2.0 USB ENUMERATION PROCESS

The MCP2221A implements the CDC class to support the USB-to-UART protocol converter functionality. Using USB-to-UART (CDC class) adapters with personal computers running the Windows Operating System (OS) requires some consideration because of the way the Windows OS responds to their connection.

When a USB-to-UART (CDC class) adapter is connected to the USB port of the PC, Windows searches for a driver. After a suitable driver is found, the system creates an entry in the registry. The entry stores relevant information about the USB-to-UART adapter, its driver and the associated COM port.

The COM port and its number are legacy-type adapters, which are still supported by Windows OS. Historically, the COM ports in a computer are part of the computer's motherboard and are assigned a different index number. With the advent of USB-to-UART adapters, the Windows OS kept the COM port concept and extended it to support the USB adapters.

Whenever a USB-to-UART adapter is first connected to a PC, the system searches the registry for an entry that is suitable for the connected adapter. If one is not found, the system asks for a suitable driver. If this step is completed, it creates a registry entry, and assigns a COM port number as well. Then, whenever the USB-to-UART adapter is connected to that PC, the system checks the registry entry, loads the specified driver and assigns the given COM port number (as found in the registry entry).

During the enumeration process, the device can specify a serial number. If it does, this number is stored in the registry entry and it is used to assign the same COM port number to the adapter in question, no matter which USB port the adapter is connected to.

USB-to-UART adapters have the option to not present a serial number during USB enumeration. In this case, the operating system would not be able to differentiate between two identical devices if neither is providing its serial number. Each time one of these two devices (with no serial number provided during enumeration) is connected to the same USB port, it will have the same COM port number assigned.

Both functionalities (with or without serial numbers) are very useful for different applications.

When the serial number is provided, an adapter using the MCP2221A solution receives the same COM port number from a Windows machine, no matter which USB port it is connected to.

The case with no serial number is useful for test/validation of products using the MCP2221A. The fact that all the tested boards are not supplying a serial number will force Windows to assign them the same COM port number (but only if connected to the same USB port).

The MCP2221A is factory-set to not use a serial number. Later in the process, if a customer wants the benefits provided by using a serial number, the Configuration Utility from Microchip can be used to enable the MCP2221A to enumerate its serial number as well.

The MCP2221A comes with a uniquely provided serial number to be used during the USB enumeration process; however, this can be changed by the user in the Configuration Utility.

The serial number enumeration enable/disable can be changed, as well, using the Configuration Utility.

All the USB-related settings mentioned above are part of the Device Configuration (Chip settings area) and they reside only in Flash. When the Chip settings area (1^{st} area) is being copied into the SRAM (at power-up), the USB settings are skipped (not copied into SRAM).

3.0 USB HID COMMUNICATION

Except for the USB CDC and UART modules, all the other modules in the MCP2221A use USB HID protocol for communication.

The USB HID protocol uses 64-byte reports.

A typical command exchange starts with a 64-byte packet that is written by the USB host (i.e., the PC). Afterward, the USB host reads the response from the device as a 64-byte packet.

3.1 USB HID Commands/Responses

3.1.1 `Status/Set Parameters`

This command offers many options for this device. It is used to poll for the status of the device. It is also used to establish certain I^2C bus parameters/conditions.

TABLE 3-1: COMMAND STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x10	`Status/Set Parameters` – command code.
1	Don't care	Any value	—
2	Cancel current I^2C/SMBus transfer (sub-command)	0x10	When this value is put in this field, the device will cancel the current I^2C/SMBus transfer and will attempt to free the I^2C bus. This command is very useful since it can cancel a transfer and free the bus. An example would be when trying to communicate with a device using a wrong address. This will cause a time-out to occur. This time-out situation can be read using the "`Status/Set Parameters`" and the cancellation of the I^2C/SMBus transfer can be achieved by this sub-command.
		Any other value	No effect.
3	Set I^2C/SMBus communication speed (sub-command)	0x20	When this value is put in this field, the device will take the next command field and interpret it as the system clock divider that will give the I^2C/SMBus communication clock.
		Any other value	No effect.
4	The I^2C/SMBus system clock divider that will be used to establish the communication speed(1)	—	The value in this field is being taken into consideration only when the Byte Index 3 contains the code for establishing a new communication speed. In all the other cases, this field's value will not matter.
5-63	Don't care	Any value	—

Note 1: To compute the desired value for the clock divider, use the following adjusted formula:
Divider = (12 MHz/$I^2C_{CLOCK\ RATE}$) – 2.

3.1.1.1 Responses

TABLE 3-2: RESPONSE 1 STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x10	`Status/Set Parameters` – command code echo.
1	—	0x00	Command completed successfully.
2	Cancel transfer	0x00	No special operation (i.e., cancel current I^2C/SMBus transfer).
		0x10	The current I^2C/SMBus transfer was marked for cancellation. The actual I^2C/SMBus transfer cancellation and bus release will need some time (a few hundreds of microseconds, depending on the communication speed initially chosen for the canceled transfer).
		0x11	The I^2C engine (inside MCP2221A) was already in Idle mode. The cancellation command had no effect.
3	—	0x00	No set I^2C/SMBus communication speed was issued.
		0x20	The new I^2C/SMBus communication speed is now considered.
		0x21	The I^2C/SMBus communication speed was not set (e.g., I^2C transfer in progress).
4	—	The divider value given at the same index in the command field	Only in the case when the code for establishing a new communication speed is given at Byte Index 3.
		0x00	When the communication speed is not being set.
5-7	Don't care	Any value	—
8	I^2C Communication state	0	If the MCP2221A I^2C Engine is in Idle Mode, else timeout occurred.
9	Lower byte (16-bit value) of the requested I^2C transfer length	—	—
10	Higher byte (16-bit value) of the requested I^2C transfer length	—	—
11	Lower byte (16-bit value) of the already transferred (through I^2C) number of bytes	—	—
12	Higher byte (16-bit value) of the already transferred (through I^2C) number of bytes	—	—
13	Internal I^2C data buffer counter	—	—
14	Current I^2C communication speed divider value	—	—
15	Current I^2C time-out value	—	—
16	Lower byte (16-bit value) of the I^2C address being used	—	—
17	Higher byte (16-bit value) of the I^2C address being used	—	—
18-19	Don't care	Any value	—

