

Mission opcodes for Wing Commander IV.

© 1991-95 by Origin Systems, Inc. BCP.

Revision 1.1

The following document is a list of commands used in WC4 missions to control/monitor events and send results back to gameflow. These commands are used in CHUNK_PROG portion of the mission files. Below is a sample program in the correct WC4 format. Note the lack of #define's and %INDEX's which have now been replaced by an '= prog_num' at the end of the program's name, like so:

```

PROG_name_of_program = prog_num
  long CMD_test_object_active, CAST_wingman
  long CMD_if_true_label, 'A'
  long CMD_destroy_object, CAST_player
  long CMD_return
  long CMD_label, 'A'
  long CMD_destroy_object, CAST_wingman
  CMD_terminator

```

The '= prog_num' format is also used for CHUNK CAST, MSGS, PART, AREA, & SPOT(= cast_num, = part_num, = area_num, = spot_num). The 'long' goes before each line except the first and last, from this point on the longs will be omitted from all sample code. Sample code will be in bold with the references to other implied lines of code and defined values in other chunks in italics. The above program is a basic routine to tell it's caller to detect if the 'CAST_wingman' is active, if so kill him, if not then attack 'CAST_player'.

Control

CMD_terminator

Program terminator. Must be the last line of every defined program in CHUNK_PROG and cannot appear more than once in each defined program.

CMD_return

Internal program terminator, used generally for staged events like:

```

PROG_name_of_program = prog_num
  CMD_test_object_active, CAST_wingman
  CMD_if_true_label, 'A'
  CMD_destroy_object, CAST_player
  CMD_return
  CMD_label, 'A'
  CMD_destroy_object, CAST_wingman
  CMD_terminator

```

Once the CMD_return is called it will no longer run the rest of the program. So that the caller of this program will only attempt to execute the first half of the program unless the conditions of the first two lines have not been met.

CMD_call_program, PROG_X

Transfers control of the program to *PROG_X*.
PROG_X must appear in the PROG_CHUNK before
being called by CMD_call_program. For example:

```
...  
PROG_first = prog_num  
    CMD_terminator  
...  
PROG_second = prog_num  
    CMD_terminator  
...  
PROG_third = prog_num  
    CMD_terminator  
...
```

PROG_first then cannot contain a line saying:
CMD_call_program, PROG_second
or:
CMD_call_program, PROG_third
PROG_second could call PROG_first. And PROG_third
could call either the first or second PROG's.

CMD_call_work_program

Transfers control to the program stored in the work register.
As with CMD_call_program the program that is being called
must appear before the program calling it. The program being
called must also be sent to the work register using
CMD_move_value_to_work. ie:

```
PROG_kill_player = prog_num  
    CMD_destroy_object, CAST_player  
    CMD_terminator
```

```
PROG_kill_wingman = prog_num  
    CMD_destroy_object, CAST_wingman  
    CMD_terminator
```

```
PROG_enemy_mission = prog_num  
    CMD_random_work_value, 1  
    CMD_goto_work_label
```

```
CMD_label, 0  
CMD_move_value_to_work, PROG_kill_player  
CMD_goto_label, 'A'  
CMD_return
```

```
CMD_label, 1  
CMD_move_value_to_work, PROG_kill_wingman  
CMD_label, 'A'  
CMD_call_work_program  
CMD_terminator
```

The caller of this program will run either of the two 'kill' programs depending upon the random value sent to the work register in line 1 of 'PROG_enemy_mission'.

CMD_end_mission, *status* Ends the mission in progress with the status *status* and returns control to gameflow. *status* is defined as any one of the following states: GAME, FINISHED, EJECTED, KILLED, ABORTED, INSTANT, and RESTART.

CMD_exit Used to exit the mission in progress. Used mainly for debugging.

CMD_label Used to delineate a branch entry point. Labels may be of any number value or character. Numbers may be typed directly while characters must be surrounded by single quotes and will be dereferenced prior to use (ie, 'A' = 65). The following

CMD_data, *SPOT_X* *SPOT_X* is defined in CHUNK_SPOT, It usually contains data pertinent to a previous command. In most cases, data spots need to be keyed to AREA_global in there definitions.

Work

CMD_move_value_to_work, *value* Moves a number value, *value*, to the work register.

CMD_move_flag_to_work, *FLAG_x* Moves the value stored in *FLAG_x* to the work register.

CMD_xchg_work_and_flag, *FLAG_x* Swaps the work register value with the value stored in *FLAG_x*.

CMD_move_result_to_work Moves the result of the previous operation to the work register. ie.:

PROG_enemy_death = prog_num
CMD_test_object_active, CAST_enemy_x
CMD_move_result_to_work
CMD_terminator

CMD_move_work_to_register, *REG_x* Used to move a value to a gameflow register *REG_x*.

