

AD-A244 931



1

**SOFTWARE DESIGN DOCUMENT
PVD CSCI (3)**

Volume 1 of 2 Sections 1.0 - 2.11.3.1

June, 1991



Prepared by:

BBN Systems and Technologies,
A Division of Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02138
(617) 873-3000 FAX: (617) 873-4315

Prepared for:

Defense Advanced Research Projects Agency (DARPA)
Information and Science Technology Office
1400 Wilson Blvd., Arlington, VA 22209-2308
(202) 694-8232, AUTOVON 224-8232

Program Manager for Training Devices (PM TRADE)
12350 Research Parkway
Orlando, FL 32826-3276
(407) 380-4518

92-00262



92 1 6 069

**APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED**

SOFTWARE DESIGN DOCUMENT PVD CSCI (3)

Volume 1 of 2 Sections 1.0 - 2.11.3.1

June, 1991

Prepared by:

BBN Systems and Technologies,
A Division of Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02138
(617) 873-3000 FAX: (617) 873-4315

Prepared for:

Defense Advanced Research Projects Agency (DARPA)
Information and Science Technology Office
1400 Wilson Blvd., Arlington, VA 22209-2308
(202) 694-8232, AUTOVON 224-8232

Program Manager for Training Devices (PM TRADE)
12350 Research Parkway
Orlando, FL 32826-3276
(407) 380-4518



Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution	
Availability	
Dist	Availability
A-1	

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

REPORT DOCUMENTATION PAGE

Form Approved
OPM No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Information and Regulatory Affairs, Office of Management and Budget, Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE June 1991	3. REPORT TYPE AND DATES COVERED Software Design Document	
4. TITLE AND SUBTITLE Software Design Document PVD CSCI (3)			5. FUNDING NUMBERS Contract Numbers: MDA972-89-C-0060 MDA972-89-C-0061	
6. AUTHOR(S) Author not specified.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Bolt Beranek and Newman, Inc. (BBN) Systems and Technologies; Advanced Simulation 10 Moulton Street Cambridge, MA 02138			8. PERFORMING ORGANIZATION REPORT NUMBER Advanced Simulation #: 9106	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency (DARPA) 3701 North Fairfax Drive Arlington, VA 22203-1714			10. SPONSORING/MONITORING AGENCY REPORT NUMBER DARPA Report Number: None.	
11. SUPPLEMENTARY NOTES None				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Distribution Statement A. Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE Distribution Code: A	
13. ABSTRACT (Maximum 200 words) A Simulation Network (SIMNET) project Software Design Document that describes the Plan View Display (PVD) Computer Software Configuration Item (CSCI number 3) of the SIMNET hardware and software training system for vehicle crew training and operational training.				
14. SUBJECT TERMS SIMNET Software Design Document for the PVD CSCI (CSCI 3).			15. NUMBER OF PAGES	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Same as report.	

Table of Contents

1	INTRODUCTION: PLAN VIEW DISPLAY (PVD)	1
1.1	BACKGROUND.....	1
1.2	EXTERNAL INTERFACES.....	1
1.3	INTERNAL STRUCTURE.....	5
1.4	CONFIGURATION AND CONFIGURATION MANAGEMENT.....	7
1.5	TERMINOLOGY AND DOCUMENTATION.....	8
1.5.1	Terms and Abbreviations	8
1.5.2	References	9
2	PVD CSC DESCRIPTIONS	10
2.1	MENU HANDLING CSC DESCRIPTION	13
2.1.1	Initial Menu Handling CSC Description	14
2.1.1.1	menu.c CSU Description (/simnet/pvd/lib/pvdiface.a)	14
2.1.1.1-1	init_menu_params	15
2.1.1.1-2	init_menus	15
2.1.1.1-3	init_menu.....	15
2.1.1.1-4	set_entry_locations.....	15
2.1.1.1-5	draw_menu	16
2.1.1.1-6	draw_disabled_backgnd	16
2.1.1.1-7	make_title	16
2.1.1.1-8	init_menu_window.....	17
2.1.1.1-9	menu_select.....	17
2.1.1.1-10	handle_menu	17
2.1.1.1-11	chg_status_of_menu_grp	17
2.1.1.1-12	change_phantom_status	18
2.1.1.1-13	change_logger_status	18
2.1.1.1-14	change_dataprobe_status.....	18
2.1.1.1-15	set_menu_icon_scale.....	19
2.1.1.1-16	draw_setup_menu.....	19
2.1.1.1-17	draw_route_menu.....	19
2.1.1.1-18	draw_catc_menu.....	19
2.1.1.1-19	draw_feature_menu.....	19
2.1.1.1-20	draw_nlos_courses_menu	20
2.1.1.1-21	draw_overlay_menu	20
2.1.1.1-22	get_current_menu.....	20
2.1.1.1-23	init_event_flag_strings.....	20
2.1.1.1-24	init_database_strings	20

	2.1.1.1-25	flag_activate_menu.....	21
	2.1.1.1-26	flag_nlos_menu.....	21
	2.1.1.1-27	flag_stealth_menu.....	21
	2.1.1.1-28	flag_catc_menu.....	21
	2.1.1.1-29	display_nlos_menu.....	21
	2.1.1.1-30	display_stealth_menu.....	21
	2.1.1.1-31	display_activate_menu.....	21
	2.1.1.1-32	display_catc_menu.....	22
	2.1.1.1-33	set_default_menu.....	22
	2.1.1.1-34	draw_default_menu.....	22
2.1.2	Menu Input CSC Description.....		23
2.1.2.1	menu_func.c CSU Description (/simet/pvd/lib/pvdfiface.a).....		23
	2.1.2.1-1	nop.....	23
	2.1.2.1-2	menu_zoom_in.....	23
	2.1.2.1-3	menu_zoom_out.....	23
	2.1.2.1-4	menu_pan.....	24
	2.1.2.1-5	menu_zoom_125.....	24
	2.1.2.1-6	menu_zoom_50.....	24
	2.1.2.1-7	menu_zoom_25.....	24
	2.1.2.1-8	menu_top_level.....	24
	2.1.2.1-9	menu_display_zoom.....	24
	2.1.2.1-10	menu_previous_view.....	24
	2.1.2.1-11	menu_next_view.....	25
	2.1.2.1-12	menu_save_zoom.....	25
	2.1.2.1-13	menu_cross_section.....	25
	2.1.2.1-14	menu_colin.....	25
	2.1.2.1-15	menu_ruler.....	25
	2.1.2.1-16	menu_help.....	25
	2.1.2.1-17	menu_track.....	26
	2.1.2.1-18	menu_test_flag.....	26
	2.1.2.1-19	menu_teleport.....	26
	2.1.2.1-20	menu_area.....	26
	2.1.2.1-21	menu_veh_to_veh.....	26
	2.1.2.1-22	menu_pt_to_veh.....	27
	2.1.2.1-23	menu_seg_pt_to_pt.....	27
	2.1.2.1-24	menu_pt_to_pt.....	27
	2.1.2.1-25	menu_emb.....	27
	2.1.2.1-26	menu_space.....	27
	2.1.2.1-27	menu_clear.....	27
	2.1.2.1-28	menu_intervis_clear.....	28
	2.1.2.1-29	menu_stealth_teleport.....	28

2.1.2.1-30	menu_stealth_teleport_and_	
	azimuth	28
2.1.2.1-31	menu_stealth_attach.....	28
2.1.2.1-32	menu_stealth_mimic	28
2.1.2.1-33	menu_dl_play.....	29
2.1.2.1-34	menu_dl_record	29
2.1.2.1-35	menu_dl_go.....	29
2.1.2.1-36	menu_quit.....	29
2.1.2.1-37	menu_stop.....	29
2.1.2.1-38	menu_freeze.....	29
2.1.2.1-39	menu_fast_forward	30
2.1.2.1-40	menu_normal_speed	30
2.1.2.1-41	menu_forward_abs.....	30
2.1.2.1-42	menu_rewind	30
2.1.2.1-43	menu_forward_rel.....	30
2.1.2.1-44	menu_backward_rel.....	30
2.1.2.1-45	menu_top_menu_2.....	31
2.1.2.1-46	menu_connect.....	31
2.1.2.1-47	menu_overlay.....	31
2.1.2.1-48	menu_operations	31
2.1.2.1-49	menu_intelligence	31
2.1.2.1-50	menu_fire_support.....	31
2.1.2.1-51	menu_tac_air.....	31
2.1.2.1-52	menu_engineer.....	31
2.1.2.1-53	menu_red.....	31
2.1.2.1-54	menu_blue.....	32
2.1.2.1-55	menu_black	32
2.1.2.1-56	menu_green	32
2.1.2.1-57	menu_thin	32
2.1.2.1-58	menu_thick.....	32
2.1.2.1-59	menu_dashed	32
2.1.2.1-60	menu_nodify.....	32
2.1.2.1-61	menu_not_modify.....	32
2.1.2.1-62	menu_display	32
2.1.2.1-63	menu_100x.....	32
2.1.2.1-64	menu_50x	33
2.1.2.1-65	menu_20x	33
2.1.2.1-66	menu_10x	33
2.1.2.1-67	menu_5x.....	33
2.1.2.1-68	menu_2x.....	33
2.1.2.1-69	menu_1x.....	34
2.1.2.1-70	menu_symbolic.....	34
2.1.2.1-71	menu_10K_m.....	34

2.1.2.1-72	menu_1k_m	34
2.1.2.1-73	menu_500_m	34
2.1.2.1-74	menu_250_m	34
2.1.2.1-75	menu_60_m.....	35
2.1.2.1-76	menu_20_m.....	35
2.1.2.1-77	menu_10_m.....	35
2.1.2.1-78	menu_5_m	35
2.1.2.1-79	menu_set_view_height	35
2.1.2.1-80	menu_set_target_height.....	35
2.1.2.1-81	menu_view_range.....	36
2.1.2.1-82	menu_toggle_intervis_align.....	36
2.1.2.1-83	menu_set_all_align.....	36
2.1.2.1-84	menu_set_opposing_align.....	36
2.1.2.1-85	menu_terrain	36
2.1.2.1-86	menu_icon.....	36
2.1.2.1-87	menu_nlos_hazards	37
2.1.2.1-88	menu_set_altitude_hazard	37
2.1.2.1-89	menu_intervis.....	37
2.1.2.1-90	menu_mouse_help	37
2.1.2.1-91	menu_zone	37
2.1.2.1-92	menu_interactive.....	37
2.1.2.1-93	menu_xy_plot.....	37
2.1.2.1-94	menu_top_menu_1.....	37
2.1.2.1-95	menu_nlos.....	38
2.1.2.1-96	menu_stealth	38
2.1.2.1-97	menu_contour.....	38
2.1.2.1-98	menu_toggle_shade	38
2.1.2.1-99	menu_set_exercise	38
2.1.2.1-100	menu_grid.....	38
2.1.2.1-101	menu_connect_pt.....	39
2.1.2.1-102	menu_dont_connect_pt.....	39
2.1.2.1-103	menu_connect_dl	39
2.1.2.1-104	menu_disconnect_dl	39
2.1.2.1-105	menu_connect_dp	39
2.1.2.1-106	menu_dont_connect_dp	39
2.1.2.1-107	menu_done.....	40
2.1.2.1-108	menu_logging_on	40
2.1.2.1-109	menu_logging_off.....	40
2.1.2.1-110	menu_clear_tanks	40
2.1.2.1-111	menu_save_route	40
2.1.2.1-112	menu_route_redraw	40
2.1.2.1-113	menu_recall_route.....	40
2.1.2.1-114	menu_mode_done.....	41

2.1.2.1-115	goto_top_menu1.....	41
2.1.2.1-116	back_to_menu1.....	41
2.1.2.1-117	back_to_menu2.....	41
2.1.2.1-118	menu_catc_vehicle	41
2.1.2.1-119	menu_catc_mover_oblique.....	41
2.1.2.1-120	menu_catc_mover_frontal.....	41
2.1.2.1-121	menu_catc_personnel	41
2.1.2.1-122	menu_catc_place_tgt.....	42
2.1.2.1-123	menu_move_tgt_or_pt.....	42
2.1.2.1-124	menu_save_target_list	42
2.1.2.1-125	menu_recall_target_list	42
2.1.2.1-126	menu_list_catc_files	42
2.1.2.1-127	menu_catc_transfer_file	42
2.1.2.1-128	menu_delete_catc_file.....	42
2.1.2.1-129	menu_show_tgt_list.....	43
2.1.2.1-130	menu_catc_remove_tgt	43
2.1.2.1-131	menu_catc_clear_tgt_list.....	43
2.1.2.1-132	menu_symbol_create.....	43
2.1.2.1-133	menu_symbol_edit	43
2.1.2.1-134	menu_overline_create.....	43
2.1.2.1-135	menu_overline_newfeatures	44
2.1.2.1-136	menu_overline_newshape.....	44
2.1.2.1-137	menu_over_save.....	44
2.1.2.1-138	menu_over_read.....	44
2.1.2.1-139	menu_over_list	44
2.1.2.1-140	menu_over_delete	44
2.1.2.1-141	menu_over_transfer.....	44
2.1.2.1-142	menu_over_edit.....	45
2.1.2.1-143	menu_checkpt.....	45
2.1.2.1-144	menu_coordpt.....	45
2.1.2.1-145	menu_conpt.....	45
2.1.2.1-146	menu_linkpt.....	45
2.1.2.1-147	menu_relpt	45
2.1.2.1-148	menu_passpt	45
2.1.2.1-149	menu_deletept.....	46
2.1.2.1-150	menu_deleteov.....	46
2.1.2.1-151	menu_reduceov.....	46
2.1.2.1-152	menu_moveov	46
2.1.2.1-153	menu_addtextov	46
2.1.2.1-154	menu_eraseov.....	46
2.1.2.1-155	menu_redrawov	46
2.1.2.1-156	menu_resetov	47
2.1.2.1-157	menu_change_format.....	47

2.1.2.1-158	do_menu_event.....	47
2.1.2.1-159	menu_event1.....	47
2.1.2.1-160	menu_event2.....	47
2.1.2.1-161	menu_event3.....	47
2.1.2.1-162	menu_event4.....	48
2.1.2.1-163	menu_event5.....	48
2.1.2.1-164	menu_event6.....	48
2.1.2.1-165	menu_event7.....	48
2.1.2.1-166	menu_event8.....	48
2.1.2.1-167	menu_select_vehicle	48
2.1.2.1-168	menu_side.....	48
2.1.2.1-169	menu_dump_table.....	49
2.1.2.1-170	menu_detail	49
2.1.2.1-171	menu_activate.....	49
2.1.2.1-172	menu_activate_1.....	49
2.1.2.1-173	menu_activate_2.....	49
2.1.2.1-174	menu_activate_3.....	49
2.1.2.1-175	menu_activate_4.....	50
2.1.2.1-176	menu_activate_5.....	50
2.1.2.1-177	menu_activate_6.....	50
2.1.2.1-178	menu_activate_7.....	50
2.1.2.1-179	menu_activate_8.....	50
2.1.2.1-180	menu_activate_9.....	51
2.1.2.1-181	menu_activate_10	51
2.1.2.1-182	menu_activate_vehicle_mode	51
2.1.2.1-183	menu_deactivate_vehicle_mode	51
2.1.2.1-184	menu_altitude_1	51
2.1.2.1-185	menu_altitude_2.....	51
2.1.2.1-186	menu_altitude_3.....	52
2.1.2.1-187	menu_altitude_4.....	52
2.1.2.1-188	menu_altitude_5.....	52
2.1.2.1-189	menu_altitude_6.....	52
2.1.2.1-190	menu_altitude_7.....	52
2.1.2.1-191	menu_altitude_8.....	52
2.1.2.1-192	menu_altitude_9.....	53
2.1.2.1-193	menu_altitude_10.....	53
2.1.2.1-194	menu_alt_abs	53
2.1.2.1-195	menu_alt_rel	53
2.1.2.1-196	menu_feature_object_info.....	53
2.1.2.1-197	menu_feature_place_object.....	53
2.1.2.1-198	menu_feature_move_object.....	54
2.1.2.1-199	menu_feature_remove_object.....	54
2.1.2.1-200	menu_feature_save_file.....	54

