Application Note

SimpleLink CC33xx Host Interfaces

Shiomi Itzhak

ABSTRACT

The CC33xx family of devices are the next generation of the Simplelink™ embedded solutions. Their main role is to meet the requirements of the new emerging Internet of Things (IoT) use cases while being compatible with the latest cutting-edge technologies like Wi-Fi® 6 and *Bluetooth*® Low Energy 5.3.

These next generation devices enable affordable, reliable and secure connectivity in embedded applications with a host processor running Linux® or an MCU host running RTOS. In order to allow interfacing with various host processors, the host interface design needs to be robust and offer the designer with flexible configuration to choose from.

Table of Contents

List of Figures

List of Tables

Trademarks

1 Introduction

The purpose of this document is to describe the host interface between the host processor and the CC33xx companion IC and provide the system designer with all the required technical information for easy integration.

The most common host processor that connects to the CC33xx companion IC is a high-level processor, typically running high OS such as Linux. In addition, a mid-end or high-end microcontroller can also be used, usually communicating through an SPI interface, providing full scalability and flexibility. Host driver implementation on Linux OS (and on selected MCU) is also delivered as part of the solution, making it easier on the system designer side that does not have to dive into all the technical details.

The following sections describe the main features, the various interface options and the host-device communication flow and protocols.

2 Main Features

Table 2-1 lists the main features supported by the CC33xx companion IC.

Table 2-1. CC33xx Main Features

Feature	Description
Wi-Fi host interface	SDIO or SPI. For more information see Section 2.1.
BLE host interface	• UART • SDIO or SPI on a shared interface For more information, see Section 2.1.
Shared host interface for Wi-Fi and Bluetooth Low Energy	Over SDIO or SPI to save hardware pins. For more information, see Section 2.2.
Automatic host interface detection	Yes
Autonomous mode	Going to sleep and wakeup without informing the host. Implemented over UART, SPI and SDIO. For more information, see Section 2.3.
Secured host interface	Expected in later phase
Out-of-band interrupt	• Separate – one for Wi-Fi and one for Bluetooth Low Energy • Shared – single one for both Wi-Fi and Bluetooth Low Energy For more information, see Section 2.4.
In-band interrupt	In SDIO mode. For more information, see Section 2.4.

2.1 Dual Host Interface

Dual host interface is the most common configuration where each IP uses its own interface. Wi-Fi may either use SDIO or SPI (the same hardware lines are used) and Bluetooth Low Energy uses UART lines. The dual interface mode is the most wasteful in terms of hardware pins. If SDIO interface is used for Wi-Fi, up to 10 lines are used, 6 for SDIO and 4 for UART. If SPI interface is used for Wi-Fi, up to 8 lines are used, 4 for SPI and 4 for UART. The calculation does not consider the extra interrupt lines. The full table with all pin count option is listed in Table 3-10.

Figure 2-1 illustrates the dual host interface setup.

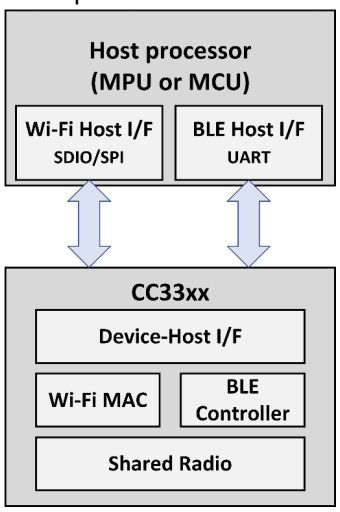

Figure 2-1. CC33xx Dual Interface

2.2 Shared Host Interface

In shared host interface, Wi-Fi and Bluetooth Low Energy share the same interface which may be either SDIO or SPI (the same hardware lines are used). The main advantage of using a shared interface is reduction in hardware pins that may be used for other purposes. If SDIO interface is used, up to six lines are used and if SPI interface is used, four lines are used. The calculation does not consider the extra interrupt lines. The full table with all pin count option is listed in Table 3-10.

Section 2.2 illustrates the dual host interface setup.

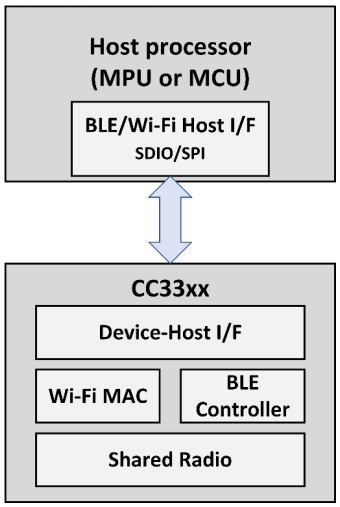

Figure 2-2. CC33xx Shared Interface

2.3 Autonomous Mode

Autonomous mode refers to ability of the device to go to sleep and wakeup without informing the host processor and is supported on all interfaces. Autonomous mode uses an internal buffer that is always on and ready to receive data, even when the device is in sleep. This is mandatory so the device does not loose data. When data is received during sleep, the device makes use of flow control mechanism to stop data from the host processor until the device is fully awake. This mode is seamless to the host processor and the device may look as if it is always awake. Autonomous mode is a real time mechanism and can even be triggered in between non-consecutive frames, making it optimal in terms of power consumption.

