

## APPLICATION NOTE 4037

# Regain Location Information by Leveraging the 1-Wire® Chain Function—A Simple Signaling and Protocol Method Determines Device Physical Location

By: Bernhard Linke, Principal Member Technical Staff

Jun 07, 2007

*Abstract: As technology has evolved towards lower-cost serial bus systems using protocol-based addressing, the knowledge of the physical location of a component has been lost. This can be a problem if the serial bus is used for instrumentation or control purposes, e.g., to measure temperature at multiple locations. This app note details how to regain location information with lower-cost serial bus systems, such as 1-Wire, through the new built-in chain function of the DS28EA00.*

This article was also featured in [Maxim's Engineering Journal, vol. 60](#) (PDF, 1.0MB).

A common characteristic of digital bus systems is sharing. Parallel bus systems, having become commonplace with the introduction of microprocessors, share data and address lines with all components connected to the bus. Chip-select signals are decoded from address lines and control signals. Through hardware design and wiring, the physical location of each component connected to the bus was always known. As technology evolved towards lower cost serial bus systems,<sup>1</sup> the address lines were sacrificed first. The chip-select function remained with SPI™ and MICROWIRE® serial buses. More advanced serial bus systems introduced protocol-based addressing, in which address information is transmitted as a preamble to the data. A prominent example of such buses is the I<sup>2</sup>C/SMBus, where the communication interface is reduced to a data and clock line. To achieve additional cost savings, the clocking information can be embedded in the data stream. Representatives of this category are 1-Wire, LIN, and SensorPath® buses.

As a consequence of protocol-based addressing, the knowledge of the physical location of a component is lost. This is a problem if the serial bus is used for instrumentation or control purposes. A typical example is temperature monitoring, as explained in application note 4334, "[Add Thermal Monitoring to Reduce Data Center Energy Consumption](#)." Address pins that, in the case of the I<sup>2</sup>C bus, are tied to logic 1, logic 0, SCL, or SDA can mitigate the problem. This way, a single pin can define four different digital states, equivalent to two bits. Although simple and efficient, this approach has its limit as the original I<sup>2</sup>C protocol provides only seven address bits. The 1-Wire bus, with its 64-bit address, allows more flexibility in this respect. However, each address pin increases cost and, therefore, is not a welcome addition.

A simple, low-cost mechanism to regain the physical location information for a serial bus with multiple attached devices marks one location as the beginning of the network (location #1) and then identifies subsequent devices in the connection, as in **Figure 1**. The connections that constitute this mechanism are shown as thick, blue lines. At the first location, the line is simply tied to ground.

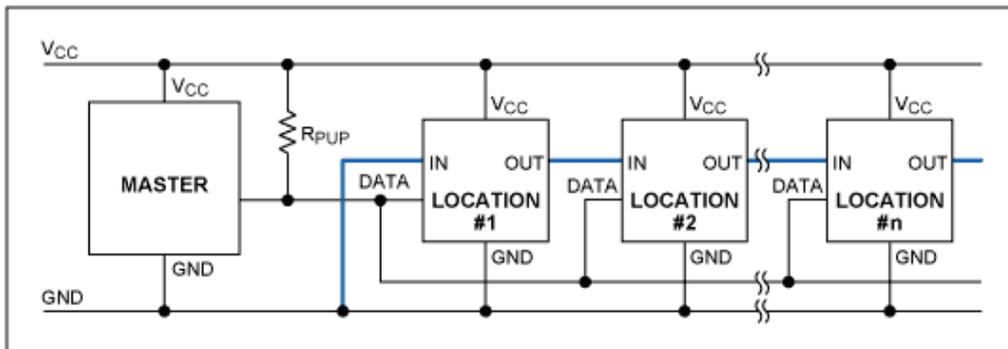


Figure 1. A serial network is enhanced for location detection.

Aside from the two pins labeled IN and OUT in Figure 1, on-chip logic is required to read the IN pin and to write to the OUT pin. With the help of these resources, the master can identify the first device, then the next device, and so on until all devices and their locations relative to each other are identified, provided that the protocol supports a network inventory function. A serial bus system that meets this requirement is the 1-Wire bus.

The 1-Wire bus is a simple signaling scheme that performs half-duplex bidirectional communications between a host/master controller and one or more slaves sharing a common data line (Figure 2). Both power and data communication for slave devices are transmitted over this single 1-Wire line. For power delivery, slaves capture charge on an internal capacitor when the line is in a high state, and then use this charge for device operation when the line is low during data transmission. A typical 1-Wire master consists of an open-drain I/O port pin with a resistor pullup to a 3V to 5V supply. This clever communication scheme also allows you to add memory, authentication, and mixed-signal functions at any time, easily and efficiently.

