Application Note 186
Creating Global Identifiers Using 1-Wire Devices

www.maxim-ic.com

INTRODUCTION

With the introduction of the microprocessor and the falling prices of silicon chips, the popularity of microprocessor-based computers grew substantially. Soon computers were connected to multiprocessor systems to solve problems faster by computing in parallel. In the next step local networks were created, to share resources and to take advantage of the parallelism of multiprocessing.

The most widely used local area network (LAN) technology is the Ethernet, which was developed by Xerox in the early 1970s. In 1983, the Institute of Electrical and Electronic Engineers (IEEE) released the first IEEE standard for Ethernet technology. The standard defines data transmission in frames, which include a Destination- and Source- Media Access Control (MAC) address for each frame. It is obvious that for a network to function these addresses have to be unique. This calls for a central organization recognized worldwide that assigns sections of the available number pool to interested parties.

In 1986, the IEEE Registration Authority was formed to register Organizationally Unique Identifiers (OUI) at the initiative of the P802 (LAN/MAN) standards group. An OUI or "company_id" is a 24-bit globally unique assigned number referenced by various standards. This number can be used to generate 48-bit Universal LAN MAC addresses to identify LAN stations uniquely, as well as protocol identifiers to identify public and private protocols. It also is used to identify hardware vendors and I/O software interface architectures. The relevant standards include CSMA/CD, Token Bus, Token Ring, FDDI, and Fibre Channel.

The OUI determines only a portion of the address or identifier. The responsibility for correctly assigning the remaining bytes of the identifier is with the company that has registered an OUI with the IEEE. Although this looks easy at the first glance, there are some tricky problems:

1) Keeping the identifiers unique (not duplicating any numbers)
2) Using the available number range economically
3) Getting the unique numbers loaded into the network interface controller

The uniqueness requires a central bookkeeping system that tracks all the numbers used so far. Since the IEEE intends not to assign another OUI or company_id value to a manufacturer until more than 90% of the range is consumed in products, it is advisable to split the available range into sections that allow different network products from the same manufacturer to be identified. This creates the risk of overlap between sections, which would violate the requirement of keeping the identifiers unique.

Once the uniqueness is managed, one has to get the number loaded into the network controller. This requires that every unit has a different number programmed into nonvolatile (NV) memory, either a special chip or some otherwise unused registers in a microcontroller or PLD. With a conventional programmer this means that one has to load a new data pattern for every single chip. This is not acceptable for production beyond prototypes.

With the introduction of 1-Wire® chips in the early 1990s and in particular with customized versions of these chips, Dallas Semiconductor defined and implemented an in-house serial number tracking system, which elegantly solves all of these problems. In 1995, the concept of UniqueWare was created to provide more flexibility. With UniqueWare, a customer-specified serialization field and constant data is factory programmed into OTP-EPROM. This application note explains how to take advantage of generic and customized 1-Wire devices as well as UniqueWare to create and manage unique identifiers.

IEEE IDENTIFIER FORMATS

The examples on the subsequent pages are based on information found on the IEEE website at http://standards.ieee.org/regauth/oui/tutorials/, and, in particular, the following documents:

- Guidelines for Use Of 24-Bit OUI/company_id Identifiers Within 48-Bit Global Identifier (EUI-48™) and 64-Bit Global Identifier (EUI-64™) When Defined By New Applications (version 31-May-2001)
- Guidelines for Use Of A 48-Bit Global Identifier (EUI-48) (version 31-May-2001)
- Guidelines For 64-Bit Global Identifier (EUI-64) Registration Authority (version 31-May-2001)
- New identifier formats based on IEEE registration (version 16-January-2001)
- Tutorial: Use of the IEEE Registration Authority assigned "company_id" with the ANSI X3.230 FC-PH Fibre Channel specification and its extensions (version 24-February-1997)

The terminology throughout these documents is not consistent. For this reason a glossary (Table 1) was compiled to explain the various technical terms in their typical context. The raw illustrations found in the referenced IEEE documents were redrawn and color was added to distinguish fields of different purpose and to visualize common characteristics. The counting of bytes, etc., is identical to that used in the IEEE documents. Although the examples shown in this application note are based on IEEE standards, the methods are valid for all applications that require a combination of a unique ("serialization") field with or without some constant data bytes.

GLOSSARY Table 1

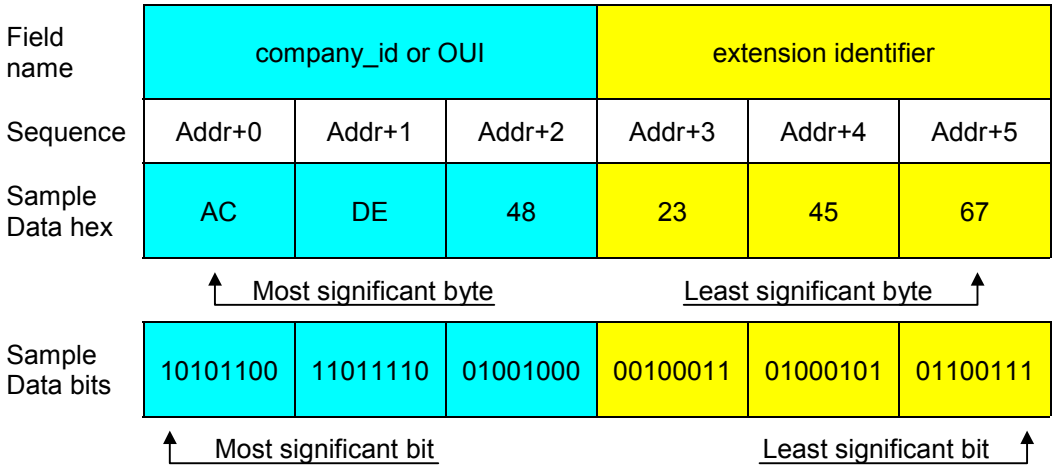
Term used in context	Context	Explanation
Organizationally Unique Identifier, OUI	MAC-48	24-bit identifier, assigned to companies by the IEEE Registration Authority Committee. Synonymous to company_id.
company_id	EUI-48, EUI-64, PC-PH	24-bit identifier, assigned to companies by the IEEE Registration Authority Committee. Synonymous to OUI.
24-bit extension identifier	MAC-48	24-bit unique number assigned and administered by a company that has an OUI.
24-bit extension identifier	EUI-48	24-bit unique number assigned and administered by a company that has a company_id.
40-bit extension identifier	EUI-64	40-bit unique number assigned and administered by a company that has a company_id.
MAC-48 identifier (Figure 1)	MAC-48	48-bit MAC (network address) for LAN. The most-significant 24 bits of this value are the OUI value. The least-significant 24 bits are the extension identifier.

