EEPROM PROGRAMMING FOR PCIe UARTs

1.0 INTRODUCTION

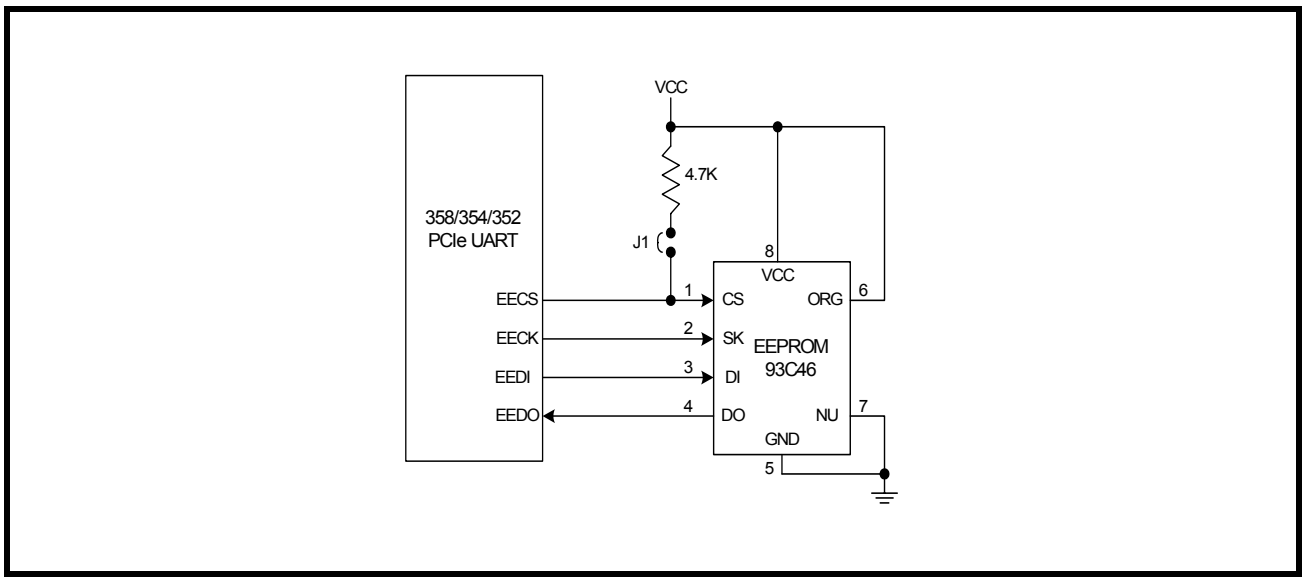
Exar's PCIe family of UARTs provides an interface to an Electrically Erasable Programmable Read Only Memory (EEPROM). The EEPROM is for storing information such as PCI vendor ID, PCI device ID and Class Code etc. This application note describes how to correctly program the EEPROM for the Exar PCIe family of UARTs.

2.0 HARDWARE INTERFACE

The EEPROM interface consists of 4 signals, EEDI, EEDO, EECS, and EECK. This EEPROM is used to store words of information such as Vendor ID, Device ID, Class Code, etc. This information is only used with the plug-and-play auto configuration of the PCI local bus. The EEPROM is not needed when auto configuration is not required in the application. However, if your design requires non-volatile memory for other purpose, it is possible to store and retrieve data on the EEPROM through a special PCI device configuration register.

The Exar PCIe UART works with a 93C46-like EEPROM whose memory is configured as 16-bit words. The 93C46 is a 1K-bit memory device that can be configured as 64 x 16 (sixty four 16-bit words) or 128 x 8 (128 bytes). In order to work with Exar PCIe UARTs, it needs to be configured as 64 x 16. The Figure 1 below shows the reference connections.

