Data Gateway Interface

Data Gateway Interface User's Guide

Description

The Data Gateway Interface (DGI) is a USB interface for handling the low-level transport of data to and from a target MCU. The DGI is available on a selection of tools and on-board debuggers, such as the Power Debugger and the EDBG, as found on Xplained Pro.

The DGI provides several interfaces utilizing the same API for configuration and communication. Each interface implements an abstraction to a physical communication interface, such as SPI and UART, or represents a service not directly tied to a physical communication interface, such as the timestamp interface.

Table of Contents

Description....................1

1. USB Communication.................... 4

2. Protocol.................... 5

   2.1. DGI_CMD_SIGN_ON....................6
   2.2. DGI_CMD_SIGN_OFF....................6
   2.3. DGI_CMD_GET_VERSION....................7
   2.4. DGI_CMD_SET_MODE....................7
   2.5. DGI_CMD_TARGET_RESET.................... 8
   2.6. DGI_CMD_INTERFACES_LIST....................8
   2.7. DGI_CMD_INTERFACES_ENABLE....................9
   2.8. DGI_CMD_INTERFACES_SET_CONFIG....................9
   2.9. DGI_CMD_INTERFACES_GET_CONFIG....................10
   2.10. DGI_CMD_INTERFACES_POLL_DATA....................10
   2.11. DGI_CMD_INTERFACES_SEND_DATA.................... 11
   2.12. DGI_CMD_INTERFACES_STATUS....................12

3. Interfaces....................13

   3.1. Timestamp....................13
      3.1.1. Parsing....................13
      3.1.2. Configuration.................... 15
   3.2. GPIO.................... 15
      3.2.1. Parsing....................15
      3.2.2. Configuration.................... 15
   3.3. SPI....................15
      3.3.1. Parsing....................16
      3.3.2. Configuration.................... 16
   3.4. USART.................... 16
      3.4.1. Parsing....................16
      3.4.2. Configuration.................... 16
   3.5. I^2C.................... 17
      3.5.1. Parsing....................17
      3.5.2. Configuration.................... 17
   3.6. Power.................... 17
      3.6.1. Parsing....................17
      3.6.2. Configuration.................... 19

4. Revision History....................22

The Microchip Web Site.................... 23

Customer Change Notification Service....................23

Customer Support.................... 23

Microchip Devices Code Protection Feature.. 23

Legal Notice..24

Trademarks.. 24

Quality Management System Certified by DNV..25

Worldwide Sales and Service..26

1. USB Communication

The DGI USB device consists of two endpoints; one OUT Bulk endpoint for sending commands from the PC, and one IN Bulk endpoint for receiving responses. The endpoint size of both the IN and OUT endpoints must be taken into account when communicating with the device. A completed packet is recognized by the transfer having a length of less than the endpoint size. If a packet has a length which is a multiple of the endpoint size, the last transfer must be a zero length transfer to complete the packet.

All communication is initiated by the host computer sending a command packet over the OUT endpoint. All commands are given a response over the IN endpoint. Unrecognized commands will receive an error response.

Commands can have a maximum length of 256 bytes. Responses to most commands are only a few bytes. However, responses can be several thousand bytes when polling incoming data from the DGI buffer.

The USB device is a custom implementation, not following any predefined USB classes, and will therefore, require the installation of a driver on most systems.

2. Protocol

All values in the protocol are big endian. Command packets consists of a command byte, a 2-byte length and command-specific parameters. The length field only counts the bytes in parameters, and will be 0 for commands without parameters.

Table 2-1. Command Packet

Field	Size	Description
Command	1 byte	The identifier of the command to be executed.
Length	2 bytes	Amount of trailing data in parameters (n).
Parameters	n bytes	Command-specific parameters.

All received packets are parsed, executed, and responded to by a response code. The response packet consists of the received command byte, a status code, and parameters, depending on the command and status code. Unknown commands will receive an error response.

Table 2-2. Response Packet

Field	Size	Description
Command	1 byte	The identifier of the command that was executed.
Status Code	1 byte	Response indicating the status of the executed command.
Parameters	n bytes	Some status codes have parameters. The parsing of the parameters are command-specific.

Below is a list of the supported commands and possible response codes. Refer to the following sub-sections for details.