TABLE 3-2: RESPONSE 1 STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
20	Bit 7: don't care	—	—
	Bit 6: ACK Status	—	If ACK was received from client value is `0`, else `1`.
	Bit 5-0: don't care	—	—
18-19	Don't care	Any value	—
22	SCL line value – as read from the pin	—	—
23	SDA line value – as read from the pin	—	—
24	Interrupt edge detector state	`0` or `1`	—
25	I^2C read pending value	`0`, `1` or `2`	This field is used by the USB host to know if the MCP2221A still has to read from a client device.
26-45	Don't care	Any value	—
46	MCP2221A Hardware Revision Major ('`A`')	—	—
47	MCP2221A Hardware Revision Minor ('`6`')	—	—
48	MCP2221A Firmware Revision Major ('`1`')	—	—
49	MCP2221A Firmware Revision Minor ('`1`')	—	—
50-55	ADC data (16-bit) values	—	3 x (16-bit) little-endian ADC channel values (CH0 LSB, CH0 MSB, CH1 LSB, CH1 MSB, CH2 LSB, CH2 MSB).
56-63	Don't care	Any value	—

3.1.2 `Read Flash Data`

This command is used to read various important data structures and strings that are stored in Flash memory on the MCP2221A.

TABLE 3-3: COMMAND STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0xB0	`Read Flash Data` – command code.
1	`Read Flash Data` sub-code. The value in this field will instruct the MCP2221A on what Flash data are to be read.	0x00	`Read Chip Settings` – reads the MCP2221A Flash settings.
		0x01	`Read GP Settings` – reads the MCP2221A Flash GP settings
		0x02	`Read USB Manufacturer Descriptor String` – reads the USB Manufacturer String Descriptor used during the USB enumeration.
		0x03	`Read USB Product Descriptor String` – reads the USB Product String Descriptor used during the USB enumeration.
		0x04	`Read USB Serial Number Descriptor String` – reads the USB Serial Number String Descriptor that is used during USB enumeration. This serial number can be changed by the user through a specific USB HID command.
		0x05	`Read Chip Factory Serial Number` – reads the factory set serial number. This serial number cannot be changed.
		Any other value	No meaning. The device will reply with a code for an unsupported command at Byte Index 1 in the Response report.
2-63	Reserved	0x00	—

3.1.2.1 Responses

TABLE 3-4: RESPONSE STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0xB0	`Read Flash Data` – command code.
1	—	0x00	Command completed successfully.
		0x01	Command is not supported.
2	Data structure length or Don’t care	—	—
3-63	Data or Don’t care	—	Depends on the issued sub-command or the returned code at Byte Index 1.

TABLE 3-5: RESPONSE STRUCTURE – `Read Chip Settings` SUB-COMMAND

Byte Index	Function Description	Value	Effect
0	—	0xB0	`Read Flash Data` – command code echo.
1	—	0x00	Command completed successfully.
2	Structure length	—	—
3	Don't care	—	—
4	Bit 7: CDC serial number enumeration enable	`1`	The USB serial number will be used during the USB enumeration of the CDC interface.
		`0`	No serial number descriptor will be presented during the USB enumeration.
	Bits 6-2: Not implemented	—	—
	Bits 1-0: Chip configuration security option	`11-10`	Permanently locked.
		`01`	Password-protected.
		`00`	Unsecured.
5	Bits 7-5	Don't care	—
	Bits 4-0: Clock output divider value	—	If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin.
6	Bits 7-6: DAC reference voltage option	`11`	Reference voltage is 4.096V (only if V_{DD} is above this voltage).
		`10`	Reference voltage is 2.048V.
		`01`	Reference voltage is 1.024V.
		`00`	Reference voltage is off (this is useful for the case in which the DAC uses another reference other than V_{RM} DAC; e.g., V_{DD}).
	Bit 5: DAC reference option	`1`	DAC reference is V_{RM} DAC voltage.
		`0`	DAC reference is V_{DD}.
	Bits 4-0: Power-up DAC value	—	—
7	Bit 7	Don't care	—
	Bit 6: Interrupt detection – negative edge	—	If set, the interrupt detection flag will be set when a negative edge occurs.
	Bit 5: Interrupt detection – positive edge	—	If set, the interrupt detection flag will be set when a positive edge occurs.
	Bits 4-3: ADC reference voltage	`11`	Reference voltage is 4.096V (only if V_{DD} is above this voltage).
		`10`	Reference voltage is 2.048V.
		`01`	Reference voltage is 1.024V.
		`00`	Reference voltage is off (this is useful for the case in which the ADC uses another reference other than V_{RM} ADC; e.g., V_{DD}).
	Bit 2	`1`	DAC reference is V_{DD} DAC voltage.
		`0`	DAC reference is V_{RM}.
	Bit 1	Don't care	—
	Bit 0	Don't care	—
8	Lower byte of the 16-bit USB VID value	—	—
9	Higher byte of the 16-bit USB VID value	—	—

TABLE 3-5: RESPONSE STRUCTURE – `Read Chip Settings` SUB-COMMAND (CONTINUED)

Byte Index	Function Description	Value	Effect
10	Lower byte of the 16-bit USB PID value	—	—
11	Higher byte of the 16-bit USB PID value	—	—
12	USB power attributes[(1)]	—	This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes' value) during the USB enumeration.
13	USB requested number of mA(s)[(1)]	—	The requested mA value during the USB enumeration will represent the value at this index multiplied by two.
14-63	Don't care	—	—

Note 1: Please consult the USB 2.0 specification for details on the correct values for power and attributes.

