AN1139: CP2615 I/O Protocol

The CP2615 I/O Protocol (IOP) is a stateless, message based protocol designed specifically to provide access to the CP2615 I/O functions. The protocol is transported by USB bulk transfers over the CP2615 Serial Interface. It includes a notification capability that is designed to work seamlessly within the event driven architecture of a host application. The following features can be accessed using the CP2615 I/O Protocol:

- Query device identification information
- Query GPIO configuration
- Observe and control the GPIO.15-0 pins
- Observe the analog pin GPIO.8/ADC
- Receive autonomous notifications of GPIO/ADC changes
- Perform small transfers on the I^2C bus
- Query error status of the I/O Protocol

KEY POINTS

- This document describes the CP2615 I/O Protocol, transported by USB bulk transfers.

1. IOP Message Summary

The following table summarizes the messages provided by the I/O Protocol. Because IOP is stateless, messages can be sent in any order. Many of the messages form request/response pairs (e.g., `iop_GetAccessoryInfo` and `iop_AccessoryInfo`). In these cases, the host application sends a request message and sometime later the accessory will send the requested information. Other request messages (e.g., `iop_SetDigitalPort`) do not have a response and are assumed to complete successfully. Finally, the protocol supports automatic notifications. If enabled the accessory automatically sends value response messages (e.g., `iop_DigitalPortValue`) whenever the monitored value changes. By enabling notifications, the host application can avoid polling for updates.

Message Name	ID	Source	Description
iop_GetAccessoryInfo	0xD100	Host	Request CP2615 identification information.
iop_AccessoryInfo	0xA100	CP2615	CP2615 identification information.
iop_GetPortConfiguration	0xD203	Host	Request pin configuration for digital port.
iop_PortConfiguration	0xA203	CP2615	Pin configuration data.
iop_SetDigitalPortNotify	0xD200	Host	Configure automatic reporting of pin changes.
iop_GetDigitalPort	0xD201	Host	Report current digital port value.
iop_DigitalPortValue	0xA201	CP2615	Current digital port value.
iop_SetDigitalPort	0xD202	Host	Set value of select pins within a digital port.
iop_SetAnalogPinNotify	0xD300	Host	Configure automatic reporting of analog pin.
iop_GetAnalogPin	0xD301	Host	Report current analog pin value.
iop_AnalogPinValue	0xA301	CP2615	Current analog pin value.
iop_DoI2cTransfer	0xD400	Host	Perform write/read transfer on I^{2}C bus.
iop_I2cTransferResult	0xA400	CP2615	Read data and status from I^{2}C transfer.
iop_GetSerialState	0xD501	Host	Report error status for serial session.
iop_SerialState	0xA501	CP2615	Serial session error status.

2. IOP Message Format

CP2615 IOP messages consist of a fixed sized header followed by a variable length payload. The payload carries fixed size, positional parameters that are determined by the message type. The message type is indicated in the header by the 16-bit Message ID field. The message header also includes a preamble to determine the start of a message and a 16-bit Length field that reflects the overall size of the message.

Byte	Description
0. Preamble MSB	0x2A
1. Preamble LSB	0x2A
2. Length MSB	Total message length (header + message payload).
3. Length LSB	Min = 6, Max = 64
4. Message ID MSB	Identifies the message.
5. Message ID LSB	Identifies the message.
6. Payload[0]	Message parameter data (size and content depend on message type).
⋮	⋮
N+6. Payload[N-1]	N = Payload size

The 16-bit preamble for the IOP message header is 0x2A2A. This is the ASCII value for "**", which is easy to identify in communication traces. The preamble in conjunction with the length field provides framing information for a message parser.

The length field is 16-bits wide and counts all bytes in the IOP message (header + payload). Thus the minimum length is a message with no payload and is equal to the size of the header which is 6.

The 16-bit message ID field uniquely identifies the message and its parameters. A message parser uses this field to determine the structure of the payload. Future versions of the protocol could append new parameters at the end of the payload, so message parsers should ignore "extra" payload to remain forward compatible.

Note: All multi-byte fields are presented and transferred in big-endian order. This applies to both the message header and the message parameters.

2.1 CP2615 Specific Values

Several messages include a digital port or analog pin parameter. Ports are collections of 16 digital pins. For the CP2615, both of these parameters are always set to 0.

