

Lesson 2

2/2

Objectives

- Describe how to run the core Flight System (cFS) from the GUI and from the command line
- Provide troubleshooting guidance

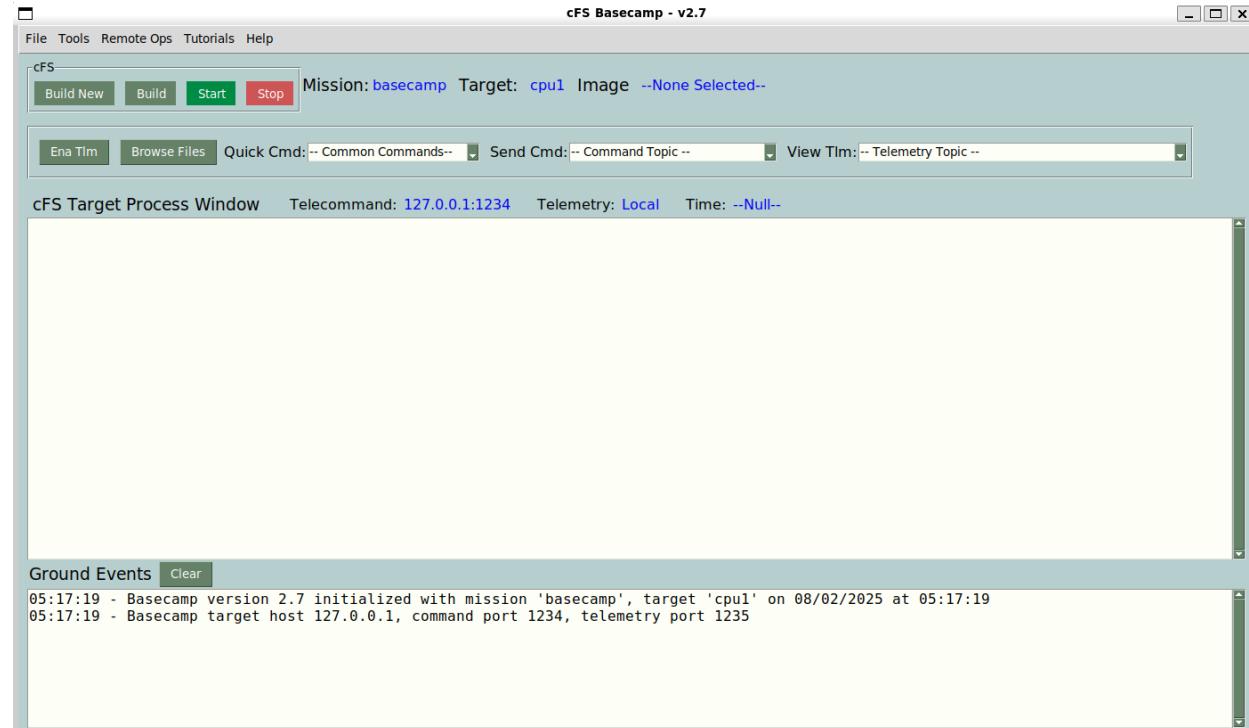
Preliminary Information

- The cFS requires some Linux resources that should not require elevated privileges to access but depending on the user's platform configuration the cFS may need to be run with elevated privileges using using SUDO
- *Appendix A: cFS Startup Troubleshooting* provides details for solving privilege related runtime issues
- Basecamp's GUI provides a single interface for building and running the cFS, however, you can still build and run from the command line and use the GUI for a command and telemetry interface
 - See *Appendix B: Start/Stop cFS from the Command Line*

Starting the cFS from the GUI (1 of 3)

Open a terminal window and issue the following commands to start the GUI (assumes you are starting in git clone location):

```
cd cfs-basecamp/gnd-sys/app  
./setvars.sh  
python3 basecamp.py
```



If you see the following error, then either setvars.sh wasn't run or the cFS has not been built and the python libraries don't exist.