EUI-48 and EUI-64 are trademarks of IEEE.

Term used in context	Context	Explanation
EUI-48 Global Identifier (Figure 1)	EUI-48	48-bit identifier used to identify a **design instance**, as opposed to a hardware instance. The most-significant 24 bits of this value are the company_id value. The least-significant 24 bits are the extension identifier.
EUI-64 Global Identifier (Figure 2)	EUI-64	64-bit identifier used to identify each **hardware instance** of a product, regardless of application. The most significant 24 bits of this value are the company_id value. The least significant 40 bits are the extension identifier.
Encapsulated MAC-48 Value	EUI-64	MAC-48 identifier "disguised" as EUI-64 identifier for transportation as EUI-64 global identifier. The upper 16 bits of the 40-bit extension identifier are set to FFFFh.
Encapsulated EUI-48 Value	EUI-64	EUI-48 global identifier "disguised" as EUI-64 identifier for transportation as EUI-64 global identifier. The upper 16 bits of the 40-bit extension identifier are set to FFFEh.
World Wide Name	FC-PH	64-bit or 128-bit identifier based on the IEEE company_id.
NAA Value	FC-PH	The upper four bits of a world wide name, which indicate the FC-PH format name, i. e., composition of the subsequent 60 bits and the possible presence of a second 64-bit field.
ULA Bytes	FC-PH	Universal LAN MAC Address, equivalent to MAC-48 address. ULA bytes 0 to 2 correspond to the company_id/OUI; ULA bytes 3 to 5 correspond to the extension identifier.
12-bit Vendor-Specified Field	FC-PH	Extension of the ULA bytes to distinguish different **ports** to a node.
36-bit Vendor-Specified Identifier	FC-PH	Identifier which uniquely indicates a **node, an N_Port, an F_Port, a Fabric, or other object**.
64-bit Vendor-Specified Identifier extension	FC-PH	Sub-identifier generated by the node to identify any Fibre Channel related **object**.
IEEE 48-bit Identifier Format (Figure 3)	FC-PH	EUI-48 global identifier "disguised" as 64-bit FC-PH World Wide Name. The NAA value 1h followed by 000h precede the EUI-48 identifier.
IEEE Extended Identifier Format (Figure 4)	FC-PH	64-bit FC-PH World Wide Name based on the six ULA bytes plus a 12-bit vendor-specified field. The NAA value 2h followed by the vendor-specified field precede the ULA bytes 0 to 5.
IEEE Registered Name Format (Figure 5)	FC-PH	64-bit FC-PH World Wide Name. The NAA value 5h is followed by the company_id and a 36-bit vendor-specified field.
IEEE Registered Extended Name Format (Figure 6)	FC-PH	128-bit FC-PH World Wide Name. The NAA value 6h is followed by the company_id, a 36-bit vendor-specified field, and a 64-bit vendor-specified identifier extension.

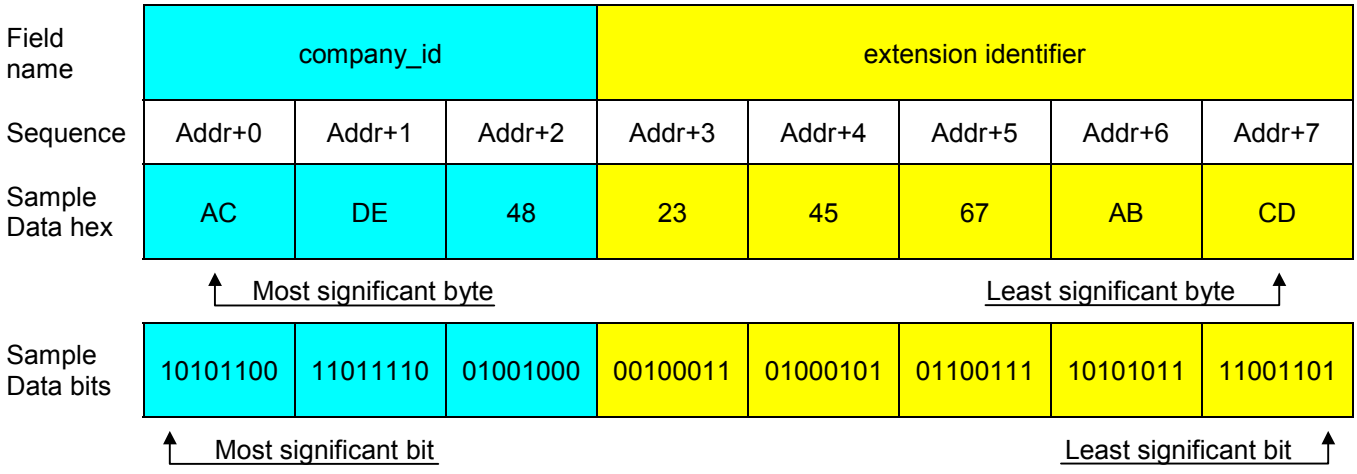
The MAC-48 (Figure 1) is the oldest of the IEEE-standardized identifiers. It consists of a 3-byte OUI and a 3-byte extension identifier, which has to be unique for any given OUI. The EUI-48 identifier is technically the same as the MAC-48. However, with the EUI-48 the 3-byte field that identifies a company is called "company_id". The actual value of OUI and company_id is the same. The sequence is identical to the memory addressing order of ISO/IEC defined memory-mapped identifiers. The most significant byte of the identifier is stored at the lower address.