	2.1.2.1-201	menu_feature_recall_file.....	54
	2.1.2.1-202	menu_feature_list_files	54
	2.1.2.1-203	menu_feature_delete_file	54
	2.1.2.1-204	menu_feature_clear_object_list	55
	2.1.2.1-205	menu_feature_redraw_objects	55
	2.1.2.1-206	menu_send_features	55
	2.1.2.1-207	menu_dont_send_features.....	55
	2.1.2.1-208	menu_feature_M977	55
	2.1.2.1-209	menu_feature_Ben_Franklin.....	55
	2.1.2.1-210	menu_feature_mi28	56
	2.1.2.1-211	menu_feature_new	56
	2.1.2.1-212	menu_view_ground_vehicles.....	56
	2.1.2.1-213	menu_view_platoons	56
	2.1.2.1-214	menu_view_companies.....	56
	2.1.2.1-215	menu_view_battalions	56
	2.1.2.1-216	menu_view_sorties.....	57
	2.1.2.1-217	menu_view_flights.....	57
	2.1.2.1-218	menu_icon_background	57
	2.1.2.1-219	menu_icon_no_background.....	57
2.1.3		Menu Display Processing CSC Description	58
	2.1.3.1	edit_menu.c	58
		2.1.3.1-1 cross_menu_entry_on	58
		2.1.3.1-2 cross_menu_entry_off.....	58
		2.1.3.1-3 set_format_utm.....	59
		2.1.3.1-4 set_format_xy.....	59
		2.1.3.1-5 grid_menu_entry_on.....	59
		2.1.3.1-6 grid_menu_entry_off	59
		2.1.3.1-7 contour_menu_entry_on.....	59
		2.1.3.1-8 contour_menu_entry_off	59
		2.1.3.1-9 help_menu_entry_on.....	59
		2.1.3.1-10 help_menu_entry_off.....	60
		2.1.3.1-11 shading_menu_entry_on	60
		2.1.3.1-12 shading_menu_entry_off.....	60
		2.1.3.1-13 choose_distinguished	60
	2.1.3.2	menu_fea.c.....	61
		2.1.3.2-1 check_mark	61
		2.1.3.2-2 edit_menu_strings.....	61
		2.1.3.2-3 edit_entry.....	62
		2.1.3.2-4 change_menu_func.....	62
		2.1.3.2-5 edit_menu_func.....	62

2.2	ICONS CSC DESCRIPTION	63
2.2.1	Icon Draw and Display CSC Description	64
2.2.1.1	graph.c	64
2.2.1.1-1	set_distinguished	64
2.2.1.1-2	clear_distinguished	65
2.2.1.1-3	view_ground_vehicles	65
2.2.1.1-4	view_platoons	65
2.2.1.1-5	view_companies	65
2.2.1.1-6	view_battalions	65
2.2.1.1-7	view_sorties	65
2.2.1.1-8	view_flights	65
2.2.1.1-9	query_view_mode	65
2.2.1.1-10	Update_Graphics_Display	66
2.2.1.2	icon_fea.c	66
2.2.1.3	dir_fire.c	66
2.2.1.3-1	do_impact	66
2.2.1.3-2	save_impact_info	67
2.2.1.3-3	show_firing_line	67
2.2.1.3-4	check_firing_duration	67
2.2.1.3-5	decode_ammo_type	68
2.2.1.3-6	decode_generic_ammo_type	68
2.2.1.3-7	decode_specific_ammo_type	68
2.2.1.3-8	decode_result	69
2.2.1.4	indir_fire.c	69
2.2.1.4-1	init_bursts	69
2.2.1.4-2	do_direct_fire	69
2.2.1.4-3	report_ifire_to_screen	70
2.2.1.4-4	pr_ifire_map_coords	70
2.2.1.4-5	draw_bursts	70
2.2.1.4-6	create_all_burst_segs	71
2.2.1.4-7	create_burst_segment	71
2.2.1.4-8	save_indir_fire_info	71
2.2.1.4-9	check_burst_duration	71
2.2.1.4-10	erase_all_bursts	72
2.2.1.4-11	check_det	72
2.2.1.4-12	set_firefinder	72
2.2.1.5	class.c	72
2.2.1.5-1	classify_object	72
2.2.1.5-2	classify_vehicle	73
2.2.1.5-3	classify_ground_vehicle	73
2.2.1.5-4	classify_air_vehicle	74
2.2.1.5-5	classify_water_vehicle	74
2.2.1.5-6	knots_object	74

2.2.1.6	lase.c.....	74
	2.2.1.6-1 lase.....	75
	2.2.1.6-2 display_lase.....	75
	2.2.1.6-3 draw_laser.....	75
	2.2.1.6-4 save_laser_info.....	75
	2.2.1.6-5 show_laser_beam.....	76
	2.2.1.6-6 check_lasing_duration.....	76
2.2.1.7	effects.c.....	76
	2.2.1.7-1 init_effects.....	76
	2.2.1.7-2 create_puff_seg.....	76
2.2.2	Icon Detail CSC Description.....	77
2.2.2.1	stat_change.c.....	77
	2.2.2.1-1 display_status_change.....	77
	2.2.2.1-2 decode_damage_cause.....	78
	2.2.2.1-3 decode_repair_cause.....	78
2.2.3	Initial Icons CSC Description.....	79
2.2.3.1	icon.c.....	79
	2.2.3.1-1 find_icon.....	79
	2.2.3.1-2 get_icon_from_bumper.....	80
	2.2.3.1-3 create_icon.....	80
	2.2.3.1-4 erase_old_tank.....	80
	2.2.3.1-5 create_smoke_seg.....	80
	2.2.3.1-6 create_minefield_flag.....	81
	2.2.3.1-7 create_hl_seg.....	81
	2.2.3.1-8 create_bore_seg.....	81
	2.2.3.1-9 create_plane_segs.....	81
	2.2.3.1-10 create_hull_segs.....	82
	2.2.3.1-11 create_turret_segs.....	82
	2.2.3.1-12 create_m2_hull_segs.....	82
	2.2.3.1-13 create_m2_turret_segs.....	82
	2.2.3.1-14 create_static_stripe_segs.....	83
	2.2.3.1-15 create_faad_turret_segs.....	83
	2.2.3.1-16 create_rwa_body_segs.....	83
	2.2.3.1-17 create_rwa_prop_segs.....	83
	2.2.3.1-18 create_rwa_hl_segs.....	84
	2.2.3.1-19 create_stealth_segs.....	84
	2.2.3.1-20 create_bfit_segs.....	84
	2.2.3.1-21 create_bfit_stripe.....	85
	2.2.3.1-22 create_bfit_hl_segs.....	85
	2.2.3.1-23 display_attachment.....	85
	2.2.3.1-24 create_attachment_icon.....	86
	2.2.3.1-25 delete_attachment_icon.....	86
	2.2.3.1-26 reinit_all_icons.....	86

2.2.3.1-27	delete_icon	86
2.2.3.1-28	move_icon.....	87
2.2.3.1-29	check_vehicle_type.....	87
2.2.3.1-30	swap_colors.....	88
2.2.3.1-31	assign_colors	88
2.2.3.1-32	highlight_icon.....	88
2.2.3.1-33	highlight_selected_icon	89
2.2.3.1-34	unhighlight_selected_icon	89
2.2.3.1-35	draw_tank	89
2.2.3.1-36	init_icon_colors.....	90
2.2.3.1-37	init_icons	90
2.2.3.1-38	set_distinguished_colors	90
2.2.3.1-39	set_other_colors	90
2.2.3.1-40	make_color_array	91
2.2.3.1-41	draw_trail.....	91
2.2.3.1-42	erase_trail.....	91
2.2.3.1-43	init_trail.....	91
2.2.3.1-44	zoom_icon.....	92
2.2.3.1-45	top_level_zoom_icon	93
2.2.3.1-46	pre_zoom_icon	93
2.2.3.1-47	calc_scale_factor.....	93
2.2.3.1-48	scale_icon	94
2.2.3.1-49	change_icon_scaling.....	94
2.2.3.1-50	calc_screen_scales.....	95
2.2.3.1-51	erase_all_tanks.....	95
2.2.3.1-52	erase_all_trails	95
2.2.3.1-53	erase_whole_trail.....	95
2.2.3.1-54	force_redraw_of_icons	96
2.2.3.1-55	redraw_trail	96
2.2.3.1-56	get_icon_scale	96
2.2.3.1-57	calc_initial_icon_size.....	97
2.2.3.1-58	zoomed_in.....	97
2.2.3.1-59	set_symbolic_mode.....	97
2.2.3.1-60	create_symbolic_icons	97
2.2.3.1-61	get_true_icon_scale	98
2.2.3.1-62	icon_get_total_scale.....	98
2.2.3.1-63	draw_marker.....	98
2.2.3.1-64	draw_lane	99
2.2.3.1-65	draw_minefield	99
2.2.3.2	detail.c	99
2.2.3.2-1	detail_display	100
2.2.3.2-2	word_print	100
2.2.3.2-3	get_object_from_guise.....	100

	2.2.3.2-4	decode_guises	101
	2.2.3.2-5	get_vehicle_model.....	101
	2.2.3.2-6	decode_appearance.....	103
	2.2.3.2-7	decode_team	103
	2.2.3.2-8	decode_capabilities.....	104
	2.2.3.2-9	get_vehicle_country.....	104
	2.2.3.2-10	get_vehicle_description.....	105
	2.2.3.2-11	get_ground_vehicle_description.....	106
	2.2.3.2-12	get_air_vehicle_description	106
	2.2.3.2-13	get_water_vehicle_description.....	107
2.3	MAP HANDLING CSC DESCRIPTION		108
2.3.1	Initial Map Handling CSC Description		109
	2.3.1.1	ps_globals.c CSU Description (/simnet/pvd/lib/paintlib.a)	109
	2.3.1.2	tdb_globals.c CSU Description (/simnet/pvd/lib/paintlib.a)	109
	2.3.1.3	write_string.c CSU Description (/simnet/pvd/lib/paintlib.a)	109
		2.3.1.3-1 write_string	110
	2.3.1.4	libtdb.a.....	110
2.3.2	Continuous Zoom CSC Description		111
	2.3.2.1	stripe.c.....	111
		2.3.2.1-1 display_stripe.....	111
	2.3.2.2	new_zoom.c	112
		2.3.2.2-1 new_zoom.c Information.....	112
		2.3.2.2-2 init_terrain_db	113
		2.3.2.2-3 reinit_db.....	113
		2.3.2.2-4 init_paint_system.....	114
		2.3.2.2-5 poly_zoom	115
		2.3.2.2-6 unified_paint_square_region.....	115
		2.3.2.2-7 pnav_paint_region.....	116
		2.3.2.2-8 pnav_zoom_to_scale.....	116
		2.3.2.2-9 paint_square_region.....	117
		2.3.2.2-10 draw_polygons.....	118
		2.3.2.2-11 draw_non_ground_polygons	118
		2.3.2.2-12 draw_patch.....	119
		2.3.2.2-13 draw_stripe.....	119
		2.3.2.2-14 zoom_to_scale	120
		2.3.2.2-15 pan	120
		2.3.2.2-16 zoom_out.....	121
		2.3.2.2-17 setup_back_fb	121
		2.3.2.2-18 backout_of_change.....	121
		2.3.2.2-19 display_back_fb.....	122

	2.3.2.2-20	copy_map_back	122
	2.3.2.2-21	copy_nonmap_back	122
	2.3.2.2-22	print_bb_desc	122
	2.3.2.2-23	init_pixel_map_descriptors	122
	2.3.2.2-24	force_square	123
	2.3.2.2-25	init_canopy_pattern	123
	2.3.2.2-26	set_no_map_on_paint	123
	2.3.2.2-27	draw_treeline	123
	2.3.2.2-28	highlight_altitude_hazard	124
	2.3.2.2-29	draw_hazard_polygons	124
	2.3.2.2-30	draw_hazards_in_patch	125
	2.3.2.2-31	draw_hazards_in_stripe	125
2.3.3	Finite Zoom CSC Description		126
2.3.3.1	finite_zoom.c		126
	2.3.3.1-1	init_discrete_zoom	127
	2.3.3.1-2	clear_stored_bitmaps	128
	2.3.3.1-3	init_stored_bitmaps	128
	2.3.3.1-4	create_file	128
	2.3.3.1-5	create_bitmap	129
	2.3.3.1-6	file_bitmap	130
	2.3.3.1-7	count_bits_set	130
	2.3.3.1-8	get_shift	130
	2.3.3.1-9	blt_paint_square_region	131
	2.3.3.1-10	display_bitmap	132
	2.3.3.1-11	get_bitmap	132
	2.3.3.1-12	get_lru_bitmap	133
	2.3.3.1-13	readin_bitmap	133
2.3.4	Save/Recall Zoom CSC Description		134
2.3.4.1	save_array.c		134
	2.3.4.1-1	save_array	135
	2.3.4.1-2	recall_array	135
2.3.4.2	save_screen.c		135
	2.3.4.2-1	save_screen	136
2.3.4.3	save_zoom.c		136
	2.3.4.3-1	save_zoom	136
	2.3.4.3-2	recall_zoom	137
2.3.4.4	display_file.c		137
	2.3.4.4-1	display_file	137
2.3.4.5	display_top.c		138
	2.3.4.5-1	display_top_map	138
2.3.5	Contours CSC Description		139
2.3.5.1	gen_contour.c		139
	2.3.5.1-1	clear_contour	140

2.3.5.2	draw_contour.c.....	140
2.3.5.2-1	draw_contour.....	140
2.3.5.2-2	draw_top_contours.....	140
2.3.5.2-3	add_contours.....	141
2.3.5.2-4	top_level.....	141
2.3.5.3	delt_contour.c CSU Description (/simnet/pvd/lib/paintlib.a)	141
2.3.5.3-1	delta_contour.....	141
2.3.5.3-2	draw_contour_stripe.....	142
2.3.5.3-3	contour_patch.....	142
2.3.5.4	init_contour.c CSU Description (/simnet/pvd/lib/paintlib.a)	143
2.3.5.4-1	init_contour.....	143
2.3.5.4-2	new_contour_interval.....	143
2.3.5.4-3	set_contour_interval.....	143
2.3.5.4-4	read_elevation_limits.....	143
2.3.5.4-5	get_contour_interval.....	143
2.3.5.5	low_contour.c.....	144
2.3.5.5-1	contour_poly.....	144
2.3.5.5-2	encode_contour.....	144
2.3.5.5-3	draw_delta_poly.....	145
2.3.6	Grid CSC Description.....	146
2.3.6.1	grid.c CSU Description (/simnet/pvd/lib/paintlib.a)	146
2.3.6.1-1	draw_horiz_grid_lines.....	147
2.3.6.1-2	draw_vert_grid_lines.....	147
2.3.6.1-3	draw_grid_lines.....	148
2.3.6.1-4	calc_first_position.....	148
2.3.6.1-5	change_grid_spacing.....	148
2.3.6.1-6	get_grid_spacing.....	148
2.3.7	Other Features CSC Description.....	149
2.3.7.1	unshade.c.....	149
2.3.7.1-1	shade_desired.....	149
2.3.7.1-2	toggle_shade.....	150
2.3.7.1-3	silent_shade_off.....	150
2.3.7.1-4	silent_shade_on.....	150
2.3.7.2	switch_db.c.....	150
2.3.7.2-1	switch_database.....	150
2.3.7.3	prev_view.c.....	150
2.3.7.3-1	previous_view.....	150
2.3.7.3-2	test_front.....	151
2.3.7.3-3	next_view.....	151
2.3.7.3-4	add_view.....	151
2.3.7.3-5	ring_dec.....	152

	2.3.7.3-6	ring_inc.....	152
	2.3.7.3-7	get_cached_bitmap.....	152
	2.3.7.3-8	create_cached_bitmap.....	152
	2.3.7.3-9	print_buffer_status.....	153
	2.3.7.3-10	reset_views.....	153
	2.3.7.3-11	display_views.....	153
2.3.7.4	objects.c.....		153
	2.3.7.4-1	print_nearest_object.....	153
2.4	PVD-LEVEL PROCESSING.....		154
2.4.1	Initial PVD-Level CSC Description.....		155
2.4.1.1	pvd.c.....		155
	2.4.1.1-1	exit_gracefully.....	155
	2.4.1.1-2	exit_disgracefully.....	156
	2.4.1.1-3	thats_all_folks.....	156
	2.4.1.1-4	process_nonappearance_packets.....	156
2.4.2	Shared Definitions CSC Description.....		157
2.4.2.1	p_assoc.h.....		157
2.4.3	Shared Procedures CSC Description.....		158
2.4.3.1	libassoc.a.....		158
2.4.3.2	libmove.a.....		158
2.4.3.3	libshm.a.....		158
2.4.3.4	libnetif.a.....		158
2.5	UTILITIES.....		159
2.5.1	General Utilities CSC Description.....		160
2.5.1.1	pvd_util.c.....		160
	2.5.1.1-1	force_to_range.....	160
2.5.1.2	pvd_windows.c CSU Description (/simnet/pvd/lib/pvdiface.a).....		161
	2.5.1.2-1	set_menu_window_top.....	161
	2.5.1.2-2	get_menu_window_top.....	161
	2.5.1.2-3	init_windows.....	161
	2.5.1.2-4	init_info_window.....	161
	2.5.1.2-5	init_info_text_window.....	161
	2.5.1.2-6	init_event_window.....	162
	2.5.1.2-7	init_event_text_window.....	162
	2.5.1.2-8	reinit_event_text_window.....	162
	2.5.1.2-9	erase_event_window_border.....	162
	2.5.1.2-10	init_time_window.....	162
	2.5.1.2-11	init_legend_window.....	162
	2.5.1.2-12	display_map_key.....	163
	2.5.1.2-13	display_map_key_text.....	163
	2.5.1.2-14	default_mouse_help.....	163