CMD_random_work_value, *value* Moves a random value from 0 to *value* inclusive to the work register.

Math

CMD_add_value_to_work, *value* Adds *value* to the value stored in the work register.

CMD_add_flag_to_work, *FLAG_x* Adds the value in *FLAG_x* to the value in the work register.

CMD_add_value_to_flag, *FLAG_x* Adds the defined value of the flag in the mission file to the flag itself.

CMD_add_work_to_flag, <i>FLAG_x</i>	Adds the value of the work register to the value stored in <i>FLAG_x</i> .
CMD_sub_value_from_work, <i>value</i>	Subtracts <i>value</i> from the value in the work register.
CMD_sub_flag_from_work, <i>FLAG_x</i>	Subtracts the value of <i>FLAG_x</i> from the value in the work register.
CMD_sub_value_from_flag, <i>FLAG_x</i>	Subtracts the original defined value of <i>FLAG_x</i> in the mission file from the current value of the flag.
CMD_sub_work_from_flag, <i>FLAG_x</i>	Subtracts the value stored in the work register from <i>FLAG_x</i> .
CMD_or_work_with_value, <i>value</i>	Performs a boolean comparison with the value stored in the work register and <i>value</i> . Sets the work register to FALSE only if both values are FALSE. All other cases set the work register to TRUE.
CMD_or_work_with_flag, <i>FLAG_x</i>	Performs a boolean comparison with the value stored in the work register and <i>FLAG_x</i> . Sets the work register to FALSE only if both values are FALSE. All other cases set the work register to TRUE.
CMD_and_work_with_value, <i>value</i>	Performs a boolean comparison with <i>value</i> and the value stored in the work register. Sets the work register to TRUE only if both values are TRUE. Sets the work register to FALSE if one or the other is FALSE.
CMD_and_work_with_flag, <i>FLAG_x</i>	Performs a boolean comparison with <i>FLAG_x</i> and the value stored in the work register. Sets the work register to TRUE only if both values are TRUE. Sets the work register to FALSE if one or the other is FALSE.
CMD_xor_work_with_value, <i>value</i>	Performs a boolean comparison with <i>value</i> and the value stored in the work register. Sets the work register to FALSE if the values are equal. Sets the work register to TRUE if the values are unequal.
CMD_xor_work_with_flag, <i>FLAG_x</i>	Performs a boolean comparison with <i>FLAG_x</i> and the value stored in the work register. Sets the work register to FALSE if the values are equal. Sets the work register to TRUE if the values are unequal.
CMD_mul_work_by_value, <i>value</i>	Multiplies the value stored in the work register by <i>value</i> .
CMD_mul_work_by_flag, <i>FLAG_x</i>	Multiplies the value stored in the work register by the value stored in <i>FLAG_x</i> .
CMD_div_work_by_value, <i>value</i>	Divides the value stored in the work register by <i>value</i> .
CMD_div_work_by_flag, <i>FLAG_x</i>	Divides the value stored in the work register by the value stored in <i>FLAG_x</i> .

Conditionals

CMD_compare_work_to_value, <i>value</i>	Compares <i>value</i> to the value stored in the work register returning the results equal, less, more, less equal, and more equal.
CMD_compare_value_to_work, <i>value</i>	Compares the value stored in the work register to <i>value</i> returning the results equal, less, more, less equal, and more equal.
CMD_compare_work_to_flag, <i>FLAG_x</i>	Compares the value in <i>FLAG_index</i> to the value stored in the work register returning the results equal, less, more, less equal, and more equal.
CMD_compare_flag_to_work, <i>FLAG_x</i>	Compares the value in the work register to the value in <i>FLAG_x</i> returning the results equal, less, more, less equal, and more equal.
CMD_compare_result_to_value, <i>value</i>	Compares the result of the previous command to <i>value</i> , returning the results equal, less, more, less equal, and more equal.
CMD_test_flag, <i>FLAG_x</i>	Returns the TRUE or FALSE value of <i>FLAG_x</i> .
CMD_if_result_label, 'X'	If the current command is in progress, branch to the label 'X'.
CMD_if_not_result_label, 'X'	If the current command has completed, branch to the label 'X'.
CMD_branch_equal_label, 'X'	Branches to the label 'X' if the previous comparison returns an equal value.
CMD_branch_not_equal_label, 'X'	Branches to the label 'X' if the previous comparison returns a not equal value.
CMD_branch_less_label, 'X'	Branches to the label 'X' if the previous comparison returns a less value.
CMD_branch_more_label, 'X'	Branches to the label 'X' if the previous comparison returns a more value.
CMD_branch_less_equal_label, 'X'	Branches to the label 'X' if the previous comparison returns a less equal value.
CMD_branch_more_equal_label, 'X'	Branches to the label 'X' if the previous comparison returns a more equal value.
CMD_goto_label, 'X'	Forces a branch to the label 'X'.
CMD_goto_work_label	Branches to the label equal to the value in the work register.

