User Manual

DA16200 SPI Host Interface

UM-WI-020

Abstract

This user manual describes the communication method between DA16200 SPI Slave and Host Processor

Contents

Figures

Tables

1 Terms and Definitions

SPI	Serial Peripheral Interface

2 References

[1] DA16200, Datasheet, Dialog Semiconductor

[2] DA16200, EVK User Manual, User Manual, Dialog Semiconductor

[3] DA16200, SDK Programmer User Manual, User Manual, Dialog Semiconductor

3 Introduction

This section introduces the subject or problem described in this document. This application note describes how an external processor system, called “External Host” hereafter, communicates with a DA16200 over SPI physical interface protocol. This document also includes the "AT Command Protocol" for use with the External Host.

3.1 PIN MUX Configuration

SPI slave is assigned to GPIOA[3:0] or GPIOA[9:6] in DA16200. Specifically, GPIOA[9:6] is assigned as SPI slave by HW default, and is recommended for use as SPI slave operation.

However, there may be a pin mux initialization code in Dialog’s SDK that may look as follows:

- _fc9k_io_pinmux(PIN_DMUX, DMUX_GPIO);   // For GPIOA 6,7
- _fc9k_io_pinmux(PIN_EMUX, EMUX_GPIO); // For GPIOA 8,9

This means GPIOA[9:6] is to be used as GPIOs, not SPI slave.

Therefore, it needs to change to the following code for SPI slave at GPIOA[9:6]:

- _fc9k_io_pinmux(PIN_DMUX, DMUX_SPIs);   // For GPIOA 6,7
- _fc9k_io_pinmux(PIN_EMUX, EMUX_SPIs); // For GPIOA 8,9

Table 1: Pin MUX Configuration of SPI

GPIO	Signal Name
GPIOA[6]	CS
GPIOA[7]	CLK
GPIOA[8]	MISO
GPIOA[9]	MOSI

4 SPI Protocol

4.1 Message Format

The format of the messages sent/received to/from the external processor is the DA16200 protocol format over SPI physical interface. The format of the message in DA16200 and the parameters included are outlined in Figure 1.

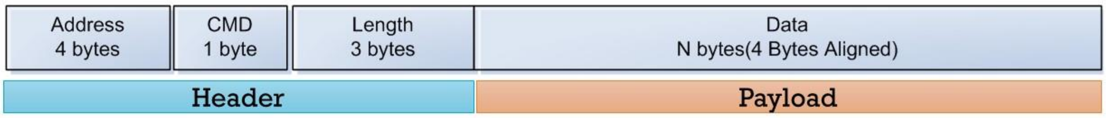


Figure 1 : Basic Format

4.1.1 Address

The address list used by External Host is outlined in Table 2.

Table 2: Address List

Address Type	Address
General Command (Write Request)	0x50080254
AT Command	0x50080260
Response Command	0x50080258
Buffer Address	Received from slave in response message

4.1.2 CMD

The format of CMD field is outlined in Table 3.

Table 3: CMD Format (Command)

Bit Field	Abr.	Description
7	Auto_Inc	1: Internal Address auto-increment, 0: Address Fixed (Not used)
6	Read/Write	1: Read, 0: Write
5:2		Not Used
1:0	CHIP_ID[1:0]	00: CHIP #0 (Default)

4.1.3 Length

Payload Length to follow.

4.2 Write Sequence

Host to Slave write operations are performed in a three SPI transaction. The following sequence takes place

