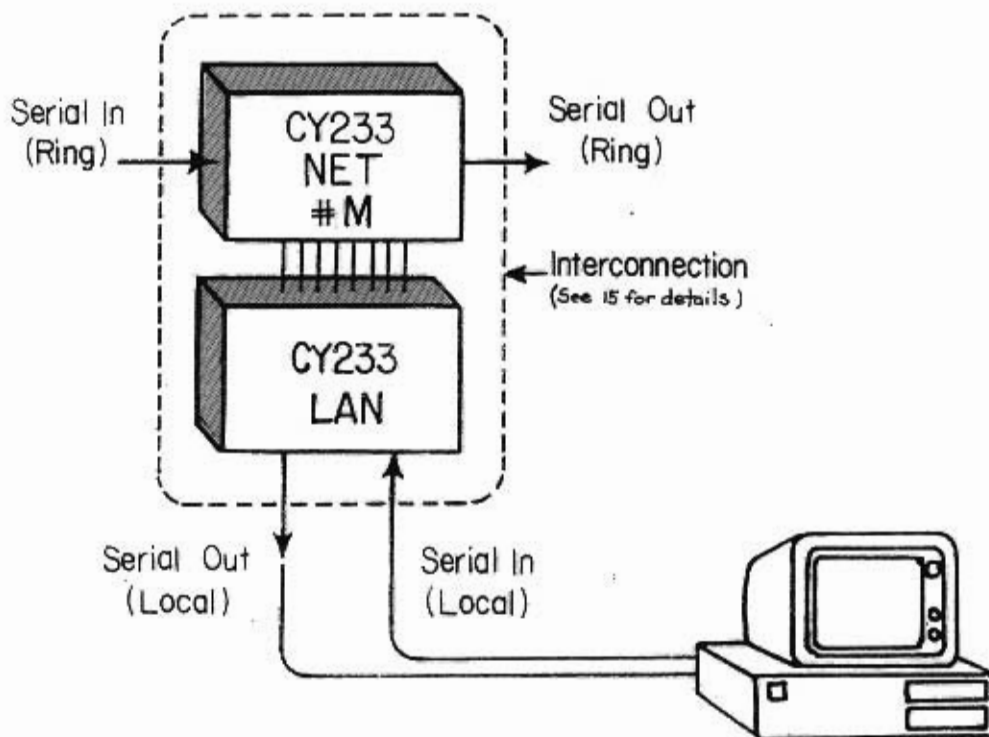
This is the mechanism that creates a token message and sends it to the next CY233 of the ring. There, if an intelligent device is attached, the token is written to that device, indicating that it may make replies to the network. When that device is finished, it resends the token, as a parallel command, so the next CY233 node on the ring has a chance to transmit. This mechanism may be used to implement a general token passing network communications scheme. Command format is:

<code>U aa <cr></code>	as loaded by the parallel device, and
<code>U aa <cr></code>	as sent serially by the CY233

Note that any value may be used for the "aa" portion of the token message. The CY233 does not perform any address tests or expect any particular value for this field.

10 LAN Serial Command Details 10

When the CY233 is operated in the special LAN mode, it is connected serially to a local serial system, and has its parallel side connected to another CY233 that is operating in the normal network mode. The details of these connections are covered in the section on Local Area Networks.



This connection scheme allows a local serial system to be connected to a network of other serial devices, through the CY233 nodes. A standard serial computer can now communicate with any other system also connected to the network.

When the LAN CY233 is operated in a message (non-UART) mode, the LAN CY233 operates with serial messages, similar to the network operation. However, some of the message commands are a little different from the standard network versions, and others should not be used in this scheme. In general, a LAN CY233 will treat a message the same way a Network CY233 does, unless that command has a special LAN function. However, some of the commands, such as sequential fill and display (M and N), transfer (X), and enable parallel error (K) do not make sense in a LAN system. While the LAN CY233 will not generate errors for these commands, they should be avoided. Only the commands listed in the table of LAN Serial Commands should be used.

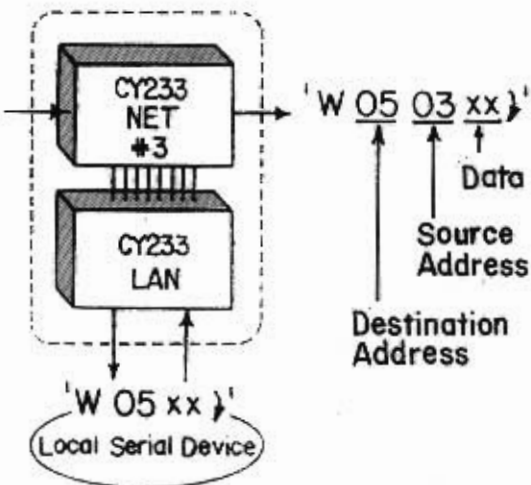
Write (W) Command

The most common command message used in a LAN system is the write message. This is the message that will be sent to other nodes of the network, so it implements the basic communications from one serial system to the other.

When a local serial system generates a write message, the format is similar to normal network write messages, including the command letter, target node address, data portion, and terminator. Note that the address portion of the message contains the address of the LAN node where the message is being sent to, the target address. As a special function in the LAN mode, the LAN CY233 automatically adds the source address of the message into the message structure. General format is:

W aa d ... d <cr>	as sent by the local serial system
W aa ss d ... d <cr>	as sent over the network, and
W aa ss d ... d <cr>	as received by the target system

The "aa" portion of the message represents the target address, while the "ss" portion is the source address, added by the LAN CY233. This allows the receiver to know who sent the message, without that information being part of the message data.



In this example, the local serial device connected (through a CY233 LAN) to network CY233 #03, writes a message to target device #05. The local address (03) is automatically appended to the target address, thus allowing intelligent targets to identify the message source.

The LAN CY233 can also be operated in the Hex Character or Binary modes, with corresponding formats to the write messages:

W aa hh ... hh <cr>	as sent by the local system in hex
W aa ss hh ... hh <cr>	as sent to the target in hex
a d ... d <0B3h>	as sent by the local system in binary
a s d ... d <0B3h>	as sent to the target in binary

Note that in Binary mode, only the write message command is supported, since the message structure does not have a command letter. The LAN CY233 assumes all received messages from the local serial system are write messages, and will send them out the network as implied writes. When they reach their target nodes, these messages will be written to the local serial systems connected to those nodes. Other LAN commands are not supported in the Binary mode.

Read (R) Command

The read command and message structure is identical to that used for the write messages of a LAN system. However, read commands should be used with great care. When the read command arrives at the target node, the Network CY233 of that node will attempt to perform a read operation from the LAN CY233. If the LAN CY233 does not have anything to read, the operation will fail. In general, read messages should only be used in a LAN if there is a host computer connected at the network level. These messages should never be directed at another CY233 node of the network. The message format is:

`R aa d ... d <cr>` as sent by the local serial system

`R aa ss d ... d <cr>` as sent over the network

Initialize (I) Command

The LAN initialize command is similar to the standard network initialize command, and has the same message format:

`I aa <cr>` as sent by the local serial system

The "aa" portion of the message represents the address of the local LAN node. If the addresses match, the LAN CY233 of the node will perform a restart operation, and will redetermine all power-on characteristics. Note that this command only resets the LAN CY233 of a LAN node. The Network CY233 of the node is not effected. A special pass command, described below, can be used to restart the Network CY233.

Query (Q) Command

The LAN CY233 also supports a query command. This command will return error status to the local serial system, since that is what is connected to the serial side of the LAN CY233. Format and function are equivalent to the standard network version:

`Q aa <cr>` as sent by the local serial system

`Q aa ss <cr>` as sent by the CY233 to the local system

The "Qaa" portion will not be sent if the echo none mode is chosen for the LAN CY233. The meaning of the error status byte is the same as for the Network CY233, except that bit 7 of the LAN serial error status indicates if the LAN has detected a timeout for a token response. This function will be discussed later in this section.

Handshake Timeout (T) Command

The LAN handshake timeout command is used to set the timeout value for handshakes between the LAN CY233 and the Network CY233 of a LAN node. The local system is connected to the LAN CY233 by a serial port, so this command will not effect communications between the LAN CY233 and the local system. This command may be useful if the Network CY233 and LAN CY233 are not operating at the same baud rates. This function is discussed in the section on Local Area Networks. Message format is:

```
T aa tttt <cr>
```

Note that this command only effects the timeout of the LAN CY233, not that of the Network CY233 of a node. A special pass command may be used to also set the timeout value of the Network CY233, and is discussed below.

Local Address Query (@) Command

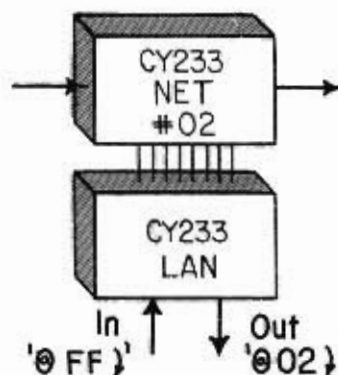
When a LAN CY233 receives the address query message, designated by the "@" character, it performs a different function than that of the Network CY233. Recall that the Network CY233 would simply echo the message along in a ring. The LAN CY233 will read the address that has been assigned to the local LAN node, and return that address value to the local serial system. This allows the local system to determine the address of the node to which it is connected. Message format is:

```
@ FF <cr>
```

as sent by the local serial system

```
@ aa <cr>
```

as returned by the LAN CY233



The address portion of the message must be "FF" when the local system sends the message to the LAN CY233. Any other address will generate an invalid command format error. The CY233 will send the message back, replacing the "FF" portion with the actual node address.

Parallel Pass (P) Command

The LAN CY233 supports one additional special command, the parallel pass command, designated by the letter "P". This command will not perform any function at the LAN CY233, but it will pass a command in parallel to the Network CY233. This allows the local serial system to control some of the Network CY233 functions, even though the local system is not connected in any way to the Network CY233 of a LAN node. The command letter acts as a prefix for the LAN CY233, and the LAN CY233 will pass any following command to the Network CY233. The parallel commands supported by the Network CY233, as described in the previous section, may be sent using the pass command. General command format is:

```
P Command <cr>
```

Where the letter "P" is a prefix to the normal parallel command supported by the Network CY233. Not all parallel commands are appropriate in a LAN system, so the most useful commands will be described below. Other parallel commands, such as all echo (J) probably do not make sense in a LAN system.

If the local serial system needs to know what other nodes are connected in the network, it can send the Sense Address (S) command to the Network CY233 as:

```
P S <cr>
```

When this command is received by the LAN CY233, it will pass the "S<cr>" portion to the Network CY233, and this will start the sense address message around the network. When the message comes around to the originating node, the Network CY233 will write it to the LAN CY233, and the LAN CY233 will send it back to the local system as:

```
S aa xx ... xx <cr>
```

This is analogous to the operation of the command in a network with parallel devices.

Once the sense address command has been sent, the Network CY233 of the local node is also enabled to pass the Token message to the LAN CY233, if such a message arrives at the node. When the LAN CY233 receives the Token message, it also passes this message to the local system as:

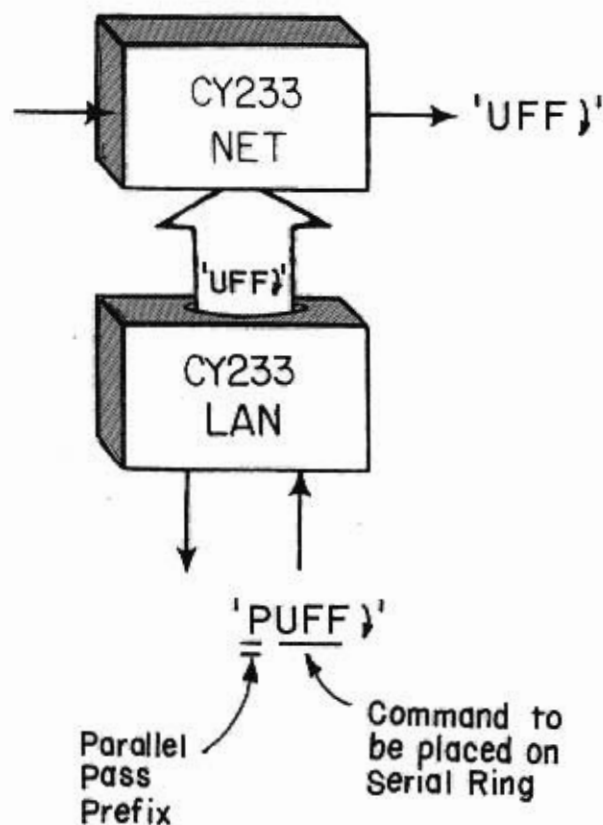
```
U aa <cr>
```

Where the "aa" portion is the address field of the Token message, not the local node address.

In a token based network, the local system that receives the token is allowed to make transmissions to other nodes of the network, normally using the Write messages described previously. Once the local system is finished, it releases the token, so the next node has a chance to transmit. To release the token, it must be passed to the Network CY233 as:

P U aa <cr>

which will cause the Network CY233 to generate a token message to the next node of the network.



Token Timeout Details

After the LAN CY233 passes the token to the local system, it also begins a special timeout function, to insure that the token is not lost to a system that is not functioning. If the local system does not make a response within five times the current handshake timeout period (usually five character times), the LAN CY233 will resend the previous token message on its own, so the rest of the network is not hung waiting for a token. If this timeout occurs, the LAN CY233 also sets bit 7 of its serial error status byte, to indicate a token timeout error. This error will be reported to the local serial system with the next query command response.

Other pass commands that the local serial system might want to use are the initialize (I) and handshake timeout (T) commands, sent as:

```
P I <cr>
```

```
P T tttt <cr>
```

The passed initialize command will cause the Network CY233 to restart itself. Thus, with the I command to the LAN CY233, and the passed I command to the Network CY233, the local system can restart both CY233s of a LAN node.

Similarly, the passed timeout command will modify the handshake timing of the Network CY233, and can be combined with the timeout command to the LAN CY233 to change the timeout values of both devices.

The remaining parallel commands are of limited use in a Local Area Network, and should probably not be passed from the local serial system to the Network CY233.

The CY233 NETWORK Mode

The Network Mode is the normal operating mode of the CY233. In this mode, multiple CY233s may share a single serial line, messages are used for all serial communications, and the various CY233 commands are available.

Each message contains a CY233 device address, and an address test is performed when the CY233 receives the start of a message. This test is performed in two steps. First, the address lines are driven with the specified pattern, and after the ADDR/strobe, the CY233 reads the address lines back and compares them to the desired address. If there is a match, the second test is performed, which is a check of the ACK/ line, after a 70 usec delay. If the ACK/ line is high, the address is considered valid, and the message function continues. If the address fails either test, the message is invalid, and the CY233 ignores the message function, but does echo the message if Echo Invalid or Echo All modes are active.

Parallel side data transfers occur with the specified handshake and timeout functions already described. Transfers may occur in the strobed and non-strobed modes. The message command and chosen character mode determine how the message contents are treated and what data transfers take place.

Master mode reads will occur when the CY233 is idle, with no messages being received or transmitted. The reads will sequence through all possible addresses, in search of a valid address. When a valid address is found, a read is attempted, and if successful, a complete message is generated for that transfer. All master mode reads generate messages with one data value only.

Once the resulting message has been sent, the CY233 will check the next sequential address if still in the Master mode. The sequence continues until the CY233 finds the next valid address. When a single parallel device wants to send multiple bytes, it must wait for the CY233 to sequence through all addresses for each byte read. Multiple bytes are sent with an individual message for each byte value.

For Master mode reads, the parallel device may drive the R-W/ line low to specify a write message be sent with the data value. If R-W/ is not driven, or driven high, read messages result from the Master mode reads.

When the Periodic delay command has been used, the CY233 will delay for the specified time after completing a Master mode read. It will then resume the search for the next valid address, as

quickly as possible, until another read is completed. This sequence will continue indefinitely, as long as the CY233 remains in the Master mode.

The Master mode address scanning is suspended when the CY233 receives serial messages from the network, or when the parallel device requests an FPL/ mode read function.

FPL/ mode reads may read multiple bytes in one message, and use the message address supplied by the parallel device. Reading continues until the message terminator value is read, or until the FPL/ line is raised, or until a read fails due to a handshake timeout. In the ASCII Hex mode, only the FPL/ line or timeout terminates the reading function.

The state of the R-W/ line determines if the resulting FPL/ mode message is a read message or a write message, similar to Master mode reads. The address and R-W/ line values are tested when the first data byte is read, with the DAV/ and ACK/ lines both low. The message is not started until after this first read operation.

If the Error Enable command is received by a serial message, the CY233 will generate a special error status signal when an error is detected. The WR/ line is used for this signal, but it is only driven when the CY233 is idle, and the R-W/ line will be high in this case. For normal write transfers, R-W/ will be low before the WR/ strobe is generated, so the error status is a unique signal combination, and simple external logic could generate a separate error indicator from this line combination.

Once an error is indicated, the parallel device can request an error status byte, which indicates the type of error. This function is also special, using a slightly modified version of the parallel handshake.

For the error status case, the local parallel device drives the ACK/ line low before the CY233 drives the DAV/ signal. (In normal data transfers, the CY233 is in control of the transfer, and the DAV/ line is driven before the ACK/ line response is expected.) Once the ACK/ line has been lowered for the error status byte, the CY233 will output the byte on the data lines, and lower the DAV/ line. This state will be held until the parallel device responds by raising ACK/ again, or until a timeout occurs (generating a new timeout error!). DAV/ will be raised, the data lines will be restored to their state before the error status transfer occurred, and WR/ will be raised as the end of the error indication.

Since this error transfer function uses a special handshake and signal combination, it is normally disabled, and must be enabled by the E command. If the E command is received, it is assumed that the parallel device is intelligent enough to respond to the CY233 error indication.

Console to CY232

This section is entirely concerned with a simple network example, connecting a CY233 to a CRT console. (A teleprinter may also be used).

The schematic shows the necessary connections. The console is shown simply as the DB-25 connector, through which serial data flows. This RS232 data is converted to TTL by the 1488 line driver and 1489 receiver.

The CY233 is shown with an 11.059 MHz crystal and a simple delay for the RESTART signal. To keep the diagram as simple as possible, the connection to the user's parallel device is not shown. However, it is clear the overall circuit is very straightforward.

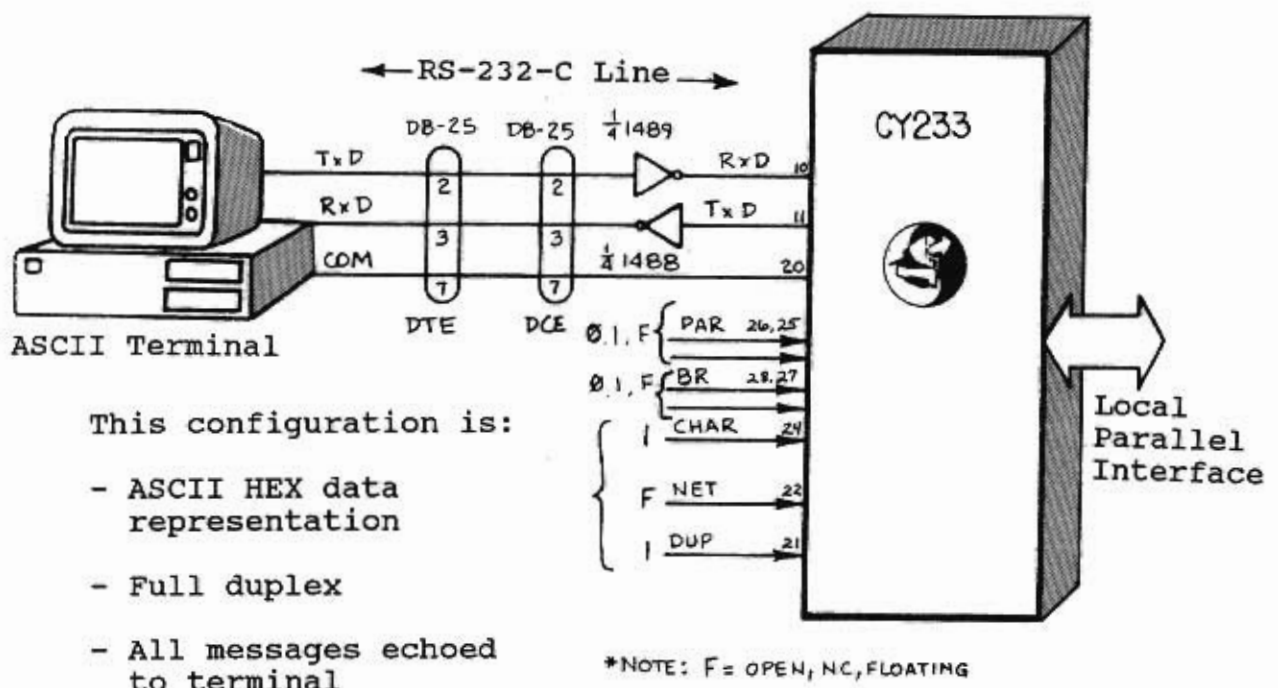


Figure 11.1 Simple Network with ASCII Terminal as Master, CY233 as Slave.

