|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length: | 1 | 2 | 2 | 2 | n-7 |
| Field: | Key | Checksum | Length | ERROR | Command |

Key: The key byte is always equal to 0x13

CHECKSUM: This field is equal to the hex values of the key + packet length + error code + command values. This summation then goes through the AND process with 0xFFFF.

Length: The total number of bytes in the packet.

ERROR: Contains the various error codes (refer to the ERROR description section below).

For example, if one wanted to send command {0x01, 0x0A} to the device, the packet would resemble the following:

Key = 0x13

CHECKSUM = hex values of (key + packet length + ERROR code + Command[0] + Command[1]) AND 0xFFFF

CHECKSUM = (0x13 + 0x09 + 0x00 + 0x00 + 0x00 + 0x01 + 0x0A) AND 0xFFFF = 0x27

Length = key length + checksum length + length field + error length + command length = 1+2+2+2+2 = 0x09

ERROR = ERROR\_OK = 0x0

Command = 0x01 0x0A

The packet would be saved in an array of bytes, as shown below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Key | CHECKSUM | Length | ERROR | Command |
| 0x13 | 0x27 0x00 | 0x09 0x00 | 0x00 0x00 | 0x01 0x0A |

Before the packet is sent anywhere, it needs to be converted into a valid hex string. The string for the example packet used above would read 13270009000000010A.

**Definitions of command variables:**

Digital = 0x1

Analog = 0x2

SPI = 0x3

EEPROM = 0x4

write = 0x5

read = 0x6

write/read = 0x7

configure = 0x8

PWM = 0x9

Firmware\_Protocol = 0xA

I2C=0XB

Input = 0x0

Output =0x1

High =0x1

Low =0x0

DEFAULT = 0x1

EXTERNAL = 0x3**Digital Operations:**

Digital configure: The configure command has the ability to set a specified digital pin to act as either input or output. To configure multiple pins at once, append each pin number followed with a byte that describe whether you want to set the pin as input or output, as described in the command section.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | 1 | 1 |  |
| Field: | Digital | Configure | PIN\_number\_1 | Input/output\_1 | PIN\_number\_2 | Input/output\_2 | ----- |

|  |  |  |
| --- | --- | --- |
| Length: | 1 | 1 |
| Field: | PIN\_number\_n | Input/output\_n |

Digital configure reply: The digital configure reply sends a reply with no command field, you need to check the ERROR field. See the error code section below.

Digital read request: The digital read request sends a read command to a specified digital pin. To read from multiple pins at once, append each desired pin number in succession.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 |  | 1 |
| Field: | Digital | Read | PIN\_number\_1 | PIN\_number\_2 | -------- | PIN\_number\_n |

Digital read reply: The digital read reply returns a value of either 1 for high or 0 for low.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length: | 1 | 1 |  | 1 |
| Field: | VALUE\_1 | VALUE\_2 | ------- | VALUE\_n |

Digital write request: The digital write request sends a value of either 1 or 0 (high or low respectively) to a specified digital pin. To configure multiple pins at once, append each pin number followed with a byte that describes whether you want to set the pin as high or low.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | 1 | 1 |  |
| Field: | Digital | Write | PIN\_number\_1 | Value\_1 | PIN\_number\_2 | Value\_2 | -------- |

|  |  |  |
| --- | --- | --- |
| Length: | 1 | 1 |
| Field: | PIN\_number\_n | Value\_n |

Digital write reply: Digital write reply sends a reply with no command field, and checks the ERROR field. See the error code section below.**Analog Operations:**

Analog configure: This packet contains the analog reference voltage value, configured as either the default voltage for your device, or an external voltage.

|  |  |  |  |
| --- | --- | --- | --- |
| Length: | 1 | 1 | 1 |
| Field: | Analog | Configure | DEFAULT/EXTERNAL |

Analog read request: An analog read request requests a value to be read from a specified analog pin. To read from multiple pins in a single transaction, append each desired pin number in succession.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 |  | 1 |
| Field: | Analog | Read | PIN\_number\_1 | PIN\_number\_2 | ------- | PIN\_number\_n |

Analog read reply: This packet contains the read value stored on a specified analog pin and is allocated 2 bytes. If a read request queries multiple pins simultaneously, the read reply packet will have a pin number followed by its read value, and then the next pin number followed by its respective value.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 2 | 1 | 2 |  |
| Field: | Analog | PIN\_number\_1 | Read\_value\_1 | PIN\_number\_2 | Read\_value\_2 | ------ |

|  |  |  |
| --- | --- | --- |
| Length: | 1 | 2 |
| Field: | PIN\_number\_n | Read\_value\_n |