FIGURE 1. PCIE UART EEPROM INTERFACE



NOTE: *The 4.7K pull-up resistor is required to load values from the EEPROM at power-up.*

3.0 PROGRAM EEPROM

The EEPROM must be organized into address/data pairs. The first word of the pair is the address. Table 1 below shows the format of the 16-bit address:

TABLE 1: EEPROM ADDRESS BIT DEFINITIONS

BIT(S)	DEFINITION
15	Parity Bit - Odd parity over entire address/data pair If there is a parity error, it will be reported in bit-3 of the REGB register in the Device Configuration Registers (offset 0x08E).
14	Final Address If 1, this will be the last data to be read. If 0, there will be more data to be read after this.
13:8	Reserved - Bits must be '0'
7:0	Target Address -- See Table 2

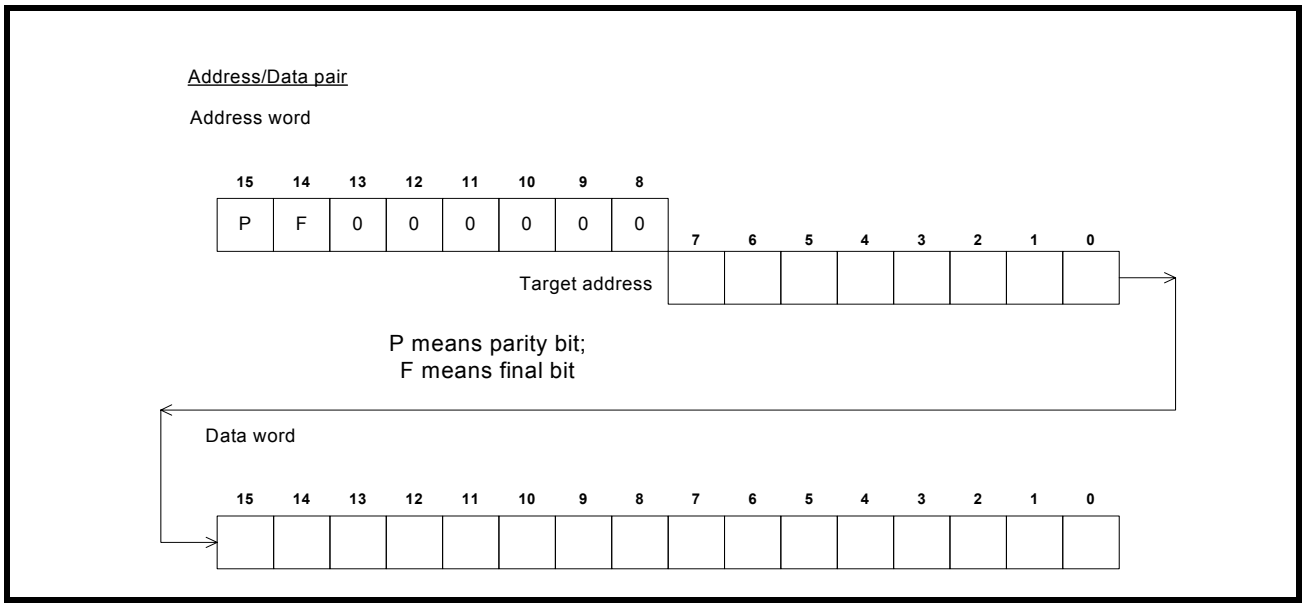
Table 2 shows the Target Addresses available for programming into bits 7:0 of the 16-bit address word. All other Target Addresses are reserved and must not be used.

TABLE 2: TARGET ADDRESS FOR EEPROM VALUES

TARGET ADDRESS	DATA	EXAR DEFAULT
0x00	Vendor ID	0x13A8
0x01	Device ID	0x0358 -- No slave 0x4358 -- XR17V354 slave present 0x8358 -- XR17V358 slave present
0x02	Class code [7:0] lower 8 bits are reserved	0x0200
0x03	Class code [23:8]	0x0700
0x04	Subsystem Vendor ID	0x0000
0x05	Subsystem ID	0x0000

The second word of the pair is the data. The default values are shown in Table 2. Figure 2 shows the format of one address/data pair:

FIGURE 2. ADDRESS / DATA PAIR



There is no requirement in position order in EEPROM for the address/data pair. Only the contents which need to be changed from the Exar defaults need to be included in the EEPROM.