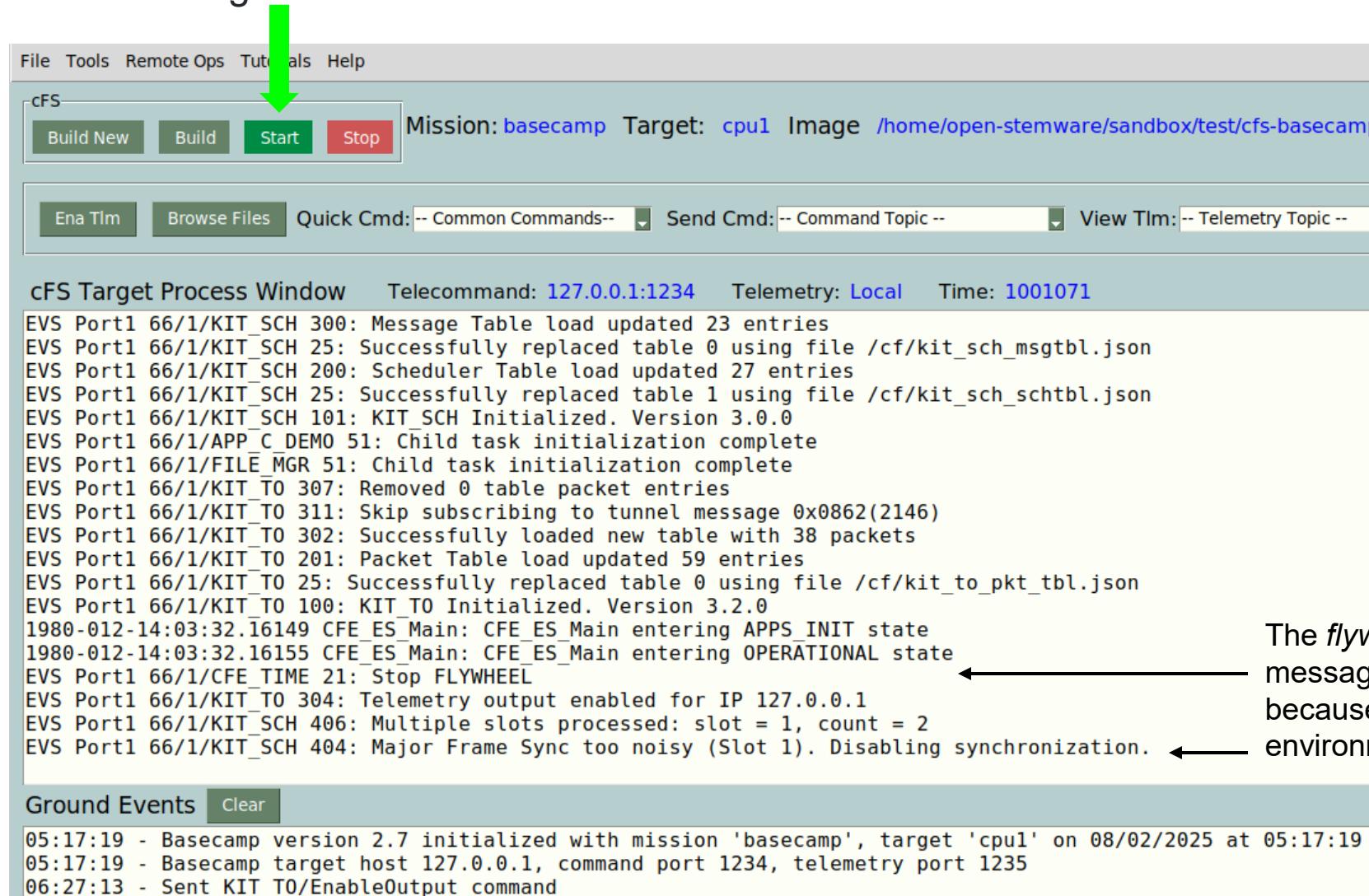


```
grandpa-dave@ubuntu:~$ cd cfs-basecamp/  
grandpa-dave@ubuntu:~/cfs-basecamp$ cd gnd-sys/app/  
grandpa-dave@ubuntu:~/cfs-basecamp/gnd-sys/app$ python3 basecamp.py  
LD_LIBRARY_PATH not defined. Run setvars.sh to correct the problem  
grandpa-dave@ubuntu:~/cfs-basecamp/gnd-sys/app$ █
```