Seven serial mode pins must be set by the user. The schematic notes that a "1" to a mode pin is accomplished by a direct connection to +5V, a "0" to ground, and "F" (free to be either) can be open or a light pullup to +5. The latter is useful on those pins which are to be actively driven: the user can drive through an open collector gate, while not preventing the CY233 from also driving the pin.

At any rate, the user must make specific choices for the mode pins. These must agree with the choices made in the console. For example, assume the console is running

```
1200 baud
odd parity
7 bit ASCII
```

Then, assume the user wants the CY233 to echo valid addressed messages, but not initiate messages on its own. The user should setup the CY233 as

```
pin 28, BDR1, 0 (gnd)
pin 27, BDR0, F (open)
pin 26, PAR1, F (open)
pin 25, PAR0, 0 (gnd)
pin 24, CHAR, F (open)
pin 22, NET, 1 (+5V)
pin 21, DUP, F (open)
```

This is network type "H". If the CY233 is to originate Read messages, then set DUP=0, NET=1 for type "G", which is echo valid addressed messages, master mode. The CY233 will generate a continuous stream of data to the terminal, until it is interrupted by a message from the console to the CY233.

The console should be set at "Full Duplex", which means it does not itself print what is typed. Instead, the computer at the far end of the line, (i.e., the CY233) must echo the message back.

If the console is set at "Half Duplex", it will print what is typed and sent, in addition to printing any response. This will lead to double characters when the CY233 is in an echo mode.

If you are a new user, and you set up the circuit, but do not have instant success, here are some things to check.

1. Is the console working? To verify, place in Full Duplex, and wire its pin 2 (TxD) to its own pin 3 (RXD). It should display what is typed.

2. Do the console and the CY233 have the same baud rate? If no characters are correctly transferred, but a few do seem to be echoed, the CY233 may be running slower than the console. The CY233 is repeating the bit pattern, but not correctly interpreting the characters.

3. Is parity correct? If the CY233 understands and echoes about half of the character set, but not the other half, there may be a parity mismatch. For example, MARK parity can be confused with EVEN or ODD whenever the parity bit would have been a 1 anyway.

The first time user should always start out with this basic network, in order to gain experience with the CY233. The console allows one to see the results, good or bad, immediately. Remember the CY233 does not echo messages until the entire address has been received. After checking the address, it makes its decision about echoing the serial message.

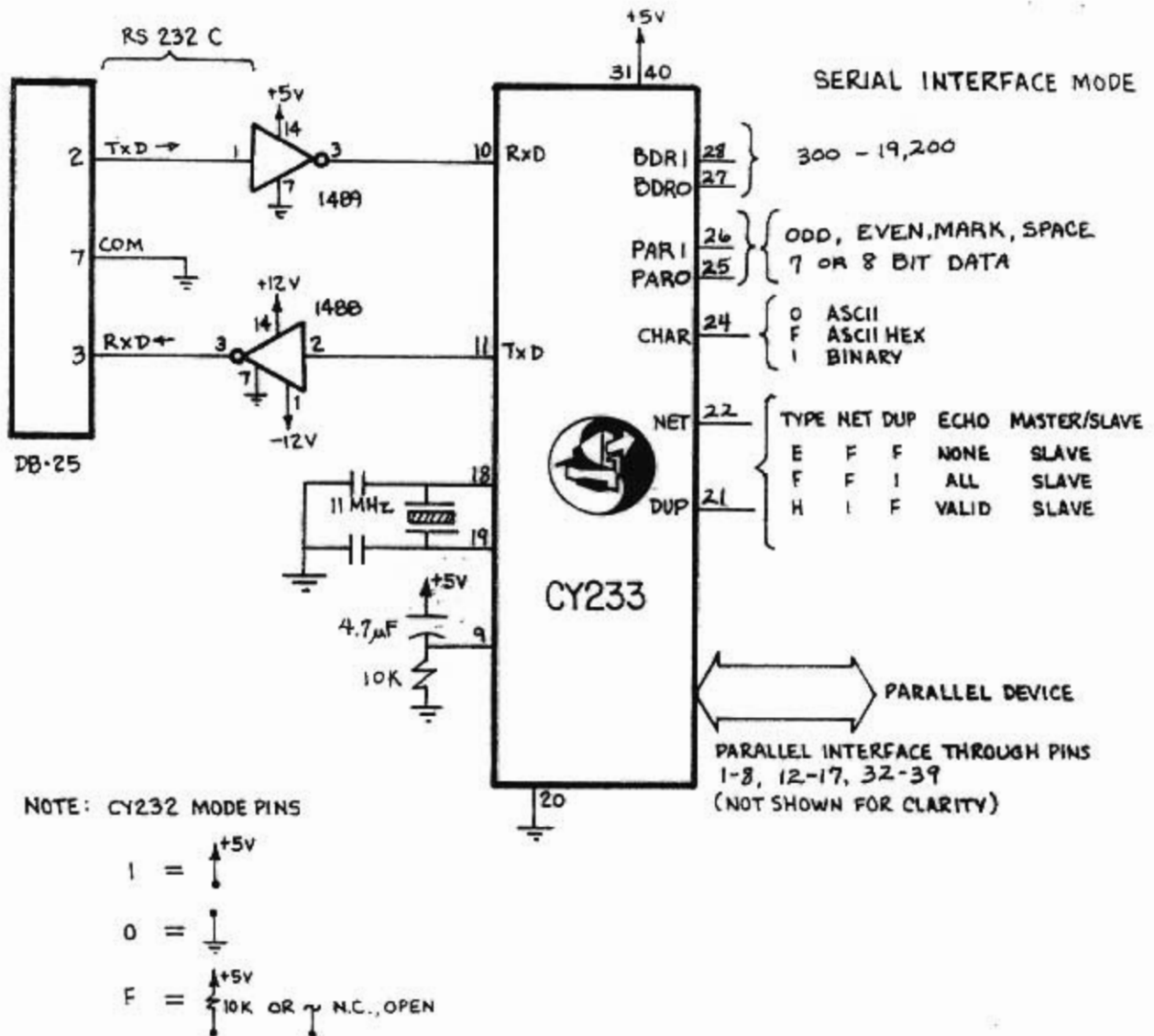


Figure 11.2 Simple Console CY233 Circuit.

An additional CY233 demonstration circuit is shown in the following schematic. This shows LEDs with built-in current limiting resistors as indicators for CY233 outputs on the parallel side. In particular, the LEDs show the address and data buses. This circuit is useful as a first demonstration because the user can readily interpret the LEDs and see if the CY233 is responding correctly to information sent by the console. Remember, a 0 turns the LED on. Notice RD/ and WR/ are tied low so the buses are latched. Also notice DAV/ is tied to ACK/ so the handshake is completed automatically. As an example, if we are using ASCII hex, and we have U-E-D/ tied low, the message

W 01 A2 <CR>

will turn on <A1, D6, D4, D3, D2, D0>, and leave the other LEDs off.

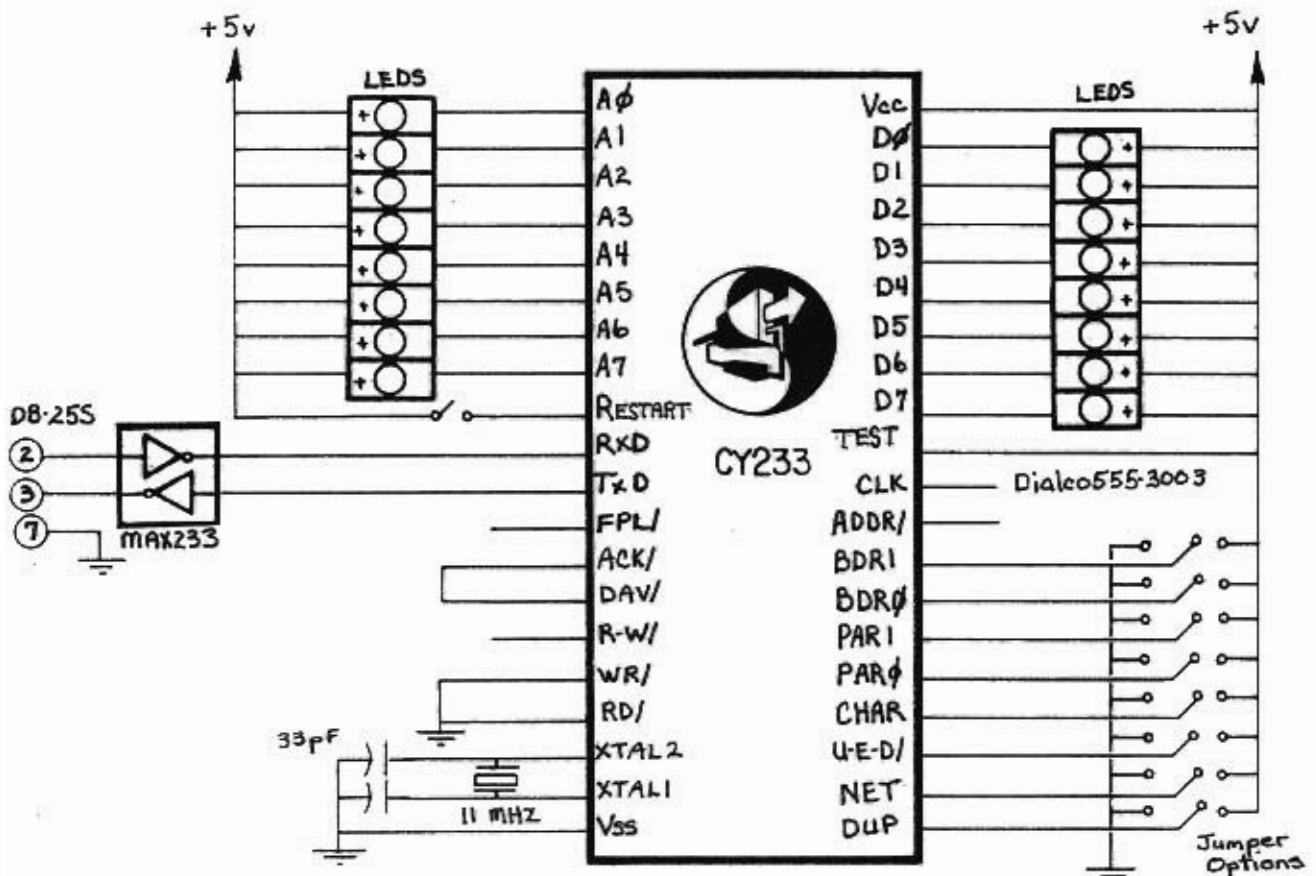


Figure 11.3 CY233 Demonstration Circuit for Prototyping

This section is concerned with a simple example of a serial network. The present example is a RS-232 line with CY233s at each end. There is no console, nor any other computer. The schematic (Figure 9.1) illustrates the situation. The purpose of this network is to move information generated by parallel devices on the left to those on the right, and similarly, move information generated by parallel devices on the right to those on the left. To do this, we use Master modes on both ends. The Master mode will continuously poll its local devices, and each time it finds it can read one of them, it will do so and form a message with the information. This message is then transmitted.

You will notice the schematic says only binary characters may be used. The reason for this is that the binary protocol has no explicit commands. All messages are assumed to be of the Write type. Thus, information generated by one of the Master mode polling operations will indeed be written to parallel devices at other CY233s. The ASCII protocols treat messages generated by a Master mode operation as a Read, and place "R" in the command position of the resulting serial message, unless the parallel device is also controlling the R-W/ line. This is useful when the ultimate receiver is a human at a console, or a microprocessor which can decode the R command and treat it accordingly. However, the convenience of treating all messages as writes, makes the Binary mode a good choice for this type of example.

In summary, if ASCII characters are used in the network shown, then the R messages will be transmitted by the originating Master CY233, but not transferred to a receiving CY233 parallel device, as only W (explicit, as in the case of ASCII, or implicit, as in the case of binary) messages are written to parallel devices. The originating parallel device must also drive the R-W/ line low to generate write messages in an ASCII mode.

Finally, note that since there is no echo, messages are automatically purged from the network. If echoing were allowed, it is possible that a message, once initiated, would circulate forever.

This example illustrates a mechanism for linking parallel devices over a long distance. Using the CY233 to generate serial data could save significant wiring costs if the devices are remotely located. In addition, by changing the range of addresses that are valid for the CY233, additional parallel hardware could be added to each side, without changing the link that ties them together.

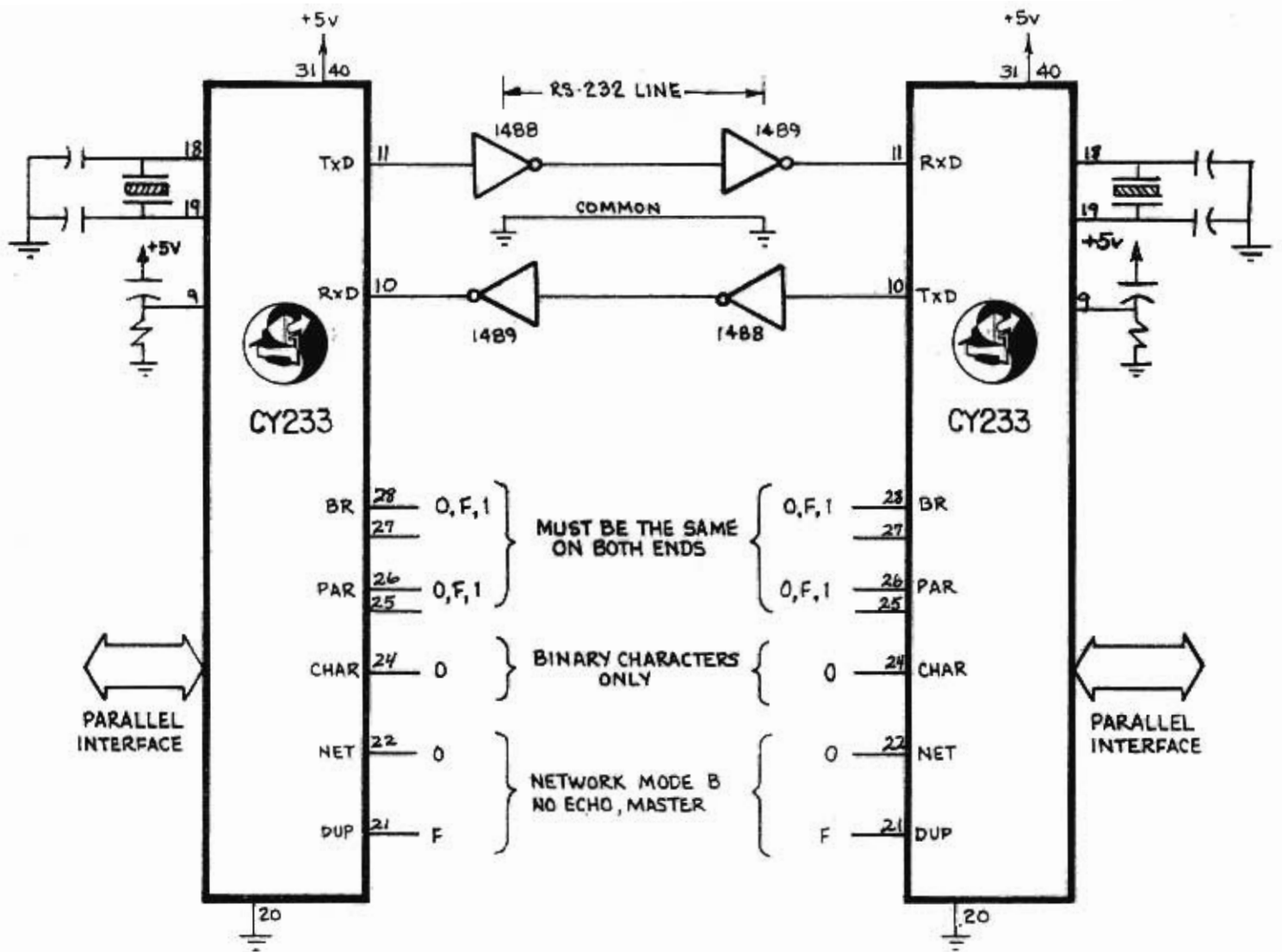


Figure 12.1 Point-to-Point Communication

13 Advanced Serial Side Examples 13

Some previous sections have discussed simple CY233 networks. For example, a console-CY233 connection, or two CY233s, one at each end of a line. In many cases, these elementary networks will solve the problem at hand. However, the CY233 is capable of functioning in much more sophisticated networks. This chapter will introduce the reader to these.

It should be pointed out again that we are separating the serial and parallel sides of the CY233 in our discussion. In general, any parallel configuration can be combined with any serial configuration. The fact that these are made independent by the CY233 is one of its important features.

This chapter will show various serial side networks schematically. Any practical implementation must account for the serial control pins (BDRI, BDRO, PAR1, PAR0, CHAR, NET, and DUP), and the entire parallel side. These are omitted from the schematic for clarity.

Basic Configurations

The CY233 can be used in three fundamental configurations:

- *bi-directional bus
- *dual bus
- *ring

The simpler configurations introduced earlier all are subsets of one of these three. Each fundamental configuration will be discussed in detail in the following.

The bi-directional bus uses only one physical circuit for transmission in both directions, i.e., RXD and TXD. Since it can only be used in one direction at a time, line-turnaround is implemented by devices connected to the bus. This requires that there be a protocol understood by all as to who talks and who listens. The STD, IEEE-796 and IEEE-488 computer buses are bi-directional.

The dual bus has two buses, one for data headed in one direction and another for the opposite direction. The S100 computer bus is an example. A protocol for talking and listening is, in general, still required.

The ring network can be likened to a chain. Information is passed from one device to another, always in the same direction. Thus, only one circuit is required between devices.

Bi-Directional Bus

The bi-directional bus is used when it is necessary to hold wiring cost to an absolute minimum. Figure 13.1 shows one possible configuration.

HALF DULEX, NO ECHO, SLAVE

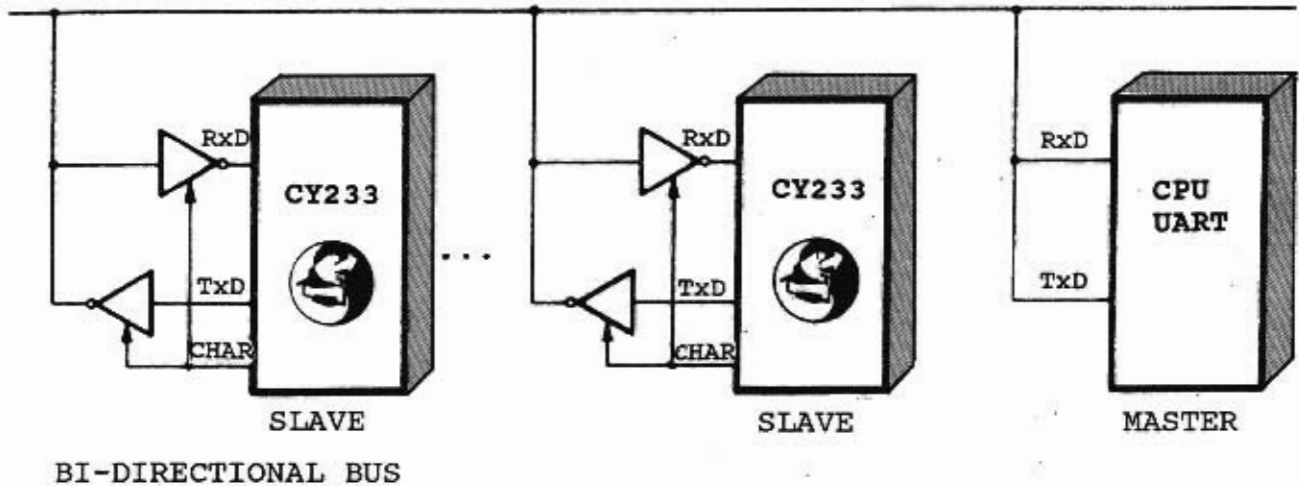


Figure 13.1 Bi-Directional Bus Network

Each device connected to the bus must have a three state driver. Only one driver will be turned on at any one time. Otherwise a collision results and the messages will be garbled. The details necessary to establish the three state driver for a CY233 are illustrated in Figure 13.2. A similar situation exists in connecting an ordinary UART to a bi-directional bus.

