Table 1 CMD packet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | CMD | Uint32 |  | Table 6 Commands summary |
| 4 | [Data 0] | Uint8 |  |  |
|  | […] | Uint8 |  |  |
| n+4 | [Data n] | Uint8 |  |  |

Table 2 ACK packet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 0 | Fixed value |
| 2 | Error/state | Uint16 | 0 | 0 = No error (ack); (Mainly as padding) |
| 4 | CMD | Uint32 |  | Original CMD |
| 8 | [Data 0] | Uint8 |  |  |
|  | … | Uint8 |  |  |
| n+8 | [Data n] | Uint8 |  |  |

Table 3 ID packet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 1 | Fixed value |
| 2 | Error/state | Uint16 | 0 | (Mainly as padding) |
| 4 | MCU HW ID | Uint16 |  |  |
| 6 | MCU HW version Major | Uint8 |  |  |
| 7 | MCU HW version Minor | Uint8 |  |  |
| 8 | MCU serial | Uint32 |  |  |
| 12 | CPU\_UID | Uint32[3] |  |  |
| 24 | ADC HW ID | Uint16 |  |  |
| 26 | ADC HW version Major | Uint8 |  |  |
| 27 | ADC HW version Minor | Uint8 |  |  |
| 28 | ADC serial | Uint32 |  |  |
| 32 | FW ID | Uint16 |  |  |
| 34 | FW version Major | Uint8 |  |  |
| 35 | FW version Minor | Uint8 |  |  |
| 36 | FW configuration | Char[8] | „RELEASE\0“ |  |
| 44 | Build time | Char[30] | „2025-05-13T12:43:13\0“ | ISO 8601, Includes padding |
| 74 | Channels count | Uint16 | n | Uint8 is enough, but padding |
| 76 | Channel 0 unit | Char[4] | „mV\0“ | Includes padding |
| 80 | Channel 0 offset (q) | float |  | (4 bytes) y = k×x+q |
| 84 | Channel 0 gain (k) | float |  | (4 bytes) |
| 88 | Channel 1 unit | Char[4] | „A\0“ | Includes padding |
| 92 | Channel 1 offset (q) | float |  | (4 bytes) |
| 96 | Channel 1 gain (k) | float |  | (4 bytes) |
|  | […] |  |  |  |
| 76+12×n | CRC | Uint16 |  | CCITT |

Table 4 Data packet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 2 | Fixed value |
| 2 | Packet num | Uint16 | 0 – 65535 | Only data packets |
| 4 | Data[Ch 0][0] | Int16 |  |  |
|  | … | Int16 |  |  |
| 402 | Data[Ch 0][199] | Int16 |  |  |
| 404 | Data[Ch 1][0] | Int16 |  |  |
|  | … | Int16 |  |  |
| 2+400×n | Data[Ch n][199] | Int16 |  |  |
|  | Parity errors[Ch 0] | Uint8 |  |  |
|  | … | Uint8 |  |  |
|  | Parity errors[Ch n] | Uint8 |  |  |
|  | [Padding] | Uint8[m] | 0 | m = n % 2 |
| 4+401×n+m | CRC | Uint16 |  | CCITT |

Table 5 Trigger packet (I get trigger)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 3 | Fixed value |
| 2 | Packet num | Uint16 | 0 | Data packet num in which trigger occurred |
| 4 | Sample num | Uint8 |  | Sample num in packet |

Table 6 Commands summary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Command** | **CMD** | **Data** | **Response** | **Explanation** |
| Ping | 0 | None | ACK | Check device / get status |
| Get ID | 1 | None | ID | Get device ID |
| Register receiver | 2 | IP, Port | ACK + IP, Port, n | Setup endpoint for data packets, return order  Table 7 Register receiver command  Table 8 Response to Register receiver command |
| Remove receiver | 3 | IP, Port | ACK | Remove endpoint for data packets  Table 9 Remove receiver command |
| Get receivers | 4 | None | ACK + n×(IP, Port) | Get registered receivers  Table 10 Response to Get receivers command |
| Start sampling | 5 | n | ACK + n | Acquire and send n samples (0 for endless; reset packet counter)  Table 11 Start sampling [on trigger] command  Table 12 Response to Start sampling [on trigger] command |
| Start sampling on trigger | 6 | n | ACK + n | Wait for trigger and start sampling as after Start sampling command  Table 11 Start sampling [on trigger] command  Table 12 Response to Start sampling [on trigger] command |
| Stop sampling | 7 | None | ACK + n | Stop sampling, return number of acquired samples  Table 13 Response to Stop sampling command |
| Trigger ACK | 8 | None | None | Acknowledge trigger packet, otherwise trigger packet considered lost and resend |
| Forse trigger (from client) | 9 | None | Trigger packet |  |

