

# ► Multistage 2015

Release Documentation – October 27th, 2015

By Fred I8

## Index

INTRODUCTION.....	2
WHAT IS MULTISTAGE?.....	2
INSTALLATION.....	2
COMPATIBILITY .....	2
REQUIRED ADDONS.....	2
KEYS .....	2
RELEASE POLICY .....	2
CREDITS .....	3
MULTISTAGE2.DLL REPLACEMENT .....	3
DESCRIPTION AND LIMITATIONS .....	3
NUMBER LIMITATIONS:.....	3
OTHER LIMITATIONS, FROM VINKA DOCS .....	3
ALL PARAMETERS – INI FILE .....	5
MISCELLANEOUS.....	5
TEXTURE .....	6
PARTICLE STREAM .....	7
BOOSTERS.....	8
STAGES .....	10
INTERSTAGES .....	13
LAUNCH ESCAPE SYSTEM .....	14
FAIRING.....	15
PAYLOADS.....	16
EFFECTS.....	17
ALL PARAMETERS – GUIDANCE FILE .....	19
ALL PARAMETERS – SCENARIO FILE.....	21
SOME MORE NOTES .....	22
THE NEW ORBIT GUIDANCE PROGRAM .....	22
COMPUTATIONAL EVALUATION OF GRAVITY TURN .....	23
COMPLEX FLIGHT.....	23
FAILURES.....	24
TELEMETRY .....	24
GROWING PARTICLES EFFECT .....	24
PARTICLES PACKAGING .....	24
DEVELOPER MODE.....	25
WHAT IF I HAVE A LIVE PAYLOAD AND I NEED TO ADD SPECIFIC PARAMETERS FOR IT IN THE SCENARIO FILE? .....	25
LOG FILE .....	25
MULTISTAGE 2015 MFD .....	26
FLIGHT SETTINGS SCREEN - FST .....	26
VEHICLE SCREEN – VEH .....	26
GUIDANCE SCREEN - GNC .....	27
PAYLOAD SCREEN – PLD .....	28
CONTROL SCREEN – CTR .....	28
MONITOR SCREEN - MNT .....	28
EXTERNAL REFERENCE VESSEL.....	29
SLS HEAVY LIFTER EXAMPLE .....	30
THANKS .....	30

# Multistage 2015

Release Documentation – October 27th, 2015

By Fred18

## Introduction

The idea of Multistage2015 is to recreate the Vinka's Multistage2 module and update it with a number of new features.

First Release: October 27<sup>th</sup>, 2015

## What is Multistage?

Multistage is a module that allows developers who are not capable or willing to code in C++ to develop multistage launchers programming a large set of features of the launcher through a windows .ini file, editable using simply notepad. The file is divided in sections which contain the main characteristics and features. Some entry in the ini file are mandatory for the module to work and to avoid the notorious CTDs, some other are optional and are part of a spectrum of features which can be used and customized at pleasure of the developer.

Of course the module is not only intended to be useful for addon developers not capable of coding, but also to all those who are willing to focus for example to space probes or other parts and can exploit the functionalities of Multistage2015 for their best.

This release includes also a Multistage2015 MFD which can be very helpful for flight performance measuring and very funny to use.

## Installation

Simply extract the content of the compressed file into the orbiter main directory.

## Compatibility

This addon is made and compatible with Orbiter 2010PI version.

## Required Addons

Stage.dll 2010 by BrianJ is required for default jettisoned items.

## Keys

[J] for jettison boosters / interstages / stages

[F] for jettison LES / Fairing

[P] for toggling the autopilot

[CTRL]+[SPACEBAR] for activating developer mode

[SPACEBAR] when developer mode is active to reset the vehicle and reload the .ini file.

## Release Policy

Multistage2015 is distributed FREEWARE. Its code is distributed along with the dll. Nobody is authorized to exploit the module or the code or parts of them commercially directly or indirectly.

You CAN distribute the dll together with your addon but in this case you MUST:

- Include credit to the author in your addon documentation;
- Add to the addon documentation the official link of Orbit Hangar Mods for download and suggest to download the latest and updated version of the module.

You CAN use parts of the code of Multistage2015, but in this case you MUST:

- Give credits in your copyright header and in your documentation for the part you used.
- Let your project be open source and its code available for at least visualization by other users.

You CAN NOT use the entire code for making and distributing the very same module claiming it as your work entirely or partly.

You CAN NOT claim that Multistage2015 is an invention of yourself or a work made up by yourself,

or anyhow let intend that is not made and coded by the author.

You install and use Multistage2015 at your own risk, author will not be responsible for any claim or damage subsequent to its use or for the use of part of it or of part of its code.

Be also aware that I am not a professional programmer so code may be difficult to read and may present issues that I am not aware of.

### Credits

The biggest credit goes to Vinka for the idea and the implementation of the multistage module which has lasted for decades now. This is anyway a new module, since no code from Vinka was available or used.

Credit to Face for pointing me to the GetPrivateProfileString code which boosted a lot the implementation of the module.

Credit to Hlynkacg because the OrientForBurn function he published in the forum inspired the attitude control function used in the module.

### multistage2.dll replacement

A fast way to turn all the original multistage2 launchers in multistage2015 vessels is to replace directly the original multistage2.dll file.

In order to do this, after the installation of Multistage2015:

- 1) Go to your orbiter\_root\Modules folder
- 2) Find the Multistage2.dll and rename it to Multistage2\_BACKUP.dll
- 3) Make a copy of the Multistage2015.dll file
- 4) Rename the copy into multistage2.dll

This will intercept all multistage2 vessels. If you want the original Vinka's Multistage2 back, just delete the multistage2.dll file you've created and rename the backup file to multistage2.dll.

### Description and Limitations

#### Number Limitations:

Stages:	10
Engines for each Stage:	32
Boosters:	10
Engines for each Booster:	4
Interstages for each stages:	1
LES:	1
Payloads:	10
Meshes for each Payload:	5
Mach effects	10
Vent effects	10
Ullage for each stage:	No limitations
Explosive Bolts for each stage / booster	1
Fairing Elements:	10
Particle Stream Definitions:	13
Particles for each engine group:	2
Exhaust Textures:	16
Exhaust for each engine group:	1

#### Other limitations, FROM VINKA DOCS

##### Boosters

Boosters can be attached only to the first stage, You can define up to 10 groups of boosters. A group can contain up to 10 boosters (each booster in the group as the same characteristics), use only different groups if you want to define different characteristics such as the thrust or the burning time. Boosters are automatically fired when the first stage engine level reach 95% of the nominal thrust or after a programmable delay. Boosters are automatically jettisoned when fuel mass is falling to 0.