Table 2-3. List of Commands

Name	Value	Description
DGI_CMD_SIGN_ON	0x00	Initializes DGI and returns a verification string. Must be the first command called.
DGI_CMD_SIGN_OFF	0x01	Disconnects and stops all interfaces.
DGI_CMD_GET_VERSION	0x02	Returns the version of the DGI implementation.
DGI_CMD_SET_MODE	0x0A	Sets the operating mode of DGI.
DGI_CMD_INTERFACES_LIST	0x08	Lists all available interface identifiers.
DGI_CMD_INTERFACES_ENABLE	0x10	Used to enable/disable interfaces.
DGI_CMD_INTERFACES_STATUS	0x11	Fetches the status for the interfaces.
DGI_CMD_INTERFACES_SET_CONFIG	0x12	Sets the configuration for the specified interface.
DGI_CMD_INTERFACES_GET_CONFIG	0x13	Gets the configuration for the specified interface.
DGI_CMD_INTERFACES_SEND_DATA	0x14	Sends data for transmission over specified interface.

Name	Value	Description
DGI_CMD_INTERFACES_POLL_DATA	0x15	Returns the data buffer for the specified interface.
DGI_CMD_TARGET_RESET	0x20	Controls the state of the reset line of the target device.

Table 2-4. List of Responses

Name	Value	Description
DGI_RESP_OK	0x80	Verifies that the command was executed correctly.
DGI_RESP_FAIL	0x99	An error occurred during execution of the command. Usually caused by wrong usage of the protocol.
DGI_RESP_DATA	0xA0	The command was executed correctly and returned data. The data is command-specific and must be parsed accordingly.
DGI_RESP_UNKNOWN	0xFF	The received command identifier is unknown.

2.1 DGI_CMD_SIGN_ON

The sign on command is always the very first command to be called. It will initialize all states, buffers, and interfaces to a known starting point. A tool-specific string is returned as an acknowledgment of the sign on.

Table 2-5. Command Packet

Field	Size	Description
DGI_CMD_SIGN_ON (0x00)	1 byte	Command ID.
Length (0)	2 bytes	No parameters.

Table 2-6. Response Packet

Field	Size	Description
DGI_CMD_SIGN_ON (0x00)	1 byte	Command ID.
DGI_RESP_DATA (0xA0)	1 byte	Response code.
Length	2 bytes	Length, n, of the acknowledgment string.
String	n bytes	Acknowledgment string. • EDBG = “EDBG Data Gateway Interface” • Power Debugger = “Powerdebugger Data Gateway Interface” • Atmel-ICE = “Atmel-ICE Data Gateway Interface”

2.2 DGI_CMD_SIGN_OFF

The sign off command is the last command to be called. It will de-initialize all states, buffers, and interfaces.

Table 2-7. Command Packet

Field	Size	Description
DGI_CMD_SIGN_OFF (0x01)	1 byte	Command ID.
Length (0)	2 bytes	No parameters.

Table 2-8. Response Packet

Field	Size	Description
DGI_CMD_SIGN_OFF (0x01)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

2.3 DGI_CMD_GET_VERSION

This command gets the version number of the DGI implementation. The latest version at the time of writing is 3.1.

Table 2-9. Command Packet

Field	Size	Description
DGI_CMD_GET_VERSION (0x02)	1 byte	Command ID.
Length (0)	2 bytes	No parameters.

Table 2-10. Response Packet

Field	Size	Description
DGI_CMD_GET_VERSION (0x02)	1 byte	Command ID.
DGI_RESP_DATA (0xA0)	1 byte	Response code.
Major version	1 byte	Incremented only for big breaking changes.
Minor version	1 byte	Incremented for each change relevant to the protocol.

2.4 DGI_CMD_SET_MODE

This command changes the operating mode of certain aspects of the DGI mechanisms and protocol. Affected commands will have details in the command-specific sections.

Table 2-11. Command Packet

Field	Size	Description
DGI_CMD_SET_MODE (0x0A)	1 byte	Command ID.
Length (1)	2 bytes	
Mode	1 byte	Each bit corresponds to a specific setting. Default value is 0. • Bit 2 0: Use 2 bytes length for poll response.

Field	Size	Description
		1: Use 4 bytes length for poll response. • Bit 0 0: Poll response does not include overflow indicator. 1: Add buffer overflow indicator to poll response.

Table 2-12. Response Packet

Field	Size	Description
DGI_CMD_SET_MODE (0x0A)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

2.5 DGI_CMD_TARGET_RESET

This command sets the state of the reset line.

