Application Report
SNLA244–January 2016

Understanding EEPROM Programming for 25-G to 28-G Repeaters and Retimers

Michael Lu

ABSTRACT

System designers often use EEPROM (Electrically Erasable Programmable Read-Only Memory) to program a set of customized start-up settings that are different from the factory default. Using the information here will make EEPROM configuration and programming easy to implement and understand for the DS250DF810 retimer and the DS280BR810 repeater. With a complete understanding of how to interpret and program EEPROM hex files for TI's 25 to 28-Gbps repeaters and retimers, system designers are better equipped to generate their own customized hex files and increase the efficiency of their final designs.

Contents

1 Introduction

EEPROM is non-volatile memory used in electronic devices to store data that must be saved when power is removed. This non-volatile memory is particularly important when an application requires different start-up configurations than the factory default settings. Upon device power-up, data saved in the EEPROM will load automatically to the device. If EEPROM is not used, interface system designs require external access to the SMBus SDA and SCL line in order to set individual registers after each power-up. With EEPROM, designers eliminate the requirement for an external microprocessor or software driver to provide their desired register settings.

Programming EEPROM for TI's 25 to 28-Gbps repeaters and retimers requires an understanding of the relationship between EEPROM register bits and Slave Mode register bits. Like their 10 to 12.5-Gbps retimer counterparts, the SMBus-to-EEPROM bit mapping for 25 to 28-Gbps devices incorporate multiple register pages per device and feature Common Channel Configuration. The primary challenge associated with programming 25 to 28-Gbps devices is that each IC package may contain more than one die.

This application note is applicable to the following 25-28G products:

Table 1. Applicable 25 to 28-G Devices

DEVICE TYPE	DEVICES INCLUDED	DIES PER PACKAGE
8-Channel Repeater	DS280BR810	2
8-Channel Retimer	DS250DF810	2

As a prerequisite, it is assumed that the reader is already familiar with the following high speed device EEPROM topics:

- How to configure high speed devices to operate in EEPROM Master Mode
- How to read EEPROM hex format
- How to calculate the CRC-8 value from a given bit stream of values
- Difference between Number of Slots versus Number of Devices
- EEPROM Memory Page Addressing
- EEPROM Programming for Devices with Multiple Register Pages
- Start Address Indexing without Address Map Headers for 10 to 12.5-Gbps Retimers (see SNLA245)
- Common Channel Configuration for 10 to 12.5-Gbps Retimers (see SNLA245)

If the aforementioned topics are unfamiliar, please reference *Understanding EEPROM Programming for High Speed Repeaters and Mux Buffers* (SNLA228) and *Understanding EEPROM Programming for 10-G to 12.5-G Retimers* (SNLA245) for more details.

2 EEPROM Device Data Fundamentals for Multi-Die Devices

The 25 to 28-Gbps devices use the same EEPROM format as the 10 to 12.5-Gbps retimers for writing Base Header Bytes and Address Map Header Bytes. A summary of the Base Header and Address Map Header EEPROM format is provided in this application note. The following subsections detail how the Base Header, Address Map Headers, and Register Data Slots are affected by the multi-die 25 to 28-Gbps devices.

2.1 Device Slot versus Die Slot

When describing the EEPROM for multi-die devices, the term *device slot* and the term *die slot* must be distinguished to avoid confusion. In this application note, *device slot* is defined as all EEPROM data settings pertaining to one 25 to 28-Gbps repeater or retimer device. *Die slot* is defined as the EEPROM data settings pertaining to a particular die within one 25 ro 28-Gbps repeater or retimer device.

2.2 Multi-Die Base Headers

2.2.1 Base Header Definitions

The first three Bytes define the Base Header. Table 2 explains the meaning of the Base Header Bytes.

Table 2. Base Header Information

BYTE	BIT(S)	BIT NAME	DESCRIPTION
0	7	CRC_EN	1 = CRC enable. If enabled, each die slot will have a CRC value calculated from all the Bytes used by the die to load from EEPROM.
	6	ADDR Map Enable	1 = Address Map Header enable. If enabled, a 2 or 3 Byte Address Map Header will be placed after the Base Header to indicate the start address of each die's configuration data.
	5	EEPROM > 256 Bytes	1 = Required EEPROM size is more than 256 Bytes. This must be enabled if a die slot must load from an EEPROM memory location greater than 0xFF.
	4	COMMON_CHANNEL	1 = Common Channel Configuration enable. If enabled, the settings for all channels are referenced from one Channel Register's settings.
	3:0	DEVICE COUNT	(Total number of Devices) - 1. Note: This value is not used by the device when the EEPROM loads data, though it is a useful debugging reference.
1	7:0	RES	Reserved. Set bits to 0.
2	7:0	Max EEPROM Burst Size	Maximum number of Bytes that are read during a burst read operation. A value of 0x10 is suitable for all supported EEPROMs.

2.2.2 Common Channel Configuration in Multi-Die Devices

When Common Channel Configuration is used for multi-die devices, it is possible to have a common set of register settings for all the channels of Die 0 and a different common set of register settings for all the channels of Die 1. This can be achieved if the Address Map Header for Die 1 points to different starting address than the Address Map Header for Die 0. However, in most applications it is recommended that the Address Map Header for Die 0 and Die 1 point to the same location. This is because in Common Channel Configuration, typically Channels 0 to 7 all share the same Channel and Share Register settings. Using the same starting address for both dies minimizes the amount of EEPROM memory space required.

2.3 Multi-Die Address Map Headers

2.3.1 Address Map Header Definitions

If Base Header Byte 0, Bit 6 = 1, Address Map Headers are used. The Address Map Header specifies the memory location where each device's die begins reading its programmed configuration data settings. Table 3 explains the meaning of the Address Map Header Bytes.

Table 3. Address Map Header Information

BYTE	BIT(S)	BIT NAME	DESCRIPTION
0	7:0	CRC Value	8-Bit CRC value for each die. CRC is computed from all the Bytes used by the relevant die to load from EEPROM.
1	7:0	EEPROM Die Start Address	Start address for the Channel Configuration page of the relevant die's EEPROM.
2 (1)	7:3	RES	Reserved. Set bits to 0.
	2:0	EEPROM Die Start Address MSBs	These bits are only set if EEPROM Size > 256 Bytes. Up to 3 MSBs can be appended to the front of the EEPROM start address indicated in Byte 1.

(1) Byte 2 is present only if the EEPROM > 256 Bytes Enable Bit is set by asserting Base Header Byte 0, Bit 5 = 1. For example, if the EEPROM start address is located at Address 0x1F4, 9 bits are required. Thus, Address Map Header Byte 1 = 0xF4, and Address Map Header Byte 2 = 0x01. If EEPROM ≤ 256 Bytes, then the Address Map Header will be 2 Bytes, not 3 Bytes.