2.4 Host Interrupt

2.4.1 Out-of-Band Interrupt

Interrupt line is used by the device to inform the host processor that there is pending event or data. Out-of-band interrupt using a separate line is supported as well and is the recommended option. The reason for recommending on the out-of-band interrupt is mainly to remain compatible with some host processors as there are some host processors that do not wake up on in-band interrupts.

In shared transport mode, a single common hardware line can be used to both Wi-Fi and BLE and thus save one pin. The host processor would then have to read a register to determine the IP source of the setup.

2.4.2 In-Band Interrupt

SDIO in-band interrupt is also supported in case the host processor is short of GPIOs or there are no available ones. In-band interrupt refers to the ability to signal the host without using a dedicated line and is done over one of the data lines. For example, in 4-bit SDIO mode, the second data line DAT[1] is used as an in-band interrupt.

3 Interfaces

3.1 Introduction

The host interface includes three types of protocols, SDIO, SPI and UART, and enables communication with two IPs, Wi-Fi controller and Bluetooth Low Energy controller. The device may connect to a processor host running Linux® or an MCU host running RTOS, as long as all requirements described throughout this paragraph are met.

3.2 SDIO Interface

3.2.1 SDIO Overview

Table 3-1 lists the characteristics of the SDIO host interface.

Table 3-1. SDIO Interface Characteristics

Characteristics	Description
SDIO version	High Speed SDIOv2 and some implementations of SDIOv3
Max frequency	50 MHz
Data bus	4 bits SDIO (1 bit SDIO is also supported)
Max baud rate	200Mbps (4 bits @50MHz)
I/O voltage levels	1.8 V

Table 3-2 lists the signals of the SDIO host interface.

Table 3-2. SDIO Interface Signals

Signal Name	Source	Description
CLK	Controller	Serial bit clock
CMD	Controller	Command line
DAT0-DAT3	Controller/Peripheral	Data lines

3.2.2 SDIO Flow Control

SDIO flow control is part of the specification. SDIO DAT0 data line is used to implement the busy signal to the host controller. Assertion of this line (pulling the pin low) would get interpreted by the host controller as the device is busy and in return the host controller would wait with its write/read operations.

Like in autonomous mode, the flow control is a real time mechanism and can even be triggered in between SDIO blocks, making it optimal in terms of power consumption.

This method is also used for autonomous mode implementation to indicate when the peripheral is ready and when it is not ready.

3.3 SPI Interface

3.3.1 SPI Overview

Table 3-3 lists the characteristics of the SPI host interface.

Table 3-3. SPI Interface Characteristics

Characteristics	Description
Max frequency	25 MHz with optional 50 MHz in non-standard SPI mode(*)
Data bus	1-bit SPI
Max baud rate	26 Mbps (1 bit @25MHz). 52 Mbps for non-standard SPI 1
I/O voltage levels	1.8 V

1. With non-standard SPI, the host processor doubles the supported clock frequency by both, sample and shift out, on rising edge of the clock.

Table 3-4 lists the signals of the SPI host interface.

Table 3-4. SPI Interface Signals

Signal Name	Source	Description
CLK	Controller	Serial bit clock
CS	Controller	Chip select
PICO (Peripheral In Controller Out)	Controller	Serial command and data input
POCI (Peripheral Out Controller In)	Peripheral	Serial busy and data output, high impedance when CS is high and during reset

3.3.2 SPI Configuration

There are four modes of operation defined in the SPI standard. For communication to be successful, the controller and peripheral devices must be configured in the same way. The four modes are all combinations of SPI clock polarity and phase. Only mode 0 is supported, meaning the SPI clock is active high where data is sampled on rising edge of the clock and shifted out on falling edge of the clock.

There are few more SPI parameters that may be configured during interface initialization from the host for better compatibility and flexibility.

- ***Little and big endian***: both are supported. In Big Endian the least significant byte is on the wire first.
- ***16-bits and 32-bits word length***: both are supported.
- ***Bit swizzle***: when enabled, bit swizzle reverses the bit order for each word on the wire. The only exception is during busy signals as described under the Flow Control section.

Table 3-5 illustrates the different option for data on the wire.

