# MGL Avionics A14 / A16 Aircraft Audio intercom system communications protocol.

# Document version 1.0 dated 11 July 2020

## General

The A14 and A16 systems share the same protocol. Differences pertain only to the physically available inputs and outputs.

The systems can be controlled via a bi-directional RS232 interface or CAN bus. In both cases the content of the data sent and received is identical, the only difference being framing of the data packets.

The RS232 link runs at 115200 baud, 8 data bits, no parity, one stop bit. Data packets are framed using the GDL-90 protocol frame. This is not described as part of this document as the details are readily available on the internet. GDL-90 is used mostly for ADS-B data transfer and should be familiar to avionics engineers.

CAN bus is limited to a maximum of 8 bytes of data per CAN packet. 29Bit CAN addresses are used. The CAN address field is used to encode protocol information allowing up to 8 CAN packets to form a single message for a maximum data size of 64 bytes.

## A14/A16 basics

The intercom system looks like a 16x8 audio crosspoint switch. A total of 16 inputs can be arbitrarily switched onto 8 outputs. 4 of these outputs are dedicated to a stereo pair for each pilot and Pax output circuit. Two outputs are routed to COM radios (for TX audio), one for an AUX output (intended for cockpit voice recorders etc) and an output to the bluetooth module.

Generally all inputs can be switched to either left or right channel of a stereo output – exceptions are wired music and bluetooth input which is also stereo – here switching either left or right has the corresponding effect on the other channel as well.

The wired music input can be used either as combined stereo input or as two independent input channels (AUX3 and AUX4). This is selected via a command.

The bluetooth system consists of a Microchip BM62 module. It is not normally required to interact with this module other than making or accepting/rejecting a call. The documentation for this module is available from Microsoft's website.

The mechanism to reconnect to a paired device is somewhat flawed in the BM62 as it only reconnects to the last device it can find in a list of 8 previous devices. It does so for only one of its two simultaneous connections. To overcome this the intercom uses the list of 8 previous devices itself to try and connect with two previous devices simultaneously. This could be two phones or a phone and a audio player etc.

Phone calls can be made and accepted from each of two connected phones. Related messages contain a “database” byte selecting/indicating device “0” or “1”.

The BM62 is also used to accept firmware uploads (data transfer) via bluetooth. The mechanism utilizes SPP. New firmware is stored in CPU flash memory separate from executing code. If on restart the new image is verified as good the bootloader will swap the old image for the new one.

The image also contains the binary code for a Razor interface head which can be requested by the Razor so it can also be firmware upgraded indirectly via the intercom. This functionality will of course not be of value for third party control applications and can be ignored.

The A14 differs from the A16 by not having the NAV1 and NAV2 inputs and there is no PAX circuit. The A14 only has a stereo Pilot circuit. The control application can find out if a A14 or A16 is connected by checking the V byte in the primary status message.

The A16 has provision for a Marker beacon receiver. This is a 17th audio source and permanently enabled onto Pilot and PAX circuit as well as the AUX output. Unused three bits in the “Mode” byte of the primary status message will be used as the marker beacon “light” states.

## Data types

| | |
|---|---|
| Byte | 8 bits unsigned |
| Word | 16 bits unsigned. Byte order: LSB first |

## Messages from the device

The device sends a status message or other message at a rate of 10Hz. If a message does not contain any changes when compared to the previous message the message is not sent unless 1 second has elapsed since last transmission. This is applicable to normal regular status messages that are sent without being requested.