MAC-48 IDENTIFIER / EUI-48 GLOBAL IDENTIFIER FORMAT Figure 1



While the number of EUI-48 identifiers is large, it is not inexhaustible. For this reason, the EUI-64 was created. As shown in Figure 2, the extension identifier is expanded to 40 bits, which expands the range from 16.777 million numbers (equivalent to 24 bits) by a factor of 65536. For byte-addressable media, the sequence is identical to the relative address of the respective byte. The most significant byte of the identifier is stored at the lower address.

EUI-64 GLOBAL IDENTIFIER FORMAT Figure 2

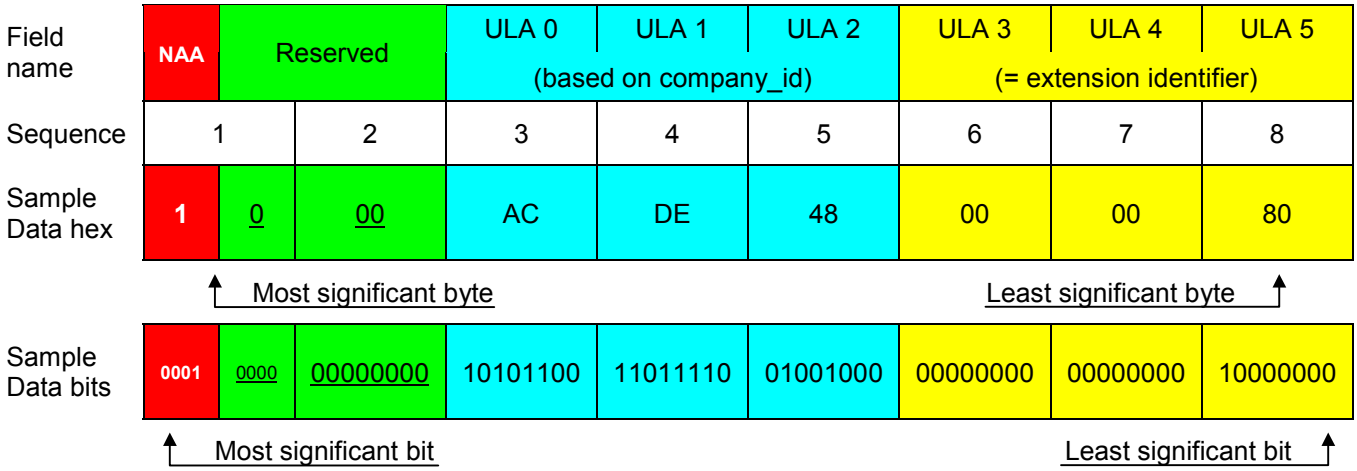


To enable migration from a 48-bit identifier to a single form of OUI/company_id based globally unique 64-bit identifiers, the IEEE has defined that the first four digits of the manufacturer's extension identifier shall not be FFFFh or FFFEh. These reserved codes are used to encapsulate and transport MAC-48 and EUI-48 identifiers as EUI-64 identifiers. The code FFFFh indicates a MAC-48 identifier; an encapsulated EUI-48 identifier is recognized by the code FFFEh.

With the introduction of the Fibre Channel, four additional 64-bit identifier formats were defined in the ANSI X3.230-1994 and ANSI X3T11 documents. For FC-PH identifier formats, a "sequence" is not explicitly defined. The format also does not specify the significance of a bit or byte; instead it specifies bit numbers. In the Fibre Channel format illustrations below, bit 0 is the same as the least significant bit; bit 63 is identical to the most significant bit.

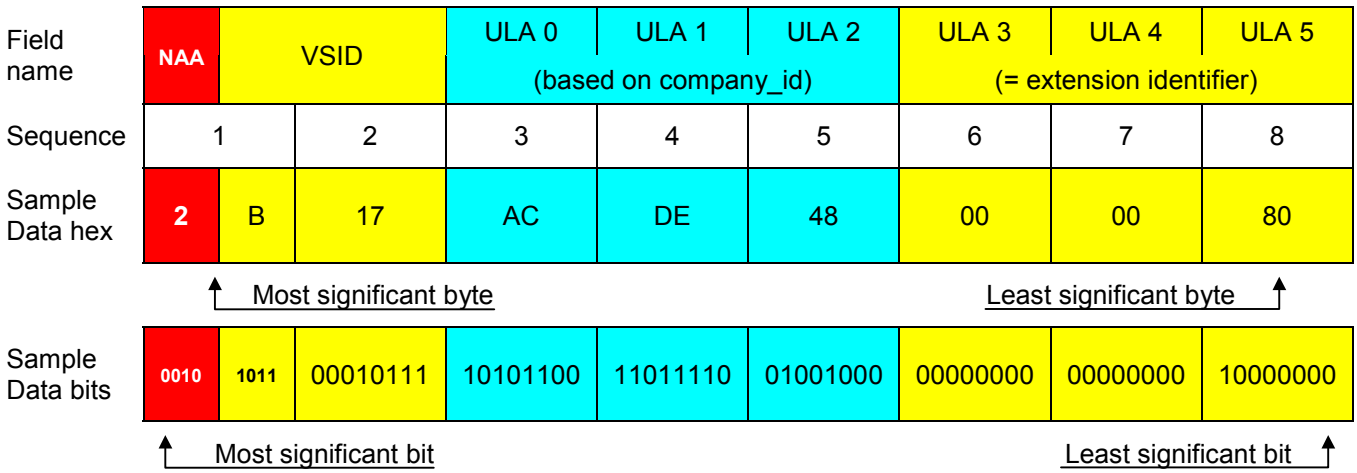
The FC-PH IEEE 48-bit identifier format, can be derived by preceding a MAC-48 identifier with the code 1000h, as shown in Figure 3. This is safe as long as company ids that begin with 1000h are not used with the EUI-64 format. Otherwise, one could not distinguish a FC-PH IEEE 48-bit identifier from an EUI-64 identifier. Instead of company_id and extension identifier, the term ULA bytes is used, which technically does not make much of a difference.