The CY233 furnishes a signal, CHAR, to inform the three state driver when the CY233 is actually transmitting TxD data. During these times, CHAR is driven low. Notice that this places a dual function on the CHAR signal, since it is used to select the communications character type when the CY233 is reset. However, after the character type has been determined, the CHAR line may be used to control the three state driver.

In order to select the ASCII Hexadecimal or Binary character modes, you must temporarily drive the CHAR line high or low, for about 1 msec after the CY233 is reset. Then you must let the signal float, if it will be used to enable a three state driver. Note that CHAR floating will select the ASCII character mode at reset, which requires no special action if CHAR is used to both select the ASCII character mode and control the three state driver. Any UART used under similar circumstances must furnish an equivalent signal.

Notice the receiver is also three stated. This is really an enable on RxD, so that TxD data will not be immediately fed back into the CY233. Otherwise, the CY233 would try to respond to its own transmitted message.

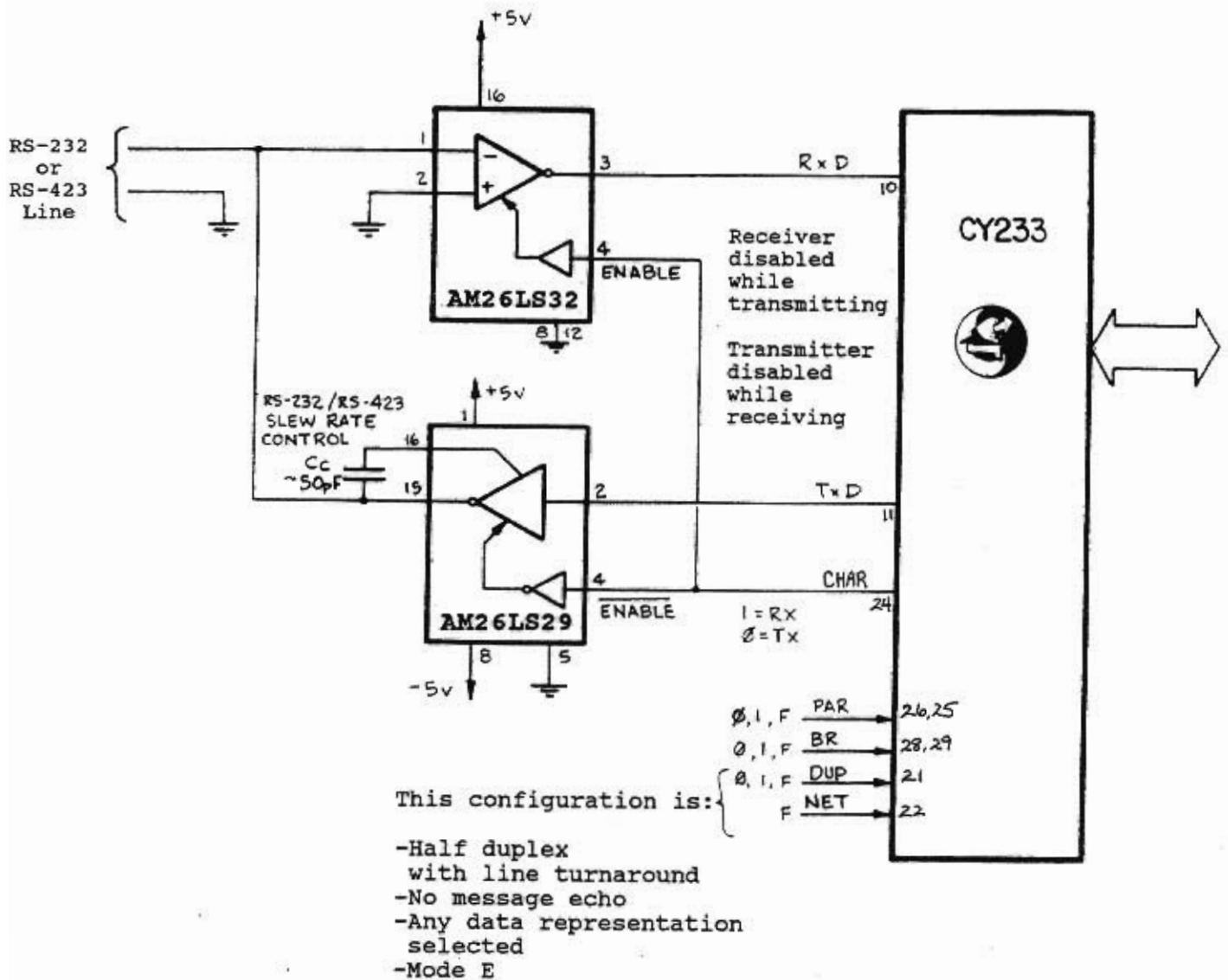


Figure 13.2 CY233 Connection to a Bi-Directional Bus

As a final note, the drivers shown in the schematic can also be used for RS-423, the unbalanced line, high speed, TTL-like standard.

Any time the CY233 is used with three state drivers in a configuration similar to the one shown, it is operating in **half-duplex**. This means it can not simultaneously send and receive data. This limitation is imposed by the serial network, not by the internal capabilities of the CY233.

The safest configuration for the slaves is a no echo protocol. However, if a console is used with a three state driver, the CY233 will likely respond in an echo valid protocol before the user types the next character. It is also possible for a CPU to interlock its transmission with reception so collisions are avoided, but this takes a certain effort and level of understanding.

When CY233s are used as masters, they put information on the line whenever they can. In this sense, it is possible they will "talk too much" and collisions result. If there is only one CY233 as a master, then the remaining CY233s could be slaves with a no echo protocol. In fact, their TxD outputs might not even be used or connected.

Dual Bus Network

The dual bus is conceptually simpler since the slave can be responding to the master while the master is still talking. Therefore, we can now be truly **full duplex**. Three state drivers are still required, since only one TxD source can be on the line at a time. The CHAR signal, as previously described, can be used to control the three state driver. However, in the dual bus configuration, it only needs to control the transmitter, since the receiver is connected to a separate network signal.

FULL DUPLEX, NO ECHO, SLAVE

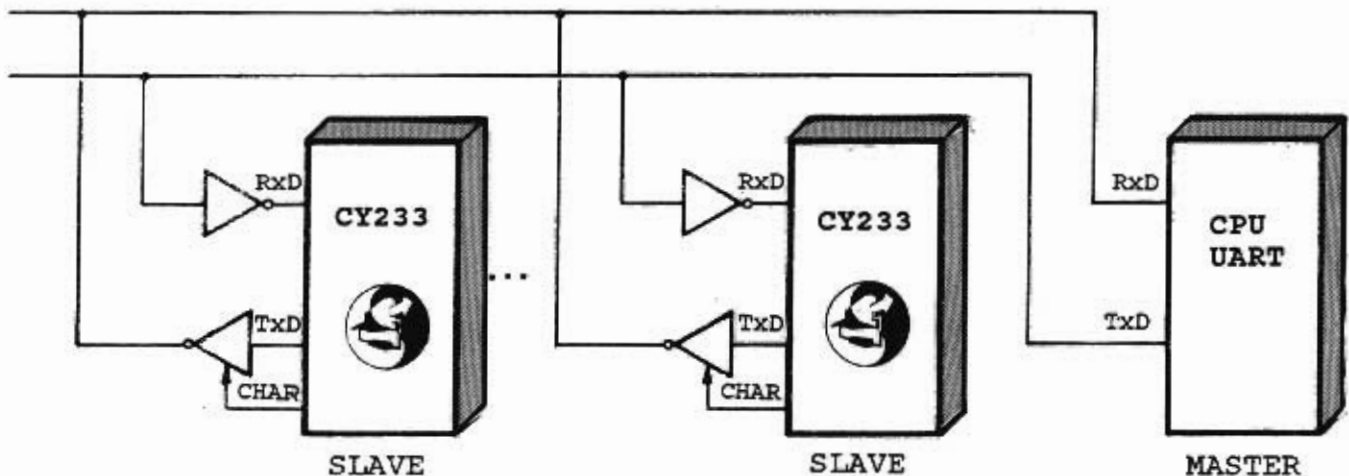


Figure 13.3 Dual Bus Network

This configuration requires that one device be appointed the master, and all others be slaves, if there are more than two devices on the bus. This is because the Tx/D from the master goes to all of the slave Rx/D inputs, and all of the slave Tx/D outputs feed the master Rx/D input. The network master is normally a computer system in this design.

Echo None or **Echo Valid** are legitimate choices for CY233s in the general dual bus configuration.

The simple console-CY233 network introduced earlier is a case of a dual bus with one master, the console, and one slave, the CY233. The point-to-point, two CY233, configuration is another case of the dual bus network.

Ring Network

The ring is the most complex and interesting network. Even though the topology is simple, out one and into the next, the variety of protocol choices is great. The most general operations of the CY233 are possible in the ring configuration, and it is the only type that does not require tri-state drivers, making it the network of choice when RS-232 connections are involved.

FULL DUPLEX, ECHO ALL or ECHO INVALID, SLAVE

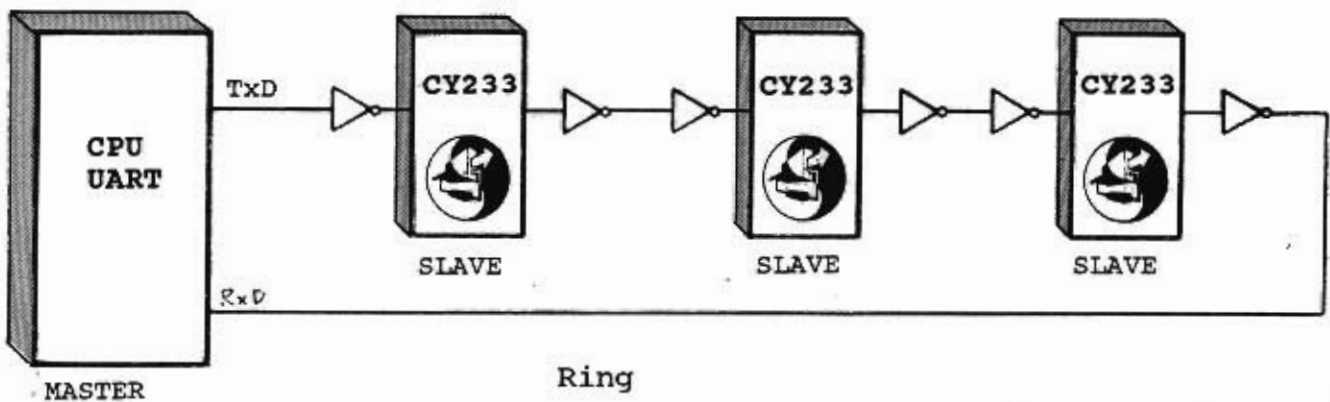


Figure 13.4 Ring Network

The great advantage of the ring is that messages can get on and get off the bus as required, and that many things can be happening at once. Whereas only one transaction can be taking place at a time on the dual and bi-directional buses, many devices can be transmitting and receiving at the same time on a ring. Of course, the penalty paid for this is that it takes longer to go all the way around the ring, so a message does not get to its destination immediately.

The most important concern in the design of a ring is ensuring there is a method to remove, i.e., purge, all messages after they have reached the necessary parties. In other words, we do not want a message to circulate indefinitely.

One way of doing this is to have a CPU as a master, and have all of the CY233s be slaves. The CY233 has two echo protocols, echo all and echo invalid, which are particularly useful for rings. With echo all, any message the CPU sends out will eventually come back to it, and the CPU will know the message was received. This is a nice verification feature. Echo invalid means the CY233 passes on the message if it is not addressed to it. The message is acted upon, and purged, at the first correctly addressed CY233. So messages will travel as far as needed, and no further.

Notice that any message that makes it all the way around a ring in the echo invalid cases implies that the addressed device was not available for the message. This could also be used as a diagnostic of the network, since the host CPU could send out a message with a known invalid address, and expect that message to come all the way around. If the CPU does not receive the message back, it means there is a break in the network, or one of the nodes is hung up somehow.

Conversational reads can be carried out by a CPU (or console) with the echo all or echo invalid protocols. Remember that network read messages will be echoed with the command and address headers in the echo invalid cases. This preserves the format of the messages, allowing them to make it all the way around the ring.

The CY233 internally will not start processing a new incoming message until it has finished handling the last message. Instead, incoming characters are stored in a FIFO buffer for later processing. This feature allows CY233s in a ring to keep messages separate from one another.

The HP-IL interface loop is an example of an echo all ring network with a single master.

In the same sense that the parallel and serial sides are independent, the serial In, RxD, and serial Out, TxD, are also independent, as shown in Figure 10.5.

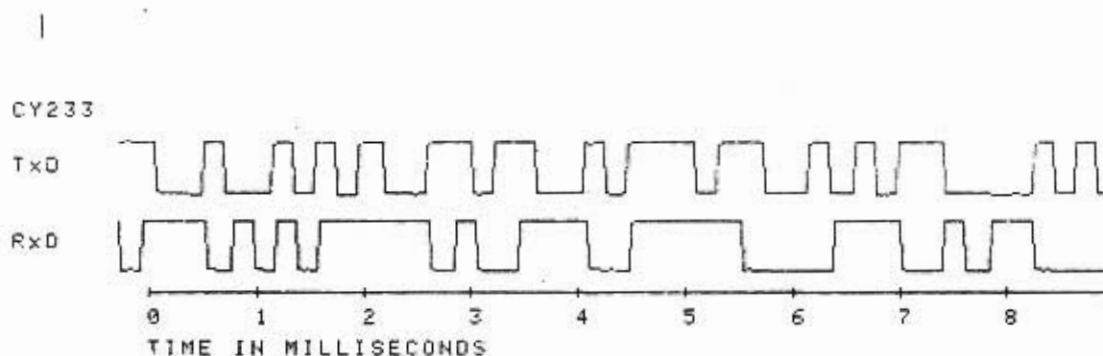


Figure 13.5 Illustrating Serial Receive and Serial Transmit operations occurring independently.

The RS-232 standard is probably the most common form of serial communications available on computer systems today. This interface format has been available for many years, and has been implemented on almost every computer ever built. We therefore expect that most applications of the CY233 will use an RS-232 format for the main controller of the network. This section is dedicated to networks run by RS-232 communications, and the operation of the CY233 in such an environment.

The line drivers of the RS-232 serial port are bilevel drivers, generating positive voltages for a logic 0 and negative voltages for a logic 1. This was explained earlier, in Section 3, on The Serial Side. Note that these drivers are always active, that is, there is no control signal to tri-state the outputs.

Since the RS-232 drivers cannot be disabled, it is not possible for two transmitters to share the same wire. This means the bus network designs will not work with a network of RS-232 devices. One network architecture does work very well with RS-232 serial ports, and is recommended whenever RS-232 ports are used. That is the ring network, in which each node is connected in a chain, with the previous node's transmitter connecting to the current node's receiver, and the current node's transmitter connected to the next node's receiver. This network is shown in Figure 14.1.

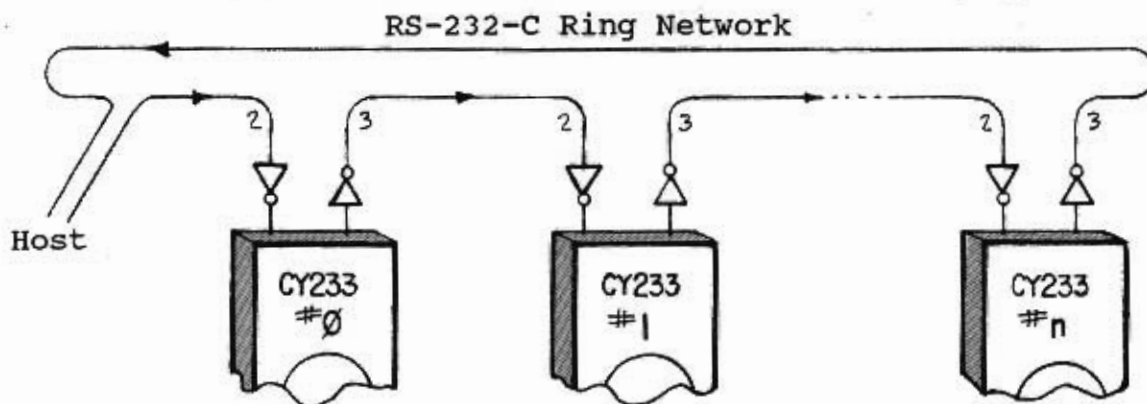


Figure 14.1 RS-232-C Ring Network

The ring network forms a chain of nodes, with messages flowing in one direction through the ring. When the host computer generates messages, they must flow from the host to the desired node, and when a node generates a response for the host, that response must flow through any subsequent nodes until it reaches the host.

When DB25 type connectors are used at the nodes, a simple wiring scheme can connect all the nodes together. A terminal will normally transmit on pin 2, and receive on pin 3 of the connector, while a computer will swap these functions. However,

not all systems follow this pinout. For example, the IBM PC computers use the terminal pinouts for their serial ports. A general wiring diagram is shown below:

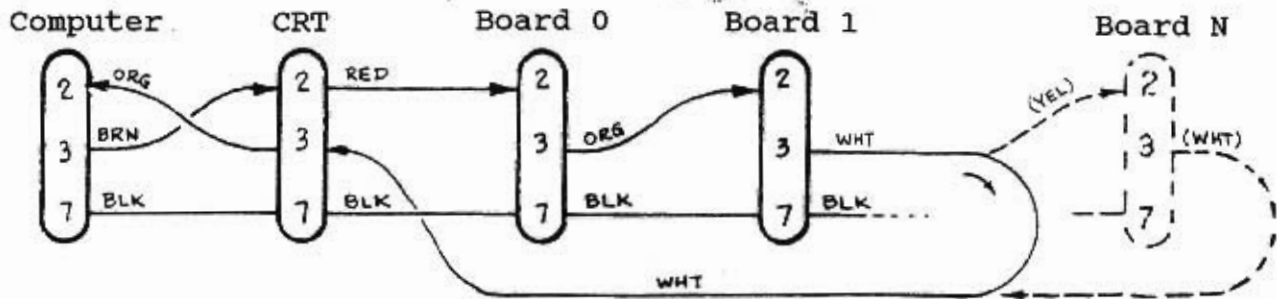


Figure 14.2 Ring Network wiring diagram

The black (BLK) system ground is tied to pin 7 of every DB25 connector. The input to each CY233 node is on pin 2, while the output to the next device on the ring is on pin 3. Progressive nodes run from pin 3 to pin 2, using the next spectrum color for each node, i.e., Node 3 would use YEL from Node 1 pin 3 to Node 2 pin 2. If the white wires are connected together between cables, but are not connected to a DB25 pin except pin 3 of the last node in the network, they may be used as a return path from the last node to the host system. This saves the need to run a separate cable from the last node back to the host, just for one wire.

The CY233 only requires three connections from one node to the next, signal ground, received data, and transmitted data. The modem status signals also supported by the RS-232 interface are not used by the CY233. For some host systems these additional signals may be left open, and the serial ports will function just fine without them. For other systems, some of the lines must be connected to a positive or negative voltage in order to properly activate the serial port. A popular connection scheme is shown below. This allows the host serial port to enable itself, and will require some jumper wires between the pins of the host serial connector.

RTS, pin 4	connects to	CTS, pin 5
DTR, pin 20	connects to	DSR, pin 6, and CD, pin 8

When to Echo

As messages circulate around the ring network, they must be echoed from node to node to insure that they are seen by the nodes for which they are addressed. In most ring networks, each node is assigned a unique address, or group of addresses if multiple parallel devices are controlled by one CY233. In this scheme, there are two echo choices that insure messages will circulate through the ring.

One choice for echo is the Echo All mode. When this mode is chosen, every message received or generated by every node will be repeated to the next node of the network. Eventually, every message will be received by the host system, including those messages generated by the host. When a message reaches the node for which it is intended, the CY233 at that node will perform the message function, such as a read or write operation, and continue passing the message along. When the host receives a message back that it created, it knows that every node has seen the message, and passed it along. This implies that the network is alive, and all connections are functioning, at least at the CY233 level.

Another echo choice is Echo Invalid, in which a message is only echoed if it is not meant for the node receiving it. This mechanism will pass messages along in the ring until they reach the desired node.

If the message is a write transfer, such as a W, X, or F command message, the proper CY233 will perform the message function, and will not echo the message to the next node. This means messages are removed from the network when they reach the addressed node.

If the message is a read transfer, such as an R, Q, or D command message, the proper CY233 will echo the message command and address portions, and then add the data and terminator, forming a complete reply message. This reply message is then echoed by all the subsequent nodes until it returns to the host, with the requested data included.