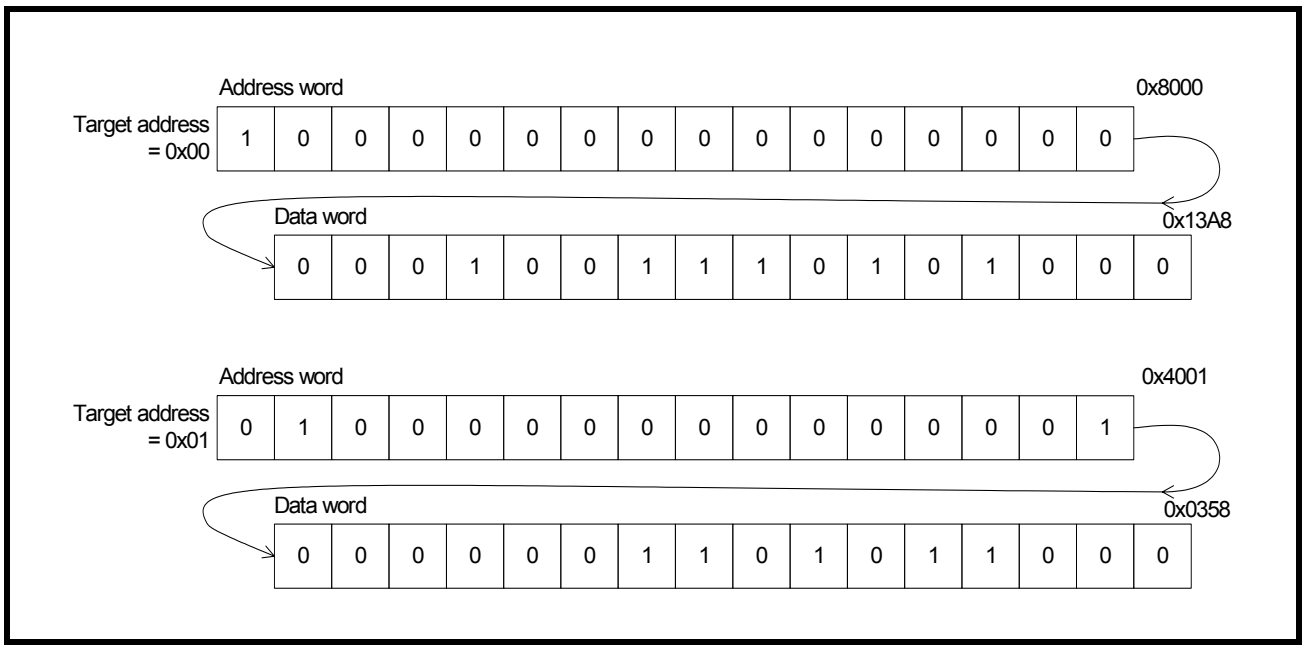
4.0 EXAMPLES

This section will present three examples to show how to program EEPROM. One is for programming default device ID and vendor ID; one is for programming subsystem device ID and vendor ID; the other is for programming vendor ID, device ID and subsystem IDs.