Table 3-5. SPI Interface Format Configuration

Format	Data on Wire			
16-Bit Little Endian no swizzle	B1(b15 to b8)	B0(b7 to b0)	B3(b31 to b24)	B2(b23 to b16)
16-Bit Little Endian with swizzle	B0(b0 to b7)	B1(b8 to b15)	B2(b16 to b23)	B3(b24 to b31)
16-Bit Big Endian no swizzle	B0(b7 to b0)	B1(b15 to b8)	B2(b23 to b16)	B3(b31 to b24)
16-Bit Big Endian with swizzle	B1(b8 to b15)	B0(b0 to b7)	B3(b24 to b31)	B2(b16 to b23)
32-Bit Little Endian no swizzle	B3(b31 to b24)	B2(b23 to b16)	B1(b15 to b8)	B0(b7 to b0)
32-bit Little Endian with swizzle	B0(b0 to b7)	B1(b8 to b15)	B2(b16 to b23)	B3(b24 to b31)
32-Bit Big Endian no swizzle	B0(b7 to b0)	B1(b15 to b8)	B2(b23 to b16)	B3(b31 to b24)
32-Bit Big Endian with swizzle	B3(b24 to b3)	B2(b16 to b23)	B1(b8 to b15)	B0(b0 to b7)

Configuring these characteristics is done during initialization of the host interface. Since the hardware lines of the host interface are SDIO lines by default, and since the same lines are used for SPI as well, the initial configuration of the SPI characteristics is applied using an SDIO command, CMD0 (see SPI chapter).

Table 3-6 summarizes all possible configurations.

Table 3-6. SPI Interface Configuration Summary

Characteristics	Supported Configuration
Clock polarity	Data is sampled on the rising edge. Data is shifted out on falling edge for standard SPI and rising edge for non-standard SPI
Clock phase	Clock idles at logical 0
Word size	16 or 32 bits
Host Endianity	Configurable
Bit order	Configurable via the bit swizzle
Chip Select polarity	Active low
Host Interrupt polarity	Configurable
Clock frequency	Up to 26 MHz for standard mode and 52 MHz for non-standard mode
Chip select assertion between words	May go high between words
Autonomous mode	Supported

3.3.3 SPI Flow Control

Unlike SDIO, SPI interface does not include a built-in flow control mechanism. One option is to add a hardware pin to indicate that the slave device is ready or not ready to receive data from the host controller. Another option is to implement a proprietary in-band protocol that uses the SPI data lines in form of dedicated commands or other signaling method.

CC33xx companion IC implements a peripheral to host controller flow control, meaning the peripheral may indicate to the host controller to stop transmission. This is done using the POCI data line by indicating a *busy* signal. This method is also used for autonomous mode implementation to indicate when the peripheral is ready and when it is not ready.

A *busy* signal is reflected by '0' value bit stream. The host controller is expected to ignore the *busy* signal on the POCI line, until there is at least one bit with the value '1' ending a word. The stream of '1' would be referred as *~busy* signal. The word size is well defined and is part of the configuration phase. The word size can be either 16 bits or 32 bits. The first data word will be the first word following the word containing the *~busy* signal. Data endianity is disregarded and does not make a difference.

Figure 3-1 illustrates 16-bit data in Big Endian format (least significant byte is on the wire first) in *write* direction.

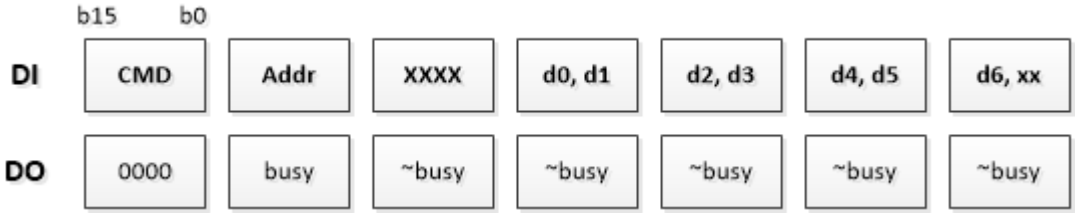

Figure 3-1. 16 Bits, Big Endian, Write Direction

Figure 3-2 illustrates 32-bit data in Big Endian format (most significant byte is on the wire first) in *write* direction.

Figure 3-2. 32 Bits, Big Endian, Write Direction

CC33xx companion IC also implements a peripheral to host controller flow control in read direction, meaning the peripheral may indicate to the host controller to stop reception (or when to start reception). It works similarly to the write direction.

Figure 3-3 illustrates 16-bit data in Big Endian format (i.e. least significant byte is on the wire first) in *read* direction.

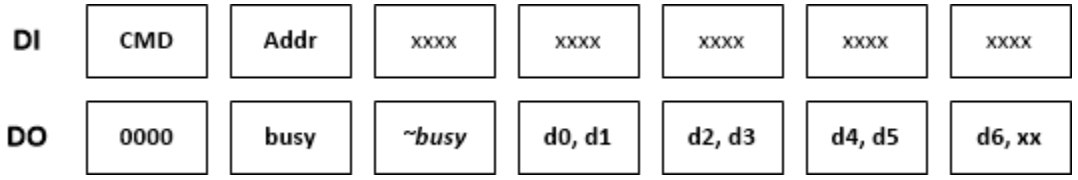

Figure 3-3. 16 Bits, Big Endian, Read Direction

Figure 3-4 illustrates 32-bit data in Little Endian format (i.e. most significant byte is on the wire first) in *read* direction.