Flags

CMD_set_flag, <i>FLAG_x</i>	Sets <i>FLAG_x</i> to TRUE (number value 1).
CMD_clear_flag, <i>FLAG_x</i>	Sets <i>FLAG_x</i> to FALSE (number value 0).
CMD_move_work_to_flag, <i>FLAG_x</i>	Moves the value stored in the work register to <i>FLAG_x</i> .
CMD_move_result_to_flag, <i>FLAG_x</i>	Moves the result of the previous operation to <i>FLAG_x</i> .
CMD_toggle_flag, <i>FLAG_x</i>	Changes <i>FLAG_x</i> to the opposite TRUE or FALSE condition.
CMD_increment_flag, <i>FLAG_x</i>	Adds 1 to <i>FLAG_x</i> . This can be used for flags containing alphanumeric characters because the characters are dereferenced prior to storing in the flag. A=65, B=66, etc.
CMD_decrement_flag, <i>FLAG_x</i>	Subtracts 1 from <i>FLAG_x</i> . As with CMD_increment_flag this can be used for alphanumeric characters.
CMD_set_debriefing_happened	Used to let the player land after a debriefing.

Areas

CMD_activate_area, <i>AREA_x</i>	Activates <i>AREA_x</i> . If the area is already active this command does nothing.
CMD_deactivate_area, <i>AREA_x</i>	Deactivates <i>AREA_x</i> . If the area is already inactive, this command does nothing.
CMD_test_area_active, <i>AREA_x</i>	Returns TRUE if <i>AREA_x</i> is active and FALSE if it is inactive.
CMD_get_work_range_from_area, <i>AREA_x</i>	Takes the distance in kilometers the caller is from the edge of <i>AREA_x</i> and moves it to the work register

Objects

CMD_activate_object, <i>CAST_x</i>	Activates <i>CAST_x</i> . If the object is already active, this command does nothing.
CMD_deactivate_object, <i>CAST_x</i>	Deactivates <i>CAST_x</i> . If the object is already inactive, this command does nothing.
CMD_test_object_active, <i>CAST_x</i>	Returns a TRUE value if <i>CAST_x</i> active, FALSE if it is inactive.
CMD_test_object_dead, <i>CAST_x</i>	Return TRUE if <i>CAST_x</i> has been killed, FALSE if it has not. This command does not differentiate between active and inactive objects.
CMD_explode_object, <i>CAST_x</i>	Causes <i>CAST_x</i> to suffer fatal damage. The object will run its death program.

CMD_get_work_range_from_object, CAST_x	Takes the distance from the caller to the specified object in kilometers and moves it to the work register.
CMD_test_object_exists, CAST_x	Returns a TRUE value if <i>CAST_x</i> is active and alive and FALSE if it is dead or inactive.
CMD_resurrect_object, CAST_x	Used to resurrect a dead or inactive object, <i>CAST_x</i> .
CMD_get_work_range_from_spot, SPOT_x	Takes the distance from the caller to <i>SPOT_x</i> in kilometers and moves it to the work register.
CMD_activate_rel_to_object, CAST_x	Used to activate <i>CAST_x</i> relative to an object other than the player. This line is followed by two <i>CMD_data</i> lines that reference the
CMD_jump_resurrect_object, CAST_x	Resurrects <i>CAST_x</i> , if he has died, with a nifty jump effect.
CMD_test_object_cloaked, CAST_x	Returns a TRUE or FALSE value depending on whether <i>CAST_x</i> is cloaked or not.
CMD_make_object_enemy, CAST_x	Forces the alignment of <i>CAST_x</i> to ENEMY in it's part.
CMD_make_object_neutral, CAST_x	Forces the alignment of <i>CAST_x</i> to NEUTRAL in it's part.
CMD_make_object_friend, CAST_x	Forces the alignment of <i>CAST_x</i> to FRIENDLY in it's part.

Mission Objectives

CMD_wait_seconds, value Used to halt the current program in progress for an amount of seconds determined by *value*. ie:

```

PROG_wait_program = prog_num
CMD_test_flag, FLAG_jumped
CMD_if_true_label, 'Z'
CMD_set_autopilot, FALSE
CMD_wait_seconds, value
CMD_if_not_complete, 'Z'
CMD_jump_from_point, CAST_enemy
CMD_set_flag, FLAG_jumped
CMD_set_autopilot, TRUE
CMD_label, 'Z'
CMD_terminator

```

NOTE: Since the program is stopped after this command is used it must appear in a program that is continuously running like an areas update or part's mission program slot. Also only

one timer can be running at a time since there is only one counter register.

CMD_wait_flag_seconds, FLAG_x This command is the same as CMD_wait_seconds only the number of seconds to wait is determined by *FLAG_x*.