TABLE 3-6: RESPONSE STRUCTURE – `Read GP Settings` SUB-COMMAND

Byte Index	Function Description	Value	Effect
0	—	0xB0	`Read Flash Data` – command code echo.
1	—	0x00	Command completed successfully.
2	—	—	Structure length.
3	—	Don't care	—
4	**GP0 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP0 is set as an output GPIO, this value will be present at the GP0 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP0 designation	`111-011`	Don't care.
		`010`	Alternate Function 0 (LED_URx).
		`001`	Dedicated function operation (SSPND).
		`000`	GPIO operation.
5	**GP1 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP1 is set as an output GPIO, this value will be present at the GP1 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP1 designation	`111-101`	Don't care.
		`001`	Dedicated function operation (clock output).
		`100`	Alternate Function 2 (interrupt detection).
		`011`	Alternate Function 1 (LED_UTx).
		`010`	Alternate Function 0 (ADC1).
		`000`	GPIO operation.
6	**GP2 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP2 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP1 designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC1).
		`010`	Alternate Function 0 (ADC2).
		`001`	Dedicated function operation (USB).
		`000`	GPIO operation

TABLE 3-6: RESPONSE STRUCTURE – `Read GP Settings` SUB-COMMAND (CONTINUED)

Byte Index	Function Description	Value	Effect
7	**GP3 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP3 is set as an output GPIO, this value will be present at the GP3 pin at Power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP1 designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC2).
		`010`	Alternate Function 0 (ADC3).
		`001`	Dedicated function operation (LED_I2C).
		`000`	GPIO operation.
8-63	—	Don't care	—

TABLE 3-7: RESPONSE STRUCTURE – `Read USB Manufacturer Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB0	`Read Flash Data` – command code echo.
1	0x00	Command completed successfully.
2	Note 2	Number of bytes + 2 in the provided USB Manufacturer Descriptor String. The actual string starts at Byte Index 4.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0(1)	—	Lower byte of the 16-bit Unicode character.
4 + 2 x Unicode_char_number + 1(1)	—	Higher byte of the 16-bit Unicode character.
(4 + 2 x Unicode_char_number + 2) – 63(1)	—	Don't care. Only if the USB String Descriptor is less than 60 bytes long in total.

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).
2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

TABLE 3-8: RESPONSE STRUCTURE – `Read USB Product Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB0	`Read Flash Data` – command code echo.
1	0x00	Command completed successfully.
2	Note 2	Number of bytes + 2 in the provided USB Product Descriptor String. The actual string starts at Byte Index 4.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0(1)	—	Lower byte of the 16-bit Unicode character.
4 + 2 x Unicode_char_number + 1(1)	—	Higher byte of the 16-bit Unicode character.
(4 + 2 x Unicode_char_number + 2) – 63(1)	—	Don't care. Only if the USB String Descriptor is less than 60 bytes long (in total).

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).
2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

TABLE 3-9: RESPONSE STRUCTURE – `Read USB Serial Number Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB0	`Read Flash Data` – command code echo.
1	0x00	Command completed successfully.
2	Note 2	The number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0(1)	—	Lower byte of the 16-bit Unicode character
4 + 2 x Unicode_char_number + 1(1)	—	Higher byte of the 16-bit Unicode character
(4 + 2 x Unicode_char_number + 2) – 63(1)	—	Don't care. Only if the USB String Descriptor is less than 60 bytes long in total.

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

TABLE 3-10: RESPONSE STRUCTURE – `Read Chip Factory Serial Number` SUB-COMMAND(1)

Byte Index	Value	Effect
0	0xB0	`Read Flash Data` – command code echo.
1	0x00	Command completed successfully.
2	—	Structure length.
3	Don't care	—
4 – (4 + Structure Length – 1)	—	Structure data – Factory Serial Number String.
(4 + Structure Length) – 63	Don't care	—

Note 1: The Chip Serial Number is typically 8 bytes in length.

3.1.3 Write Flash Data

This command is used to write various important data structures and strings into the Flash memory of the device.

TABLE 3-11: COMMAND STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0xB1	Write Flash Data – command code.
1	Write Flash Data sub-code. The value in this field will instruct the MCP2221A about the particular Flash settings to be altered.	0x00	Write Chip Settings – writes the MCP2221A Flash settings.
		0x01	Write GP Settings – writes the MCP2221A Flash GP settings.
		0x02	Write USB Manufacturer Descriptor String – writes the USB Manufacturer String Descriptor used during the USB enumeration.
		0x03	Write USB Product Descriptor String – writes the USB Product String Descriptor used during the USB enumeration.
		0x04	Write USB Serial Number Descriptor String – writes the USB Serial Number String Descriptor used during the USB enumeration.
		Any other value	No meaning. The device will reply with a code for an unsupported command at Byte Index 1 in the response report.
2-63	Data to be written	—	Data format depends on the Write Flash Data sub-code (at Byte Index 1).

TABLE 3-12: SUB-COMMAND STRUCTURE – `Write Chip Settings` SUB-COMMAND

Byte Index	Function Description	Value	Effect
0	—	0xB1	`Write Flash Data` – command code.
1	—	0x00	`Write Chip Settings` – writes the MCP2221A Flash device settings.
2	Bit 7: CDC serial number enumeration enable	`1`	The USB serial number will be used during the USB enumeration of the CDC interface.
		`0`	No serial number descriptor will be presented during the USB enumeration.
	Bits 6-2: Not implemented	—	—
	Bits 1-0: Chip configuration security option	`11-10`	Permanently locked.
		`01`	Password-protected.
		`00`	Unsecured.
3	Bits 7-5	Don’t care	—
	Bits 4-0: Clock output divider value	—	If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin.
4	Bits 7-6: DAC reference voltage option	`11`	Reference voltage is 4.096V (only if VDD is above this voltage).
		`10`	Reference voltage is 2.048V.
		`01`	Reference voltage is 1.024V.
		`00`	Reference voltage is off (this is useful for the case in which the DAC uses another reference other than VRM DAC, i.e., VDD).
	Bit 5: DAC reference option	`1`	DAC reference is VDD.
		`0`	DAC reference is VRM DAC voltage.
	Bits 4-0: Power-up DAC value	—	—
5	Bit 7	Don’t care	—
	Bit 6: Interrupt detection – negative edge	—	If set, the interrupt detection flag will be set when a negative edge occurs.
	Bit 5: Interrupt detection – positive edge	—	If set, the interrupt detection flag will be set when a positive edge occurs.
	Bits 4-3: ADC reference voltage	`11`	Reference voltage is 4.096V (only if VDD is above this voltage).
		`10`	Reference voltage is 2.048V.
		`01`	Reference voltage is 1.024V.
		`00`	Reference voltage is off (this is useful for the case in which the ADC uses another reference other than VRM ADC; e.g., VDD).
	Bit 2: ADC reference option	`1`	ADC reference voltage is VRM ADC.
		`0`	ADC reference voltage is VDD.
	Bit 1	Don’t care	—
	Bit 0	Don’t care	—
6	Lower byte of the 16-bit USB VID value.	—	—
7	Higher byte of the 16-bit USB VID value.	—	—