4.1 PROGRAMMING DEVICE ID AND VENDOR ID

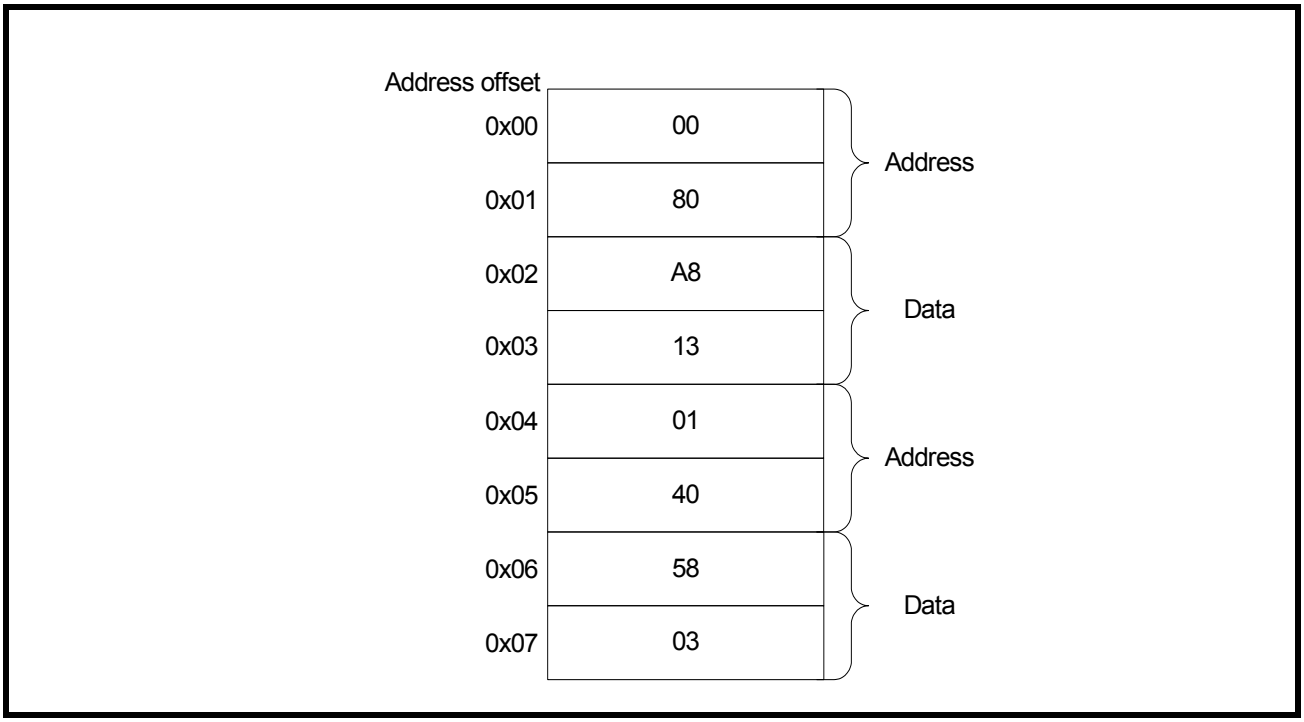
This example program the default Exar vendor ID (0x13A8) and device ID (0x0358).

FIGURE 3. DEFAULT VENDOR ID AND DEVICE ID



Then we can arrange those data and put them into EEPROM:

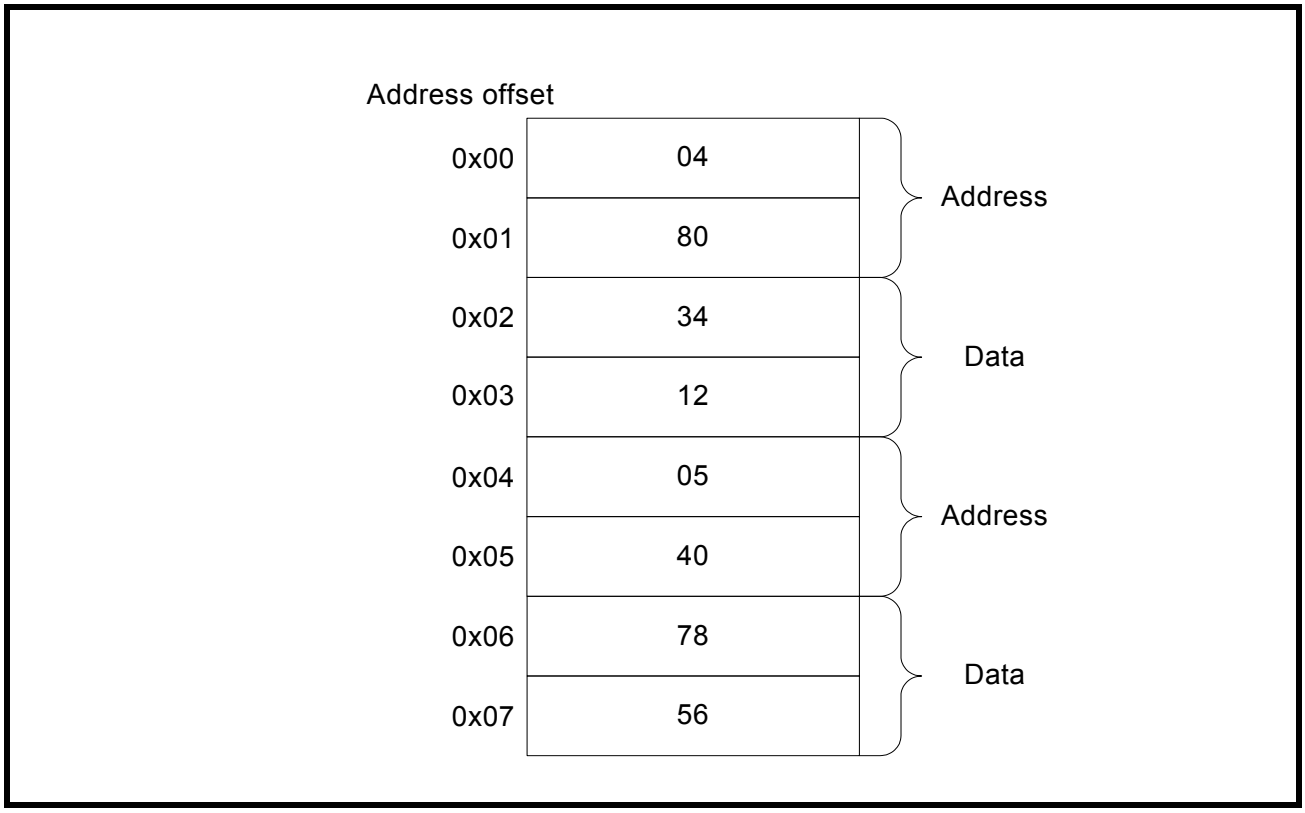
FIGURE 4. EEPROM VALUES IN BYTE ALIGNMENT



4.2 Programming Subsystem Vendor ID and Subsystem Device

In this example, nothing changes except subsystem vendor ID (0x1234) and subsystem device ID (0x5678). Thus, we only need to program subsystem vendor ID and subsystem device ID. **Figure 5** shows the content of subsystem IDs in EEPROM according to the rule in **section 3.0**.