FC-PH IEEE 48-BIT IDENTIFIER FORMAT Figure 3



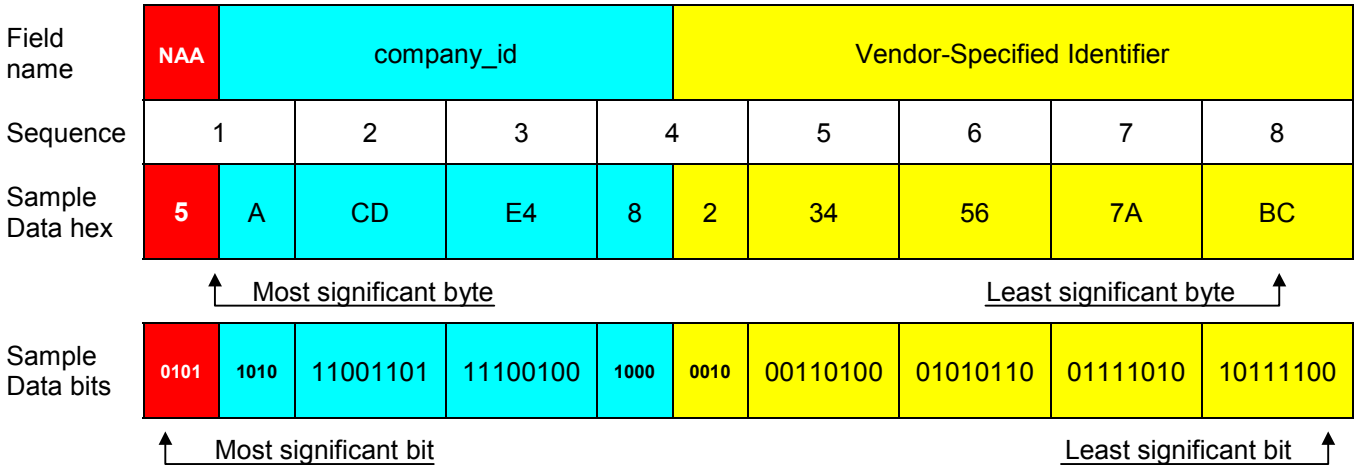
The FC-PH IEEE extended identifier format (Figure 4) is very similar to the FC-PH IEEE 48-bit identifier. Instead of the code 1000h, these identifiers begin with a '2' followed by a 3-digit VSID field, a vendor-specified identifier. This does not create confusion because company IDs beginning with a '2' are not assigned. Technically, the VSID can be regarded as extension of the extension identifier, giving the user control over a total of 36 bits, which is four bits less than with the EUI-64 identifier format.

FC-PH IEEE EXTENDED IDENTIFIER FORMAT Figure 4



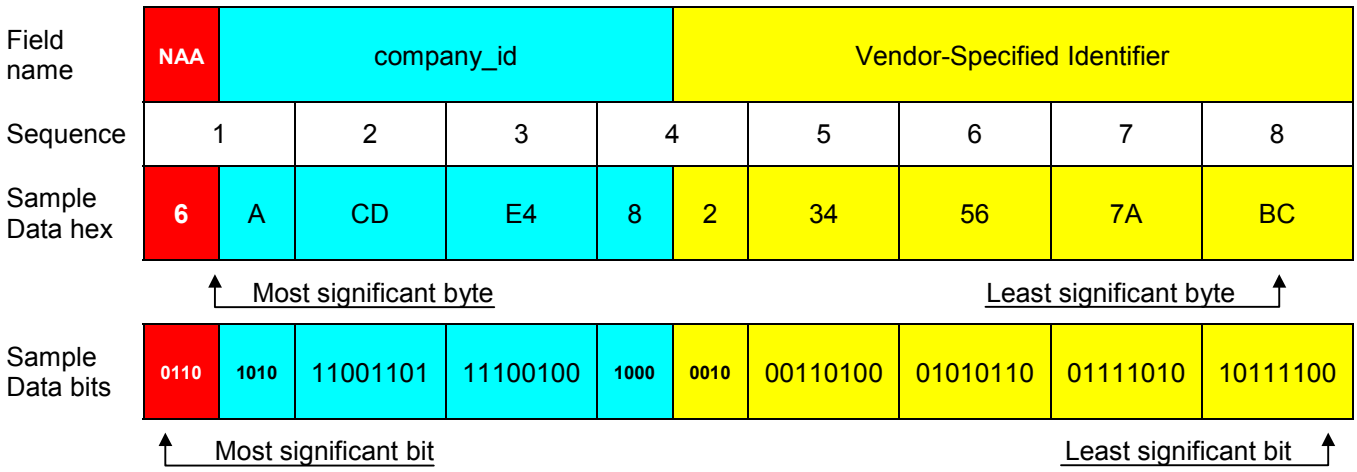
With the FC-PH IEEE registered name format (Figure 5) the 12 VSID bits are part of a contiguous 36-bit vendor-specified identifier. As a consequence, the company ID has moved up by 12 bits. This type of identifier begins with a '5', which avoids confusion with EUI-64 identifiers since company IDs beginning with a '5' are not assigned. In contrast to all previously described identifiers, with this format the boundary between the constant section and the ever-changing vendor-specified field or extension identifier is right in the middle of a byte. This creates some inconveniences when using UniqueWare to create identifiers of this type.

FC-PH IEEE REGISTERED NAME FORMAT Figure 5



The FC-PH IEEE registered extended name format has a total length of 128 bits. Except for the leading number '6', the first 64 bits (Figure 6) follow the same format as the FC-PH IEEE registered name. Since company IDs beginning with a '6' are not assigned, there is no risk of confusing a FC-PH IEEE registered name with EUI-64 identifier. The second block of 64 bits (not shown) is called Vendor-Specified Identifier Extension. There are no rules on the formatting of the data within the second block.

FC-PH IEEE REGISTERED EXTENDED NAME FORMAT (FIRST BLOCK) Figure 6



REALIZATION OF IEEE IDENTIFIERS IN SILICON

As shown in Figures 1 to 6, all identifiers consist of up to three sections:

- An IEEE-assigned 24-bit company_id,
- A space of 24, 36, 40, or 100 **variable** (unique) bits that is to be maintained by the company that is registered by the IEEE, and,
- Depending on the identifier type, an additional field of four or 16 **non-varying** bits.