TABLE 3-12: SUB-COMMAND STRUCTURE – `Write Chip Settings` SUB-COMMAND (CONTINUED)

Byte Index	Function Description	Value	Effect
8	Lower byte of the 16-bit USB PID value.	—	—
9	Higher byte of the 16-bit USB PID value.	—	—
10	USB power attributes	—	This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes value) during the USB enumeration.
11	USB requested number of mA(s)	—	The requested mA value during the USB enumeration will represent the value at this index multiplied by two.
12-19	8-byte password (for Flash modifications protection)	—	—
20-63	—	Don't care	—

TABLE 3-13: SUB-COMMAND STRUCTURE – `Write GP Settings` SUB-COMMAND

Byte Index	Function Description	Value	Effect
0	—	0xB1	`Write Flash Data` – command code.
1	—	0x01	`Write GP Settings` – it will write the MCP2221A Flash GP settings.
2	**GP0 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP0 is set as an output GPIO, this value will be present at the GP0 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bit 2-0: GP0 designation	`111-011`	Don't care.
		`010`	Dedicated function operation (SSPND).
		`001`	Alternate Function 0 (LED_URx).
		`000`	GPIO operation.
3	**GP1 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP1 is set as an output GPIO, this value will be present at the GP1 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP1 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP1 designation	`111-101`	Don't care.
		`100`	Alternate Function 2 (interrupt detection).
		`011`	Alternate Function 1 (LED_UTx).
		`010`	Alternate Function 0 (ADC1).
		`001`	Dedicated function operation (clock output).
		`000`	GPIO operation.
4	**GP2 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP2 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP2 designation	`111-100`	Don't care.
		`011`	Alternate Function1 (DAC1).
		`010`	Alternate Function 0 (ADC2).
		`001`	Dedicated function operation (clock output).
		`000`	GPIO operation.

TABLE 3-13: SUB-COMMAND STRUCTURE – `Write GP Settings` SUB-COMMAND (CONTINUED)

Byte Index	Function Description	Value	Effect
5	**GP3 Power-up Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP3 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset.
	Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation.	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP3 designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC2).
		`010`	Alternate function 0 (ADC3).
		`001`	Dedicated function operation (LED_I2C).
		`000`	GPIO operation.
6-63		Don't care	—

TABLE 3-14: SUB-COMMAND STRUCTURE – `Write USB Manufacturer Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB1	`Write Flash Data` – command code.
1	0x02	`Write USB Manufacturer Descriptor String` – writes the USB Manufacturer String Descriptor used during the USB enumeration.
2	Note 2	Number of bytes + 2 in the provided USB Serial Number Descriptor String.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0[1]	—	Lower byte of the 16-bit Unicode character.
4 + 2 x Unicode_char_number + 1[1]	—	Higher byte of the 16-bit Unicode character.

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).
2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

TABLE 3-15: SUB-COMMAND STRUCTURE – `Write USB Product Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB1	`Write Flash Data` – command code.
1	0x03	`Write USB Product Descriptor String` – writes the USB Product String Descriptor used during the USB enumeration.
2	Note 2	Number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0[1]	—	Lower byte of the 16-bit Unicode character.
4 + 2 x Unicode_char_number + 1[1]	—	Higher byte of the 16-bit Unicode character.

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).
2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

TABLE 3-16: SUB-COMMAND STRUCTURE – `Write USB Serial Number Descriptor String` SUB-COMMAND

Byte Index	Value	Effect
0	0xB1	`Write Flash Data` – command code.
1	0x04	`Write USB Serial Number Descriptor String` – writes the USB Serial Number String Descriptor used during the USB enumeration.
2	Note 2	Number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4.
3	0x03	The value at this index must always be 0x03.
4 + 2 x Unicode_char_number + 0(1)	—	Lower byte of the 16-bit Unicode character.
4 + 2 x Unicode_char_number + 1(1)	—	Higher byte of the 16-bit Unicode character.

Note 1: "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

2: The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

3.1.3.1 Responses

TABLE 3-17: RESPONSE STRUCTURE – `Read Chip Factory Serial Number` SUB-COMMAND

Byte Index	Value	Effect
0	0xB1	`Write Flash Data` – command code.
1	0x00	Command completed successfully.
	0x02	Command not supported.
	0x03	Command not allowed.
2-63	Don't care	—

3.1.4 Send Flash Access Password

This command is used to send a user-supplied password that will be compared to the one stored in the device's Flash when Flash updates (Chip/GP configuration, USB strings) are required and the Flash data are password-protected.

In the case where no protection mechanism is in place or the Flash data have been permanently locked, this command has no meaning.

TABLE 3-18: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0xB2	`Send Flash Access Password` – command code.
1	Don't care	—
2	—	Password Byte 1.
3	—	Password Byte 2.
4	—	Password Byte 3.
5	—	Password Byte 4.
6	—	Password Byte 5.
7	—	Password Byte 6.
8	—	Password Byte 7.
9	—	Password Byte 8.
10-63	—	Don't care.

3.1.4.1 Responses

TABLE 3-19: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0xB2	`Send Flash Access Password` – command code echo.
1	0x00	Command completed successfully.
	0x03	Command not allowed (when the number of failed Flash updates has been reached, no password will be accepted).
2-63	Don't care	—

3.1.5 I^{2}C Write Data

This command is used to write user-given data to the I^{2}C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

- The I^{2}C engine will send the “Start” condition.
- The selected I^{2}C client address is sent next and the I^{2}C engine will wait for the client to send an Acknowledge bit.
- The user data follow next and the I^{2}C engine awaits for the Acknowledge bit from the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I^{2}C engine will send the “Stop” condition on the bus.

TABLE 3-20: COMMAND STRUCTURE(1)

Byte Index	Value	Effect
0	0x90	I^{2}C Write Data – command code.
1	Low Byte	Requested I^{2}C transfer length – 16-bit value – low byte.
2	High Byte	Requested I^{2}C transfer length – 16-bit value – high byte.
3	I^{2}C Client Address	8-bit value representing the I^{2}C client address to communicate with (even – address to write, odd – address to read) (**Note 2**).
4-63	—	User data to be sent to the selected I^{2}C client device.