### *The primary status message*

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | A16 = 0, A14 = 1 |
| ID | Byte | 2 |
| Mode | Byte | Bit 0: 1=Wired music input is AUX 3,4<br>Bit 1: 1=SideToneFromCOM1 Enabled<br>Bit 2: 1=SideToneFromCOM2 Enabled<br>Bit 3: 1=Call button active<br>Bit 4: 1=Hang up button active |
| SelectedPilot | Word | Audio source bit pattern (L or R) and Selected |
| SelectedPAX | Word | Audio source bit pattern (L or R) and Selected |
| EnabledLines | Word | Audio source bit pattern for Enabled sources |
| MICActive | Byte | Bit 0-5: 1=Signal above VOX level (channel open)<br>Bit 6: 0=TX on COM1 1=TX on COM2<br>Bit 7: 1=Pilot and PAX joined |

| Field Name | Type | Description |
|---|---|---|
| BT | Byte | Bit 0-3 BT link status connection 1<br>Bit 4-7 BT link status connection 2 |
| SigActive | Word | Audio source bit pattern. 1=Signal detect |

## Audio source bit pattern

This bit pattern is used for multiple purposes. Each bit represents a particular audio source.

In order from Bit 0 to Bit 15:

0: MIC1

1: MIC2

2: MIC3

3: MIC4

4: MIC5

5: MIC6

6: COM1

7: COM2

8: NAV1

9: NAV2

10: MUSICL / AUX 3

11: MUSICR / AUX 4

12: AUX1

13: AUX2

14: BTL

15: BTR

### *The secondary status message*

The secondary status message is sent at a rate of 5Hz if any change or 1 Hz if no change.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 6 |
| Signal strength 1 | Byte | 0..Max strength 1 |
| Signal strength 2 | Byte | 0..Max strength 2 |
| Max strength 1 | Byte | Max signal strength 1 |

| Field Name | Type | Description |
|---|---|---|
| Max strength 2 | Byte | Max signal strength 2 |
| Service status 1 | Byte | Phone service status 1 |
| Service status 2 | Byte | Phone service status 2 |
| Pilot volume | Byte | Current pilot volume 0..63 |
| Pax volume | Byte | Current pax volume 0..63 |

Note: On A14 pax volume is equal to pilot volume.
Phones report their received signal strength relative to a max value. This is typically 5. The current value can be anything from 0 to max value (inclusive). Typically this is used for a bar graph display.
Service status is as received from BM62 module. Please refer to BM62 documentation.

## Setup messages

Setup messages are requested by the host on a regular bases if needed. For example you may want to adjust a microphone level. In this case you would request the setup message for the particular microphone input.

Setup messages will alternate with the primary status message but the total message rate will not exceed 10Hz. Once requested the setup message will be sent for 2 seconds if it is not re-requested. Typically you should request it once per second until no longer needed. You can only have one type of setup message active at any one time according to the last request.

| Setup number | Description |
|---|---|
| 1 | Microphone input 1 |
| 2 | Microphone input 2 |
| 3 | Microphone input 3 |
| 4 | Microphone input 4 |
| 5 | Microphone input 5 |
| 6 | Microphone input 6 |
| 7 | COM1 input |
| 8 | COM2 input |
| 9 | NAV1 input |
| 10 | NAV2 input |
| 11 | AUX1 input |
| 12 | AUX2 input |
| 13 | AUX3 input / MUSIC L |
| 14 | AUX4 input / MUSIC R |

| Setup number | Description |
|---|---|
| 15 | MUSIC input |
| 16 | BT audio input |
| 17 | All Audio levels |

## *Microphone inputs*

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 3 |
| MIC | Byte | 0..5 = MIC1 to 6 |
| VOXLevel | Byte | 0..10. 0=VOX is off. |
| MICGain | Byte | 0..63  0=minimum gain. Steps of 0.75db. |
| MICEnabled | Byte | 1=Microphone input is enabled for use |