##### Stages

Only one stage can be fired at a time, when the fuel mass of the reach 0, the stage is jettison and the next stage (if any) is ignited. If the first stage is running out of fuel and boosters are still thrusting, the first stage separation will be delayed until all boosters are jettisoned. You can define up to 10 stages.

##### Interstages

An interstage can be defined between stages. The interstage is dropped after previous stage jettison and before next stage ignition. Up to 9 interstages can be defined.

### **Payloads**

A maximum of 10 payloads can be defined. You can program the focus so that, at the defined payload jettison, the vessel focus goes to this payload giving you the control of the payload.

This change of focus is not very eye friendly because the image jump to the new focus vessel with a different position and size. If a fairing is defined, the fairing must be jettison prior to payload jettison. During the payload separation phase, the last stage remains active. Therefore, you can change the attitude, rotation speed before jettisoning the payload. If you have kept enough fuel, you can even restart the stage main engine to change the orbit. The first payload that will be jettison must be placed on top of the payload stack.

### **Fairing**

A fairing can protect the payloads. This fairing can be composed of 10 elements maximum.

## ALL PARAMETERS – INI FILE

### MISCELLANEOUS

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[MISC]</b>	Miscellaneous Parameters	Optional	
<b>COG=xxx</b>	Center Of gravity Elevation in Meters, used to display the launcher at a certain altitude above ground	Optional	No
<b>GNC_DEBUG=0/1</b>	If set to 1 shows some debug information in the Debug String	Optional	No – Information displayed are different
<b>TELEMETRY=0/1</b>	If set to 1 each minute of flight save Telemetry data into a txt file which contains Comma Separated Values. See Teleemetry paragraph	Optional	No – Saved data are partly different
<b>FOCUS=xxx</b>	Can be set between 1 and the number of payloads defined when the specified payload is jettisoned. The focus will change from the multistage vessel to the newly created payload.	Optional	No
<b>THRUST_REAL_POS=0/1</b>	If set to 1 thrust from the engines is applied where the engines are and in the direction specified with engine_dir. This is valid for both stages and boosters	Optional	<b>NEW! HOT!</b>
<b>VERTICAL_ANGLE=xxx</b>	[degrees] Set the angle from vertical with which the vehicle is sitting on the ground	Optional	<b>NEW!</b>

TEXTURE

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[TEXTURE_LIST]</b>	User must declare all the texture files that are used by engine exhausts, the list will be preloaded by the dll		
<b>TEX_x=abc</b>	x is the progressive texture number. Abc means the name of the texture file (without the .dds) that will be rendered as exhausts. Maximum of 16 textures allowed	Optional	No

## PARTICLE STREAM

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[PARTICLESTREAM_x]</b>	Particle Stream Definition, x is the progressive number of the particle definition, from 1 to 13	Section is optional, parameters are mandatory if particle is defined	No
<b>Name=abc</b>	Name of particle stream, to be used later on	Mandatory	No
<b>Srcsize=xxx</b>	Size of the Particle	Mandatory	No
<b>Srcrate=xxx</b>	Rate of Particle Production	Mandatory	No
<b>V0=xxx</b>	Initial velocity of the emitted particles	Mandatory	No
<b>Srcspread=xxx</b>	Spread of the emitted particles	Mandatory	No
<b>Lifetime=xxx</b>	Life of the particles in seconds	Mandatory	No
<b>Growthrate=xxx</b>	Growth rate of the particles	Mandatory	No
<b>Atmslowdown=xxx</b>	Atmospheric slow down of the particle	Mandatory	No
<b>Ltype=EMISSIVE/DIFFUSE</b>	Ltype of the particle	Mandatory (if not specified a warning in the log file will be produced and the emissive type will be applied)	No
<b>Levelmap=LVL_FLAT / LVL_LIN / LVL_SQRT / LVL_PLIN</b>	Level Map of the particle	Mandatory (if not specified a warning in the log file will be produced and LVL_LIN will be applied)	No
<b>Lmin=xxx</b>	Minimum ref Level	Mandatory	No
<b>Lmax=xxx</b>	Maximum ref Level	Mandatory	No
<b>Atmsmap= ATM_FLAT / ATM_PLIN / ATM_PLOG</b>	Atmospheric Map	Mandatory (if not specified a warning in the log file will be produced and ATM_PLIN will be applied)	No
<b>Amin=xxx</b>	Atmo min ref level	Mandatory	No
<b>Amax=xxx</b>	Atmo max ref level	Mandatory	No
<b>Tex=abc</b>	Texture name of the particle	Mandatory(if not specified a warning in the log file will be produced and contrail3 will be applied)	No
<b>GrowFactor_size=xxx</b>	Growing Particles factor relevant to Size (see dedicated chapter)	Optional	<b>NEW! HOT!</b>
<b>GrowFactor_rate=xxx</b>	Growing Particles factor relevant to Growthrate (see dedicated chapter)	Optional	<b>NEW! HOT!</b>

**BOOSTERS**

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[BOOSTER_x]</b>	Booster Section, x is the number from 1 to 10	Optional	
<b>N=xxx</b>	Number of boosters in the defined boosters' group	Mandatory	No
<b>Meshname=abc</b>	Name of mesh of the boosters (remember that then the meshes abc_1, abc_2... abc_N will be searched)	Mandatory	No
<b>Off=(xx,yy,zz)</b>	[meters] Offset of the first booster mesh	Mandatory	No
<b>Height=xxx</b>	[meters] Height of the booster	Optional	No
<b>Angle=xxx</b>	[degrees] rotation of the first mesh in the group around Z axis	Optional (default value is 0)	No
<b>Diameter=xxx</b>	[meters] Diameter of the booster	Optional	No
<b>Thrust=xxx</b>	[Newton] Thrust of the single booster of the group (will be multiplied by N for total boosters group thrust)	Mandatory	No
<b>Emptymass=xxx</b>	[Kg] empty mass of the single booster of the group	Mandatory	No
<b>Fuelmass=xxx</b>	[Kg] fuel mass of the single booster of the group	Mandatory	No
<b>Burntime=xxx</b>	[seconds] Burn time of the boosters of the group	Mandatory	No
<b>Burndelay=xxx</b>	[seconds] Delay of booster group ignition	Optional	No
<b>Speed=(xx,yy,zz)</b>	[m/s] Separation speed of the boosters of the group (of the first booster in the group, the others will be symmetrical about Z axis)	Optional (default is (3,0,0) )	No (default value added)
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] Separation rotational speed of the boosters in the group (of the first booster in the group, the others will be symmetrical about Z axis)	Optional (default is (0,-0.1,0) )	No (default value added)
<b>Module=abc</b>	Name of the module of the jettisoned booster	Optional (default is "stage" )	No
<b>Eng_x=(xx,yy,zz)</b>	Position of the engine of the booster, x is the	Mandatory	No