Starting the cFS from the GUI (2 of 3)

Click <Start> to run the cFS in a new Linux process

- After a several seconds messages will appear in the ‘cFS Process Window’ to indicate the cFS is running



Starting the cFS from the GUI (3 of 3)

If you scroll up in the cFS Process Window you can find when `cfe_es_startup.scr` is opened

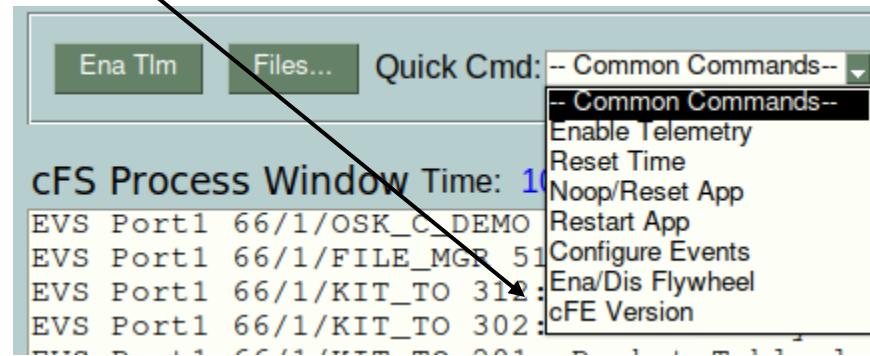
cFS Process Window Time: 1049773

```
1980-013-03:32:10.58743 CFE_ES_StartApplications: Opened ES App Startup file: /cf/cfe_es_startup.scr ←
1980-013-03:32:10.58762 CFE_ES_ParseFileEntry: Loading shared library: /cf/cfe_assert.so
1980-013-03:32:10.58801 [BEGIN] CFE FUNCTIONAL TEST
1980-013-03:32:10.58802 [BEGIN] 01 CFE-STARTUP
1980-013-03:32:10.58813 CFE_ES_ParseFileEntry: Loading shared library: /cf/osk_c_fw.so
OSK C Application Framework Library Initialized. Version 4.0.0
1980-013-03:32:10.64302 CFE_ES_ParseFileEntry: Loading file: /cf/osk_c_demo.so, APP: OSK_C_DEMO
1980-013-03:32:10.71270 CFE_ES_ParseFileEntry: Loading file: /cf/ci_lab.so, APP: CI_LAB_APP
1980-013-03:32:10.72164 CFE_ES_ParseFileEntry: Loading file: /cf/kit_to.so, APP: KIT_TO
OS_ModuleLoad_Impl():111:Error loading shared library: ./cf/sch_lab.so: cannot open shared object file: No such file or
directory
1980-013-03:32:10.72197 CFE_ES_ParseFileEntry: Loading file: /cf/sch_lab.so, APP: SCH_LAB_APP
1980-013-03:32:10.72201 CFE_ES_LoadModule: Could not load file:/cf/sch_lab.so. EC = -1
1980-013-03:32:10.72209 CFE_ES_ParseFileEntry: Loading file: /cf/file_mgr.so, APP: FILE_MGR
1980-013-03:32:10.72241 CFE_ES_ParseFileEntry: Loading file: /cf/file_xfer.so, APP: FILE_XFER
```

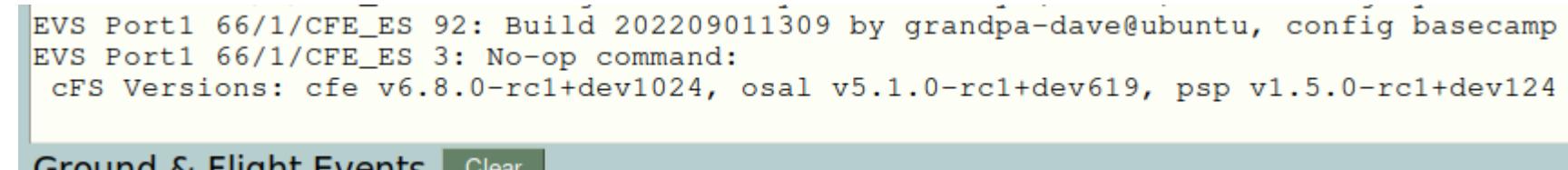
A successful startup includes success messages from each library and app as they are loaded and initialized