The maximum message length supported by the CP2615 is 64. This maximum message size limits the I^{2}C data sizes to 54 in the `iop_DoI2cTransfer` message.

3. IOP Message Descriptions

3.1 iop_GetAccessoryInfo

ID: 0xD100, Source: Host

This message requests specific information about the accessory (i.e. system) that contains the CP2615. The result is returned in the `iop_AccessoryInfo` message.

Payload Byte	Description
N/A	This message has no parameters.

3.2 iop_AccessoryInfo

ID: 0xA100, Source: CP2615

This message returns accessory specific information that is not provided by the Identification Information control message. It is sent after receiving the `iop_GetAccessoryInfo` message.

Payload Byte	Descriptions
0. PartID MSB	Encoded part number of the chip (0x1400 for CP2615-A01, 0x1500 for CP2615-A02).
1. PartID LSB	
2. OptionID MSB	Customer specified value read from the configuration.
3. OptionID LSB	
4. ProtocolVersion MSB	IOP protocol version.
5. ProtocolVersion LSB	

3.3 iop_GetPortConfiguration

ID: 0xD203, Source: Host

This message requests the pin configuration data for all pins in the specified port. The result is returned in the `iop_PortConfiguration` message.

Payload Byte	Description
0. PortNumber	The port to query.

3.4 iop_PortConfiguration

ID: 0xA203, Source: CP2615

This message reports the pin configuration data for all pins in the indicated port. It is sent after receiving an `iop_GetPortConfiguration` message.

Payload Byte	Description
0. PortNumber	The port returned in this message.
1. PinFunction MSB	Bitmask: 0 = Alternate, 1 = GPIO.
2. PinFunction LSB	
3. PinDirection MSB 4. PinDirection LSB	Bitmask: 0 = Input, 1 = Output.
5. PinOutMode MSB 6. PinOutMode LSB	Bitmask: 0 = Open-Drain, 1 = Push-Pull.

3.5 iop_SetDigitalPortNotify

ID: 0xD200, Source: Host

This message enables one or more pins in a port to generate notification messages. Notifications will begin immediately once enabled for a pin (i.e. there are not separate messages to globally start/stop notifications). The firmware will watch pins that have notification enabled and will automatically send an `iop_DigitalPortValue` message if their value changes. Write a PinMask value of 0x0000 to disable all notifications on a port.

Payload Byte	Description
0. PortNumber	The target digital port.
1. PinMask MSB	Bitmask determines which pins will return notification messages.
2. PinMask LSB	

3.6 iop_GetDigitalPort

ID: 0xD201, Source: CP2615

This message requests the current value for the specified digital port. The result is returned in the iop_DigitalPortValue message. The PinMask parameter is copied into the return `iop_DigitalPortValue` message and is not interpreted.

Payload Byte	Description
0. PortNumber	The digital port to query.
1. PinMask MSB	Returned in the PinMask field of the iop_DigitalPortValue message.
2. PinMask LSB	

3.7 iop_DigitalPortValue

ID: 0xA201, Source: CP2615

This message returns the current digital value for the indicated port. It is sent after receiving an `iop_GetDigitalPort` message. The PinMask parameter is copied from the `iop_GetDigitalPort` message. This message is also sent if any of the port pins with notifications enabled change value. In this case, the PinMask parameter indicates which pins changed value.

Payload Byte	Description
0. PortNumber	The target digital port.
1. PinMask MSB	Bitmask indicates which pins changed or were queried by the iop_GetDigitalPort message.
2. PinMask LSB	
3. PortValue MSB	Bitfield returns current digital pin values (all pins are returned).
4. PortValue LSB	

3.8 iop_SetDigitalPort

ID: 0xD202, Source: Host

This message writes one or more digital output pins in a port. The PinMask determines which pins will be written. If a target pin is not a digital output, the pin is not written. A subsequent `iop_DigitalPortValue` message may be returned if the target pin values have changed and notifications are enabled on any of those pins.