	progressive number, from 1 to 4 at maximum		
<b>Eng_diameter=xxx</b>	[meters] diameter of the exhaust of the engine	Optional (default is half of booster diameter)	No (default value added)
<b>Eng_tex=abc</b>	Name of the texture of the engine	Optional	No
<b>Eng_dir=(xx,yy,zz)</b>	Engine direction, exhaust will be rotated and this vector will be used in case of thrust_real_pos=1 <i>Note: only 1 parameter for all the engines is required. The direction will be applied to all the engines of the booster group</i>	Optional	<b>NEW!</b>
<b>Eng_pstream1=abc</b>	Engine first particle stream name	Optional	No
<b>Eng_pstream2=abc</b>	Engine second particle stream name	Optional	No
<b>Curve_x=(xx,yy)</b>	Thrust curve for booster, x is the progressive number, maximum 10 points are allowed. Xx=time of power Yy=thrust level	Optional	No
<b>Expbolts_pos=(xx,yy,zz)</b>	Explosive Bolts position	Optional	<b>NEW!</b>
<b>Expbolts_pstream=abc</b>	Name of the explosive bolts particle stream	Optional (mandatory if expbolts_pos is defined)	<b>NEW!</b>
<b>Expbolts_anticipation=xxx</b>	[seconds] Time of expbolts effect before booster cutoff	Optional (mandatory if expbolts_pos is defined)	<b>NEW!</b>

## STAGES

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[STAGE_x]</b>	Stage section, x is the stage number from 1 to 10, maximum 10 allowed, no holes in the sequence allowed, stage_1 is mandatory		
<b>Meshname=abc</b>	Name of the mesh	Mandatory	No
<b>Off=(xx,yy,zz)</b>	Offset of the mesh	Mandatory	No
<b>Height=xxx</b>	[meters] Height of the stage	Mandatory	No
<b>Diameter=xxx</b>	[meters] Diameter of the stage	Mandatory	No
<b>Thrust=xxx</b>	[N] Thrust of the stage	Mandatory	No
<b>Emptymass=xxx</b>	[kg] empty mass of the stage	Mandatory	No
<b>Fuelmass=xxx</b>	[kg] fuel mass of the stage	Mandatory	No
<b>Burntime=xxx</b>	[sec] time of burning of the stage	Mandatory	No
<b>Ignite_delay=xxx</b>	[sec] delay before stage ignition (not to be used with first stage)	Optional	No
<b>Speed=(xx,yy,zz)</b>	[m/s] speed of separation of stage once jettisoned	Optional	No
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] rotational speed of separation of stage once jettisoned	Optional	No
<b>Module=abc</b>	Name of the module of the stage once jettisoned	Optional (default is "stage")	No
<b>Pitchthrust=xxx</b>	[N] pitch thrust	Optional	No
<b>Yawthrust=xxx</b>	[N] yaw thrust	Optional	No
<b>Rollthrust=xxx</b>	[N] roll thrust	Optional	No
<b>Eng_x=(xx,yy,zz)</b> <b>Eng_x=(xx,yy,zz,tt)</b>	Engine position, x is the progressive number from 1 to 32, maximum 32 allowed.  New fourth optional parameter has been added. It's a scale parameter, in order to scale the exhaust of the specified engine, example ENG_3=(0,0,-10,0.7) will produce an exhaust 30% smaller than the others	Optional (default is (0,0,-height*0.5) )	<b>Yes!</b>

<b>Eng_diameter=xxx</b>	[meters] diameter of the exhaust of the engines	Optional (default is half of stage diameter)	No
<b>Eng_dir=(xx,yy,zz)</b>	Direction of the engine, exhaust will be rendered in the opposite direction, thrust will be applied in this direction if thrust_real_pos=1 is defined <i>Note: only 1 parameter for all the engines is required. The direction will be applied to all the engines of the booster group</i>	Optional	<b>NEW! HOT!</b>
<b>Eng_tex=abc</b>	Name of the texture of the exhaust	Optional	No
<b>Eng_pstream1=abc</b>	Name of the first particle stream engines	Optional	No
<b>Eng_pstream2=abc</b>	Name of the second particle stream of the engines	Optional	No
<b>Reignitable=0/1</b>	If set to 0 once the stage has been turned off it cannot be turned on again	Optional (default is 1)	<b>NEW! HOT!</b>
<b>Battery=xx</b>	[Hours] battery life in hours, if present battery will discharge while mission is proceeding and if battery gets to 0 the stage will be dead (no engines or rcs). If not present batteries will simply never discharge	Optional	<b>NEW! HOT!</b>
<b>Ullage_thrust=xx</b>	[Newton] ullage thrust in Newton, if present ullage are added	Optional	<b>NEW!</b>
<b>Ullage_anticipation=xx</b>	[sec] time of ullage ignition before engine ignition	Optional	<b>NEW!</b>
<b>Ullage_overlap=xx</b>	[sec] time of ullage remaining ignited once stage has ignited	Optional	<b>NEW!</b>
<b>Ullage_N=xx</b>	Number of ullage engines	Optional (mandatory if ullage is defined)	<b>NEW!</b>
<b>Ullage_angle=xx</b>	[degrees] angle of first ullage engine from the vertical	Optional (default is 0)	<b>NEW!</b>
<b>Ullage_diameter=xx</b>	[meters] diameter of ullage exhausts	Optional (mandatory if ullage is defined)	<b>NEW!</b>

<b>Ullage_length=xx</b>	[meters] length of ullage exhausts	Optional (mandatory if ullage is defined)	<b>NEW!</b>
<b>Ullage_dir=(xx,yy,zz)</b>	Direction of first ullage engine (the others will be symmetrical with Z axis)	Optional (mandatory if ullage is defined)	<b>NEW!</b>
<b>Ullage_pos=(xx,yy,zz)</b>	Position of first ullage engine (the others will be symmetrical with Z axis)	Optional (mandatory if ullage is defined)	<b>NEW!</b>
<b>Ullage_tex=abc</b>	Name of the texture for the ullage exhausts	Optional	<b>NEW!</b>
<b>Ullage_rectfactor=xx</b>	Scale of asymmetry of distribution of ullage exhausts. Applies only if ullage_N is even, the higher than 1 the more the exhausts are compressed laterally	Optional (default is 1, square distribution)	<b>NEW!</b>
<b>Expbolts_pos=(xx,yy,zz)</b>	Position of explosive bolts	Optional	<b>NEW!</b>
<b>Expbolts_pstream=abc</b>	Name of explosive bolts particle stream	Optional	<b>NEW!</b>
<b>Expbolts_anticipation=xx</b>	[sec] Time of anticipation of explosive bolts before stage cutoff	Optional	<b>NEW!</b>
<b>Particles_packed_to_engine=xx</b>	This command allows to pack all the particles definition of a stage into a single particle stream for a single engine. To get more details on its use check the dedicated paragraph below.	Optional	<b>NEW!</b>