Verify Operations

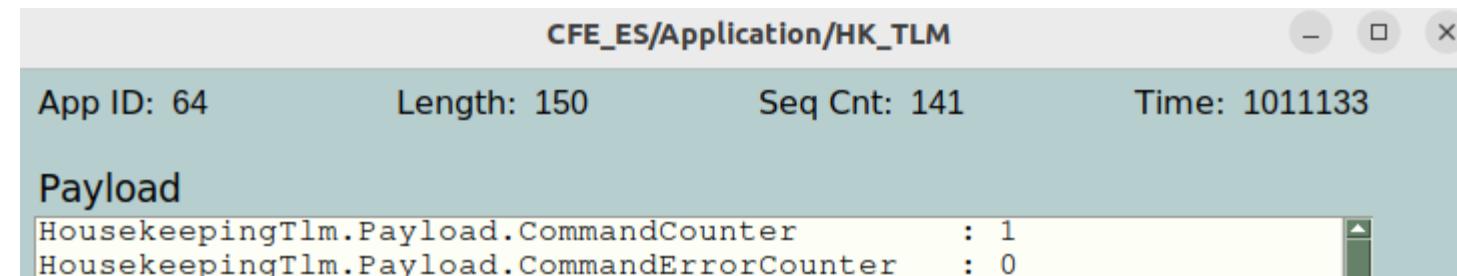
Using the Quick Cmd 'cFE Version' option is a quick way to verify the cFS target is operational



'cFE Version' sends a cFE Executive Service No Op command that reports the current cFE version in an event message



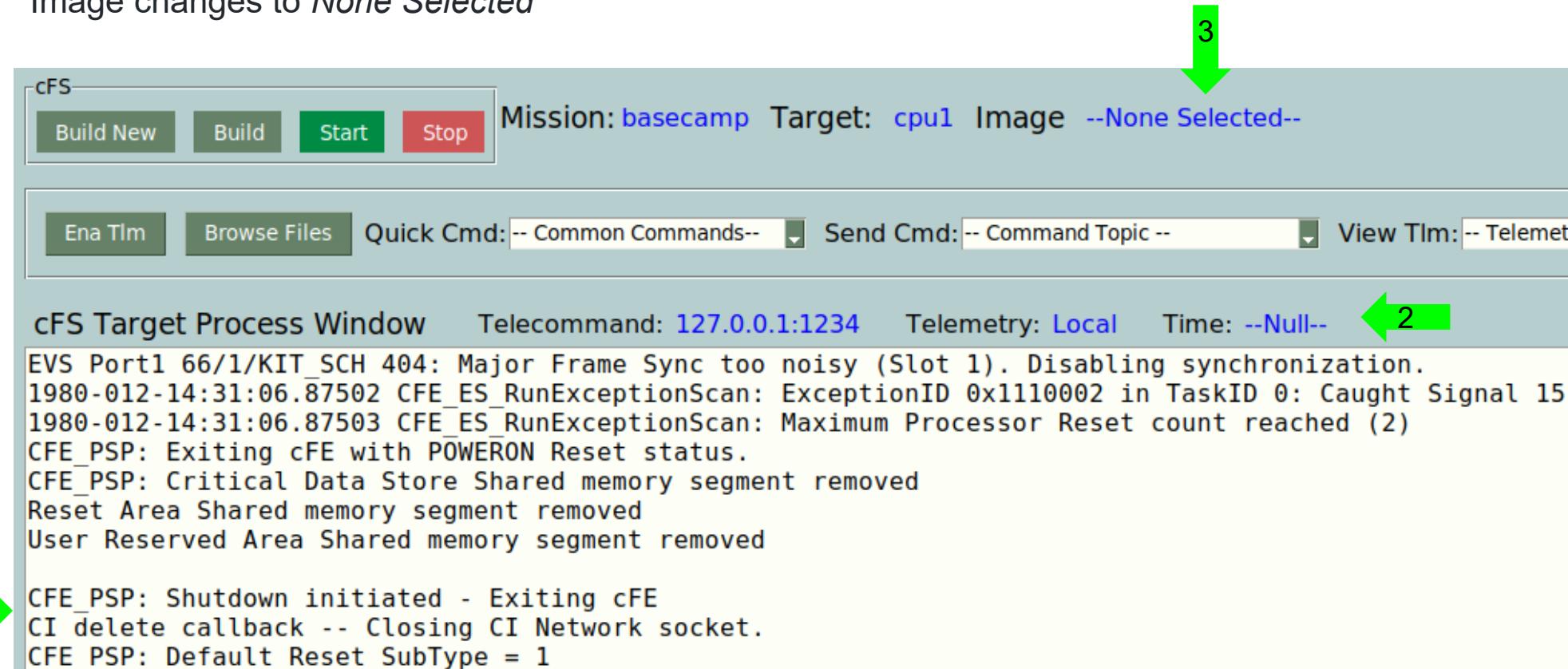
The No Op command also increments ES's command counter that is reported in its Housekeeping telemetry