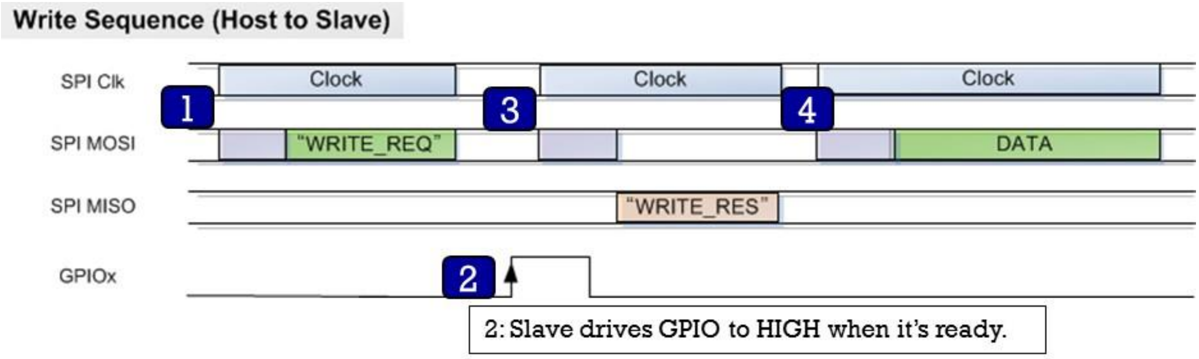

Write Sequence (Host to Slave)
SPI Clk
SPI MOSI
SPI MISO
GPIOx
Clock
Clock
Clock
1
3
4
2
"WRITE_REQ"
DATA
"WRITE_RES"
2: Slave drives GPIO to HIGH when it's ready.

Figure 2 : Write Sequence

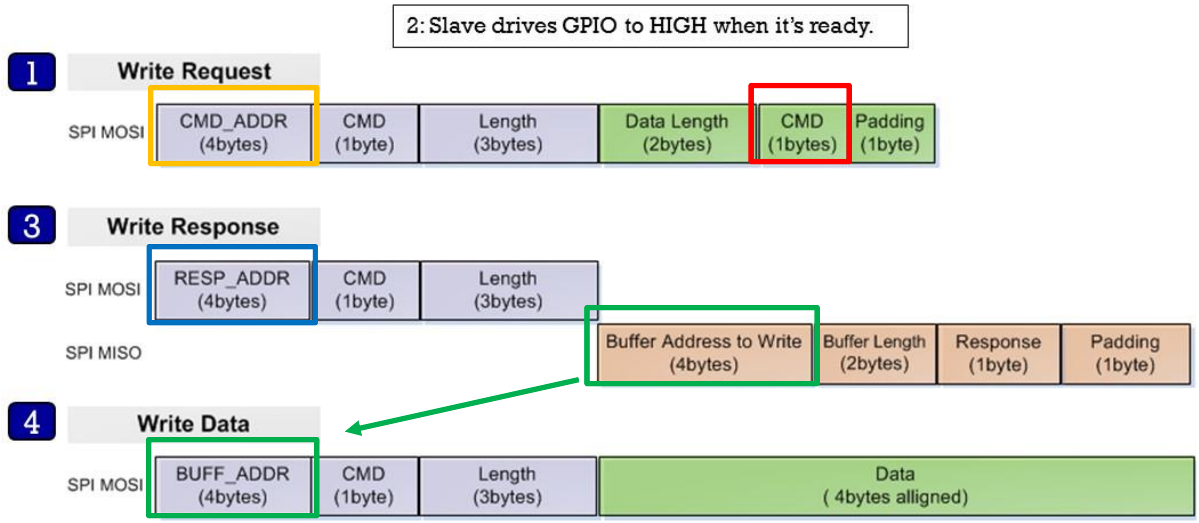


Figure 3: Structure

1. The Host sends a "WRITE_REQ" command (0x80, red rectangle in Figure 3) to the "General Command" address (0x50080254).(yellow rectangle in Figure 3).
2. The Host should waits for GPIO High from slave.
3. The Host reads the Write Response message by "Response Command" address (0x50080258, blue rectangle in Figure 3) and parse it using "`struct _st_host_response`" (see Table 4) .
4. The Host sends data to address (BUFF_ADDR) which is received from the Slave in the Write Response message.(Green rectangle in Figure 3).

Example

When the host wants to write 8 bytes data (0x8877665544332211) to DA16200:

1. Host sends:

(0x50-0x08-0x02-0x54)-(0x80)-(0x00-0x00-0x04)-(0x08-0x00-0x80-0x00)*

*Note that payload data is LSB first. Refer to the figure 8 in this document.

2. Host waits until GPIO is high from DA16200.
3. Host sends

(0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08) then read response from DA16200.

Assume the buffer address from Slave is 0x12345678 for easy description.

Then the read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x81-0x00.