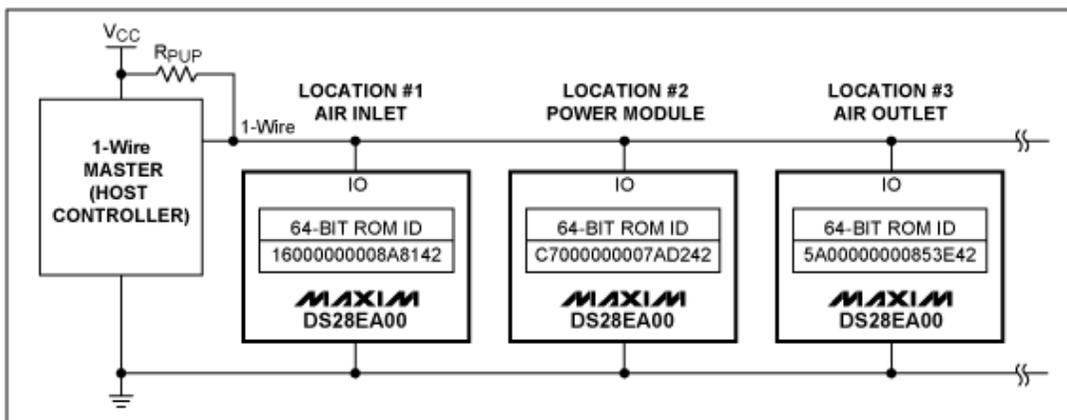


Figure 2. A typical 1-Wire network without the chain function is shown.

An important, fundamental feature in every 1-Wire system is that each slave device has a unique, unalterable read-only memory (ROM), 64-bit, factory-lasered serial number (ID) that will never be repeated in another device. Besides providing a unique electronic ID to the end product, this 64-bit ROM ID allows the master device to select a slave device among the many that can be connected to the same bus wire. Also part of the 64-bit ROM ID is an 8-bit family code that identifies the device type and functionality supported.

Typically, when a system initially starts up, the 1-Wire master might not know the number of devices on the 1-Wire bus or their ROM IDs. By taking advantage of the wired-AND property of the bus, the master can use a process of elimination to identify all devices. Through the *Search ROM\** function,<sup>2</sup> the master can determine the ROM IDs of devices on the bus. However, Search ROM results do not provide information on an individual

device's location. For example, for the ROM IDs shown in Figure 2, the devices are discovered in the sequence shown in **Table 1**.

**Table 1. Device Discovery Sequence**

Sequence	ROM ID (Hexadecimal)	Position in Network
1	C700000007AD242	Location #2
2	5A00000000853E42	Location #3
3	16000000008A8142	Location #1

## The Chain-Function Concept

Using the chain function to determine the device's physical location in a 1-Wire networked environment is based on:

- System-level knowledge of the physical routing of the 1-Wire network, i.e., where the first, second, etc., device/node is physically located.
- Learning the sequence, first to last, in which devices (ROM IDs) are physically connected.

Item a) is "by-design" system knowledge that comes from the physical layout of the 1-Wire network wiring. Item b) is typically the unknown set of information, which is easily discovered using the chain-function communication. This chain function (**Figure 3**) uses two pins, an input (active-low EN) to enable a device to respond during the discovery, and an output (active-low DONE) to inform the next device in the chain that the discovery of the previous device is done. The active-low DONE signal connects to the active-low EN input of the next device. The active-low EN input of the first device in the chain is hardwired to GND. The connections to operate the chain function are shown as thick, blue lines. Aside from the active-low EN and active-low DONE pins, sequence discovery requires a new network function command, *Conditional Read ROM*. This command reports a device's ROM ID only if certain conditions (explained below) are met. *Read ROM*, in contrast, causes all devices in a network to transmit their ROM IDs at the same time. The DS28EA00 is the first device to implement and support this new chain function (as shown in Figure 3).