The maximum EEPROM size allowed for the 25 to 28-G devices is 16 Kbits (2048 Bytes). As with the 10 to 12.5-Gbps retimer devices, if EEPROM size > 256 Bytes, the EEPROM must support page addressing in order to be used properly with TI high speed devices.

The Address Map Header for each device is located immediately after the 3 Base Header Bytes. The 25 to 28-G device obtains the starting memory location of its Address Map Header by determining its SMBus address index, I_{SMB_ADDR}, and then computing the start address of its corresponding Address Map Header, $ADDR_{MAP\_START}$, as shown in Table 4.

Table 4. I_{SMB_ADDR} and $ADDR_{MAP\_START}$ Locations

RETIMER SMBus WRITE ADDRESS	I_{SMB_ADDR} (SMBus ADDRESS INDEX)	$ADDR_{MAP\_START}$ (ADDRESS MAP MEMORY LOCATION)	
		If EEPROM size ≤ 256 Bytes	If EEPROM size > 256 Bytes
0x30	0	3	3
0x32	1	5	6
0x34	2	7	9
0x36	3	9	12
0x38	4	11	15
0x3A	5	13	18
0x3C	6	15	21
0x3E	7	17	24
0x40	8	19	27
0x42	9	21	30
0x44	10	23	33
0x46	11	25	36
0x48	12	27	39
0x4A	13	29	42
0x4C	14	31	45
0x4E	15	33	48

2.3.2 Address Map Headers in Multi-Die Devices

In the 25 to 28-Gbps devices, there is one Address Map Header per die slot. Because there are multiple dies per device, there are multiple Address Map Headers per device slot. Although both the DS250DF810 and the DS280BR810 have two dies per package, the EEPROM Address Map Header always allocates space for **four** Address Map Headers per device slot. Since there are not actually four dies within the DS250DF810 and the DS280BR810 ICs, the Address Map Header for the third and fourth die can be populated with the same header as Die 0 in order to ensure that the EEPROM hex file is valid.

2.3.3 EEPROM Configuration in Multi-Die Devices without Address Map Headers

If Address Map Headers are not used, the DS250DF810 and DS280BR810 compute a fixed starting memory location for each device slot, as well as the start location of each die slot within the device. To do this, the device first determines its SMBus address index, I_{SMB_ADDR}, in the array of permissible SMBus Write Address Bytes, shown previously in Table 4.

The 25 to 28-Gbps device then computes the length of the data slot, N_{DATA_SLOT}. The length of each data slot is determined by device IC and whether the Common Channel bit is enabled in the Base Header. The following is an example of the N_{DATA_SLOT} calculation when the DS250DF810 retimer is used.

Table 5. N_{DATA_SLOT} Calculation per Die for DS250DF810

	COMMON_CHANNEL = 1	COMMON_CHANNEL = 0
Channel Register Bytes	72 × 1 Channel = 72	72 × 4 Channels = 288
Share Register Bytes	2	2
CRC Byte[(1)]	1	1
N_{DATA_SLOT} (Total Bytes per Device Slot)	**75**	**291**

(1) Without Address Map Headers, the CRC Byte is programmed as an extra Byte after the last Share Register Byte in each data slot. This Byte is still present even if CRC is not enabled in the Base Header. For multi-die devices, a CRC Byte is allocated for each die.

There is one significant difference when computing the $ADDR_{DATA\_START}$ location for these multi-die devices without Address Map Headers. When Address Map Headers are not used, the 25 to 28-Gbps devices assume that there are N_{DATA_SLOT} Bytes in EEPROM for all four dies, despite there only being two physical dies per device slot. Therefore, the start addresses for each die slot are computed as follows:

- $ADDR_{DATA\_START\_DIE0} = 3 + (I_{SMB\_ADDR} \times \{4 \times N_{DATA\_SLOT}\})$
- $ADDR_{DATA\_START\_DIE1} = 3 + (I_{SMB\_ADDR} \times \{4 \times N_{DATA\_SLOT}\}) + (1 \times N_{DATA\_SLOT})$
- $ADDR_{DATA\_START\_DIE2} = 3 + (I_{SMB\_ADDR} \times \{4 \times N_{DATA\_SLOT}\}) + (2 \times N_{DATA\_SLOT})$
- $ADDR_{DATA\_START\_DIE3} = 3 + (I_{SMB\_ADDR} \times \{4 \times N_{DATA\_SLOT}\}) + (3 \times N_{DATA\_SLOT})$

NOTE: An offset of three Bytes accounts for the Base Header Bytes.

As an example, consider the case where Common Channel Configuration is used and a DS250DF810 with SMBus Address 0x36 attempts to load settings from EEPROM. Referencing Table 4 and Table 5, $I_{SMB\_ADDR} = 3$ and $N_{DATA\_SLOT} = 75$. Therefore, the device slot's start address $ADDR_{DATA\_START}$ per die can be calculated as follows:

- $ADDR_{DATA\_START\_DIE0} = 3 + (3 \times \{4 \times 75\}) = 903$ or 0x387
- $ADDR_{DATA\_START\_DIE1} = 3 + (3 \times \{4 \times 75\}) + (1 \times 75) = 978$ or 0x3D2
- $ADDR_{DATA\_START\_DIE2} = 3 + (3 \times \{4 \times 75\}) + (2 \times 75) = 1053$ or 0x41D
- $ADDR_{DATA\_START\_DIE3} = 3 + (3 \times \{4 \times 75\}) + (3 \times 75) = 1128$ or 0x468

When Address Map Headers are not used, the Base Header Byte 0, Bit 5 (EEPROM Size > 256 Bytes Enable Bit) must be set to 1 if the physical EEPROM memory space is greater than 256 Bytes. If this Base Header bit is not asserted, the device will not calculate the correct memory locations when the desired EEPROM address Byte exceeds Address 0xFF.

Due to the complexity of configuring EEPROM for multi-die devices and the excess unused EEPROM memory space required for the nonexistent Die 2 and Die 3, users are discouraged from programming more than one device when Address Map Headers are not used. Address Map Headers not only simplify EEPROM programming for multi-die devices, but also more efficiently uses EEPROM memory space.

2.4 Multi-Die Register Data Slot Settings

The DS250DF810 and DS280BR810 have two dies per device. In each die, the SMBus register map architecture uses one global Share Register Page and four individual Channel Register Pages. Therefore, with two dies, Die 0 contains one set of Share Register data and Channel Register data related to Channels 0 to 3, and Die 1 contains a different set of Share Register data and Channel Register data related to Channels 4 to 7.