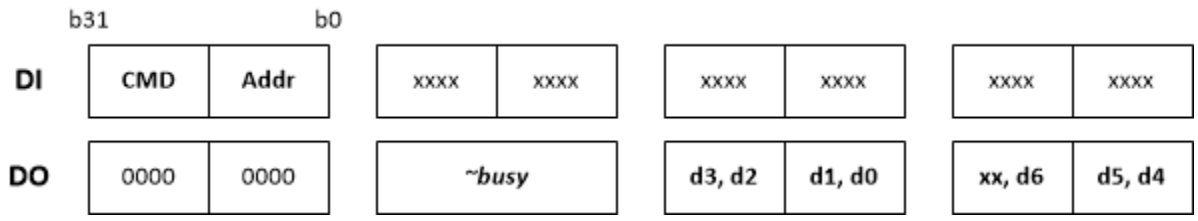

Figure 3-4. 32 Bits, Big Endian, Read Direction

3.4 Uart Interface

3.4.1 UART Overview

Table 3-7 lists the characteristics of the UART host interface.

Table 3-7. UART Interface Characteristics

Characteristics	Description
UART version	4-wire (H4)
Max baud rate	3Mbaud
Flow control	Hardware (RTS/CTS)
Automatic baud rate detection	No
I/O voltage levels	1.8 V

Table 3-8 lists the signals of the UART host interface.

Table 3-8. UART Interface Signals

Signal Name	Source	Description
TxD	Peripheral	Serial out
RxD	Controller	Serial in
RTS (Request to Send)	Peripheral	Request-to-Send to signal the host controller to stop/ resume transmission
CTS (Clear to Send)	Controller	Clear-to-Send to signal the peripheral to stop/resume transmission

3.4.2 UART Configuration

Table 3-9 summarizes all possible configurations.

Table 3-9. UART Interface Configuration

Characteristics	Supported Configuration
Baud rate	9600baud, 115200 baud or 3M baud
Automatic baud rate detection	No
Flow control	Hardware (RTS/CTS)
Data bits	5-8 bits
top bits	1, 1.5 or 2 stop bits
Parity	Even, odd or no parity
Line break	Detection and generation
Autonomous mode	Supported

3.4.3 UART Flow Control

The universal asynchronous receiver-transmitter (UART) is a transport protocol that transfers a byte of data as a stream of individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. The UART is composed of four lines for data transmission (Tx), data reception (Rx), flow control holding the host transmissions (RTS) and flow control holding the device transmissions (CTS).

In UART interface there is no Controller/Peripheral relationship defined by the Hardware and each entity can send data to the other side independently in full duplex mode. The hardware flow control makes use of two hardware lines, RTS (Request to Send) and CTS (Clear to Send) to allow each side to indicate to the other side when it is ready to handle data. These wires are cross-coupled between the two devices, so RTS on one device is connected to CTS on the other device and vice versa. Each device uses its RTS to output if it is ready to accept new data and read CTS to see if it is allowed to send data to the other device.

As long as a device is ready to accept more data, it will keep the RTS line asserted. It will de-assert RTS some time before its receive buffer is full. There might still be data on the line and in the other device transmit registers that has to be received even after RTS has been de-asserted (in this case, the device finishes transmitting the byte and stops the transmission). The other device is required to respect the flow control signal and pause the transmission until RTS is again asserted.

Figure 3-5 illustrates the host and device hardware connectivity.

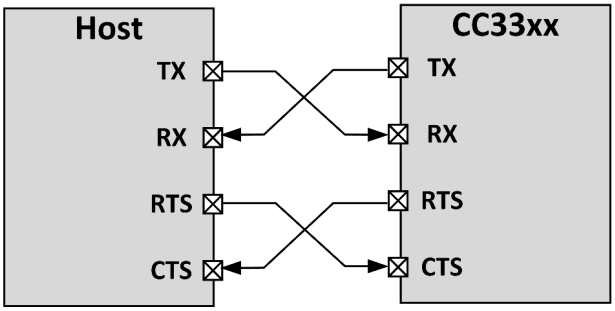

Figure 3-5. UART Interface Flow Control

3.5 Pin Count Options

Table 3-10 summarizes all possible pin count options. Note that 1-bit SDIO mode has not been tested yet.

Table 3-10. Pin Count Options

Configuration	Wi-Fi	BLE	Interrupt	#pins
Dual Interface	4 bits SDIO	UART	OOB	6 (Wi-Fi) + 1 (OOB interrupt) + 4 (BLE) = 11 pins
Dual Interface	1-bit SDIO	UART	OOB	3 (Wi-Fi) + 1 (OOB interrupt) + 4 (BLE) = 8 pins
Shared Interface	4 bits SDIO	N/A	IB	6 (Wi-Fi/BLE) = 6 pins
Dual Interface	SPI	UART	OOB	4 (Wi-Fi) + 1 (OOB interrupt) + 4 (BLE) = 9 pins

4 Host Communication

4.1 Protocol Overview

Regardless of what hardware interface is used, the traffic is carried over a well-defined protocol. Each hardware interface uses a different wrapper that implements the specific hardware protocol but the actual payload is the same. The protocol details are listed in this section for completeness but do not require any porting efforts from the developer/integrator.

