

## AN214847

# CYW4329/CYW4330: Cypress Vendor-Specific Bluetooth Commands

Associated Part Family: CYW4329/CYW4330

This document provides descriptions of the Cypress vendor-specific Bluetooth commands.

## Contents

1	Introduction .....	1	3.6	Download_Minidriver .....	11
1.1	Cypress Part Numbering Scheme .....	1	3.7	Write_RAM .....	11
2	IoT Resources .....	1	3.8	Enable_Radio .....	11
3	Write_BD_ADDR .....	1	3.9	Read_RAM .....	12
3.1	Update_UART_Baud_Rate .....	2	3.10	Launch_RAM .....	12
3.2	Write_SCO_PCM_Int_Param .....	3	3.11	Write_High_Priority_Connection .....	13
3.3	Read_SCO_PCM_Int_Param .....	4	3.12	Write_I2SPCM_Interface_Param .....	13
3.4	Set_Sleepmode_Param .....	6	3.13	Read_Controller_Features .....	14
3.5	Read_Sleepmode_Param .....	9		Document History Page .....	16

## 1 Introduction

This document provides descriptions of the Cypress vendor-specific Bluetooth commands.

### 1.1 Cypress Part Numbering Scheme

Cypress is converting the acquired IoT part numbers from Broadcom to the Cypress part numbering scheme. Due to this conversion, there is no change in form, fit, or function as a result of offering the device with Cypress part number marking. The table provides Cypress ordering part number that matches an existing IoT part number

Table 1. Mapping Table for Part Number between Broadcom and Cypress

Broadcom Part Number	Cypress Part Number
BCM4329	CYW4329
BCM4330	CYW4330

## 2 IoT Resources

Cypress provides a wealth of data at <http://www.cypress.com/internet-things-iot> to help you to select the right IoT device for your design, and quickly and effectively integrate the device into your design. Cypress provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates. Customers can acquire technical documentation and software from the Cypress Support Community website (<http://community.cypress.com/>).

For More details on Bluetooth, please log on to Bluetooth website (<http://www.bluetooth.com/>)

## 3 Write\_BD\_ADDR

OCF 0x001

This command writes the Bluetooth Device Address of the Bluetooth device.

### Command Parameters:

**BD\_ADDR**

Type: Bluetooth Device Address

**Purpose:** The Bluetooth Device Address (for more information, refer to the Read\_BD\_ADDR command in the latest version of Bluetooth Core Specification)

**Size:** 6 bytes

**Return Parameters:**

**Status**

**Type:** uint8

**Purpose:** Error code as specified in Bluetooth Core Specification

**Size:** 1 byte

### 3.1 Update\_UART\_Baud\_Rate

OCF 0x018

This command changes the baud rate at which the UART transport communicates. A Command Complete event will be generated at the previous baud rate. If the Command Complete event indicates success, the UART transport will then shift to the new baud rate.

**Command Parameters:**

**Encoded\_Baud\_Rate**

**Type:** uint16 (little endian)

**Purpose:** This is the Hardware register representation of the new baud rate. The value is determined by the following formula. For the BCM2045, the value of Encoded\_Baud\_Rate can be 0x0000, which implies that the encoded form is not used, and that the actual baud rate follows. When using the UART transport of the BCM2045 in three-wire SLIP (H5) mode, if Update\_UART\_Baud\_Rate is issued, use of the encoded form is disallowed. If the encoded form is used, the value is determined as follows:

```
// Upper four bits
uint32 encoded_segment = ( ( (24000000 / baud_rate) % 16 ) / 2 );
encoded_baud_rate = ( ( (uint16)encoded_segment ) << 4 );

// Lower four bits of high nibble
encoded_segment = ( ( ( (24000000 / baud_rate) % 16 ) / 2 ) +
                    ( ( (24000000 / baud_rate) % 16 ) % 2 ) );
encoded_baud_rate |= (uint16)encoded_segment;

// Lower byte
encoded_segment = ( 256 - ( (24000000 / baud_rate) / 16 ) );
encoded_baud_rate |= (encoded_segment << 8);
```

Note: This formula is based only on a 24 MHz UART clock. For the BCM2048, either a 24 MHz or 48 MHz clock can be selected; this calculation does not apply when the 48 MHz UART clock is used.