2.4.1 Multi-Die Slot Mapping

When mapping either the DS250DF810 or DS280BR810 SMBus register pages to EEPROM, the EEPROM data for Channel 0 is programmed first, followed by Channels 1, 2, 3, and lastly, the Share Register data for Die 0. The EEPROM data for the second die immediately follows, with EEPROM data for Channel 4, then Channels 5, 6, 7, and the Share Register data for Die 1.

2.4.2 EEPROM Data for the DS250DF810 and DS280BR810

While an EEPROM allows devices to start up with settings different than the factory default, the EEPROM only maps a subset of the SMBus register bits. SMBus register bits in the device that are not designed for EEPROM cannot be changed from default at device startup. The number of Bytes mapped to EEPROM per Channel Register and Share Register Page varies depending on the device type. The difference in Byte size per device type is summarized in Table 6.

Table 6. Number of Bytes Mapped to EEPROM per 25-G to 28-G Device

DEVICE TYPE	FROM CHANNEL REGISTER PAGE	FROM SHARE REGISTER PAGE	DEVICES INCLUDED
8-Channel Repeater	8	1	DS280BR810
8-Channel Retimer	72	2	DS250DF810

For detailed information about bit mapping from Channel Register and Share Register Pages to EEPROM, refer to the tables in Section 6.

2.4.3 CRC Calculation for Multi-Die Devices

The CRC for each die slot is calculated by evaluating the CRC-8 polynomial for all the Bytes each die uses when loading its die slot settings from EEPROM.

2.4.3.1 CRC Calculation with Address Map Header

If the Address Map Header is enabled by asserting Base Header 0, Bit 6 = 1, the CRC Byte for each die slot is computed using the following Bytes:

- Base Header (3 Bytes)
- Non-CRC Bytes of the Die Slot's Address Map Header (1 or 2 Bytes)
- Die Slot's Channel Register Data (8 or 72 Bytes per Channel)
- Die Slot's Share Register Data (1 or 2 Bytes)

When the Address Map Header is enabled, the CRC Byte for each die is stored in the respective die's Address Map Header Byte 0.

2.4.3.2 CRC Calculation without Address Map Header

If the Address Map Header is disabled by asserting Base Header 0, Bit 6 = 0, the CRC Byte for each die slot is computed using the following Bytes:

- Base Header (3 Bytes)
- Die Slot's Channel Register Data (8 or 72 Bytes per Channel)
- Die Slot's Share Register Data (1 or 2 Bytes)

When the Address Map Header is disabled, the CRC Byte for each die is stored in the Byte location immediately following the last Share Register Byte of that particular die slot.

3 Example 1: EEPROM Hex File for 2 Devices, CRC Disabled

A simple case for programming EEPROM is shown here with two multi-die devices. In this example, the following settings are desired.

- Enable Address Map Header.
- Enable Common Channel Configuration.
- Set EEPROM ≤ 256 Bytes.
- Device Count = 2.
- Disable CRC Byte. Since CRC is disabled, the first Address Map Header Byte value for each Die Slot Address Map Header is arbitrary.

An example of a hex file with two DS280BR810 devices that meet these requirements is shown below.

```
:20000000510010001300130013001300550055005500558640 3A0A6D42000000000000026
:20002000000000000000000000000000000000000000000000000000000000000000000C0
:20004000000000000000000000000000000000000000000086403A0A6D42000000000E7
:20006000000000000000000000000000000000000000000000000000000000000000000080
:20008000000000000000000000000000000000000000000000000000000000000000000060
:2000A000000000000000000000000000000000000000000000000000000000000000000040
:2000C000000000000000000000000000000000000000000000000000000000000000000020
:2000E000000000000000000000000000000000000000000000000000000000000000000000
:00000001FF
```

= EEPROM Base Header

= Device 0 Address Map Header, Die 0
= Device 0 Address Map Header, Die 1
= Device 0 Address Map Header, Die 2
= Device 0 Address Map Header, Die 3
= Device 0 Common Channel Register Data
= Device 0 Share Register Data

= Device 1 Address Map Header, Die 0
= Device 1 Address Map Header, Die 1
= Device 1 Address Map Header, Die 2
= Device 1 Address Map Header, Die 3
= Device 1 Common Channel Register Data
= Device 1 Share Register Data

From the DS280BR810 hex file, the Base Header Bytes are 0x510010. From Table 2, this means:

- CRC is disabled (Reg 0x00[7] = 0'b).
- Address Map Header is used (Reg 0x00[6] = 1'b).
- EEPROM ≤ 256 Bytes (Reg 0x00[5] = 0'b).
- Common Channel is enabled (Reg 0x00[4] = 1'b).
- DEVICE COUNT = 2 Devices (Reg 0x00[3:0] = 0001'b).
- Max EEPROM Burst size = 16 Bytes (Reg 0x02 = 0x10).

Since the DS280BR810 is a multi-die device, there are four Address Map Headers allocated per device slot, with one Address Map Header for each die. Because EEPROM size ≤ 256 Bytes, the Address Map Header for each die is two Bytes long. The starting address location for the two devices can be determined by the non-CRC Bytes of these Address Map Headers:

- Device 0, Die 0 [CRC, Start Address] = [0x00, 0x13]
- Device 0, Die 1 [CRC, Start Address] = [0x00, 0x13]
- Device 0, Die 2 [CRC, Start Address] = [0x00, 0x13]

- Device 0, Die 3 [CRC, Start Address] = [0x00, 0x13]
- Device 1, Die 0 [CRC, Start Address] = [0x00, 0x55]
- Device 1, Die 1 [CRC, Start Address] = [0x00, 0x55]
- Device 1, Die 2 [CRC, Start Address] = [0x00, 0x55]
- Device 1, Die 3 [CRC, Start Address] = [0x00, 0x55]

Since Common Channel Configuration is enabled, all channels in each die derive their device settings from only one set of Channel Register settings, followed immediately by the die's Share Register data. In this example, Channel Register and Share Register EEPROM data are the same for all channels. Therefore, both Die 0 and Die 1 point to the same start address to conserve EEPROM memory. The Channel and Share Register EEPROM data for Device 0, Die 0 and Die 1, are shown in Table 7.

Table 7. Device 0 Settings for Die 0 and Die 1

Channel Register Data	`0x86403A0A6D420000`
Share Register Data	`0x00`

NOTE: Address Map Headers for Die 2 and Die 3 are still present, though their values are arbitrary and ignored because these dies do not exist in the IC. To output a valid EEPROM hex file, these Address Map Headers are populated with the same Address Map Header values as Die 0.

4 Example 2: EEPROM Hex File for 1 Device, No Address Map Header

When programming a single device, an Address Map Header is not always necessary. In this example, the following settings are desired.