The protocol consists of several command options:

- Generic command format. In this command, no *host_rx* is involved

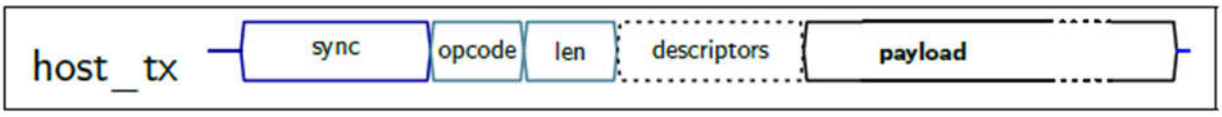

Figure 4-1. Generic Command

- *send()* or *sendto()* command format. The device returns a status response indicating the success of the command

Figure 4-2. Send or Sendto Command

- *recv()* or *recvfrom()* command format, in which the response contains data packet or not

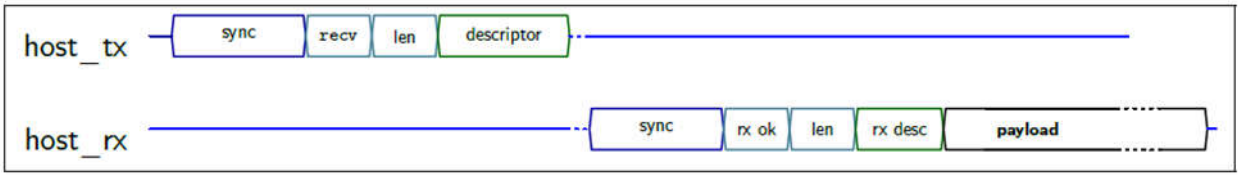

Figure 4-3. Recv or Recvfrom Command

- *read_status* sequence format, where the host requests the device to provide an immediate status

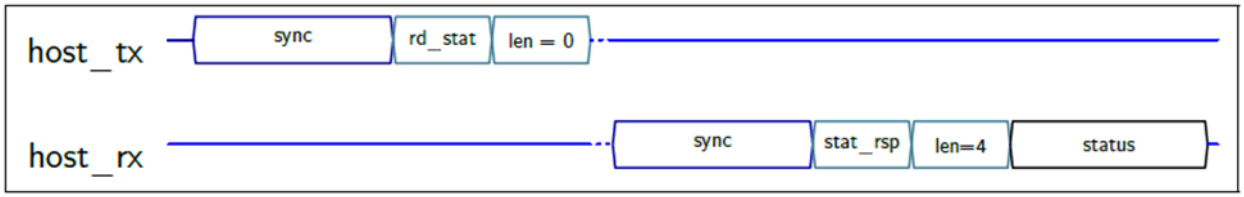

Figure 4-4. Read_status Command

- *cnys* sequence format, where the host requests the device to provide a single, highest priority, packet

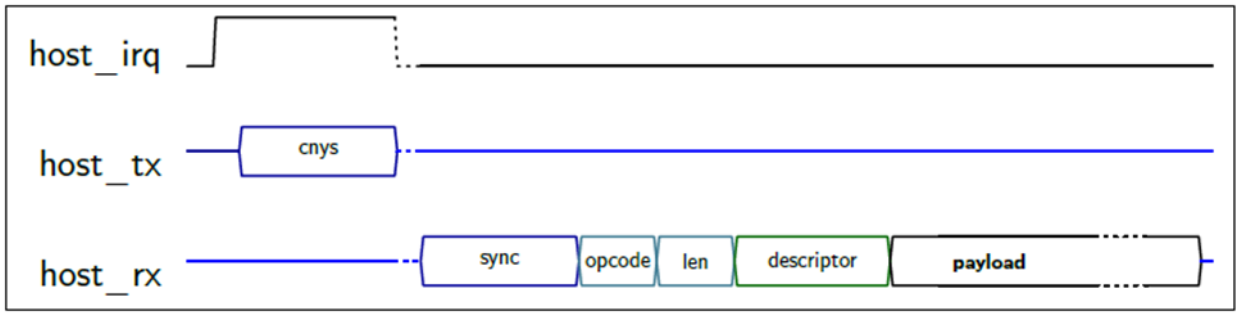

Figure 4-5. Cnys Command

Table 4-1 describes the different command fields.

Table 4-1. Protocol Structure

Field	Description
SYNC	Constant pattern which is used to synchronize the boundary of a command or response
Opcode	Identifies the command or response (depends on direction of the traffic)
Length	Length of everything that follows this field
Descriptors	Depending on the opcode/command, descriptors may follow that provide further information. For example, for a send() operation, the descriptor identifies the socket, flags and length of the actual payload. It also requires a Tx status response field
Payload	Anything else that the command needs
CNYS	Constant pattern which triggers the device to respond with received data or event

4.2 SDIO Wrapper

The SDIO wrapper is a standard SDIO command. For completion of the definitions, CMD53 structure (used in all W-Fi SDIO transactions for data and control) is illustrated in Figure 4-6.