**Size:** 2 bytes

**Use\_Encoded\_Form**

**Type:** Boolean

**Purpose:** An abstraction of whether the encoded form is to be used. If not using the encoded form of baud rate, the Encoded\_Baud\_Rate parameter will be 0x0000, and a four-byte baud rate in integer form will follow in the Explicit\_Baud\_rate parameter.

**Size:** 0 bytes (abstract)

**Explicit\_Baud\_Rate**

**Type:** uint32 (big endian)

**Purpose:** If Encoded\_Baud\_Rate is 0x0000, which means that the encoded form of baud rate is not being used, the Explicit\_Baud\_Rate parameter will be present, containing the baud rate in integer form.

**Size:** 4 bytes (if present)

**Baud\_Rate**

**Type:** uint32 (big endian)

Purpose: An abstraction of the baud rate, whether present in the command in encoded form in the Encoded\_Baud\_Rate or included as Explicit\_Baud\_Rate.

Size: 0 bytes (abstract)

#### Return Parameters:

##### Status

Type: uint8 (big endian)

Purpose: Error code as specified in Bluetooth Core Specification

Size: 1 byte

## 3.2 Write\_SCO\_PCM\_Int\_Param

OCF 0x01C

This command writes the SCO and PCM interface parameters.

#### Command Parameters:

##### SCO\_Routing

Type: uint8

Purpose: Specifies whether the SCO path is through PCM interface or transport interface.

Values: (May not be combined bit-wise):

Value	Description
0x0	PCM
0x1	Transport
0x2	Codec
0x3	I <sup>2</sup> S

Size: 1 byte

##### PCM\_Interface\_Rate

##### Size: 1 byte

Type: uint8

Purpose: Specifies the PCM clock frequency.

Values: (May not be combined bit-wise):

Value	Description
0x0	128 KBps
0x1	256 KBps
0x2	512 KBps
0x3	1024 KBps
0x4	2048 KBps

Size: 1 byte

##### Frame\_Type

Type: uint8

Purpose: Specifies the PCM frame type; short frame or long frame.

Values: (May not be combined bit-wise):

Value	Description
0x0	Short
0x1	Long

##### Sync\_Mode

Type: uint8

Purpose: Specifies whether Bluetooth module to be the master or slave for PCM\_SYNC signal

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Slave
0x1	Master

Size: 1 byte

#### Clock\_Mode

Type: uint8

Purpose: Specifies whether Bluetooth module to be the master or slave for PCM\_CLK signal

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Slave
0x1	Master

Size: 1 byte

#### Return Parameters:

##### Status

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

### 3.3 Read\_SCO\_PCM\_Int\_Param

OCF 0x01D

This command reads SCO and PCM interface parameters.

#### Command Parameters:

None

#### Return Parameters:

##### Status

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

##### SCO\_Routing

Type: uint8

Purpose: Indicates whether the SCO path is through PCM interface or transport

Values: (may not be combined bit-wise):

Value	Description
-------	-------------

0x0	PCM
0x1	Transport
0x2	Codec

0x3	I2S
-----	-----

Size: 1 byte

##### PCM\_Interface\_Rate

Type: uint8

Purpose: Indicates the PCM clock frequency.

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	128 KBps
-----	----------

0x1            256 KBps  
 0x2            512 KBps  
 0x3            1024 KBps  
 0x4            2048 KBps

Size:            1 byte

#### Frame\_Type

Type:            uint8  
 Purpose:        Indicates the PCM frame type: short frame or long frame  
 Values:         (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Short
0x1	Long

Size:            1 byte

#### Sync\_Mode

Type:            uint8  
 Purpose:        Indicates whether the Bluetooth module is the master or slave for the PCM\_SYNC signal.  
 Values:         (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Slave
0x1	Master

Size:            1 byte

#### Clock\_Mode

Type:            uint8  
 Purpose:        Indicates whether the Bluetooth module is the master or slave for the PCM\_CLK signal.  
 Values:         (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Slave
0x1	Master

Size:            1 byte

Permission to sleep in Mode 1 is obtained if the BT\_WAKE signal is not asserted. Permission to sleep in Mode 2 occurs after the Sleep Request/Sleep Request ACK exchange. In Modes 3 and 5, if the byte is set to 0, the device will not be able to sleep during the low-power modes. If it is set to 1, the device will be able so sleep during the low-power modes.