Thus, Echo Invalid reduces the network traffic by eliminating messages that find the proper node, yet it preserves messages that must send a reply back to the host system. When the CY233 nodes of the network are not busy echoing every message over the entire network, they will be free to respond to local FPL/ read requests and Master mode functions more often than when they must echo every message that comes along. The performance of the network will increase if the Echo Invalid mode is used.

There are additional benefits to the Echo Invalid case. Recall that the Echo All case would return every message generated by the host, if the network was up and running, so that reception of a generated message implied that the network was functioning. This can be very useful diagnostic information.

It is possible to perform a similar function with the Echo Invalid network, if at least one address is not used by a network node. When the host system generates a message addressed to a node that does not exist, that message will be passed all the way around the network, eventually returning to the host. This can be used as a diagnostic test of the network, under full control of the host system. If the message is not returned, it means at least one node is not functioning properly, or that a cable connection has been broken. This situation could cause an alarm condition, in which an operator is instructed to check and repair the network. By periodically generating this diagnostic test

message, the host system can automatically perform checks on the integrity of the network.

In a similar fashion, if the host ever receives a message that was meant for what should be a valid address, that would indicate some problem at that specific node. For example, if the parallel device uses the ACK/ line to keep all addresses invalid while the parallel hardware is "busy", any message meant for that address would test as invalid, and be passed along in the network. This would indicate to the host that the addressed node was not able to accept the message when it was received by that CY233. If this condition persists, the host could again sound an alarm, and indicate that a specific node was not responding.

These diagnostic functions may be very important to the operation of the network, especially if there are many nodes, or long connections between the nodes. The Echo All selections cannot perform the same functions as the Echo Invalid selections in this case, and the Echo Invalid modes offer more flexibility. When no spare addresses are available, the ring can still be tested by using the special "@FF<cr>" command message that will also be echoed all the way around the ring if all connections are good and all CY233s are functioning.

Ring Communications

Suppose we have a small ring network, consisting of three CY233 nodes, set up as addresses 01, 02, and 03. In addition, the network will include the host computer, which acts as the main network controller. This network is illustrated below.

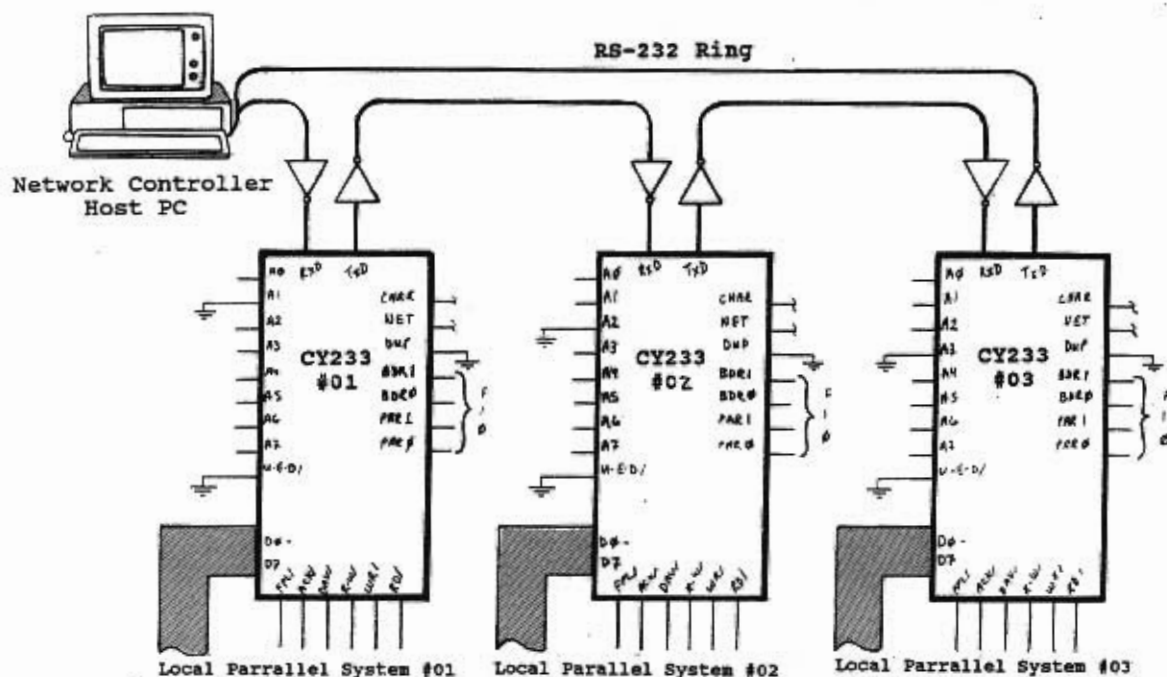


Figure 14.3 A small ring network.

The CY233 nodes have been connected in address sequence order, but this is not required for the network to function. In fact, addresses can be mixed in any desired fashion, and one CY233 could respond to more than one address. When the ring is operated with an Echo Invalid mode, the only general requirement of the network is that each CY233 has a unique set of valid addresses. We will presume that the network is using Echo Invalid, and that the CY233s are being operated in the ASCII character mode.

Before communications is attempted on the ring, the host system might want to test the network to be sure that all nodes are alive and connected. As mentioned in the previous section, with Echo Invalid, this test is performed by generating a message with an invalid address. Also, if the adaptive baud rate selection has been made, the network must be initialized with two carriage returns before any standard messages are sent, so the host controller might send out:

`<cr><cr>`
`W07Test<cr>` or `@FF<cr>`

If all is well, the host should at least receive

`W07Test<cr>` or `@FF<cr>`

back after the message has been passed through the network. When the adaptive baud rate is used, the two initial carriage returns will also be echoed all the way around, setting up all nodes of the network.

The above test message may be repeated anytime the host wishes to check the network again. Note that any invalid address could be used. Selecting one that is not immediately adjacent to the last valid address leaves room for network expansion, without changing the diagnostic checks.

Once the network integrity has been established, the host may begin communications with the parallel devices connected to the CY233s. Sending a message to device two might have the following format:

`W02Message Content<cr>`

We will examine the operations involved in getting this message to node two in greater detail.

Since node 01 is the first node connected from the host, all messages generated by the host will immediately go to CY233 number 01. When it receives the W command letter and 02 address portions of the message, a local address test is performed. The network may continue sending serial data during this test, as serial communications and parallel functions operate independently within the CY233. Characters will be buffered by

the CY233 until it knows what to do with them. CY233 number 01 will fail the address test and will designate the message as invalid.

It will then send the three message header characters to the next CY233, since Echo Invalid is in effect. Notice that the address testing function requires all three characters before the test may proceed, so there is a three character delay between receiving the start of a message and echoing it along if it is invalid. This delay occurs at each node from which the message must be echoed to the next node.

After echoing the message header, CY233 number 01 will continue to echo the data portion as it arrives from the host. This process will continue until the terminator is received. While the CY233 is echoing the invalid message it is considered busy, in the sense that it will not respond to Master mode or FPL/ read requests from the parallel device. These functions are only performed while the CY233 is not processing network messages.

When the "W02" message header is received by CY233 number 02, it also performs an address test. This time the test shows the address to be valid, so the CY233 prepares to transfer the data portion to the local parallel device. The CY233 will initiate the parallel transfer by writing the character to its data bus and activating the DAV/ signal. This will be done as each character is received. Since Echo Invalid has been chosen, CY233 number 02 does not echo the message to the next node, so the message is removed from the network when it reaches the desired node. If Echo All is chosen, CY233 number 02 would transfer the message to the local parallel device and echo it along to the next node. In either case, the parallel device at node 02 would receive:

Message Content<cr>

Any nodes connected after node 02 would not see any portion of the message from the above example. They would remain in the "ready" state, waiting for messages from the network, or local read requests from the parallel devices.

Now suppose that the host computer wishes to query CY233 number 01 for status information, using the Q command message. The host would send:

Q01<cr>

When CY233 number 01 receives the Q and 01 message header, it performs an address test, and finds the address to be valid. It therefore resends the "Q01" header along to node 02, adds the status information as to hex digits, and the message terminator. The message from node 01 then becomes:

Q01ss<cr>

where "ss" is the two digit status information. Notice that Echo Invalid will also echo the message header portion of a valid message, if that message requires a reply from the node. Messages that require such a reply include read, query, and dump commands. Echoing the valid header of these commands insures that complete messages will be passed around the ring, so the reply will make it back to the host computer.

As node 02 receives the complete status query message, it will perform an address test when the first three characters arrive, and find that the address is invalid. At this point it will continue echoing the message, passing it to node 03. This node will also find the address is invalid, and pass the message to the host computer. Thus, the original query message from the host was modified by the proper node, to include the status information, and this modified reply message was circulated all the way back to the host system.

A final consideration in ring communications is dealing with messages that originate at the nodes. There are two possible functions that will generate such messages, both controlled by the local parallel devices. When the CY233 is in a Master mode, it will automatically query the local device for information, and create a message around the data that is read. Each message generated in the master mode will contain the value of one data transfer. For example, if node 03 is in the master mode, and the local device sends the characters:

123

the CY233 will generate a message for each character, so the host system will receive:

R031<cr>
R032<cr>
R033<cr>

If the local device would pull the R-W/ line low, the resulting messages would be write messages, instead of read messages. However, in a ring network, with a host computer as main controller, read messages are a better choice, since the computer knows unambiguously that the read was generated by a specific node.

A second way for the local parallel device to originate information is by using the FPL/ line. This operation is very similar to the Master mode, except that the CY233 does not search for a valid address, and more than one value may be transferred per message. For example, if node 03 again wants to send:

123

to the host, but this time it uses the FPL/ line, the data could be sent in one message as:

R03123<cr>

Since the CY233 is not performing an address test for FPL/ reads, the address lines must be set to the proper address to generate a message that the host computer can understand.

The CY233 will not initiate a Master mode or FPL/ read when it is busy processing a network message, even if the message is just being echoed through to the next node. When there are no network messages to deal with, the CY233 performs the requested read operation.

In the Master mode, the read generates a message with one data value, as explained before. If a network message arrives while the read is being performed, the network message is buffered and held until the Master mode read is complete, and the resulting message is being transmitted. The network message is then acted upon before any more Master mode reads are attempted. This protocol insures that messages are never corrupted by locally generated data.

When FPL/ reads are performed, the local device can force multiple reads for one reply message, by holding the FPL/ line low. The CY233 will stay in the FPL/ read mode as long as the line is low and the local device is supplying more data. Once the CY233 starts an FPL/ reply message, that message will not be interrupted until the reads are complete. If a network message arrives during the FPL/ operation, that message will be buffered until the FPL/ reads are finished.

Since the local device can force an indefinite number of transfers in one FPL/ message, there is a danger that the CY233 receive buffer will overflow with network messages, if the local device keeps the CY233 in the FPL/ mode for too long. It is best if FPL/ messages are kept as short as possible, to avoid any buffer overflow problems. Once the local buffer overflows, any following data will be lost. In current CY233s, the receive buffer is 20 bytes long, so FPL/ messages should be limited to this number of bytes if possible.

From the above discussions, we can see that the ring network is a very flexible and powerful architecture that can take advantage of most of the CY233 supported features. Activity can occur at several ring nodes simultaneously, with some receiving messages, while others are generating local data. The ring structure is a natural choice when RS-232 devices are involved, and the simple message structure of the CY233 allows communications to occur with many nodes in a ring design.

The Local Area Network Controller

The CY233 incorporates a very special operating mode, called the Local Area Network (LAN) mode. The purpose of this mode is to allow local serial devices to function on a CY233 network. In concept, the local serial device, and a CY233 operating in the LAN mode, replace the local parallel device of the standard CY233 network. Most of the special CY233 commands should not be used in this mode, but communications can occur on a message or non-message basis. The CY233 in the LAN mode has its own set of message commands, described in a previous section on CY233 Commands.

Building a local area network from CY233s represents a very cost effective way to connect several computer systems together. The systems do not require any special hardware beyond a standard serial port. Most communications will be message based, so driver software will be required, but the simple message structure of the CY233 keeps this from being too complex as well.

A CY233 local area network will not have the same performance as an expensive, high-end network, such as an ethernet design, but at 19.2K baud or even higher, it could work quite well for a network that requires occasional communications between systems, rather than sharing large amounts of information on a constant basis.

One node of a CY233 LAN network contains four elements. The first element is a CY233 operating in the Network mode. This device connects serially to other nodes of the network, and behaves like the standard CY233 of a standard network. In fact, this Network CY233 does not know it is being used in a special way. Most LAN implementations will connect these devices in a ring configuration.

The local parallel device for this first CY233 is a second CY233, operating in the LAN mode. The second CY233 connects serially to the local serial device (terminal or computer), which is the third element of the node. The LAN CY233 and Network CY233 are connected through their parallel sides in a special back-to-back arrangement, that frees the LAN CY233 serial side for connection with the local serial device.

The final element of the node is a tri-state buffer, with outputs connected to both CY233 address ports. 10K Ohm pull-up resistors should also be included at the buffer outputs. This buffer is set to drive the Network CY233 with the normal address of this node in the network. The LAN CY233 controls the buffer output, and may modify the address to the Network CY233 when messages are being generated by the local serial device. These messages can then be sent to another node on the network.

The signaling between the NET CY233 and the LAN CY233 is quite complicated, however all of this is transparent to the user. The user sees only a local serial channel! The inter-CY233 connections should be made as follows:

Network CY233	LAN CY233	Comments
Data Bus	Data Bus	Connect together, pull-ups required
Addresses	Addresses	Also connect to outputs of tri-state buffer and pull-ups
DAV/	ACK/	Network DAV to LAN ACK
ACK/	DAV/	Network ACK to LAN DAV
R-W/	FPL/	Network R-W/ to LAN FPL
FPL/	R-W/	Network FPL to LAN R-W/
WR/ float	WR/ to OE/	LAN WR/ to address buffer output enable. Net WR/ not used
RD/ float	RD/ tied 1	LAN RD/ tied high. Net RD/ not used

Other signals retain their standard functions, such as baud rate, parity, and character modes. These should be set as required by the local serial device and the network. Note that the Network and LAN CY233s can operate at different rates, and even with different character modes, if one is in Hex mode and the other is in ASCII mode!

A LAN node design is shown on the following page. This schematic illustrates the connections listed above, and uses jumper options for the signal functions that are selectable for each part. The only connections required to the node are serial cables for the network and local serial device, and the power supply for the node itself.

Some care must be exercised when the two CY233s of a node are not operated at the same baud rate. The handshake timeout values may require modification when the Network and LAN CY233s are running at different baud rates, since the baud rate ultimately determines how fast a CY233 can process data transfers.

Since handshake timeouts for rates faster than 9600 baud use the 9600 baud times, it should be possible to operate the LAN CY233 at 9600 baud to the local serial device, while operating the Network CY233 at a faster baud rate. As long as messages are not long enough to overflow the internal CY233 buffers, this would pass messages more quickly between the nodes of the network than between the network and the local serial devices. The network should be able to handle more messages in this case than if both CY233s at a node work at the same rates.

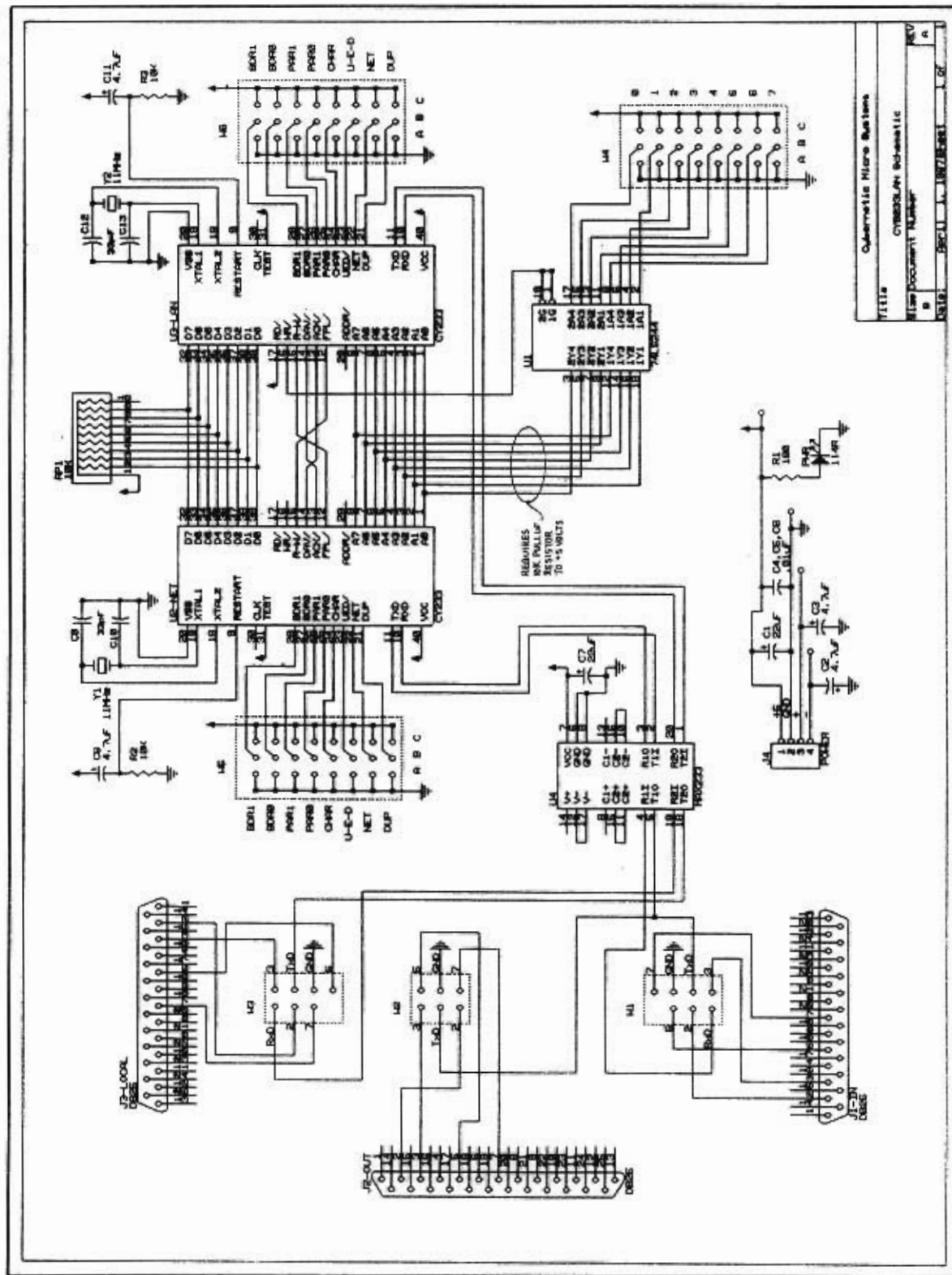


Figure 15.1 Local Area Network node design.

The LAN CY233 is put into the LAN mode by tying its RD/ line high. This function is tested when the CY233 is reset. In this mode, the CY233 understands that it is connected to another CY233 through the parallel device lines. The LAN CY233 changes its handshake protocol in order to successfully transfer parallel data between itself and the Network CY233.

Since data flows from the network into a node in serial form, and from a node to a local serial device in serial form, there will be no Master modes in a CY233 LAN configuration. The Network CY233 must be used in a slave mode, and the LAN CY233 ignores master mode, working only in slave mode, but with the various echo functions still enabled. Note that the LAN CY233 would echo back to the local serial device if an echo mode was enabled.