Note 1: When the requested transfer length is more than 60 bytes, subsequent “I^{2}C Write Data” commands will transport the remainder of the user data (until the requested length).

2: The I^{2}C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

3.1.5.1 Responses

TABLE 3-21: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x90	I^{2}C Write Data – command code echo.
1	0x00	Command completed successfully.
	0x01	I^{2}C engine is busy (command not completed).
2	—	Reserved.
3-63	Don’t care	—

3.1.6 I²C Write Data Repeated-START

This command is used to write user-given data to the I²C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

- The I²C engine will send the “Repeated Start” condition.
- The selected I²C client address is sent next and the I²C engine will wait for the client to send an Acknowledge bit.
- The user data follow next and the I²C engine waits for the Acknowledge bit from the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I²C engine will send the “Stop” condition on the bus.

TABLE 3-22: COMMAND STRUCTURE (1)

Byte Index	Value	Effect
0	0x92	I²C Write Data Repeated-START – command code (cI2C_CMD_RSTART_WRDATA7).
1	Low Byte	Requested I²C transfer length – 16-bit value – low byte.
2	High Byte	Requested I²C transfer length – 16-bit value – high byte.
3	I²C Client Address	8-bit value representing the I²C client address to communicate with (even – address to write, odd – address to read) (**Note 2**).
4-63	—	User data to be sent to the selected I²C client device

Note 1: When the requested transfer length is more than 60 bytes, subsequent “I²C Write Data Repeated-START” commands will transport the remainder of the user data (till the requested length).

2: The I²C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

3.1.6.1 Responses

TABLE 3-23: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x92	I²C Write Data Repeated-START – command code echo (cI2C_CMD_RSTART_WRDATA7).
1	0x00	Command completed successfully.
	0x01	I²C engine is busy (command not completed).
2	0x10	I²C engine sent START condition.
	0x40 (1)	I²C engine start the write user data to client procedure
3-63	Don’t care	—

Note 1: This value applies only if Byte Index 1 is 0x00.

3.1.7 I2C Write Data No STOP

This command is used to write user-given data to the I2C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

- The I2C engine will send the “Start” condition.
- The selected I2C client address is sent next and the I2C engine will wait for the client to send an Acknowledge bit.
- The user data follow next and the I2C engine waits for the Acknowledge bit from the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I2C engine will **not** send the “Stop” condition on the bus.

TABLE 3-24: COMMAND STRUCTURE (1)

Byte Index	Value	Effect
0	0x94	I2C Write Data No STOP – command code.
1	Low Byte	Requested I2C transfer length – 16-bit value – low byte.
2	High Byte	Requested I2C transfer length – 16-bit value – high byte.
3	I2C Client Address	8-bit value representing the I2C client address to communicate with (even – address to write, odd – address to read) **(Note 2)**.
4-63	—	User data to be sent to the selected I2C client device.

Note 1: When the requested transfer length is more than 60 bytes, subsequent “I2C Write Data No STOP” commands will transport the remainder of the user data (till the requested length).

2: The I2C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

3.1.7.1 Responses

TABLE 3-25: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x94	I2C Write Data No STOP – command code echo.
1	0x00	Command completed successfully.
	0x01	I2C engine is busy (command not completed).
2	—	Reserved.
3-63	Don’t care	—

3.1.8 I²C Read Data

This command is used to read user-given data to the I²C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

- The I²C engine will send the “Start” condition.
- The selected I²C client address is sent next and the I²C engine will wait for the client to send an Acknowledge bit.
- The user data are read next and the I²C engine sends the Acknowledge bit to the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be read from the I²C client on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I²C engine will send the “Stop” condition on the bus.

TABLE 3-26: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x91	I²C Read Data – command code.
1	Low Byte	Requested I²C transfer length – 16-bit value – low byte.
2	High Byte	Requested I²C transfer length – 16-bit value – high byte.
3	I²C Client Address	8-bit value representing the I²C client address to communicate with (even – address to write, odd – address to read) (**Note 1**).
4-63	Don’t care	—

Note 1: The I²C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

3.1.8.1 Responses

TABLE 3-27: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x91	I²C Read Data – command code echo.
1	0x00	Command completed successfully.
	0x01	I²C engine is busy (command not completed).
2	—	Reserved.
3-63	Don’t care	—

3.1.9 I^2C Read Data Repeated-START

This command is used to read user-given data to the I^2C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effect:

- The I^2C engine will send the "Repeated-START" condition.
- The selected I^2C client address is sent next and the I^2C engine will wait for the client to send an Acknowledge bit.
- The user data are read next and the I^2C engine sends the Acknowledge bit to the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be read from the I^2C client on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I^2C engine will send the "Stop" condition on the bus.

TABLE 3-28: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x93	I^2C Read Data Repeated-START – command code.
1	Low Byte	Requested I^2C transfer length – 16-bit value – low byte.
2	High Byte	Requested I^2C transfer length – 16-bit value – high byte.
3	I^2C Client Address	8-bit value representing the I^2C client address to communicate with (even – address to write, odd – address to read) (**Note 1**).
4-63	Don't care	—

Note 1: The I^2C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

3.1.9.1 Responses

TABLE 3-29: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x93	I^2C Read Data Repeated-START – command code echo.
1	0x00	Command completed successfully.
	0x01	I^2C engine is busy (command not completed).
2	—	Reserved.
3-63	Don't care	—

3.1.10 I^2C Read Data – Get I^2C Data

This command is used to read back the data from the I^2C client device.

TABLE 3-30: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x40	I^2C Read Data – Get I^2C Data – command code.
1-63	Don’t care	—

3.1.10.1 Responses

TABLE 3-31: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x40	I^2C Read Data – Get I^2C Data – command code echo.
1	0x00	Command completed successfully.
	0x41	Error reading the I^2C client data from the I^2C engine.
2	—	Reserved.
3	0-60	The number of read-back data bytes to follow in this packet: from 0 to a maximum of 60 bytes of read-back bytes.
	127	This value is signaled when an error has occurred and the following data should not be taken into account.
4-63	—	User data or Don’t care.