For the purpose of realizing identifiers, the non-varying bits can be combined with the company_id to form the constant section of the identifier.

Using 1-Wire devices, there are essentially three methods to implement identifiers:

1) Take the 24 least significant bits of the serialization field of the ROM registration number of any **generic** 1-Wire device and use firmware to add the IEEE company_id, and, depending on the format, add the remaining bits to complete the identifier. Note that only one device type (i.e. same family code) should be used, otherwise additional risks of duplicating IEEE identifiers occur because the serialization of each device type begins at zero and increments. Therefore, a DS2401 and a DS2502, for example, could both have the same serial number if the family code is not considered.
2) Extract a **complete MAC-48/EUI-48 identifier** from the ROM registration number of a **customer-specific** 1-Wire device.
3) Read the **complete identifier** from the factory-programmed and write-protected EPROM memory of a **UniqueWare™** device.

Each method has its strengths and weaknesses, as summarized in Table 2.

COMPARISON OF IMPLEMENTATION METHODS Table 2

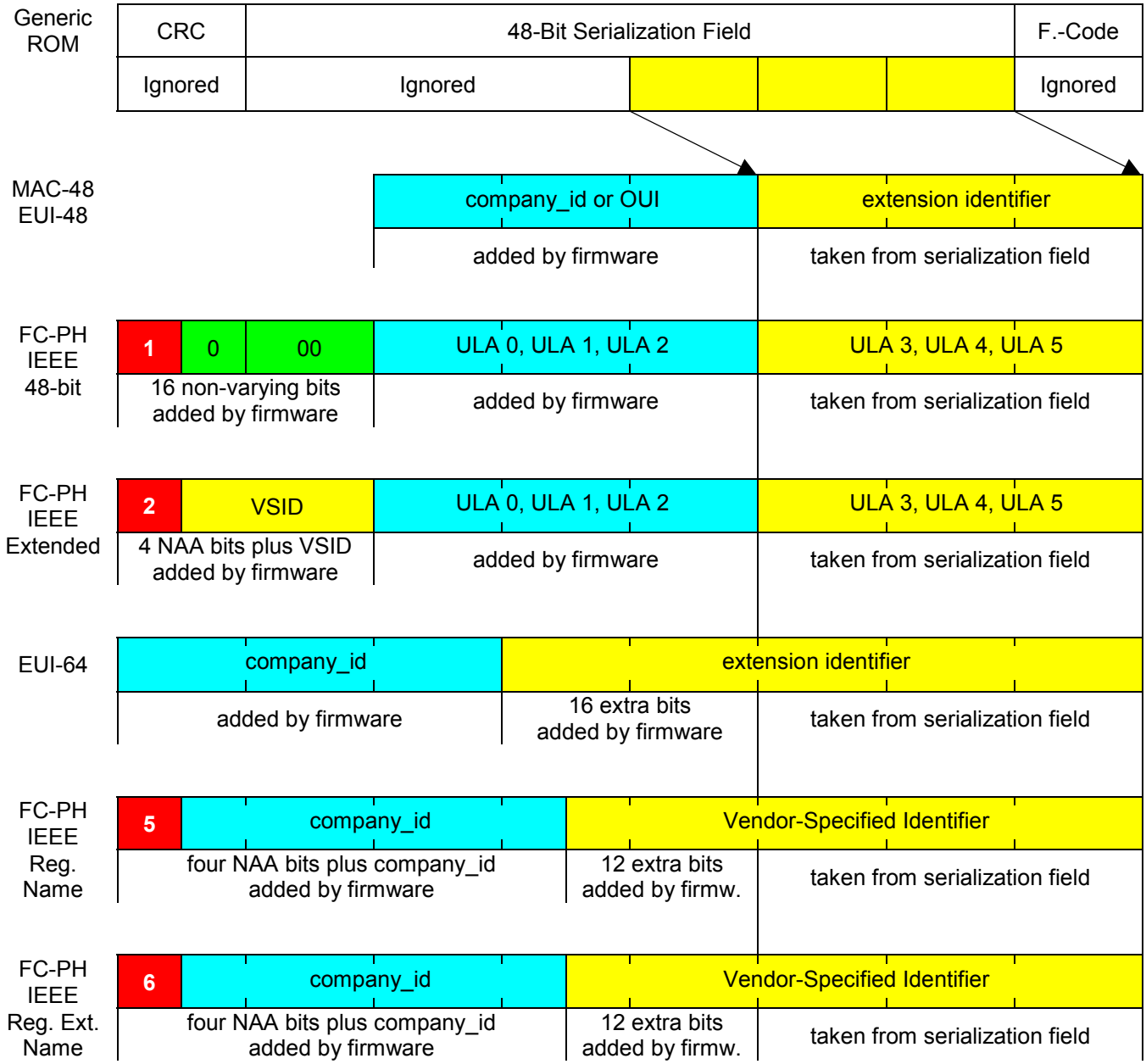
Method	Advantages	Disadvantages
Generic Parts	▪ Works with any 1-Wire chip with 64-bit ROM registration number ▪ Lower cost than customized parts ▪ Short lead-time ▪ Minimum order size: 1 part or 1 reel	▪ No control of the serialization field ▪ Special efforts in firmware to generate the complete identifier ▪ Risk of duplicating identifiers since the lower 24 bits of the ROM serialization field are not unique
Custom ROM Parts	▪ Directly implements MAC-48/EUI-48 identifiers ▪ At high volume, less costly than UniqueWare ▪ Easy setup—just fill out form	▪ Long lead-time (12 weeks) ▪ Requires firmware to generate formats other than MAC-48/EUI-48 ▪ Amortization of setup fee requires large production volume ▪ High minimum order size, ≈ 10k parts
UniqueWare	▪ Format flexibility ▪ Spare user-programmable memory for circuit board identification ▪ No setup fee ▪ Shorter lead-time than with custom ROM parts	▪ More costly than generic parts ▪ At high volume more costly than custom parts ▪ More complex setup—requires the use of special setup software ▪ Minimum order 1000 parts

UniqueWare is a trademark of Dallas Semiconductor.