## *General inputs*

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 4 |
| Channel | Byte | 0..7 =  COM1,COM2,NAV1,NAV2,AUX1,2,3,4<br>8 = Setting for MUSIC (AUX3,4 used as Music L/R)<br>9 = Setting for BT (BT L and BT R use same settings) |
| Input gain | Byte | 0..63 in steps of 0.75db |
| Output gain | Byte | 0..63 in steps of 0.75db (only for COM1 and COM2) |
| Enabled | Byte | Bit 0: 1=Pilot L<br>Bit 1: 1=Pilot R<br>Bit 2: 1=Pax L<br>Bit 3: 1=Pax R<br>Bit 4: 1=Mute on MIC Pilot active<br>Bit 5: 1=Mute on MIC PAX active |

### *TX Audio levels*

Note: This message will be sent at the same rate as other setup messages if any of the setup messages are requested. This message can also be requested on its own.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 5 |
| Inputs | Array of byte | 16 bytes. Each byte represents the audio level at the respective input. Follows order of the Audio source bit pattern. |
| Outputs | Array of byte | 8 bytes, Represents 8 audio outputs in order:<br>PilotL PilotR PAXL PAXR COM2 COM1 BT AUX |
| PTT states | Byte | Bit 0: 1=PTT Pilot active<br>Bit 1: 1=PTT PAX active<br>Bit 2: 1=PTT output COM1 active<br>Bit 3: 1=PTT output COM2 active |

Values are represented in a decibel scale. 0=no signal. 255 = signal is clipping (max value reached or exceeded).

This message should be requested to implement a diagnostics display to the user. It should provide a simple overview of which inputs have a signal and what the current levels are. Note that input signals are shown AFTER gain control and outputs are shown BEFORE gain control.

## Messages from host to device

Messages from host to device fall into two classes: Messages intended to be routed to the BM62 bluetooth module and messages routed to the intercom itself. Generally there is no need to communicate with the bluetooth module directly as the intercom handles most required work.

The first byte of the message sets the destination for the message:

0: Message to be passed through to the bluetooth device.

1: Message for the intercom.

| Field Name | Type | Description |
|---|---|---|
| Routing | Byte | 1 (Message for intercom) |
| Command | Byte | 0..122 |
| Data | Bytes | Optional data as required by command |

## *Command list*

| Command | Description |
|---|---|
| 0..10 | These commands are not used and reserved |
| 11 | Select outputs to enable Bluetooth audio on<br>1 Data byte – Table SetSelected |
| 12 | Select outputs to enable Wired music audio on<br>1 Data byte – Table SetSelected |
| 13 | Select active TX on COM1 |
| 14 | Select active TX on COM2 |
| 15 | Do not Join Pilot and PAX audio |
| 16 | Join Pilot and PAX audio |
| 17 | Increment VOX MIC 1 (range 0-10 internally limited) |
| 18 | Decrement VOX MIC 1 (range 0-10 internally limited) |
| 19 | Increment VOX MIC 2 (range 0-10 internally limited) |
| 20 | Decrement VOX MIC 2 (range 0-10 internally limited) |
| 21 | Increment VOX MIC 3 (range 0-10 internally limited) |
| 22 | Decrement VOX MIC 3 (range 0-10 internally limited) |
| 23 | Increment VOX MIC 4 (range 0-10 internally limited) |
| 24 | Decrement VOX MIC 4 (range 0-10 internally limited) |
| 25 | Increment VOX MIC 5 (range 0-10 internally limited) |
| 26 | Decrement VOX MIC 5 (range 0-10 internally limited) |
| 27 | Increment VOX MIC 6 (range 0-10 internally limited) |
| 28 | Decrement VOX MIC 6 (range 0-10 internally limited) |
| 29 | Increment gain MIC 1 (range 0-63 internally limited) |
| 30 | Decrement gain MIC 1 (range 0-63 internally limited) |
| 31 | Increment gain MIC 2 (range 0-63 internally limited) |
| 32 | Decrement gain MIC 2 (range 0-63 internally limited) |
| 33 | Increment gain MIC 3 (range 0-63 internally limited) |
| 34 | Decrement gain MIC 3 (range 0-63 internally limited) |
| 35 | Increment gain MIC 4 (range 0-63 internally limited) |
| 36 | Decrement gain MIC 4 (range 0-63 internally limited) |
| 37 | Increment gain MIC 5 (range 0-63 internally limited) |
| 38 | Decrement gain MIC 5 (range 0-63 internally limited) |
| 39 | Increment gain MIC 6 (range 0-63 internally limited) |