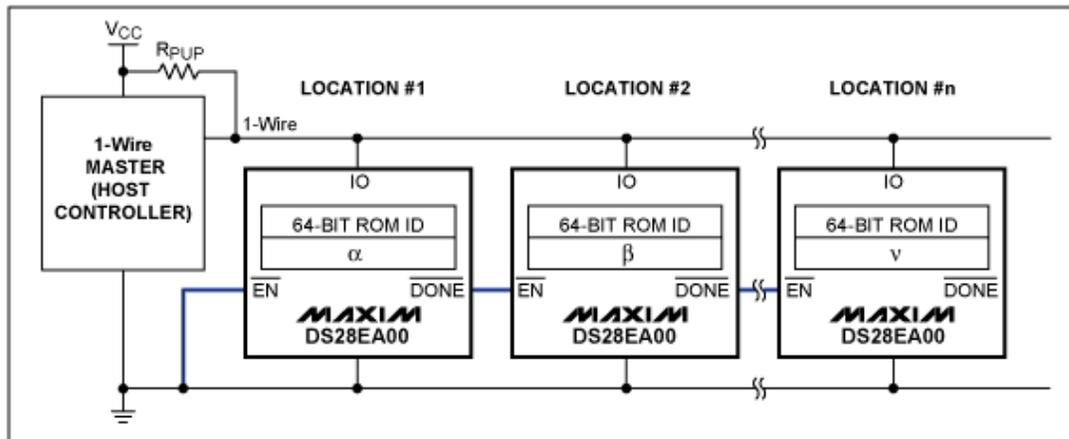


Figure 3. The DS28EA00 is shown in a typical 1-Wire network employing the chain function.

Besides their use in sequence discovery, the active-low EN and active-low DONE pins can serve as digital PIOs. To achieve this coexistence, the chain function defines three chain states, *OFF*, *ON*, and *DONE*. The transition from one chain state to another is controlled through the new *Chain* command. **Table 2** summarizes the chain states and their associated behaviors.

**Table 2. Chain States**

	Device Behavior
--	-----------------

Chain State	Active-Low EN (PIOB)	Active-Low DONE (PIOA)	Conditional Read ROM
OFF (default)	PIO (high impedance)	PIO (high impedance)	Ignored
ON	Active-low EN input	Pullup on	Recognized if active-low EN is logic 0
DONE	No function	Pulldown on (active-low DONE is logic 0)	Ignored

The power-on default chain state is OFF, where active-low DONE (PIOA) and active-low EN (PIOB) are solely controlled through separate *PIO Access Read* and *Write* commands, as described in the [DS28EA00 data sheet](#). In the Chain ON state, active-low DONE is pulled high to the device's internal  $V_{DD}$  supply through an on-chip  $\sim 40k\Omega$  resistor ( $R_{CO}$ ), thus applying a logic 1 to the active-low EN pin of the next device. The Conditional Read ROM command is recognized only in the Chain ON state, provided that active-low EN is at logic 0. This condition is met by a maximum of one device in the network during the sequence-discovery procedure.

To transition a device from the Chain OFF to the Chain ON or Chain DONE state, the master uses the Chain command. **Figure 4** shows the possible transitions. After the Chain command code, the master must send a suitable *Chain Control* byte. To minimize the possibility of receiving an erroneous Chain command, this control byte is first transmitted in its true form, and then in its inverted form. As a feedback for a successful state change, the master receives AAh confirmation bytes.

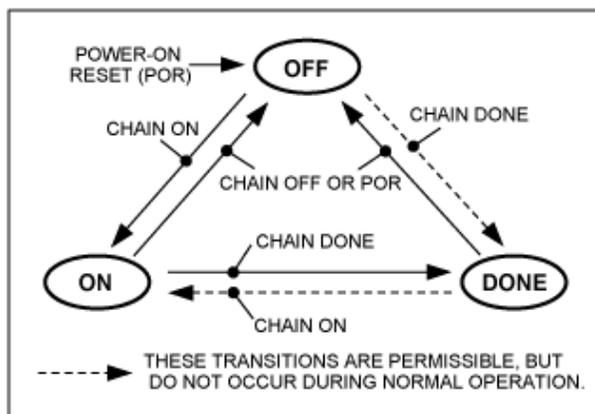


Figure 4. This diagram illustrates transitions between chain states.

To begin a sequence discovery, the master must put all devices into the Chain ON state. After a device's ROM ID is read through Conditional Read ROM, the master puts the device into the Chain DONE state, which enables the next device in the Chain to respond to the Conditional Read ROM command. As the sequence discovery progresses, devices are sequentially transitioned into the Chain DONE state until all devices are identified. Finally, all devices are put into the Chain OFF state, which releases the active-low EN and active-low DONE pins to become PIOs and restores their power-on default state.

## Sequence Discovery Example

**Precondition** A master controls a network as shown in Figure 3. All devices support the chain function. To discover the sequence of devices in the chain, i.e., the number of locations and for each location the device's ROM ID (also known as the Registration Number), the master performs the following procedure.