CMD_take_off, CAST_x Used to initiate takeoff sequence with nifty camera angles. Usually it's the first line in the player's navigation program.

```
PROG_player_navigation = prog_num
  CMD_take_off, CAST_x
  CMD_radio_message, MSG_take_off
  CMD_goto_way_spot_area, SPOT_area_1
  CMD_radio_message, MSG_search_area
  CMD_goto_way_spot_area, SPOT_home
  CMD_radio_message, MSG_head_home
  CMD_landing, CAST_x
  CMD_radio_message, MSG_head_home
  CMD_terminator
```

CMD_landing, CAST_x Used to initiate the landing nifty sequence. usually it's the last line of the player's navigation program. ie.:

```
PROG_player_navigation = prog_num
  CMD_take_off, CAST_x
  CMD_radio_message, MSG_take_off
  CMD_goto_way_spot_area, SPOT_area_1
  CMD_radio_message, MSG_search_area
  CMD_goto_way_spot_area, SPOT_home
  CMD_radio_message, MSG_head_home
  CMD_landing, CAST_x
  CMD_radio_message, MSG_head_home
  CMD_terminator
```

CMD_destroy_object, CAST_x The caller of this program will attempt to destroy *CAST_x*.

CMD_defend_object, CAST_x The caller will attempt to defend *CAST_x* from all objects of opposite alignment, staying within 7* kilometers of the defended object.

CMD_defend_point, SPOT_x The caller will attempt to defend *SPOT_x* from all objects of opposite alignment, staying within 7* kilometers of the defended spot.

CMD_follow_leader, CAST_x Sets the caller to from on the wing of *CAST_x*. *CAST_x* must be of the same alignment as the caller and will from on the position defined in his *CHUNK_PART* entry.

CMD_jump_to_point, CAST_x Used to deactivate an object, *CAST_x*, with the nifty jump effect. If *CAST_x* is the player this must be preceded by a *CMD_set_next_mission* as it will end the mission in progress.

CMD_jump_from_point, CAST_x	Used to activate <i>CAST_x</i> with the nifty jump effect (tm). If <i>CAST_x</i> is already active this command does nothing.
CMD_goto_area, SPOT_x CMD_data, SPOT_y	<p>The caller of this command will attempt to travel to the area to which <i>SPOT_x</i> is relative. The area is considered to have been reached when the caller has passed within the sphere tolerance of <i>SPOT_x</i>.</p> <p>The next command must be a <i>CMD_data</i> with <i>SPOT_y</i> containing a global spot which is the velocity with which the caller will pass through the point <i>SPOT_x</i>.</p> <p>The velocity is defined as x, y, z components given in meters per second. Positive y is north, positive x is east, and positive z is up. If you wanted the ship to be headed North by Northwest and level, the spot would be defined something like this:</p> <p><i>SPOT_y</i> = spot_num spot(AREA_global, -75, 150, 0)</p> <p>Note the heading must be keyed to AREA_global. Most <i>CMD_data</i> spots are global.</p>
CMD_goto_way_point, SPOT_x CMD_data, SPOT_y	<p>The caller will attempt to travel to <i>SPOT_x</i>. The spot is considered to have been reached when the caller reaches a sphere 500 meters in radius.</p> <p>The next command must be a <i>CMD_data</i> with a velocity spot, <i>SPOT_y</i>, as described above under <i>CMD_goto_area</i>.</p>
CMD_go_close_to_point, SPOT_x CMD_data, SPOT_y	<p>The caller of this command will attempt to travel to a sphere around <i>SPOT_x</i>. The sphere is 500 meters in radius. This command is identical to <i>CMD_goto_way_point</i>.</p> <p>The next command must be a <i>CMD_data</i> with a velocity spot, <i>SPOT_y</i>, as described above under <i>CMD_goto_area</i>.</p>
CMD_goto_way_spot_area, SPOT_x CMD_data, SPOT_y	<p>Identical to <i>CMD_goto_area</i>, except that the area is considered to have been reached when the perimeter of the area has been crossed.</p> <p>The next command must be a <i>CMD_data</i> with a velocity spot, <i>SPOT_y</i>, as described above under <i>CMD_goto_area</i>.</p>
CMD_landing_clearance	Grants the player landing clearance to land on his assigned baseship without having to request clearance normally.
CMD_get_special_flag	This command was implemented for mission D2 of WC3. It returns a TRUE or FALSE value depending on whether the player has decided to go after Flint or not.

CMD_get_ship_damage, CAST_x Returns a value from 0 to 100 depending on the percent of damage *CAST_x* has sustained, 0 being no damage and 100 being full damage, although most WC3 ships were destroyed at 75.

CMD_set_ship_damage, CAST_x Sets the damage of *CAST_x* from 0 to 100 depending on the value in the work register, 0 being no damage and 100 being full damage, although most WC3 ships were destroyed at 75.