Table 2-13. Command Packet

Field	Size	Description
DGI_CMD_TARGET_RESET (0x20)	1 byte	Command ID.
Length (1)	2 bytes	
Reset state	1 byte	• Bit 0 0: Not asserted (released, pulled high by external pull-up). 1: Asserted (pulled low).

Table 2-14. Response Packet

Field	Size	Description
DGI_CMD_TARGET_RESET (0x20)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

2.6 DGI_CMD_INTERFACES_LIST

This command is used to discover the available interfaces on the tool. It will receive a list of all interfaces.

Table 2-15. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_LIST (0x08)	1 byte	Command ID.
Length (0)	2 bytes	No parameters.

Table 2-16. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_LIST (0x08)	1 byte	Command ID.
DGI_RESP_DATA (0x80)	1 byte	Response code.
Count (n)	1 byte	Number of interfaces in following list.
Interface list	n bytes	List of identifiers for available interfaces.

Related Links

Interfaces

2.7 DGI_CMD_INTERFACES_ENABLE

This command controls the OFF/ON state of the interfaces. A list of multiple interface states can be passed. The return code will stop execution and return a failure response for the first interface failing status update.

Table 2-17. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_ENABLE (0x10)	1 byte	Command ID.
Length (n×2)	2 bytes	
Interface ID	1 byte	Identifier of interface.
State to set	Possible values are: • 0: Off. • 1: On. • 2: On, timestamped.	

Table 2-18. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_ENABLE (0x10)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

Related Links

Timestamp

2.8 DGI_CMD_INTERFACES_SET_CONFIG

This command sets the configuration of an interface. See the interface-specific section for details.

Table 2-19. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_SET_CONFIG (0x12)	1 byte	Command ID.
Length (1+6×n)	2 bytes	
Interface ID	1 byte	Identifier of interface to set configuration for.
Config ID	2 bytes	Identifier of configuration parameter to set. Repeated n times.
Config value	4 bytes	Value of configuration parameter. Repeated n times.

Table 2-20. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_SET_CONFIG (0x12)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

2.9 DGI_CMD_INTERFACES_GET_CONFIG

This command gets the configuration of an interface. See the interface-specific section for details.

Table 2-21. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_GET_CONFIG (0x13)	1 byte	Command ID.
Length (1)	2 bytes	
Interface ID	1 byte	Identifier of interface to set configuration for.

Table 2-22. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_GET_CONFIG (0x13)	1 byte	Command ID.
DGI_RESP_DATA (0xA0)	1 byte	Response code.
Length	2 bytes	Length, n, of configuration.
Config ID	2 bytes	Identifier of configuration parameter to set. Repeated n times.
Config value	4 bytes	Value of configuration parameter. Repeated n times.

2.10 DGI_CMD_INTERFACES_POLL_DATA

This command polls data from the receive buffer of an interface. It needs to be called often to avoid overflow conditions in the device buffers. Only call this command for interfaces that are ON and do not have timestamping enabled. Data for interfaces using timestamped mode can be polled from the timestamp interface. See the timestamp section for details. The mode set command affects the response of this command.

Table 2-23. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_POLL_DATA (0x15)	1 byte	Command ID.
Length (1)	2 bytes	
Interface ID	1 byte	Identifier of interface to poll data from.

Table 2-24. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_POLL_DATA (0x15)	1 byte	Command ID.
DGI_RESP_DATA (0xA0)	1 byte	Response code.
Interface ID	1 byte	Interface ID as given in Table 3-1
Length (n)	2/4 bytes depending on mode set	Amount of data received.
Overflow indicator*	0/4 bytes	A non-zero value means an overflow has occurred. Only available if specifically set by a set mode command. Not included in the length field even if enabled.
Data	n bytes	Raw data that has been received from the interface.

Related Links

Timestamp

2.11 DGI_CMD_INTERFACES_SEND_DATA

This command sends data over the specified interface. The interface must be enabled first. Data is buffered and will be sent to the master at the clock speed determined by the configuration, or at the speed determined by the master of the physical interface (as for SPI, I^{2}C, and USART). The command will return true as long as the data buffer is free. If there is pending data in the send buffer already, this command will return a failure.