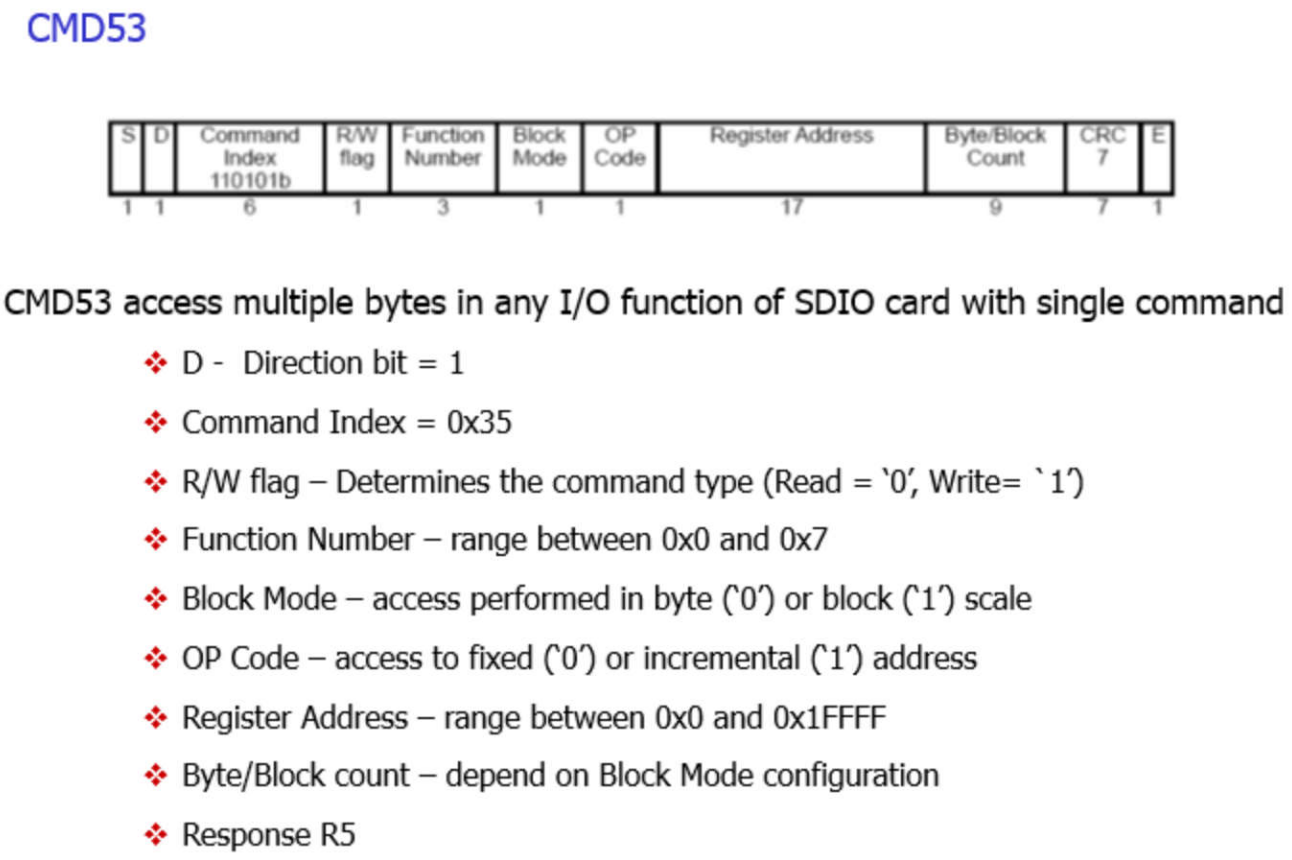

Figure 4-6. SDIO Wrapper

4.3 SPI Wrapper

The SPI wrapper includes two 16-bits words of command and address. These fields are populated by the host driver automatically as illustrated in Figure 4-7.

start	start bit = 0
tx	transmission bit = 1
command	command index, CMD0 = 0x00
reserved	all 0's
ops	Data output polarity select, 1 = Positive-edge driven flop, 0 = Negative-edge driven flop
fbrw	Number of fixed-busy response words, 0 is an invalid entry
fbre	Fixed-busy response enable
iod	Host-Interrupt Open Drain, 1 = Open Drain, 0 = Push-Pull
ip	Host-Interrupt Polarity, 1 = high, 0 = low
cs	unused
ws	data word size, two bytes = 0, four bytes = 1
bs	data bit swizzle, b7/b15/b31 on the wire first = 0, b0 on the wire first = 1
de	data endianess, little endian = 0, big endian, Byte 0 first = 1
wspi	WLAN SPI mode = 1
CRC7	CRC7, can be precalculated, $G(x) = x^7 + x^3 + 1$
end	end bit = 1

Figure 4-7. SPI Wrapper

5 Boot Flow

The host processor may communicate with the companion IC over either SPI or SDIO. The same hardware lines are used for the host interface where the device automatically detect the host interface mode according to a pre-defined sequence. In regards to the host interface, if a dual interface mode is used, the procedure includes a primary host interface (for Wi-Fi) and a secondary host interface (for Bluetooth Low Energy).

Figure 5-1 illustrates the primary interface initialization flow from the host side.