The Host Ring: A LAN with Host Computer as One Node

The LAN CY233 may be operated in two major sub-modes. These modes may determine the basic architecture of the network. The first type uses the LAN CY233 in the UART mode, selected by tying U-E-D/ high, as for the standard UART mode. This operation requires a certain design for the entire network, in which there is a host computer system node at the network level. This design is similar to a standard CY233 network, where some number of CY233s are connected to the serial interface of a host system. The design is shown in the following figure:

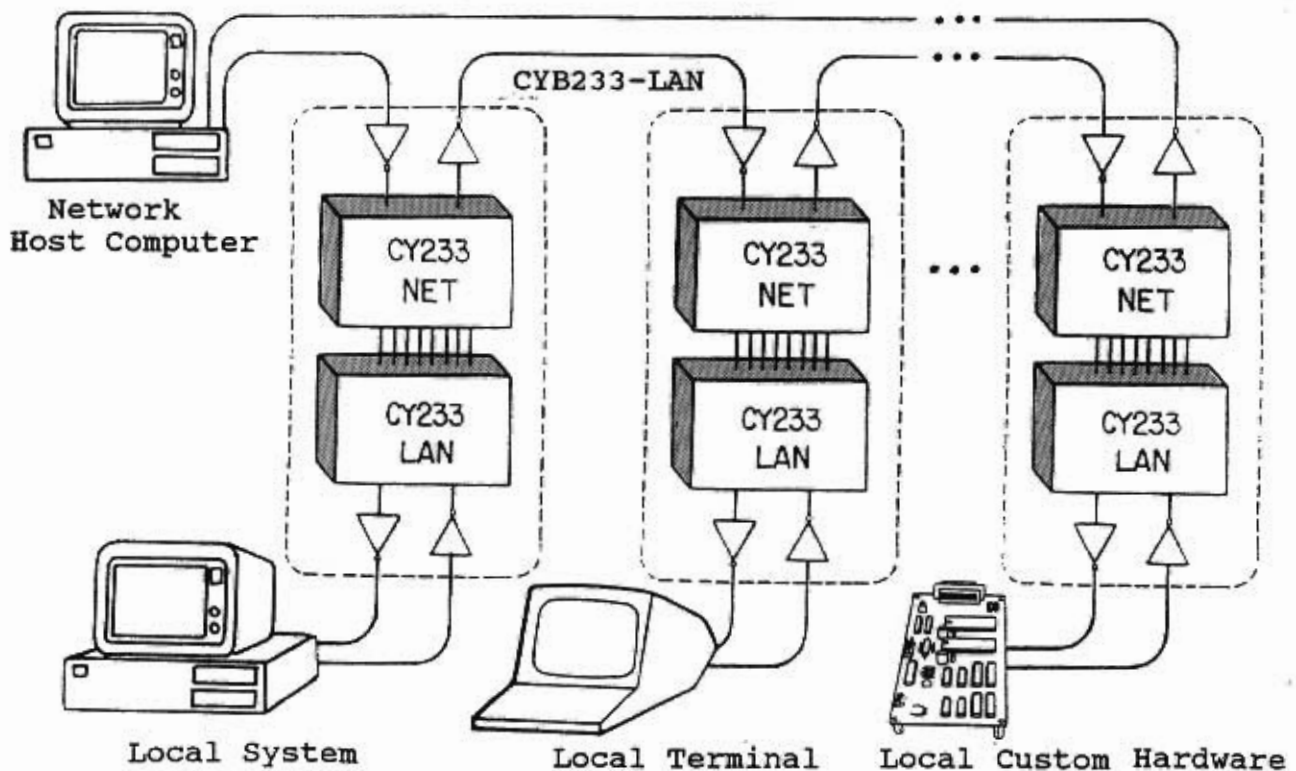


Figure 15.2 Host Computer at the Network Level.

In this type of network, the host computer becomes the network controller, and is responsible for controlling the communications between the various nodes. The advantage to this design is that the local serial devices may be simple terminals, computers, or special serial hardware that are not aware of the network design. The local devices simply communicate with serial data streams, and the host system identifies the source and destination of the messages.

This architecture also allows the mixing of CY233 LAN nodes, connected to serial hardware, with standard CY233 nodes that connect to parallel hardware. Network communications and local data transfers look very similar to the local devices, except some expect parallel data, while others expect serial data.

When the host generates write messages to a network node, the data are transferred in parallel between the appropriate Network CY233 and its local LAN CY233. The LAN CY233 then sends the data serially to the local serial device. Since the LAN CY233 is operating in the UART mode, the local device sees only the data, never any message structure or addressing functions.

For example, if the network node is set as address 01, the host would generate standard Write command messages to send serial data to local serial device number 01, as shown below:

Host Controller Sends

W01Message to Number One<cr>

and Local Serial Device 01 Receives

Message to Number One<cr>

When the local serial device sends data to the LAN CY233, the data are transferred in parallel to the Network CY233, where a read message is generated around each byte value transferred. The address used for the read message is the node address supplied by the node address buffer device. This allows the host system to identify the source of the data, even though the local serial device never sends messages, but only data. This operation is analogous to the standard CY233 network, operating with parallel hardware in a Master mode.

The origination of local serial data is shown below. Again, the local serial device only deals with data bytes, while the network and host controller system deals with CY233 messages.

Local Serial Device 01 Sends

A Reply<cr>

and Host Controller Receives

```
R01A<cr>
R01 <cr>
R01R<cr>
R01e<cr>
R01p<cr>
R01l<cr>
R01y<cr>
R01<cr>
```

Note that the last message received by the host will contain only one carriage return, since it is read as the terminator byte by the Network CY233.

The Peer Ring: A CY233 Based LAN

The second major sub-mode for the LAN CY233 is to operate in a message mode, similar to the Network CY233. In this mode, the local serial device sends data in the standard message format of the CY233, and receives data in a slightly modified message format. An advantage to this mode is that the network architecture no longer requires a host controller system as one of the nodes. The network can consist of entirely CY233 based nodes, with a local serial device at each node! This architecture is shown below:

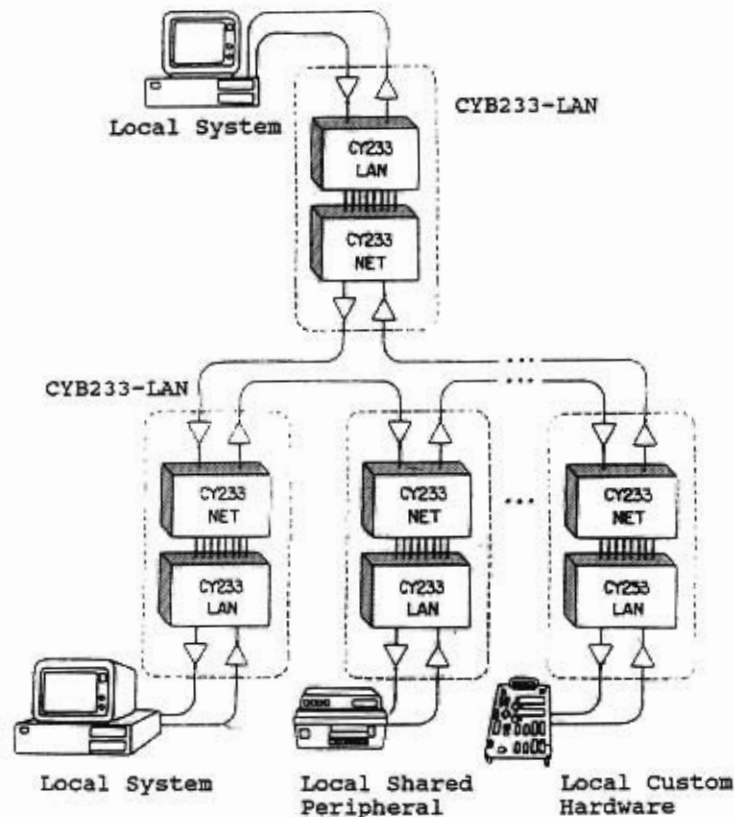


Figure 15.3 Network of only CY233 nodes.

In this design, all the local serial devices communicate using CY233 messages. This allows any serial device to send a message to any other serial device in the network. Since there is no host computer at the network level, the Network CY233s should be operated in an Echo Invalid mode, so messages are removed from the network when they reach the proper node. Otherwise, a message generated for the network might circulate through the network indefinitely!

When a local serial device wishes to send a message to another node on the network, it generates a write message, with the address of the target node as part of the standard message header.

The LAN CY233, connected to this local device, receives this message, and transfers it to the Network CY233. As part of this function, the LAN CY233 disables the local address buffer, and forces the address lines to match that supplied by the message.

The Network CY233 then reconstructs the write message, with the target address, and sends it out to the network. When the message reaches the desired target node, it is transferred to the target serial device as described later.

In addition to changing the local address to that of the target node, the LAN device also adds the local address of the source node as the first data values (two hex bytes in the ASCII and Hex modes, but one byte in Binary mode). This function is illustrated below. Assume that the source node is address ss, while the target node is address tt. The messages in ASCII mode would be as follows:

Local Serial Device ss Sends

```
W tt d ... d <cr>
```

Network Message Becomes

```
W tt ss d ... d <cr>
```

When a node receives a write message from the network, the data are transferred between the Network and LAN CY233s. The LAN CY233 then reconstructs the write message to the local device, including the address of the node, as supplied by the local address buffer device. For the messages shown above, the local serial device would receive:

```
W tt ss d ... d <cr>
```

This is identical to the message circulated by the network, and includes the source address as additional information to the data generated by the originating serial device.

In this case, tt is the local address of the receiving node, the target of the message, and ss is the address of the sending node, the source of the message. This allows the receiving device to automatically get information about the origin of the message, as well as the local address and message contents. An intelligent local serial device could then implement communications with several nodes, and not mix up the origin of the messages. The PEER ring allows any device on the ring to unambiguously communicate with any other device on the ring!

As an example, suppose serial device 03 wants to send a message to serial device 06. The message sent by device 03 would be:

```
W06Message to Six<cr>
```

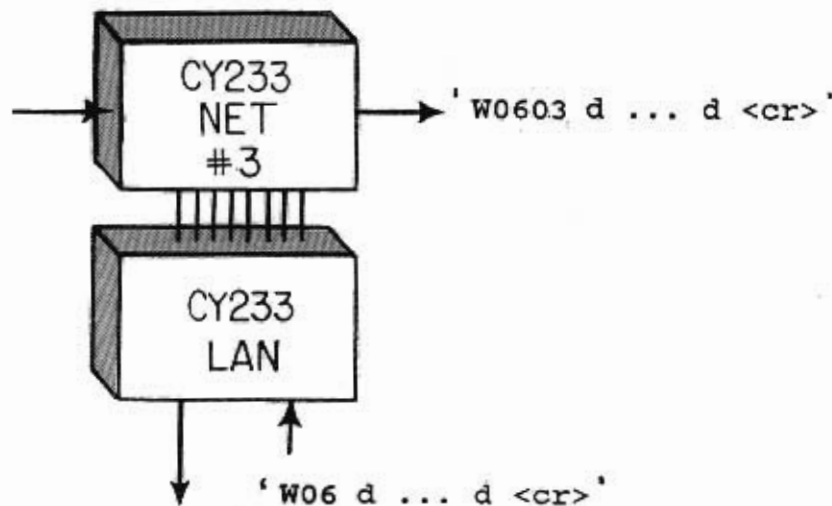
In the transfer between the LAN and Network CY233s at node 03, the message changes to:

```
W0603Message to Six<cr>
```

When the message arrives at node 06, it is transferred to the local device as:

```
W0603Message to Six<cr>
```

This allows the receiving device to get both its local node address and the address of the sending node. Any replies from the receiving node would require this information to generate a proper reply. Notice that the sender is responsible for identifying the destination of the message, but the CY233 automatically adds the source information.



"Who am I"

If the local system would like to know the address of its node, without receiving a message from the network, it can send a special address query command to the LAN CY233 of the node. When the local system sends the message "@FF<cr>", the LAN CY233 will read the node address from the address buffer, and send the value back to the local system. This is illustrated below, where the node address is ss:

Local Serial Device Sends

```
@ FF <cr>
```

LAN CY233 of Local Node Responds

```
@ ss <cr>
```

The entire message is always echoed back, with the "FF" field replaced by the two hex characters representing the local node address.

Other commands, such as the address sense message and token control message may also be used with this network. The commands supported by the LAN CY233 are described in the earlier section on CY233 Commands, and should be reviewed at this time.

The network architecture just described could also be modified to include a host computer at the network level. In this case, any echo mode could be chosen, so long as the host controller removes messages as required. Also, additional levels of communications could occur if a host controller would capture and modify messages of the network. The CY233 design is flexible enough to allow various levels of network implementation, so long as a proper subset of the CY233 commands is used with each type of design.

In the network of only CY233 nodes, if the local serial device sends a read message instead of a write message, the procedure is the same as described above, except that the resulting network message will be a read message instead of a write message. When the read message reaches the target node, the Network CY233 at that node will attempt a local read of the LAN CY233 at that node. If the LAN CY233 has nothing to read, this could cause a problem, so read messages are generally not used in this type of network. However, if there is a host system at the node level, read messages, with the address of the node from which the message was generated, could be sent to the host, and would identify the source of the message.

Other CY233 commands should be used with care in the LAN configuration. The special LAN commands are described in the section on CY233 Commands. Several LAN commands are treated specially, while other commands will behave the same for both the Network and LAN CY233s. The other data transfer commands, such as Fill and Display, should not be used. The timeout commands may be used to modify the handshake times, especially if the Network and LAN CY233s operate at very different baud rates. Also, the query command can give error status summaries. The various command forms and options are listed in the CY233 Commands section.

The LAN mode allows the CY233 to operate in an environment that the normal Network mode cannot handle. This increases the possible applications for the CY233, but puts some restrictions on the allowed message forms. The mode is very powerful, and allows serial devices to be connected in a local network, at costs substantially below those of other network schemes.

The special commands supported by the LAN CY233 allow these networks to be operated in several different modes, ranging from pure data transmissions to fully token based controls. Since the use of any commands is optional, the complexity of the Local Area Network can be configured to suit the needs of the application.

This section gives the user several specific examples of interfacing the CY233 to the user's parallel circuitry. Since the CY233 has a great variety of interface control signals, there may be many different ways of "skinning the same cat." The examples have been chosen to illustrate a number of these ways. The user will be able to mix and match techniques from these examples.

Further Description of Parallel Data Transfers

Several details about the control and handshake signals warrant additional comment.

Recall that the CY233 begins each proposed transaction by checking the validity of the address. If the address is valid, then it goes forward with the read or write of data.

The address bus (A0-A7) is first set up and then the ADDR/ strobe (pin 29) indicates to the user that a new address is present. This can be used like an ALE strobe, if desired. Most applications, however, will not need this line. The address bus remains temporarily latched with the proposed address and then the CY233 reads what is actually present on the address bus. If the user has tied address lines high or low, what is present may be different from what was proposed. In this case, the address is invalid and the transaction with the parallel device is aborted, and the address bus is returned to all 1s.

The UART-Encode-Decode/ (U-E-D/, pin 23) line is checked before each proposed address to see if the address is to be represented to the user as straight binary or 1 of 8. If the decoded mode is selected, pin 23 low, and the address is greater than decimal 7, a modulo 8 representation is used.

Since the user may wish to dynamically allow, or disallow, a transfer, there is a slight delay after the address is presented before it is checked. The R-W/ line (pin 15) is taken low if the proposed transfer is a write to the parallel device. It is left high if a read is proposed. This allows the user to choose, based on which type of transaction is anticipated.

If the user wishes to disallow a transaction, he has two choices. The first is tie or drive the address pins so that the address read back does not match the proposed address. The other choice is to pull ACK/ (pin 13) low, which forces any address to be invalid, due to the second step of the address test procedure. There is more time to decide if the proposed address is valid when driving the ACK/ pin, since the CY233 delays for about 70 usec between testing the address pins and testing the ACK/ pin.

Assuming the address is valid, the transaction proceeds. The exact time between the address check and the actual data transfer will depend on the mode selected and the incoming data rate. In any case, it is at least several hundred microseconds.

Now assume the transaction to happen. If it is a write to the parallel device, data is latched on the data bus, D0-D7. This will be accompanied by a WR/ (pin 16) strobe in the strobed transfer case. The data bus is latched with true data before the WR/ strobe is generated. Subsequently, DAV/ will be taken low to indicate stable data on the bus. At this point the CY233 waits for the user to take ACK/ low, to indicate data has been received by the parallel device. In many cases, the user will have latches loaded by WR/, and ACK/ may be tied to DAV/. If the user does not pull ACK/ low within the handshake time out limit, the CY233 times out and assumes the transaction is complete. At the conclusion of the transaction, DAV/ is taken high.

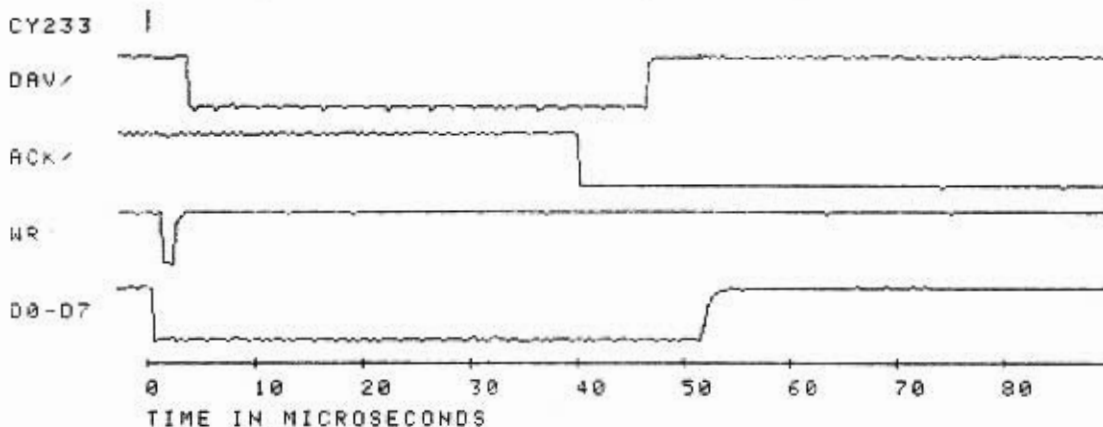


Figure 16.1 Typical write handshake

The user has a choice of a latched or multiplexed data bus. If DAV/ is tied low for the strobed transfer, or BUSEN/ is tied low for the non-strobed transfer, the latched mode is selected. Data will remain stable on the bus until a new transaction is to be attempted, whether it be a read or a write. If the multiplexed mode is selected, the data bus is floated to a three state condition after the transaction (either read or write) is completed. In the multiplexed mode, the data bus is floated between each write transfer of a multi-transfer write message.

If the transaction is to be a read, the DAV/ line is pulled low to indicate that the parallel circuits should place data on the data bus. When this has been done, ACK/ is driven low to indicate so, and the CY233 reads data in, with a RD/ strobe in the strobed mode. If data is available from a simple enable, actuated by DAV/ or RD/, then ACK/ and DAV/ may be tied together. If ACK/ is left floating and the internal timeout occurs during a

read operation, the CY233 will not perform the read. In this case, the serial response of the CY233 is only the message terminator, with no data transmitted. Use of the timeout does not substantially slow down the overall network because it is about the same time as a complete RS-232 character. The data bus is always left in a three state float after a read. DAV/ is taken high after the RD/ strobe. The use of the handshake time delay is sometimes helpful in interfacing to slower parallel devices, such as analog-to-digital converters.

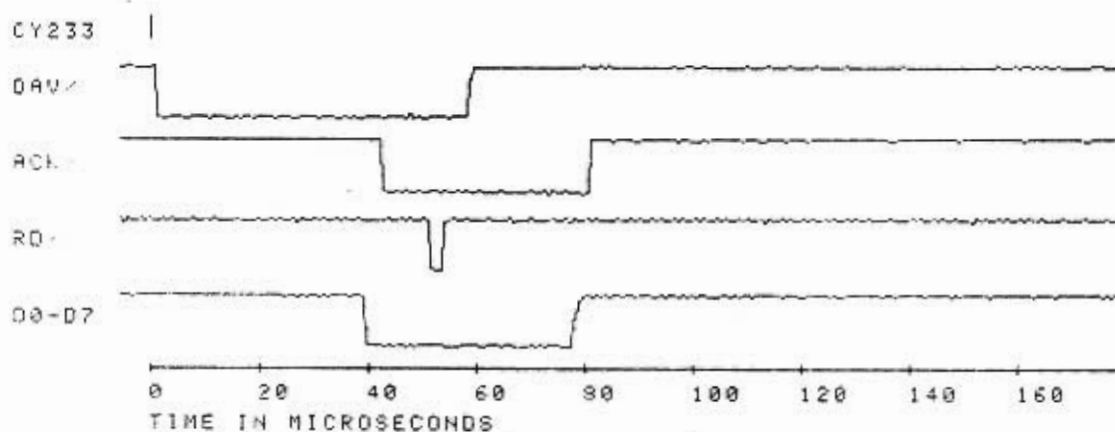


Figure 16.2 Typical read handshake

Example Using an 8212 as a Latched Output Port

Figure 15.3 shows the details of one possible user circuit. While each of the following examples shows only one user device, this is for clarity and does not express any limitation on the part of the CY233. The only thing the user must keep in mind is dividing up the address range, and assigning the requisite block of addresses to each physical device. Each device will then get its own chip select at the appropriate times.

We have chosen encoded straight binary addressing in this example. This allows for a practical maximum of 255 devices (decimal 0 through 254). Decimal 255 should not be used because the address bus will often be floated to all 1's.