	2.5.1.2-15	freedraw_mouse_help	163
	2.5.1.2-16	resize_mouse_help	163
	2.5.1.2-17	move_mouse_help	164
	2.5.1.2-18	abort_mouse_help	164
	2.5.1.2-19	catc_place_tgt_mvr_mouse_help	164
	2.5.1.2-20	catc_place_tgt_mouse_help	164
	2.5.1.2-21	catc_move_tgt_pickup_mouse_ help	164
	2.5.1.2-22	catc_move_tgt_place_mouse_ help	165
	2.5.1.2-23	catc_remove_tgt_mouse_help	165
	2.5.1.2-24	display_mouse_key	165
	2.5.1.2-25	set_mouse_text	165
	2.5.1.2-26	Function Summary - display_intervis_key	165
	2.5.1.2-27	display_intervis_key_text	166
	2.5.1.2-28	display_icon_key	166
	2.5.1.2-29	display_icon_key_text	166
	2.5.1.2-30	draw_fixed_faad	166
	2.5.1.2-31	draw_fixed_fwa	166
	2.5.1.2-32	draw_fixed_m2	167
	2.5.1.2-33	draw_fixed_tank	167
	2.5.1.2-34	draw_fixed_static	167
	2.5.1.2-35	draw_fixed_burst	168
	2.5.1.2-36	display_key_text	168
	2.5.1.2-37	init_utm_window	168
	2.5.1.2-38	get_exercise_time	169
	2.5.1.2-39	display_time	169
	2.5.1.2-40	init_zone_window	169
	2.5.1.2-41	draw_window_border	169
	2.5.1.2-42	init_zoom_scale	169
	2.5.1.2-43	calc_zoom_scale	170
	2.5.1.2-44	init_icon_scale	170
	2.5.1.2-45	show_icon_scale	170
	2.5.1.2-46	init_exercise	170
	2.5.1.2-47	change_exercise	170
	2.5.1.2-48	change_intervis_key_text	171
	2.5.1.3	init_bell.c	171
2.5.2		Text Input/Output CSC Description	171
	2.5.2.1	type_to.c	171
		2.5.2.1-1 type_to_device	172
	2.5.2.2	t_windows.c	172
		2.5.2.2-1 set_text_window	172
	2.5.2.3	interact.c	172

	2.5.2.3-1	get_typed_input.....	172
	2.5.2.3-2	get_typed_input_float.....	172
	2.5.2.3-3	get_typed_input_str.....	173
	2.5.2.3-4	input_name.....	173
2.5.3		Input Processing CSC Description.....	174
	2.5.3.1	u_flags.c.....	174
	2.5.3.1-1	process_unix_flags.....	174
	2.5.3.2	move.c CSU Description (/simnet/pvd/lib/pvdfiface.a).....	175
	2.5.3.2-1	move_window.....	175
	2.5.3.2-2	clear_window.....	175
	2.5.3.2-3	select_region.....	176
	2.5.3.2-4	constrain_to_screen.....	176
	2.5.3.3	pvd_iface.c.....	176
	2.5.3.3-1	init_user_interface.....	177
	2.5.3.3-2	init_fonts.....	177
	2.5.3.3-3	close_user_interface.....	177
	2.5.3.3-4	init_cursor.....	177
	2.5.3.3-5	init_working_cursor.....	177
	2.5.3.3-6	init_cross_cursor.....	177
	2.5.3.3-7	button.....	178
	2.5.3.3-8	handle_mouse.....	178
	2.5.3.3-9	set_mark.....	178
	2.5.3.3-10	set_first_point.....	178
	2.5.3.3-11	set_point.....	179
	2.5.3.3-12	toggle_selected.....	179
	2.5.3.3-13	get_coord.....	179
	2.5.3.3-14	xy_get_coord.....	180
	2.5.3.3-15	user_to_pixel.....	180
	2.5.3.3-16	pixel_to_user.....	180
	2.5.3.3-17	add_selected_point.....	180
	2.5.3.3-18	current_loc.....	181
	2.5.3.3-19	xy_current_loc.....	181
	2.5.3.4	libmove.a.....	181
	2.5.3.5	init_flags.c.....	181
	2.5.3.6	select.c CSU Description (/simnet/pvd/lib/pvdfiface.a).....	181
	2.5.3.6-1	tank_selected.....	182
	2.5.3.6-2	toggle_nearest_tank.....	182
	2.5.3.6-3	remove_selected_tank.....	182
	2.5.3.6-4	add_selected_tank.....	183
	2.5.3.6-5	my_min.....	183
	2.5.3.6-6	my_max.....	183

	2.5.3.6-7	vehicle_in_region.....	183
	2.5.3.6-8	clear_selected_vehicles	184
	2.5.3.6-9	select_vehicle_by_name.....	184
	2.5.3.6-10	is_vehicle_id_string.....	184
	2.5.3.6-11	vehicle_id_cmp	185
2.5.4	Customization CSC Description.....		185
2.5.4.1	init_env.c.....		186
	2.5.4.1-1	init_environment.....	186
	2.5.4.1-2	get_data_directory.....	186
	2.5.4.1-3	get_vr_host.....	186
	2.5.4.1-4	get_pt_host.....	186
	2.5.4.1-5	get_rc_host.....	187
	2.5.4.1-6	get_dp_address	187
	2.5.4.1-7	get_db_directory.....	187
	2.5.4.1-8	make_path_name	187
	2.5.4.1-9	get_discrete_zoom_levels	187
	2.5.4.1-10	get_patches_to_cache.....	187
	2.5.4.1-11	get_stealth_address.....	188
	2.5.4.1-12	get_db_names.....	188
2.5.4.2	text.c		188
	2.5.4.2-1	text_display	188
	2.5.4.2-2	clear_screen	188
	2.5.4.2-3	home	188
	2.5.4.2-4	move_cursor_to	189
	2.5.4.2-5	prepare_line	189
	2.5.4.2-6	print_header.....	189
	2.5.4.2-7	print_vehicle	189
	2.5.4.2-8	kph.....	190
	2.5.4.2-9	display_int_at.....	190
	2.5.4.2-10	delete_text_tank.....	190
	2.5.4.2-11	display_page_num	191
	2.5.4.2-12	get_screen_size	191
	2.5.4.2-13	get_velocity	191
	2.5.4.2-14	get_turret_degrees.....	192
	2.5.4.2-15	get_gun_degrees.....	192
2.5.5	Map/Icon CSC Description		193
2.5.5.1	find_near.c.....		193
	2.5.5.1-1	find_nearest_route_pt.....	193
2.5.5.2	measures.c		194
	2.5.5.2-1	heading	194
	2.5.5.2-2	heading_fraction.....	194
	2.5.5.2-3	rel_turret.....	194
	2.5.5.2-4	rel_gun.....	195

	2.5.5.2-5	angle_normalize	195
	2.5.5.2-6	dist2	195
	2.5.5.2-7	dist_3	195
	2.5.5.2-8	cmc_msec	196
2.5.6	Debug CSC Description		196
2.5.6.1	memory.c		197
	2.5.6.1-1	get_all_memory	197
	2.5.6.1-2	get_biggest_memory_chunk	197
2.5.6.2	tell.c		197
	2.5.6.2-1	tell_all	197
	2.5.6.2-2	tell_real	198
	2.5.6.2-3	tell_integer	198
2.6	NETWORK PROCESSING		199
2.6.1	Initial Network Process CSC Description		200
2.6.1.1	virtual_net.c		200
	2.6.1.1-1	init_virtual_net	200
	2.6.1.1-2	get_next_packet	201
2.6.2	Send Packets CSC Description		202
2.6.2.1	intern_send.c		202
	2.6.2.1-1	fake_deactivate	202
	2.6.2.1-2	init_internal_ring_buffers	203
	2.6.2.1-3	send_buffer_to_self	203
	2.6.2.1-4	fake_filter	203
	2.6.2.1-5	enqueue_buffer_into_fake_ring	203
	2.6.2.1-6	process_internal_ring_buffer	204
2.6.3	Filter Packets CSC Description		204
2.6.3.1	collision.c		204
	2.6.3.1-1	do_collision	204
2.6.4	Store CSC Description		205
2.6.4.1	hash_fn.c		205
	2.6.4.1-1	Hash_Vehicle_ID	206
	2.6.4.1-2	matching_vehicle_IDs	206
	2.6.4.1-3	equal_vehicle_IDs	206
	2.6.4.1-4	get_vehicle_index	206
2.6.4.2	libtable.a		207
2.6.5	Other Packets CSC Description		207
2.6.5.1	pvd_misc.c		208
	2.6.5.1-1	Collect_World_State	208
	2.6.5.1-2	UpdateText_Display	209
	2.6.5.1-3	out_of_range	209
	2.6.5.1-4	init_ascii	209
	2.6.5.1-5	make_new_vehicle_index	209
	2.6.5.1-6	refresh_timestamps	210

	2.6.5.1-7	clear_net_input.....	210
	2.6.5.1-8	spin.....	210
	2.6.5.1-9	get_spin_count.....	210
	2.6.5.1-10	init_text_exercises.....	210
	2.6.5.1-11	set_exercise	211
	2.6.5.1-12	current_exercise	211
	2.6.5.1-13	get_exercise	211
	2.6.5.1-14	get_vehicle_exercise	211
	2.6.5.1-15	get_dynamic_appearance_data.....	212
	2.6.5.1-16	print_vehicle_id.....	212
	2.6.5.1-17	fprint_vehicle_id.....	212
	2.6.5.1-18	print_simulation_address	213
	2.6.5.1-19	fprint_simulation_address.....	213
	2.6.5.1-20	display_event_flag	213
	2.6.5.1-21	modify_last_status_packet	213
2.6.6	Purge CSC Description.....		214
	2.6.6.1	purge.c.....	214
		2.6.6.1-1	remap_selected_tanks.....
		2.6.6.1-2	remap_tagged_tanks
		2.6.6.1-3	purge_expired_vehicles.....
		2.6.6.1-4	set_time_out_msec
		2.6.6.1-5	purge_vehicle
2.6.7	Protocol Definitions CSC Description.....		216
	2.6.7.1.	network.h	216
	2.6.7.2	libp2p.a.....	216
2.7	GRAPHICS CSC DESCRIPTION		217
	2.7.1	Initial Graphics	218
		2.7.1.1	init_cmap.c.....
			2.7.1.1-1
			flag_map_colors.....
			2.7.1.1-2
			init_terrain_cmap
			2.7.1.1-3
			mapcolor.....
		2.7.1.2	init_graph.c
			2.7.1.2-1
			init_graph.....
2.7.2	Graphics Display Support.....		220
		2.7.2.1	color_map.c.....
			2.7.2.1-1
			make_color_map
			2.7.2.1-2
			make_window_colors
			2.7.2.1-3
			make_icon_colors
			2.7.2.1-4
			make_terrain_colors.....
			2.7.2.1-5
			make_contour_colors
		2.7.2.2	highlight.c CSU Description
			(/simnet/pvd/lib/pvdiface.a)
			2.7.2.2-1
			highlight_tank.....

	2.7.2.2-2	unhighlight_tank.....	223
2.7.2.3	rgb.c		223
	2.7.2.3-1	rgb.....	223
2.8	OVERLAYS CSC DESCRIPTION.....		224
2.8.1	Initial Overlays CSC Description.....		225
2.8.1.1	overlayif.c.....		225
	2.8.1.1-1	overlay_mouse.....	225
	2.8.1.1-2	zoom_overlay.....	226
	2.8.1.1-3	erase_overlay	226
	2.8.1.1-4	read_over.....	226
	2.8.1.1-5	get_overlay.....	226
	2.8.1.1-6	save_over.....	227
	2.8.1.1-7	delete_over_file.....	227
	2.8.1.1-8	edit_over.....	227
	2.8.1.1-9	reset_overlay.....	227
	2.8.1.1-10	select_overobj.....	228
	2.8.1.1-11	delete_overobj	228
	2.8.1.1-12	reduce_overobj	228
	2.8.1.1-13	move_overobj.....	228
	2.8.1.1-14	addtext_overobj.....	229
	2.8.1.1-15	init_erasing_cursor.....	229
	2.8.1.1-16	init_drawing_cursor.....	229
	2.8.1.1-17	init_size_cursor.....	229
	2.8.1.1-18	gen_add_text.....	229
	2.8.1.1-19	get_string.....	230
2.8.1.2	new_overlay.c CSU Description (/simnet/pvd/lib/liboverlay.a)		230
	2.8.1.2-1	overlay_init.....	230
	2.8.1.2-2	display_overlays.....	231
	2.8.1.2-3	clear_overlays.....	231
	2.8.1.2-4	calc_zoom_ratio	231
	2.8.1.2-5	over_user_to_pixel.....	231
	2.8.1.2-6	over_point_in_box	232
2.8.2	Line Draw CSC Description		233
2.8.2.1	overlineif.c		233
	2.8.2.1-1	overline_create.....	234
	2.8.2.1-2	overline_addtext.....	234
	2.8.2.1-3	size_loop	234
	2.8.2.1-4	overline_delete.....	235
	2.8.2.1-5	overline_newfeatures	235
	2.8.2.1-6	show_overline	235
	2.8.2.1-7	overline_newshape.....	236
	2.8.2.1-8	select_overline	236

2.8.2.1-9	overline_loop	237
2.8.2.1-10	overline_pencil_on	237
2.8.2.1-11	overline_erase	237
2.8.2.1-12	draw_line_window	238
2.8.2.1-13	del_line_window	238
2.8.2.1-14	line_reg_def	238
2.8.2.1-15	color_line	238
2.8.2.1-16	uncolor_line	239
2.8.2.1-17	draw_name	239
2.8.2.1-18	draw_time	239
2.8.2.1-19	draw_line	239
2.8.2.1-20	draw_lsize	240
2.8.2.1-21	room_for_lsize	240
2.8.2.1-22	draw_ltype	240
2.8.2.1-23	draw_blabel2	241
2.8.2.1-24	draw_blabel1	241
2.8.2.1-25	draw_endlabels	241
2.8.2.2	ovline_func.c	242
2.8.2.2-1	name_label	242
2.8.2.2-2	name_action	242
2.8.2.2-3	time_label	242
2.8.2.2-4	time_action	243
2.8.2.2-5	unit1_label	243
2.8.2.2-6	unit1_action	243
2.8.2.2-7	unit2_label	244
2.8.2.2-8	unit2_action	244
2.8.2.2-9	solid_label	244
2.8.2.2-10	solid_action	244
2.8.2.2-11	dashed_label	245
2.8.2.2-12	dashed_action	245
2.8.2.2-13	thin_label	245
2.8.2.2-14	thin_action	246
2.8.2.2-15	thick_label	246
2.8.2.2-16	thick_action	246
2.8.2.2-17	lgeneric_size	247
2.8.2.2-18	lsquad_action	247
2.8.2.2-19	lsection_action	247
2.8.2.2-20	lplatoon_action	248
2.8.2.2-21	ltroop_action	248
2.8.2.2-22	lbattalion_action	248
2.8.2.2-23	lgroup_action	248
2.8.2.2-24	lgeneric_type	249
2.8.2.2-25	boundary_label	249

	2.8.2.2-26	boundary_action.....	249
	2.8.2.2-27	ldeparture_label.....	250
	2.8.2.2-28	ldeparture_action	250
	2.8.2.2-29	phase_label.....	250
	2.8.2.2-30	phase_action	250
	2.8.2.2-31	pdeployment_label	251
	2.8.2.2-32	pdeployment_action.....	251
	2.8.2.2-33	nofire_label.....	251
	2.8.2.2-34	nofire_action	252
	2.8.2.2-35	delay_label	252
	2.8.2.2-36	delay_action	252
	2.8.2.2-37	free_label	252
	2.8.2.2-38	free_action	253
	2.8.2.2-39	ldone_create_action	253
2.8.2.3	overdraw.c CSU Description		
	(/simnet/pvd/lib/liboverlay.a)		253
	2.8.2.3-1	draw_overline.....	253
	2.8.2.3-2	add_segment	254
	2.8.2.3-3	erase_segment	254
	2.8.2.3-4	add_size	255
	2.8.2.3-5	overline_features	255
	2.8.2.3-6	add_name.....	256
	2.8.2.3-7	add_time.....	256
	2.8.2.3-8	add_type.....	256
	2.8.2.3-9	add_free_text.....	257
	2.8.2.3-10	add_endlabels.....	257
	2.8.2.3-11	add_midlabel.....	257
	2.8.2.3-13	find_offset.....	258
	2.8.2.3-14	get_need	259
	2.8.2.3-15	get_angle	259
2.8.2.4	overline.c CSU Description		
	(/simnet/pvd/lib/liboverlay.a)		259
	2.8.2.4-1	overline_init.....	259
	2.8.2.4-2	overline_reset	259
	2.8.2.4-3	get_free_overline	260
	2.8.2.4-4	copy_overline.....	260
	2.8.2.4-5	zoom_overline	260
	2.8.2.4-6	erase_overline.....	260
	2.8.2.4-7	insert_overline	261
	2.8.2.4-8	mouse_to_overline	261
2.8.3	Unit Symbols CSC Description		262
2.8.3.1	symbolif.c.....		262
	2.8.3.1-1	symbol_create.....	263