- Enable CRC Byte.
- Disable Address Map Header.
- Enable Common Channel Configuration.
- Device Count =1.
- Set EEPROM < 256 Bytes.

An example of a hex file with a DS250DF810 that meets these requirements is shown below.

```
:20000000900010077300063021 0F00A0F46D102DC9E5600015A2BB182A202310FC7B600036
:200020001A4040004050809 0C0D0D1D5D8EAF7FDEEEFFF0000000000034428000000521346E
:200040004813A9001F9FF12248C300900F87067300063021 0F00A0F46D102DC9E560001 55A
:20006000A2BB182A202310FC7B60001A4040004050809 0C0D0D1D5D8EAF7FDEEEFFF0000B5
:200080000000003442800000052134 4813A9001F9FF12248C300900F15000000000000007C
:2000A000000000000000000000000000000000000000000000000000000000000000000040
:2000C000000000000000000000000000000000000000000000000000000000000000000020
:2000E000000000000000000000000000000000000000000000000000000000000000000000
:00000001FF
```

= EEPROM Base Header

= Device 0 Common Channel Register Data, Die 0

= Device 0 Share Register Data, Die 0

= Device 0 CRC-Byte, Die 0

= Device 0 Common Channel Register Data, Die 1

= Device 0 Share Register Data, Die 1

= Device 0 CRC Byte, Die 1

In this DS250DF810 hex file, the Base Header Bytes are 0x900010. From Table 2, this means:

- CRC is enabled (Reg 0x00[7] = 1'b).
- Address Map Header is not used (Reg 0x00[6] = 0'b).
- EEPROM ≤ 256 Bytes (Reg 0x00[5] = 0'b).
- Common Channel is enabled (Reg 0x00[4] = 1'b)
- DEVICE COUNT = 1 Device (Reg 0x00[3:0] = 0000'b).
- Max EEPROM Burst size = 16 Bytes (Reg 0x02 = 0x10).

Since an Address Map Header is not used, the first device slot begins immediately after the Base Header. Without Address Map Headers, each die's start address has a fixed memory location. Therefore, the Channel Register data, Share Register data, and CRC Byte for each die are stored in separate memory locations.

When Common Channel Configuration is enabled, all channels derive their device settings from only one set of Channel Register settings, followed immediately by the die's Share Register data. Because the DS250DF810 is a multi-die device, Die 0's Common Channel Register data comes first, followed by its Share Register data and CRC Byte. Immediately afterwards, Die 1's Common Channel Register data, Share Register data, and CRC Byte follow. In this example, Die 0 and Die 1 have a different set of Common Channel Register settings, therefore leading to different CRC Byte values. The Channel and Share Register EEPROM data for Die 0 and Die 1 in this example are shown in Table 8.

Table 8. Slot Settings for Die 0 and Die 1

Channel Register Data, Die 0 (CH 0-3)	0x0773000630210F00A0F46D102DC9E5600015A2BB182A202310FC7B60001A40 400040508090C0D0D1D5D8EAF7FDEEEFFF0000000000344280000000521344813 A9001F9FF12248C300
Share Register Data, Die 0	0x900F
Channel Register Data, Die 1 (CH 4-7)	0x0673000630210F00A0F46D102DC9E5600015A2BB182A202310FC7B60001A40 400040508090C0D0D1D5D8EAF7FDEEEFFF0000000000344280000000521344813 A9001F9FF12248C300
Share Register Data, Die 1	0x900F

The CRC Byte is calculated from the Base Header and relevant die slot Bytes. The values used for the CRC calculation in this example are shown in the following calculations.

Device 0, Die 0 CRC Calculation

```
900010 0773000630210F00A0F46D102DC9E5600015A2BB182A202310FC7B60001A4040004050 8090C0D0D1
D5D8EAF7FDEEEFFF0000000000034428000000521344813A9001F9FF12248C300 900F
```

CRC-8 calculation of the above data Bytes for Die 0 yields CRC = 0x87.

Device 0, Die 1 CRC Calculation

```
900010 0673000630210F00A0F46D102DC9E5600015A2BB182A202310FC7B60001A4040004050 8090C0D0D1
D5D8EAF7FDEEEFFF0000000000034428000000521344813A9001F9FF12248C300 900F
```

CRC-8 calculation of the above data Bytes for Die 1 yields CRC = 0x15.

NOTE: Since the DS250DF810 is a two-die device, there is no practical need to search for EEPROM data regarding Die 2 and Die 3. However, because memory locations are fixed when Address Map Headers are not used, memory space is still reserved for the presence of Die 2 and Die 3 EEPROM data, even though Die 2 and Die 3 do not exist in the silicon. The CRC Bytes for Die 2 and Die 3, since these dies do not exist in the IC package, are ignored.

5 Example 3: EEPROM Hex File for 4 Devices, CRC Enabled

When programming multiple EEPROM device slots, Address Map Headers are recommended. In this example, the following settings are desired:

- Enable CRC Byte.
- Enable Address Map Header.
- Disable Common Channel Configuration.
- Device Count = 4.
- Set EEPROM > 256 Bytes.

An example of a hex file with 4 x DS280BR810 devices (4 unique EEPROM slots) that are CRC enabled is shown below.

```
:20000000E300108F33008F54008F33008F33006A7500E496006A75006A750001B70042D8DB
:200020000001B70001B700A7F9007D1A01A7F900A7F90087403A0A6D42000080403A0A6DAD
:2000400042000080403A0A6D42000080403A0A6D4200000080403A0A6D42000080403A0A41
:200060006D42000080403A0A6D42000086403A0A6D4200000087403A0A6D42000080403AB1
:200080000A6D42000080403A0A6D42000080403A0A6D4200000080403A0A6D4200008040CE
:2000A0003A0A6D42000080403A0A6D42000086403A0A6D4200000087403A0A6D42000080A7
:2000C000403A0A6D42000080403A0A6D42000080403A0A6D4200000080403A0A6D420000D4
:2000E00080403A0A6D42000080403A0A6D42000086403A0A6D4200000087403A0A6D420027
:2001000000080403A0A6D42000080403A0A6D42000080403A0A6D4200000080403A0A6D4213
:20012000000080403A0A6D42000080403A0A6D42000086403A0A6D420000000000000000A0
:200140000000000000000000000000000000000000000000000000000000000000000009F
:200160000000000000000000000000000000000000000000000000000000000000000007F
:200180000000000000000000000000000000000000000000000000000000000000000005F
:2001A0000000000000000000000000000000000000000000000000000000000000000003F
:2001C0000000000000000000000000000000000000000000000000000000000000000001F
:2001E0000000000000000000000000000000000000000000000000000000000000000FF
:00000001FF
```