| Command | Description |
|---|---|
| 40 | Decrement gain MIC 6 (range 0-63 internally limited) |
| 41 | Increment gain COM 1 (range 0-63 internally limited) |
| 42 | Decrement gain COM 1 (range 0-63 internally limited) |
| 43 | Increment gain COM 2 (range 0-63 internally limited) |
| 44 | Decrement gain COM 2 (range 0-63 internally limited) |
| 45 | Increment gain NAV 1 (range 0-63 internally limited) |
| 46 | Decrement gain NAV 1 (range 0-63 internally limited) |
| 47 | Increment gain NAV 2 (range 0-63 internally limited) |
| 48 | Decrement gain NAV 2 (range 0-63 internally limited) |
| 49 | Increment gain AUX 1 (range 0-63 internally limited) |
| 50 | Decrement gain AUX 1 (range 0-63 internally limited) |
| 51 | Increment gain AUX 2 (range 0-63 internally limited) |
| 52 | Decrement gain AUX 2 (range 0-63 internally limited) |
| 53 | Increment gain AUX 3 (range 0-63 internally limited) |
| 54 | Decrement gain AUX 3 (range 0-63 internally limited) |
| 55 | Increment gain AUX 4 (range 0-63 internally limited) |
| 56 | Decrement gain AUX 4 (range 0-63 internally limited) |
| 57 | Increment gain wired Music (range 0-63 internally limited) |
| 58 | Decrement gain wired Music (range 0-63 internally limited) |
| 59 | Increment gain Bluetooth audio (range 0-63 internally limited) |
| 60 | Decrement gain Bluetooth audio (range 0-63 internally limited) |
| 61 | Increment gain COM2 output (range 0-63 internally limited) |
| 62 | Decrement gain COM2 output (range 0-63 internally limited) |
| 63 | Increment gain COM1 output (range 0-63 internally limited) |
| 64 | Decrement gain COM1 output (range 0-63 internally limited) |
| 65 | Increment gain Pilot audio output (range 0-63 internally limited) |
| 66 | Decrement gain Pilot audio output (range 0-63 internally limited) |
| 67 | Increment gain PAX audio output (range 0-63 internally limited) |
| 68 | Decrement gain PAX audio output (range 0-63 internally limited) |
| 69 | Increment gain AUX audio output (range 0-63 internally limited) |
| 70 | Decrement gain AUX  audio output (range 0-63 internally limited) |
| 71 | Data byte 0 = Each microphone input has its own VOX setting<br>Data byte 1 = VOX is a combined setting and applied to all MIC inputs. |
| 72 | Requests stored setup. This is sent as two messages E2Setup1 and |