	2.8.3.1-2	draw_create_window	263
	2.8.3.1-3	sym_reg_def	264
	2.8.3.1-4	del_create_window	264
	2.8.3.1-5	select_symbol	264
	2.8.3.1-6	symbol_edit	265
	2.8.3.1-7	symbol_move	265
	2.8.3.1-8	symbol_delete	266
	2.8.3.1-9	show_symbol	266
	2.8.3.1-10	symbol_addtext	266
	2.8.3.1-12	uncolor_selected	267
	2.8.3.1-13	symbol_reduce	267
	2.8.3.1-14	resize_symbol	267
2.8.3.2	symbol_func.c		268
	2.8.3.2-1	desig_label	268
	2.8.3.2-2	desig_action	268
	2.8.3.2-3	hi_eche_label	268
	2.8.3.2-4	hi_eche_action	269
	2.8.3.2-5	friend_label	269
	2.8.3.2-6	friend_action	269
	2.8.3.2-7	foe_label	270
	2.8.3.2-8	foe_action	270
	2.8.3.2-9	actual_label	270
	2.8.3.2-10	actual_action	271
	2.8.3.2-11	proposed_label	271
	2.8.3.2-12	proposed_action	271
	2.8.3.2-13	generic_box(.....	272
	2.8.3.2-14	unit_label	272
	2.8.3.2-15	unit_action	272
	2.8.3.2-16	headq_label	273
	2.8.3.2-17	headq_action	273
	2.8.3.2-18	css_label	273
	2.8.3.2-19	css_action	274
	2.8.3.2-20	generic_size	274
	2.8.3.2-21	squad_label	274
	2.8.3.2-22	squad_action	274
	2.8.3.2-23	section_label	275
	2.8.3.2-24	section_action	275
	2.8.3.2-25	platoon_label	275
	2.8.3.2-26	platoon_action	276
	2.8.3.2-27	troop_label	276
	2.8.3.2-28	troop_action	276
	2.8.3.2-29	battalion_label	277
	2.8.3.2-30	battalion_action	277

2.8.3.2-31	group_label.....	277
2.8.3.2-32	group_action	278
2.8.3.2-33	generic_branch.....	278
2.8.3.2-34	air_def_art_label.....	278
2.8.3.2-35	air_def_art_action	279
2.8.3.2-36	armor_label.....	279
2.8.3.2-37	armor_action	279
2.8.3.2-38	nbc_label	280
2.8.3.2-39	nbc_action.....	280
2.8.3.2-40	cavalry_label	280
2.8.3.2-42	field_art_label.....	281
2.8.3.2-43	field_art_action	281
2.8.3.2-44	infantry_label	282
2.8.3.2-45	infantry_action.....	282
2.8.3.2-46	medical_label.....	282
2.8.3.2-47	medical_action	283
2.8.3.2-48	airborne_label.....	283
2.8.3.2-49	airborne_action	283
2.8.3.2-50	antitank_label	284
2.8.3.2-51	antitank_action.....	284
2.8.3.2-52	army_avi_label.....	284
2.8.3.2-53	army_avi_action	285
2.8.3.2-54	repair_label.....	285
2.8.3.2-55	repair_action	285
2.8.3.2-56	generic_supply.....	285
2.8.3.2-57	ammo_all_label	286
2.8.3.2-58	ammo_all_action.....	286
2.8.3.2-59	ammo_art_label.....	287
2.8.3.2-60	ammo_art_action	287
2.8.3.2-61	ammo_rock_label.....	287
2.8.3.2-62	ammo_rock_action	288
2.8.3.2-63	ammo_small_label.....	288
2.8.3.2-64	ammo_small_action	288
2.8.3.2-65	ammo_specl_label.....	289
2.8.3.2-66	ammo_specl_action	289
2.8.3.2-67	ammo_conv_label	289
2.8.3.2-68	ammo_conv_action.....	290
2.8.3.2-69	generic_weapon	290
2.8.3.2-70	weap_auto_label.....	290
2.8.3.2-71	weap_auto_action	291
2.8.3.2-72	weap_mortar_label	291
2.8.3.2-73	weapon_mortar_action.....	291
2.8.3.2-74	weap_airdef_label.....	292

2.8.3.2-75	weap_airdef_action.....	292
2.8.3.2-76	weap_antit_label.....	292
2.8.3.2-77	weap_anit_action.....	293
2.8.3.2-78	weap_gun_label.....	293
2.8.3.2-79	weap_gun_action.....	293
2.8.3.2-80	weap_howit_label.....	294
2.8.3.2-81	weap_howit_action.....	294
2.8.3.2-82	weap_antig_label.....	294
2.8.3.2-83	weap_antig_action.....	295
2.8.3.2-84	weap_recoil_label.....	295
2.8.3.2-85	weap_recoil_action.....	295
2.8.3.2-86	weap_airgun_label.....	296
2.8.3.2-87	weap_airgun_action.....	296
2.8.3.2-88	weap_miss_label.....	296
2.8.3.2-89	weap_miss_action.....	297
2.8.3.2-90	weap_airmiss_label.....	297
2.8.3.2-91	weap_antmiss_label.....	297
2.8.3.2-92	weap_antmiss_label.....	298
2.8.3.2-93	weap_antmiss_action.....	298
2.8.3.2-94	armor_tank_label.....	298
2.8.3.2-95	armor_tank_action.....	299
2.8.3.2-96	generic_weapsize.....	299
2.8.3.2-97	weap_light_action.....	299
2.8.3.2-98	weap_light_action.....	300
2.8.3.2-99	weap_med_label.....	300
2.8.3.2-100	weap_med_action.....	300
2.8.3.2-101	weap_heavy_label.....	301
2.8.3.2-102	weap_heavy_action.....	301
2.8.3.2-103	done_create_label.....	301
2.8.3.2-104	done_create_action.....	302
2.8.3.2-105	next_create_label.....	302
2.8.3.2-106	next_create_action.....	302
2.8.3.2-107	prev_create_label.....	303
2.8.3.2-108	prev_create_action.....	303
2.8.3.3	symbol.c CSU Description	
	(/simnet/pvd/lib/liboverlay.a).....	303
2.8.3.3-1	symbol_init.....	303
2.8.3.3-2	symbol_reset.....	304
2.8.3.3-3	get_free_symbol.....	304
2.8.3.3-4	insert_symbol.....	304
2.8.3.3-5	draw_symbol.....	304
2.8.3.3-6	calc_basic_dim.....	305
2.8.3.3-7	mouse_to_symbol.....	305

	2.8.3.3-8	erase_symbol	306
	2.8.3.3-9	zoom_symbol.....	306
	2.8.3.3-10	copy_symbol.....	307
2.8.3.4	symdraw.c CSU Description (/simnet/pvd/lib/liboverlay.a)		307
	2.8.3.4-1	draw_box.....	307
	2.8.3.4-2	draw_size.....	307
	2.8.3.4-3	draw_branch	308
	2.8.3.4-4	draw_hi_eche	308
	2.8.3.4-5	draw_designation.....	308
	2.8.3.4-6	draw_symtext.....	309
	2.8.3.4-7	draw_proposed	309
	2.8.3.4-8	draw_supply	309
	2.8.3.4-9	draw_weapon.....	310
	2.8.3.4-10	draw_weapsize	310
	2.8.3.4-11	draw_unit.....	310
	2.8.3.4-12	add_headq.....	311
	2.8.3.4-13	draw_css	311
	2.8.3.4-14	draw_squad	311
	2.8.3.4-15	draw_section.....	311
	2.8.3.4-16	draw_platoon	312
	2.8.3.4-17	draw_troop.....	312
	2.8.3.4-18	draw_battalion	312
	2.8.3.4-19	draw_group	313
	2.8.3.4-20	draw_air_def.....	313
	2.8.3.4-21	draw_armor	313
	2.8.3.4-22	draw_nbc.....	313
	2.8.3.4-23	draw_calvalry.....	314
	2.8.3.4-24	draw_field.....	314
	2.8.3.4-25	draw_infantry	314
	2.8.3.4-26	draw_medical	314
	2.8.3.4-27	draw_airborne	315
	2.8.3.4-28	draw_antitank.....	315
	2.8.3.4-29	draw_aviation.....	315
	2.8.3.4-30	draw_repair	315
	2.8.3.4-31	draw_ammo.....	316
	2.8.3.4-32	add_ammo_art.....	316
	2.8.3.4-33	add_ammo_rock.....	316
	2.8.3.4-34	add_ammo_small.....	316
	2.8.3.4-35	add_ammo_spec1	316
	2.8.3.4-36	add_ammo_conv	317
	2.8.3.4-37	draw_weap_auto	317
	2.8.3.4-38	add_high_trayec	317

	2.8.3.4-39	add_airdef	317
	2.8.3.4-40	add_flat_trayec	318
	2.8.3.4-41	add_launcher	318
	2.8.3.4-42	draw_weap_gun	318
	2.8.3.4-43	draw_weap_miss	318
	2.8.3.4-44	draw_armor_tank	319
	2.8.3.4-45	draw_weap_med	319
	2.8.3.4-46	draw_weap_heavy	319
2.8.4	Control Points CSC Description		320
2.8.4.1	controlptif.c		320
	2.8.4.1-1	select_controlpt	321
	2.8.4.1-2	color_controlpt	321
	2.8.4.1-3	uncolor_controlpt	321
	2.8.4.1-4	controlpt_check	322
	2.8.4.1-5	controlpt_coord	322
	2.8.4.1-6	controlpt_con	322
	2.8.4.1-7	controlpt_link	322
	2.8.4.1-8	controlpt_rel	322
	2.8.4.1-9	controlpt_pass	323
	2.8.4.1-10	move_controlpt	323
	2.8.4.1-11	place_controlpt	323
	2.8.4.1-12	controlpt_delete	323
	2.8.4.1-13	controlpt_reduce	324
	2.8.4.1-14	resize_controlpt	324
	2.8.4.1-15	controlpt_addtext	324
2.8.4.2	controlpt.c CSU Description (/simnet/pvd/lib/liboverlay.a)		324
	2.8.4.2-1	controlpt_init	324
	2.8.4.2-2	controlpt_reset	324
	2.8.4.2-3	mouse_to_controlpt	325
	2.8.4.2-4	get_free_controlpt	325
	2.8.4.2-5	insert_controlpt	325
	2.8.4.2-6	zoom_controlpt	326
	2.8.4.2-7	erase_controlpt	326
2.8.4.3	controlptdrw.c CSU Description (/simnet/pvd/lib/liboverlay.a)		326
	2.8.4.3-1	draw_controlpt	326
	2.8.4.3-2	draw_pttext	327
	2.8.4.3-3	draw_ptshape	327
	2.8.4.3-4	draw_circlept	328
	2.8.4.3-5	draw_squarept	328
	2.8.4.3-6	draw_solidpt	328

2.8.5	Manage Overlays CSC Description	329
2.8.5.1	over_files.c CSU Description (/simnet/pvd/lib/liboverlay.a)	329
2.8.5.1-1	init_over	330
2.8.5.1-2	load_overlay	330
2.8.5.1-3	save_overline	330
2.8.5.1-4	save_controlpt	330
2.8.5.1-5	save_symbol	330
2.8.5.1-6	read_overlines	331
2.8.5.1-7	read_controlpts	331
2.8.5.1-8	read_symbols	331
2.8.5.1-9	list_over_files	331
2.9	POPUP WINDOWS CSC DESCRIPTION	332
2.9.1	Initial Popup Windows CSC Description	333
2.9.1.1	pop_windows.c	333
2.9.1.1-1	pop_windows_present	333
2.9.1.1-2	init_popup	334
2.9.1.1-3	setup_new_window	334
2.9.1.1-4	delete_window	334
2.9.1.1-5	translate_window	335
2.9.1.1-6	translate_region	335
2.9.1.1-7	create_window	335
2.9.1.1-8	erase_user_box	336
2.9.1.1-9	erase_window	336
2.9.1.1-10	big_font_data	337
2.9.1.1-11	draw_window	337
2.9.1.1-12	copy_array	337
2.9.1.1-13	get_window_num	338
2.9.1.1-14	handle_popup_windows	338
2.9.1.1-15	point_in_box	338
2.9.1.1-16	move_pop_window	339
2.9.1.1-17	check_placability	339
2.9.1.1-18	offset_box	340
2.9.1.1-19	box_overlaps_window	340
2.9.1.1-20	move_via_back_fb	340
2.9.1.1-21	copy_planes_back	341
2.9.1.1-22	place_new_window	341
2.9.1.1-23	retile_screen	342
2.9.1.1-24	dump windows	342
2.9.1.2	init_window.c	342
2.9.1.2-1	init_window_colors	342

2.9.2	Manage Popup Windows CSC Description.....	343
2.9.2.1	plot_box.c.....	343
2.9.2.1-1	plot_box.....	344
2.9.2.2	help.c.....	344
2.9.2.2-1	help.....	344
2.9.2.2-2	kill_help_window.....	345
2.9.2.2-3	display_window_text.....	345
2.9.2.2-4	display_window_string.....	345
2.9.2.2-5	strip_char.....	346
2.9.2.2-6	find_max_chars.....	346
2.9.2.3	error_box.c.....	346
2.9.2.3-1	error_box.....	346
2.9.2.3-2	kill_error_window.....	347
2.9.2.3-3	error_window_exists.....	347
2.9.2.3-4	check_error_window.....	347
2.9.2.3-5	string_error_box.....	347
2.9.2.4	option.c.....	348
2.9.2.4-1	option_reg_define.....	348
2.9.2.4-2	define_block.....	348
2.9.2.4-3	display_block.....	349
2.9.2.4-4	erase_block.....	349
2.9.2.4-5	display_title.....	350
2.9.2.4-6	mouse_to_option.....	350
2.9.2.4-7	draw_text_label.....	351
2.9.2.4-8	display_from_blk.....	351
2.9.2.4-9	erase_region.....	352
2.9.2.4-10	reset_region.....	352
2.9.2.4-11	draw_select_box.....	352
2.9.2.4-12	mark_box.....	353
2.10	TOOLS CSC DESCRIPTION.....	354
2.10.1	Ruler CSC Description.....	355
2.10.1.1	ruler.c.....	355
2.10.1.1-1	text_ruler.....	355
2.10.2	Intervisibility CSC Description.....	356
2.10.2.1	libcheckvis.a CSU Description (/simnet/lib).....	356
2.10.2.2	vehicle_to_v.c.....	356
2.10.2.2-1	do_vehicle_to_vehicle.....	357
2.10.2.2-2	update_vehicle_to_vehicle.....	357
2.10.2.2-3	erase_rays.....	357
2.10.2.2-4	get_tags.....	358
2.10.2.2-5	set_tags.....	358
2.10.2.2-6	clear_all_veh_to_veh.....	358

	2.10.2.2-7	encode_force	358
	2.10.2.2-8	view_all_align	358
	2.10.2.2-9	view_opposing_align	358
2.10.2.3	get_elev.c		359
	2.10.2.3-1	get_elevation	359
2.10.2.4	get_view.c		359
	2.10.2.4-1	get_view_height	359
	2.10.2.4-2	get_target_height	359
	2.10.2.4-3	set_view_height	359
	2.10.2.4-4	set_target_height	360
	2.10.2.4-5	set_view_range	360
	2.10.2.4-6	get_view_range	360
2.10.2.5	intervis.c		360
	2.10.2.5-1	intervisibility	360
	2.10.2.5-2	init_intervisibility	361
	2.10.2.5-3	new_intervis	361
	2.10.2.5-4	draw_coded_line	361
	2.10.2.5-5	set_ray_color	362
	2.10.2.5-6	intervis_360	362
	2.10.2.5-7	real_intervis_360	363
	2.10.2.5-8	intervis_vehicles	363
	2.10.2.5-9	emb_attenuate	364
	2.10.2.5-10	draw_x	364
	2.10.2.5-11	attenuation_factor	365
	2.10.2.5-12	print_visibility_code	365
2.10.2.6	clipper.c		365
	2.10.2.6-1	clip_to_map	365
	2.10.2.6-2	clip_to_rect	366
2.10.3	CrossSection CSC Description		367
2.10.3.1	cross.c		367
	2.10.3.1-1	cross_section	368
	2.10.3.1-2	colin_cross	369
	2.10.3.1-3	kill_cross_window	369
	2.10.3.1-4	compute_elevation_segs	370
2.11	REMOTE DEVICES INTERFACES		371
2.11.1	Stealth Interface CSC Description		372
2.11.1.1	place.c		373
	2.11.1.1-1	send_event_flag	373
	2.11.1.1-2	teleport_stealth	373
	2.11.1.1-3	attach_stealth	374
	2.11.1.1-4	mimic_vehicle	374
	2.11.1.1-5	set_stealth_exercise	374
	2.11.1.1-6	handle_stealth_error	375