#### Enable\_Tristate\_Control\_Of\_UART\_Tx\_Line

Type: Boolean  
 Purpose:        Applicable To Sleep modes 1, 2, 7. When set to 0, the device will not tristate its UART TX line before going to sleep. When set to 1, the device will tristate its UART TX line before going to sleep.  
 Size:            1 byte

#### Active\_Connection\_Handling\_On\_Suspend

Type:            uint8  
 Purpose:        Suspends Behavior; applicable to Sleep modes 3, 5. When set to 0, this flag indicates that upon detecting a USB suspend, the device should sleep whenever possible. This means that the device will stay up if necessary to maintain active ACL and/or SCO connections, and will wake up to perform any scheduled periodic activities if configured to do so. When set to 1, the device will immediate go to sleep upon detecting a USB SUSPEND and will not wake up until USB RESUME is detected. This will cause all connections (whether ACL or

SCO, parked or not, etc.) to be dropped. All periodic activity will also be suspended. When this flag is set to 1, all other parameters are ignored.  
 Values: (May not be combined bit-wise):

Value	Description
0x0	Maintain connections; sleep when timed activity allows
0x1	Sleep until resume is detected

Size: 1 byte

#### Resume\_Timeout

Type: uint8

Purpose: Applicable to Sleep modes 3, 5. After the device issues a USB RESUME, it will wait this many seconds for the Host to resume USB operations before issuing another USB RESUME. If this value is set to 0, the device will never reissue RESUME and will, instead, wait indefinitely for the host to act on the initial RESUME.

Size: 1 byte

#### Enable\_BREAK\_To\_Host

Type: Boolean

Purpose: Applicable to Sleep Mode 12. If 0 and Sleep Mode 12 are selected, disables setting a break condition to the host, making the sleep mechanism unidirectional.

Size: 1 byte

#### Pulsed\_HOST\_WAKE

Type: Boolean

Purpose: Applicable to Sleep modes 1, 12. After asserting BT\_WAKE (Mode 1) or setting or clearing a BREAK condition (Mode 12), if the host does not wake up, clears the condition and retries.

Size: 1 byte

#### Return Parameters:

##### Status

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

## 3.4 Set\_Sleepmode\_Param

OCF 0x027

This command activates the selected Sleep mode algorithm and specifies the respective timer thresholds.

#### Command Parameters:

##### Sleep\_Mode

Type: uint8

Purpose: Sleep mode algorithm selection

Value: (May not be combined bit-wise):

Value	Description
0x0	No Sleep mode
0x1	UART
0x2	UART with messaging
0x3	USB
0x4	H4IBSS
0x5	USB with Host wake
0x6	SDIO
0x7	UART CS-N

0x8 SPI  
 0x9 H5  
 0xA H4DS  
 0xC UART with BREAK

Size: 1 byte

#### **Idle\_Threshold\_Host**

Type: uint8

Purpose: Host idle threshold, applicable to Sleep modes 1, 2, 5, 7. This is the number of firmware loops executed with no activity before the Host wake line is deasserted. Activity includes HCI traffic, excluding certain Sleep mode commands, and the presence of SCO connections if the "Allow Host Sleep During SCO" flag is not set to 1. Each count of this parameter is roughly equivalent to 300 ms. For example, when this parameter is set to 16 (0x10), the Host wake line will be deasserted after approximately 4.8 seconds of inactivity.

Size: 1 byte

#### **Idle\_Threshold\_HC**

Type: uint8

Purpose: Host controller (HC) idle threshold, applicable to Sleep modes 1, 2, 3, 4, 5, 6, 7, and 9. This is the number of firmware loops executed with no activity before the HC is considered idle. Depending on the mode, the HC may then attempt to sleep. Activity includes HCI traffic, excluding certain Sleep mode commands, and the presence of ACL/SCO connections. Each count of this parameter is roughly equivalent to 300 ms. For example, when this parameter is set to 16 (0x10), the HC will be considered idle after approximately 4.8 seconds of inactivity.

