Application Note 1063: May 01, 2002

Microcontroller Recognizes Addresses in RS-485 Systems

Note describes a simple circuit to create an RS-485 slave data transceiver capable of recognizing its assigned address. Only three ICs are required, a microcontroller, a physically tiny UART and an RS-485 transceiver. Enabling software is provided.

One of many benefits of using the RS-485 data-interface standard, rather than RS-232, is its capability of implementing multi-drop networks. Such networks usually carry 9-bit data words, in which the ninth (parity) bit identifies each word as address or data.

One decision posed by small microcontrollers like IC1, which does not include a hardware universal asynchronous receiver-transmitter (UART), is whether to add an external-component UART or write your own UART in software. External UARTs once represented a large increase in board area, complexity, and price, and the ones available were usually an overkill for small µC applications. On the other hand, it may be difficult to spare the program memory and processor resources needed for a software UART. The program memory in IC1, for example, is only 1K x 14 bits of EEPROM. Available today is a third alternative–a low-cost external UART (IC2) that is also physically small. Use of this device liberates the program memory otherwise needed for a software UART.

An RS-485 bus can carry up to 256 transceiver modules of the type shown in Figure 1. IC3 is the RS-485 transceiver, and IC4 is a "µC supervisor" that holds the µC in reset until a valid supply voltage is present. The µC's assembly-language program can be downloaded from Maxim's website.

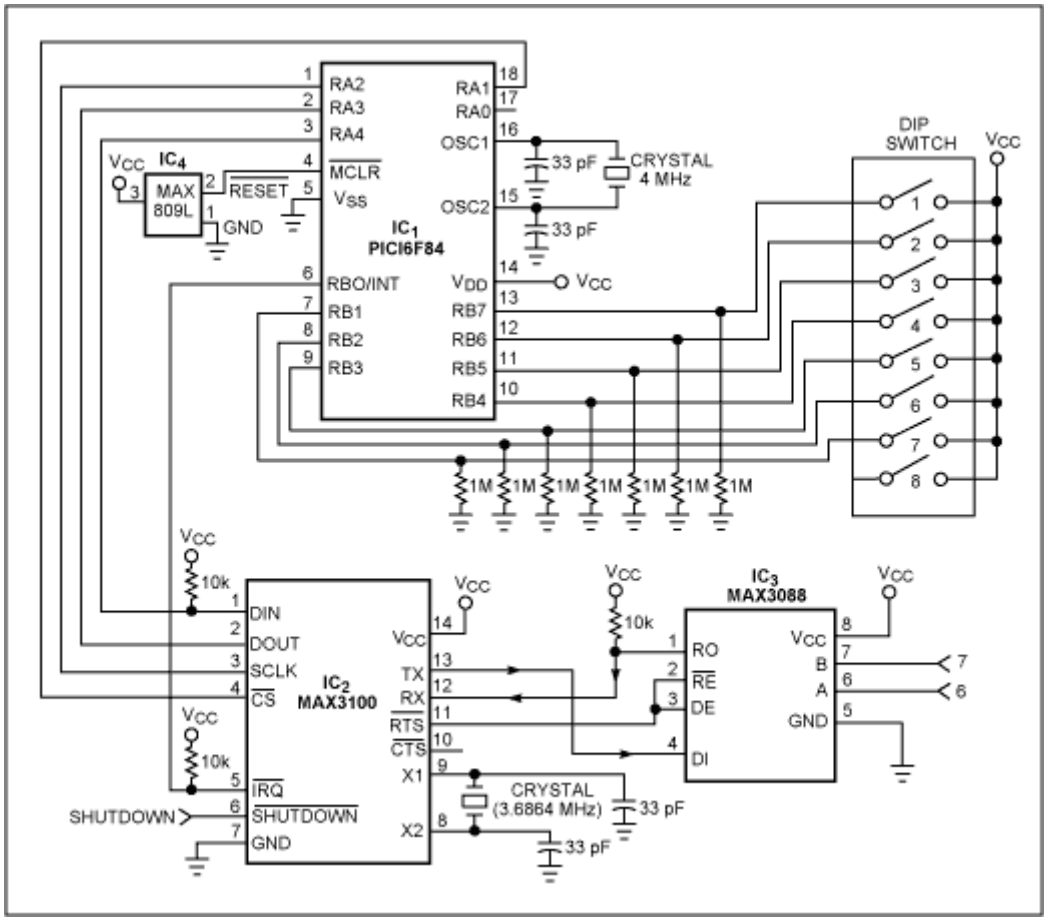

Figure 1. RS-485 networks can accommodate as many as 256 slave transceivers. Adding a small UART (IC2) and µC (IC1) to the RS-485 transceiver (IC3) forms a slave data-transceiver module that responds to its own network address.

The application shown is a slave-test configuration, but you can modify the code to accommodate any specific RS-485 address-recognition application. The circuit works as follows:

When an address is transmitted over the bus, IC2 in each slave module initiates a parity interrupt. IC1 in each module then reads all the data in its internal FIFO, locates the address word, and compares that address with its own address stored in the eight DIP switches. A match causes the slave to clear the interrupt and transmit (to the master) an ASCII "A" (HEX41) followed by its own address. If the slave module reads the FIFO contents without finding a match, it clears the current address-word interrupt and waits for the next one.

A similar version of this article appeared in the November 11, 2001 issue of *EDN* magazine.

MORE INFORMATION

MAX3088: QuickView -- Full (PDF) Data Sheet (288k) -- Free Sample

MAX3100: QuickView -- Full (PDF) Data Sheet (408k) -- Free Sample