## INTERSTAGES

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[SEPARATION_XY]</b>	Intestage definition, x is lower stage, y is upper stage  An adapter is possible defining x as last stage and y as last stage+1		
<b>Meshname=abc</b>	Meshname	Mandatory	No
<b>Off=(xx,yy,zz)</b>	[meters] mesh offset	Mandatory	No
<b>Height=xxx</b>	[meters] height	Optional	No
<b>Diameter=xxx</b>	[meters] diameter	Optional	No
<b>Emptymass=xxx</b>	[kg] empty mass	Mandatory	No
<b>Separation_delay=xxx</b>	[sec] time between staging and interstage separation	Optional	No
<b>Speed=(xx,yy,zz)</b>	[m/s] translation separation speed	Optional	No
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] rotation separation speed	Optional	No
<b>Module=abc</b>	Module name of the jettisoned interstage	Optional (default is "stage")	No

## LAUNCH ESCAPE SYSTEM

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[LES]</b>	Launch Escape System Section	Optional	<b>NEW!</b>
<b>Meshname=abc</b>	Name of the mesh	Mandatory if les is defined	<b>NEW!</b>
<b>Off=(xx,yy,zz)</b>	[meters] offset of the mesh	Mandatory if les is defined	<b>NEW!</b>
<b>Height=xx</b>	[meters] height of the Les	Mandatory if les is defined	<b>NEW!</b>
<b>Diameter=xx</b>	[meters] diameter of Les	Mandatory if les is defined	<b>NEW!</b>
<b>Emptymass=xx</b>	[kg] emptymass of the Les	Mandatory if les is defined	<b>NEW!</b>
<b>Speed=(xx,yy,zz)</b>	[m/s] translation speed of jettisoned les	Optional	<b>NEW!</b>
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] rotation speed of jettisoned les	Optional	<b>NEW!</b>
<b>Module=abc</b>	Name of the module of the jettisoned les	Optional (default is "stage")	<b>NEW!</b>

## FAIRING

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[FAIRING]</b>	Fairing section	Optional	
<b>N=xx</b>	Number of fairing meshes	Mandatory	No
<b>Meshname=abc</b>	Name of meshes of fairing (remember that abc_1...abc_N will be searched)	Mandatory	No
<b>Off=(xx,yy,zz)</b>	[meters] Offset of first mesh	Mandatory	No
<b>Height=xx</b>	[meters] Height of fairing	Mandatory	No
<b>Angle=xx</b>	[degrees] Rotation angle of first fairing mesh	Optional (Default is 0)	No
<b>Diameter=xx</b>	[meters] Diameter of fairing	Mandatory	No
<b>Emptymass=xx</b>	[kg] empty mass of fairing	Mandatory	No
<b>Speed=(xx,yy,zz)</b>	[m/s] translation speed of jettisoned fairing	Optional	No
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] rotation speed of jettisoned fairing	Optional	No
<b>Module=abc</b>	Name of module of jettisoned fairing	Optional (default is "stage")	No

## PAYLOADS

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[PAYLOAD_x]</b>	Payload section, x is the progressive number from 1 to 10, maximum 10 allowed		
<b>Meshname=abc</b>	Meshname. Up to 5 meshes allowed	Mandatory	No
<b>Off=(xx,yy,zz)</b>	[meters] mesh offset, up to 5, one for each mesh	Mandatory	No
<b>Height=xx</b>	[meters] payload height	Mandatory	No
<b>Diameter=xx</b>	[meters] payload diameter	Mandatory	No
<b>Mass=xx</b>	[kg] payload mass	Mandatory	No
<b>Module=abc</b>	Name of module of jettisoned payload	Mandatory	No
<b>Name=abc</b>	Name of jettisoned payload	Mandatory	No
<b>Speed=(xx,yy,zz)</b>	[m/s] translation speed of jettisoned payload	Optional	No
<b>Rot_speed=(xx,yy,zz)</b>	[deg/s] rotation speed of jettisoned payload	Optional	No
<b>Rotation=(xx,yy,zz)</b>	[degrees] rotation of payload around the three axis	Optional	<b>NEW! HOT!</b>
<b>Render=0/1</b>	If set to 1 renders the payload mesh even if fairings are on	Optional	No
<b>Live=0/1</b>	If set to 1 create a live payload instead of a dead mesh! If a payload is live the mass parameter can be avoided and the vehicle will load the live payload mass. If there is the need of manual overriding the live payload mass value, just keep the mass parameter	Optional	<b>NEW! HOT!</b>



## EFFECTS

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[FX_MACH]</b>	Section of Mach I effect	Optional	<b>NEW!</b>
<b>Pstream=abc</b>	Name of the effect particle stream	Mandatory	<b>NEW!</b>
<b>Mach_min=xx</b>	Minimum mach number for effect trigger	Mandatory	<b>NEW!</b>
<b>Mach_max=xx</b>	Maximum mach number for effect cutoff	Mandatory	<b>NEW!</b>
<b>Off_x=(xx,yy,zz)</b>	[meters] position of mach effect. X is the progressive number, maximum 10 allowed. Note: all of the mach effects will share the particle definition, the direction and the other parameters, it will be only replicated x times in the position desired by the user	Mandatory	<b>NEW!</b>
<b>Dir=(xx,yy,zz)</b>	Direction of mach effect exhaust stream	Mandatory	<b>NEW!</b>

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[FX_VENT]</b>	Section for venting effect (before launch)	Optional	<b>NEW!</b>
<b>Pstream=abc</b>	Name of particle stream to be used	Mandatory	<b>NEW!</b>
<b>Off_x=(xx,yy,zz)</b>	[meters] position of exhaust number x. x is the progressive number from 1 to 10, maximum 10 allowed	Mandatory	<b>NEW!</b>
<b>Dir_x=(xx,yy,zz)</b>	Direction of exhaust number x	Mandatory	<b>NEW!</b>
<b>Time_fin_x=xxx</b>	Second of the countdown when vent number x will be turned off. Check will be performed up to MET=+5 sec.	Optional (default value is 0)	<b>NEW!</b>