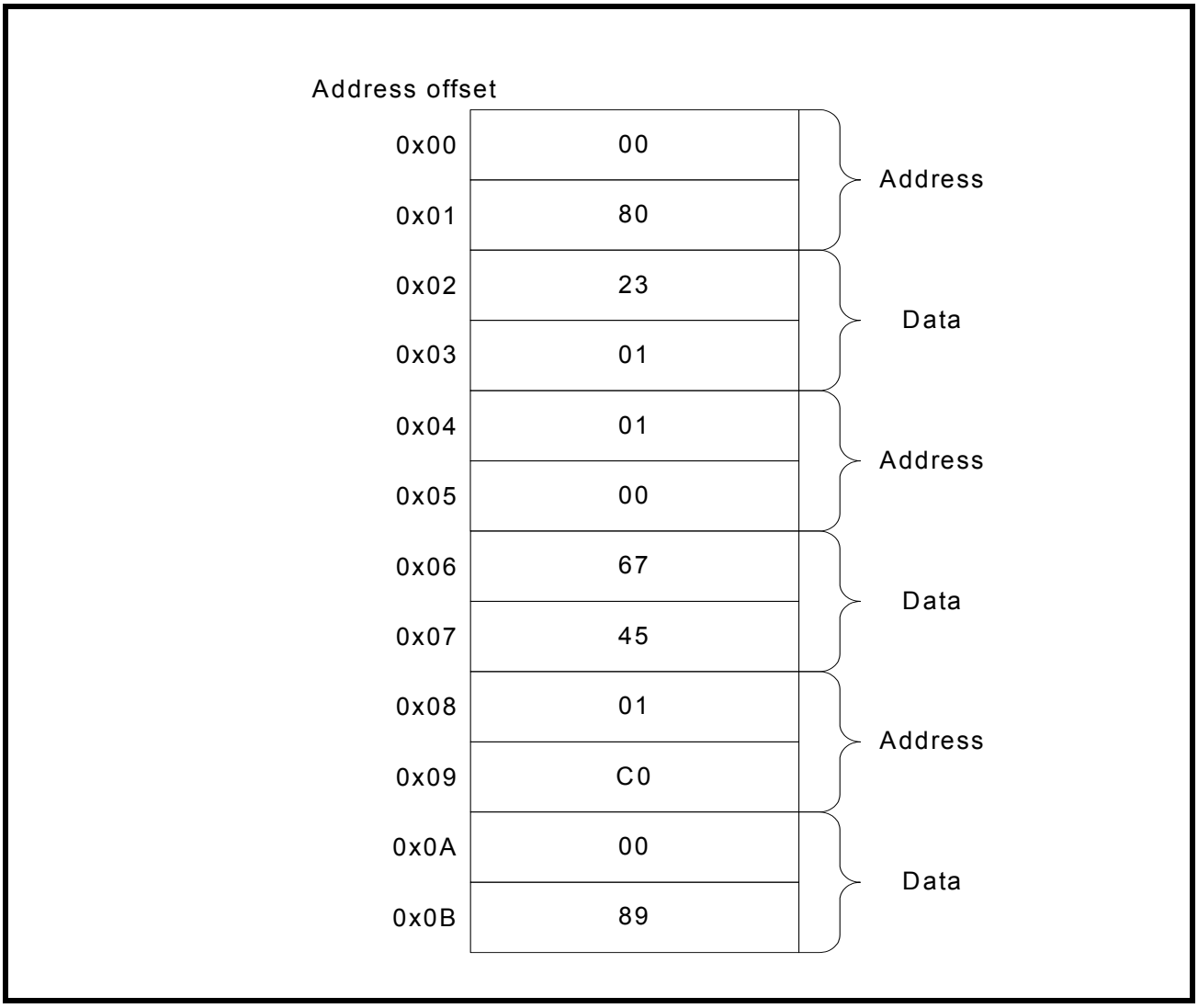
Figure 5. Values of Subsystem IDs in EEPROM



4.3 PROGRAMMING VENDOR ID, DEVICE ID AND SUBSYSTEM VENDOR ID

This example programs vendor ID (to 0x0123), device ID (to 0x4567) and subsystem vendor ID (to 0x8900). Figure 6 show the content of EEPROM according to the rule in section 3.0.