4. Host sends

(0x12-0x34-0x56-0x78)-(0x80)-(0x00-0x00-0x08)-(0x11-0x22-0x33-0x44-0x55-0x66-0x77-0x88)

4.3 Read Sequence & Structure

Figure 4 shows a Slave device transmitting data to the Host when payload is available. This sequence is performed in a *two* SPI transaction.

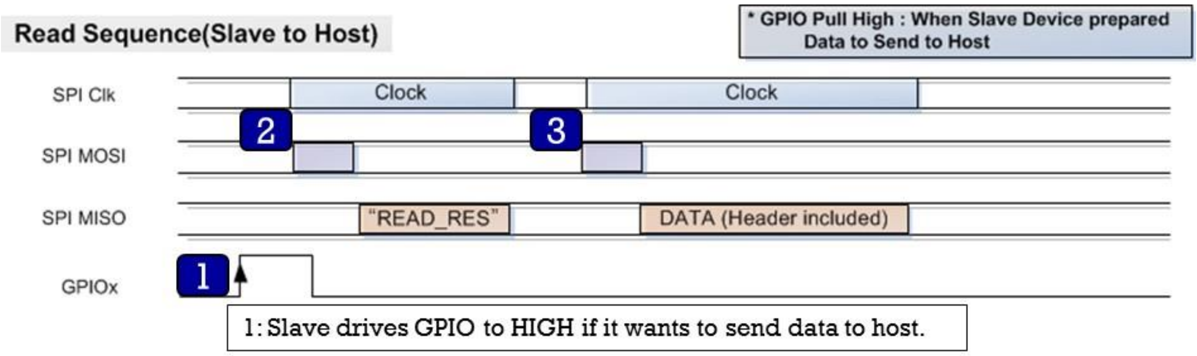

Read Sequence(Slave to Host)
* GPIO Pull High : When Slave Device prepared Data to Send to Host
SPI Clk
Clock
Clock
SPI MOSI
2
3
SPI MISO
"READ_RES"
DATA (Header included)
GPIOx
1
1: Slave drives GPIO to HIGH if it wants to send data to host.

Figure 4: Read Sequence

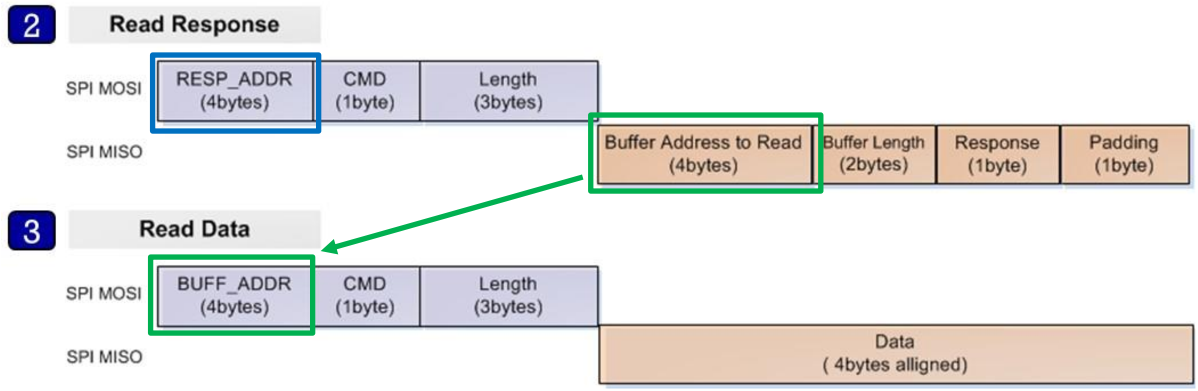


Figure 5: Structure

1. The Slave will toggle *high* the interrupt line to inform the Host when data is available.
2. The Host reads the response message from "Response Command" address (0x50080258, blue rectangle in Figure 5), and parse it using "`struct _st_host_response`". (see Table 4).

3. The Host reads data from address (BUFF_ADDR) which is received from Slave in the response message. (Green rectangle in Figure 5)

Example

1. When the host becomes *high* on GPIO from DA16200,

2. Host sends:

   (0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08) then read response from DA16200.