Stopping the cFS from the GUI

From the GUI click <Stop> to terminate the cFS process

1. The cFS Process terminates
2. Time changes to *Null*
3. Image changes to *None Selected*



If the cFS was started from a terminal window, the GUI <Stop> button will have no effect

Appendix A

cFS Startup Troubleshooting

Potential Startup Error (1 of 2)

A known potential startup error is a permission denied as shown here

```
cFS Process Window Time: Null
exe_dir = /home/grandpa-dave/test/cfs-basecamp/cfe-eds-framework/build/exe/cpu1
exe_file = core-cpu1
OS_Posix_GetSchedulerParams():189:Policy 1: available, min-max: 1-99
OS_Posix_GetSchedulerParams():189:Policy 2: available, min-max: 1-99
OS_Posix_TaskAPI_Impl_Init():375:Selected policy 2 for RT tasks, root task = 99
OS_Posix_TaskAPI_Impl_Init():392:Could not setschedparam in main thread: Operation not permitted (1)
OS_API_Init():151:OS_API_Impl_Init(0x1) failed to initialize: -1
OS_API_Init():228:Warning: Microsecs per sec value of 0 does not equal 1000000 (MicroSecPerTick: 0 TicksPerSecond: 0)
OS_printf():300:BUG: OS_printf() called when OSAL not initialized: CFE_PSP_Panic Called with error code = 0x%08X. Exiting.
OS_printf():300:BUG: OS_printf() called when OSAL not initialized: The cFE could not start.
```

Potential Startup Error (2 of 2)

2/2

Potential causes and solutions

- The OSAL has a permissive mode setting that should be set so resources can be accessed
- To perform a clean build from the command line
 - Issue '*make distclean*' to clear all cmake caches
 - Issue '*make SIMULATION=native prep*' to set OSAL's permissive mode should be set
- Basecamp's GUI <Build New> button performs these steps the two steps above
- If you continue to get the error, you can run in privileged mode
 - The configuration settings are in cfs-basecamp/gnd-sys/app/basecamp.ini
 - In [CFS_TARGET] section set SUDO_START_CFS to True or False (default)
 - In [APP] section set PASSWORD = <your Linux user account password>; only required if SUDO_START_CFS is set to True

Basecamp Shell Scripts

- The Basecamp cFS buttons start shell scripts to perform the operation
- If you're comfortable with writing shell scripts you can modify them to suite your needs
- The shell scripts are located in **cfs-basecamp/gnd-sys/app**
- <Build New>
 - *build_cfs_topicids.sh*
- <Build>
 - *build_cfs.sh*
- <Start>
 - *start_cfs.sh* or *sudo_start_cfs.sh* depending on basecamp.ini configuration
- <Stop>
 - *stop_cfs.sh*

Appendix B

Start/Stop cFS from the Command Line

Starting the cFS from the Command Line (1 of 2)

2/2

**Open a terminal window and issue the following commands
(assumes starting in git clone location):**

```
> cd cfs-basecamp/cfe-eds-framework/build/exe/cpu1  
> ./core-cpu1
```

