Things to Know about Fox/Golf/LTM Flight Software

Just recently we (Eric K1TVV and I) had a hard time getting software to work on an older LTM module. I finally figure it out after re-acquiring some knowledge that I'd forgotten. Eric suggested I write it up, so here is some kind of general knowledge about the flight software for Fox, LTM and Golf (which are all generations of the same software).

The Flash Loader

This software was originally written by Bdale Garbee and Keith Packard in part to use with the flight computer their company (https://altusmetrum.org/) sells to use with model rockets. Bdale also designed and built or had built the IHU (processor) for Fox satellites, later called the LIHU in Golf-TEE and LTM.

The Flash Loader is a software module that resides in the bottom of flash memory in the STM32L151 processor and can load the main flight software above itself. The flash loader itself must be installed in flash memory using ST software and the STLink hardware. After that it is possible to load flight software using a small Python program (the host loader) without any special hardware.

The flash loader is the first thing executed whenever the processor starts or resets. It first looks at a signal line called "umbilical attached". If this signal line is low, it immediately jumps to the flight code. Otherwise, it starts executing, creating or connecting to a serial port to await commands. (One can override the default behavior by writing certain values into SRAM—this allows one to reset the processor and execute the flight code even without the attached line low.) There are commands to jump to the flight code which one can enter by hand, or which the host loader executes when it is done loading the flight code.

Just as a note, the term "load" denotes putting code into the processor's non-volatile flash memory. This code is executed directly out of flash memory, so when the processor is reset or powered on, there is no "loading" that takes place. The process of initializing on reset or power cycle is generally called "booting".

There are two versions of the flash loader. The Altus Metrum version uses USB to create a port on a host computer (COM port for Windows or /dev/ttyACM port for Linux). The host loader finds and uses this COM port to upload the software. Because we wanted to use a serial line interface on Golf-TEE, I modified the A-M loader to use the processor's UART device. Further, Heimir Thor Sevrinsson (W1ANT) wrote a serial line version for the Golf RT-IHU. I am only talking about the LIHU version right now.

The original USB LIHU flash loader fits in 4K of flash memory from address 0x8000000 to 0x8001000. It assumes that the vector table for the flight code starts at 0x8001000. Unfortunately, the serial line flash loader is too large to fit in that space (and I don't remember why). Thus, it assumes that the flight code vector table starts at the next available location, 0x8002000. That means that in order for the flash loader to jump to the flight code, the flight code must be built to start at a different address depending on the loader type in the IHU. This is done by changing the configuration file (config.h) to include the statement #define USB_LOADER or not.

Flight Software Console

On the ground testers and software developers generally use the flight software console to issue commands, check status, and initialize the software for flight. The console always requires a terminal emulator connected to a serial port on the host computer. Like the flash loader, the console can connect to the processor via serial lines or directly via USB. If it is via serial lines, one can connect the serial lines to any sort of serial interface to connect to the host computer (there is a serial-to-USB interface on the CIU board in Golf). For Fox-1, MESAT-1 and possibly others, the console is actually a direct USB interface run by the flight software. The flight software is compiled to use one or the other using #define USB_CONSOLE in config.h. Usually USB_CONSOLE and USB_LOADER are both defined or both not defined.

Note that for debugging, serial is much easier since the serial port on the host computer does not disappear each time the flight computer resets or reloads. I also generally use the STLink device to load the software during development since that allows me to use the gdb debugger that is part of the build software.

Flight Software Version Numbers

Software versions are called Ny.zm where N is generally "X" or "V". If the version number starts with X, the software should not be flown. In most cases, this means it is compiled to be debugged (either extra print statements, or not optimized, or with config values convenient for debugging, but not right for flight). If the version number starts with V, it is compiled optimized and generally has debug print statements removed, and flight values configured.

y specifies the satellite that the software is built for. Many (but not all) flight software versions will at least boot on other satellites. For example, the LTM generic code will run on the Golf-TEE LIHU. The satellites vs. numbers table is in Appendix A

z is a major version number, m is a letter specifying the minor version. I always try to change the small letter when there is any change to the software and it is released outside of my shack. The major version will change when there is a major (often incompatible) change or if the minor version exceeds z.

Example:

X10.3j is a debug version of flight software for generic LTM testing with major version 3, minor version j.

Appendix A: Satellite number table

Note that the satellite numbers are also used to identify their telemetry so that FoxTelem can figure out how to decode the telemetry and which tab to display it on. Since FoxTelem is used for several different purposes, some of these numbers don't have IHU software loads). These definitions are in the FoxTelem satellite master files (search for foxId=):

Satellite Name	Satellite Number
Fox-1a (AO-85)#	1
Fox-1b (RadFxSat, AO-91)@	2
Fox-1Cliff (AO-95)#	3
Fox-1D (AO-92)#	4
Fox-1E (RadFxSat-2 AO-109)@	5
HuskySat (HO-103)#	6
CubeSat Simulator (DUV)*	7
Golf-TEE	8
Generic LTM for testing	9
MESAT-1	10
ARISS*	11
CubeSat Simulator (PSK)*	99

*There is no LIHU or RT-IHU flight software for these satellites.
#In orbit, but EOL
@In orbit partially operational (as of May 2023)