| Command | Description |
|---|---|
| | E2Setup2. |
| 73 | Reset everything to factory default |
| 74 | Request setup. Data byte is setup number 1-17. This is used to request setup for MIC VOX/Gain/levels and other inputs/outputs. |
| 75 | Enable/Disable COM1 audio source. Data in Table SetAudioEnable |
| 76 | Enable/Disable COM2 audio source. Data in Table SetAudioEnable |
| 77 | Enable/Disable NAV1 audio source. Data in Table SetAudioEnable |
| 78 | Enable/Disable NAV2 audio source. Data in Table SetAudioEnable |
| 79 | Enable/Disable AUX1 audio source. Data in Table SetAudioEnable |
| 80 | Enable/Disable AUX2 audio source. Data in Table SetAudioEnable |
| 81,82,83 | Enable/Disable AUX3 audio source. Data in Table SetAudioEnable |
| 84 | Enable/Disable AUX4 audio source. Data in Table SetAudioEnable |
| 85 | Enable/Disable Wired Music audio source. Data in Table SetAudioEnable |
| 86 | Enable/Disable Bluetooth audio source. Data in Table SetAudioEnable |
| 87 | Not used |
| 88 | Increment Pilot and PAX volume. Both will be set the same |
| 89 | Decrement Pilot and PAX volume. Both will be the same |
| 90 | Wired music input is Music L/R |
| 91 | Wired music input is reassigned to AUX 3 and AUX 4 |
| 92 | Do not used. Development use gateway only. |
| 93 | Data is microphone split.<br>0=MIC 1 on on Pilot circuit, 2-6 on PAX.<br>1=MIC  1,2 on Pilot circuit, 3-6 on PAX |
| 94 | Input enables. Two Data bytes forming a Word. Bit Pattern is used to enable and disable channels for use. Note: this does not affect any operation in the device – this value is reported back via the output messages. It is used to disable unused inputs from the user panel. It is suggested to use the audio source bit pattern. |
| 95 | If Data1=$AA and Data2=$12 starts writing EEProm settings to BM62 bluetooth module. The only time this is used is if you fit a new module. This then configures the module to work with the intercom system. |
| 96 | System tests development function. Factory only. Do not use – gateway function. |
| 97 | Playback Audio recording. CVR.<br>Data1 sets function:<br>0: Stop CVR Playback and continue recording.<br>1: Stop recording and start CVR playback from oldest recording<br>2: Send all stored CVR records on RS232 |

| Command | Description |
|---|---|
|  | 3: Acknowledge receipt of a sector (related to function 2)<br>4: Erase CVR recording<br><br>If playback is activated, ASCII message will be sent “Playback active” when oldest record has been located – this can take a short while.<br>“Playback end” when last recording playback has completed. Recording will continue automatically.<br>“Erase complete” when CVR recording has been erased (takes a short while). |
| 98 | Data=1 → Enable sidetone from COM1 during TX |
| 99 | Data=1 → Enable sidetone from COM2 during TX |
| 100 | Start transmission of Binary for Razor interface (this will cause the Razor to reprogram itself with the Razor binary stored in the firmware for the intercom). |
| 101 | Set pilot Mute source. Selects the audio sources that will be muted for the pilot circuit if the Pilot MIC(s) are active. Use Audio source bit pattern. |
| 102 | Set PAX Mute source. Selects the audio sources that will be muted for the PAX circuit if the PAX MIC(s) are active. Use Audio source bit pattern. |
| 103 | Get input Name. Data is index for audio source name request:<br>0-6: NAV1,NAV2,AUX1,AUX2,AUX3,AUX4,MUSIC |
| 104 | Set input Name. Data1 is indexs for audio source name to set (see 103). Data2 is length of name (1-7 characters). Followed by 1-7 ASCII characters. |
| 105 | Start self test. Note: requires test harness to be plugged into both D25 connectors as shown in A16 manual. |
| 106 | End test (not normally used – test ends itself when done). |
| 107 | Make a short “Beep” sound. Used to acknowledge key press etc. |
| 108 | Full reset of stored settings to factory default. Data1 must be $AA |
| 109 | Select outputs to enable COM1 audio on<br>1 Data byte – Table SetSelected |
| 110 | Select outputs to enable COM2 audio on<br>1 Data byte – Table SetSelected |
| 111 | Select outputs to enable NAV1 audio on<br>1 Data byte – Table SetSelected |
| 112 | Select outputs to enable NAV2 audio on<br>1 Data byte – Table SetSelected |
| 113 | Select outputs to enable AUX1 audio on<br>1 Data byte – Table SetSelected |
| 114 | Select outputs to enable AUX2 audio on<br>1 Data byte – Table SetSelected |