**Initialization** The master issues a *Skip ROM* command followed by a Chain ON command, which puts all

devices in the Chain ON state. The pullup of the active-low DONE pin through  $R_{CO}$  changes the active-low DONE/active-low EN connections to logic 1 at all devices except for the first device in the chain.

**First Cycle** The master sends a Conditional Read ROM command, which causes the first device in the chain to respond with its 64-bit ROM ID. The master memorizes the ROM ID and the fact that this is the first device in the chain. Next, the master transmits a Chain DONE command. Passed through the active-low DONE pin of device #1, this command asserts logic 0 at the active-low EN pin of the second device in the chain and also prevents device #1 from responding again.

**Second Cycle** The master sends a Conditional Read ROM command. Because device #2 is now the only device in the chain with a logic 0 at active-low EN, it responds with its ROM ID, which is stored by the master with the sequence number '2'. (Device #1 cannot respond to the command, as it is in Chain DONE state.) Next, the master transmits a Chain DONE command.

**Additional Cycles** To identify the ROM IDs of the remaining devices and their physical sequence, the master repeats the steps of Conditional Read ROM and Chain DONE. If there is no response to the Conditional Read ROM command, all devices in the chain have been identified.

**Ending** At the end of the discovery process, all devices in the chain are in the Chain DONE state. The master ends the sequence discovery by issuing a Skip ROM command followed by a Chain OFF command. This puts all the devices into the Chain OFF state, and transfers control of the PIO pins to the *PIO Access Function* commands (e.g., to blink a LED). For the full description of the Conditional Read ROM and the Chain command and flow charts, refer to the [DS28EA00 data sheet](#).

Assuming a standard 1-Wire speed (960 $\mu$ s for a reset/presence detect cycle and 65 $\mu$ s per time slot), the initialization and ending combined take ~7ms (one-time overhead). The discovery and location detection takes ~7.7ms per device. Under the same conditions, Search ROM takes ~14ms per device. Within 100ms, for example, one could identify and locate 12 devices with the chain function, but only identify 7 devices if relying solely on the Search ROM function.

## Additional Aspects

**Cable Capacitance** Category 5 phone cables, which are commonly used to build 1-Wire networks, have a capacitance of approximately 50pF/m between the wires of a twisted pair. Depending on the size of the network, this can add a significant load on the line when putting all devices into the Chain ON state. Active pullup of the 1-Wire line may be necessary when in parasite-power mode to prevent the voltage from dropping below the permissible minimum value. This precaution is not needed with central or local  $V_{CC}$  power supply.

**Conditional Read ROM** The [DS2401](#) also understands this command code as Read ROM for compatibility with the DS2400 Silicon Serial Number, which was discontinued in 1993. For this reason, do not connect any DS2401 to a network that uses the chain function. Therefore, when using a 1-Wire port adapter, select one that does not include the DS2401. The DS2405 rev. A also responds to the Conditional Read ROM command code like a DS2401. The DS2405 rev. B, in production since 1998 and discontinued in April 2010, ignores the Conditional Read ROM command code.

**1-Wire Master Circuits** For the embedded environment, a variety of low-cost, discrete and IC-based 1-Wire masters can be implemented. Discrete solutions range from a resistor pullup to a spare  $\mu$ C port pin or advanced drivers.<sup>3, 4</sup> Integrated drivers optimized to drive 1-Wire lines include the [DS2480B](#) (serial port, UART) and the [DS2482](#) (I<sup>2</sup>C port, **Figure 5**). The 8-channel version of the DS2482 has three address pins, which allow a single host controller to operate up to 64 separate 1-Wire networks. Application note 192, "[Using the DS2480B Serial 1-Wire Line Driver](#)," explains the DS2480B from the software developer's point of