= EEPROM Base Header
= Device 0 Address Map Header, Dies 0-3
= Device 1 Address Map Header, Dies 0-3
= Device 2 Address Map Header, Dies 0-3
= Device 3 Address Map Header, Dies 0-3
= Device 0 Channel + Share Register Data, Die 0
= Device 0 Channel + Share Register Data, Die 1
= Device 1 Channel + Share Register Data, Die 0
= Device 1 Channel + Share Register Data, Die 1
= Device 2 Channel + Share Register Data, Die 0
= Device 2 Channel + Share Register Data, Die 1
= Device 3 Channel + Share Register Data, Die 0
= Device 3 Channel + Share Register Data, Die 1

In the DS280BR810 hex file, the Base Header Bytes are 0xE30010. From Table 2, this means:

- CRC is enabled (Reg 0x00[7] = 1'b).
- Address Map Header is enabled (Reg 0x00[6] = 1'b).
- EEPROM > 256 Bytes (Reg 0x00[5] = 1'b).
- Common Channel disabled (Reg 0x00[4] = 0'b)
- DEVICE COUNT = 4 Devices (Reg 0x00[3:0] = 0011'b).
- Max EEPROM Burst size = 16 Bytes (Reg 0x02 = 0x10).

Since the DS280BR810 is a multi-die device, there are four Address Map Headers allocated per device slot, with one Address Map Header for each die. Because EEPROM size > 256 Bytes, the Address Map Header for each die is three Bytes long. Note that since each channel register is unique, Die 0 and Die 1 Address Map Headers have a different data start address and a different CRC value. The starting address location for each device die slot can be determined by the non-CRC Bytes of each die's Address Map Header:

- Device 0, Die 0, 2, 3 [CRC, Start Address] = [0x8F, 0x0033]
- Device 0, Die 1 [CRC, Start Address] = [0x8F, 0x0054]
- Device 1, Die 0, 2, 3 [CRC, Start Address] = [0x6A, 0x0075]
- Device 1, Die 1 [CRC, Start Address] = [0xE4, 0x0096]
- Device 2, Die 0, 2, 3 [CRC, Start Address] = [0x01, 0x00B7]
- Device 2, Die 1 [CRC, Start Address] = [0x42, 0x00D8]
- Device 3, Die 0, 2, 3 [CRC, Start Address] = [0xA7, 0x00F9]
- Device 3, Die 1 [CRC, Start Address] = [0x7D, 0x011A]

By searching for the start address relevant to each device die, the remaining Bytes for that die's EEPROM register settings can be found. For example, in Device 3, Die 1, the Bytes of Device 3, Die 1 configuration data begin at Address 0x011A. Since Common Channel Configuration is disabled, each die slot contains four sets of Channel Register data followed by the Share Register data. The Channel and Share Register EEPROM data for Device 3 in this example is shown in Table 9.

Table 9. Device 3 Slot Settings for Die 0 and Die 1

Die 0, Channel 0-3 Register Data	0x87403A0A6D42000080403A0A6D42000080403A0A6D42000080403A0A6D420000
Die 0, Share Register Data	0x00
Die 1, Channel 4-7 Register Data	0x80403A0A6D42000080403A0A6D42000080403A0A6D42000086403A0A6D420000
Die 1, Share Register Data	0x00

The CRC Byte for each die slot is calculated from the Base Header, non-CRC Bytes of the die slot's Address Map Header, and the Channel Register and Share Register data Bytes. The values used for the CRC calculation in Device 3, Die 0 and Die 1 are shown in the following calculations.

Device 3, Die 0 CRC Calculation

```
E30010F90087403A0A6D42000080403A0A6D42000080403A0A6D42000080403A0A6D42000000
```

CRC-8 calculation of the above data Bytes for Die 0 yields CRC = 0xA7.

Device 3, Die 1 CRC Calculation

```
E300101A0180403A0A6D42000080403A0A6D42000080403A0A6D42000086403A0A6D42000000
```

CRC-8 calculation of the above data Bytes for Die 1 yields CRC = 0x7D.

NOTE: Address Map Headers for Die 2 and Die 3 are still present, though their values are arbitrary and ignored because these dies do not exist in the IC. To output a valid EEPROM hex file, these Address Map Headers and CRC Bytes are populated with the same Address Map Header and CRC Byte values from Die 0.

6 EEPROM Register Maps

The SMBus-to-EEPROM register maps for the DS280BR810 and the DS250DF810 are provided in the following tables.

Table 10. EEPROM Register Map Lookup Table

DEVICE TYPE	CHANNEL REGISTER MAPPING	SHARE REGISTER MAPPING
8-Channel Repeater	Table 11	Table 12
8-Channel Retimer	Table 13	Table 14

To read each table, the blue column represents the EEPROM Address Byte, while the remaining columns to the right show Bits 7:0 for the corresponding EEPROM Byte. The matching SMBus register bit for each EEPROM address bit is shown in green. As an example, in Table 11, EEPROM Address Byte 0x05, Bit 4 maps to DS280BR810 SMBus Reg 0x0C[2], which is 0 by default.

NOTE: Share Register EEPROM data follows the last channel's Channel Register EEPROM data for each device's die slot. The actual number of Channel Registers used per EEPROM die slot depends on the device type (repeater or retimer) and whether Common Channel Configuration is enabled. The EEPROM Register Maps presented in this section assume that Common Channel Configuration has been enabled.

Table 11. Default EEPROM Map for DS280BR810 Channel Register Data

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		0 (0x00)	0x03 [7]	0x03 [6]	0x03 [5]	0x03 [4]	0x03 [3]	0x03 [2]	0x03 [1]	0x03 [0]
Value	0x80		1	0	0	0	0	0	0	0
SMB Register		1 (0x01)	0x04 [5]	0x04 [4]	0x04 [3]	0x04 [1]	0x04 [0]	0x05 [7]	0x05 [6]	0x05 [5]
Value	0x40		0	1	0	0	0	0	0	0
SMB Register		2 (0x02)	0x05 [4]	0x05 [3]	0x05 [2]	0x06 [7]	0x06 [6]	0x06 [5]	0x06 [4]	0x06 [3]
Value	0x38		0	0	1	1	1	0	0	0
SMB Register		3 (0x03)	0x06 [2]	0x06 [1]	0x06 [0]	0x08 [7]	0x08 [6]	0x08 [5]	0x08 [4]	0x08 [3]
Value	0x0A		0	0	0	0	1	0	1	0
SMB Register		4 (0x04)	0x0A [6]	0x0A [5]	0x0A [4]	0x0B [6]	0x0B [4]	0x0B [3]	0x0B [2]	0x0B [1]
Value	0x6D		0	1	1	0	1	1	0	1
SMB Register		5 (0x05)	0x0B [0]	0x0C [6]	0x0C [3]	0x0C [2]	0x0C [1]	0x0C [0]	0x0D [6]	0x0D [3]
Value	0x42		0	1	0	0	0	0	1	0
SMB Register		6 (0x06)	0x0D [2]	0x0D [1]	0x0D [0]	0x0F [5]	0x0F [4]	0x0F [3]	0x0F [2]	0x0F [1]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		7 (0x07)	0x0F [0]	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Value	0x00		0	0	0	0	0	0	0	0