Table 7 Register receiver command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | CMD | Uint32 | 2 |  |
| 4 | IP | Uint8[4] |  | 0.0.0.0 for CMD packet source IP |
| 8 | Port | Uint16 |  | 0 for CMD packet source port |

Table 8 Response to Register receiver command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 0 | Fixed value |
| 2 | Error/state | Uint16 | 0 | 0 = No error (ack); (Mainly as padding) |
| 4 | CMD | Uint32 | 2 | Original CMD |
| 8 | IP | Uint8[4] |  | IP of previously registered receiver |
| 12 | Port | Uint16 |  | Port of previously registered receiver |
| 13 | n | Uint8 |  | Order of the receiver in a list of registered receivers |

Table 9 Remove receiver command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | CMD | Uint32 | 3 |  |
| 4 | IP | Uint8[4] |  | IP of previously registered receiver |
| 8 | Port | Uint16 |  | Port of previously registered receiver |

Table 10 Response to Get receivers command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 0 | Fixed value |
| 2 | Error/state | Uint16 | 0 | 0 = No error (ack); (Mainly as padding) |
| 4 | CMD | Uint32 | 4 | Original CMD |
| 8 | [IP[0]] | Uint8[4] |  | IP of previously registered receiver |
| 12 | [Port[0]] | Uint16 |  | Port of previously registered receiver |
|  | […] |  |  |  |
| 8+n×6 | [IP[n]] | Uint8[4] |  | IP of previously registered receiver |
| 12+n×6 | [Port[n]] | Uint16 |  | Port of previously registered receiver |

Table 11 Start sampling [on trigger] command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | CMD | Uint32 | 5 / 6 |  |
| 4 | n | Uint64 |  | How many samples to acquire, 0 for endless sampling |

Table 12 Response to Start sampling [on trigger] command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 0 | Fixed value |
| 2 | Error/state | Uint16 | 0 | 0 = No error (ack); (Mainly as padding) |
| 4 | CMD | Uint32 | 5 / 6 | Original CMD |
| 8 | n | Uint64 |  | For check |

Table 13 Response to Stop sampling command

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Byte offset** | **Field** | **Type** | **Example** | **Note** |
| 0 | Packet type | Uint16 | 0 | Fixed value |
| 2 | Error/state | Uint16 | 0 | 0 = No error (ack); (Mainly as padding) |
| 4 | CMD | Uint32 | 7 | Original CMD |
| 8 | n | Uint64 |  | How many samples was acquired |

Table 14 Typical workflow

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Central node / PC**  **(client)** | **Measuring node**  **(server)** | **Note** |
| 1 | Sends Get ID command |  |  |
| 2 |  | Responds with ID packet |  |
| 3 | Sends Register receiver command with own IP and listening port |  |  |
| 4 |  | Responds with ACK packet |  |
| 5 | Sends Start sampling command |  |  |
| 6 |  | Responds with ACK packet |  |
| 7 |  | Sends Data packet to registered receivers |  |
| 8 | Process received data | Sends Data packet to registered receivers |  |
| 9 | Process received data |  |  |
| … | […] rest is optional |  |  |
| n | Sends Stop sampling command | Sends Data packet to registered receivers |  |
| n+1 |  | Responds with ACK packet |  |

Table 15 Trigger workflow

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Central node / PC**  **(client)** | **Measuring node**  **(server)** | **Note** |
| 1 | Send trigger (CMD 9) |  | Or HW trigger |
| 2 |  | Send trigger packet |  |
| 3 |  | If start on trigger is true, send data |  |
| 4 | Send trigger ACK |  |  |
| 5 |  | Check ACK trigger, false => send trigger packet, Timeout 10x => ERR timeout |  |
| 6 | Trigger packet timeout => send trigger cmd. Timeout 2x => ERR timeout |  |  |

**Little endian**

**UDP\_CMD\_PORT 10578**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **GRAF** | | | | | | | |
| **Error data** | **X range** | | **Y min** | **Y max** | **Auto range** | | **clean** |
| **Adresa generátoru taxtové pole** | | **GET ID** | | **Save data** | | **Systémové zprávy** | |
| **Potvrdit** | | **Registrer receiver textové pole** | | **Výpis cesty kam ukládám** | |
| **Adresa klienta textové pole** | | **Register receiver** | | **Start sampling** | |
| **Potvrdit** | | **Remove receiver textové pole** | | **Start sampling on trigger** | |
| **PING** | | **Remove receiver** | | **Stop sampling** | |
|  | | **Get receivers** | |  | |