**SPI Operations:**

SPI configure: This packet contains the settings for configuring SPI communication. The data fields include the bit order (most or least significant bit first), clock divider (2, 4, 6, 8, 16, 32, 64 or 128), mode (0, 1, 2 or 3) and the chip select pin number.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | 1 | 1 |
| Field: | SPI | Configure | Bit\_Order | Clock\_Divider | set\_SPI\_mode | nCS\_Slave\_Pin |

Bit\_Order : {LSBFIRST = 0x0 or MSBFIRST = 0x1 }

Clock\_Divider : { SPI\_CLOCK\_DIV2 = 0x2 , SPI\_CLOCK\_DIV4 = 0x4 , SPI\_CLOCK\_DIV8 = 0x8 , SPI\_CLOCK\_DIV16 = 0x10 , SPI\_CLOCK\_DIV32 = 0x20 , SPI\_CLOCK\_DIV64 = 0x40 or SPI\_CLOCK\_DIV128 = 0x80}

Data\_Mode : { SPI\_MODE0 =0x0 , SPI\_MODE1 = 0x1 , SPI\_MODE2 = 0x2 , SPI\_MODE3 =0x3}

nCS\_Slave\_Pin = pin that selects the slave SPI device

SPI write/read:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 2 | SPI\_Data\_Length\_in\_bytes |
| Field: | SPI | write\_Read | SPI\_Data\_Length\_in\_bytes | Data\_Written\_to\_SPI\_BUS |

SPI write/read reply:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 2 | SPI\_Data\_Length\_in\_bytes |
| Field: | SPI | write\_Read | SPI\_Data\_Length\_in\_bytes | Data\_Read\_to\_SPI\_BUS |

**PWM Operations:**

PWM configure: For a PWM transaction, one needs to set a specified pin as output first, by the same process as with a digital configuration.

PWM write: This packet sends a value ranging from 0 to 255 to a specified PWM pin. To write to multiple pins at once, append each pin number followed with a value.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | 1 |  | 1 | 1 |
| Field: | PWM | PIN\_number\_1 | Value\_1 | PIN\_number\_2 | Value\_2 | ---- | PIN\_number\_n | Value\_n |

PWM write reply: The device sends a reply with no command field, and checks the ERROR field. See the error code section below.

**EEPROM Operations:**

EEPROM read request:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 |
| Field: | EEPROM | Read | Start\_address | Length |

EEPROM read reply:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | n |
| Field: | EEPROM | Read | Start\_address | Length | Data |

EEPROM write:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | n |
| Field: | EEPROM | Write | Start\_address | Length | Data |

**I2C operations TBD**

I2C write request:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length | 1 | 1 | 1 | 1 | n |
| Field | I2C | write | DEVICE\_ADDRESS | Number\_of\_bytes\_to\_write = n | DATA |

I2C write Reply:

|  |  |  |  |
| --- | --- | --- | --- |
| Length | 1 | 1 | 1 |
| Field | I2C | write | DEVICE\_ADDRESS |

You need to check the Error field for errors

I2C read request:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length | 1 | 1 | 1 | 1 |
| Field | I2C | read | DEVICE\_ADDRESS | Number\_of\_bytes\_to\_read = n  If n=0, the device returns the number of available bytes |

I2C read reply

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length | 1 | 1 | 1 | 1 | n |
| Field | I2C | read | DEVICE\_ADDRESS | Number\_of\_bytes\_to\_write = n | DATA |

**Firmware Protocol request operations:**

Firmware Protocol Query:

|  |  |
| --- | --- |
| Length: | 1 |
| Field: | Firmware\_Protocol |

Firmware Protocol Reply:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length: | 1 | 1 | 1 | 1 | 1 |
| Field: | Firmware\_Protocol | Firmware \_Major\_Ver | Firmware \_Minor\_ver | Firmware \_Major\_Ver | Firmware \_Minor\_ver |

**ERROR Descriptions:**

The error field describes whether a transaction has succeeded or not, and the error number refers to the description of the error.

|  |  |  |
| --- | --- | --- |
| **ERROR** | **Value** | **Description** |
| ERROR\_OK | 0X0 | No error reported |
| ERROR\_CHECKSUM\_WRONG | 0X1 | Wrong checksum |
| ERROR\_WRONG\_CHARACTER\_FOUND | 0X2 | An invalid character was entered |
| ERROR\_WRONG\_PACKET\_LENGTH | 0X3 | The actual length did not match the expected length |
| ERROR\_WRONG\_KEY | 0X4 | The key doesn’t match the expected value |
| ERROR\_UNKNOWN\_COMMAND | 0X5 | The command was entered incorrectly |