All 1-Wire devices with network capability have a unique 64-bit ROM registration number. The registration number begins with an 8-bit family code followed by a 48-bit binary serialization field. The remaining eight bits are a cyclic redundancy check (CRC) of the first 56 bits. When reading the registration number, the bytes are received in this sequence: family code, 1st (= least significant) byte of the serial number, 2nd serial number byte, …, 6th (= most significant) serial number byte, CRC byte. The least significant bit of each byte is transmitted first.

In binary counting, the highest number that can be represented in 24 bits is 16777215 decimal. As long as no more than 16777215 numbers for a given family code have been used, the lower 24 bits of the serialization field are unique. Under this condition such parts can safely be used to provide the content of the variable section of IEEE identifiers, as shown in Figure 7.

CREATING IDENTIFIERS FROM GENERIC PARTS Figure 7



The 64 bits of the second block of the registered extended name format (not shown) are generated under software control.

This is how it works: take the lowest 24 bits of the serialization field and add what is missing by firmware. Although very simple at the first glance, there may be additional cost and overhead involved when making meaningful use of the 16 extra bits in the extension identifier of the EUI-64 identifier or the VSID field or the 12 extra bits in the vendor-specified identifier of the FC-PH identifier formats. It might be necessary to get PLDs or microcontrollers programmed differently to fill in the extra bits with the desired code variations. Depending on the technology, this may be just a logistical task or it could involve cost for ROM code masks.

The most likely chip for creating identifiers in this simple way is the DS2401 silicon serial number. Since its inception, however, more than 16 million parts have been built. Therefore, the lower 24 bits of the 48-bit serialization field are no longer unique, which creates the risk of duplicating IEEE identifiers. As an alternative, one could consider selecting and staying with a generic device that has a much lower production rate. However, there is no guarantee that a device that is currently produced at a lower production rate will not someday exceed the 16 million quantity and therefore also have the potential to create duplicate 24-bit values.

CUSTOM ROM VS GENERIC PARTS Figure 8

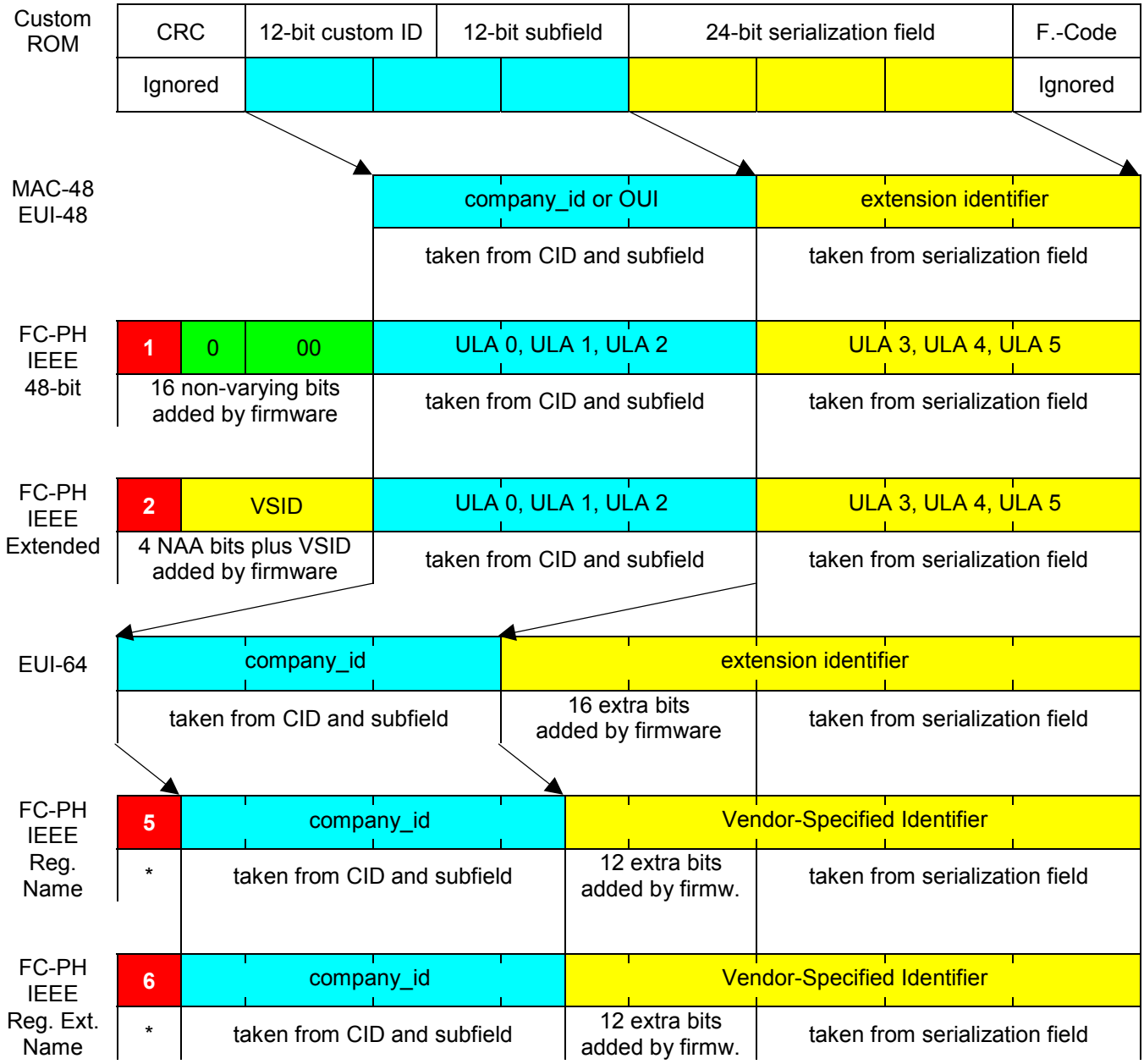
Generic ROM	CRC	48-Bit Serialization Field			F.-Code
Custom ROM **Option B**	CRC	12-bit custom ID	12-bit subfield	24-bit Serialization Field	F.-Code
Custom ROM **Option C**	CRC	12-bit custom ID	16-bit subfield	20-bit Serialization Field	F.-Code

With custom ROM parts, the 48-bit serialization field is split into three sections called custom ID, user-defined subfield, and serialization field (Figure 8). The custom ID is always 12 bits long. This field, assigned by Dallas Semiconductor at the time of registration, uniquely identifies the customer for which this part is created. The user-defined subfield is either 8 bits (option A), 12 bits (option B), or 16 bits long (option C). To distinguish generic parts from custom ROM parts, the most significant bit of the family code of custom ROM parts is set to ‘1’.