Size: 1 byte

#### **BT\_WAKE\_Active\_Mode**

Type: uint8

Purpose: Applicable To Sleep modes 1, 2, 7. This flag indicates whether the BT\_WAKE line is active low or high. GPIO0 is typically used for BT\_WAKE.

Value: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Active Low
0x1	Active High

Size: 1 byte

#### **HOST\_WAKE\_Active\_Mode**

Type: uint8

Purpose: Applicable to Sleep modes 1, 2, 5, 7. This flag indicates whether the HOST\_WAKE line is active-low or active-high. GPIO3 is typically used for HOST\_WAKE on the BCM2035, and GPIO1 is used on ARM-based chips.

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Active Low
0x1	Active High

Size: 1 byte

#### **Allow\_Host\_Sleep\_During\_SCO**

Type: Boolean

Purpose: Applicable to Sleep modes 1, 2, 3, 5, 7. When this flag is set to 0, the Host is not allowed to sleep while an SCO is active. In Modes 1 and 2, the device will keep the Host wake line asserted while an SCO is active. In Mode 3, the device will immediately issue a USB RESUME if the Host issues a SUSPEND. When this flag is set to 1, the Host can sleep

while an SCO is active. This flag should only be set to 1 if SCO traffic is directed to the PCM interface.

**Size:** 1 byte

**Combine\_Sleep\_Mode\_And\_LPM**

**Type:** Boolean

**Purpose:** Applicable to Sleep modes 1, 2, 3, 5, 7. In Mode 0, always set byte 7 to 0. In all sleep modes, the device always requires permission to sleep between scans/periodic inquiries regardless of the setting of this byte. In Modes 1 and 2, if byte is set, device must have "permission" to sleep during the low-power modes of sniff, hold, and park. If byte is not set, device can sleep without permission during these modes. Permission to sleep in Mode 1 is obtained if the BT\_WAKE signal is not asserted. Permission to sleep in Mode 2 occurs after the Sleep Request/Sleep Request ACK exchange. In Mode 3 and 5, if the byte is set to 0, the device will not be able to sleep during the low-power modes. If it is set to 1, the device will be able so sleep during the low-power modes.

**Size:** 1 byte

**Enable\_Tristate\_Control\_Of\_UART\_Tx\_Line**

**Type:** Boolean

**Purpose:** Applicable to Sleep modes 1, 2, 7.  
When set to 0, the device will not tristate its UART TX line before going to sleep.  
When set to 1, the device will tristate its UART TX line before going to sleep.

**Size:** 1 byte

**Active\_Connection\_Handling\_On\_Suspend**

**Type:** uint8

**Purpose:** Suspend Behavior, applicable to Sleep modes 3, 5. When set to 0, this flag indicates that upon detecting a USB SUSPEND, the device should sleep whenever possible. This means that the device will stay up if necessary to maintain active ACL and/or SCO connections and will wake up to perform any scheduled periodic activities, if configured to do so. When set to 1, the device will immediate go to sleep upon detecting a USB SUSPEND and will not wake up until USB RESUME is detected. This will cause all connections (whether ACL or SCO, parked or not, etc.) to be dropped. All periodic activity will also be suspended. When this flag is set to 1, all other parameters are ignored.

**Values:** (May not be combined bit-wise):

Value	Description
0x0	Maintain connections; sleep when timed activity allows.
0x1	Sleep until resume is detected.
<b>Size:</b>	1 byte

#### **Resume\_Timeout**

**Type:** uint8

**Purpose:** Applicable to Sleep modes 3, 5. After the device issues a USB RESUME, it will wait this many seconds for the Host to resume USB operations before issuing another USB RESUME. If this value is set to 0, the device will never reissue RESUME and will instead wait indefinitely for the Host to act on the initial RESUME.

**Size:** 1 byte

#### **Enable\_BREAK\_To\_Host**

**Type:** Boolean

**Purpose:** Applicable to Sleep mode 12. If 0 and Sleep mode 12 is selected, disables setting a break condition to the Host, making the sleep mechanism unidirectional.