	2.11.1.1-7	handle_attached_packet	375
	2.11.1.1-8	get_location	375
	2.11.1.1-9	construct_sim_from_stealth	375
2.11.1.2	p_stlth.h		376
2.11.2	Logger Interface and Control CSC Description		376
2.11.2.1	logger_comm.c		376
	2.11.1.2-1	init_connections	377
	2.11.1.2-2	get_loggerAvailReplies.....	377
	2.11.1.2-3	find_loggers.....	377
	2.11.1.2-4	get_loggerConnectAck.....	378
	2.11.1.2-5	connect_to_logger.....	378
	2.11.1.2-6	disconnect_dl	379
	2.11.1.2-7	play_logger.....	379
	2.11.1.2-8	record_logger.....	379
	2.11.1.2-9	continue_logger.....	379
	2.11.1.2-10	logger_freeze.....	380
	2.11.1.2-11	logger_stop.....	380
	2.11.1.2-12	logger_speed.....	380
	2.11.1.2-13	logger_seek	380
	2.11.1.2-14	get_logger_address.....	381
	2.11.1.2-15	get_logger_name	381
	2.11.1.2-16	get_logger_name_from_address.....	381
	2.11.1.2-17	get_logger_broadcast_address.....	381
2.11.2.2	p_logger.h.....		381
2.11.2.3	liblogpvd.a CSU Description (/simnet/pvd/libsrc).....		382
2.11.2.4	libptcomm.a CSU Description (/simnet/pvd/libsrc/libcomm).....		382
	2.11.2.4-1	libptcomm.a Information	382
	2.11.2.4-2	pvd_logger_exit() Information.....	382
2.11.3	Initial Remote Interfaces CSC Description.....		383
2.11.3.1	pvdrtc.c.....		383
	2.11.3.1-1	handle_log_rtc	383
	2.11.3.1-2	send_rtc.....	383
	2.11.3.1-3	init_pdu.....	384
	2.11.3.1-4	avail_reply_handler	384
	2.11.3.1-5	connect_ack_nak_handler.....	384
	2.11.3.1-6	information_handler.....	384
	2.11.3.1-7	clock_tick_handler	385
	2.11.3.1-8	status_reply_handler	385
	2.11.3.1-9	do_avail_request.....	385
	2.11.3.1-10	do_connect_request.....	385
	2.11.3.1-11	do_disconnect.....	386

2.11.3.1-12	do_start_recording_command	386
2.11.3.1-13	do_start_playback_command	387
2.11.3.1-14	do_continue_command	387
2.11.3.1-15	do_suspend_command	387
2.11.3.1-16	do_stop_command	388
2.11.3.1-17	do_seek_relative_command	388
2.11.3.1-18	do_play_speed_command	388
2.11.3.1-19	do_status_request	388
APPENDIX A: TYPE NAMES AND WHERE THEY ARE DEFINED		A-1
APPENDIX B: MACRO FUNCTIONS AND WHERE THEY ARE DEFINED		B-1
APPENDIX C: FUNCTIONS AND CALLING FUNCTIONS BY DIRECTORY AND FILE		C-1
INDEX BY SECTION NUMBER		INDEX-1

1 INTRODUCTION: PLAN VIEW DISPLAY (PVD) CSCI DESCRIPTION

1.1 BACKGROUND

The Plan View Display (PVD) provides the real-time viewing of an exercise or the playback and review of an exercise at a later time after recording by the Data Logger. The PVD is used by a variety of SIMNET personnel, and provides a view of the battlefield as seen from directly above--sometimes referred to as a "bird's eye view." The PVD is also used to control the stealth vehicle.

The PVD contains a color-coded display showing various geographic details, such as roads, rivers, and terrain relief. Individual vehicles are represented on the PVD by icons that indicate the position and orientation of the hull and the relative orientation of the turret. The user can vary the size of the icons by selecting a magnification level. Shading of the geographic display is performed according to an algorithm which assumes a sun elevation of 45 degrees from the horizon and a sun shining from the top right-hand corner of the display.

A shot is indicated by a line segment emanating from the firing vehicle. The user can select particular vehicles and obtain detailed information about them, such as vehicle location, speed, alignment, identification number, and repair and supply status.

The user can zoom in or out on any selected area of the battlefield, pan the display in any direction, obtain intervisibility readings, and optionally overlay grid and/or contour lines at selected intervals. Any terrain cross-section is viewable by selecting its two endpoints. Numeric values for the terrain elevation and corresponding map coordinates (UTM or XY) for any point in the field of view are similarly obtainable.

Intervisibility is based on the idea of running a ray from an eye to a target and checking whether the ray encounters any obstacles. Levels of visibility are: target fully visible; target partially visible; and, target invisible. These levels are color-coded for display.

1.2 EXTERNAL INTERFACES

PVD external interfaces are illustrated in Figure 1.2-1, "PVD External Interfaces," and summarized in Table 1.2-1, "Summary of PVD External Interfaces." PVD receives vehicle data from vehicle appearance network packets, exercise playback data from the Data Logger, and Stealth data from the Stealth vehicle. Each of the external interfaces above uses basic network services. PVD interfaces with SIM hosts using SIMNET simulation and data collection protocols; with the Stealth, using Stealth protocol; and, with the Data Logger, using Logger protocol. PVD protocols include message structures called 'Protocol Data Units' (PDUs). PDUs used by PVD interfaces are summarized in Table 1.2-2, "Summary of PVD Protocol Data Units (PDUs)."

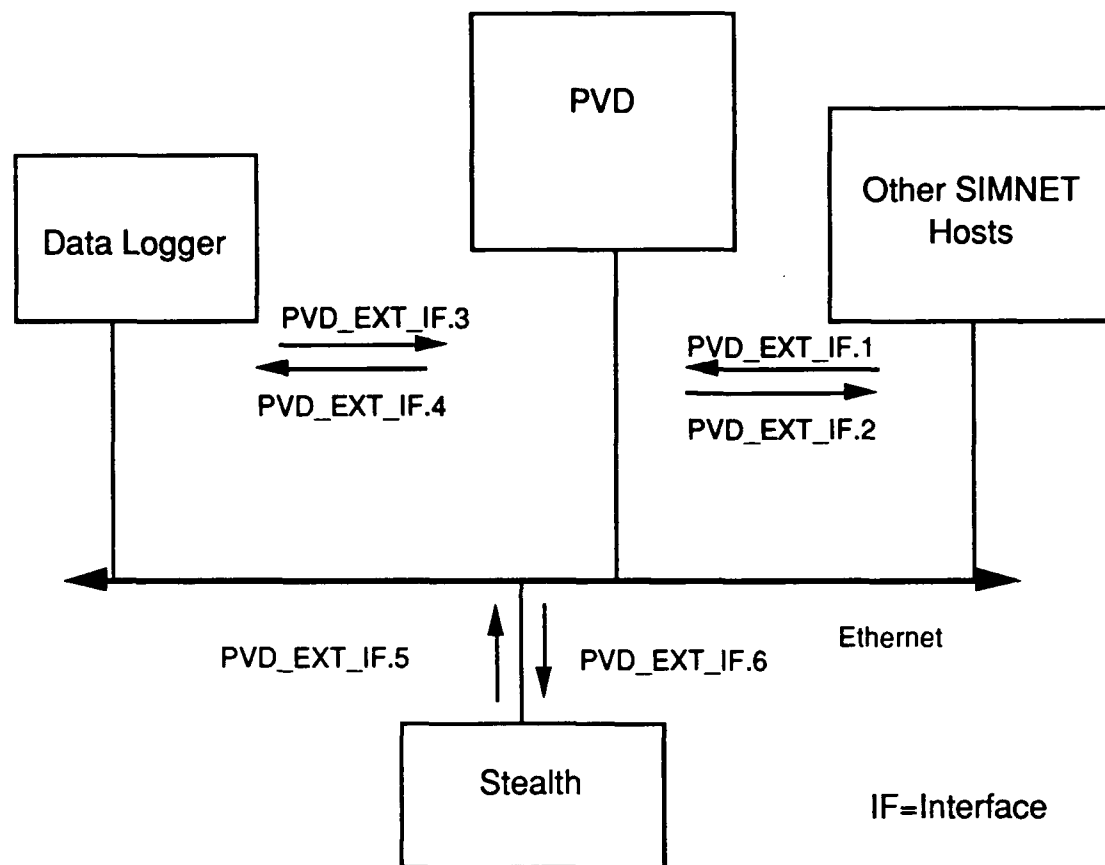


Figure 1.2-1: PVD External Interfaces.

Table 1.2-1 summarizes PVD external interfaces.

Table 1.2-1: Summary of PVD External Interfaces.

Interface ID	From	To	Description	Protocol
PVD_EXT_IF.1	SIM host	PVD	Vehicle Appearance, Exercise Events, etc.	Simulation & Data Collection
PVD_EXT_IF.2	PVD	SIM host	Vehicle and Exercise Selections, Event Waypoints, etc.	Simulation & Data Collection
PVD_EXT_IF.3	Logger	PVD	Connect Response	Logger
PVD_EXT_IF.3	Logger	PVD	Playback Vehicle Appearance, Exercise Events, etc.	Logger
PVD_EXT_IF.4	PVD	Logger	Connect Request	Logger
PVD_EXT_IF.4	PVD	Logger	Playback Controls	Logger
PVD_EXT_IF.5	Stealth	PVD	Stealth Appearance	Stealth
PVD_EXT_IF.6	PVD	Stealth	Stealth Input	Stealth

The PVD Protocol Data Units which support the external interfaces are summarized in Table 1.2-2.

Table 1.2-2: Summary of PVD Protocol Data Units (PDUs).

PDU	CSCI From	CSCI To	Synchron-ization	Protocol	Comments
StealthError	Stealth	PVD	a response to any cmd from the PVD	Stealth	0 if command successful, positive integer; error number if the command failed
Stealth Visibility	PVD	Stealth	none	Stealth	Allows the Stealth to be dynamically sent to any (or all) exercise ID's contained in the PDU
Teleport	PVD	Stealth	none	Stealth	The TeleportPDU is sent to the Stealth vehicle by the PVD to cause it to move to a new location and/or assume a new orientation; contains location and azimuth
Attach	PVD	Stealth	none	Stealth	Command Issued by the PVD causing the Stealth to attach to a particular vehicle; contains vehicleID to which the Stealth should become attached
Attached	Stealth	PVD	Sent by the Stealth whenever it becomes attached to a new vehicle Also sent by the Stealth vehicle whenever it ceases to be attached to a vehicle, for whatever reason	Stealth	Allows the PVD to show the stealth's current attachment; In the detached case, the 'detached' field is set to 1

(Table 1.2-2 is continued on the following page.)

Table 1.2-2 (continued)

PDU	CSCI From	CSCI To	Synchron-ization	Protocol	Comments
Mimic	PVD	Stealth	none	Stealth	Command from the PVD to the Stealth to show the "Out the window view" from a particular viewport in a particular simulator; contains vehicleID that Stealth is to mimic and CIG channel of mimic'ed vehicle to display
Meta-morphose	PVD	Stealth	none	Stealth	Assume the dynamics of another vehicle; contains * Assume the dynamics of another vehicle; Contains thirteen vehicle Dynamics values: FlyingCarpet 1 Jeep 2 M1 3 M2 4 RWA 5 FWA 6 FreeFly 7 Hug 8 Tether 9 Orbit 10 Compass 11 MimicLowMag 12 MimicHiMag 13
Vehicle Appearance	Stealth	PVD	periodic	Stealth	See Simulation Protocol in "SIMNET Network and Protocols."

Other protocols used by PVD are: Simulation, Data Collection, and Logger. Simulation and Data Collection PDUs are described in "SIMNET Network and Protocols, BBN, Report No. 1702." Logger PDUs are described in the Data Logger document, "Data Logger CSCI."

1.3 INTERNAL STRUCTURE

The Plan View Display screen is divided into 6 sections. The most significant section is the map which provides a color-coded, two-dimensional view of the battlefield. The map combines terrain and icon displays. Three sections are located to the right of the map and two sections are located at the bottom of the map.

The sections to the right of the map are named Information, Events, and Menu. Detailed information about a selected vehicle, prompts resulting from menu choices, and user error messages are displayed in the Information Section. The Events Section lists exercise events (collisions, ground or vehicle impacts, and indirect fire bursts). The On-Screen Menu Section can contain the Connect Menu, Top Menu 1, Top Menu 2, or the Overlays Menu.

The leftmost section at the bottom of the PVD screen provides optional displays of map color keys (terrain, intervisibility, or icon) or a help message. The rightmost section displays the coordinates of a selected location, the current icon size and the zoom ratio.

By default, the PVD displays vehicles with an exercise ID of 1. This ID number causes all vehicles on the network to be displayed. A *set exercise* command on the PVD can be issued to display vehicles with another exercise ID number.

PVD functionality may be broken down into these top-level CSCs. Further decomposition of top-level CSCs into lower-level CSCs and CSUs is described in Section 2.

- Menu Handling
- Icons
- Map Handling
- PVD-Level Processing
- Utilities
- Network Processing
- Graphic Support
- Overlays
- Popup Windows
- Tools
- Remote Device Interfaces

Their interfaces are illustrated in Figure 1.3-1, "PVD Internal Interfaces," and summarized in Table 1.3-1, "Summary of PVD Internal Interfaces."

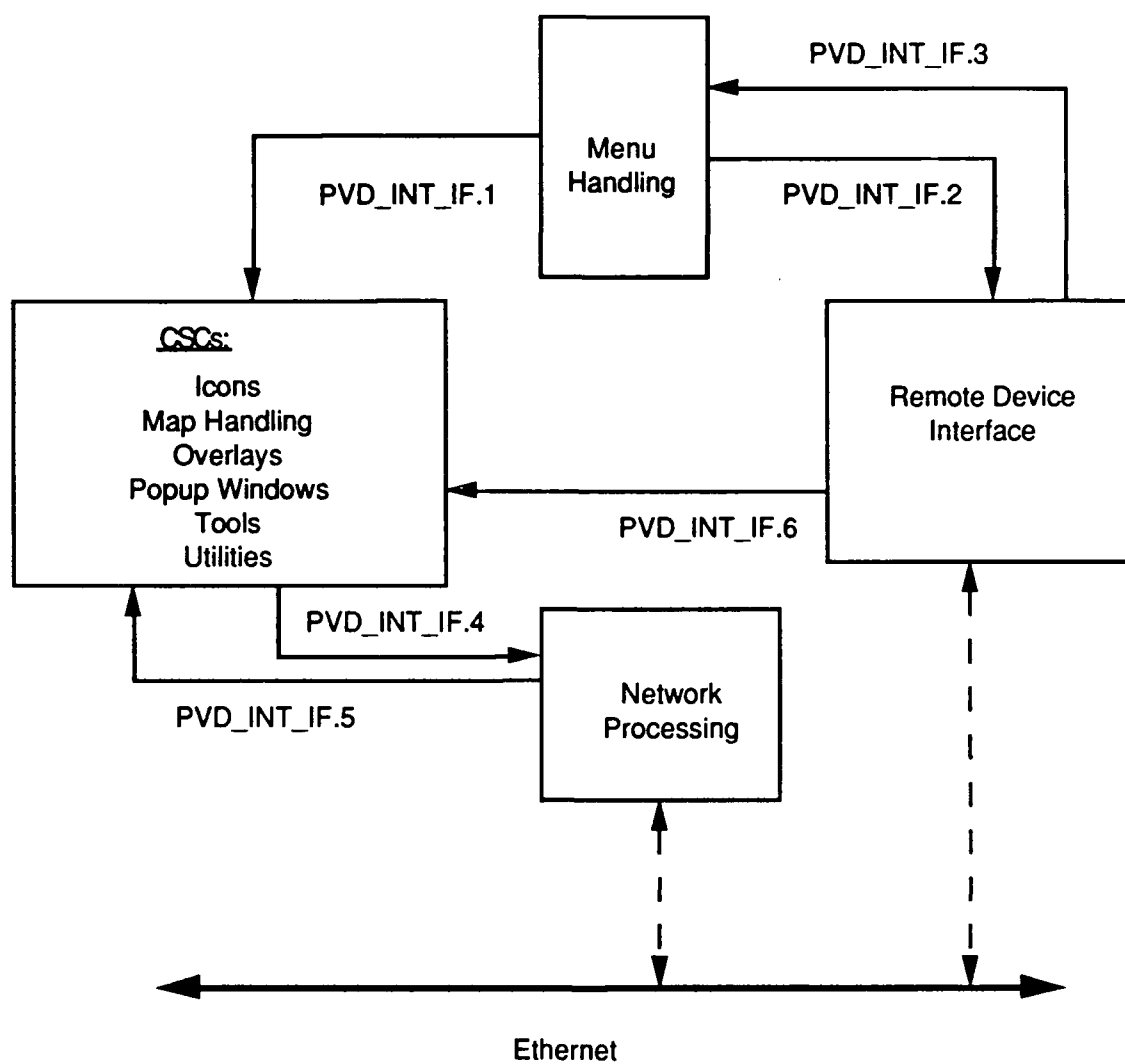


Figure 1.3-1: PVD Internal Interfaces.

Table 1.3-1 summarizes PVD Internal Interfaces.

Table 1.3-1: PVD Internal Interfaces.