The arrangement of option B resembles the format of a MAC-48/EUI-48 identifier, if custom ID and subfield are combined to form the IEEE company_id. Option C moves four bits from the serialization field to the user-defined subfield. This allows to split the range of 16.7 million MAC-48/EUI-48 identifiers into 16 blocks of approximately one million numbers.

Starting with a MAC-48/EUI-48 identifier directly read from a custom ROM chip, further IEEE identifiers can be generated, as shown in Figure 9. The major difference to the generic parts approach is the fact that the chip already provides the company_id (more or less correct). The effort to make use of the additional 12 or 16 bits of some formats is the same as with generic parts. The main advantage of custom ROM parts over generic parts is in the control of the serialization field, which is guaranteed unique for any given combination of custom ID, subfield, and family code.

CREATING IDENTIFIERS FROM CUSTOM ROM PARTS Figure 9



* Four NAA bits added by firmware.

The 64 bits of the second block of the registered extended name format (not shown) are generated under software control.

The third way of creating IEEE identifiers in silicon is through UniqueWare, a factory-programming service of 1-Wire EPROM chips with customer-specified data that includes one serialization field. Although two text fields are possible with UniqueWare, only one text field is needed for IEEE identifiers. UniqueWare data also includes formatting such as a length byte at the beginning of the data string and a CRC16 double-byte at the end to verify data integrity. Each registered data set includes a unique 4-byte Project ID, which serves the same purpose as the custom ID of custom ROM parts. UniqueWare parts have a customized 64-bit registration number with a custom ID of 5E7h and are only available with customer-specific data programmed and write-protected. Even if the customer-specified data is the same, the finished parts can be distinguished by their project IDs.

As a more flexible and late-definition alternative to custom ROM parts, UniqueWare is well-suited to create any type of identifier as shown in Figure 10. Any of these identifiers can be composed of a hexadecimal serialization field and a constant text field. This eliminates the need to manipulate data by firmware. The text field stores the company_id and format-specific codes, as used with the FC-PH formats.

CREATING IDENTIFIERS USING UniqueWare Figure 10

MAC-48/EUI-48

Address	0C 0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text			Serialization			Project ID	Length
Use	(ignore)	company_id or OUI			extension identifier			(ignore)	(ign.)

FC-PH IEEE 48-bit

Address	0E 0D	0C	0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text					Serialization			Project ID	Length
Use	(ignore)	1	0	00	ULA 0, ULA 1, ULA 2		ULA 3, ULA 4, ULA 5			(ignore)	(ign.)

FC-PH IEEE Extended

Address	0E 0D	0C	0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text					Serialization			Project ID	Length
Use	(ignore)	2	VSID		ULA 0, ULA 1, ULA 2		ULA 3, ULA 4, ULA 5			(ignore)	(ign.)

EUI-64

Address	0E 0D	0C	0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text			Serialization					Project ID	Length
Use	(ignore)	company_id			extension identifier					(ignore)	(ign.)

FC-PH IEEE Registered Name

Address	0E 0D	0C	0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text				Serialization				Project ID	Length
Use	(ignore)	5	company_id			Vendor-Specified Identifier				(ignore)	(ign.)

FC-PH IEEE Registered Extended Name

Address	0E 0D	0C	0B	0A	09	08	07	06	05	04 01	00
UNW Field	CRC16	Constant text				Serialization				Project ID	Length
Use	(ignore)	6	company_id			Vendor-Specified Identifier				(ignore)	(ign.)

The 64 bits of the second block of the registered extended name format (not shown) are generated under software control.

The PC-PH registered (extended) name format requires a serialization field of 4.5 bytes, which is not directly supported by UniqueWare. The best compromise is selecting a 4-byte serialization field, which rolls the most significant four bits of the vendor-specified identifier into the text field. The alternate option of choosing a 5-byte serialization field and a 3-byte text field creates the risk of rollover from the serialization field into the company_id.

Rather than composing an identifier from a text field and a serialization field, one could specify a single 6-byte or 8-byte hexadecimal serialization field as well. Initially, the resulting data is exactly the same. The difference, however, becomes visible as soon as a carry from the extension identifier or vendor-specified identifier or from ULA 3 byte is due. If set up as serial number only, the carry will rollover into and change the company_id, which is not permissible. If the data is set up with serialization and text field, the rollover is prevented and the chip tester that programs UniqueWare parts automatically stops. For this reason the serial number-only approach is not recommended.

CUSTOM ROM SETUP AND ORDERING GUIDELINES

To set up a custom part one needs to fill out a 1-page Custom ROM Registration Form. The form can be ordered from the Dallas/Maxim Semiconductor Auto ID Customer Service, phone 1-972-371-6824, fax 1-972-371-6600. On the form, fill out the address section, select the device and package type, and specify whether you need tape and reel. Write the three **least-significant** digits of the IEEE company_id in the user-defined subfield of ROM option B. To indicate that you request a specific Dallas custom ID, write the three **most-significant** digits of your company_id in the CID field of the "Dallas Use Only" block. Next, forward the document to Dallas, using the fax number or e-mail address indicated on the form.

Dallas custom IDs are typically assigned in numerical order, starting with 000 and incrementing hexadecimally. According to the IEEE registration database (http://standards.ieee.org/regauth/oui/oui.txt) more than 95% of all company_ids begin with 000 to 040, a range of custom IDs that is already taken with the DS2401. To overcome this problem, when filling out the registration form, specify a Dallas custom ID that begins with 1xx to Fxx and plan to modify via firmware the bits that don't match the company_id. In most cases, the other two digits of the custom ID will match the respective digits of the IEEE company_id; the data in the subfield will always fit.