Table 2-25. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_SEND_DATA (0x14)	1 byte	Command ID.
Length (1+n)	2 bytes	n is limited to 250 bytes.
Interface ID	1 byte	Identifier of interface to send data to.
Data	n bytes	Data to send over interface.

Table 2-26. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_SEND_DATA (0x14)	1 byte	Command ID.
DGI_RESP_OK (0x80)	1 byte	Response code.

2.12 DGI_CMD_INTERFACES_STATUS

This command gets the status of all available subscriptions.

Table 2-27. Command Packet

Field	Size	Description
DGI_CMD_INTERFACES_STATUS (0x11)	1 byte	Command ID.
Length (0)	2 bytes	

Table 2-28. Response Packet

Field	Size	Description
DGI_CMD_INTERFACES_STATUS (0x11)	1 byte	Command ID.
DGI_RESP_DATA (0xA0)	1 byte	Response code.
Interface ID	1 byte	Identifier for interface. This byte is repeated for each interface.
Status	1 byte	Status of interface. • Bit 0: Started. • Bit 1: Timestamped mode. • Bit 2: Overflow occurred. This byte is repeated for each interface.

3. Interfaces

All functionality of the DGI is centered around the implemented interfaces. All interfaces use the same USB protocol, but every interface has its own configuration parameters and handling of communication. For details, refer to the interface-specific sections. Note that not all interfaces are available on all boards implementing the DGI device. The available interfaces can be read through the USB protocol.

Table 3-1. List of Interfaces

Name	Identifier	Description
Timestamp	0x00	Service interface which appends timestamps to all received events on associated interfaces.
SPI	0x20	Communicates directly over SPI in Slave mode.
USART	0x21	Communicates directly over USART in Slave mode.
I^2C	0x22	Communicates directly over I^2C in Slave mode.
GPIO	0x30	Monitors and controls the state of GPIO pins.
Power	0x40 (data) and 0x41 (sync events)	Receives data and sync events from the attached power measurement co-processors.
Reserved	0xFF	Special identifier used to indicate no interface.

Related Links

DGI_CMD_INTERFACES_LIST

3.1 Timestamp

The data returned over the timestamp interface is a sequential stream of timestamped packets of data belonging to the interfaces that have timestamping enabled. The first byte in each packet is the interface identifier and will decide how the rest of the packet must be parsed.

The timestamp is relying on a 16-bit timer, which is sampled and embedded into each packet. The timer tick frequency can be read from the timestamp configuration. It is in the area of about half a microsecond. When the timer overflows, a packet will be embedded in the stream to indicate this event. Note that if a data packet is being embedded as the timer overflows, an overflow packet will not be embedded. Instead, it will be indicated in the header of the data packet.

All timestamped packets are generated from module interrupts within the DGI device, which can not be interrupted by the timer overflow interrupt. This means that there is a possibility that the timer has overflowed before the timer was sampled and embedded. To be able to keep the timestamp in sync and accurate for such events the packets are also embedding the timer overflow bit. This bit is sampled after the timer itself, and can potentially be set even if the sampled timer value was in sync.

3.1.1 Parsing

The timestamp data is a buffer containing data from several interfaces in the order they were received. Each entry has the format as shown below.

Table 3-2. Data Format

Field	Size	Description
Interface ID	1 byte	Identifies the interface this entry is related to.
Interface specific data	n bytes	The length and interpretation of data is specific to the interface ID.

To handle the timestamp properly, declare a variable to accumulate the ticks as the timer overflows (hereby denoted by T_c). The timestamp of an entry (hereby denoted by T) is the sum of T_c and the timestamp tag of the entry (hereby denoted by T_t). Note that T is a value of ticks since sampling was started. The length of a tick can be found in the configuration section. While iterating the data coming from the timestamp interface, resolve the interface ID and handle the timestamp and interpretation of the data according to the details in the following sections.

Timestamp [0x00]

An entry with the timestamp interface ID is embedded for every overflow of the 16-bit timer. T_c should be incremented by $2^{16} = 65536$. The data section of this entry contains a counter that is incremented once for each entry of this type.

Table 3-3. Timestamp Specific Data

Field	Size	Description
Counter	1 byte	Incremented for each entry of this type.

SPI [0x20], USART [0x21], I^2C [0x22], GPIO [0x30]

For each character received over SPI, I^2C, USART, or a change on the GPIO lines, an entry is made in the timestamp buffer. The entry contains a sample of the 16-bit timer (T_t) which must be added to T_c to get T.