**Size:** 1 byte

#### **Pulsed\_HOST\_WAKE**

**Type:** Boolean



**Purpose:** Applicable to Sleep modes 1, 12. After asserting BT\_WAKE (Mode 1) or setting or clearing a BREAK condition (Mode 12), if the host does not wake up, clears the condition and retries.

**Size:** 1 byte

**Return Parameters:**

**Status**

**Type:** uint8 (Error code as specified in Bluetooth Core Specification)

**Size:** 1 byte

### 3.5 Read\_Sleepmode\_Param

OCF 0x028

This command reads back the sleep-mode-related parameters.

**Command Parameters:**

**None**

**Return Parameters:**

**Status**

**Type:** uint8 (Error code as specified in Bluetooth Core Specification)

**Size:** 1 byte

**Sleep\_Mode**

**Type:** uint8

**Purpose:** See [Set\\_Sleepmode\\_Param](#) on page 6.

**Values:** (May not be combined bit-wise):

Value	Description
0x0	No sleep mode
0x1	UART
0x2	UART with messaging
0x3	USB
0x4	H4IBSS
0x5	USB with host wake
0x6	SDIO
0x7	UART CS-N
0x8	SPI
0x9	H5
0xA	H4DS
0xB	HIDD
0xC	UART with BREAK

**Size:** 1 byte

**Idle\_Threshold\_Host**

**Type:** uint8

**Purpose:** See [Set\\_Sleepmode\\_Param](#) on page 6.

**Size:** 1 byte

**Idle\_Threshold\_HC**

**Type:** uint8

**Purpose:** See [Set\\_Sleepmode\\_Param](#) on page 6.

**Size:** 1 byte

**BT\_WAKE\_Active\_Mode**

**Type:** uint8

**Purpose:** See [Set\\_Sleepmode\\_Param](#) on page 6.

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Active Low
0x1	Active High

Size: 1 byte

#### HOST\_WAKE\_Active\_Mode

Type: uint8

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Active Low
0x1	Active High

Size: 1 byte

#### Allow\_Host\_Sleep\_During\_SCO

Type: Boolean

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Size: 1 byte

#### Combine\_Sleep\_Mode\_And\_LPM

Type: Boolean

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Size: 1 byte

#### Enable\_Tristate\_Control\_Of\_UART\_Tx\_Line

Type: Boolean

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Size: 1 byte

#### Active\_Connection\_Handling\_On\_Suspend

Size: 1 byte

Type: uint8

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Values: (May not be combined bit-wise):

Value	Description
-------	-------------

0x0	Maintain connections; sleep when timed activity allows
0x1	Sleep until resume is detected

#### Resume\_Timeout

Type: uint8

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

Size: 1 byte

#### Enable\_BREAK\_To\_Host

Size: 1 byte

Type: Boolean

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

#### Pulsed\_HOST\_WAKE

Size: 1 byte

Type: Boolean

Purpose: See [Set\\_Sleepmode\\_Param](#) on page 6.

### 3.6 Download\_Minidriver

OCF 0x02E

This command triggers the device to reboot into a state where it is prepared to receive a download of a minidriver.

#### Command Parameters:

**None**

#### Return Parameters:

**Status**

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

### 3.7 Write\_RAM

OCF 0x04C

This command writes data into the ARM 32-bit linear address space or EEPROM. This command is primarily intended for use when the device has received a Download\_Minidriver vendor-specific HCI command, placing it into Download mode. A minidriver would typically be downloaded to RAM to facilitate reading or writing firmware and/or configuration data, and then the minidriver would receive Write\_RAM commands to write to Flash or EEPROM. When receiving configuration data to RAM, a minidriver is unnecessary and Write\_RAM commands containing the configuration data are typically issued immediately after the Download\_Minidriver command.

#### Command Parameters:

**Address**

Type: uint32 (little endian)

Purpose: The address to be written to. Addresses greater than or equal to 0xFF000000 represent a virtual address in an EEPROM, with address 0xFF000000 corresponding to address zero in the EEPROM.