   Assume the buffer address from Slave is 0x12345678 for easy description and the data length to be sent from DA16200 is 8 bytes.

   The read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x83-0x00.

3. Host sends:

   (0x12-0x34-0x56-0x78)-(0xC0)-(0x00-0x00-0x08) then read data from DA16200.

   Note that read data is LSB first.(see Figure 9)

5 AT Command – Sequences & Structures

AT commands are instructions used to control a modem. AT is the abbreviation of ATtention. Every command line starts with "AT" or "at". ... Note that the starting "AT" is the prefix that informs the modem about the start of a command line. It is not part of the AT command name.

Figure 6 illustrates how to use the AT Command through SPI in "DA16200". This is because AT Command uses a predetermined address and the maximum size of data is defined.

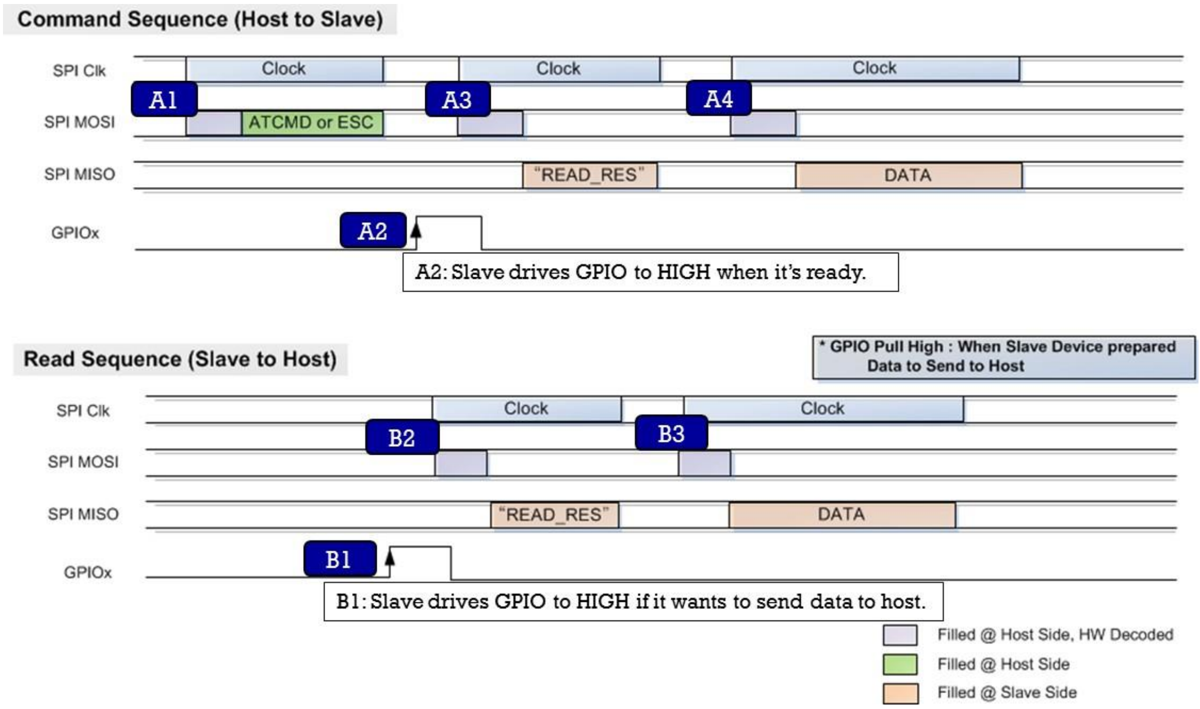

Command Sequence (Host to Slave)
SPI Clk
SPI MOSI
SPI MISO
GPIOx
Clock
Clock
Clock
A1
A3
A4
ATCMD or ESC
"READ_RES"
DATA
A2
A2: Slave drives GPIO to HIGH when it's ready.
Read Sequence (Slave to Host)
* GPIO Pull High : When Slave Device prepared Data to Send to Host
SPI Clk
SPI MOSI
SPI MISO
GPIOx
Clock
Clock
B2
B3
"READ_RES"
DATA
B1
B1: Slave drives GPIO to HIGH if it wants to send data to host.
Filled @ Host Side, HW Decoded
Filled @ Host Side
Filled @ Slave Side