It also contains a timer overflow flag, which requires special consideration. If this byte is non-zero it means that the timer has overflowed during handling the current entry, and T_t must be examined to decide if it happened before or after the timestamp was sampled. If T_t is less than 256 or so, the overflow happened prior to sampling and T_c must be incremented by 2^{16} prior to calculating T. In the other case, T_t is greater than 256, T_c is incremented by 2^{16} after T has been calculated. The overflow flag is cleared after this, so no entry with the timestamp ID will be embedded in the stream for this overflow.

Table 3-4. Specific Data

Field	Size	Description
Timestamp	2 bytes	The 16-bit timer value.
Timer overflow flag	1 byte	If this is non-zero the timer has overflowed during the handling of this entry.
Data	1 byte	The received data.

Power sync [0x41]

An entry of this type is embedded for every 1000 samples, and gives the timestamp of the $n\times1000^{th}$ sample in the power stream. In this context, n denotes the count of entry occurrences of this type since the power interface was started. The structure of this entry is the same as described above for SPI, USART, etc. The data field contains a counter, which is incremented for each entry of this type.

Table 3-5. Specific Data

Field	Size	Description
Timestamp	2 bytes	The 16-bit timer value.
Timer overflow flag	1 byte	If this is non-zero the timer has overflowed during the handling of this entry.
Counter	1 byte	Incremented for each packet.

3.1.2 Configuration

The timestamp configuration contains the timer tick frequency (denoted by f_T) and prescaler (denoted by p). These values can be used to calculate a timestamp in seconds (denoted by t) from the tick count (denoted by T) by the formula $t = T\frac{p}{f_T}$.

Table 3-6. Configuration Parameters

Field	ID	Description
Prescaler	0	Prescaler, p, of the tick duration.
Frequency	1	Frequency, f_T, of the timer module.

3.2 GPIO

The GPIO interface consists of four lines available, which can be individually set to input or output through the configuration interface. This interface can only be used in Timestamp mode. Input lines are monitored and will trigger an entry to be added to the timestamp buffer on each change. Output lines can be controlled through the send data command.

3.2.1 Parsing

Each received data byte corresponds to an input pattern on the GPIO pins. If a bit is 1 it means that the corresponding GPIO pin is high, a 0 means a low level.

3.2.2 Configuration

The GPIO configuration controls the direction of the pins.

Table 3-7. Configuration Parameters

Field	ID	Description
Input pins	0	Setting a bit to 1 means the pin is monitored.
Output pins	1	Setting a bit to 1 means the pin is set to output and can be controlled by the send command.

3.3 SPI

The SPI interface is used for serial transfer of data. It operates in Slave mode, and requires the master to initiate all communication. Whenever the SPI master does a transfer, a character is added in the incoming buffer and a character is sent from the SPI send buffer. If no data is availabe in the send buffer it will send

0xFF as an idle character. This must be taken into consideration in the protocol to be able to distinguish idle characters from real data.

To avoid the risk of getting out of synchronization with the SPI transfer and starting receiving data in the middle of a character, the configuration allows for forcing the interface to wait for the chip select line to toggle before starting.

3.3.1 Parsing

The data received over the SPI interface is the raw data. No special handling is required.

3.3.2 Configuration

The SPI configuration controls the mode of transfer used.

Table 3-8. Configuration Parameters

Field	ID	Description
Character length	0	The number of bits in one character (5-8).
SPI mode	1	Sets the Transfer mode used for SPI: • 0: Clock idle low, sample on rising edge • 1: Clock idle low, sample on falling edge • 2: Clock idle high, sample on falling edge • 3: Clock idle high, sample on rising edge
Force CS sync	2	Setting this parameter will make the SPI interface wait for a chip select toggle before starting SPI transfers.

3.4 USART

The USART interface is used for serial transmission of data. It can operate in both Synchronous and Asynchronous modes. In Synchronous mode the USART works in Slave mode and the clock line must be supplied from an external master.

3.4.1 Parsing

The data received over the USART interface is the raw data. No special handling is required.

3.4.2 Configuration

The configuration sets the transfer parameters of the USART.