Table 12. Default EEPROM Map for DS280BR810 Share Register Data

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		8 (0x08)	0x05 [2]	0x07 [0]	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Value	0x00		0	0	0	0	0	0	0	0

Table 13. Default EEPROM Map for DS250DF810 Channel Register Data

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		0 (0x00)	0x03 [7]	0x03 [6]	0x03 [5]	0x03 [4]	0x03 [3]	0x03 [2]	0x03 [1]	0x03 [0]
Value	0x00		0	0	0	0	0	0	0	0

Table 13. Default EEPROM Map for DS250DF810 Channel Register Data (continued)

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		1 (0x01)	0x08 [7]	0x08 [6]	0x08 [5]	0x08 [4]	0x08 [3]	0x08 [2]	0x08 [1]	0x08 [0]
Value	0x73		0	1	1	1	0	0	1	1
SMB Register		2 (0x02)	0x09 [7]	0x09 [6]	0x09 [5]	0x09 [4]	0x09 [3]	0x09 [2]	0x09 [1]	0x09 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		3 (0x03)	0x0A [7]	0x0A [6]	0x0A [5]	0x0A [4]	0x0B [7]	0x0B [6]	0x0B [5]	0x0B [4]
Value	0x06		0	0	0	0	0	1	1	0
SMB Register		4 (0x04)	0x0B [3]	0x0B [2]	0x0B [1]	0x0B [0]	0x0D [5]	0x0D [4]	0x0D [3]	0x0D [2]
Value	0x30		0	0	1	1	0	0	0	0
SMB Register		5 (0x05)	0x11 [7]	0x11 [6]	0x11 [5]	0x11 [3]	0x11 [2]	0x11 [1]	0x11 [0]	0x12 [7]
Value	0x21		0	0	1	0	0	0	0	1
SMB Register		6 (0x06)	0x12 [5]	0x12 [4]	0x12 [3]	0x12 [2]	0x12 [1]	0x12 [0]	0x13 [5]	0x13 [4]
Value	0x0F		0	0	0	0	1	1	1	1
SMB Register		7 (0x07)	0x13 [3]	0x13 [2]	0x13 [1]	0x14 [7]	0x14 [6]	0x14 [5]	0x14 [4]	0x14 [3]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		8 (0x08)	0x14 [2]	0x15 [7]	0x15 [4]	0x15 [3]	0x15 [2]	0x15 [1]	0x15 [0]	0x16 [7]
Value	0xA0		1	0	1	0	0	0	0	0
SMB Register		9 (0x09)	0x16 [6]	0x16 [5]	0x16 [4]	0x16 [3]	0x16 [2]	0x16 [1]	0x16 [0]	0x17 [7]
Value	0xF4		1	1	1	1	0	1	0	0
SMB Register		10 (0x0A)	0x17 [6]	0x17 [5]	0x17 [4]	0x17 [3]	0x17 [2]	0x17 [1]	0x17 [0]	0x18 [6]
Value	0x6D		0	1	1	0	1	1	0	1
SMB Register		11 (0x0B)	0x18 [5]	0x18 [4]	0x18 [2]	0x19 [5]	0x19 [4]	0x19 [3]	0x19 [2]	0x19 [1]
Value	0x10		0	0	0	1	0	0	0	0
SMB Register		12 (0x0C)	0x19 [0]	0x1A [7]	0x1A [6]	0x1A [5]	0x1A [4]	0x1A [3]	0x1A [2]	0x1B [1]
Value	0x2D		0	0	1	0	1	1	0	1
SMB Register		13 (0x0D)	0x1B [0]	0x1C [7]	0x1C [6]	0x1C [5]	0x1C [4]	0x1C [3]	0x1C [2]	0x1E [7]
Value	0xC9		1	1	0	0	1	0	0	1
SMB Register		14 (0x0E)	0x1E [6]	0x1E [5]	0x1E [3]	0x1E [2]	0x1E [1]	0x1E [0]	0x1F [4]	0x1F [3]
Value	0xE5		1	1	1	0	0	1	0	1
SMB Register		15 (0x0F)	0x1F [2]	0x1F [1]	0x1F [0]	0x20 [7]	0x20 [6]	0x20 [5]	0x20 [4]	0x20 [3]
Value	0x60		0	1	1	0	0	0	0	0
SMB Register		16 (0x10)	0x20 [2]	0x20 [1]	0x20 [0]	0x21 [7]	0x21 [6]	0x21 [5]	0x21 [4]	0x21 [3]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		17 (0x11)	0x21 [2]	0x21 [1]	0x21 [0]	0x23 [6]	0x2A [7]	0x2A [6]	0x2A [5]	0x2A [4]
Value	0x15		0	0	0	1	0	1	0	1
SMB Register		18 (0x12)	0x2A [3]	0x2A [2]	0x2A [1]	0x2A [0]	0x2B [5]	0x2B [4]	0x2B [3]	0x2B [2]
Value	0xA2		1	0	1	0	0	0	1	0
SMB Register		19 (0x13)	0x2B [1]	0x2B [0]	0x2C [6]	0x2C [5]	0x2C [4]	0x2C [3]	0x2C [2]	0x2C [1]
Value	0xBB		1	0	1	1	1	0	1	1
SMB Register		20 (0x14)	0x2C [0]	0x2D [7]	0x2D [6]	0x2D [5]	0x2D [4]	0x2D [3]	0x2D [2]	0x2D [1]
Value	0x18		0	0	0	1	1	0	0	0
SMB Register		21 (0x15)	0x2D [0]	0x2F [7]	0x2F [6]	0x2F [5]	0x2F [4]	0x2F [3]	0x2F [2]	0x2F [1]
Value	0x2A		0	0	1	0	1	0	1	0
SMB Register		22 (0x16)	0x30 [6]	0x31 [6]	0x31 [5]	0x31 [4]	0x31 [3]	0x31 [1]	0x31 [0]	0x32 [7]
Value	0x20		0	0	1	0	0	0	0	0
SMB Register		23 (0x17)	0x32 [6]	0x32 [5]	0x32 [4]	0x32 [3]	0x32 [2]	0x32 [1]	0x32 [0]	0x33 [7]
Value	0x23		0	0	1	0	0	0	1	1

