**Communication Arduino to Raspberry PI**

Raspberry PI is the master and initiates all communication. Arduino is the slave and responds to commands.

|  |  |  |
| --- | --- | --- |
| **Command ID** | **Parameters** | **Description** |
| **Status info:** | | |
| 0x01 | - | Error status (TBD) |
| 0x02 | - | Get Command queue length (ready to accept commands) |
| 0x03 | - | Reset all peripherals |
| 0x04 | - | Test sequence |
|  |  |  |
| **Motor Control:** | | |
| 0x11 | DIR SPEED | Set left motor speed and direction |
| 0x12 | DIR SPEED | Set right motor speed and direction |
|  |  |  |
| **Power Information:** | | |
| 0x21 | - | Get battery current |
| 0x22 | - | Get system current |
| 0x23 | - | Get battery voltage |
| 0x24 | - | Get system voltage |
|  |  |  |
| **Turret Control:** | | |
| 0x31 | ANGLE | Set turret horizontal angle (max 360) |
| 0x32 | ANGLE | Set turret vertical angle (max 360) |
| 0x33 | - | Fire missile from turret 1 |
| 0x34 | - | Fire missile from turret 2 |
| 0x35 | - | Fire all missiles turret 1 |
| 0x36 | - | Fire all missiles turret 2 |
| 0x37 | - | Fire all missiles both turret |
| 0x38 | ON/OFF | Turn laser on/off |
| 0x39 |  | Target locked |
|  |  |  |
| 0x41-0xFF |  | RESERVED |
| **Configdata** |  |  |
| 0x41 | Mode | Change control mode (autotarget+manualdrive vs autodrive) |
| 0x42 | MaxSpeed | set the maximal speed of the system |
| 0x43 | E\_Mode | Set energy mode of the system |

**Packet Structure:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Start byte  0x5A | Length  (1 byte) | Command ID (1 byte) | Data (Fixed length) | Checksum (1 byte) | Stop byte  0xA5 |

Checksum is simple XOR of Command ID and Data.

**Arduino to Raspberry Pi communication**

UART communication will be used for communication between raspberry and Arduino.

UART is chosen as it is possible to spoof the Raspberry from a laptop (and vice versa) in order to speed up the design of the system as a whole; easier to achieve modularity between the two parts. Additionally it natively supports bi-directional communication.

APIs

**Path recognition**

* Provides
  + FindPath(parameters)
    - Parameters: Angle, DistanceLeft, DistanceRight, PathWidth
* Requires
  + image data pointer
    - parameters: Column, height, width, filter?

**Image/Sign detection**

* Provides
  + Sign(parameters)
    - Parameters: SignType, distance
* Requires
  + image data pointer
    - parameters: Column, height, width, filter?

**Control Loop**

* Provides
* Requires
  + FindPath(parameters)
  + Sign(parameters)
  + GetCmd()
  + SetCmd(parameters)

**Command Encoder**

* Provides
  + GetCmd() returns: (decodes packets to: )
    - Cmd\_ID
    - Parameters of command
  + SetCmd(parameters)
    - Parameters: Cmd\_ID, parameters of command (encodes to packets)
* Requires
  + PushCmd()
  + PopCmd()

**Command Communication (has buffer)**

* Provides
  + PushCmd(parameters) (Queues commands to be send)
    - See packet structure
  + PopCmd() (stores commands to be send)
* Requires
  + SendUART()
  + ReceiveUART()
  + SendTCP()
  + ReceiveTCP()

**Networking (command override)**

* Provides
  + SendTCP()
  + ReceiveTCP()
* Requires
  + Python TCP implementation

**UART (commands)**

* Provides
  + SendUART()
  + ReceiveUART()
* Requires
  + Python UART implementation