Table 3-9. Configuration Parameters

Field	ID	Description
Baud rate	0	The transfer speed of the interface in Asynchronous mode.
Character length	1	The number of bits of data in each character (5-8).
Parity type	2	The type of parity bit; 0 = Even, 1 = Odd, 2 = Space, 3 = Mark, 4 = None

Field	ID	Description
Stop bits	3	Count of stop bits used; 0 = 1 bit, 1 = 1.5 bits, 2 = 2 bits
Synchronous mode	4	If this is non-zero Synchronous mode will be used. Otherwise Asynchronous mode is used.

3.5 I^{2}C

The I^{2}C interface is used to transfer data serially. It operates in Slave mode and must, therefore, have a master connected to control the data flow.

Data is transmitted from the master by addressing the slave with the write flag. Then data can be sent byte by byte to the slave device.

To support transferring data from the PC to the I^{2}C master, the master has to poll the slave device regularly by sending the address with the read flag. The first byte read indicates the count of bytes of data waiting. If it is non-zero there is waiting data, and this data must be read out byte by byte in the same operation (no Stop or repeated Start).

3.5.1 Parsing

The data received over the I^{2}C interface is the raw data. No special handling is required.

3.5.2 Configuration

The I^{2}C configuration sets the operation parameters of the interface.

Table 3-10. Configuration Parameters

Field	ID	Description
Speed	0	The expected operation speed of the interface in Hertz helps the slave device adjust the timings. Up to 400 kHz is supported.
Address	1	Address of the slave device.

3.6 Power

The power interface is used to transfer power measurements and related data. It relies on a coprocessor that does the power measurements and transmits a stream of formatted data.

There are currently two flavors of the power measurement coprocessor, which are referred to as the XAM and the PAM. XAM is used on Xplained Pro boards that are embedding power measurement capabilities. PAM is used on the Power Debugger and offers a greater feature set. Look in the documentation for the Xplained Pro and Power Debugger for more details about the feature sets.

3.6.1 Parsing

The data from the power interface is a stream of packets of variable lengths. The upper two bits of the first byte of each packet describe the type and decide the rest of the interpretation. Below is a table of valid packet types.

Table 3-11. Packet Types

Type	ID (upper two bits)	Description
Primary sample	0b10	3-byte packet of A channel current sample.
Auxiliary sample	0b00	2-byte packet of B channel current sample and A and B channel voltage sample.
Notification	0b11	Notification packet for special events.
Reserved	0b01	

Notification

The notification packet is totally one byte long. It provides a way to give notifications about events.

Table 3-12. Notification Interpretation

Field	Bit position	Description
ID	7:6	Set to 0b11 for this packet.
Extended	5	Reserved for future use.
Type	4	If 0, data field contains an event, otherwise it is a sample rate.
Data	3:0	Type of event or sample rate.

Table 3-13. Events

Event	Value	Description
Sync tick	0	This event is embedded in the stream after every 1000 samples for PAM.

Primary sample

The primary sample packet contains a sample of the A channel current. It holds information about the range of the current sample, which must be used to index the correct calibration values.

Table 3-14. Primary Sample Interpretation

Field	Bit position	Description
ID	23:22	Set to 0b10 for this packet.
Range	21:20	For XAM the range field is used as an index for the calibration to be used. For PAM, 0 means low range and 1 means high range, while 2 and 3 means dummy sample and invalid sample, respectively, and the previous value should be used instead.
Sample rate	19:16	The sample rate can be ignored as this is constant for the current implementation. For XAM it is 16 kHz. For PAM it is 62.5 kHz.
Sample	15:0	The raw value of the sample is used together with the calibration data to calculate the actual current value.

Auxiliary sample

The auxiliary sample packet transmits A- and B-channel voltage and B-channel current. It is currently used only for PAM. Whenever such a packet is received it should be timestamped with the same timestamp as the latest primary packet.

Table 3-15. Auxiliary Sample Interpretation

Field	Bit position	Description
ID	15:14	Set to 0b00 for this packet.
Channel	13:12	0 = B Current, 1 = B Voltage, 2 = A Voltage.
Sample	11:0	For voltage data divide the raw value by -200 to get the measured voltage. For current data use the calibration values.

3.6.2 Configuration

The power configuration consists of a generic section that can be used to control the power measurement and related functionality. There is also a section that is specific to the type of power coprocessor used. The specific section contains the calibration that is required to interpret the incoming data and get the correct output values.