Once an agreement on the Dallas custom ID has been reached, send an order for at least the minimum order quantity plus a payment of the setup fee to the Dallas customer service. This ensures the prompt registration of a custom part. Once the registration is completed, you will receive a special part number for future orders. Custom ROM parts are non-cancelable and non-returnable.

Due to the IEEE's policy of conserving company_ids, it may be necessary to split the range of 16.7 million numbers into subsections, e. g., to share the number pool with different business units or to indicate different product revisions. To keep the door open for such a split, it is advisable to request ROM option C. The position of the three **least-significant** digits of the IEEE company_id within the user-defined subfield remains the same as with option B. The lower (rightmost) four bits of the subfield specify the range. This allows for up to 16 ranges of approximately one million parts each. The original setup fee includes one range. There may be additional fees when further ranges are activated. Every range gets its own part number. The parts of different ranges should be branded differently, which makes it possible to identify the range of a part without reading it electronically.

UniqueWare SETUP AND ORDERING GUIDELINES

UniqueWare requires that the customer uses a special program to generate a data file that controls the tester that finally programs the chips. The UniqueWare setup software can be downloaded from ftp://ftp.dalsemi.com/pub/auto_id/ as a self-extracting file "unwsetup.exe". Detailed instructions are found in the UniqueWare Project Setup Manual Revision 2.00B.

Once the software is installed, fill out the address section, select the device type (typically a 1024-bit device, DS2502 or DS1982) and select the file size option "no file name". The number of bytes for serialization is either 3 (for MAC-48/EUI-48, FC-PH IEEE 48-bit as well as extended), 5 (EUI-64), or 4 with the PC-PH registered (extended) name format. The counting style is hexadecimal with binary number representation. The serialization start number typically is zero, unless one intends to split the number range into several blocks. In the text section specify "one text following the serialization field". **The text does not consist of characters only.** In any case and format, it is advisable to first write the text data into Figure 10, just above the address line of the desired format. Now enter the text into the setup software starting with the byte at the lowest text address and then *go left* to the next higher address until all bytes of the text section are entered. Only this sequence ensures that the data will be read from the programmed parts in the correct way.

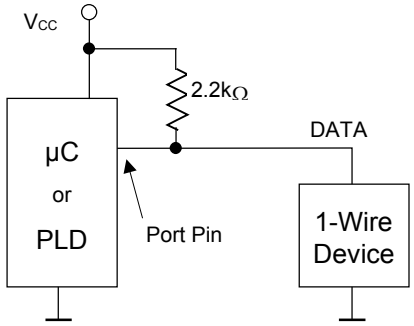
When the UniqueWare data file is completed, e-mail it as attachment to AutoID.Support@dalsemi.com. You can request prototype samples made from generic chips for verification. Approve samples and place order for the first batch of parts with the Dallas Semiconductor customer service, phone 1-972-371-6824, fax 1-972-371-6600. This is the precondition for getting a project ID assigned. The project ID is included in the special part number that you receive for future orders. UniqueWare parts are non-cancelable and non-returnable.

Similar to custom ROM, UniqueWare also allows splitting the available number pool into several ranges. To split the range, create multiple data files that are identical except for the serialization start number. Each data file will be processed individually, i. e., it will get its unique project ID and there will be no automatic cross-checks against earlier projects for potential serialization overlap. Therefore, when submitting another file for a previously established identifier format, express in writing which project ID(s) this new file is related to and specify the caps (= upper serialization limits) that are to be set for these project ID(s). This is the only way to prevent serialization overlap. In contrast to custom ROM parts, UniqueWare devices all look the same; **the project ID is not branded on the package.**

INTERFACING THE 1-Wire DEVICE

Since they require only a single data line plus ground reference, 1-Wire devices are very easy to interface. Most 1-Wire devices do not even have a V_{CC} pin; they draw the energy for operation right from the data line. Figure 11 shows a simple 1-Wire interface that fits to a bidirectional port, such as the open-drain port 0 of an 8051-compatible microcontroller. The protocol to read the 1-Wire device is generated under software control. A detailed discussion of 1-Wire interfaces including software examples is found in Application Note *Reading and Writing iButtons via Serial Interfaces*. For an dual-read mode interface that allows one to read a 1-Wire device by an external reader while the application circuit is powered down, see Application Note *Printed Circuit Board Identification Using 1-Wire Products*.

MINIMAL 1-Wire INTERFACE Figure 11



APPLICATION EXAMPLE

The TINI® board, which is featured for its electronic nameplate in the Application Note 178, *Printed Circuit Board Identification Using 1-Wire Products*, actually has an Ethernet Address, implemented as UniqueWare in a DS2502 chip. The data is set up according to Figure 10, MAC-48/EUI-48 format. The least significant byte of a numerical field is stored at the lower address. Figure 12 shows the UniqueWare data read from a particular TINI board. In contrast to Figure 10, the addresses are **ascending from left to right**. This data pattern is registered as project ID 00001129. The extension identifier is 0093B8. The IEEE company_id of Dallas Semiconductor is 006035, which can be verified at the IEEE registration database. Since the UniqueWare data consumes only one memory page, the remaining three memory pages were available for circuit board identification **at no extra cost**.

UniqueWare ETHERNET ADDRESS Figure 12

Address	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000x	0A	29	11	00	00	B8	93	00	35	60	00	68	59	FF	FF	FF
001x	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

Color Code Legend:

Length Byte
UniqueWare Project ID
UniqueWare constant text field

CRC16
UniqueWare serialization
(no color = unused byte)

REFERENCES

http://www.techfest.com/networking/index.htm
Links to resources on networking and other computer-related information. TechFest.com is the part-time project of a technical professional who works in the computer networking industry. The number of links and the amount of information is overwhelming.

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/index.htm
This link leads to very professional documents on various types of networks and protocols. Most of the documents are excerpts of the *Internetworking Technologies Handbook*, ISBN 1-58705-001-3, a comprehensive reference for networking professionals. The files on the web are not a substitute for the book, since many graphics are missing.

TINI is a registered trademark of Dallas Semiconductor.