Payload Byte	Description
0. PortNumber	The target digital port.
1. PinMask MSB	Bitmask determines which pins will be written.
2. PinMask LSB	
3. PortValue MSB	Bitfield holds pin values that will be written.
4. PortValue LSB	

3.9 iop_SetAnalogPinNotify

ID: 0xD300, Source: Host

This message is used to enable or disable notifications from the specified analog pin. Notifications will begin immediately once enabled for a pin (i.e. there are not separate messages to globally start/stop notifications). Analog notifications are `iop_AnalogPinValue` messages that are sent automatically whenever a change is detected on the target analog pin. The magnitude of the change must be greater than or equal to the provided Threshold value to trigger a notification. A Threshold value of 0 disables notifications from the pin. Setting the Threshold to a non-zero value will trigger the first notification message.

Payload Byte	Description
0. PinNumber	The target analog pin.
1. Threshold MSB	0 = Off. N = magnitude of change needed to trigger a notification.
2. Threshold LSB	

3.10 iop_GetAnalogPin

ID: 0xD301, Source: Host

This message requests the current value for the specified analog pin. The result is returned in the `iop_AnalogPinValue` message.

Payload Byte	Description
0. PinNumber	The analog pin to query.

3.11 iop_AnalogPinValue

ID: 0xA301, Source: CP2615

This message returns the current value for the indicated analog pin. It is sent after receiving an `iop_GetAnalogPin` message or whenever the value of an analog pin with notifications enabled changes.

Payload Byte	Description
0. PinNumber	The target analog pin.
1. PinValue MSB	Value read from the analog pin. 10-bit right-justified, full-scale VDD corresponds to 0x3FFF.
2. PinValue LSB	
3. TimeStamp[3] (MSB)	32-bit Timestamp. 1 ms resolution.
4. TimeStamp[2]	
5. TimeStamp[1]	
6. TimeStamp[0] (LSB)	

3.12 iop_DoI2cTransfer

ID: 0xD400, Source: Device

This message requests that the specified I^{2}C transfer be completed and the results returned. The transfer consists of a write cycle followed by a read cycle. Each cycle ends with a stop condition. Repeated start between the write and read cycles is not supported. Either the write or the read cycle may be skipped by setting the associated byte count to 0. Transfer status and any data read from the I^{2}C bus is returned in an `iop_I2cTransferResult` message with a matching tag parameter.

Payload Byte	Description
0. Tag	Arbitrary identifier that will be returned in the subsequent iop_I2cTransferResult message. The host typically increments the tag for each iop_DoI2cTransfer message.
1. SlaveAddress	I^{2}C slave address. Bit0 (R/W) is ignored.
2. ReadCount	Number of bytes to read. Set to 0 for write-only transfer. Max value is 54 for CP2615.
3. WriteCount	Number of bytes to write. Set to 0 for read-only transfer. Max value is 54 for CP2615.
4. WriteData[0]	I^{2}C write data.
⋮	⋮
N+3. WriteData[N-1]	N = WriteCount. Max value for N is 54 (64 - 6 - 4).

3.13 iop_I2cTransferResult

ID: 0xA400, Source: CP2615

This message returns the status and any data read from a matching `iop_DoI2cTransfer` message. It is sent when the corresponding I^2C transfer is complete. The Tag parameter is copied from the preceding `iop_DoI2cTransfer` message.

Payload Byte	Description
0. Tag	Tag from preceding iop_DoI2cTransfer message.
1. SlaveAddress	I^2C slave address. Bit0 (R/W) is ignored.
2. Status	Final status of I^2C transfer. 0 = Success, non-zero = Error.
3. ReadCount	Number of bytes read. 0 for write-only transfer. Max value is 54 for CP2615.
4. ReadData[0]	I^2C read data.
⋮	⋮
N+3. ReadData[N-1]	N = ReadCount. Max value for N is 54 (64 - 6 - 4).

3.14 iop_GetSerialState

ID: 0xD501, Source: Host

This message requests the error status for the serial session. The result is returned in the `iop_SerialState` message.

Payload Byte	Description
N/A	This message has no parameters.

3.15 iop_SerialState

ID: 0xA501, Source: CP2615

This message returns the error status for the serial session. It is sent after receiving the `iop_GetSerialState` message. Errors are automatically cleared when the message is sent.

