# iAcommsDriver

## Configuration Parameters

|  |  |  |
| --- | --- | --- |
| variable | type | purpose |
| PortName | String | Serial port name |
| ID | integer | acomms ID for addressing packets |
| PSK\_minipackets | boolean | Use fsk or psk mini packets |
| Community (global) | string | get vehicle name |

## Subscriptions

|  |  |  |
| --- | --- | --- |
| variable | type | purpose |
| ACOMMS\_TRANSMIT\_DATA | String | Passing ascii data to driver for transmission |
| ACOMMS\_TRANSMIT\_DATA\_BINARY | Binary string | Passing binary data to driver for transmission |
| ACOMMS\_TRANSMIT\_RATE | Double | Integer rate |
| ACOMMS\_TRANSMIT\_DEST | Double | Integer ID of destination (0 for broadcast) |
| NAV\_X, NAV\_Y | double | Used for posting of range pulses |
| LOGGER\_DIRECTORY | string | to put log file into same directory as pLogger |

## Publications

|  |  |  |
| --- | --- | --- |
| variable | type | purpose |
| ACOMMS\_RECEIVED\_DATA | Binary string | Data received in a transmission |
| ACOMMS\_RECEIVED\_DATA\_HEX | String | Received data in hex format |
| ACOMMS\_TRANSMITTED\_DATA\_HEX | string | Transmitted data in hex format |
| ACOMMS\_RECEIVED\_ALL | string | DebugString of received ModemTransmission protobuf |
| ACOMMS\_RECEIVED\_SIMPLE | String | Brief summary of reception |
| ACOMMS\_TRANSMIT\_SIMPLE | string | Brief summary of transmission |
| ACOMMS\_DRIVER\_STATUS | string | status of driver, updated every 5 seconds |
| ACOMMS\_DRIVER\_WARNING | String | For debugging information |
| VIEW\_RANGE\_PULSE | string | Posting of range pulses on transmission or reception |
| ACOMMS\_IMPULSE\_RESPONSE | string | Raw CAIRE message from modem |
| ACOMMS\_SNR\_OUT, ACOMMS\_SNR\_IN, ACOMMS\_DQR | double | Data picked from ACOMMS\_RECEIVED\_ALL for ease of access by other applications |

## Basic Usage

### Driver status

The driver will publish its current status to ACOMMS\_DRIVER\_STATUS at least once every 5 seconds. Status can be “transmitting”, “receiving”, “ready”, or “not running” (only occurs at startup). Transmission requests will be ignored if the driver is not ready.

### Transmitting

Set the destination and rate in advance using the ACOMMS\_TRANSMIT\_RATE and ACOMMS\_TRANSMIT\_DEST variables. Rate 100 is used for mini-packets and a destination of 0 is used for broadcasts. Initiate a transmission by posting to ACOMMS\_TRANSMIT\_DATA or ACOMMS\_TRANSMIT\_DATA\_BINARY. You must use the binary variable if your data contains the byte 0x00. Data will automatically be packaged into frames according to the set rate and truncated if necessary (see micromodem documentation). The driver will post a hex translation of the transmitted data to ACOMMS\_TRANSMITTED\_DATA\_HEX and a brief summary of the transmission information will be posted to ACOMMS\_TRANSMIT\_SIMPLE. A yellow range pulse is posted when transmitting.

### Receiving

All receptions should be accompanied by a posting to ACOMMS\_RECEIVED\_ALL containing all receive information, including statistics. A brief summary will be posted to ACOMMS\_RECEIVED\_SIMPLE. If data was received, it will be posted to ACOMMS\_RECEIVED\_DATA as binary and ACOMMS\_RECEIVED\_DATA\_HEX as a hex translation. Note that there will be no indication in the received data if individual frames were lost – currently you must check ACOMMS\_RECEIVED\_ALL or ACOMMS\_RECEIVED\_SIMPLE.

The raw impulse response message from the modem is caught and posted to ACOMMS\_IMPULSE\_RESPONSE, primarily for logging purposes. Individual statistics can be posted as their own variables for ease of use. Currently snr\_in, snr\_out, and dqr are posted individually.

## Message Formats

ACOMMS\_RECEIVED\_ALL is created by calling the DebugString() method on the ModemTransmission protobuf. Line endings are replaced with the placeholder “<|>”.

ACOMMS\_TRANSMIT\_SIMPLE and ACOMMS\_RECEIVED\_SIMPLE are defined in lib\_acomms\_messages.

Hex formatted messages use colon delimiters between bytes. For example the phrase “Hello world” would be posted as “48:65:6c:6c:6f:20:77:6f:72:6c:64”. Hex values less than 10 will be posted using one digit instead of two (e.g. “61:0:61”).

## Minipackets (rate 100)

Minipackets can carry 13 bits of information passed in two bytes. The micromodem will always perform a bitwise and with 0x1f on the first byte. If only a single byte is passed to the driver for transmission, it will be packed with 0x00 in the first position. See the following examples:

acomms\_transmit\_data\_binary --> acomms\_received\_data  
a) 0x6161 --> 0x0161  
b) 0x0061 --> 0x0061  
c) 0x6100 --> 0x0100  
d) 0x61 --> 0x0061

ACOMMS\_TRANSMITTED\_DATA\_HEX can be used to check the data actually being transmitted in a minipacket.

# Lib\_acomms\_messages

Library used for passing acomms related messages containing multiple pieces of information.

## SIMPLIFIED\_RECEIVE\_INFO

### Fields

|  |  |  |
| --- | --- | --- |
| field | type | description |
| Vehicle name | String | Name of the vehicle that sent the transmission |
| Source | Integer | Source id of the transmitter |
| Rate | Integer | Transmission rate (100 for mini) |
| Num frames | Integer | Total number of expected frames |
| Num good frames | Integer | Number of frames correctly received |
| Num bad frames | Integer | Number of frames with errors |

### Format

“vehicle\_name,%s:source,%d:rate,%d:num\_frames,%d:num\_good\_frames,%d:num\_bad\_frames,%d”

## SIMPLIFIED\_TRANSMIT\_INFO

### Fields

|  |  |  |
| --- | --- | --- |
| field | type | description |
| Vehicle name | String | Name of the vehicle that sent the transmission |
| Rate | Integer | Transmission rate (100 for mini) |
| Dest | integer | Destination ID (0 for broadcast) |
| Num frames | Integer | Total number of frames sent |

### Format

“vehicle\_name,%s:rate,%d:dest,%d:num\_frames,%d”