CMD_get_num_mines_around_spot, SPOT_x Counts the number of mines around *SPOT_x* and moves it to the work register. (For an example see mission J2 of WC3).

CMD_radio_message, MSG_x When used in the players navigation program displays the *MSG_x* in the notes field of the navigation map. ie.:

```

PROG_player_navigation = prog_num
  CMD_take_off, CAST_player
  CMD_radio_message, MSG_take_off
  CMD_goto_way_spot_area, SPOT_area_x
  CMD_radio_message, MSG_x
  CMD_goto_way_spot_area, SPOT_home
  CMD_radio_message, MSG_head_home
  CMD_landing, CAST_player
  CMD_radio_message, MSG_head_home
  CMD_terminator

```

MSG_x in this case would contain notes pertaining to the mission, something like "Destroy all enemy encountered" or "Rendezvous with shuttle". ie.:

```

MSG_x = msgs_num
  nav_msg("Disable transports")

```

Otherwise this command can also be used to play a normal radio message defined in the caller's profile or the MSGS chunk.

CMD_search_and_destroy, SPOT_x
CMD_data, SPOT_y

The caller will attempt to destroy anything of opposite alignment within a radius of *SPOT_x*.

The next command must be **CMD_data** with parameters listed in *SPOT_y*. The data is passed through the x,y,z coordinates of the spot. The x coordinate is the spherical range around *SPOT_x* within which the caller will **search** for enemies. The y coordinate is the cruising speed in meters per second that the caller will in it's sweeps of the search area.

CMD_set_leader, CAST_x

Sets the caller to take *CAST_x* as it's leader.

CMD_set_formation, SPOT_x	Sets a flying formation relative to their leader's velocity in meters <i>SPOT_x</i> being the x, y, z offset.
CMD_set_wingman, CAST_x	Tells the caller to take <i>CAST_x</i> as a wingman.
CMD_set_visible, CAST_x	Sets <i>CAST_x</i> to be visible on the nav map at all times.
CMD_set_not_visible, CAST_x	Sets <i>CAST_x</i> to be invisible on the nav map at all times.
CMD_set_follower, CAST_x	Sets <i>CAST_x</i> to follow the player when the player autopilots.
CMD_set_not_follower, CAST_x	Sets <i>CAST_x</i> to stop following the player when the player autopilots.
CMD_cease_fire, CAST_x	Sets <i>CAST_x</i> to stop firing.
CMD_fly_aimlessly, SPOT_x	The caller will fly without destination within a 7 kilometer sphere around <i>SPOT_x</i> .

Gameflow Commands

CMD_set_act, X	Used to set the next act in gameflow after a mission has completed. <i>X</i> will be a specific act defined in gameflow.
CMD_set_scene, X	Used to set the next scene in gameflow after a mission has completed. Usually run in the global exit program. <i>X</i> will be a specific gamestate defined in gameflow.
*CMD_set_mission	
CMD_get_gameflow_reg, REG_x	Get's the value of register <i>REG_x</i> from gameflow and moves it to the work register.
CMD_set_gameflow_reg, REG_x	Moves the value in the work register to the gameflow register <i>REG_x</i> .
CMD_play_movie, MSG_movie_x	Used to play a movie during a mission. This command ends the current mission in progress so must be preceded by a CMD_set_next_mission . The movie and mission are defined in CHUNK_MSGS . ie.:

```

PROG_planet_fall = prog_num
CMD_set_next_mission, MSG_mission_X
CMD_play_movie, MSG_movie_X
CMD_terminator

```

The string in **CHUNK_MSGS** should look like this:

```
MSG_movie_x = prog_num
```

nav_msg(movie_x)

movie_x is the filename of the movie to be played.

CMD_set_next_mission, MSG_mission_x Used to set the next mission to run after playing a movie. See above example. The string for the mission in **CHUNK_MSGS** should look like this.

**MSG_mission_x = prog_num
nav_msg(mission_x)**

mission_x is the filename of the mission to be played.

CMD_play_next_mission, MSG_mission_x Used to interrupt the mission in progress and play '**MSG_mission_x**' without returning to gameflow.

CMD_jump_to_next_mission, MSG_mission_x Used to interrupt the mission in progress and play '**MSG_mission_x**' with a nifty jump effect.

World Functions

***CMD_switch_to_camera, CAM_?** Forces the camera to a different position, usually defined in the missions world file.?

Mission Control

CMD_set_autopilot, TRUE Turns autopilot on - *TRUE*, or off - *FALSE*. If *TRUE* autopilot is enabled for the player under normal autopilot constraints. If *FALSE* autopilot is disabled until it is set to *TRUE* again regardless of enemy activity.