Figure 6: AT command sequence

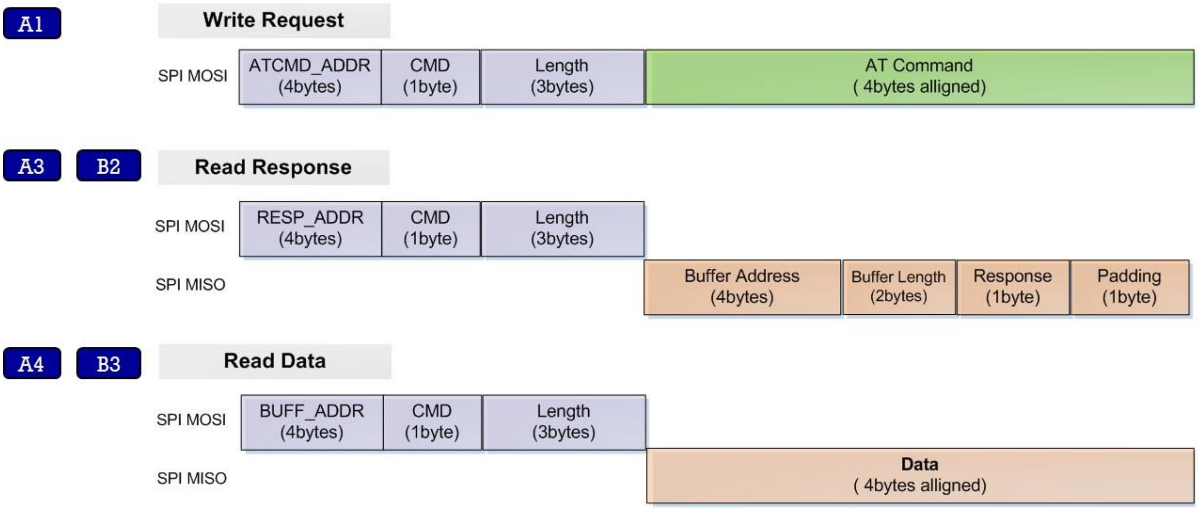


Figure 7: Structure

A1: The Host sends a "AT" or "ESC" command to "AT Command" address.

A2: The Host waits for GPIO interrupt to go *high*

A3: The Host reads the response message from address and parses it using
`"struct _st_host_response"`

A4: The Host reads "OK", "Error" or data from address (BUF_ADDR), depending on the type of command.

B1: The Slave will toggle high the interrupt line to inform Host when data is available

B2: The Host reads the response message from "Response Command" address, and parses it using
`"struct _st_host_response".`

B3: The Host reads data from address (BUF_ADDR) parsed from the response message.

6 Header Format

Write Request (Host to Slave)

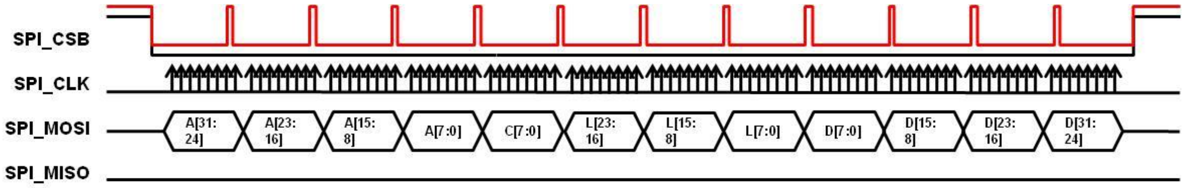
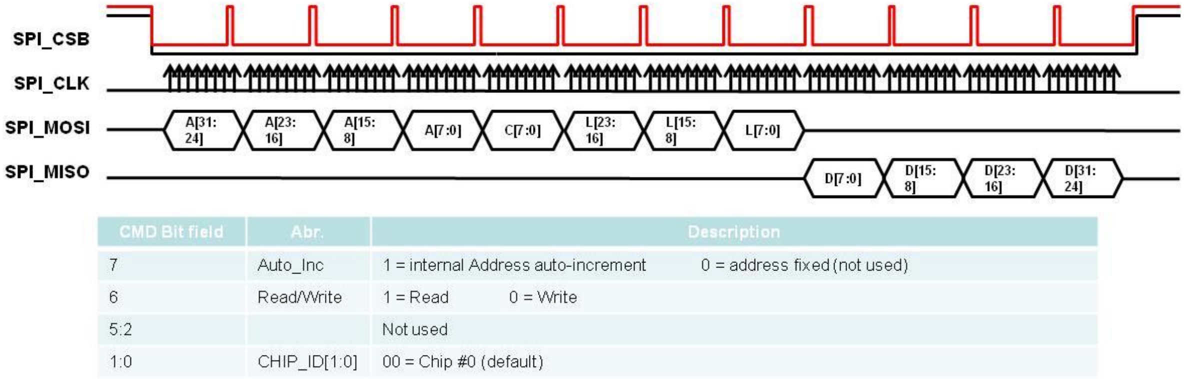