Section / Parameter	Explanation	Mandatory/Optional	Changed from Vinka?
<b>[SOUND]</b>			
<b>Main_thrust</b>	Specify the sound file to be played for main thrust	Optional	No
<b>Hover_thrust</b>	Specify the sound file to be played for hover thrust	Optional	No
<b>RCS_Thrust_attack</b>	Specify the sound fil to be played for RCS thrust	Optional	No
<b>RCS_Thrust_sustain</b>	Specify the sound fil to be played for RCS thrust	Optional	No
<b>Air_conditionning</b>	Specify the sound file to be played for air conditionning	Optional	No
<b>Cockpit_ambience_1 .. 9</b>	Specify the sound file to be played for cockpit ambience	Optional	No

## ALL PARAMETERS – GUIDANCE FILE

Command	Explanation	Changed from Vinka?
<b>engine(xx,yy,zz)</b>	Set Engine Level from xx to yy in time zz	No
<b>Roll(xx,yy,zz,tt)</b>	Vinka Roll Program	No
<b>Pitch(xx,yy,tt)</b>	Vinka Pitch step, from xx, to yy in time tt	No
<b>Fairing()</b> <b>Fairing(xx)</b>	Fairing() jettison fairing Fairing(xx) jettison fairing when altitude xx in km is reached. For example 10=fairing(80) will start checking at MET=10 seconds for altitude and will jettison fairing when altitude reaches 80km	<b>YES!</b>
<b>Les()</b> <b>Les(xx)</b>	Like fairing but with Les	<b>NEW!</b>
<b>Disable(pitch Roll Jettison)</b>	Disables pitch control, roll control or autojettison	No
<b>Playsound(abc)</b>	Plays soundfile abc	No
<b>Target(xx)</b>	Check for ApA. If the value xx (in km) is reached cutoff the engines	No
<b>Aoa(xx)</b> <b>Aoa(xx,tt)</b>	Set attitude in order to get the desired xx aoa  If parameter tt is present, the command last for time tt otherwise for 60 seconds. Time not applicable if loaded by MFD	<b>YES!</b>
<b>Attitude(xx,yy,zz)</b> <b>Attitude(xx,yy,zz,tt)</b>	Set Attitude of xx pitch, yy heading, zz roll  If parameter tt is present, the command last for time tt otherwise for 60 seconds. Time not applicable if loaded by MFD	<b>YES!</b>
<b>Spin(xx)</b> <b>Spin(xx,tt)</b>	Spin the vehicle with the specified xx roll velocity  If parameter tt is present, the command last for time tt otherwise for 60 seconds. Time not applicable if loaded by MFD	<b>YES!</b>

<b>Inverse()</b>	Switch the flight mode from upright to upside down or viceversa <i>Note: its use in the guidance program will not be saved in the scenario file. Therefore if you use it, close the scenario and open it again the rocket will start to roll back to the original mode.</i>	<b>NEW!</b>
<b>Engineout()</b> <b>Engineout(xx)</b>	Turn off engine number xx. If no engine is specified it turns off the last engine set in the ini file	<b>NEW!</b>
<b>Orbit(PeA,ApA)</b> <b>Orbit(PeA,ApA,Inc)</b> <b>Orbit(PeA,ApA,Inc,Mode)</b> <b>Orbit(PeA,ApA,Inc,Mode,GravTurnPitch)</b> <b>Orbit(PeA,ApA,Inc,Mode,GravTurnPitch,Abside)</b>	The complete orbit autopilot! All these forms of calls are valid. PeA and ApA are in km Inclination is in degrees, positive to launch northward, negative to launch southward. If not specified or less than launch latitude azimuth=90° will be used Mode =1 for upright, -1 for upside down GravTurnPitch is the initial gravity turn pitch in case automatic value is not satisfactory Abside is the target altitude of cutoff in case automatic value is not satisfactory (note this, may lead to less precision of the autopilot)	<b>NEW! HOT!!!</b>
<b>Defap(xx)</b>	Activates the Default Autopilots, input an integer number to choose which to activate (it's enough to count them in the HUD from left to right to get the right number)	<b>NEW!</b>
<b>Glimit(xx)</b>	When triggered will spend 250 seconds checking if the vehicle is accelerating more than xx*G. If so it throttles down the vehicle (warning, it will not throttle it back by itself)	<b>NEW!</b>

## ALL PARAMETERS – SCENARIO FILE

Only customizable parameters are specified here

ITEM	Explanation	Changed from Vinka?
<b>CONFIG_FILE</b> abc	Vehicle config file, path relative to orbiter root	No
<b>FAILURE_PROB</b> xx	Probability of failure of the vehicle	<b>NEW!</b>
<b>GUIDANCE_FILE</b> abc	Guidance file to load on startup	No
<b>TELEMETRY_FILE</b> abc	Reference telemetry file to be viewed in the MFD	<b>NEW!</b>
<b>COMPLEX</b>	If present activates the complex flight that: 1) randomizes engine performances. 2) Will create aerodynamic forces which will try to turn the rocket out of the gravity turn. If the wind angle gets too wide while drag is still significant (>500kN) the vehicle will be destroyed	<b>NEW! HOT!</b>
<b>GROWING_PARTICLES</b>	If present activates the growing particle effects (see dedicated chapter)	<b>NEW! HOT!</b>
<b>ALT_STEPS</b> xxx,yyy,zzz,ttt	[meters] Altitude Steps manual setting (See notes of the following section)	<b>NEW!</b>
<b>PEG_PITCH_LIMIT</b> xxx	[degrees] Peg Pitch Limit manual setting (See notes of the following section)	<b>NEW!</b>
<b>PEG_MC_INTERVAL</b> xxx	[seconds] Peg Major Cycle Interval (See notes of the following section)	<b>NEW!</b>
<b>DENY_IGNITION</b>	If this is present the stage won't reignite (it's used in automatic when a not reignitable stage is used)	<b>NEW!</b>

## SOME MORE NOTES

### The New Orbit Guidance Program

One of the main new features of Multistage2015 is without any doubt the new Orbit call for the guidance program, but how exactly does it work?

It takes the vehicle from ground to final orbit in full sequence, requiring as input basically only target perigee, target apogee and target inclination.

The first thing to know it's that the Orbit call divides the ascent in 4 phases, based on altitudes:

*The Null Phase, from 0 to first altitude Step:*

Nothing happens, the rocket simply climbs and clears the tower.

*The Roll Phase, from first altitude Step to second altitude Step:*

Roll program, note that for Orbit program if the roll isn't over at the reaching of the second altitude step it's not a big deal, everything will work anyway. The Roll program will turn the vehicle to the desired azimuth and will leave it at the Gravity Turn Initial Pitch.

*Initial Pitch Phase: from second altitude Step to third Altitude Step:*

In this phase the launch vehicle turns to the gravity turn initial pitch. In case the initial pitch is small it is advisable to raise the third altitude step in order to have a more smooth and realistic initial turn.