| Command | Description |
|---|---|
| 115<br>116<br>117 | Select outputs to enable AUX3 audio on<br>1 Data byte – Table SetSelected<br>Select outputs to enable AUX4 audio on<br>1 Data byte – Table SetSelected<br>Select outputs to enable Wired music audio on<br>1 Data byte – Table SetSelected<br>Note: This will be treated as 1 setting common for left and right. This is not used if music input is reassigned as AUX3 and 4. |
| 118<br>119<br>120<br>121 | Send version information message<br>Send serial number<br>Do not route BM62 messages to RS232 port (default)<br>Get Bluetooth name (results in ASCII message) |
| 122 | Set Bluetooth name. Data1 is length of name (1-32 characters). Followed by 1-32 ASCII characters. This is the name of the bluetooth device that will be advertised on a receiving device for pairing.<br>Note: Many phones will only remember the first name they know when pairing a device – once paired and you change the name, they may still only show the old name. They may only show the new name if you delete the connection on the phone and pair again. Even this may not work as the name is cached in the phone. |

### *Table SetSelected*

This table describes a 1 byte data value used to enable/disable audio source onto a audio destination (Pilot and PAX left and right channels).

| Data | Description |
|---|---|
| 0 | Toggle current state Pilot Left and Right (Left and Right will be the same) |
| 1 | Enable Pilot Left and Right (Left and Right will be the same) |
| 2 | Disable Pilot Left and Right (Left and Right will be the same) |
| 3 | Toggle current state PAX Left and Right (Left and Right will be the same) |
| 4 | Enable PAX Left and Right (Left and Right will be the same) |
| 5 | Disable PAX Left and Right (Left and Right will be the same) |
| 6 | Toggle current state Pilot Left |
| 7 | Enable Pilot Left |
| 8 | Disable Pilot Left |
| 9 | Toggle current state PAX Left |
| 10 | Enable PAX Left |
| 11 | Disable PAX Left |
| 12 | Toggle current state Pilot Right |

| Data | Description |
|---|---|
| 13 | Enable Pilot Right |
| 14 | Disable Pilot Right |
| 15 | Toggle current state PAX Right |
| 16 | Enable PAX Right |
| 17 | Disable PAX Right |

### *Table SetAudioEnable*

| Data | Description |
|---|---|
| 0 | Toggle current state |
| 1 | Enable Audio source (unmute) |
| 2 | Disable Audio source (mute) |

### *ASCII Message*

The device may send ASCII messages to indicate certain conditions. The host application should pop-up the message and display the last received message to the user. The user shall have a function to dismiss the popup.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 7 |
| Length | Byte | Length of following message (up to 20 characters) |
| Message | Array of byte | ASCII message |

### *Version Message*

The version message is sent on request. Version info shown here is example.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 13 |

| Field Name | Type | Description |
|---|---|---|
| Length | Byte | 5 |
| Type | Array of byte | Text:  A16-1 |
| Length | Byte | 6 |
| Version | Array of byte | Text:  261119 |
| Filler | Byte | 255 |
| Length | Byte | 01/20/20 |
| Text | Array of byte | V1 REL 1 |

### *Serial number Message*

The serial number message is sent on request.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 10 |
| Serial | Array of byte | 4 bytes. Read as 32 bit integer with LSB first. Serial number. |

### *Bluetooth system messages*

The device will interact with the BM62 bluetooth module and react to certain events and inform the host system via the bluetooth system message

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 1 |
| V | Byte | 0 |
| ID | Byte | 14 |
| Kind | Byte | The kind of message following |
| Data | Array of byte | Data for the message |

| Kind | Description |
|---|---|
| 2 | Call status, 2 bytes data |
| 3 | Caller ID, Variable size data following |

| Kind | Description |
|---|---|
| 0x17 | Linked device information |
| 0x21 | Read local device name reply |

Please refer to the BM62 documentation from Microchip. The “Kind” number is identical to the Command in the reply format. The data following is the same as sent by the device.