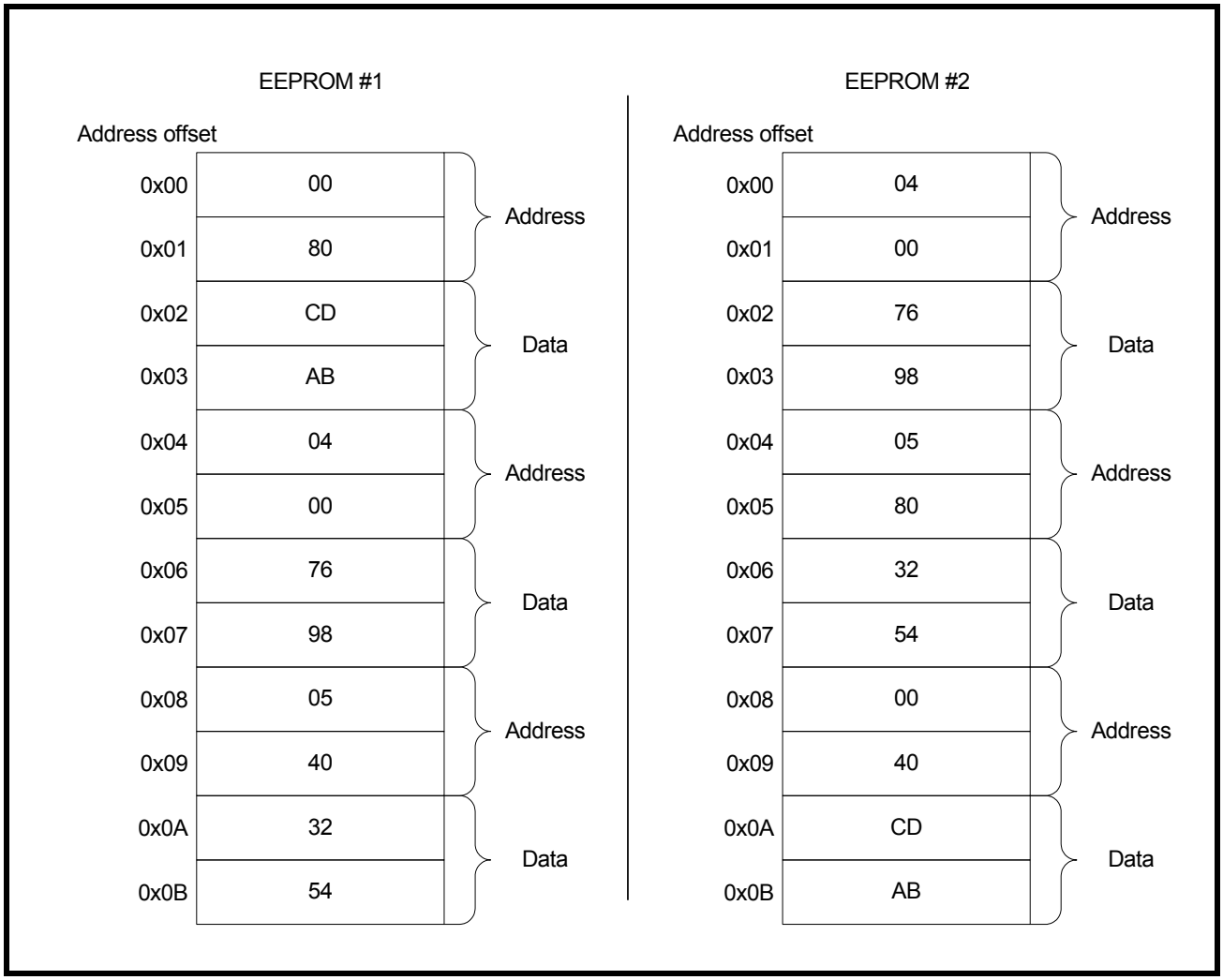
FIGURE 6. VALUES IN EEPROM IN BYTE ALIGNMENT



4.4 PROGRAMMING VENDOR ID, SUBSYSTEM VENDOR ID AND SUBSYSTEM DEVICE ID

This example programs vendor ID (to 0xABCD), subsystem vendor ID (to 0x9876) and subsystem device ID (to 0x5432). Since there is no sequential requirement, the order of the content in EEPROM could be different. Figure 7 shows two possibilities of EEPROM content.

FIGURE 7. PROGRAM VENDOR ID, SUBSYSTEM VENDOR ID AND SUBSYSTEM DEVICE ID



5.0 SUPPORT

For any questions, please contact uarttechsupport@exar.com.

NOTICE

EXAR Corporation reserves the right to make changes to the products contained in this publication in order to improve design, performance or reliability. EXAR Corporation assumes no responsibility for the use of any circuits described herein, conveys no license under any patent or other right, and makes no representation that the circuits are free of patent infringement. Charts and schedules contained here in are only for illustration purposes and may vary depending upon a user's specific application. While the information in this publication has been carefully checked; no responsibility, however, is assumed for inaccuracies.

EXAR Corporation does not recommend the use of any of its products in life support applications where the failure or malfunction of the product can reasonably be expected to cause failure of the life support system or to significantly affect its safety or effectiveness. Products are not authorized for use in such applications unless EXAR Corporation receives, in writing, assurances to its satisfaction that: (a) the risk of injury or damage has been minimized; (b) the user assumes all such risks; (c) potential liability of EXAR Corporation is adequately protected under the circumstances.

Datasheet June 2010.

Send your UART technical inquiry with technical details to hotline: uarttechsupport@exar.com.