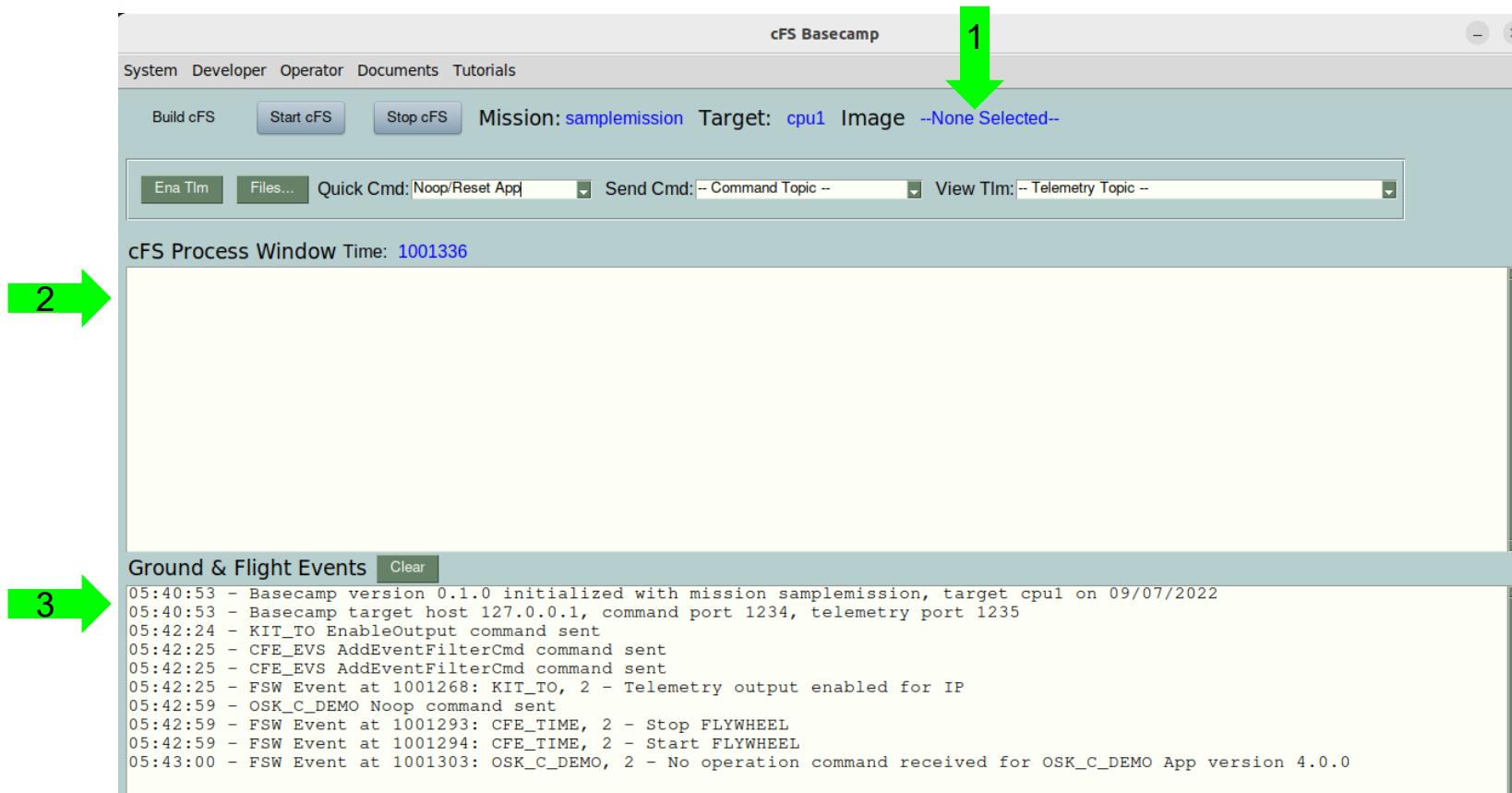


```
grandpa-dave@ubuntu:~/cfs-basecamp$ cd cfe-eds-framework/build/exe/cpu1/  
grandpa-dave@ubuntu:~/cfs-basecamp/cfe-eds-framework/build/exe/cpu1$ ./core-cpu1  
OS_BSP_Initialize():Maximum user msg queue depth = 10  
CFE_PSP: Default Reset SubType = 1  
CFE_PSP: Default CPU ID = 1  
CFE_PSP: Default Spacecraft ID = 66  
CFE_PSP: Default CPU Name: cpu1  
OS_Posix_GetSchedulerParams():189:Policy 1: available, min-max: 1-99  
OS_Posix_GetSchedulerParams():189:Policy 2: available, min-max: 1-99  
OS_Posix_TaskAPI_Impl_Init():375:Selected policy 2 for RT tasks, root task = 99  
OS_Posix_TaskAPI_Impl_Init():392:Could not setschedparam in main thread: Operation not permitted (1)  
CFE_PSP: Instantiated software timebase 'cFS-Master' running at 10000 usec  
CFE_PSP: Using POSIX monotonic clock as CFE timebase  
CFE_PSP: Using MMAP simulated EEPROM implementation  
CFE_PSP: Physical RAM access not implemented  
CFE_PSP: I/O Port access not implemented  
CFE_PSP: EEPROM Range (2) created: Start Address = 7F1CE43B7000, Size = 00080000 Status = 0  
CFE_PSP: Starting the cFE with a POWER ON reset.  
CFE_PSP: Clearing out CFE CDS Shared memory segment.  
CFE_PSP: Clearing out CFE Reset Shared memory segment.  
CFE_PSP: Clearing out CFE User Reserved Shared memory segment.  
1980-004-06:22:10.24802 CFE_ES_SetupResetVariables: POWER ON RESET due to Power Cycle (Power Cycle).  
1980-004-06:22:10.24806 CFE_ES_Main: CFE_ES_Main in EARLY_INIT state
```