Figure 8: Write Sequence

Read Response (Slave to Host)



CMD Bit field	Abr.	Description
7	Auto_Inc	1 = internal Address auto-increment 0 = address fixed (not used)
6	Read/Write	1 = Read 0 = Write
5:2		Not used
1:0	CHIP_ID[1:0]	00 = Chip #0 (default)

Figure 9: Read Sequence

7 Definition & Structures for Implementation

Table 4 : Definition

```
#define      HOST_MEM_WRITE_REQ         (0x80)
#define      HOST_MEM_WRITE_RES         (0x81)
#define      HOST_MEM_READ_REQ          (0x82)
#define      HOST_MEM_READ_RES          (0x83)
#define      FC9K_GEN_CMD_ADDR          (0x50080254)  // Address to Write Command
#define      FC9K_RESP_ADDR             (0x50080258)  // Address to Read Response
#define      FC9K_ATCMD_ADDR            (0x50080260)  // Address to Send AT Command
```

Table 5: Response Structure

```
typedef struct _st_host_response
{
    u32 buf_address;
    u16 host_length;
    u8  resp;
    u8  dummy;
} st_host_response;
```

Table 6: Request Structure

```
typedef struct _st_host_request
{
    u16 host_write_length;
    u8  host_cmd;
    u8  dummy;
} st_host_request;
```

Revision History

Revision	Date	Description
1.0	7-Apr-2020	Initial Version

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

Disclaimer

Unless otherwise agreed in writing, the Dialog Semiconductor products (and any associated software) referred to in this document are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of a Dialog Semiconductor product (or associated software) can reasonably be expected to result in personal injury, death or severe property or environmental damage. Dialog Semiconductor and its suppliers accept no liability for inclusion and/or use of Dialog Semiconductor products (and any associated software) in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, express or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications. Notwithstanding the foregoing, for any automotive grade version of the device, Dialog Semiconductor reserves the right to change the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications, in accordance with its standard automotive change notification process.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document is subject to Dialog Semiconductor's Standard Terms and Conditions of Sale, available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

Dialog, Dialog Semiconductor and the Dialog logo are trademarks of Dialog Semiconductor Plc or its subsidiaries. All other product or service names and marks are the property of their respective owners.

Contacting Dialog Semiconductor

United Kingdom (Headquarters)
Dialog Semiconductor (UK) LTD
Phone: +44 1793 757700

Germany
Dialog Semiconductor GmbH
Phone: +49 7021 805-0

The Netherlands
Dialog Semiconductor B.V.
Phone: +31 73 640 8822

Email:
enquiry@diasemi.com

North America
Dialog Semiconductor Inc.
Phone: +1 408 845 8500

Japan
Dialog Semiconductor K. K.
Phone: +81 3 5769 5100

Taiwan
Dialog Semiconductor Taiwan
Phone: +886 281 786 222

Web site:
www.dialog-semiconductor.com

Hong Kong
Dialog Semiconductor Hong Kong
Phone: +852 2607 4271

Korea
Dialog Semiconductor Korea
Phone: +82 2 3469 8200

China (Shenzhen)
Dialog Semiconductor China
Phone: +86 755 2981 3669

China (Shanghai)
Dialog Semiconductor China
Phone: +86 21 5424 9058