*Gravity Turn Phase: from third altitude Step to fourth Altitude Step:*

Here gravity turn takes place, it's an open loop phase, where the rocket is kept with proper roll and heading and the only relevance for pitch is the aoa.

*PEG Phase: from fourth altitude Step upward:*

This is the key step, that will take the rocket and bring it to its target orbit

By default, for Earth Altitude Steps are: 100m, 350m, 1.400m, 35.000m . For other bodies they are scaled on a planet mass proportion.

Altitude Steps can be managed through scenario file and/or the MFD.

In order to have an exact solution of the PEG equations the rocket always points toward a precise abside.

If in the orbit call no abside is specified, or 0 is specified, the abside is chosen via a default procedure as follows: if the target perigee is above 80 km then the target abside will be the perigee, otherwise it will be the apogee. I digged around orbiter apis but could not find an appropriate function to get the "almost zero drag" altitude, so the autocheck works smoothly only for Earth, if launching from another body significantly different from Earth it could be better to specify the abside in the orbit call (last parameter).

If instead an abside is specified in the call the guidance program will target the abside and relevant vertical speed for cutoff conditions. Usually this is useful for particular vehicles, like space shuttle was, whiche needs to have cutoff with a significant upward velocity at a certain altitude. Note that this usually leads the autopilot to be slightly less precise on the final absides reached.

The PEG major cycle has by default an interval of 0.1 seconds (for apollo was 10 seconds...), but also this value is customizable through scenario and/or the MFD.

Between one interval and the other the pitch is calculated as a linear variation of the parameters and the time, (please see also the original NASA document relevant to PEG). The estimation cycle of PEG has been a bit revisited and is based on DeltaV prediction that can be made quite exactly in orbiter. Please note that since the frequency is 10Hz, linear estimation is done only for 1 tenth of seconds, 10% of the overall time.

The last point is that in order to avoid out of reason behavior of the autopilot, the range of pitch at which PEG can turn the vehicle is limited. The value is called PEG Pitch Limit and by default is 35 degrees (plus or minus). This value can be modified as well via the MFD or via the proper input in the scenario file.

### Computational Evaluation of Gravity Turn

What is the gravity turn? It is called gravity turn the procedure of maneuvering the vehicle in order to maintain the direction of thrust aligned with the velocity vector. Why is it called "gravity turn"? because when you lift off for examples from Earth, you pitch down a few degrees your velocity vector will start to "fall" down towards Earth's surface. In order to excessive aerodynamic stress on the vehicle the vehicle itself shall be pitched down to follow the velocity vector triggering the velocity vector to fall even more and so on. The entire curve described in this phase is characterized by the first disturbance to the vertical attitude, the gravity turn initial pitch. A high disturbance will result in a more rapid gravity turn with a low apex of the described curve of trajectory. A small disturbance will result in a much slower rate of turn, letting the apex higher. For a rocket climbing to orbit the sooner it is possible to pitch down the better it is, but with planets with consistent atmosphere such as the Earth it cannot be done too early to avoid the increase of aerodynamic stress.

For this reason a particular attention has been paid to the gravity turn: when the scenario opens up the guidance computer of the ms2015 module takes all the parameters and tries to simulate the gravity turn until he found the minimum pitch value that leads the rocket to have an altitude of at least the fourth altitude step from which the PEG algorithm will take over. It then adds a 5% just for safety (if the rocket does not reach the proper altitude PEG will never start and the gravity turn will lead the rocket to crash back on earth at full thrust). BUT the computational evaluation of gravity turn is good but not perfect. It cannot know if for example engines are being throttled down for maxQ reasons, and each step is

so dependent on the other that even if the results are quite good they cannot be taken as gold, also because actually the best conditions at which the rocket is left at PEG takeover depends also on the final orbit we want to achieve: for a low orbit of 160x160 km we will want to be almost horizontal for maximum tangential acceleration. For an high orbit of 300x300 km we will want to be still pointing up and gaining some more vertical speed to get to 300km. Therefore the value computational calculated usually works, but is a "suggested" value, that can be optimized by the user. In the orbit call of the guidance program users can set this at their pleasure, in order to find the optimal value. I've made hundreds of tests about this, but since all the rockets are different and the firing sequences are different it cannot be made completely closed to the user inputs, and by specifying which is the "suggested" value the user can have an idea of which value shall be used for that particular orbit. If the user does not input anything the guidance computer will use the suggested value.

### Complex Flight

When the complex flight option is activated (both by scenario file and via the MFD) an estimation is made for each engine of the rocket (boosters included) which will set the parameters (amplitude and period) of thrust oscillation during the whole powered flight. You can monitor this oscillation for each engine through the MFD –THR screen where you can check the performance together with the thrust and the throttle. This is a very nice parameter to randomize the flights a bit, in order not to have the very same situation for the same scenario, but always a bit of difference.

If Complex Flight is active also an additional aerodynamic force will be applied to the vehicle, and will try to take its nose away from the gravity turn, as it is in reality. If the angle with the wind (AOA and Slip Angle) gets too wide while drag is still high (above 500 kN) the aerodynamic force will prevail and the vehicle will loose attitude control and destroy itself.

## Failures

Failure probability can be set only via scenario file. If higher than 0 a probability estimation is calculated at the beginning of the simulation. If a failure happens then some conditions are checked: if when the failure happens the first stage is running there is a fixed probability of 25% that the rocket will actually explode... so sit there and cross your fingers. In the other cases (not explosion or not the first stage running) one engine will be shut down and it will not be possible to restart it anymore. If Orbit program is activated, it will try to reach the orbit with the new situation.

## Telemetry

If Telemetry=1 will be set in the ini file, then each minute a telemetry file .txt with comma separated values will be saved, in particular the following parameters will included in the file:

Met – Altitude – Speed – Pitch – Thrust – Mass - Vertical Speed- Acceleration

Note that the file can anyway be saved and loaded manually through the MFD at anytime.

## Growing Particles Effect



A peculiar effect was implemented in Multistage2015 which is the growing particles effect. Basically size and growthrate of particle streams can change with the altitude changing of the vehicle.

Two parameters can be set in the ini file, as seen in .ini details above but to trigger the effect also the word GROWING\_PARTICLES must be present in the scenario file in the vehicle section, so by default

the effect is off. This is because the particle effect may affect graphic performance and it should be avoided to have it triggered by mistake.