CMD_set_baseship, CAST_x Define's which cast member is treated as the baseship, which ship the player can land on. ie:

**PROG_carrier_birth = prog_num
CMD_set_baseship, CAST_x
CMD_terminator**

The caller of this program would then set *CAST_x* as the baseship. *SELF* can replace *CAST_x* if the caller is the baseship.

Debugging

CMD_crash Sends you to DOS with the debugging info at the top of the screen if it is enabled, used only for debugging.

CMD_nothing Used for debugging only.

Pseudo-commands -- renamed existing commands

CMD_if_true_label

CMD_branch_equal_label

CMD_if_false_label

CMD_branch_not_equal_label

LABEL

CMD_label

CMD_increment_work
CMD_decrement_work

CMD_add_value_to_work,
CMD_sub_value_to_work,

CMD_shift_left_work
CMD_shift_right_work

CMD_mul_work_by_value,
CMD_div_work_by_value,

CMD_if_not_complete
CMD_if_complete

CMD_if_result_label
CMD_if_not_result_label

Wing Commander 4 Mission Specification

Mission : MISSA1

Wingman / Ship Selection Notes

Maniac will be the only wingman for this mission.

Description / Choreography

At the beginning of the mission Maniac challenges Blair(A) to fight it out in virtual dogfight mode.

If Blair decides to accept the challenge(B) we go to:

Maniac sends a taunt(C) and then him and Blair go at it in virtual mode. When either Maniac(D) or Blair(E) takes the other to 100% damage they are attacked by what looks like border world fighters while still in virtual mode. At which point Maniac sends his "Holy shit" message(F). Then they will have to dodge the enemy while they're lasers power up for 60 seconds. After dispatching the enemy Maniac sends his "nice work" message(G) and they head for the Orlando Depot. Upon arriving at the depot they find it under attack by a Border Worlds marked ship, Seether, that immediately fires a torpedo at the depot destroying it. Seether then taunts the player(H), does his special move, and jumps out of the jump point before the player can react. Maniac then sends his "Son of a bitch" message(I).

If Blair decides to not accept Maniac's challenge(J) we go to:

Maniac and Blair are suddenly jumped by pirates while still in virtual mode. Maniac sends his "Holy shit" message(F). They will then have to dodge the fighters while in virtual mode waiting for their guns to recharge for 60 seconds. After dispatching the enemy Maniac sends his "nice work" message(G) and they head for the Orlando depot. Upon arriving at the depot they find it under attack by a Border Worlds marked ship, Seether, who immediately fires a torpedo at the depot destroying it. Seether then taunts the player(H), does his special move, and jumps out the jump point before he can react. Maniac then sends his "Son of a bitch" message(I).

Maniac and Blair will then go to the Blue Point depot(K).

Special Art / Objects

Virtual mode lasers. HUD virtual score status. Orlando and Blue Point depots.

Communication Events

(A) Maniac- "We're scheduled to hop a shuttle at the Orlando depot to make the jump to Sol. So to make it interesting, I had our ships rigged for virtual dogfight mode. I know you've always wanted to take a shot at me, so here's your chance. Our guns' power generators have been temporarily altered to fire non lethal blasts. Your HUD will show "virtual" damage on both our ships. First one with 100% is the loser. So we can finally see who's better with a flight stick. Whaddya say?..."

(B) Blair- "You're on, pal."

(C) Maniac- "Let's see if you still got it, "Farmer" Blair."

(D) Maniac- "Guess you been down on the farm too long, Colonel. See you at the Orlando depo--"

(E) Maniac- □So ya got lucky again. Just like that run on Kilrah. See ya at the Orlando depo--□

(F) Maniac- □Holy Shit! Switching to □battle-mode□! I just hope the auto-reconfig is fast enough--□

(G) Maniac- □Nice work, old-timer. Let□s head to Orlando. We got a shuttle to catch.□

(H) Seether- □What you see before you, Colonel Blair, is just one of what will be many victories for the Border worlds.□

(I) Maniac- □Son-of-a-bitch! Wish our ships could jump. I□d love to nail that bastard. Well we□ve got no choice, buddey. Nearest depot is Blue Point. We can grab a shuttle to HQ there.□

(J) Blair- □No time for that now, Maniac. Let□s head to Orlando.□

(K) Blue Point- □Welcome to Blue Point, gentlemen. You have clearance.□

Special Sound Effects

- Guns power-up sound effect after virtual mode.
- Clicking sound effect for when guns are disabled.

Additional Comments

None.

Nav Points

NAV ROUTE: Area 1, Area 2, Area 3

Area 1	Nav 1 (After virtual dogfight mode is over) [2x Razor]
Area 2	Orlando [Orlando depot, Banshee(Seether)]
Area 3	Blue Point [Blue Point depot, 2x Confed Arrow]

Wing Commander 4 Mission Specification

Mission : MISSL3

Success / Failure notes

Either rescuing a disabled Catscratch.
Or safely inserting Dekker in his pod onto the lead ship(Confed Cruiser) of a Confed convoy. The player will decide during the mission which mission he will attempt.