Table 13. Default EEPROM Map for DS250DF810 Channel Register Data (continued)

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		24 (0x18)	0x33 [6]	0x33 [5]	0x33 [4]	0x33 [3]	0x33 [2]	0x33 [1]	0x33 [0]	0x34 [6]
Value	0x10		0	0	0	1	0	0	0	0
SMB Register		25 (0x19)	0x34 [5]	0x34 [4]	0x34 [3]	0x34 [2]	0x34 [1]	0x34 [0]	0x35 [7]	0x35 [6]
Value	0xFC		1	1	1	1	1	1	0	0
SMB Register		26 (0x1A)	0x35 [4]	0x35 [3]	0x35 [2]	0x35 [1]	0x35 [0]	0x36 [6]	0x36 [5]	0x36 [4]
Value	0x7B		0	1	1	1	1	0	1	1
SMB Register		27 (0x1B)	0x36 [2]	0x39 [6]	0x39 [5]	0x39 [4]	0x39 [3]	0x39 [2]	0x39 [1]	0x39 [0]
Value	0x60		0	1	1	0	0	0	0	0
SMB Register		28 (0x1C)	0x3A [7]	0x3A [6]	0x3A [5]	0x3A [4]	0x3A [3]	0x3A [2]	0x3A [1]	0x3A [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		29 (0x1D)	0x3D [7]	0x3D [6]	0x3D [5]	0x3D [4]	0x3D [3]	0x3D [2]	0x3D [1]	0x3D [0]
Value	0x1A		0	0	0	1	1	0	1	0
SMB Register		30 (0x1E)	0x3E [7]	0x3E [6]	0x3E [5]	0x3E [4]	0x3E [3]	0x3E [2]	0x3E [1]	0x3E [0]
Value	0x40		0	1	0	0	0	0	0	0
SMB Register		31 (0x1F)	0x3F [7]	0x3F [6]	0x3F [5]	0x3F [4]	0x3F [3]	0x3F [2]	0x3F [1]	0x3F [0]
Value	0x40		0	1	0	0	0	0	0	0
SMB Register		32 (0x20)	0x40 [7]	0x40 [6]	0x40 [5]	0x40 [4]	0x40 [3]	0x40 [2]	0x40 [1]	0x40 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		33 (0x21)	0x41 [7]	0x41 [6]	0x41 [5]	0x41 [4]	0x41 [3]	0x41 [2]	0x41 [1]	0x41 [0]
Value	0x40		0	1	0	0	0	0	0	0
SMB Register		34 (0x22)	0x42 [7]	0x42 [6]	0x42 [5]	0x42 [4]	0x42 [3]	0x42 [2]	0x42 [1]	0x42 [0]
Value	0x50		0	1	0	1	0	0	0	0
SMB Register		35 (0x23)	0x43 [7]	0x43 [6]	0x43 [5]	0x43 [4]	0x43 [3]	0x43 [2]	0x43 [1]	0x43 [0]
Value	0x80		1	0	0	0	0	0	0	0
SMB Register		36 (0x24)	0x44 [7]	0x44 [6]	0x44 [5]	0x44 [4]	0x44 [3]	0x44 [2]	0x44 [1]	0x44 [0]
Value	0x90		1	0	0	1	0	0	0	0
SMB Register		37 (0x25)	0x45 [7]	0x45 [6]	0x45 [5]	0x45 [4]	0x45 [3]	0x45 [2]	0x45 [1]	0x45 [0]
Value	0xC0		1	1	0	0	0	0	0	0
SMB Register		38 (0x26)	0x46 [7]	0x46 [6]	0x46 [5]	0x46 [4]	0x46 [3]	0x46 [2]	0x46 [1]	0x46 [0]
Value	0xD0		1	1	0	1	0	0	0	0
SMB Register		39 (0x27)	0x47 [7]	0x47 [6]	0x47 [5]	0x47 [4]	0x47 [3]	0x47 [2]	0x47 [1]	0x47 [0]
Value	0xD1		1	1	0	1	0	0	0	1
SMB Register		40 (0x28)	0x48 [7]	0x48 [6]	0x48 [5]	0x48 [4]	0x48 [3]	0x48 [2]	0x48 [1]	0x48 [0]
Value	0xD5		1	1	0	1	0	1	0	1
SMB Register		41 (0x29)	0x49 [7]	0x49 [6]	0x49 [5]	0x49 [4]	0x49 [3]	0x49 [2]	0x49 [1]	0x49 [0]
Value	0xD8		1	1	0	1	1	0	0	0
SMB Register		42 (0x2A)	0x4A [7]	0x4A [6]	0x4A [5]	0x4A [4]	0x4A [3]	0x4A [2]	0x4A [1]	0x4A [0]
Value	0xEA		1	1	1	0	1	0	1	0
SMB Register		43 (0x2B)	0x4B [7]	0x4B [6]	0x4B [5]	0x4B [4]	0x4B [3]	0x4B [2]	0x4B [1]	0x4B [0]
Value	0xF7		1	1	1	1	0	1	1	1
SMB Register		44 (0x2C)	0x4C [7]	0x4C [6]	0x4C [5]	0x4C [4]	0x4C [3]	0x4C [2]	0x4C [1]	0x4C [0]
Value	0xFD		1	1	1	1	1	1	0	1
SMB Register		45 (0x2D)	0x4D [7]	0x4D [6]	0x4D [5]	0x4D [4]	0x4D [3]	0x4D [2]	0x4D [1]	0x4D [0]
Value	0xEE		1	1	1	0	1	1	1	0
SMB Register		46 (0x2E)	0x4E [7]	0x4E [6]	0x4E [5]	0x4E [4]	0x4E [3]	0x4E [2]	0x4E [1]	0x4E [0]
Value	0xEF		1	1	1	0	1	1	1	1

Table 13. Default EEPROM Map for DS250DF810 Channel Register Data (continued)