Size: 4 bytes

**Data**

Type: uint8 array

Purpose: The data to write to the target address.

Size: Up to 251 bytes

#### Return Parameters:

**Status**

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

### 3.8 Enable\_Radio

OCF 0x034

This command turns the radio on or off (Airplane mode if off).

#### Command Parameters:

**Enable\_Radio**

Type: Boolean

**1**

**Enable the radio**

**0**

**Disable the radio**

Size: 1 byte

#### Return Parameters:

**Status**

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

### 3.9 Read\_RAM

OCF 0x04D

This command reads data from the ARM 32-bit linear address space or EEPROM. This command is primarily intended for use when the device has received a Download\_Minidriver vendor-specific HCI command, placing it into Download mode. A minidriver would typically have been downloaded to RAM to facilitate reading or writing firmware and/or configuration data, and then the minidriver would receive Read\_RAM commands to read or verify the contents of its Flash or EEPROM.

#### Command Parameters:

##### Address

Type: uint32 (little endian)

Purpose: The address to be read from. Addresses greater than or equal to 0xFF000000 represent a virtual address in an EEPROM, with address 0xFF000000 corresponding to address zero in the EEPROM. Reads from the EEPROM are limited to 32 bytes when in Bluetooth mode (when a Download\_Minidriver vendor-specific HCI command has not been issued to the device).

Size: 4 bytes

##### Length

Type: uint8

Purpose: The length of data to be read.

Size: 1 byte

#### Return Parameters:

##### Status

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

##### Data

Type: uint8 array

Purpose: The data that was read from the requested address.

Size: Up to 251 bytes

### 3.10 Launch\_RAM

OCF 0x04E

The commands Jumps into the target address in Thumb mode, typically a minidriver entry point, or in the case of an 0xFFFFFFFF target address, an implied Bluetooth mode reentry vector with acceptance of runtime RAM configuration data. This command is primarily intended to vector from firmware in Download mode (having received a Download\_Minidriver vendor-specific HCI command) into a minidriver, which has just been received, to reset the device by jumping to the reset vector when the minidriver is no longer needed, or in the case of an 0xFFFFFFFF target address, to transition from Download mode back to Bluetooth mode with acceptance of configuration data which was downloaded by Write\_RAM vendor-specific HCI commands.

#### Command Parameters:

##### Address

Type: uint32 (little endian)

Purpose: The address to be written to. A value of 0xFFFFFFFF indicates that a prior series of Write\_RAM HCI commands contained a run-time RAM configuration data image, and that the firmware should reboot into Bluetooth mode, using that configuration data.

Size: 4 bytes

Return Parameters:

##### Status

Type: uint8 (Error code as specified in Bluetooth Core Specification)

Size: 1 byte

### 3.11 Write\_High\_Priority\_Connection

OCF 0x057

This command marks a connection as high priority with a Tpoll value of 40 slots. Transmit data pending for this connection cannot be interrupted by transmit data for other connections. The command can also be used to restore normal priority to a connection. This command is supported only by Flash-based systems.

**Command Parameters:**
**Connection\_Handle**

Type: uint16 (little endian)  
 Purpose: The connection handle to be marked as normal or high priority  
 Size: 2 bytes

**Priority**

Type: uint8  
 Purpose: Priority setting for the connection handle  
 Values: (May not be combined bit-wise):

Value	Description
0x0	Normal
0x1	High
Size:	1 byte

**Return Parameters:**
**Status**

Type: uint8 (Error code as specified in Bluetooth Core Specification)  
 Size: 1 byte

## 3.12 Write\_I2SPCM\_Interface\_Param

OCF 0x06D

This command configures the I<sup>2</sup>S/PCM interface.