Interface ID	From	To	Description	Protocol
PVD_INT_IF.1	Menu H.	Other CSCs	Menu Selections	Calls
PVD_INT_IF.2	Menu H.	Remote Dev. Interfaces	Logger Request Connect	Calla
PVD_INT_IF.3	Remote Dev. Interfaces	Menu H.	Logger Connect Reply	Return
PVD_INT_IF.4	CSUs	Network Process.	Vehicle and Exercise IDs, Events Waypoints	Shared Memory
PVD_INT_IF.5	Network Process.	CSUs	Vehicle/Stealth Appearance, Exercise Events, etc.	Shared Memory
PVD_INT_IF.6	Remote Dev. Interfaces	CSUs	Playback Vehicle Appearance, Exercise Events	Shared Memory
PVD_INT_IF.6	Remote Dev. Interfaces	CSUs	Stealth Menu Selections	Shared Memory

For detailed information on vehicle and exercise event message contents and structures, refer to "SIMNET Network and Protocols", chapters 7 and 8.

1.4 CONFIGURATION AND CONFIGURATION MANAGEMENT

PVD is written in the "C" programming language. It uses a small number of UNIX system calls, consistent with AT&T UNIX System V.

It is currently implemented on Concurrent 5600 and 6600 series machines. Briefly, the minimum hardware required to support the PVD software is:

- 5600 Concurrent host
- 4 MBy main memory
- 318 MBy hard disk
- 12 plane Concurrent Graphics system
- Communications Machinery Corporation ethernet card

The PVD currently runs under Concurrent Real Time Unix (RTU) version 4.0A. In most cases it will only require a recompile to operate with other versions of RTU. The PVD also requires appropriately customized versions of the following auxiliary files:

```
/simnet/bin/pvd_init.dat
/simnet/bin/enp.bin
/simnet/data/network.def
```

1.5 TERMINOLOGY AND DOCUMENTATION

1.5.1 Terms and Abbreviations

bit map	pattern of bit settings which determine a visual display. Each "on" bit turns on a picture-element (pixel).
bitblt	bit block transfer.
client	a user's application program.
CSC	Computer Software Component. A distinct part of a computer software configuration item (CSCI). CSCs may be further decomposed into other CSCs and Computer Software Units (CSUs).
CSCI	Computer Software Configuration Item. A configuration item for computer software.
CSU	Computer Software Unit. An element of a Computer Software Component (CSC) that is separately testable.
fb	frame buffer (graphics programming).
host/Host	mainframe
IF	Interface
PDU	Protocol Data Unit.
plane	in graphics programming, 'plane' refers to a block of raster memory for storing a bit map. One memory plane allows two bit-settings per pixel- on or off. Multiple planes provide multiple bit settings per pixel; multiple planes are used to achieve hue, layering, or depth effects.
Protocol	a set of procedures and data formats that govern data communications.
PVD	PlanView Display.
resource	in windows programming, a resource is a complex shareable set of data structures that defines a window object.
segment	in graphics programming, 'segment' refers to set of related graphics commands.
server	A program that controls the graphics system. A server acts as an intermediary between a client and system resources.
shell	a UNIX command interpreter.
SIM host	simulator host (vehicle or Management Command and Control).
Subscription	Membership of a simulation entity in a multicast group; subscribing and unsubscribing are SIMNET network services provided by the association protocol of SIMNET
Toolkit	contains sets of X-Window options, each set controlling an aspect of windows, for example, a window's geometry.
UI	User Interface
UTM	Universal terrain meridian
Widgets	X-Window "objects" such as, menus, scrollbars, and buttons.

1.5.2 References

The following documents are relevant to an understanding of the PVD and its implementation in "C":

- *The "C" Programming Language*. Kernighan and Ritchie, Prentice-Hall, 1978.
- *DOD-STD-2167A & Associated DIDs and DOD-STD-2168 & Associated DID*. David Maibor Associates. Needham Heights, MA, February 29, 1988.
- *Graphics Application Programming Manual*, Massachusetts Computer Corporation, 1988.
- *Introducing UNIX SYSTEM V*. Rachel Morgan and Henry McGilton, New York, McGraw-Hill, 1987
- "SIMNET Data Collection and Review," BBN STC, October 1988.
- *The SIMNET Maintenance Manual*, version 6.2, March 1990.
- "SIMNET Network and Protocols," BBN Report No. 1702.
- "SIMNET Stealth Vehicle Functional Specification and Operator's Manual," BBN Report No. 7348.

2 PVD CSC DESCRIPTIONS

In this section, top-level CSCs are further decomposed into lower-level CSCs and CSUs. Each CSU is generally mapped to a source file or library. The contents of each CSU is summarized.

Some top-level CSCs are broken down into lower-level CSCs. At the end of each top-level CSC description, a breakdown into lower-level CSCs, if any, is shown, and each lower-level CSC functionally summarized. If there are no lower-level CSCs corresponding to the top-level CSC, it is decomposed into one or more CSUs. If there are lower-level CSUs, each of these, in turn, is decomposed into one or more CSUs.

Interfacing Constraints:

PVD interfaces are designed to be as modular as possible. This constrains the representation of PVD interfaces as follows:

- 1) Design elements at the same level of hierarchy are represented as interfacing directly only with their design peers; that is, a top-level CSC is represented as interfacing directly only with other top-level CSCs, a lower-level CSC is represented as interfacing directly with other lower-level CSCs, and a CSU, as interfacing directly only with other CSUs.
- 2) Design peers subordinated under different parent nodes are represented as interfacing with one another only through their parent nodes. For example, a CSU does not interface directly with a CSU belonging to a different CSC; instead, the CSUs interface through their parent CSCs.

CSU function calls represents an exception to the second constraint. One function may call any other function within its scope of linkage.

Breakdown of the PVD CSCI is illustrated in Figure 2-1.

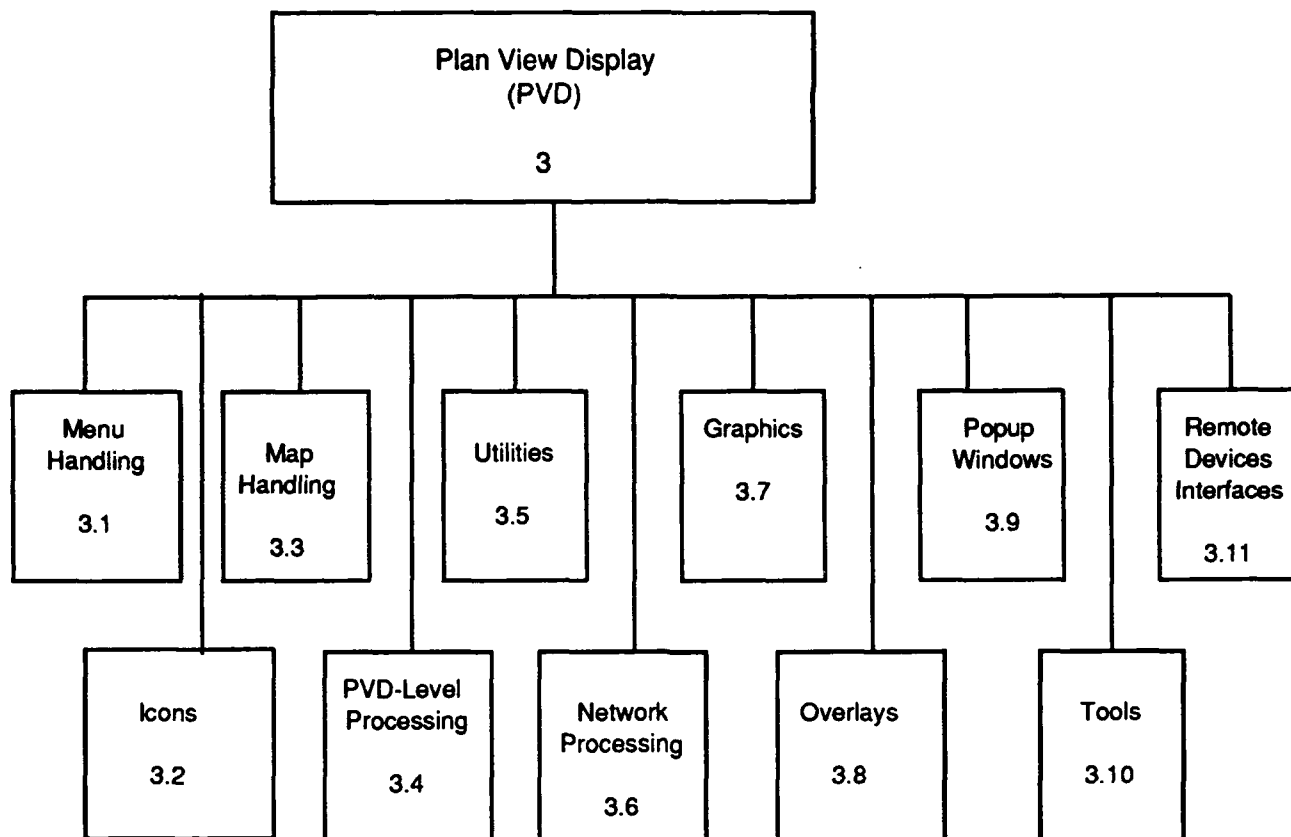


Figure 2-1: PVD CSCI Structure.

This section provides a functional summary decomposition for each of the CSCs listed above.

About CSU Descriptions:

A PVD CSU has been mapped to a source file (*.c, *.h, or *.a). A full CSU description includes the following items where they apply:

Section number Filename

[brief summary of what file does]

Global Variables

External Static Variables

Function number Function

Parameters

Returns

Error Messages

Calls

[list of functions called--generally does not include standard 'C' functions or UNIX system calls]

Function number Function

Search Paths

Unless explicitly provided, the search path for a source file is--/simnet/pvd. *Unless explicitly provided*, the search path for a library (filename.a) is--/simnet/lib.

2.1 MENU HANDLING CSC DESCRIPTION

This CSC controls all PVD menu displays. It provides display and input handling of selected menu commands and options, error replies, further prompts, and navigates among the various menu displays. This CSC also processes and dispatches menu selections to other CSCs for further processing. Figure 2.1-1 illustrates the decomposition of this CSC.

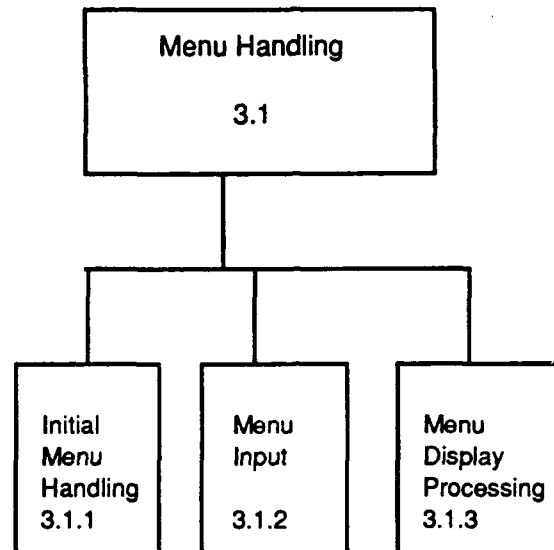


Figure 2.1-1: Menu Handling CSC Structure.

2.1.1 Initial Menu Handling CSC Description

This CSC is the main entry for menu handling.

This CSC functionality is broken down into the CSUs shown in Figure 2.1.1-1.

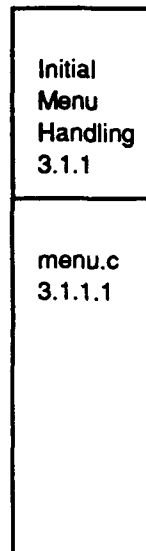


Figure 2.1.1-1: Menu Handling--Initial Menu Handling CSC Structure.

2.1.1.1 menu.c CSU Description (/simnet/pvd/lib/pvdiface.a)

This CSU dispatches a command choice from a menu. This file interprets the user's gestures with the mouse and invokes the appropriate actions.

Table 2.1.1.1-1: Function Summary - init_menu_params()

This function initializes the parameters which describe the menu, for example, menu width.

Table 2.1.1.1-2: Function Summary - init_menus()

Calls	
Function	Where Described
init_menu_params	this file
init_menu	Concurrent graphics library subroutine
set_menu_window_top	Pvd_windows.c
init_event_flag_strings	this file
init_database_strings	this file
display_nlos_menu	this file
display_stealth_menu	this file
display_catc_menu	this file
display_activate_menu	this file
init_activate_strings	activate.c

Table 2.1.1.1-3: Function Summary - init_menu(menu_table, menu, type)

Parameters		
Parameter	Type	Where Typedef Declared
menu_table	MENU_ENTRY	menu.h
menu	pointer to register MENU	menu.h
type	int	standard

Calls	
Function	Where Described
set_entry_locations	this file

Table 2.1.1.1-4: Function Summary - set_entry_locations(menu)

Parameters		
Parameter	Type	Where Typedef Declared
menu	pointer to MENU	menu.h

Table 2.1.1.1-5: Function Summary - draw_menu(menu)

Parameters		
Parameter	Type	Where Typedef Declared
menu	pointer to register MENU	menu.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgiclearpln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
make title	this file
draw disabled backgnd	this file

Table 2.1.1.1-6: Function Summary - draw_disabled_backgnd(grp, row)

Parameters		
Parameter	Type	Where Typedef Declared
grp	pointer to MENU GRP	menu.h
row	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.1.1.1-7: Function Summary - make_title(row_loc, title)

Parameters		
Parameter	Type	Where Typedef Declared
row_loc	int	standard
title	pointer to char	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine

Table 2.1.1.1-8: Function Summary - init_menu_window()

Calls	
Function	Where Described
mgidew	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine

Table 2.1.1.1-9: Function Summary - menu_select(row, column, x, y)

Parameters		
Parameter	Type	Where Typedef Declared
row	pointer to int	standard
column	pointer to int	standard
x, y	pointer to int	standard

Return Values		
Return Value	Type	Meaning
0	int	menu item not selected
1	int	menu item selected

Table 2.1.1.1-10: Function Summary - handle_menu(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

**Table 2.1.1.1-11: Function Summary - chg_status_of_menu_grp
(grp_name, menu, status)**

Parameters		
Parameter	Type	Where Typedef Declared
grp_name	pointer to char	standard
menu	pointer to MENU	menu.h
status	int	standard

Calls	
Function	Where Described
drraw menu	this file

Table 2.1.1.1-12: Function Summary - change_phantom_status(status)

Parameters		
Parameter	Type	Where Typedef Declared
status	int	standard

Calls	
Function	Where Described
change_status of menu_grp	this file

Table 2.1.1.1-13: Function Summary - change_logger_status(status)

Parameters		
Parameter	Type	Where Typedef Declared
status	int	standard

Calls	
Function	Where Described
chg_status of menu_grp	this file

Table 2.1.1.1-14: Function Summary - change_dataprobe_status(status)

Parameters		
Parameter	Type	Where Typedef Declared
status	int	standard

Calls	
Function	Where Described
chg_status of menu_grp	this file

Table 2.1.1.1-15: Function Summary - set_menu_icon_scale(value)

Parameters		
Parameter	Type	Where Typedef Declared
value	int	standard

Calls	
Function	Where Described
check mark	Menu_fea.c

Table 2.1.1.1-16: Function Summary - draw_setup_menu()

Calls	
Function	Where Described
draw_menu	this file

Table 2.1.1.1-17: Function Summary - draw_route_menu()

Calls	
Function	Where Described
mgiclearpin	Concurrent graphics library subroutine
draw_menu	this file

Table 2.1.1.1-18: Function Summary - draw_catc_menu()

Calls	
Function	Where Described
uninitialize mark	
draw routes	
draw menu	this file
set mouse text	Pvd_windows.c
mgiclearpin	Concurrent graphics library subroutine
check mark	Menu_fea.c

Table 2.1.1.1-19: Function Summary - draw_feature_menu()

Calls	
Function	Where Described
uninitialize mark	
mgiclearpin	Concurrent graphics library subroutine
draw features	
draw menu	this file
set mouse text	Pvd_windows.c

Table 2.1.1.1-20: Function Summary - draw_nlos_courses_menu()

Calls	
Function	Where Described
uninitialize mark	
draw routes	
draw menu	this file
set mouse text	Pvd_windows.c
check_mark	Menu_fea.c

Table 2.1.1.1-21: Function Summary - draw_overlay_menu()

Calls	
Function	Where Described
draw menu	this file
overlay mouse	Overlayif.c

Table 2.1.1.1-22: Function Summary - get_current_menu()

Return Values		
Return Value	Type	Meaning
current_menu	pointer to MENU	returns pointer to current menu

Table 2.1.1.1-23: Function Summary - init_event_flag_strings()

Calls	
Function	Where Described
pathname	
get db directory	Init env.c
edit menu strings	menu_fea.c

Table 2.1.1.1-24: Function Summary - init_database_strings()

Calls	
Function	Where Described
get db names	Init env.c
edit menu strings	Menu_fea.c

Functions 25 - 28 enable access to their respective menus:

Table 2.1.1.1-25: Function Summary - flag_activate_menu()

This function enables access to the activate menu.

Table 2.1.1.1-26: Function Summary - flag_nlos_menu()

This function enables access to the nlos menu.