Wingman / Ship Selection Notes

During the briefing Catscratch is sent to tractor in a satellite at another nav point so he will not be on the active roster of wingmen.

Before this series of missions Pliers will offer the player another cloaking device. If the player decides to take the cloaking device there will be no wingman.

If the player does not break up the fight between Dekker and Maniac earlier in the series, Maniac will fly extremely poorly the next time he's selected as a wingman.

If the player does break up the fight between Maniac and Dekker, Dekkers performance will be denegated.

Description / Choreography

The player and his fly to the first Nav point were they receive a comm from the Intrepid informing them that Catscratch needs assistance(A). Dekker comms the player from his pod wanting an answer(B&C). The player will then decide whether to continue with his original mission(D) or go after Catscratch(E).

If he continues his original mission Dekker comms the player(F) then the player and his wingman head to the convoy. After destroying the fighter cover, they must successfully insert Dekker onto the largest ship in the convoy(Confed Cruiser). After 60 seconds if there are no enemy fighters in the area Dekker sends his mission complete message(G). If there are still enemy fighters left in the area he will abort the mission(H). The player and his wingman then return to the Intrepid(I&J).

If the player decides to go after Catscratch, Sosa comms the player(K), and a new area will appear on the nav map. Upon arriving at the area Catscratch comms the player(L) telling him to take out all of the enemy so he can eject. The enemy include a Confed Cruiser that is bearing down on catscratch preparing to tractor in his ship. The players wingman adds his two cents(M). After destroying all of the enemy in the area Catscratch ejects. If the player does not destroy the Cruiser before it gets too close to Catscratch, he will blow his own ship up(N) destroying himself and the Cruiser. The player and his wingman then return to the Intrepid(O&P).

Special Art / Objects

-Confed carrier undergoing repairs.

-The separate engine object to destroy on the carrier so the player can "take out his engines" as per briefing.

Communication Events

(A) Sosa- "Sir, Catscratch is in big trouble. He's recovered the satellite, but the enemy's tracked him down and leeches his craft. He's a sitting duck, sir..."

(B) Dekker- "What do you say, Colonel? We gonna scrub the original mission?"

(C) Dekker- "What's it gonna be, boss? It's gettin' pretty damn hot in here. Are the mission objectives 86'd?"

(D) Blair- "We've got too many people relying on us; we can't pull the plug. I'm sorry, but the kid will have to fend for himself."

(E) Blair- "Can't let the kid die. We're going after him."

(F) Dekker- "Your the boss."

(G) Dekker- "Lot's of goodies that go boom on this ship, and they're all ours, Colonel. We've alerted fleet command to send in their big boats and we're launching the pods. Scoop us up and take us home."

(H) Dekker- "We're buggin' out, Colonel. I'm not hangin my men ou to dry here."

(I) Sosa- "Let me play you a recording, sir." ***She will then play the next scene***
Catscratch- "I do this for the Border Worlds, and for Velina Sosa..." ***Return to*** Sosa- "He went down with his ship, sir. He gave everything. You have clearance."

(J) Sosa- "I'm downloading a new nav map to you, sir."

(K) Catscratch- "If you guys can clear out some of these bogies, I can eject."

(L) Wingman- "Colonel, if he ejects, we can demolish his ship. The enemy won't ever get there satillite back then."

(M) Catscratch- "I do this for the Border Worlds, and for Velina Sosa..."

(N) Intrepid- "Your cleared, sir. And... Thank you. From the bottum of my heart."

(O) Intrepid- "You're clear... sir..."

Special Sound Effects

None.

Additional Comments

Destroying the engines of the Carrier means that it will have to have a sister object that disables the Carrier after it is destroyed. This may not be implemented in time.

Nav Points

	NAV ROUTE: Area 1, Area 2, Area 3, Area 1 (Note- If the player decides to go after Catscratch the Nav route will be)	
	Area 1, Area 2, Area 4, Area 1	
Area 1	Intrepid [2x Banshee, Intrepid]	
Area 2	Nav 1 (Note- The player will decide at this point whether to go after Catscratch or continue with his mission objectives.)	original
Area 3	Convoy [2x Confed Transport, Confed Cruiser]	
Area 4	Catscratch [2x Hellcat, Confed Cruiser, Avenger(Catscratch)]	

Wing Commander 4 Mission Specification

Mission : MISSL4A (L-300)

Success/ Failure Notes

Success: First half of mission, Successful insertion and retrieval of Dekker. Both will rely on Blair destroying 100% of gaurdian fighters (no effect on overall outcome of mission). Second half o the mission, Disable the BL transport and provide Dekker enough time to take it over by destroying 6 of 6 Dragons.