Starting the cFS from the Command Line (2 of 2)

The GUI can still be used with a cFS process that was not started using the GUI, however

1. The *Image* name will not be loaded
2. The *cFS Process Window* will be blank. The terminal window where the cFS was started serves as the cFS Monitor Display
3. Flight events will still appear in the Ground & Flight Events window



Stopping the cFS from the Command Line

Enter Ctrl-c in the terminal window that was used to start the cFS

1. The exception is caught and cFS process is terminated

```
1980-012-14:03:34.44183 CFE_ES_Main: CFE_ES_Main entering APPS_INIT state
1980-012-14:03:34.44184 CFE_ES_Main: CFE_ES_Main entering OPERATIONAL state
EVS Port1 66/1/CFE_TIME 21: Stop FLYWHEEL
EVS Port1 66/1/KIT_SCH 404: Major Frame Sync too noisy (Slot 1). Disabling synchronization.
^C1980-012-14:03:45.46238 CFE_ES_RunExceptionScan: ExceptionID 0x1110001 in TaskID 0: Caught SIGINT
1980-012-14:03:45.46240 CFE_ES_RunExceptionScan: Processor Reset count not reached (1/2)
CFE_PSP: Exiting cFE with PROCESSOR Reset status.

CFE_PSP: Shutdown initiated - Exiting cFE
CI delete callback -- Closing CI Network socket.
```

Terminate a cFS process from a terminal window not used to start the cFS

1. Determine the cFS process identifier
2. Use elevated privileges to “kill” the process

```
1 grandpa-dave@ubuntu:~/Desktop$ pgrep core
50891
2 grandpa-dave@ubuntu:~/Desktop$ sudo kill 50891
[sudo] password for grandpa-dave:
grandpa-dave@ubuntu:~/Desktop$ pgrep core
grandpa-dave@ubuntu:~/Desktop$
```

cFS Target Monitor Terminal

```
EVS Port1 66/1/CFE_TIME 20: Start FLYWHEEL
EVS Port1 66/1/CFE_TIME 21: Stop FLYWHEEL
1980-012-14:04:59.78396 CFE_ES_RunExceptionScan: ExceptionID 0x1110002 in TaskID 0: Caught Signal 15
1980-012-14:04:59.78397 CFE_ES_RunExceptionScan: Maximum Processor Reset count reached (2)
CFE_PSP: Exiting cFE with POWERON Reset status.
CFE_PSP: Critical Data Store Shared memory segment removed
Reset Area Shared memory segment removed
User Reserved Area Shared memory segment removed

CFE_PSP: Shutdown initiated - Exiting cFE
CI delete callback -- Closing CI Network socket.
```