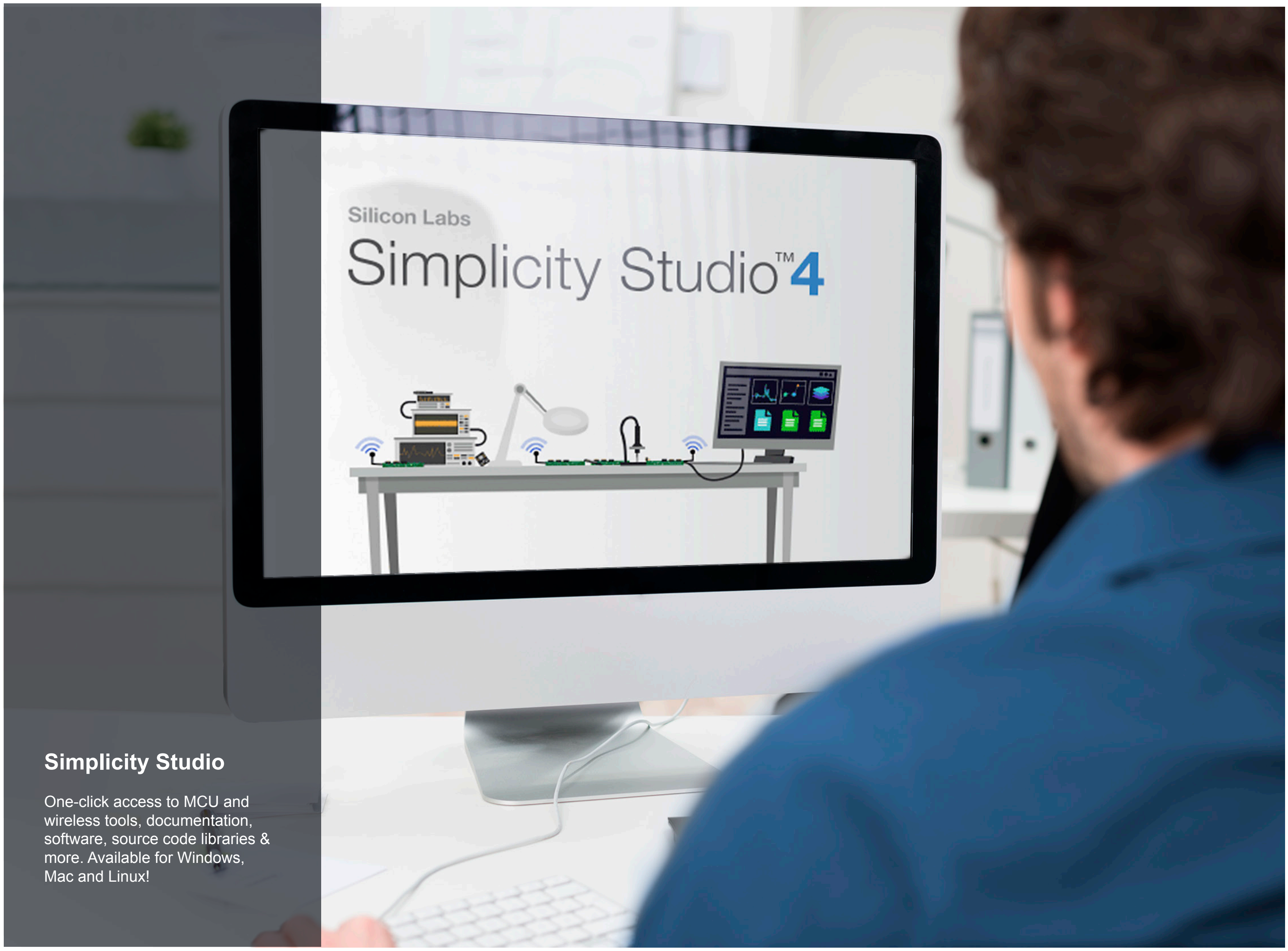
Payload Byte	Description
0. ReceiveError	Receive stream errors. 0 = no errors. Non-zero value indicates the following error(s) have occurred: • 1xxx xxxx: UART RX FIFO overrun • x1xx xxxx: Parity error • xx1x xxxx: Receive buffer overflow

4. Revision History

Revision 0.1

March, 2018

- Initial revision.



Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Disclaimer

Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

Trademark Information

Silicon Laboratories Inc.® , Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Micrium, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri, and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com