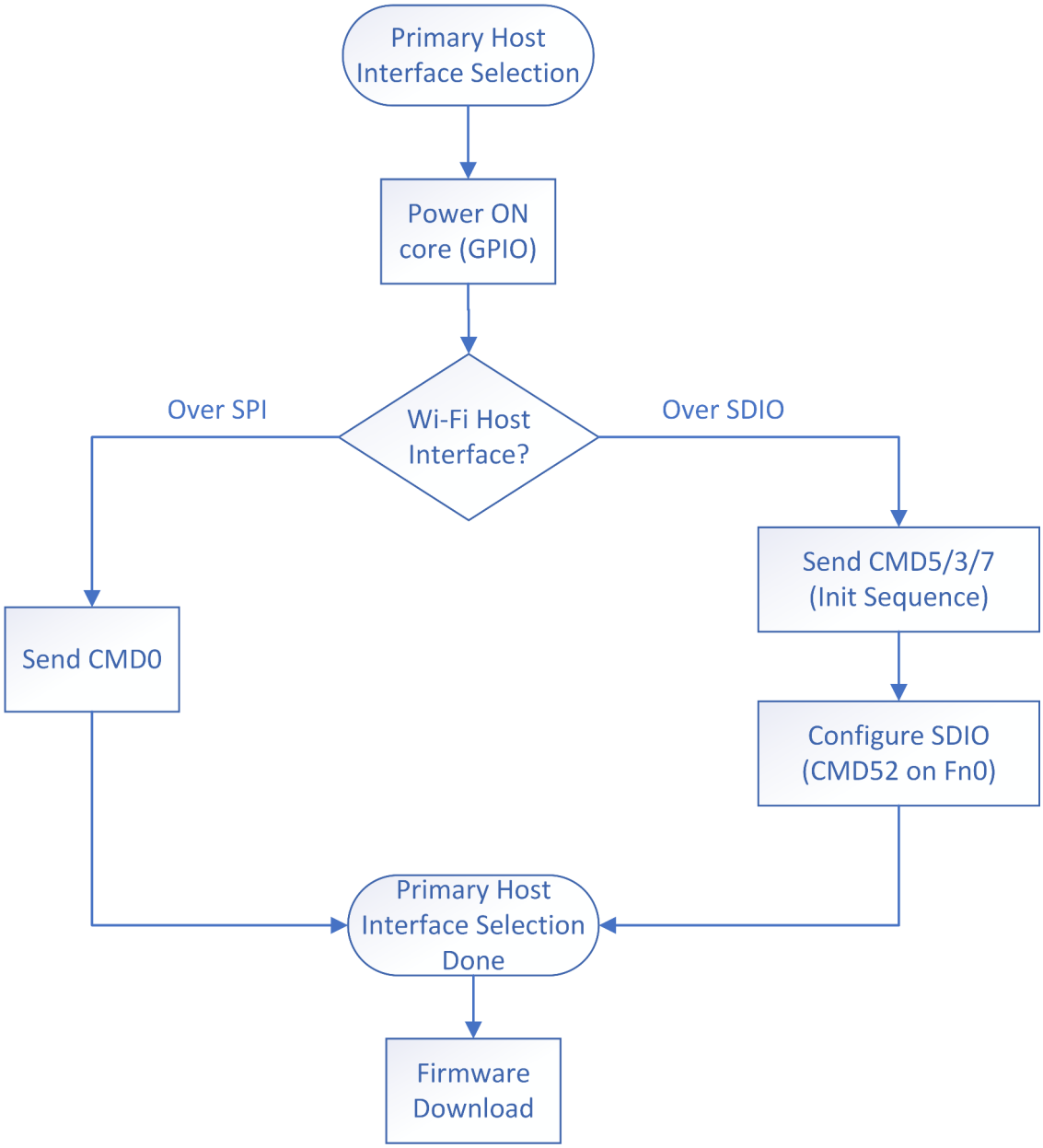

Figure 5-1. Primary Interface Initialization

After the device is powered on, the host is responsible to send the correct sequence of commands. For SPI mode, a single SDIO command with opcode 0 (CMD0) is sent in order to set the device to SPI mode. For SDIO mode, the sequence includes CMD5/3/7 and then a sequence of CMD52 commands to set additional configuration parameters to the device.

If Bluetooth Low Energy is required, a secondary interface initialization flow may be required. The Bluetooth Low Energy may be used in shared mode or in dual mode (over UART). From the device side, if UART is used then until the UART is properly configured the device shall configure the RTS line as manual controlled GPIO output and assert the RTS to further protect the Core from receiving any host commands until the UART is configured and ready. Figure 5-2 illustrates the secondary interface initialization flow from the device side.