3.1.11 Set GPIO Output Values

This command is used to change the GPIO output value for those GP pins assigned for GPIO operation (GPIO outputs).

TABLE 3-32: COMMAND STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x50	Set GPIO Output Values – command code.
1	—	Don't care	—
2	Alter GP0 output (enable/disable)	0x00	Do not modify GP0 output (if GP0 is set as GPIO output).
		Any other value	The next byte (Index 3) will be the value used to set GP0 output (only if GP0 is set for GPIO output).
3	GP0 output value	0x00	GP0 (if set up for GPIO output operation) will take a logical value of '0'.
		Any other value	GP0 (if set up for GPIO output operation) will take a logical value of '1'.
4	Alter GP0 pin direction (enable/disable)	0x00	Leave the GP0 GPIO designation as is (input or output).
		Any other value	The next byte (index 5) will be the value used to set GP0's pin direction (only if GP0 is set for GPIO operation).
5	GP0 pin direction (input or output)	0x00	Set GP0 GPIO as output.
		Any other value	GP0 (if set up for GPIO operation) will be set as a digital input.
6	Alter GP1 output (enable/disable)	0x00	Do not modify GP1 output (if GP1 is set as GPIO output).
		Any other value	The next byte (Index 7) will be the value used to set GP1 output (only if GP1 is set for GPIO output).
7	GP1 output value	0x00	GP1 (if set up for GPIO output operation) will take a logical value of '0'.
		Any other value	GP1 (if set up for GPIO output operation) will take a logical value of '1'.
8	Alter GP1 pin direction (enable/disable)	0x00	Leave the GP1 GPIO designation as is (input or output).
		Any other value	The next byte (Index 9) will be the value used to set GP1's pin direction (only if GP1 is set for GPIO operation).
9	GP1 pin direction (input or output)	0x00	Set GP1 GPIO as output.
		Any other value	GP1 (if set up for GPIO operation) will be set as a digital input.
10	Alter GP2 output (enable/disable)	0x00	Do not modify GP2 output (if GP2 is set as GPIO output).
		Any other value	The next byte (Index 11) will be the value used to set GP2 output (only if GP2 is set for GPIO output).
11	GP2 output value	0x00	GP2 (if GP2 is set up for GPIO output operation) will take a logical value of '0'.
		Any other value	GP2 (if GP2 is set up for GPIO output operation) will take a logical value of '1'.
12	Alter GP2 pin direction (enable/disable)	0x00	Leave the GP2 GPIO designation as is (input or output).
		Any other value	The next byte (Index 13) will be the value used to set GP2's pin direction (only if GP2 is set for GPIO operation).
13	GP2 pin direction (input or output)	0x00	Set GP2 GPIO as output.
		Any other value	GP2 (if set up for GPIO operation) will be set as a digital input.

TABLE 3-32: COMMAND STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
14	Alter GP3 output (enable/disable)	0x00	Do not modify GP3 output (if GP3 is set as GPIO output).
		Any other value	The next byte (Index 11) will be the value used to set GP3 output (only if GP3 is set for GPIO output).
15	GP3 output value	0x00	GP3 (if set up for GPIO output operation) will take a logical value of '0'.
		Any other value	GP3 (if set up for GPIO output operation) will take a logical value of '1'.
16	Alter GP3 pin direction (enable/disable)	0x00	Leave the GP3 GPIO designation as is (input or output).
		Any other value	The next byte (Index 17) will be the value used to set GP3's pin direction (only if GP3 is set for GPIO operation)
17	GP3 pin direction (input or output)	0x00	Set GP3 GPIO as output.
		Any other value	GP3 (if set up for GPIO operation) will be set as a digital input.
18-63	Reserved	0x00	—

3.1.11.1 Responses

TABLE 3-33: RESPONSE 1 STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x50	`Set GPIO Output Values` – command code.
1	—	0x00	Command completed successfully.
2	Alter GP0 output (enable/disable) status	0xEE	If GP0 is not set for GPIO operation.
		Any other value	If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
3	GP0 output value status	0xEE	If GP0 is not set for GPIO operation.
		Any other value	If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
4	Alter GP0 pin direction (enable/disable)	0xEE	If GP0 is not set for GPIO operation.
		Any other value	If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
5	GP1 pin direction (input or output)	0xEE	If GP1 is not set for GPIO operation.
		Any other value	If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
6	Alter GP1 output (enable/disable) status	0xEE	If GP1 is not set for GPIO operation.
		Any other value	If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
7	GP1 output value status	0xEE	If GP1 is not set for GPIO operation
		Any other value	If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.

TABLE 3-33: RESPONSE 1 STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
8	Alter GP1 pin direction (enable/disable)	0xEE	If GP1 is not set for GPIO operation.
		Any other value	If the GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
9	GP1 pin direction (input or output)	0xEE	If GP1 is not set for GPIO operation.
		Any other value	If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
10	Alter GP2 output (enable/disable) status	0xEE	If GP2 is not set for GPIO operation.
		Any other value	If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
11	GP2 output value status	0xEE	If GP2 is not set for GPIO operation.
		Any other value	If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
12	Alter GP2 pin direction (enable/disable)	0xEE	If GP2 is not set for GPIO operation.
		Any other value	If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
13	GP2 pin direction (input or output)	0xEE	If GP2 is not set for GPIO operation.
		Any other value	If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
14	Alter GP3 output (enable/disable) status	0xEE	If GP3 is not set for GPIO operation.
		Any other value	If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
15	GP3 output value status	0xEE	If GP3 is not set for GPIO operation.
		Any other value	If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
16	Alter GP3 pin direction (enable/disable)	0xEE	If GP3 is not set for GPIO operation.
		Any other value	If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
17	GP3 pin direction (input or output)	0xEE	If GP3 is not set for GPIO operation.
		Any other value	If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure.
18-63	—	Don’t care	—

3.1.12 Get GPIO Values

This command is used to retrieve the GPIO direction and pin value for those GP pins assigned for GPIO operation (GPIO inputs or outputs).