Failure: Fail to capture the Black Lance cap ship.

Wingman / Ship Selection Notes

Standard wingmen selection.

Player must fly an Avenger.

Description / Choreography

Once again, Blair goes after a convoy, leading the advance forces to the convoy and inserting Dekker and the marines. This mission is essentially identical to Mission L3B. However, becasue there has now been some warning of the Border Worlds forces' intentions, the mission should be a little tougher (I.e. this new convoy might be surrounded by more fighters than the one in Mission L3).

The player must insert Dekker and then destroy all fighters in the action sphere. Upon destroying the last enemy, Blair recieves a comm from Dekker (A). Once objectives are met, Blair recieves a transmission from Wilford (C). Then Sosa immediately comes up on screen (D) followed by Dekker who is now back in the pod (E).

When the player arrives at the new nav point he sees a transport unlike any seen before, it is devoid of insignia. This ship is a Black Lance Transport and it has 3 Dragons as escort. The objective is to disable the transport and insert Dekker and his men in the Pods. When the capship is disabled, the player recieves comm (F) from Dekker. The player must then fire the pods and after a moment, he recieves another comm from Dekker (G). The player must now destroy 6 of 6 Black Lance Dragons to "buy" Dekker enough time which prompts comm (H) from Dekker.

Special Art / Objects

Black Lance Transport

Communication Events

(A) - "Lots of goodies that go boom on this ship, and they're all ours, Colonel. We've alerted fleet command to send in their big boats and we're launching the pods. Scoop us up and take us home."

(C) - "Colonel, we've got an urgent situation: Intelligence reports an extraordinary Confed ship in your vicinity, with a top secret cargo that could be highly critical to our efforts. It's imperative that you disable and take control of this vessel. Do not destroy it. We need what's on that ship."

(D) - "I'm downloading the new nav point for you, sir."

(E) - "We're ready for re-insertion, boss. I'm dy'in to see what's in that tub."

(F) - "Fire us in there, boss."

(G) - "We're in boss. Keep those bogies away from me while I do my thing."

(H) - "We're in control here. Looks like we've just bagged something very special."

Special Sound Effects

Nav Points

NAV ROUTE: Convoy - Intercept - Intrepid

Area 1 CONVOY
[1 Confed Cruiser, 2X Confed transports, 4X hellcats]

Area 2 INTERCEPT
[Black Lance Transport, 2 waves of 3X Dragons]

Area 3 INTREPID
[Intrepid, 2-4X Banshees]

Additional Comments

Wing Commander 4 Mission Specification

Mission : MISSLA

Success / Failure notes

Success is accomplished if the player and his wingman disable all of the new fighters in the testing grounds area.

Wingman / Ship Selection Notes

Before this series of missions Pliers will offer the player another cloaking device. If the player decides to take the cloaking device there will be no wingman.

Description / Choreography

The player and his wingman will attempt to disable the newly built confed fighters (Bearcats) in an asteroid belt. First he will need to destroy a Radar Bouy surrounded by Turret mines. He will have 60 seconds to destroy the radar bouy. If he does not destroy it in time, four more Hellcats will appear. The player will also encounter four extra Hellcats at the Testing Grounds area if he fails to destroy the Bouy in time. If the player decides to go straight to the Testing Grounds he will be intercepted by four more hellcats en-route as well as at the testing area. It will be necessary for the player to use his Leech gun to disable the new fighters. Once disabled The BWS Tango and it's escort appears and sends it's 'nice work' message (A). The player and his wingman then head back to the Intrepid (B&C).

Special Art / Objects

- Confed's new fighter (Bearcat).
- Asteroids.
- Leech guns.
- Drydock.
- Turret Mines.
- Radar Bouy

Communication Events

(A) BWS Tanago- "Nice work, Colonel. This is the BWS Tango and we're here to collect those birds. We'll take it from here."

(B) Intrepid- "You're cleared, Colonel. I hear Pliers is doing cartwheels across the launch deck, so take care not to run him over."

(C) Intrepid- "Better luck next time, Colonel. You're clear."

Special Sound Effects

- Leech guns.

Additional Comments

This is a remote second wing mission. During the mission the player will get updates from the two pilots he has picked for the wing2 remote mission, informing him of their progress. If they fail their mission the player will encounter another four Hellcats on the way back to the Intrepid.

Nav Points

NAV ROUTE: Area 1, Area 2, Area 1

Area 1	Intrepid [2x Banshee, Intrepid]
Area 2	Radar Bouy [2x Hellcats, 12x Turret Mine, Radar Bouy]
Area 3	Testing grounds [4x Bearcats, Drydock, 4xHellcats-Note: The four Hellcats will only appear if the Radar Bouy was not destroyed in time]

if (Local Flag < 5)
 ...

if (Flag name)
 ...
else
 ...
 ...

End generator