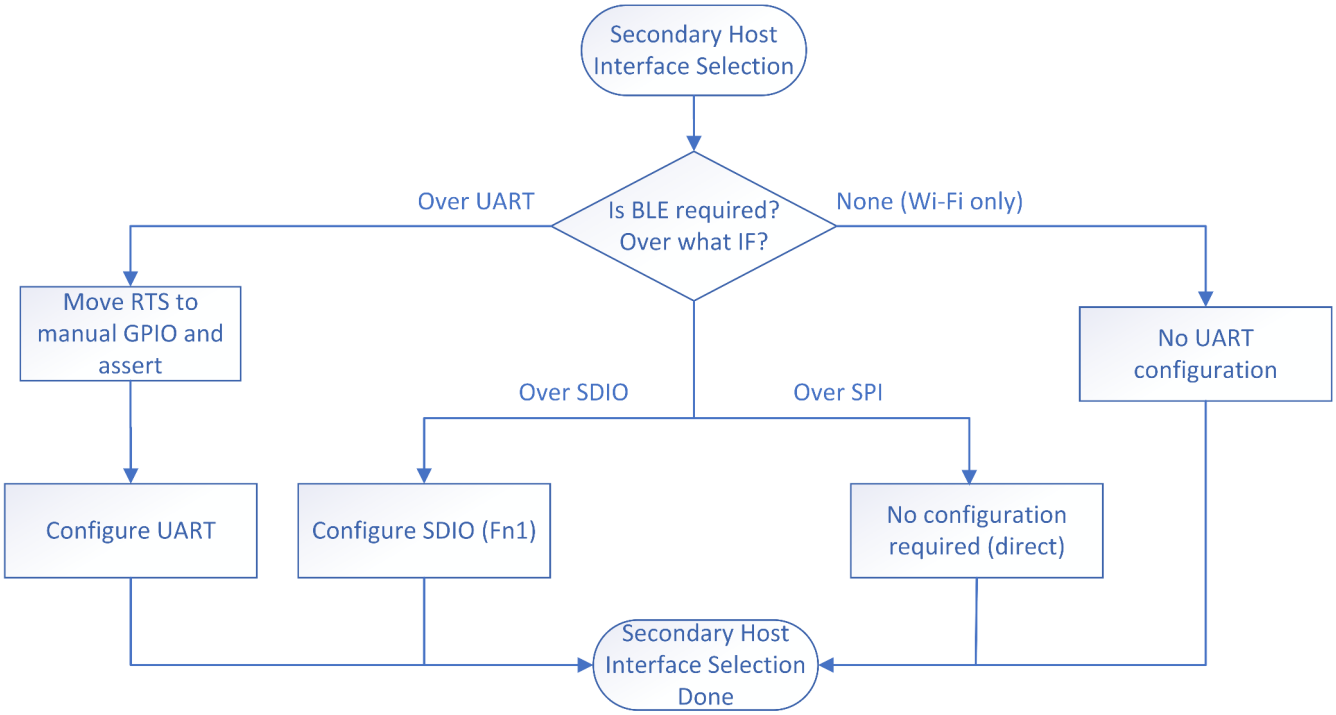

Figure 5-2. Secondary Interface Initialization

5.1 SDIO

During initialization to SDIO mode, the data lines are not involved and only the clock and command lines are used. However, the procedure might pause and fail if SDIO DAT0 line is asserted (set to low). This is interpreted as a busy signal and the host defers its transmission until the line is de-asserted. This should not occur from the device side unless there is an external pull down.

The main commands involved in the initialization procedure are:

- CMD5: this command is used to inquire the voltage range needed by the device
- CMD3: this command is used to assign a logical address to the device
- CMD7: this command is used to select the desired device according to the logical address and enable it for data transfer
- CMD52: this command is used to access common I/O area, function 0. Reads/writes one byte using one command/response pair

5.2 SPI

The SDIO CMD0 command format is illustrated in Figure 5-3.

- This 48-bit command must be presented to the device after power-on or after reset
- For this command the first bit on the wire is always bit 47
- The command consists of 3x16-bit words and is based on the SDIO CMD0

- To put the device in WSPI Mode, bit 8 of the command must be set

b47	b46	b45 b40	b39 b20	b19	b18 b16	b15	b14	b13	b12	b11	b10	b9	b8	b7 b1	b0
start	tx	command	reserved	ops	fbrw	fbre	iod	ip	cs	ws	bs	de	wspi	CRC7	end

start	start bit = 0
tx	transmission bit = 1
command	command index, CMD0 = 0x00
reserved	all 0's
ops	Data output polarity select, 1 = Positive-edge driven flop, 0 = Negative-edge driven flop
fbrw	Number of fixed-busy response words, 0 is an invalid entry
fbre	Fixed-busy response enable
iod	Host-Interrupt Open Drain, 1 = Open Drain, 0 = Push-Pull
ip	Host-Interrupt Polarity, 1 = high, 0 = low
cs	unused
ws	data word size, two bytes = 0, four bytes = 1
bs	data bit swizzle, b7/b15/b31 on the wire first = 0, b0 on the wire first = 1
de	data endianess, little endian = 0, big endian, Byte 0 first = 1
wspi	WLAN SPI mode = 1
CRC7	CRC7, can be precalculated, $G(x) = x^7 + x^3 + 1$
end	end bit = 1

Figure 5-3. SDIO CMD0 Command

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

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