If additional addresses are required, the user can implement indirect addressing, where a data port is loaded with a portion of the address and the address bus supplies the remaining 8 bits. In this case, notice only that portion of the address supplied by the address bus can be used to check for validity of the address. For this reason, the address bus will hold the more significant portion of the address.

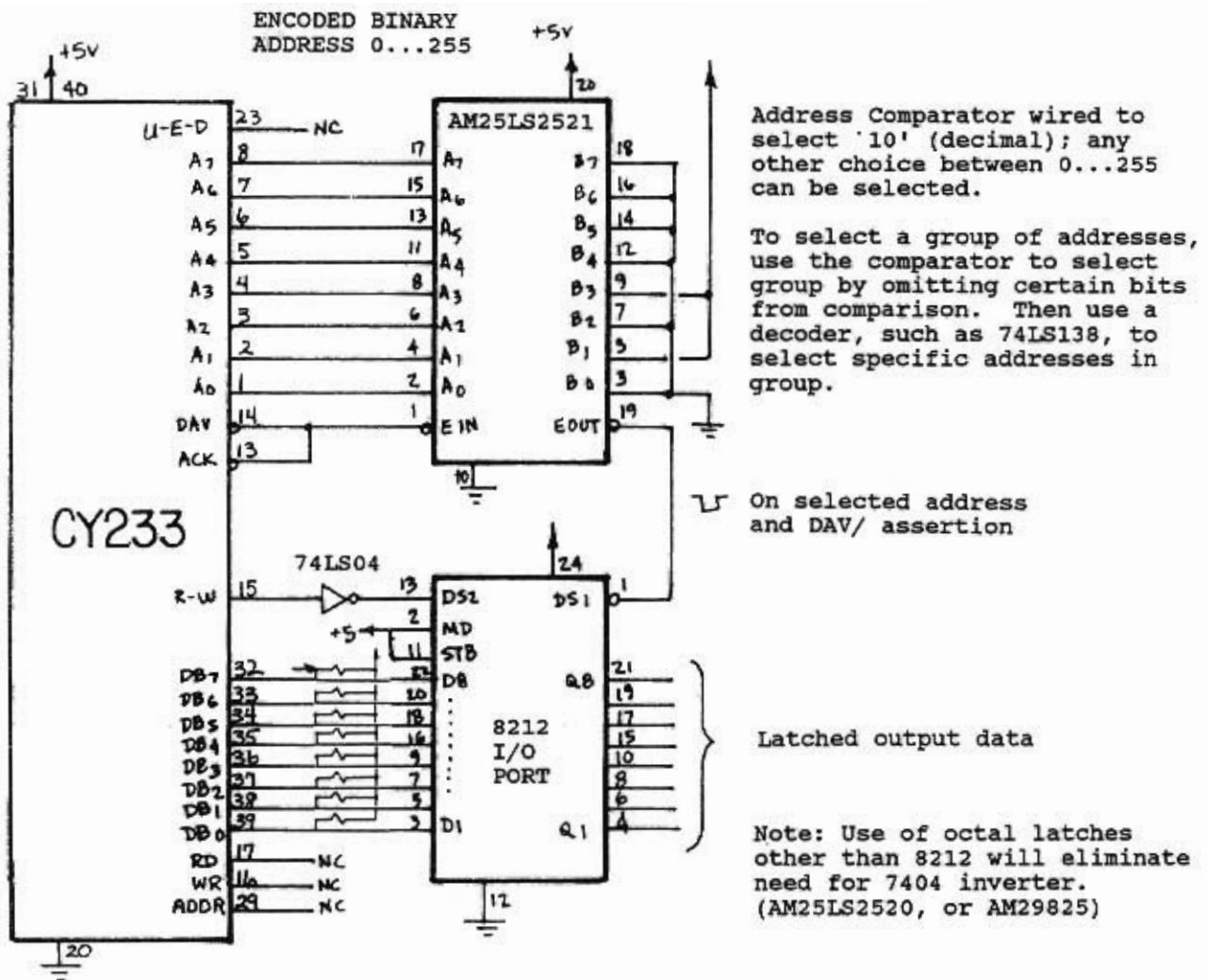


Figure 16.3 Using an 8212 as a Latched Output Port

In the example, the user desired that this 8212 respond to decimal address 10. The Am25LS2521 is an eight bit address comparator, with enable. The desired address is wired into the B side, and the actual address presented to the A side. EOUT/ goes low when the addresses match. EOUT/ is enabled by EIN/, which is connected to DAV/ in this example. Thus address comparison takes place only in the immediate vicinity of the data transfer. Thus this example can not dynamically choose to accept or reject the address. The address is always accepted, if it is the correct number.

The 8212 has two chip selects, DS1/ and DS2. Since we desire that the CY233 write to the 8212, the R-W/ line must be inverted to be asserted high for a write. This is done with a simple inverter. Other latches have two asserted low enables, which eliminates the need for the inverter. When both device selects are asserted, the 8212 latches data from the data bus.

Notice the RD/ and WR/ strobes are not used in this example.

Example Using 8212 as an Interrupting Source

This example is different from the last in that decoded addressing is used and the CY233 is operated in an interrupt fashion, by switching between Slave and Master modes. When the selected address line goes low, and DAV/ is asserted, the 8212 sees one device select. If R-W/ is high, the case of a read by the CY233, then the 8212 is selected. This selection gates 8212 data onto the data bus and also clears the interrupt flip flop.

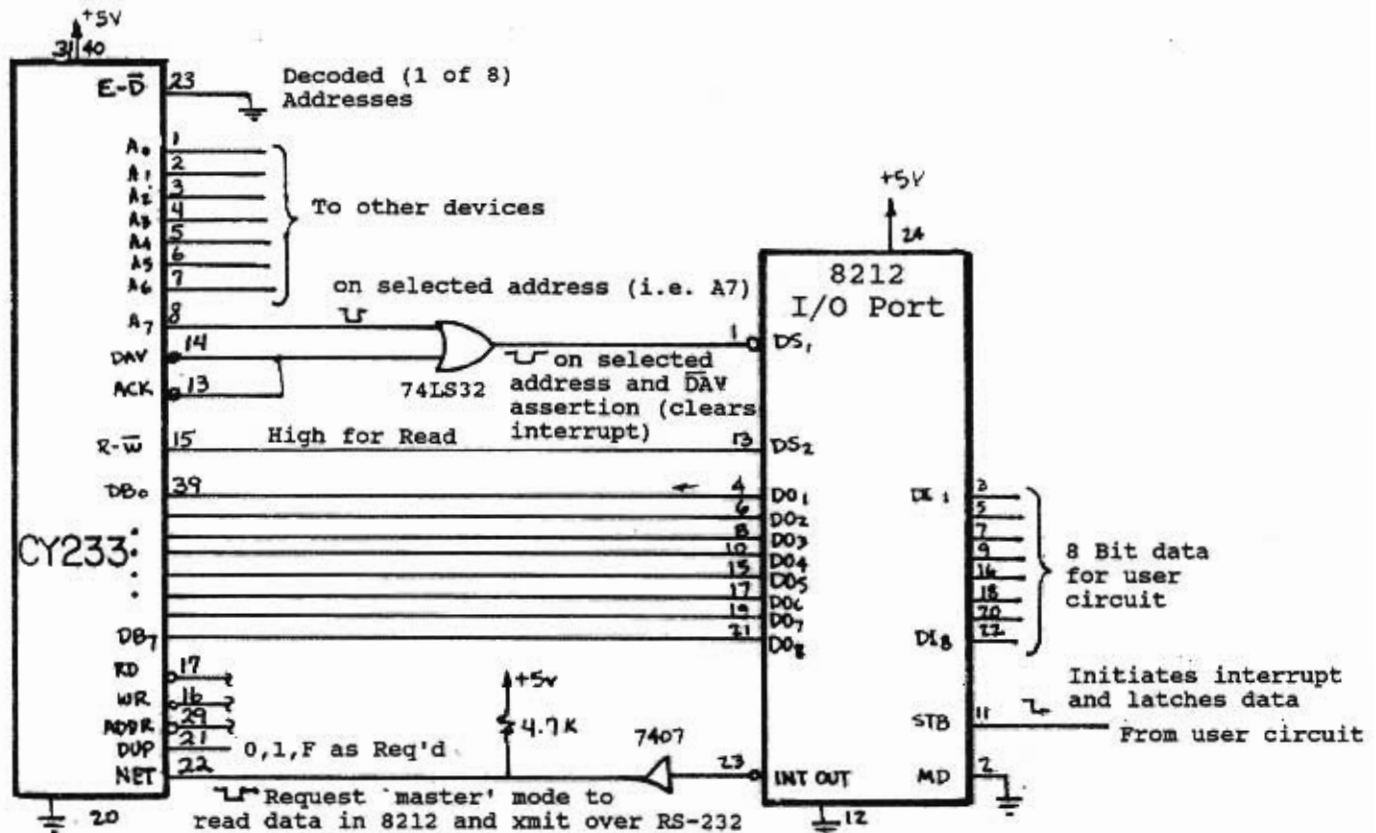


Figure 16.4 Using 8212 as Interrupting Input

The 8212 interrupt flip flop was originally set by a negative going strobe, STB, from the user circuit. This falling edge latched user data into the 8212 and set the flip flop. The negative true assertion of the interrupt flip flop is connected to the NET input of the CY233. The master/slave modes of the CY233 are set up so that as NET is changed from 0 (master) to F (slave), the echo mode specified by DUP does not change. The open collector buffer between the INT/ output of the 8212 and the CY233 is necessary so that when INT is high, the CY233 can test it as F, i.e., the CY233 can establish both a 0 or 1.

As in the previous example, RD/ and WR/ are not necessary for this circuit.

Example of Connecting to an 8255

This example is one of connecting an LSI device to the CY233. It uses the RD/ and WR/ strobes directly. DAV/ and ACK/ are not used, and are tied together for an automatic handshake.

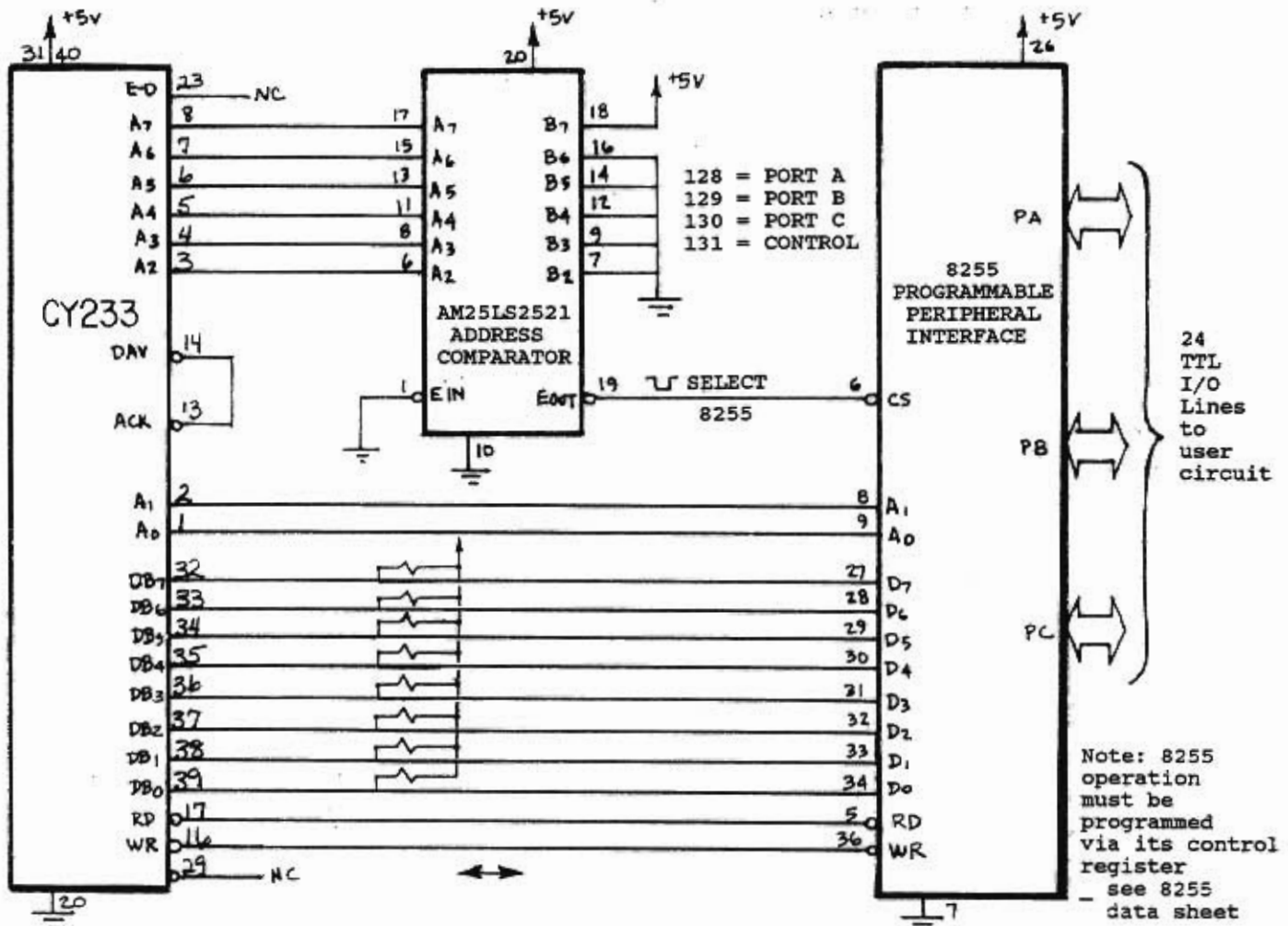


Figure 16.5 Using the 8255 as an I/O Port

Encoded addressing is used. A block of four addresses is set aside for the 8255. The AM25LS2521 comparator establishes the base address, which was chosen to be 128 decimal in this example. The 8255 decodes the low two bits of the address bus directly. All the address comparator must do is detect the base address and enable the CS/ of the 8255. The 8255 connects directly to the RD/ and WR/ strobes. Notice this example has information flowing both ways between the CY233 and the parallel device.

The 8255 is also different from the previous examples in that the operation of the 8255 must be programmed by setting up a control register. The programming of the 8255 via the CY233 is no different from programming it on a CPU bus. The control register is at an address 128+3, or 83h. Therefore, to set up the 8255:

Notice the various 8255 ports can be dynamically assigned to be input or output. However, in using the 8255, be aware there is a "bug" in the 8255 such that its output ports go to 0 when the mode is changed from read to write, or vice-versa.

Example Using the CLK Output

This example shows a utilization of the CLK (pin 30) output, and is also an example of interfacing to an analog-to-digital converter. The ADC in the example is the National Semiconductor ADC0808. This unit has an 8:1 analog multiplexer feeding an 8 bit converter. The analog range is determined by voltages applied to VREF+ and VREF-.

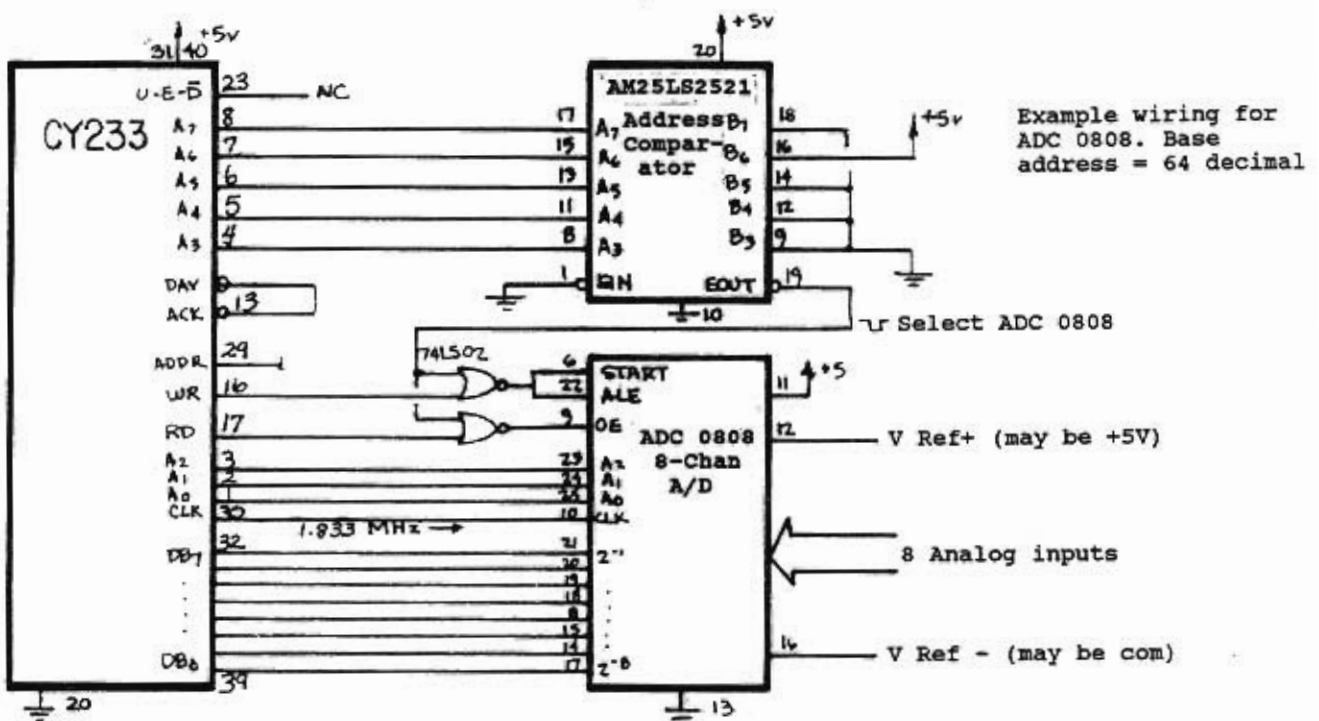


Figure 16.6 Using the CLK output

Encoded addressing is used, with the address comparator establishing a base address. The base address enables RD/ and WR/ strobes to the ADC. The ADC starts a conversion by latching the address (A0-A2) of the analog channel desired. This is done by a write command. Approximately 100uS later the conversion is complete. Since the CY233 can not do a read within 100uS after a write, there is no need to explicitly check the ADC EOC (end of conversion) line. The address of the read which actually fetches the ADC result is unimportant, other than it be within the base address range.

As an example of the driver program for this ADC:

```
X<40h+channel_number> <dummy_data> <cr>
R<40h+channel_number> {<ADC response> <cr>}
```

where the data read by the CY233 and the appended terminator of the CY233 are in curly brackets.

It is possible to operate this converter, and many others, in an even simpler way. This is to use a single read transaction to both start the conversion process and transfer the resultant data. We can do this because the delay between when the address is presented, as so indicated by the ADDR/ strobe, and the actual data bus transfer could be several hundred microseconds. The exact time depends on the particular CY233 mode. It will normally be longer than successive approximation ADCs, but is likely to be shorter than dual slope type converters. The user simply must measure the time to verify it is satisfactory for the intended use. Also, by using the CY233 handshake, and delaying the ACK/ response to the read request until EOC is true, the natural handshake timing delay of the CY233 will keep it from reading data until it is ready.

To execute this shortened transfer, ADDR/ NOR EOUT/ forms a positive true device select for the ADC, the same as RD/ NOR EOUT/ forms the device select in the illustration. This latches the low three bits of the address bus into the converter, and starts the conversion. The subsequent RD/ pulse enables data from the ADC onto the bus.

The driver in this mode looks like:

```
R<40h+channel number> {<ADC response> <cr>}
```

Additional Ideas

Remember that ACK/ can always be used to prevent a valid address, allowing us to implement another form of interrupt driven system, beyond that afforded by changing NET between 0 and F. Let the output of the address comparator be called EOUT/, and be asserted low, as is the INT/ line from the user circuitry. If we form EOUT/ NOR INT/ and connect this to ACK/, ACK/ can be high only when INT/ and EOUT/ are both low. Thus, only when the parallel device has data it wishes to send, will the address be considered valid. This can be used both with externally generated conversational reads, and with the CY233 in a master mode.

Also, for timing interrupts, the 555 timer makes a convenient source. Remember there must be some buffer gate between it and the CY233. This is either the NOR gate in the previous paragraph or the open collector inverter in the 8212 interrupt example, where NET=F (not NET=1) is the alternate state to the master mode. The Periodic delay command can also be used to set a specific sampling rate for the CY233 in a master mode.

In certain special cases, the CY233 can appear to be a 12 or 16 bit device, for writes from the serial line to parallel devices. This is accomplished by utilizing part, or all, of the address bus for data. Obviously the allowable addressing range is reduced. If 4 bits of the address bus are borrowed, we can have a 12 bit data write and a 4 bit address, by redefining what is data and what is address in the message. Similarly, if we define all 16 bits as data, there is only one address in the system.