**Command Parameters:**
**I2S\_Enable**

Type: uint8  
 Purpose: Turns on or off the I<sup>2</sup>S/PCM interface.  
 Values: (May not be combined bit-wise):

Value	Description
0x0	Disable
0x1	Enable
Size:	1 byte

**Is\_Master**

Type: uint8  
 Purpose: Select master/slave.  
 Values: (May not be combined bit-wise):

Value	Description
0x0	Slave
0x1	Master
Size:	1 byte

**Sample\_Rate**

Type: uint8  
 Purpose: Sets the sample rate.  
 Values: (May not be combined bit-wise):

Value	Description
0x0	8 kHz
0x1	16 kHz
0x2	4 kHz
Size:	1 byte
<b>Clock_Rate</b>	
Type:	uint8
Purpose:	Sets the I <sup>2</sup> S interface clock rate.
Values:	(May not be combined bit-wise):

Value	Description
0x0	128 kHz
0x1	256 kHz
0x2	512 kHz
0x3	1024 kHz
0x4	2048 kHz
Size:	1 byte

**Return Parameters:**

<b>Status</b>	
Type:	uint8 (Error code as specified in Bluetooth Core Specification)
Size:	1 byte

### 3.13 Read\_Controller\_Features

OCF 0x06E

This command reads the features of the Host controller.

**Command Parameters:**

None

**Return Parameters:**

<b>Status</b>											
Type:	uint8										
Purpose:	Error code as specified in Bluetooth Core Specification										
Size:	1 byte										
<b>Features[0]</b>											
Type:	uint32 (little endian)										
Purpose:	Bits 31:0 of the controller features										
Size:	4 bytes										
Values:	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0x1</td><td>Multi-AV transport bandwidth reducer</td></tr> <tr> <td>0x2</td><td>WBS SBC</td></tr> <tr> <td>0x4</td><td>FW LC-PLC</td></tr> <tr> <td>0x8</td><td>FM SBC internal stack</td></tr> </table>	Value	Description	0x1	Multi-AV transport bandwidth reducer	0x2	WBS SBC	0x4	FW LC-PLC	0x8	FM SBC internal stack
Value	Description										
0x1	Multi-AV transport bandwidth reducer										
0x2	WBS SBC										
0x4	FW LC-PLC										
0x8	FM SBC internal stack										
<b>RESERVED</b>											
Type:	uint8										
Purpose:	Bits 39:32 of the controller features, but none are currently defined.										
Size:	1 byte										

## Document History Page

**Document Title: AN214847 - CYW4329/CYW4330 Cypress Vendor-Specific Bluetooth Commands**

**Document Number: 002-14847**

Rev.	ECN No.	Orig. of Change	Submission Date	Description of Change
**	—	—	07/14/2011	43XX-AN1400-R: Initial release
*A	5464392	UTSV	10/06/2016	Updated in Cypress template

## Worldwide Sales and Design Support

### Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturers' representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

#### Products

ARM® Cortex® Microcontrollers	<a href="http://cypress.com/arm">cypress.com/arm</a>
Automotive	<a href="http://cypress.com/automotive">cypress.com/automotive</a>
Clocks & Buffers	<a href="http://cypress.com/clocks">cypress.com/clocks</a>
Interface	<a href="http://cypress.com/interface">cypress.com/interface</a>
Internet of Things	<a href="http://cypress.com/iot">cypress.com/iot</a>
Lighting & Power Control	<a href="http://cypress.com/powerpsoc">cypress.com/powerpsoc</a>
Memory	<a href="http://cypress.com/memory">cypress.com/memory</a>
PSoC	<a href="http://cypress.com/psoc">cypress.com/psoc</a>
Touch Sensing	<a href="http://cypress.com/touch">cypress.com/touch</a>
USB Controllers	<a href="http://cypress.com/usb">cypress.com/usb</a>
Wireless/RF	<a href="http://cypress.com/wireless">cypress.com/wireless</a>

#### PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#)

#### Cypress Developer Community

[Forums](#) | [WICED IoT Forums](#) | [Projects](#) | [Video](#) | [Blogs](#) | [Training](#) | [Components](#)

#### Technical Support

[cypress.com/support](http://cypress.com/support)



Cypress Semiconductor  
198 Champion Court  
San Jose, CA 95134-1709

Phone : 408-943-2600  
Fax : 408-943-4730  
Website : [www.cypress.com](http://www.cypress.com)

© Cypress Semiconductor Corporation, 2011-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit [cypress.com](http://cypress.com). Other names and brands may be claimed as property of their respective owners.