TABLE 3-34: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x51	Get GPIO Values – command code.
1-63	Don’t care	—

3.1.12.1 Responses

TABLE 3-35: RESPONSE 1 STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x51	Get GPIO Values – command code.
1	—	0x00	Command completed successfully.
2	GP0 pin value	0xEE	If GP0 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP0 is already set for GPIO operation, the value represents the GP0 logic pin value.
3	GP0 direction value	0xEF	If GP0 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP0 is already set for GPIO operation, the value represents the GP0 pin designation (0x00 for output and 0x01 for input).
4	GP1 pin value	0xEE	If GP1 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP1 is already set for GPIO operation, the value represents the GP1 logic pin value.
5	GP1 direction value	0xEF	If GP1 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP1 is already set for GPIO operation, the value represents the GP1 pin designation (0x00 for output and 0x01 for input).
6	GP2 pin value	0xEE	If GP2 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP2 is already set for GPIO operation, the value represents the GP2 logic pin value.
7	GP2 direction value	0xEF	If GP2 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP2 is already set for GPIO operation, the value represents the GP2 pin designation (0x00 for output and 0x01 for input).
8	GP3 pin value	0xEE	If GP3 is not set for GPIO operation
		Other values (0x00 or 0x01)	If GP3 is already set for GPIO operation, the value represents the GP3 logic pin value.
9	GP3 direction value	0xEF	If GP3 is not set for GPIO operation.
		Other values (0x00 or 0x01)	If GP3 is already set for GPIO operation, the value represents the GP3 pin designation (0x00 for output and 0x01 for input).
10-63	—	Don’t care	—

3.1.13 Set SRAM settings

This command is used to alter various run-time Chip settings. The altered settings reside in SRAM memory and they will not affect the Chip's power-up/Reset default settings. These altered settings will be active until the next chip power-up/Reset.

TABLE 3-36: COMMAND STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x60	`Set SRAM settings` – command code.
1	—	Don't care	—
2	**Clock Output Divider Value** – this allows the user to modify the clock output value on-the-fly at run time		
	Bit 7: Enable loading of a new clock divider	`1`	Bits[4:0] will be loaded into the clock divider.
		`0`	Clock divider value won't be altered.
	Bits 6-5	Don't care	—
	Bits 4-3: Duty cycle	`00`	0% duty cycle.
		`01`	25% duty cycle.
		`10`	50% duty cycle.
		`11`	75% duty cycle.
	Bits 2-0: Clock divider value	—	—
3	**DAC Voltage Reference** – this allows the user to modify the DAC reference voltage		
	Bit 7: Enable loading of a new DAC reference	`1`	Bits[2:0] will be used for DAC reference voltage selection.
		`0`	DAC reference will remain unaltered.
	Bits 6-3	Don't care	—
	Bits 2-1: DAC V_{RM} voltage selection; these bits are used to change the DAC V_{RM} voltage	`11`	V_{RM} voltage is 4.096V (only if V_{DD} is higher than this value).
		`10`	V_{RM} voltage is 2.048V.
		`01`	V_{RM} voltage is 1.024V.
		`00`	V_{RM} voltage is off.
	Bit 0: This bit is used to change the DAC reference voltage	`1`	DAC voltage reference is the internal DAC voltage reference module (DAC V_{RM}).
		`0`	DAC voltage reference is V_{DD}.
4	**Set DAC Output Value**		
	Bit 7: Enable loading of a new DAC value	`1`	Bits[4:0] will be used for DAC reference voltage selection.
		`0`	The current DAC value will remain unaltered.
	Bits 6-5	Don't care	—
	Bits 4-0: The new DAC value	—	—

TABLE 3-36: COMMAND STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
5	**ADC Voltage Reference** – this allows the user to modify the ADC reference voltage		
	Bit 7: Enable loading of a new ADC reference	1	Bits[2:0] will be used for ADC reference voltage selection.
		0	ADC reference will remain unaltered.
	Bits 6-3	Don't care	—
	Bits 2-1: These bits are used to change the DAC VRM voltage	11	VRM voltage is 4.096V (only if VDD is higher than this value).
		10	VRM voltage is 2.048V.
		01	VRM voltage is 1.024V.
		00	VRM voltage is off.
	Bit 0: This bit is used to change the DAC reference voltage	1	VDD ADC voltage reference is the internal ADC voltage reference module (ADC VRM).
		0	ADC voltage reference is VDD.
6	**Set Up the Interrupt Detection Mechanism and Clear the Detection Flag** – useful for preparing the interrupt detection module to detect a new interrupt condition		
	Bit 7: Enable the modification of the interrupt detection conditions	1	The interrupt detection settings and flag will change.
		0	The interrupt detection settings and flag will remain unchanged.
	Bits 6-5	Don't care	—
	Bit 4	—	Enable the modification of the positive edge detection.
	Bit 3: The new value for the positive edge detector	1	Interrupt detection will trigger on positive edges.
		0	Interrupt detection will not trigger on positive edges.
	Bit 2	—	Enable the modification of the negative edge detection.
	Bit 1: The new value for the negative edge detector	1	Interrupt detection will trigger on negative edges.
		0	Interrupt detection will not trigger on negative edges.
	Bit 0: Clear the interrupt detection flag	1	Clear the interrupt detection flag.
		0	Leave the interrupt detection flag as is.
7	Bit 7: Alter GPIO configuration: alters the current GP designation	1	Alter the GP designation. The values from Byte Index 8 will be used to load a new set of values into the SRAM GP settings.
		0	Do not alter the current GP designation.
8	**GP0 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP0 is set as an output GPIO, this value will be present at the GP0 pin
	Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation	1	GPIO Input mode.
		0	GPIO Output mode.
	Bits 2-0: GP0 designation	111-011	Don't care.
		010	Alternate Function 0 (LED_URx).
		001	Dedicated function operation (SSPND).
		000	GPIO operation.

