AltOS Companion Port

Table of Contents

License

Companion Port

Many Altus Metrum products come with an eight pin Micro MaTch connector, called the Companion Port. This is often used to program devices using a programming cable. However, it can also be used to connect TeleMetrum to external companion boards (hence the name).

The Companion Port provides two different functions:

- Power. Both battery-level and 3.3V regulated power are available. Note that the amount of regulated power is not huge; TeleMetrum contains a 150mA regulator and uses, at peak, about 120mA or so. For applications needing more than a few dozen mA, placing a separate regulator on them and using the battery for power is probably a good idea.
- SPI. The flight computer operates as a SPI master, using a protocol defined in this document. Companion boards provide a matching SPI slave implementation which supplies telemetry information for the radio downlink during flight

Companion SPI Protocol

The flight computer implements a SPI master communications channel over the companion port, and uses this to get information about a connected companion board and then to get telemetry data for transmission during flight.

At startup time, the flight computer sends a setup request packet, and the companion board returns a board identifier, the desired telemetry update period and the number of data channels provided. The flight computer doesn't interpret the telemetry data at all, simply packing it up and sending it over the link. Telemetry packets are 32 bytes long, and companion

packets use 8 bytes as a header leaving room for a maximum of 12 16-bit data values.

Because of the limits of the AVR processors used in the first two companion boards, the SPI data rate is set to 187.5kbaud.

SPI Message Formats

This section first defines the command message format sent from the flight computer to the companion board, and then the various reply message formats for each type of command message.

Table 1. Companion Command Message

Offset	Data Type	Name	Description
0	uint8_t	command	Command identifier
1	uint8_t	flight_state	Current flight computer state
2	uint16_t	tick	Flight computer clock (100 ticks/second)
4	uint16_t	serial	Flight computer serial number
6	uint16_t	flight	Flight number
8			

Table 2. Companion Command Identifiers

Value	Name	Description
1	SETUP	Supply the flight computer with companion information
2	FETCH	Return telemetry information
3	NOTIFY	Tell companion board when flight state changes

The flight computer will send a SETUP message shortly after power-up and will then send FETCH messages no more often than the rate specified in the SETUP reply. NOTIFY messages will be sent whenever the flight state changes.

'flight_state' records the current state of the flight, whether on the pad, under power, coasting to apogee or descending on the drogue or main chute.

'tick' provides the current flight computer clock, which be used to synchronize data recorded on the flight computer with that recorded on the companion board in post-flight analysis.

'serial' is the product serial number of the flight computer, 'flight' is the flight sequence number. Together, these two uniquely identify the flight and can be recorded with any companion board data logging to associate the companion data with the proper flight.

NOTIFY commands require no reply at all, they are used solely to inform the companion board when the state of the flight, as computed by the flight computer, changes. Companion boards can use this to change data collection parameters, disabling data logging until the flight starts and terminating it when the flight ends.

SETUP reply message

Table 3. SETUP reply contents

Offset	Data Type	Name	Description
0	uint16_t	board_id	Board identifier
2	uint16_t	board_id_inverse	~board_id—used to tell if a board is present
4	uint8_t	update_period	Minimum time (in 100Hz ticks) between FETCH commands
5	uint8_t	channels	Number of data channels to retrieve in FETCH command
6			

The SETUP reply contains enough information to uniquely identify the companion board to the end user as well as for the flight computer to know how many data values to expect in reply to a FETCH command, and how often to fetch that data.

To detect the presence of a companion board, the flight computer checks to make sure that board_id_inverse is the bit-wise inverse of board_id. Current companion boards use USB product ID as the board_id, but the flight computer does not interpret this data and so it can be any value.

FETCH reply message

Table 4. FETCH reply contents

Offset	Data Type	Name	Description
0	uint16_t	data0	0th data item
2	uint16_t	data1	1st data item
...			

The FETCH reply contains arbitrary data to be reported over the flight computer telemetry link. The number of 16-bit data items must match the 'channels' value provided in the SETUP reply message.

History and Motivation

To allow cross-programming, the original TeleMetrum and TeleDongle designs needed to include some kind of connector. With that in place, adding the ability to connect external cards to TeleMetrum was fairly simple. We set the software piece of this puzzle aside until we had a companion board to use.

The first companion board was TeleScience. Designed to collect temperature data from the nose and fin of the airframe, the main requirement for the companion port was that it be able

to report telemetry data during flight as a back-up in case the TeleScience on-board data was lost.

The second companion board, TelePyro, provides 8 additional channels for deployment, staging or other activities. To avoid re-programming the TeleMetrum to use TelePyro, we decided to provide enough information over the companion link for it to independently control those channels.

Providing a standard, constant interface between the flight computer and companion boards allows for the base flight computer firmware to include support for companion boards.