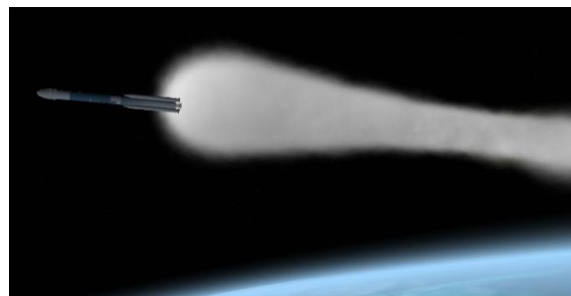
The formulas governing the effect are:

$$G.R.CurrAlt = G.R.SeaLvl + G.R.Factor * \log_{10} \left( \frac{Pressure_{SeaLevel}}{Pressure_{CurrentAltitude}} \right)$$

And

$$Size_{CurrAlt} = Size_{SeaLvl} + S.Factor * \log_{10} \left( \frac{Pressure_{SeaLevel}}{Pressure_{CurrAltitude}} \right)$$

Both can be set positive or negative so particles may decrease or increase in both size or growthrate with altitude increasing.



The position of particles emitting is also updated at size variation: in order to keep the particle ideally “tangent” to the engine emitting it if particle size increases then the particle stream is generated half of the size farther in engine direction, and this happens for the whole ascent.

If the effect is triggered particles updating happens with 1hz frequency, so once per second. In order to avoid visual interruptions of the particles flow a minimum time of overlap of the “old” and the “new” particle stream had to be placed. Currently the overlap time is set to 0.1s.

## Particles packaging

A new parameters for stages have been inserted which is the Particles\_packed\_to\_engine. This call allows to save a lot of graphic performances by merging the particle streams of a stage into a single



stream. The call explicitly refers to the engine to which the unified stream shall be referenced to.

For example a definition like this:

```
[...]
eng_1=(-1.8347,3.3135,-32.408)
eng_2=(-1.8347,-3.3135,-32.408)
eng_3=(1.8347,-3.3135,-32.408)
eng_4=(1.8347,3.3135,-32.408)
eng_5=(0,0,-32.408)
eng_diameter=2.7
ENG_PSTREAM1=growingcont
particles_packed_to_engine=5
[...]
```

Assign a single particle stream (named growingcont) to engine n.5 (as can be seen it is the center engine of the array).

However, engine arrays may not have a center engine. For example the space shuttle had 3 main engine so no center engine could be picked.

Therefore there is the possibility to set the single particle stream generation at the center of the engine arrays. This can be achieved by setting the `particles_packed_to_engine` call to a negative number. A call like the one in the example above can be set also to

```
Particles_packed_to_engine= -2
```

This will lead to have level of particles driven by engine 2 but particles generated from the center of the stage engines array.

### Developer Mode

By pressing [CTRL]+[SPACEBAR] the Developer Mode will be activated. This will be highlighted by a large red text that will show up on the screen.

In the developer mode you can reset the entire vehicle and reload the ini file by just pressing [SPACEBAR]. So if you want to see if your meshes are placed correctly, if the payload is rotated in the right way, if anything is as you would like it to be just keep open the ini file in your favourite editor. You can make changes, save the ini file, press [SPACEBAR] and the vehicle will be updated live.

Note: for live payloads the reset may result in some weird behavior. My suggestion is to set everything

with a payload not live, and once happy, add live=1 and reload.

What if I have a live payload and I need to add specific parameters for it in the scenario file?

In this case simply load the scenario with the live payload as it is. Once loaded close the simulation and get the CurrentState.scn file. It will contain the launcher and the payload attached to it, so you can modify directly the payload sections as you wish, rename the CurrentState.scn to whatever you please and launch it again to enjoy the ride!

### Log File

Please note that in the orbiter.log file a LOT of information will be noted during your flights. From mesh names, to engine positions, stage ignition, even failures calculation results. So if anything does not behave in the way you thought, please first give a check to the orbiter.log file, that can be a big help.

## Multistage 2015 MFD

Together with the vehicle module, an MFD mode module is provided.

To activate the MFD mode, once installed the addon in orbiter directory, open orbiter, and in the Launchpad go to Modules tab – Miscellaneous and activate Multistage2015\_MFD.

The Multistage2015\_MFD has a number of screens for setting, monitoring and interact with any multistage2015 vehicle.

The left column of buttons is always the same for every screen and it's basically the menu of the MFD. The right column of buttons represent the screen specific commands.

Let's see the details and the commands screen by screen.

### Flight Settings Screen - FST



Shows some general vehicle information and gives the possibility to manually set the altitude steps, the Peg Major Cycle Interval, the Peg Pitch Limit and toggles the Complex flight on/off. It also shows the calculated gravity turn initial pitch, for reference if a manual input wants to be set in the guidance program.

Recapping:

[ALT] – Sets the altitude steps, input must be XXX,YYY,ZZZ,TTT

[PMC] – Sets the Peg Major Cycle interval (input in seconds)

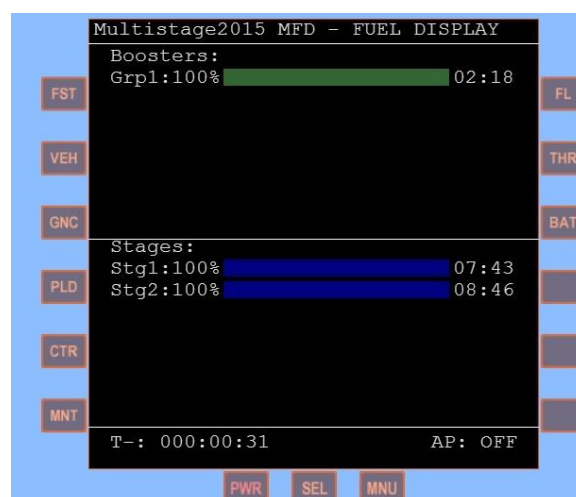
[PLM] – Sets Peg Pitch Limit (input in degrees)

[COM] – Toggle Complex Flight on / off

### Vehicle Screen – VEH

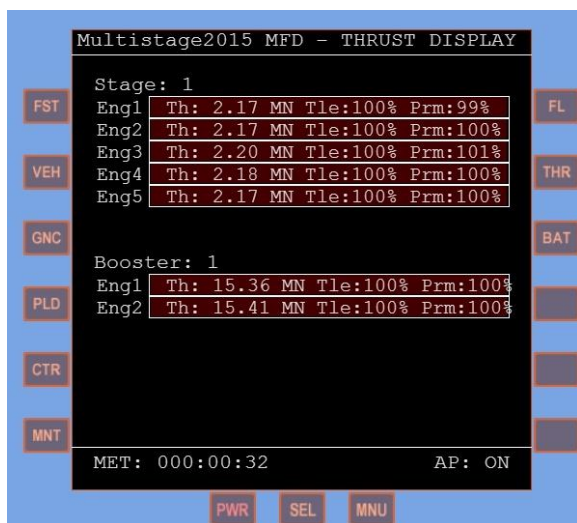
This screen contains actually 3 screens inside, that can be chose via the right column buttons:

The Fuel Screen FL:



Which shows fuel percentage and time remaining for all the tanks in the rocket.