The principal reason for this effort would be to interface to digital-to-analog converters. However, many of the newer devices are double buffered, which means the entire data word can be assembled in the device before being applied to the actual DAC. This prevents glitches from occurring when old data is mixed within the DAC.

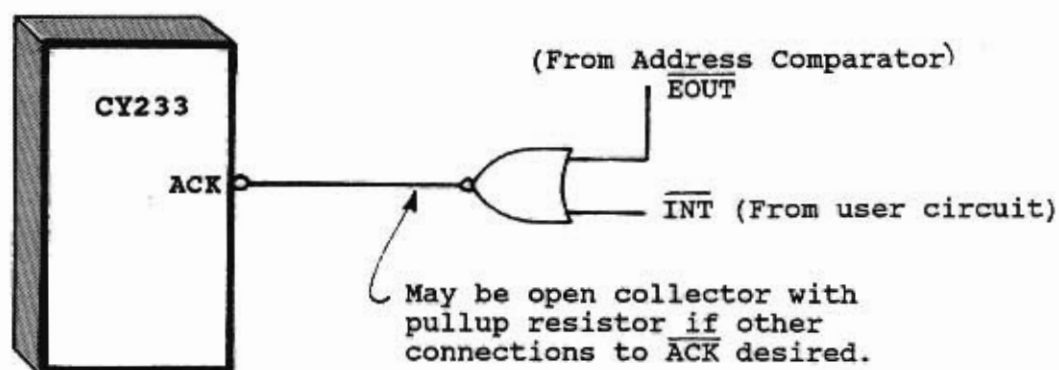


Figure 16.7 Alternative Interrupt Detail

CYxxx Family Members

Cybernetic Micro Systems has a number of different integrated circuits available for industrial control, automation, and testing functions. These devices are highly integrated intelligent controllers that can operate independently, after being set up by commands from a host system. Each device has a specific command set that it understands, and parameters that are used to select the available options.

This family of controllers operates over a common command interface, that consists of a parallel data bus of 8 bits, and some handshake control lines for timing the transfer of commands and data between the host and CYxxx device. Since these devices are highly intelligent and autonomous, they often are located directly with the application, at some distance from the host computer system. In these cases, it is desirable to connect the host system to the controller via a serial port, allowing the controller to be located some distance from the computer, and programmed through simple character transfers between the serial port. Since the controllers all have parallel command interfaces, the CY233 makes a convenient serial front end to these chips. Also, the addressing functions of the CY233 allow many controller chips to share a single serial port, making it possible for one computer system to command and control a number of external functions.

This section presents a brief summary of the CYxxx family of controllers, and some examples of connecting the CY233 to a CYxxx device. Connections for all parts in the family will be similar to those examples shown here. We plan to expand the family into additional applications, and expect that the same kind of command interface will be kept for the newer members.

CY360 Waveform Synthesizer

The CY360 is an Intelligent Waveform Synthesizer, designed to provide stimulus for various testing and control applications. The part generates digitally controlled waveforms that are turned into the desired analog signals through an external analog-to-digital converter.

Functions available to the user include sine wave, square wave, triangle wave, sawtooth wave, pseudo random wave, and dual phase sine wave. These functions are selectable by parameter, with frequency, resolution, amplitude, and repetition programmable by command, not by external timing components. The user simply issues a sequence of commands specifying the desired wave type and characteristics, and then tells the CY360 to generate the desired waveform.

In addition to the predefined functions, the user may specify a custom waveform, either through a linear approximation function, or through a value table. These options allow the CY360 to generate a complex sequence of waveforms, all under command control of a host system.

The CY360 also contains synchronizing signals that allow the waveforms to be controlled by external trigger signals, and a number of other features that makes it useful from the system perspective.

CY5xx Stepper Motor Controllers

A very popular set of controllers is the CY500, CY512, and CY525 Stepper Motor Controllers. These devices also work from a command set, used to specify the nature of the desired motion. Programmable parameters include number of steps, initial and high speed step rates, acceleration time, and stepping direction. Once these parameters have been set, the CY5xx devices will generate the proper pulse train and phase signals to execute the desired motion. This allows the host computer to tend to other tasks while the motors are running, and removes the burden of generating an exactly timed pulse sequence for the motors.

The controllers also include trigger and wait signals, as well as conditional test functions that make them more useful in the overall systems design.

Also, internal program buffers allow the user to define a complex sequence of motions, including time delays and synchronization to external functions. The CY5xx devices can then run the programs on their own, while the host system attends to other tasks.

Figure 13.1 shows the connections required between the CY525 and CY233 devices. The standard handshake lines of the CY233 directly match to the handshake required by the CY525, making this a very natural and easy combination. Most other controllers of the CYxxx family will use a similar design.

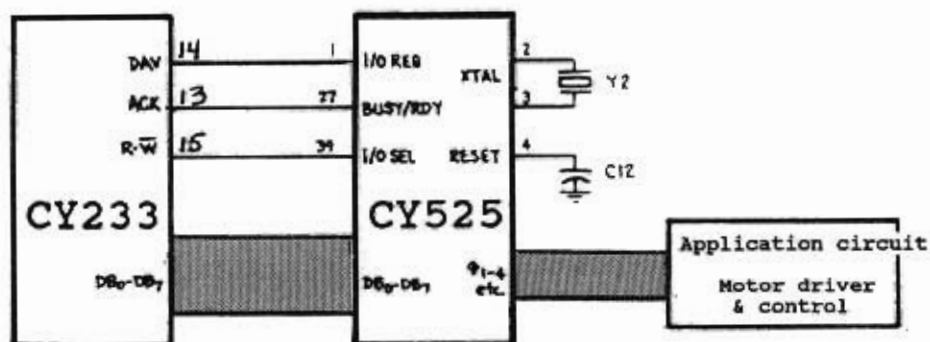


Figure 17.1 CY525 connected to CY233

Since the CY525 has a bi-directional data bus, allowing the host to send commands and read internal parameter and status values, the CY233 R-W/ line is used to control the transfer direction. Both read and write commands may be used with the CY525. Some members of the CYxxx family are write only, while others use the command sequence to determine whether the next transfer will be a read or a write. In these cases, the R-W/ line may not be required.

The CY233 DAV/ signal indicates when the CY233 is ready for a data transfer. This signal directly connects to the I/O Request signal of the CYxxx controller. When the controller detects a low level on I/O Request, it performs the data transfer, such as reading the command character from the CY233. It then generates a low Busy signal from the Busy/Ready line, indicating that the transfer is complete. This signal may be directly connected to the CY233 ACK/ signal, also completing the handshake for the CY233.

The RD/ and WR/ strobes from the CY233 are not needed by the CYxxx devices, and may be left open.

The CYxxx family of parts works on an ASCII command set, where most commands are single ASCII letters, followed by numeric parameters where needed. Commands are ended by a carriage return character. This command sequence is compatible with the ASCII character mode of the CY233, making this the most natural mode of operation for the CY233 when it is connected to a CYxxx device. The message structure of the CY233 is a natural extension of this command protocol.

For example, to set the number of steps for a motion, the CY525 uses a command such as the following:

```
N 1234<cr>
```

which sets the number of steps to 1234 decimal. When using a CY233, the command sequence must only be modified to include the CY233 command letter and valid address. The CY525 command becomes the data portion of the CY233 message, and the carriage return terminator ends both the message and the CY525 command. Remember that the CY233 will write the terminator in the ASCII character mode, if the W command message is used. If our above example is sent to a CY525 at address 02, the required CY233 message, as sent by the serial port of the host, would be:

```
W02N 1234<cr>
```

which has the CY233 message prefix "W02" before the normal CY525 command. The proper CY233 receives the "W02" prefix and finds the address to be valid. It will then write the data portion of the message to the local CY525 as:

```
N 1234<cr>
```

This is exactly the command format required by the CY525 controller. Examples for the other controller chips would be similar.

CY600 Data Acquisition Controller

The CY600 device is a special Data Acquisition Controller. It is designed to interface between analog-to-digital converters, ranging from 8 to 16 bits, and a host computer. The direct connection with the host is an 8 bit parallel data bus, with handshake lines, and is similar to that of other members of the CYxxx controller family.

The CY600 can monitor several external analog signals, scale values, perform correlations, and compute general arithmetic values based on the signals. In addition, it can run several independent tasks simultaneously, each scanning and computing different values. When a programmed condition occurs, the CY600 can interrupt the host system, which may then interrogate for the condition and computed values.

By performing many signal processing tasks, the CY600 can act as a programmable monitor for the computer system, only interrupting when something significant happens. This leaves the computer available for other functions, and relieves it from the periodic data samples required if the computer must perform all signal processing on its own.

Since the CY600 can perform the signal processing functions on its own, once the steps have been specified by the host system, it is possible to locate the CY600 remotely from the computer, near the signal sources, and control the device through a CY233 serial front end. This also enables a network of CY600 devices to monitor many functions, with the host computer acting as a main controller, system coordinator, and exception handler. The following figure illustrates the CY233 to CY600 connections:

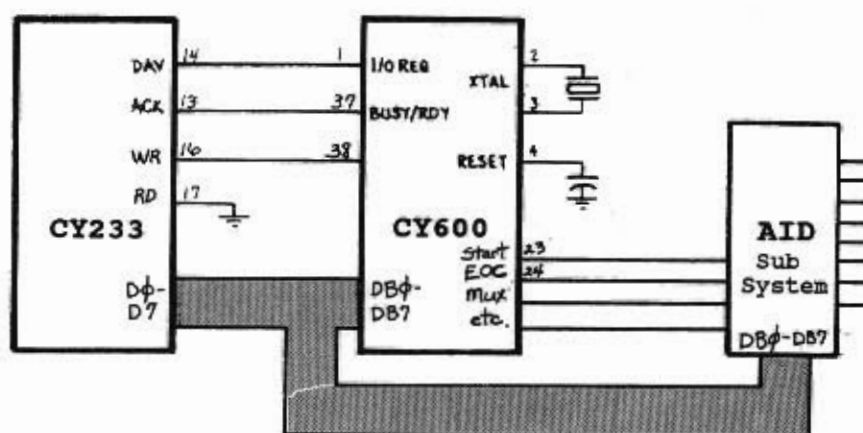


Figure 17.2 CY600 connected to CY233

This interface is similar to that shown for the CY525, with one major exception. The CY600 requires some external components to perform its functions. In a minimum mode, the analog-to-digital converters are connected to the CY600 data bus, and in a maximum mode, an external memory and I/O device is connected to the CY600 data bus, with the converters connected to the I/O ports of the extra device. In both modes, the CY600 has something connected to its data bus, in addition to the host computer or CY233 data bus.

The additional components require a slightly different protocol for the CY600 command handshake. Since the CY600 can be performing asynchronous signal processing functions, the local data bus will probably be busy when the CY233 wishes to write a command to the CY600. This requires the CY233 to wait until the CY600 is ready for data, before actually driving the bus.

The CY600 has a special signal, Bus Control, which is used to enable or disable the host system from driving the bus. Normally, Bus Control is high, and the host should not be driving the CY600 data bus. This allows the CY600 to perform local operations between itself and the external components also connected to the bus. When the host wishes to write to the CY600, this is indicated on the I/O Request line, but the data bus is not driven until the CY600 brings Bus Control low. The CY600 will then read the information from the host, and bring Busy/Ready low to indicate the transfer is complete.

When using a CY233 to control the CY600, the CY233 must be used in the non-strobed transfer mode, enabled by tying the CY233 RD/ signal low. DAV/ is connected to I/O Request, and ACK/ is connected to Busy/Ready. This is identical to the CY525 connections already shown. The CY600 Bus Control signal is connected to the CY233 WR/ pin, which assumes the BUSEN/ function in the non-strobed transfer mode. This additional connection is all that is required to handle the special handshake requirements of the CY600! Now, when the CY233 wishes to write to the CY600, the DAV/ signal will go low, but the data will not be written until BUSEN/ goes low. Data will be stable as long as BUSEN/ stays low, and the handshake transfer will be complete when the ACK/ signal goes low. This perfectly matches the requirements of the CY600, and allows this device to be controlled through a CY233.

The two illustrated connections, between the CY233 and the CY525, and between the CY233 and CY600, represent the two possibilities in connecting the CY233 to any CYxxx family member. All other devices will use connections similar to either the CY525 case, or the CY600 case.

CY750 Programmable Controller

The CY750 is another member of the CYxxx controller family. It performs general purpose digital control functions. The device contains 16 I/O lines, organized as an 8 bit I/O port, and 8 individually controllable and testable I/O lines.

The CY750 is programmed through an 8 bit data bus, with handshake protocol similar to that of the CY525. The CY750 does not have an I/O Select signal to determine data bus direction, so the R-W/line of the CY233 is not used.

Through the CY750 command set, the device has the ability to read or write values, set or test individual control lines, pulse lines, count events, and perform time delays. This allows the user to define a general sequence of digital I/O events, with the CY750 providing the proper sequencing and testing. The ability to perform conditional functions enables the CY750 to alter the sequence of events, based on externally supplied information. The CY750 is useful anywhere a smart I/O port would be required.

Absolute Maximum Ratings

Ambient Temperature under bias.....0°C to 70°C
 Storage Temperature.....-65°C to +150°C
 Voltage on any pin with respect to GND....-0.5V to Vcc+0.5V
 Power Dissipation.....0.2 watts

TABLE III

DC & OPERATING CHARACTERISTICS

(TA = 0°C to 70°C, Vcc = +5V ±10%)

SYM	PARAMETER	MIN	MAX	UNIT	REMARKS
ICC	pwr supply current		18	mA	
VIH	input high level	1.9	Vcc	V	(3.5V for XTAL, RESTART)
VIL	input low level	-.5	0.9	V	
ILO	data bus leakage		10	uA	high impedance state
VOH	output high level	2.4		V	IOH = -60 uA
VOL	output low level		.45	V	IOL = 1.6 mA
FCY	crystal frequency	3.5	12	MHz	see clock circuits

Electrical Conventions

All CY233 signals are based on a positive logic convention, with a high voltage representing a "1" and a low voltage representing a "0". Signals which are active low are indicated by a slash after the pin name, i.e., ACK/.

All input lines except the data bus include weak pull-up resistors. If the pins are left open, the input signals will be high. The data bus pins must have external pull-up resistors to output a high value. Where appropriate, an input line will be considered in the floating state if the CY233 can drive it both high and low.

The data bus is bidirectional, and is tri-state during nonactive modes. Note that data bus signals are positive logic.

Reset Circuitry

The Restart (pin #9) line must be held high upon power-up to properly initialize the CY233. This may be accomplished via the use of a 4.7 uFd capacitor, as shown in Figure 18.1. Restart must be high for 10 msec after power stabilizes on power-up. Once the CY233 is running, Restart need only be high for about 10 usec (11 MHz crystal).

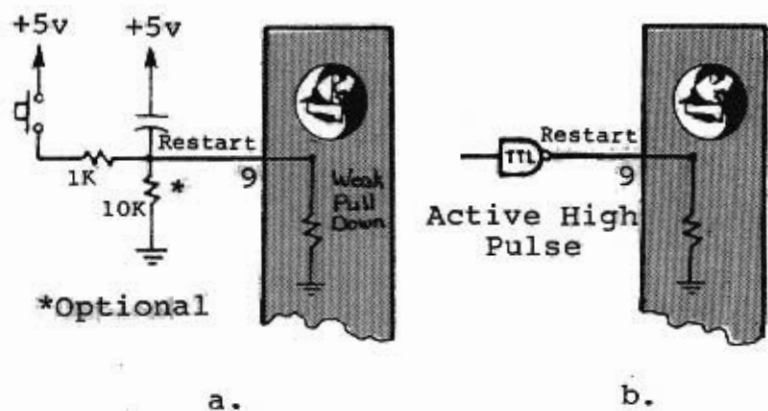


Figure 18.1 a) Restart Circuitry.
b) External Restart.

Clock Circuits

The CY233 may be operated with crystal or external clock circuits. These two options are shown in Figure 18.2. All timing discussed in this manual assumes an 11.059 MHz series resonant crystal. Note that 11.0 MHz, such as a CTS Knights MP110 or equivalent will work fine. The CY233 requires an 11 MHz clock in order to generate the standard baud rates, although any crystal in the allowable range can be used with the adaptive mode, within the timing resolution limits of the CY233. Use 7.3728 MHz to make the fastest possible rate a standard 38400 baud.

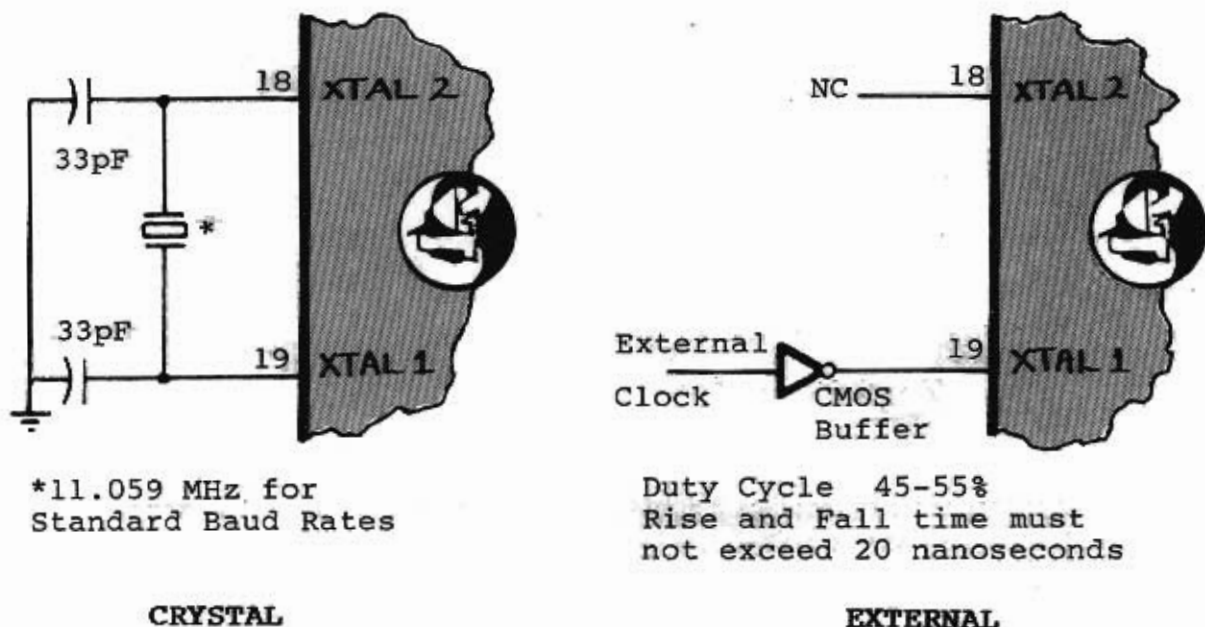


Figure 18.2 Clock Circuits for CY233.

19 Basic Language Driver for the CY233 19

The following BASIC program was written for the IBM-PC family of computers. It turns the PC into a terminal emulator, allowing you to directly communicate with the CY233 through the computer keyboard and display.

```
100 '
110 '   CRT EMULATOR FOR CY233
120 '   Cybernetic Micro Systems, Inc.
130 '
140 '   This program waits for keyboard input, then
150 '   displays the keys on the screen and sends them
160 '   out the COM1 port. It also reads any received
170 '   data from COM1 and displays it on the screen.
180 '
190 CLS
200 LF$=CHR$(10) : NL$=CHR$(0) : ES$=CHR$(27)
210 '
220 '   Open COM1 serial port with baud rate and parity.
230 '   NOTE: settings MUST match CY233 selections.
240 '
250 OPEN "COM1:9600,N,8,1,CS0,DS0,CD0" AS #1
260 LOCATE 5,5,1
270 PRINT "Ready to Go!"
280 '
290 '   Open the screen for displays
300 '
310 OPEN "SCRN:" FOR OUTPUT AS #2
400 '
410 '   Check for keyboard input. Display and send
420 '   any keys to COM port. Stop when Escape key input.
430 '
440 A$=INKEY$ : IF A$=ES$ GOTO 630
450 IF A$<>" " THEN PRINT #1,A$; : PRINT #2,A$;
460 '
470 '   Check for any received data from COM port, and
480 '   display it, then loop back to check keyboard.
490 '
500 WHILE NOT EOF(1)
510   J%=LOC(1) : B$=INPUT$(J%,#1) : LF%=0
520   LF%=INSTR(LF%+1,B$,LF$)
530   IF LF%>0 THEN MID$(B$,LF%,1)=NL$ : GOTO 520
540   PRINT #2,B$;
550 WEND
560 GOTO 440
600 '
610 '   Exit program when Escape key pressed.
620 '
630 CLOSE #1 : CLOSE #2
640 STOP
```


With the above example driver program, your computer behaves like a terminal. That is, any characters typed from the keyboard are sent out the serial port, and any characters received from the serial port are displayed on the screen. In addition, the display will show the keys as they are typed.