Table 3-16. Generic Configuration Parameters

Field	ID	Description
Type	0	Type of coprocessor for power measurement. • XAM = 0x10 • PAM = 0x11
Channel	1	Setting a bit to 1 will activate the related channel; bit 0=A, bit 1=B.
Calibrate	2	For XAM, any write will trigger calibration. For PAM, the value will decide the type of calibration performed; 2 = Reset to factory, 8 = A channel, 9 = B channel.
Lock range	3	For PAM, setting to 1 will lock the A channel in high range. For XAM, not implemented.
Output voltage	4	For PAM, sets the output voltage of the target supply to the given value in mV. For XAM, not implemented.

XAM

The XAM coprocessor calibration starts at parameter ID 10. It consists of four blocks of configuration parameters, one for each supported range. The blocks have the structure as described in the table below. Note that 'N' refers to the range index.

Table 3-17. XAM Range Calibration Parameters

Field	ID	Type	Description
Token	N×12 + 10	uint16	The lower byte contains the id of the range on the form 0xnn (n = N+1). Note that the sample field in the data packet uses 0-indexing, where this field uses 1-indexing. The upper byte identifies the state of the calibration: • 0 = Uncalibrated • 1 = Factory calibrated • 2 = User calibrated
Offset correction	N×12 + 13	uint16	Subtract this value from the raw value to correct offset error.

Field	ID	Type	Description
Float gain correction	N×12 + 14	float	Multiply the offset corrected raw value with this gain correction.
μA Resolution	N×12 + 20	float	Resolution in μA. Factor to multiply the corrected raw value with to get the current in μA.

PAM

The PAM coprocessor, having a wider range, has a more complex model for calibration. To read the PAM calibration it is necessary to read the entire block of parameter values from 10 to 175 into a buffer. The buffer can then be parsed into the following sections:

The 32-byte header:

Table 3-18. PAM Calibration Header

Field	Offset	Type	Description
Format	0	uint8	Calibration format version. Set to 2 for this version.
Data invalidation	1	uint8	Is set to zero to indicate successful calibration.
Local A-channel calibration	2	int16	Contains an averaged offset value for the high range of channel A, as used internally by the PAM.
User calibration flag	4	uint8	Indicates whether the calibration data is the result of a factory (0) or user (1) calibration.

The 512-byte **A-Channel** calibration parameters:

The 'N' parameter represents the eight voltage ranges on the PAM, as determined by this formula: $V = 1.6 + \frac{N}{1.8}$. Use the calibration point closest to the actual voltage reported by the tool.

Table 3-19. PAM CHA Calibration Parameters

Field	Offset	Type	Description
Format	0	uint8	
Data invalidation	1	uint8	Is set to zero to indicate successful calibration.
High range calibration data	N×60 + 8	See below	Contains the calibration data for voltage N for the high range of the A-channel as described below.
Low range calibration data	N×60 + 22	See below	Contains the calibration data for voltage N for the low range of the A-channel as described below.

For each calibration voltage, the PAM A-channel calibration data contains a two-segment linearization for each range. The top segment of the low range and the low segment of the high range uses the same calibration currents and helps ensure a good transition between the ranges.

Table 3-20. PAM CHA Calibration Data

Field	Offset	Type	Description
High level offset	0	int16	
High level gain	2	float	

Field	Offset	Type	Description
Crosspoint	6	int16	The cross-point indicates the raw data value where the high and low segments meet.
Low level offset	8	int16	
Low level gain	10	float	

The **B-Channel** calibration data is contained in the remainder of the calibration:

The 'N' parameter represents the eight voltage ranges on the PAM.

Table 3-21. PAM CHB Calibration Parameters

Field	Offset	Type	Description
Format	0	uint8	
Data invalidation	1	uint8	Is set to zero to indicate successful calibration.
Offset	N×14 + 2	int16	
Gain	N×14 + 4	float	

4. Revision History

Doc Rev.	Date	Comments
B	10/2017	Added missing Interface ID byte in DGI_CMD_INTERFACES_POLL_DATA response
A	07/2017	• New document template • New document number: Microchip version DS40001905A replaces Atmel version 32223A • Corrected Calibrate field Description in Power Interface Configuration section • Updated Range field Description in Primary sample section in Power Parsing section
32223A	09/2016	Initial document release

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under “Support”, click on “Customer Change Notification” and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-2227-3

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-3326-8000
Fax: 86-21-3326-8021

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

France - Saint Cloud
Tel: 33-1-30-60-70-00

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820