## The Thrust Screen – THR:



Which shows engines thrust, throttle and performance

## And Batteries Screen - BAT



Which shows batteries status of each stage.

## Guidance Screen - GNC



This is a very important screen because it gives the opportunity to both check and interact with the guidance program.

You can move across the preset commands through the [UP] and [DN] buttons, delete a specific command through the [DEL] button or add a new one with the [ADD] button.

If you want to keep the guidance file once prepared you can simply press [SAV] and a guidance file will be saved in orbiter\_root\config\Multistage2015\Guidance\ directory with the following syntax:

Name of vehicle\_sysMJD\_GNC.txt

The [AP] button toggles the autopilot on or off.

Please note that before the launch, if you add a command with negative time it will set the Countdown to that time, i.e. adding

-30=orbit(180,220,51.5,1)

Will set the mission timer to T-30 if nothing has been set with a previous time.

### Payload Screen – PLD



This screen shows the information of the payloads onboard the vehicle.

[FAI] – Simulates keypress [F] so fairing or les jettison

[JET] – Simulates keypress [J] so booster, stage, interstage or payload jettison

### Control Screen – CTR



This screen gives the opportunity to check if the vehicle is pointing in the direction which is given by the autopilot and with what error.

Moreover, with the right column keys gives the opportunity to toggle particular attitude controls if

something is not working properly and the pilot needs to take over, or if the user simply wants to test himself in following the proper attitude.

[ATT] – Toggle Autopilot Controls for all the three axis

[PIT] – Toggle Autopilot Control for pitch

[YAW] – Toggle Autopilot Control for yaw

[ROL] – Toggle Autopilot Control for roll

### Monitor Screen - MNT



The monitor screen shows live-time graph of most of the parameters of telemetry.

It also can show with dimmed colors a preset telemetry values in order to have a reference to check.

The scale range is by default automatic but can be set manually through the appropriate button.

In particular the user can move through the times shown with the [UP] and [DN] buttons.

Each of the graphs can be turned off by pressing the [TOG] button

It is possible to manually set the range of each graph by pressing the [SET] button. Min,max format is required for this

[LD] – will load a telemetry file from the HD, please note that the path to the file shall be relative to orbiter\_root path

[SAV] – will save a telemetry file in orbiter\_root\Config\Multistage2015\Telemetry directory with the following syntax

Name of vehicle\_sysMJD\_TLM.txt

### External Reference Vessel

If we are sitting inside a non Multistage2015 vessel we can point the MFD to a Multistage2015 vessel inside the scenario and operate with it exactly as if we were in the Multistage launcher.

This comes particularly in handy if we are launching “live” payloads and we want to enjoy the ride from the flight deck of our payload and interact with the launcher at the same time.

Also it gives option to simulate a Mission Control Center from the ground.

In order to operate this, while we are sitting in the non multistage2015 vehicle we must open the Multistage2015\_MFD and click the [SEL] button on top left. A dialog will open and we have to input the name of the vehicle we want to stick the MFD to. Only names of Multistage2015 vessels currently present in the scenario will be accepted.



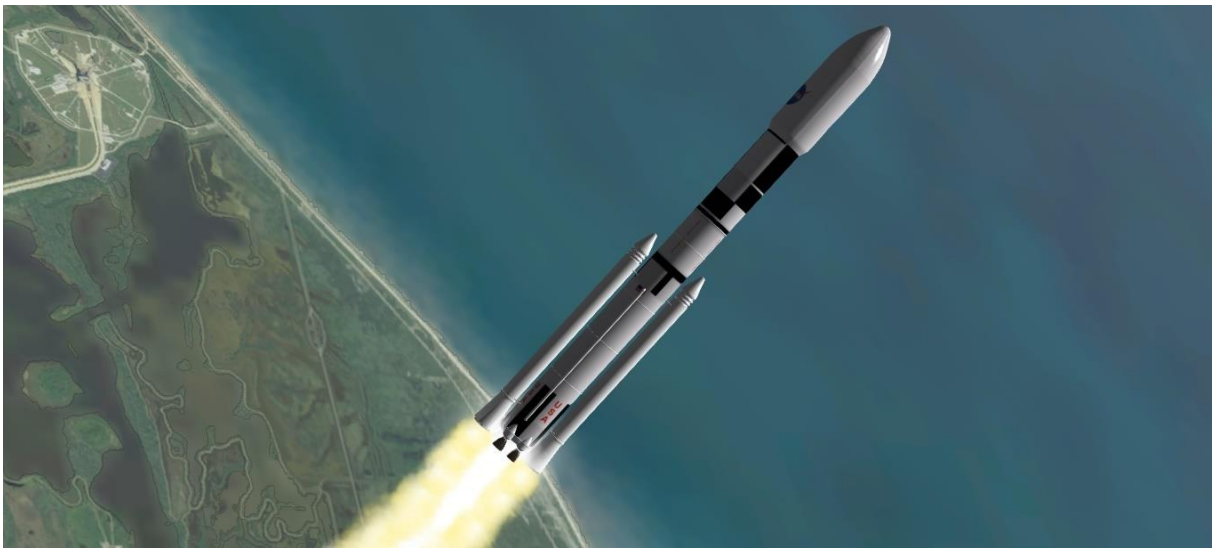
The fact that the MFD is not set to the current focus vessel has a graphical highlight because all the screen titles will become green with the reference vessel name shown.



It is possible to reset the external reference clicking the [RST] button in the Flight Settings Screen, in the lower right corner.

### SLS Heavy Lifter Example

Together with the package an example is provided. It is actually quite a realistic example, since it is based on the early design of SLS, and all the proper calculation relevant to mass, fuel and engine performance has been made. This has been fully developed by me, so consider it a present for having downloaded the Multistage2015 module 😊



### Thanks

Thanks to all the members of the OF community and to the Italian FOI community who helped me with their answers about the most difficult implementations of the module.

A particular thanks to some beta testers who patiently tried the module and pointed me towards the various glitches present during developing. To name just some of them (please those who are not included don't get disappointed): boogabooga, Interceptor, crisbeta, romanasul. 4throck.

Thanks to the patience and support of my girlfriend who has been asking me about this project as if it was a real life one, understanding my passion for it.

And as usual the biggest thanks goes to Dr. Martin Schweiger for creating Orbiter!

*Have fun up there!*

*Fred18*