To operate the above program, connect the CY233 circuit design through a COM port of your system. COM1 is used by the program, but changing the OPEN statement would allow you to use another COM port. It is very important to match the baud rate and parity settings of the CY233 and the driver program. The example uses 9600 baud, 8 bit data characters, and no parity. Other selections are possible by changing the OPEN statement.

While the example driver program is written in PC BASIC, to run on a PC family computer, a modified version should be able to run on other types of computer systems. Possible changes would include the format and device names for the OPEN statements. "COM1" is the serial port to which the CY233 is connected, and "SCRN:" is the system display screen.

The example driver program does not modify the keyboard input before it is sent to the CY233, so be sure all messages start with their upper case command letters. The CY233 should be operating in the ASCII Character mode, and all data transfers must be preceded by the command letter and hex address characters.

Any response or echo from the CY233 will be displayed along with the keyboard entries you make. If the CY233 is set up to echo messages, they will display as double characters on the screen, one from the keyboard entry, and one from the CY233 echo. Recall that the CY233 will not start to echo until the command letter and address characters have been received. The decision to echo is only made after the address test, when the CY233 knows if the messages is valid or not.

The following checklist will simplify getting your CY233 up and running.

1. Connect pin 20 to ground, pins 31 and 40 to +5 volts.
2. Choose a baud rate to match your serial interface. See the CY233 baud rate table for details.
3. Choose a parity and character length to match your serial interface. See the parity and length table for details.
4. Set the ASCII character mode for now, pin 24 floating.
5. Set the network mode to "echo all--slave" by tying pin 21 high and leaving pin 22 floating.
6. Tie DAV/, pin 14, to ACK/, pin 13, for an auto handshake.
7. Tie UART-Encode-Decode/, pin 23, low for decoded addresses.
8. Be sure Restart, pin 9, is high for at least 10 milliseconds after power stabilizes. The CY233 may be reset at any time.
9. After proper reset, the data bus should be tri-state, and other outputs should be high (>3V).
10. Observe that CLK, pin 30, has a 1.83 MHz clock with 11 MHz crystal. 
11. Connect RxD, pin 10, to your terminal or computer output, using a buffer to translate the interface voltage levels to the TTL levels required by the CY233. DO NOT connect the CY233 directly to an RS-232-C line. Damage to the CY233 will result if voltages outside the 0 to 5 volt range are applied to the CY233.
12. Connect TxD, pin 11, to your terminal or computer input, using a driver to translate the TTL levels of the CY233 to the voltages required by the interface, i.e. +/- 12 volts for RS-232-C.
13. Send the ASCII characters W02 (57h, 30h, 32h) to the CY233.
14. After reception of the ASCII "2", the CY233 will respond by sending all three characters back (echo all mode), and the Address line 2, pin 3, will go low. This indicates that the baud rates, parity, and character modes are properly defined. Note that LEDs on the address and data lines will make it easier to observe the actions of the CY233 on the parallel side.

15. Send the ASCII character A (41h). The CY233 should echo this character back, and the character should strobe on the CY233 data bus.
16. Send any other printable characters. They should also be echoed back, and be written on the data bus. Note that characters may be easier to see if a latched transfer mode is used, i.e., tie pins 16 and 17 low (RD/ and BUSEN/).
17. Send the ASCII carriage return (0Dh) to terminate the write command. This will also be echoed and written to the data bus, but then Address line 2 will go high again, and the data bus will switch to tri-state, unless it is latched.
18. Tie the ACK/ line, pin 13, low. This will make all addresses invalid.
19. Send the characters W03 (57h, 30h, 33h). The CY233 will echo the characters back, after reception of the "3", but no address lines will go low, since this is an invalid address.
20. The CY233 will continue to echo any characters sent, but nothing will happen on the data bus. This sequence should be ended by sending a carriage return again.
21. Connect ACK/ back to DAV/.
22. Send the characters R01 (52h, 30h, 31h). The CY233 will echo back the characters after receiving the "1", then it will send back two additional characters. The first is the present state of the data bus, sent as one character, and the second character is a carriage return, which terminates the read command.
23. Try repeating the above sequences with encoded addresses, by leaving pin 23 floating. Notice the difference in address line patterns in the encoded mode.
24. Try changing the echo modes by changing connections on NET, pin 22, and DUP, pin 21. The CY233 detects these changes dynamically, and will use the new setting with the next message it receives. Notice that valid and invalid messages are echoed differently in each of the modes.
25. Try the other character modes, such as ASCII hex or Binary. Be sure to observe the different requirements for types of characters and terminators. The hex mode allows only the hex digits for data transferred on the data bus, while the binary mode assumes all commands are write operations, and has an 0B3h terminator.

Notes:

CY233 Features

- Ring or Bus Network capability
- Special Local Area Network mode
- Selectable Token support
- Half or Full duplex operation
- Line turnaround capability
- Selectable address range
- Supports 255 encoded addresses
- Supports 8 decoded addresses
- Strobed parallel data transfers
- Handshake parallel data transfers
- Low power CMOS technology
- Single 5 volt, 40 pin device
- TTL compatible I/O levels
- Works with standard drivers
- RS232 serial timing and format
- Baud rates from 300 to 57600
- 7 or 8 bit character data lengths
- Odd, Even, Mark, Space, or No parity
- Master or Slave device operation
- Simple UART data transfer mode
- Powerful expanded command set

Modes of Operation

CY233 LINC (Local Intelligent Network Controller) provides a general mechanism for passing data between a serial network, or computer port, and local parallel devices. It operates in one of three basic modes, which determine the complexity of the connections and the format of the serial data.

UART Mode

is the simplest operating mode in which there is one CY233 connected between a single serial device, such as an RS232 port of a personal computer, and a single parallel device, such as a parallel printer. Data may flow unidirectionally or bidirectionally between the devices, and the serial device deals with only the direct data flow.

Network Mode

is a medium complexity operating mode that allows a single serial port from a host computer to control multiple parallel devices or special parallel hardware. The CY233 uses addressing functions to select between a maximum of 255 unique parallel devices. These devices may all be connected to a single CY233, with that CY233 responding to any address. Alternately, the possible addresses may be split into smaller groups, with a CY233 responsible for each group of addresses. It is possible to have a network of 255 CY233s in this case, with each CY233 responsible for only one parallel device and one address assignment. Any combination between these extremes is also possible. A special prototyping board, the CYB-233, is available from Cybernetics for support of this operating mode.

Host Ring and Party Line Networks

are two of the various network connection schemes possible in this mode. These topologies will usually have a host computer controlling several CY233 nodes, with parallel device hardware connected to each node. Data flow is bidirectional, with the host writing messages to and reading messages from the CY233 nodes.

Wire Saver

is another possible connection, in which parallel data is generated at one location, transmitted serially to another location, and reconstructed in parallel at the destination. In this case, the CY233s act to convert the data between the parallel and serial sides. This scheme can be very

economical if there is any distance between the locations, since it saves the need to run expensive parallel cables over that distance. Also, data flow can be bidirectional in the Wire Saver design.

CY233 Message Structure

is used to transfer data in the Network mode. This structure (see opposite page) is used to specify the address of the parallel device involved in a specific set of data transfers. The structure is very simple, and involves minimum overhead for the number of devices that can be controlled through one serial channel.

LAN (Local Area Network)

is the most complex mode of operation, in which multiple serial devices can be connected in a network. In this scheme, two CY233s are required at each network node, one to connect that node to the others of the network, and a second to connect the local serial device to the node. Optional token support functions may be used to control the flow of messages in such networks. The CYB-233-LAN prototyping kit, available from Cybernetics, implements one LAN node.

Host Ring LAN

is the first of two possible network designs, in which a network master computer is connected at the network level. Communications between the various nodes of the network are controlled through this master/server system. The local serial devices could be other computers, terminals, or custom serial hardware.

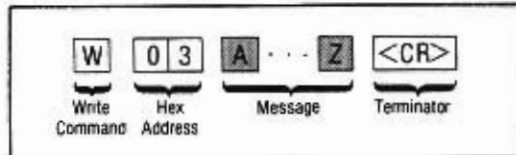
Peer Ring LAN

is the second LAN design, in which the network consists of only CY233 nodes, connected to local systems. In this design, communications can occur between any two nodes of the network, without involving a master/server computer. This is a very general communications scheme, and allows a local area network to be designed without special hardware requirements. The standard serial communications ports of the local systems are connected to the CY233 nodes, which form the only special hardware in the network, and represent a very economical network design.

Basic Message Structure

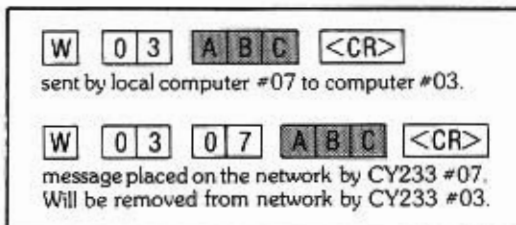
ASCII Message Format

When multiple CY233s are connected in a network, messages cause data to flow between nodes of the network. A message is a simple structure that consists of four elements: a command letter, an address, the data contents, and a message terminator. A typical message is shown below:



This message will write 'A..Z<cr>' to the device attached to the network as address #03. The above message format is typical of Host Ring and Party Line networks in which a host computer sends messages to devices.

When most devices on the net are computers, it is desirable to identify the source as well as the destination of the message. This is done automatically in LAN networks, both Peer Ring and Host Ring LANs. A local computer creates a write message with data and a target address to which the data is to be sent. The local CY233 appends its address to the target address, as it places the message on the network.



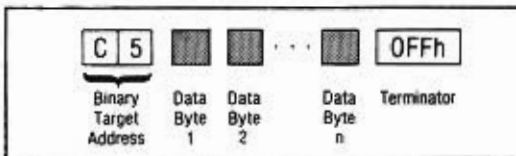
The above structure allows any computer on the network to communicate with any other unambiguously. Network query and status commands allow each local computer to poll the network for automatic configuration.

Token Support

CY233s recognize local intelligent devices and perform special functions in support of token passing modes in which local devices wait for the 'token' before placing their message on the network. Watchdog timeout functions prevent the token from being 'stuck' at any node that does not respond. Each local CY233 takes responsibility for keeping the token passing, if its local device fails to respond in a specified period.

Binary Message Format

For simplest modes, such as the Wire Saver, a simple binary mode message uses only the destination address, data contents, and the special terminator, OFFh. All messages in this mode are implied writes. For example:

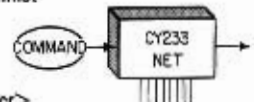


CY233 Commands

All CY233 commands are listed below. While many are self explanatory, the interpretation of some of them is dependent upon the network topology and operating mode. A complete description is beyond the scope of this data sheet, however a detailed user manual (~150 pages) is available from Cybernetic Micro Systems.

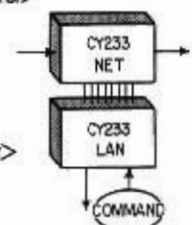
Network Serial Commands

- @ Test Network Link, @aa<cr>
- A..F Undefined, Hex Character conflict
- G Reserved command letter
- H Undefined
- I Initialize CY233, Iaa<cr>
- J Enable Echo All mode, Jaa<cr>
- K Enable parallel Error status, Kaa<cr>
- L Undefined
- M Fill consecutive locations, Maahh...hh<cr>
- N Display consecutive locations, Naacchh...hh<cr>
- O Reserved command letter
- P Set Periodic Master mode delay, Paatttt<cr>
- Q Query for serial error status, Qaass<cr>
- R Read from parallel device, Raad...d<cr>
- S Sense local address, Saabb...11<cr>
- T Set handshake Timeout delay, Taatttt<cr>
- U Token message, Uaa<cr>
- V Undefined
- W Write to parallel device, Waad...d<cr>
- X Transfer to parallel device, Xaad...d<cr>
- Y.. User Reserved, ASCII 59h..5Fh



LAN Serial Commands

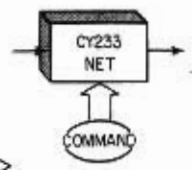
- @ Return LAN Node Address, @aa<cr>
- G Reserved command letter
- I Initialize CY233, Iaa<cr>
- O Reserved command letter
- P Pass Parallel Command to Network CY233, PCmd<cr>
- Q Query for serial error status, Qaass<cr>
- R Generate Read Message to Network, Raad...d<cr>
- T Set LAN handshake Timeout delay, Taatttt<cr>
- W Generate Write Message to Network, Waad...d<cr>



Commands not listed above should not be used in a Local Area Network.

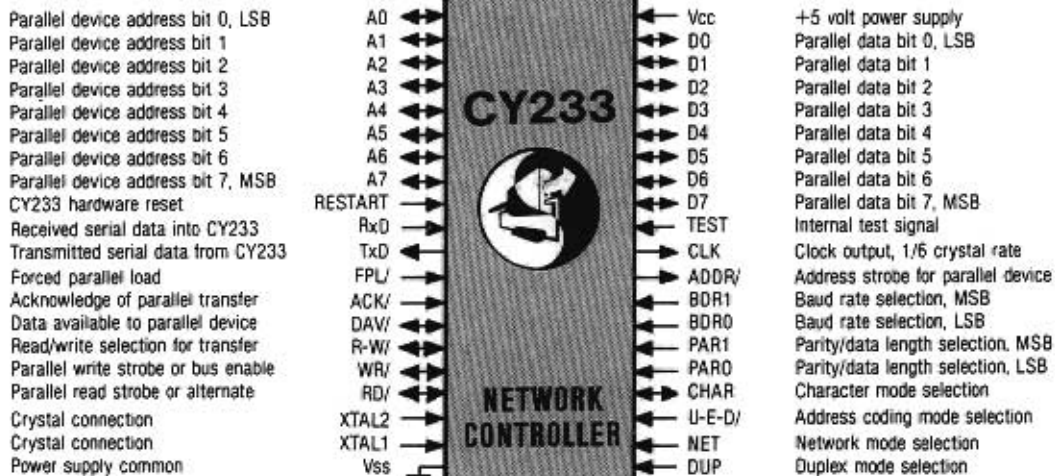
Parallel Commands

- G Reserved command letter
- I Initialize CY233, I<cr>
- J Enable Echo All mode, J<cr>
- K Enable parallel Error status, K<cr>
- L Load value of Status byte 2, Lvv<cr>
- O Reserved command letter
- P Set Periodic Master mode delay, Ptttt<cr>
- S Sense nodes on Ring, S<cr>
- T Set handshake Timeout delay, Ttttt<cr>
- U Generate Token message, Uaa<cr>



Only the commands listed above are available as Parallel commands. Any others will set the parallel status error bit.

Pin Configuration and Description



Some pin descriptions below show three possible states. TTL 0, TTL 1, or floating (F).
Also, many pins on the CY233 are bidirectional, assuming various functions in the different operating modes of the device.

Pin	Mnemonic	Function			
1 (I/O)	A0	Parallel device address bit 0, LSB			
2 (I/O)	A1	Parallel device address bit 1			
3 (I/O)	A2	Parallel device address bit 2			
4 (I/O)	A3	Parallel device address bit 3			
5 (I/O)	A4	Parallel device address bit 4			
6 (I/O)	A5	Parallel device address bit 5			
7 (I/O)	A6	Parallel device address bit 6			
8 (I/O)	A7	Parallel device address bit 7, MSB			
These pins specify the current address to the parallel device. They are used for address generation and testing.					
9 (I)	Restart	CY233 hardware reset			
The CY233 is hardware reset by this active high signal.					
10 (I)	RxD	Received serial data into CY233			
TTL level serial data is input to the CY233 through this pin.					
11 (O)	TxD	Transmitted serial data from CY233			
TTL level serial data is output on this pin.					
12 (I)	FPL/	Forced parallel load			
The local parallel device can force a multi-byte read or parallel command operation using this pin. The read address must also be supplied.					
13 (I)	ACK/	Acknowledge of parallel transfer			
14 (I/O)	DAV/	Data available to parallel device			
15 (I/O)	R-W/	Read/Write selection for transfer			
These pins are used for handshake control of parallel data transfers, indicating when data are stable and have been accepted by the CY233 and local parallel device.					
16 (I/O)	WR/	Parallel write strobe or Bus Enable			
17 (I/O)	RD/	Parallel read strobe or alternate select			
As an alternative to the handshake data transfers, the CY233 also supports strobed transfers. These pins control the timing of the transfers in the strobed mode. Alternate functions are available in the handshake mode.					
18 (I)	XTAL2	Crystal connection			
19 (I)	XTAL1	Crystal connection			
An external 11 MHz crystal or clock source is connected to these pins. All baud rates and device timing are derived from this clock.					
20 (I)	VSS	Power supply common			
21 (I)	DUP	Duplex mode selection			
22 (I)	NET	Network mode selection			
Select the communications mode through these pins.					
	NET	DUP	ECHO	MODE	USEFUL FOR
	0	0	Invalid	Master	Ring Network
	0	F	None	Master	Point to Point
	0	1	All	Master	Ring Network
	F	0	Invalid	Slave	Ring Network
	F	F	None	Slave	Half Dup, Line Turnaround
	F	1	All	Slave	Console to CY233
	1	0	Valid	Master	Console to CY233
	1	F	Valid	Slave	Bus
	1	1	Valid	Slave	Bus
23 (I)	U-E-D/	Address coding mode selection			
Select the CY233 addressing format through this pin.					
	1	Uncoded UART mode, addresses not used			
	F	Encoded Address, positive true binary			
	0	Decoded Address, negative true, 1 of 8			

Pin	Mnemonic	Function		
24 (I/O)	CHAR	Character mode selection		
This pin selects the data representation for the contents of serial messages. When the CY233 is running, it may also be used to enable tri-state transmit drivers.				
	F	ASCII Character mode		
	1	ASCII Hexadecimal character mode		
	0	Binary Character mode		
25 (I)	PAR0	Parity/Data length selection, LSB		
26 (I)	PAR1	Parity/Data length selection, MSB		
Select parity and character length options with these pins.				
	PAR1, 0	Parity	Data Length	Total Character Length
	F F	Mark	7	10
	F 1	Even	7	10
	F 0	Odd	7	10
	1 F	Space	7	10
	1 1	None	8	10
	1 0	Space	8	11
	0 F	Mark	8	11
	0 1	Even	8	11
	0 0	Odd	8	11
27 (I)	BDR0	Baud rate selection, LSB		
28 (I)	BDR1	Baud rate selection, MSB		
These pins select the CY233 baud rate as follows:				
	BDR1, 0	Rate with 11.059 Mhz clock		
	F F	Self Adaptive		
	F 1	57600		
	F 0	19200		
	1 F	9600		
	1 1	4800		
	1 0	2400		
	0 F	1200		
	0 1	600		
	0 0	300		
29 (O)	ADDR/	Address strobe for parallel device		
A short strobe is generated on this line when the CY233 has written a new address to lines A0-A7, but before the lines are tested. Could be used to trigger external addressing logic.				
30 (O)	CLK	Clock output, 1/6 crystal rate		
31 (I)	TEST	Internal test signal, connect to Vcc		
32 (I/O)	D7	Parallel data bit 7, MSB		
33 (I/O)	D6	Parallel data bit 6		
34 (I/O)	D5	Parallel data bit 5		
35 (I/O)	D4	Parallel data bit 4		
36 (I/O)	D3	Parallel data bit 3		
37 (I/O)	D2	Parallel data bit 2		
38 (I/O)	D1	Parallel data bit 1		
39 (I/O)	D0	Parallel data bit 0, LSB		
These bi-directional lines are used for all parallel data transfers between the CY233 and local parallel devices.				
40 (I)	VCC	+5 Volt power supply		

CY233 Electrical Specifications

SYMBOL	PARAMETER	MIN	MAX	UNITS	REMARKS
ABSOLUTE MAXIMUM RATINGS:					
	VCC low supply current		18	mA	
	V _{IH} input high level	1.9	Vcc	V	(3.5V for XTAL RESTART)
	V _{IL} input low level	-5	0.8	V	
	I _{LD} data bus leakage		10	µA	high impedance state
	V _{OH} output high level	2.4	V		I _{OH} = -80 µA
	V _{OL} output low level	0.45	V		I _{OL} = 1.8 mA
	Power Dissipation		0.2	watts	
	FCY crystal frequency	3.5	12	MHz	see clock circuits
DC & OPERATING CHARACTERISTICS:					
		TA = 0°C to 70°C, Vcc = +5V ± 10%			