ADSBee 1090 Firmware Reference Guide

Notes about how the firmware works and why.

John McNelly

john@pantsforbirds.com

# Inter-Processor SPI Communication

Inter-processor SPI communication is done with maximum transfer lengths of 64 Bytes, since the vast majority of transfers are individual raw transponder packets being forwarded in real time, which are < 20 Bytes. Transfers of large objects like the Settings struct (up to 8kB) are done in small packets 64 Bytes at a time. This takes a while due to the extra overhead, but is a very infrequent occurrence so the impact on performance is minimal.

#### Master Single Write to Slave

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Transfer 1** | Master Write Packet | | | | | |
| **Byte** | 0 | 1 | 2:3 | 4 | 5:(n-2) | (n-1):n |
| **MOSI** | CMD  kWriteToSlave | ADDR | OFFSET | LEN  (unused, since length can be inferred from clocks) | DATA | CRC |
| **MISO** |  |  |  |  |  |  |

#### Master Single Read from Slave

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Transfer 1** | Master Read Request packet | | | | |
| **Byte** | 0 | 1 | 2:3 | 4:5 | 6 |
| **MOSI** | CMD  kReadFromSlave | ADDR | OFFSET | LEN | CRC |
| **MISO** |  |  |  |  |  |

Handshake line goes HI.

|  |  |  |  |
| --- | --- | --- | --- |
| **Transfer 2** | Slave Read Response Packet | | |
| **Byte** | 0 | 1:(n-2) | (n-1):n |
| **MOSI** |  |  |  |
| **MISO** | CMD  kReadFromSlave | DATA | CRC |

Handshake line goes LO.

#### Slave Single Write to Master

Handshake line goes HI.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Transfer 1** | Slave Write Packet | | | | | |
| **Byte** | 0 | 1 | 2:3 | 4 | 5:(n-2) | (n-1):n |
| **MOSI** |  |  |  |  |  |  |
| **MISO** | CMD  kWriteToMaster | ADDR | OFFSET | LEN | DATA | CRC |

Handshake line goes LO.

#### Slave Single Read from Master

Handshake line goes HI.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Transfer 2** | Slave Read Request Packet | | | | | Master Read Response Packet | | |
| **Byte** | 0 | 1 | 2:3 | 4 | 5 | 6 | 7:(n-2) | (n-1):n |
| **MOSI** |  |  |  |  |  | CMD  kDataBlock | DATA | CRC |
| **MISO** | CMD  kReadFromMaster | ADDR | OFFSET | LEN | CRC |  |  |  |

Handshake line goes LO.

#### REQ Field

When the ADDR field is not being used to indicate a byte offset, it is used to indicate an object type.

|  |  |
| --- | --- |
| **REQ Value** |  |
| 0x1 (kSlaveRequestWrite) | Asks RP2040 to write LEN Bytes from object corresponding to ADDR, with Byte offset OFFSET.  This status response is sent as a reply to a single transfer with CMD = kMasterCommandRequestWrite. |
| 0x2 (kSlaveRequestRead) | Asks RP2040 to read LEN Bytes and store them in object corresponding to ADDR, with Byte offset OFFSET.  This status response is sent as a reply to a single transfer with CMD = kMasterCommandRequestRead |
|  |  |
|  |  |
|  |  |
|  |  |

### Behaviors

#### RP2040 Writes Small Object to ESP32

* RP2040 asserts chip select.
* RP2040 sends Single Transfer with CMD = kMasterCommandFastWrite.
* RP2040 de-asserts chip select.

#### RP2040 Writes Large Object to ESP32

* Rp2040 sends Single Transfer with CMD = kMasterCommandRequestWrite.
* ESP32 asserts HANDSHAKE GPIO line.
  + RP2040 receives HANDSHAKE interrupt.
  + RP2040 asserts chip select.
  + RP2040 reads 5 Bytes to get important part of Status Response.
    - STATUS: kStatusRequestMasterWrite
    - ADDR: Address of the object that the RP2040 was trying to write.
    - LEN: Size of chunk to write.
    - OFFSET: How far into the object we’ve gotten so far.
  + RP2040 reads LEN Bytes from the object corresponding with ADDR with Byte offset OFFSET and sends them in the DATA payload, with a CRC16 checksum in the CRC field.
  + RP2040 de-asserts chip select.
  + ESP32 checks the CRC against DATA and stores the payload as necessary. If the CRC fails, the ESP32 can request the same section again.
  + This subsection repeats until the ESP32 is done receiving data and stops asserting the HANDSHAKE GPIO line.

#### ESP32 Reads Small or Large Object from RP2040

* Same as RP2040 Writes Large Object to ESP32, but begins from ESP32 asserting HANDSHAKE GPIO line.

RP2040 Reads Small or Large Object from ESP32