TABLE 3-36: COMMAND STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
9	**GP1 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP1 is set as an output GPIO, this value will be present at the GP1 pin.
	Bit 3: GPIO direction (input/output) – works only when GP1 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP0 designation	`111-101`	Don't care.
		`100`	Alternate Function 2 (interrupt detection).
		`011`	Alternate Function 1 (LED_UTx).
		`010`	Alternate Function 0 (ADC1).
		`001`	Dedicated function operation (clock output).
		`000`	GPIO operation.
10	**GP2 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP2 is set as an output GPIO, this value will be present at the GP2 pin.
	Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP2 designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC1).
		`010`	Alternate Function 0 (ADC2).
		`001`	Dedicated function operation (USBCFG).
		`000`	GPIO operation.
11	**GP3 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When GP3 is set as an output GPIO, this value will be present at the GP3 pin.
	Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP3 Designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC2).
		`010`	Alternate Function 0 (ADC3).
		`001`	Dedicated function operation (LED_I2C).
		`000`	GPIO operation.
12-63	Reserved	0x00	—

3.1.13.1 Responses

TABLE 3-37: RESPONSE 1 STRUCTURE

Byte Index	Value	Effect
0	0x60	`Set SRAM settings` – command code echo.
1	0x00	Command completed successfully.
2-63	Don't care	—

3.1.14 Get SRAM Settings

This command is used to retrieve the run-time Chip and GP settings.

TABLE 3-38: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x61	Get SRAM Settings – command code echo.
1-63	0x00	Command completed successfully.

3.1.14.1 Responses

TABLE 3-39: RESPONSE 1 STRUCTURE

Byte Index	Function Description	Value	Effect
0	—	0x61	Get SRAM Settings – command code echo.
1	—	0x00	Command completed successfully.
2	Length in bytes of the SRAM Chip settings area	—	—
3	Length in bytes of the SRAM GP settings area	—	—
4	Bit 7: CDC serial number enumeration enable	1	The USB serial number will be used during the USB enumeration of the CDC interface.
		0	No serial number descriptor will be presented during the USB enumeration.
	Bits 6-2:Not implemented	—	—
	Bits 1-0: Chip configuration security option	10	Permanently locked.
		01	Password-protected.
		00	Unsecured.
5	Bits 7-5	Don’t care	—
	Bits 4-0: Clock Output divider value	—	If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin. (Bits[4:3] for duty cycle and bits[2:0] for the clock divider.)
6	Bits 7-6: DAC reference voltage option	11	Reference voltage is 4.096V.
		10	Reference voltage is 2.048V.
		01	Reference voltage is 1.024V.
		00	Reference voltage is off (this is useful for the case in which the DAC uses another reference other than VRM DAC; e.g., VDD).
	Bit 5: DAC reference option	1	DAC reference is VRM DAC voltage.
		0	DAC reference is VDD.
	Bits 4-0: Power-up DAC value	—	—

TABLE 3-39: RESPONSE 1 STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
7	Bit 7	Don't care	—
	Bit 6: Interrupt detection – negative edge	—	If set, the interrupt detection flag will be set when a negative edge occurs.
	Bit 5: Interrupt detection – positive edge	—	If set, the interrupt detection flag will be set when a positive edge occurs
	Bits 4-3: ADC reference voltage	`11`	Reference voltage is 4.096V (only if V_{DD} is above this voltage).
		`10`	Reference voltage is 2.048V.
		`01`	Reference voltage is 1.024V.
		`00`	Reference voltage is off (this is useful for the case in which the ADC uses another reference other than V_{RM} DAC; e.g., V_{DD}).
	Bit 2: ADC reference option	`1`	ADC reference is V_{RM} ADC.
		`0`	ADC reference is V_{DD}.
	Bit 1	Don't care	—
	Bit 2	Don't care	—
8	Lower byte of the 16-bit USB VID value	—	—
9	Higher byte of the 16-bit USB VID value	—	—
10	Lower byte of the 16-bit USB PID value	—	—
11	Higher byte of the 16-bit USB PID value	—	—
12	USB power attributes	—	This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes value) during the USB enumeration.
13	USB requested number of mA(s)	—	The requested mA value during the USB enumeration will represent the value at this index multiplied by two.
14	Current Supplied Password Byte 1	—	—
15	Current Supplied Password Byte 2	—	—
16	Current Supplied Password Byte 3	—	—
17	Current Supplied Password Byte 4	—	—
18	Current Supplied Password Byte 5	—	—
19	Current Supplied Password Byte 6	—	—
20	Current Supplied Password Byte 7	—	—
21	Current Supplied Password Byte 8	—	—

TABLE 3-39: RESPONSE 1 STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
22	**GP0 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO Output value	—	When the GP0 is set as an output GPIO, this value will be present at the GP0 pin.
	Bit 3: GPIO Direction (Input/Output) – Works only when GP0 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP0 Designation	`111-011`	Don't care.
		`010`	Alternate Function 0 (LED_URx).
		`001`	Dedicated function operation (SSPND).
		`000`	GPIO operation.
23	**GP1 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO Output value	—	When the GP1 is set as an output GPIO, this value will be present at the GP1 pin.
	Bit 3: GPIO Direction (Input/Output) – Works only when GP1 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP1 Designation	`111-101`	Don't care.
		`100`	Alternate Function 2 (interrupt detection).
		`011`	Alternate Function 1 (LED_UTx).
		`010`	Alternate Function 0 (ADC1).
		`001`	Dedicated function operation (clock output).
		`000`	GPIO operation.
24	**GP2 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When the GP2 is set as an output GPIO, this value will be present at the GP2 pin.
	Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP2 Designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC1).
		`010`	Alternate Function 0 (ADC2).
		`001`	Dedicated function operation (USBCFG).
		`000`	GPIO operation.

TABLE 3-39: RESPONSE 1 STRUCTURE (CONTINUED)

Byte Index	Function Description	Value	Effect
25	**GP3 Settings**		
	Bits 7-5	Don't care	—
	Bit 4: GPIO output value	—	When the GP3 is set as an output GPIO, this value will be present at the GP3 pin.
	Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation	`1`	GPIO Input mode.
		`0`	GPIO Output mode.
	Bits 2-0: GP3 designation	`111-100`	Don't care.
		`011`	Alternate Function 1 (DAC2).
		`010`	Alternate Function 0 (ADC3).
		`001`	Dedicated function operation (LED_I2C).
		`000`	GPIO operation.
26-63		Don't care	—

3.1.15 `Reset Chip`

This command is used to force a Reset of the MCP2221A device. This command is useful when the Flash memory is updated with new data. The MCP2221A would need to be re-enumerated to see the new data.

Note:	This command is the only command that does not expect a response.

TABLE 3-40: COMMAND STRUCTURE

Byte Index	Value	Effect
0	0x70	`Reset Chip` – command code.
1	0xAB	—
2	0xCD	—
3	0xEF	—
4-63	0x00	Reserved.