Table 2.1.1.1-27: Function Summary - flag_stealth_menu()

This function enables access to the stealth menu.

Table 2.1.1.1-28: Function Summary - flag_catc_menu()

This function enables access to the catc menu.

Table 2.1.1.1-29: Function Summary - display_nlos_menu()

Calls	
Function	Where Described
edit_menu_func	menu_fea.c

Table 2.1.1.1-30: Function Summary - display_stealth_menu()

Calls	
Function	Where Described
edit_menu_func	Menu fea.c

Table 2.1.1.1-31: Function Summary - display_activate_menu()

Calls	
Function	Where Described
edit_menu_func	menu_fea.c

Table 2.1.1.1-32: Function Summary - display_catc_menu()

This function is inactive.

Table 2.1.1.1-33: Function Summary - set_default_menu(menu_name)

Parameters		
Parameter	Type	Where Typedef Declared
menu_name	pointer to char	standard

Table 2.1.1.1-34: Function Summary - draw_default_menu()

Calls	
Function	Where Described
draw menu	this file

2.1.2 Menu Input CSC Description

This CSC processes user commands and options. It makes use of these Utilities CSCs, Text Input/Output (Section 3.5.2) and Command Parsing (Section 3.5.3).

This CSC functionality is broken down into the CSUs shown in Figure 2.1.2-1.

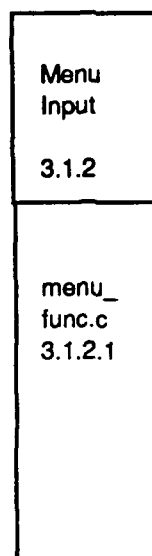


Figure 2.1.2-1: Menu Handling--Menu Input CSC Structure.

2.1.2.1 menu_func.c CSU Description (/simet/pvd/lib/pvdiface.a)

This CSU contains routines for managing all functions called by the menu software, as defined in menu.c.

Table 2.1.2.1-1: Function Summary - nop()

This function performs no action.

Table 2.1.2.1-2: Function Summary - menu_zoom_in()

Calls	
Function	Where Described
handle_quickzoom	handle_input.c

Table 2.1.2.1-3: Function Summary - menu_zoom_out()

Calls	
Function	Where Described
handle_zoomout	handle_input.c

Table 2.1.2.1-4: Function Summary - menu_pan()

Calls	
Function	Where Described
handle_pan	handle_input.c

Table 2.1.2.1-5: Function Summary - menu_zoom_125()

Calls	
Function	Where Described
handle_zoom_to_scale	handle_input.c

Table 2.1.2.1-6: Function Summary - menu_zoom_50()

Calls	
Function	Where Described
handle_zoom_to_scale	handle_input.c

Table 2.1.2.1-7: Function Summary - menu_zoom_25()

Calls	
Function	Where Described
handle_zoom_to_scale	handle_input.c

Table 2.1.2.1-8: Function Summary - menu_top_level()

Calls	
Function	Where Described
handle_update_display	handle_input.c

Table 2.1.2.1-9: Function Summary - menu_display_zoom()

Calls	
Function	Where Described
handle_display_zoom	handle_input.c

Table 2.1.2.1-10: Function Summary - menu_previous_view()

Calls	
Function	Where Described
handle_previous_view	handle_input.c

Table 2.1.2.1-11: Function Summary - menu_next_view()

Calls	
Function	Where Described
handle_next_view	handle_input.c

Table 2.1.2.1-12: Function Summary - menu_save_zoom()

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
save_zoom	Save_zoom.c
init_cursor	Pvd_iface.c

Table 2.1.2.1-13: Function Summary - menu_cross_section()

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
cross_section	Cross.c
clear_selected_points	
init_cursor	PVD_iface.c

Table 2.1.2.1-14: Function Summary - menu_colin()

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.d
colin_cross	Cross.c
init_cursor	Pvd_iface.c

Table 2.1.2.1-15: Function Summary - menu_ruler()

Calls	
Function	Where Described
text_ruler	Ruler.c

Table 2.1.2.1-16: Function Summary - menu_help()

Calls	
Function	Where Described
help	Help.c

Table 2.1.2.1-17: Function Summary - menu_track()

Calls	
Function	Where Described
init track	track.c

Table 2.1.2.1-18: Function Summary - menu_test_flag()

Calls	
Function	Where Described
send event flag	
clear selected vehicles	Select.c

Table 2.1.2.1-19: Function Summary - menu_teleport()

Calls	
Function	Where Described
clear selected vehicles	Select.c

Table 2.1.2.1-20: Function Summary - menu_area()

Calls	
Function	Where Described
get view height	Get view.c
get target height	Get view.c
get view range	Get view.c
init working cursor	Pvd iface.c
real intervis 360	Intervis.c
init cursor	Pvd iface.c

Table 2.1.2.1-21: Function Summary - menu_veh_to_veh()

Calls	
Function	Where Described
do vehicle to vehicle	Vehicle to v.c
clear selected vehicles	Select.c

Table 2.1.2.1-22: Function Summary - menu_pt_to_veh()

Calls	
Function	Where Described
get view height	Get view.c
get target height	Get view.c
get elevation	Get elev.c
mgjunscalexy	Concurrent graphics library subroutine
intervis_vehicles	Intervis.c
encode_force	vehicle_to_v.c

Table 2.1.2.1-23: Function Summary - menu_seg_pt_to_pt()

Calls	
Function	Where Described
get view height	Get view.c
get target height	Get view.c

Table 2.1.2.1-24: Function Summary - menu_pt_to_pt()

Calls	
Function	Where Described
intervisibility	Intervis.c
get view height	Get view.c
get target height	Get view.c

Table 2.1.2.1-25: Function Summary - menu_emb()

Calls	
Function	Where Described
emb attenuate	Intervis.c
get target height	Get view.c

Table 2.1.2.1-26: Function Summary - menu_space()

Calls	
Function	Where Described
get all memory	Memory.c

Table 2.1.2.1-27: Function Summary - menu_clear()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
force redraw of icons	Icon.c

Table 2.1.2.1-28: Function Summary - menu_intervis_clear()

Calls	
Function	Where Described
uninitialize mark	
mgiclearph	Concurrent graphics library subroutine
clear all veh to veh	Vehicle to v.c
force redraw of icons	Icon.c

Table 2.1.2.1-29: Function Summary - menu_stealth_teleport()

Calls	
Function	Where Described
mgjunscalexy	Concurrent graphics library subroutine
get elevation	Get_elev.c
teleport stealth	Place.c

Table 2.1.2.1-30: Function Summary - menu_stealth_teleport_and_azimuth()

Calls	
Function	Where Described
mgiscalexy	Concurrent graphics library subroutine
get elevation	Get_elev.c

Table 2.1.2.1-31: Function Summary - menu_stealth_attach()

Calls	
Function	Where Described
teleport stealth	Place.c
attach stealth	Place.c

Table 2.1.2.1-32: Function Summary - menu_stealth_mimic()

Calls	
Function	Where Described
map tank index	
get location	Place.c
teleport stealth	Place.c
mimic vehicle	Place.c

Table 2.1.2.1-33: Function Summary - menu_dl_play()

Calls	
Function	Where Described
play logger	Logger comm.c

Table 2.1.2.1-34: Function Summary - menu_dl_record()

Calls	
Function	Where Described
record logger	Logger comm.c

Table 2.1.2.1-35: Function Summary - menu_dl_go()

Calls	
Function	Where Described
refresh_timestamps	Pvd_misc.c
check mark	Menu_fea.c
continue_logger	Logger_comm.c

Table 2.1.2.1-36: Function Summary - menu_quit()

Calls	
Function	Where Described
pvd_logger_exit	Libptcomm.a
change_logger_status	Menu.c
refresh_timestamps	Pvd_misc.c
check mark	Menu_fea.c

Table 2.1.2.1-37: Function Summary - menu_stop()

Calls	
Function	Where Described
logger_stop	Logger_comm.c

Table 2.1.2.1-38: Function Summary - menu_freeze()

Calls	
Function	Where Described
logger_freeze	Logger_comm.c
check mark	Menu_fea.c

Table 2.1.2.1-39: Function Summary - menu_fast_forward

Calls	
Function	Where Described
get typed input	Interact.c
pvd logger fast forward	
net flush	see libnetif.a in MCC CSCI
logger speed	Logger comm.c

Table 2.1.2.1-40: Function Summary - menu_normal_speed()

Calls	
Function	Where Described
logger speed	Logger comm.c

Table 2.1.2.1-41: Function Summary - menu_forward_abs()

Calls	
Function	Where Described
logger seek	Logger comm.c

Table 2.1.2.1-42: Function Summary - menu_rewind()

Calls	
Function	Where Described
logger seek	Logger comm.c
clear net input	Pvd misc.c

Table 2.1.2.1-43: Function Summary - menu_forward_rel()

Calls	
Function	Where Described
logger seek	Logger comm.c
clear net input	Pvd misc.c

Table 2.1.2.1-44: Function Summary - menu_backward_rel()

Calls	
Function	Where Described
logger seek	Logger comm.c
clear net input	Pvd misc.c

Table 2.1.2.1-45: Function Summary - menu_top_menu_2()

Calls	
Function	Where Described
draw menu	Menu.c

Table 2.1.2.1-46: Function Summary - menu_connect()

Calls	
Function	Where Described
draw setup menu	Menu.c

Table 2.1.2.1-47: Function Summary - menu_overlay()

Calls	
Function	Where Described
draw menu	Menu.c

Table 2.1.2.1-48: Function Summary - menu_operations()

This function is not implemented.

Table 2.1.2.1-49: Function Summary - menu_intelligence()

This function is not implemented.

Table 2.1.2.1-50: Function Summary - menu_fire_support()

This function is not implemented.

Table 2.1.2.1-51: Function Summary - menu_tac_air()

This function is not implemented.

Table 2.1.2.1-52: Function Summary - menu_engineer()

This function is not implemented.

Table 2.1.2.1-53: Function Summary - menu_red()

This function is not implemented.

Table 2.1.2.1-54: Function Summary - menu_blue()

This function is not implemented.

Table 2.1.2.1-55: Function Summary - menu_black()

This function is not implemented.

Table 2.1.2.1-56: Function Summary - menu_green()

This function is not implemented.

Table 2.1.2.1-57: Function Summary - menu_thin()

This function is not implemented.

Table 2.1.2.1-58: Function Summary - menu_thick()

This function is not implemented.

Table 2.1.2.1-59: Function Summary - menu_dashed()

This function is not implemented.

Table 2.1.2.1-60: Function Summary - menu_nodify()

This function is not implemented.

Table 2.1.2.1-61: Function Summary - menu_not_modify()

This function is not implemented.

Table 2.1.2.1-62: Function Summary - menu_display()

This function is not implemented.

Table 2.1.2.1-63: Function Summary - menu_100x()

Calls	
Function	Where Described
check_mark	Menu_fea.c
set_symbolic_mode	Icon.c
scale_icon	Icon.c

Table 2.1.2.1-64: Function Summary - menu_50x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-65: Function Summary - menu_20x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-66: Function Summary - menu_10x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-67: Function Summary - menu_5x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-68: Function Summary - menu_2x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-69: Function Summary - menu_1x()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c
scale icon	Icon.c

Table 2.1.2.1-70: Function Summary - menu_symbolic()

Calls	
Function	Where Described
check mark	Menu fea.c
set symbolic mode	Icon.c

Table 2.1.2.1-71: Function Summary - menu_10K_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change grid interval	handle input.c

Table 2.1.2.1-72: Function Summary - menu_1k_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change grid interval	handle input.c

Table 2.1.2.1-73: Function Summary - menu_500_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change grid interval	handle input.c

Table 2.1.2.1-74: Function Summary - menu_250_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change grid interval	handle input.c

Table 2.1.2.1-75: Function Summary - menu_60_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change contour interval	handle input.c

Table 2.1.2.1-76: Function Summary - menu_20_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change contour interval	handle input.c

Table 2.1.2.1-77: Function Summary - menu_10_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change contour interval	handle input.c

Table 2.1.2.1-78: Function Summary - menu_5_m()

Calls	
Function	Where Described
check mark	Menu fea.c
change contour interval	handle input.c

Table 2.1.2.1-79: Function Summary - menu_set_view_height()

Calls	
Function	Where Described
get view height	Get view.c
set view height	Get view.c
change intervis key height	

Table 2.1.2.1-80: Function Summary - menu_set_target_height()

Calls	
Function	Where Described
get target height	Get view.c
set target height	Get view.c
change intervis key text	Pvd windows.c

Table 2.1.2.1-81: Function Summary - menu_view_range()

Calls	
Function	Where Described
get view range	Get view.c
set view range	Get view.c
change intervis key text	Pvd windows.c

Table 2.1.2.1-82: Function Summary - menu_toggle_intervis_align()

Calls	
Function	Where Described
view all align	Vehicle to v.c
view opposing align	Vehicle to v.c
edit entry	Menu fea.c

Table 2.1.2.1-83: Function Summary - menu_set_all_align()

Calls	
Function	Where Described
view all align	Vehicle to v.c
check mark	Menu fea.c

Table 2.1.2.1-84: Function Summary - menu_set_opposing_align()

Calls	
Function	Where Described
view opposing align	Vehicle to v.c
check mark	Menu fea.c

Table 2.1.2.1-85: Function Summary - menu_terrain()

Calls	
Function	Where Described
check mark	Menu fea.c
display map key	Pvd windows.c

Table 2.1.2.1-86: Function Summary - menu_icon()

Calls	
Function	Where Described
check mark	Menu fea.c
display icon key	Pvd windows.c

Table 2.1.2.1-87: Function Summary - menu_nlos_hazards()

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
highlight_altitude_hazard	New_zoom.c
init_cursor	Pvd_iface.c

Table 2.1.2.1-88: Function Summary - menu_set_altitude_hazard()

Calls	
Function	Where Described
get_typed_input_float	interact.c

Table 2.1.2.1-89: Function Summary - menu_intervis()

Calls	
Function	Where Described
check_mark	Menu_fea.c
display_intervis_key	Pvd_windows.c

Table 2.1.2.1-90: Function Summary - menu_mouse_help()

Calls	
Function	Where Described
check_mark	Menu_fea.c
display_mouse_key	Pvd_windows.c

Table 2.1.2.1-91: Function Summary - menu_zone()

This function is not implemented.

Table 2.1.2.1-92: Function Summary - menu_interactive()

This function is not implemented.

Table 2.1.2.1-93: Function Summary - menu_xy_plot()

This function is not implemented.

Table 2.1.2.1-94: Function Summary - menu_top_menu_1()

Calls	
Function	Where Described
draw_menu	Menu.c

Table 2.1.2.1-95: Function Summary - menu_nlos()

Calls	
Function	Where Described
draw menu	Menu.c

Table 2.1.2.1-96: Function Summary - menu_stealth()

Calls	
Function	Where Described
draw menu	Menu.c

Table 2.1.2.1-97: Function Summary - menu_contour()

Calls	
Function	Where Described
handle_contour	handle_input.c

Table 2.1.2.1-98: Function Summary - menu_toggle_shade()

Calls	
Function	Where Described
toggle_shade	Unshade.c

Table 2.1.2.1-99: Function Summary - menu_set_exercise()

Calls	
Function	Where Described
get_exercise	Pvd_misc.c
set_exercise	Pvd_misc.c
get_typed_input	Interact.c
change_exercise	Pvd_windows.c

Table 2.1.2.1-100: Function Summary - menu_grid()

Calls	
Function	Where Described
handle_grid	handle_input.c

Table 2.1.2.1-101: Function Summary - menu_connect_pt()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-102: Function Summary - menu_dont_connect_pt()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-103: Function Summary - menu_connect_dl()

Calls	
Function	Where Described
check mark	Menu fea.c
connect to logger	Logger.comm.c

Table 2.1.2.1-104: Function Summary - menu_disconnect_dl()

Calls	
Function	Where Described
disconnect_dl	Logger.comm.c
check mark	Menu fea.c

Table 2.1.2.1-105: Function Summary - menu_connect_dp()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-106: Function Summary - menu_dont_connect_dp()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-107: Function Summary - menu_done()

Calls	
Function	Where Described
init connections	Logger.comm.c

Table 2.1.2.1-108: Function Summary - menu_logging_on()

Calls	
Function	Where Described
logging on	t windows.c
check mark	Menu fea.c

Table 2.1.2.1-109: Function Summary - menu_logging_off()

Calls	
Function	Where Described
logging off	t windows.c
check mark	Menu fea.c

Table 2.1.2.1-110: Function Summary - menu_clear_tanks()

Calls	
Function	Where Described
clear selected vehicles	Select.c

Table 2.1.2.1-111: Function Summary - menu_save_route()

Calls	
Function	Where Described
save target list	

Table 2.1.2.1-112: Function Summary - menu_route_redraw()

Calls	
Function	Where Described
draw routes	

Table 2.1.2.1-113: Function Summary - menu_recall_route()

Calls	
Function	Where Described
recall target list	
draw routes	

Table 2.1.2.1-114: Function Summary - menu_mode_done()

This function is not activated and never called.

Table 2.1.2.1-115: Function Summary - goto_top_menu1()

This function is not activated and never called.