The A16 intercom takes care of most protocol and interfacing details with the BM62 module leaving interaction required by the host system to a minimum. For the most part you deal with the call status that signals an incoming call, send accept or hangup/reject or make a call.

For these interactions you would pass BM62 commands 0x00 (Make call) and 0x02 (MMI)

You can interact with two phones and make calls on either phone. You cannot access the phones internal database. You make a call by passing a phone number.

The A16 keeps track of active calls and will handle protocol related details with the BM62.

The Phone service status in secondary status message from the A16 is based on the contents of BM62 command 0x0B received.

## *Messages to the BM62 bluetooth module*

| Field Name | Type | Description |
|---|---|---|
| Routing | Byte | 0 (Message for BM62) |
| Data | Bytes | Data as required |

Data is passed to the BM62 unmodified. You need to prepare a command as outlined in the BM62 documentation including checksum.
The A16 will pass this on and handle the acknowledge phase.

## *Getting a copy of all data sent by the BM62*

Note: This is for developmental purposes only. It is not normally needed.

Command 96 with Data1 of 7 will enable BM62 passthrough mode. Data1 of 8 disables passthrough.

You will receive messages containing BM62 data – the data will contain all bytes received. The data is not formatted into BM62 messages.

| Field Name | Type | Description |
|---|---|---|
| M | Byte | Protocol ID = 0 |
| Data | Array of | One or more bytes of data received from BM62 |

| Field Name | Type | Description |
|---|---|---|
| | byte | |

Data is read from the BM62 into a buffer. If the buffer reaches a fillcount of 50 a message is sent and the buffer cleared. If there is no more data from the BM62 for about 20 milliseconds and there is data in the buffer it will be sent and the buffer cleared.

## CAN based message transfer

Transferring messages via can uses the same message contents as described in this document for both directions.

CAN messages are limited to 8 bytes of data per packet. This implementation uses CAN at 250KBaud data rate with 29 bit identifiers.

The identifier bits are used to implement an addressing scheme as well as a multi-packet protocol.

Bits 0..3: These bits are used as a destination identifier. In this implementation this is “don't care”.

Bits 4..7: Packet Type. In this implementation this is “don't care”

Bits 8..11: Packet number: The number of the packet in a multipacket message. First packet is 0.

Bits 12...15: Number of packets in this message. If this is equal to Bits 8..11 then we have received the last packet and can process the message.

Bits 16..17: No function.

Bits 18..28: 11 bits CAN address.

We use the CAN address for both a function and a device address. We use the low 4 bits as function and the remaining 7 higher order bits as device address.

For this implementation the A16 device address is 74 (Decimal) and the function is “do not care” but should be assumed to be “0” for future use.

Messages **SENT** by the A16 are addressed to device 76 decimal, function 0.

## Flash memory data storage order

Block 1: Items that do not change often

InputGains: array[0..15] of byte;

OutputGains: array[0..7] of byte;

NAV1Name: string[7];

NAV2Name: string[7];

AUX1Name: string[7];

AUX2Name: string[7];

```
  AUX3Name: string[7];
  AUX4Name: string[7];
  MUSICName: string[7];
  SelectedPilotL: longint;
  SelectedPilotR: longint;
  SelectedPAXL: longint;
  SelectedPAXR: longint;
  MusicWIsAux: boolean;
  VOXCombined: byte;
  MICSplit: byte;
  E2Spare1: byte;
  EnabledLines: word;
  SideToneFromCOM1: boolean;
  SideToneFromCOM2: boolean;
  MuteOnMICPilot: longint;
  MuteOnMICPAX: longint;
Block 2: Items that can change frequently
  SelectedSource: longint;
  VOXLevel: array[0..7] of byte;
  PilotVolume: byte;
  PAXVolume: byte;
  TXCOM: Byte;
  JOIN: Byte;
```