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		47 (0x2F)	0x4F [7]	0x4F [6]	0x4F [5]	0x4F [4]	0x4F [3]	0x4F [2]	0x4F [1]	0x4F [0]
Value	0xFF		1	1	1	1	1	1	1	1
SMB Register		48 (0x30)	0x60 [7]	0x60 [6]	0x60 [5]	0x60 [4]	0x60 [3]	0x60 [2]	0x60 [1]	0x60 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		49 (0x31)	0x61 [7]	0x61 [6]	0x61 [5]	0x61 [4]	0x61 [3]	0x61 [2]	0x61 [1]	0x61 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		50 (0x32)	0x62 [7]	0x62 [6]	0x62 [5]	0x62 [4]	0x62 [3]	0x62 [2]	0x62 [1]	0x62 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		51 (0x33)	0x63 [7]	0x63 [6]	0x63 [5]	0x63 [4]	0x63 [3]	0x63 [2]	0x63 [1]	0x63 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		52 (0x34)	0x64 [7]	0x64 [6]	0x64 [5]	0x64 [4]	0x64 [3]	0x64 [2]	0x64 [1]	0x64 [0]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		53 (0x35)	0x67 [7]	0x67 [6]	0x67 [5]	0x69 [3]	0x69 [2]	0x69 [1]	0x69 [0]	0x6A [7]
Value	0x34		0	0	1	1	0	1	0	0
SMB Register		54 (0x36)	0x6A [6]	0x6A [5]	0x6A [4]	0x6A [3]	0x6A [2]	0x6A [1]	0x6A [0]	0x6B [7]
Value	0x42		0	1	0	0	0	0	1	0
SMB Register		55 (0x37)	0x6B [6]	0x6B [5]	0x6B [4]	0x6B [3]	0x6B [2]	0x6B [1]	0x6B [0]	0x6C [7]
Value	0x80		1	0	0	0	0	0	0	0
SMB Register		56 (0x38)	0x6C [6]	0x6C [5]	0x6C [4]	0x6C [3]	0x6C [2]	0x6C [1]	0x6C [0]	0x6D [7]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		57 (0x39)	0x6D [6]	0x6D [5]	0x6D [4]	0x6D [3]	0x6D [2]	0x6D [1]	0x6D [0]	0x6E [7]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		58 (0x3A)	0x6E [6]	0x6F [7]	0x6F [6]	0x6F [5]	0x70 [3]	0x70 [2]	0x70 [1]	0x70 [0]
Value	0x05		0	0	0	0	0	1	0	1
SMB Register		59 (0x3B)	0x76 [7]	0x76 [6]	0x76 [5]	0x76 [4]	0x76 [3]	0x76 [2]	0x76 [1]	0x76 [0]
Value	0x21		0	0	1	0	0	0	0	1
SMB Register		60 (0x3C)	0x77 [6]	0x77 [5]	0x77 [4]	0x77 [3]	0x77 [2]	0x77 [1]	0x79 [1]	0x79 [0]
Value	0x34		0	0	1	1	0	1	0	0
SMB Register		61 (0x3D)	0x7D [7]	0x7D [6]	0x7D [5]	0x7D [4]	0x7D [3]	0x7D [2]	0x7D [1]	0x7D [0]
Value	0x48		0	1	0	0	1	0	0	0
SMB Register		62 (0x3E)	0x7E [7]	0x7E [6]	0x7E [5]	0x7E [4]	0x7E [3]	0x7E [2]	0x7E [1]	0x7E [0]
Value	0x13		0	0	0	1	0	0	1	1
SMB Register		63 (0x3F)	0x7F [5]	0x7F [4]	0x7F [3]	0x7F [2]	0x7F [1]	0x7F [0]	0x8E [0]	0x96 [3]
Value	0xA9		1	0	1	0	1	0	0	1
SMB Register		64 (0x40)	0x96 [2]	0x96 [1]	0x96 [0]	0x98 [5]	0x98 [4]	0x98 [3]	0x98 [2]	0x98 [1]
Value	0x00		0	0	0	0	0	0	0	0
SMB Register		65 (0x41)	0x98 [0]	0x99 [7]	0x99 [6]	0x99 [5]	0x99 [4]	0x99 [3]	0x99 [2]	0x99 [1]
Value	0x1F		0	0	0	1	1	1	1	1
SMB Register		66 (0x42)	0x99 [0]	0x9A [7]	0x9A [6]	0x9A [5]	0x9A [4]	0x9A [3]	0x9A [2]	0x9A [1]
Value	0x9F		1	0	0	1	1	1	1	1
SMB Register		67 (0x43)	0x9A [0]	0x9B [7]	0x9B [6]	0x9B [5]	0x9B [4]	0x9B [3]	0x9B [2]	0x9C [5]
Value	0xF1		1	1	1	1	0	0	0	1
SMB Register		68 (0x44)	0x9C [4]	0x9C [3]	0x9C [2]	0x9C [1]	0x9C [0]	0x9D [3]	0x9D [2]	0x9D [1]
Value	0x22		0	0	1	0	0	0	1	0
SMB Register		69 (0x45)	0x9E [7]	0x9E [6]	0x9E [5]	0x9E [4]	0x9E [3]	0x9E [2]	0xA5 [7]	0xA5 [6]
Value	0x48		0	1	0	0	1	0	0	0

Table 13. Default EEPROM Map for DS250DF810 Channel Register Data (continued)

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		70 (0x46)	0xA5 [5]	0xA6 [6]	0xA6 [5]	0xA6 [4]	0xA6 [3]	0xA6 [2]	0xA6 [1]	0xA6 [0]
Value	0xC3		1	1	0	0	0	0	1	1
SMB Register		71 (0x47)	0xA9 [7]	0xA9 [6]	0xA9 [5]	0xA9 [4]	0xA9 [3]	0xA9 [2]	0xA9 [1]	0xA9 [0]
Value	0x00		0	0	0	0	0	0	0	0

Table 14. Default EEPROM Map for DS250DF810 Share Register Data

EEPROM ADDRESS BYTE			BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SMB Register		72 (0x48)	0x04 [3]	0x05 [2]	0x05 [1]	0x05 [0]	0x0A [7]	0x0A [6]	0x0A [5]	0x0A [4]
Value	0x90		1	0	0	1	0	0	0	0
SMB Register		73 (0x49)	0x0A [3]	0x0A [2]	0x0A [1]	0x0A [0]	0x10 [3]	0x10 [2]	0x10 [1]	0x10 [0]
Value	0x0F		0	0	0	0	1	1	1	1

7 Summary

In this application note, EEPROM programming is explained for the DS250DF810 retimer and DS280BR810 repeater. Since these devices both have multiple register pages and multiple dies per IC package, it is important to understand how to configure the EEPROM data correctly. To program an EEPROM efficiently and accurately, system designers must decide when to use programming features such as Common Channel Configuration, Address Map Headers, and CRC Bytes. With a complete understanding of how to program and interpret EEPROM hex files for 25 to 28-G devices, system designers are better equipped to generate their own customized hex files and increase the efficiency of their final designs.

8 References

- DS280BR810 Data Sheet (SNLS511)
- DS250DF810 Data Sheet (SNLS495)
- DS100DF410EVK, DS110DF410EVK and DS125DF410EVM Evaluation Board Software Installation, Setup, and Operating Guide (SNLU126)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265