view. A similar document is also available for the DS2482 driver.<sup>5</sup>

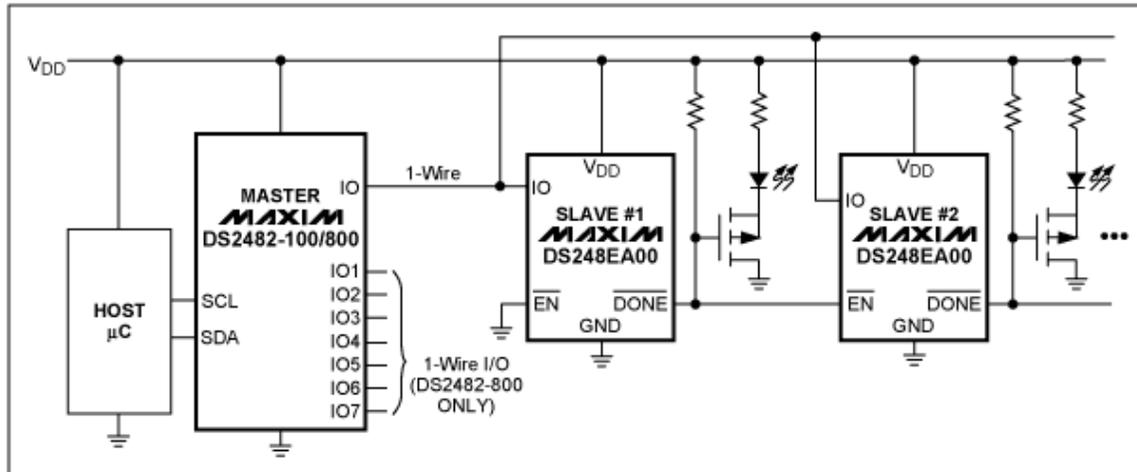


Figure 5. The DS2482 single-channel, I<sup>2</sup>C-to-1-Wire bridge device is used as a 1-Wire master. The active-low DONE output is also used to drive an LED, which does not affect the chain function.

**Power Supply** If  $V_{CC}$  power is available, all DS28EA00s can perform a temperature conversion simultaneously. A subsequent *Conditional Search* command can be issued to identify only those devices that report an alarming temperature. The device's ROM ID, in conjunction with its location determined from the sequence discovery, quickly tells where corrective action is required. Without  $V_{CC}$  power, temperature conversions must occur sequentially. In addition, one must ensure that the voltage on the 1-Wire data line does not drop too far at the beginning of the sequence discovery (when all devices transition from Chain OFF to Chain ON).

**Speed** The fast timing of overdrive speed is not suited to operate a 1-Wire network that contains multiple devices or extends over ~3m; standard speed should be used instead. Depending on the number of devices in the network, particularly if using a parasitic power supply, extended recovery time is required, even if using standard speed.<sup>6</sup>

**Troubleshooting** If the sequence detection does not work as expected, look for a voltage drop on the 1-Wire line after issuing the Chain ON command. If the voltage dips below 3.0V, there is a risk that the command was not executed properly. To prevent such a drop, either use a 1-Wire driver circuit with active pullup after Chain ON, or use a central power supply. The sequence detection will also fail if the first device in the chain has its active-low EN input open, active-low EN tied to the 1-Wire line, or active-low EN tied to  $V_{CC}$ . Make sure that the network does not contain a DS2401. Connecting two or more networks in parallel to the same 1-Wire port is not permissible as all "first devices" will respond simultaneously, resulting in a ROM ID with an invalid CRC byte.

## Conclusion

The chain function is a new feature that allows the master to determine the physical sequence of devices in a linear network under software control, without human intervention. The DS28EA00 1-Wire digital thermometer is the first device to integrate this chain function. Compared to concepts that derive device location data from address pins, the DS28EA00 is the most cost-effective choice for multipoint temperature-measurement applications.

## References

1. Application note 3967, "[Selecting a Serial Bus](#)"
2. Application note 187, "[1-Wire Search Algorithm](#)"
3. Application note 4206, "[Choosing the Right 1-Wire® Master for Embedded Applications](#)"
4. Application note 244, "[Advanced 1-Wire Network Driver](#)"
5. Application note 3684, "[How to Use the DS2482 I<sup>2</sup>C 1-Wire Master](#)"
6. Application note 3829, "[Determining the Recovery Time for Multiple-Slave 1-Wire Networks](#)"

\*Commands and states are capitalized for clarity (and the first instance is italicized).

1-Wire is a registered trademark of Maxim Integrated Products, Inc.

MICROWIRE is a registered trademark of National Semiconductor Corporation.

SensorPath is a registered trademark of Winbond Electronics Corporation.

Related Parts		
<a href="#">DS2480B</a>	Serial to 1-Wire Line Driver	<a href="#">Free Samples</a>
<a href="#">DS2482-100</a>	Single-Channel 1-Wire Master	<a href="#">Free Samples</a>
<a href="#">DS28EA00</a>	1-Wire Digital Thermometer with Sequence Detect and PIO	<a href="#">Free Samples</a>

---

#### More Information

For Technical Support: <http://www.maximintegrated.com/support>

For Samples: <http://www.maximintegrated.com/samples>

Other Questions and Comments: <http://www.maximintegrated.com/contact>

---

Application Note 4037: <http://www.maximintegrated.com/an4037>

APPLICATION NOTE 4037, AN4037, AN 4037, APP4037, Appnote4037, Appnote 4037

Copyright © by Maxim Integrated Products

Additional Legal Notices: <http://www.maximintegrated.com/legal>