Table 2.1.2.1-116: Function Summary - back_to_menu1()

Calls	
Function	Where Described
draw_menu	Menu.c

Table 2.1.2.1-117: Function Summary - back_to_menu2()

Calls	
Function	Where Described
draw_menu	Menu.c

Table 2.1.2.1-118: Function Summary - menu_catc_vehicle()

Calls	
Function	Where Described
check_mark	Menu_fea.c

Table 2.1.2.1-119: Function Summary - menu_catc_mover_oblique()

Calls	
Function	Where Described
check_mark	Menu_fea.c

Table 2.1.2.1-120: Function Summary - menu_catc_mover_frontal()

Calls	
Function	Where Described
check_mark	Menu_fea.c

Table 2.1.2.1-121: Function Summary - menu_catc_personnel()

Calls	
Function	Where Described
check_mark	Menu_fea.c

Table 2.1.2.1-122: Function Summary - menu_catc_place_tgt()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-123: Function Summary - menu_move_tgt_or_pt()

Calls	
Function	Where Described
check mark	Menu fea.c

Table 2.1.2.1-124: Function Summary - menu_save_target_list()

Calls	
Function	Where Described
save target list	

Table 2.1.2.1-125: Function Summary - menu_recall_target_list()

Calls	
Function	Where Described
recall target list	
draw routes	

Table 2.1.2.1-126: Function Summary - menu_list_catc_files ()

Calls	
Function	Where Described
list catc files	

Table 2.1.2.1-127: Function Summary - menu_catc_transfer_file()

Calls	
Function	Where Described
transfer catc file	

Table 2.1.2.1-128: Function Summary - menu_delete_catc_file()

Calls	
Function	Where Described
delete catc file	

Table 2.1.2.1-129: Function Summary - menu_show_tgt_list()

Calls	
Function	Where Described
uninitialize mark	
menu clear	
check mark	Menu_fea.c
draw routes	
rt or catc present	

Table 2.1.2.1-130: Function Summary - menu_catc_remove_tgt()

Calls	
Function	Where Described
remove targets	
check mark	Menu_fea.c

Table 2.1.2.1-131: Function Summary - menu_catc_clear_tgt_list()

Calls	
Function	Where Described
clear targets	

Table 2.1.2.1-132: Function Summary - menu_symbol_create()

Calls	
Function	Where Described
symbol create	Symbolif.c

Table 2.1.2.1-133: Function Summary - menu_symbol_edit()

Calls	
Function	Where Described
symbol edit	Symbolif.c

Table 2.1.2.1-134: Function Summary - menu_overline_create()

Calls	
Function	Where Described
overline create	

Table 2.1.2.1-135: Function Summary - menu_overline_newfeatures()

Calls	
Function	Where Described
overline_newfeatures	

Table 2.1.2.1-136: Function Summary - menu_overline_newshape()

Calls	
Function	Where Described
overline_newshape	

Table 2.1.2.1-137: Function Summary - menu_over_save()

Calls	
Function	Where Described
save_over	

Table 2.1.2.1-138: Function Summary - menu_over_read()

Calls	
Function	Where Described
read_over	

Table 2.1.2.1-139: Function Summary - menu_over_list()

Calls	
Function	Where Described
list_over_files	

Table 2.1.2.1-140: Function Summary - menu_over_delete()

Calls	
Function	Where Described
delete_over_files	

Table 2.1.2.1-141: Function Summary - menu_over_transfer()

Calls	
Function	Where Described
transfer_over_file	

Table 2.1.2.1-142: Function Summary - menu_over_edit()

Calls	
Function	Where Described
edit_over	

Table 2.1.2.1-143: Function Summary - menu_checkpt()

Calls	
Function	Where Described
controlpt_check	

Table 2.1.2.1-144: Function Summary - menu_coordpt()

Calls	
Function	Where Described
controlpt_coord	

Table 2.1.2.1-145: Function Summary - menu_conpt()

Calls	
Function	Where Described
controlpt_con	

Table 2.1.2.1-146: Function Summary - menu_linkpt()

Calls	
Function	Where Described
controlpt_link	

Table 2.1.2.1-147: Function Summary - menu_relpt()

Calls	
Function	Where Described
controlpt_rel	

Table 2.1.2.1-148: Function Summary - menu_passpt()

Calls	
Function	Where Described
controlpt_pass	

Table 2.1.2.1-149: Function Summary - menu_deletept()

Calls	
Function	Where Described
controlpt_delete	

Table 2.1.2.1-150: Function Summary - menu_deleteov()

Calls	
Function	Where Described
delete_overobj	

Table 2.1.2.1-151: Function Summary - menu_reduceov()

Calls	
Function	Where Described
reduce_overobj	

Table 2.1.2.1-152: Function Summary - menu_moveov()

Calls	
Function	Where Described
move_overobj	

Table 2.1.2.1-153: Function Summary - menu_addtextov()

Calls	
Function	Where Described
addtext_overobj	

Table 2.1.2.1-154: Function Summary - menu_eraseov()

Calls	
Function	Where Described
erase_overlay	

Table 2.1.2.1-155: Function Summary - menu_redrawov()

Calls	
Function	Where Described
zoom_overlay	Overlayif.c

Table 2.1.2.1-156: Function Summary - menu_resetov()

Calls	
Function	Where Described
reset_overlay	

Table 2.1.2.1-157: Function Summary - menu_change_format()

Calls	
Function	Where Described
display xy	
set format xy	
display utm	

Table 2.1.2.1-158: Function Summary - do_menu_event(event_number)

Parameters		
Parameter	Type	Where Typedef Declared
event_number	int	standard

Calls	
Function	Where Described
send event flag	
clear selected vehicles	Select.c

Table 2.1.2.1-159: Function Summary - menu_event1()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-160: Function Summary - menu_event2()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-161: Function Summary - menu_event3()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-162: Function Summary - menu_event4()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-163: Function Summary - menu_event5()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-164: Function Summary - menu_event6()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-165: Function Summary - menu_event7()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-166: Function Summary - menu_event8()

Calls	
Function	Where Described
do_menu_event	this file

Table 2.1.2.1-167: Function Summary - menu_select_vehicle()

Calls	
Function	Where Described
select_vehicle_by_name	

Table 2.1.2.1-168: Function Summary - menu_side()

Calls	
Function	Where Described
set_distinguished	g.graph.c

Table 2.1.2.1-169: Function Summary - menu_dump_table()

Calls	
Function	Where Described
dump table to file	

Table 2.1.2.1-170: Function Summary - menu_detail()

This function is not implemented.

Table 2.1.2.1-171: Function Summary - menu_activate()

Calls	
Function	Where Described
draw menu	Menu.c

Table 2.1.2.1-172: Function Summary - menu_activate_1()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-173: Function Summary - menu_activate_2()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-174: Function Summary - menu_activate_3()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-175: Function Summary - menu_activate_4()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-176: Function Summary - menu_activate_5()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-177: Function Summary - menu_activate_6()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-178: Function Summary - menu_activate_7()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-179: Function Summary - menu_activate_8()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-180: Function Summary - menu_activate_9()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-181: Function Summary - menu_activate_10()

Calls	
Function	Where Described
get_activate_mode	activate.c
activate_vehicle	activate.c
deactivate_vehicle	activate.c

Table 2.1.2.1-182: Function Summary - menu_activate_vehicle_mode()

Calls	
Function	Where Described
set_activate_mode	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-183: Function Summary - menu_deactivate_vehicle_mode()

Calls	
Function	Where Described
set_activate_mode	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-184: Function Summary - menu_altitude_1()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-185: Function Summary - menu_altitude_2()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-186: Function Summary - menu_altitude_3()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-187: Function Summary - menu_altitude_4()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-188: Function Summary - menu_altitude_5()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-189: Function Summary - menu_altitude_6()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-190: Function Summary - menu_altitude_7()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-191: Function Summary - menu_altitude_8()

Calls	
Function	Where Described
set_altitude	activate.c
check_mark	Menu_fea.c

Table 2.1.2.1-192: Function Summary - menu_altitude_9()

Calls	
Function	Where Described
set altitude	activate.c
check mark	Menu fea.c

Table 2.1.2.1-193: Function Summary - menu_altitude_10()

Calls	
Function	Where Described
set altitude	activate.c
check mark	Menu fea.c

Table 2.1.2.1-194: Function Summary - menu_alt_abs()

Calls	
Function	Where Described
set activation altitude absolute	activate.c
check mark	Menu fea.c

Table 2.1.2.1-195: Function Summary - menu_alt_rel()

Calls	
Function	Where Described
set activation altitude relative	activate.c
check mark	Menu fea.c

Table 2.1.2.1-196: Function Summary - menu_feature_object_info()

Calls	
Function	Where Described
object_info	
check mark	Menu fea.c

Table 2.1.2.1-197: Function Summary - menu_feature_place_object()

Calls	
Function	Where Described
place objects	
check marks	Menu fea.c

Table 2.1.2.1-198: Function Summary - menu_feature_move_object()

Calls	
Function	Where Described
move_objects	
check_mark	Menu_fea.c

Table 2.1.2.1-199: Function Summary - menu_feature_remove_object()

Calls	
Function	Where Described
remove_objects	
check_mark	Menu_fea.c

Table 2.1.2.1-200: Function Summary - menu_feature_save_file()

Calls	
Function	Where Described
save_feature_file	

Table 2.1.2.1-201: Function Summary - menu_feature_recall_file()

Calls	
Function	Where Described
recall_feature_file	

Table 2.1.2.1-202: Function Summary - menu_feature_list_files()

Calls	
Function	Where Described
list_feature_files	

Table 2.1.2.1-203: Function Summary - menu_feature_delete_file()

Calls	
Function	Where Described
delete_feature_file	

Table 2.1.2.1-204: Function Summary - menu_feature_clear_object_list()

Calls	
Function	Where Described
clear_feature_list	

Table 2.1.2.1-205: Function Summary - menu_feature_redraw_objects()

Calls	
Function	Where Described
force feature redraw	

Table 2.1.2.1-206: Function Summary - menu_send_features()

Calls	
Function	Where Described
check_mark	Menu_fea.c

Table 2.1.2.1-207: Function Summary - menu_dont_send_features()

Calls	
Function	Where Described
check_mark	Menu_fea.c
set feature send flag	

Table 2.1.2.1-208: Function Summary - menu_feature_M977()

Calls	
Function	Where Described
set feature type	
check_mark	Menu_fea.c

Table 2.1.2.1-209: Function Summary - menu_feature_Ben_Franklin()

Calls	
Function	Where Described
set feature type	
check_mark	Menu_fea.c

Table 2.1.2.1-210: Function Summary - menu_feature_mi28()

Calls	
Function	Where Described
set feature type	
check mark	Menu_fea.c

Table 2.1.2.1-211: Function Summary - menu_feature_new()

Calls	
Function	Where Described
set feature type	
check mark	Menu_fea.c

Table 2.1.2.1-212: Function Summary - menu_view_ground_vehicles()

Calls	
Function	Where Described
view ground vehicles	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-213: Function Summary - menu_view_platoons()

Calls	
Function	Where Described
view platoons	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-214: Function Summary - menu_view_companies()

Calls	
Function	Where Described
view companies	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-215: Function Summary - menu_view_battalions()

Calls	
Function	Where Described
view battalions	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-216: Function Summary - menu_view_sorties()

Calls	
Function	Where Described
view sorties	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-217: Function Summary - menu_view_flights()

Calls	
Function	Where Described
view flights	Graph.c
clear selected vehicles	Select.c
check mark	Menu_fea.c

Table 2.1.2.1-218: Function Summary - menu_icon_background()

Calls	
Function	Where Described
set_opaque_mode	
check mark	Menu_fea.c

Table 2.1.2.1-219: Function Summary - menu_icon_no_background()

Calls	
Function	Where Described
set_opaque_mode	
check mark	Menu_fea.c

2.1.3 Menu Display Processing CSC Description

This CSC functionality is broken down into the CSUs shown in Figure 2.1.3-1.

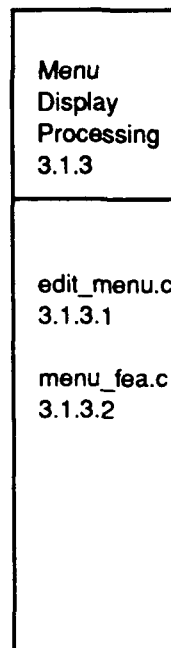


Figure 2.1.3-1: Menu Handling--Menu Display Processing CSC Structure.

2.1.3.1 edit_menu.c

This CSU contains all the routines which change the menu entries.

Table 2.1.3.1-1: Function Summary - cross_menu_entry_on()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-2: Function Summary - cross_menu_entry_off()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-3: Function Summary - set_format_utm()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-4: Function Summary - set_format_xy()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-5: Function Summary - grid_menu_entry_on()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-6: Function Summary - grid_menu_entry_off()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-7: Function Summary - contour_menu_entry_on()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-8: Function Summary - contour_menu_entry_off()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-9: Function Summary - help_menu_entry_on()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-10: Function Summary - help_menu_entry_off()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-11: Function Summary -shading_menu_entry_on()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-12: Function Summary - shading_menu_entry_off()

Calls	
Function	Where Described
edit_entry	Menu_fea.c

Table 2.1.3.1-13: Function Summary - choose_distinguished()

Calls	
Function	Where Described
edit_entry	Menu_fea.c
choose_other	

2.1.3.2 menu_fea.c

This CSU supports all menu drawing features, such as check marks for making menu selections.

Table 2.1.3.2-1: Function Summary - check_mark(grp_name, name, menu)

Parameters		
Parameter	Type	Where Typedef Declared
grp_name	pointer to char	standard
name	pointer to char	standard
menu	pointer to MENU	menu.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.1.3.2-2: Function Summary - edit_menu_strings(grp_name, old_name, new_name, menu, redraw)

Parameters		
Parameter	Type	Where Typedef Declared
grp_name	pointer to char	standard
old_name	pointer to char	standard
new_name	pointer to char	standard
menu	pointer to MENU	menu.h
redraw	int	standard

Calls	
Function	Where Described
get current menu	Menu.c
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Return Values		
Return Value	Type	Meaning
entry_name	NAME_LIST	returns a pointer to the new name list structure

Table 2.1.3.2-3: Function Summary - edit_entry(grp_name, old_name, new_name, menu)

Parameters		
Parameter	Type	Where Typedef Declared
grp_name	pointer to char	standard
old_name	pointer to char	standard
new_name	pointer to char	standard
menu	pointer to MENU	menu.h

Calls	
Function	Where Described
edit_menu_strings	this file

Table 2.1.3.2-4: Function Summary - change_menu_func (menu, name, func)

Parameters		
Parameter	Type	Where Typedef Declared
menu	pointer to MENU	menu.h
name	pointer to NAME_LIST	menu.h
func	pointer to int	standard

Table 2.1.3.2-5: Function Summary - edit_menu_func(grp_name, old_name, new_name, func, menu)

Parameters		
Parameter	Type	Where Typedef Declared
grp_name	pointer to char	standard
old_name	pointer to char	standard
new_name	pointer to char	standard
menu	pointer to MENU	menu.h

Calls	
Function	Where Described
edit_menu_strings	this file
change_menu_func	this file

2.2 ICONS CSC DESCRIPTION

This CSC represents individual vehicles on the PVD by graphics icons that indicate the position and orientation of the hull and the relative orientation of the turret. It allows a user to vary the size of the icons with the selection of a magnification level. A shot is indicated by a line segment emanating from the firing vehicle. This CSC also allows a user to select particular vehicles and obtain detailed information such as the vehicle location, speed, alignment, identification number, and repair and supply status.

This CSC allows a user to vary the size of an icon with the selection of a magnification level.

By default, the PVD displays vehicles in exercise 1. The user can display vehicles in a different exercise by supplying a different exercise number.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-2.

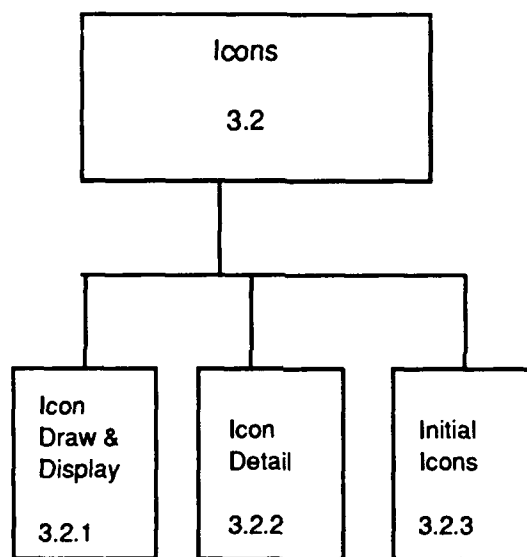


Figure 2-2: Icons CSC Structure.

2.2.1 Icon Draw and Display CSC Description

This lower-level CSC provides the processing for icon drawing and displays.

This CSC functionality is broken down into the CSUs shown in Figure 2.2.1-1. Icon selection functionality is also supported by the CSU, select .c. (Section 3.5.5.*).

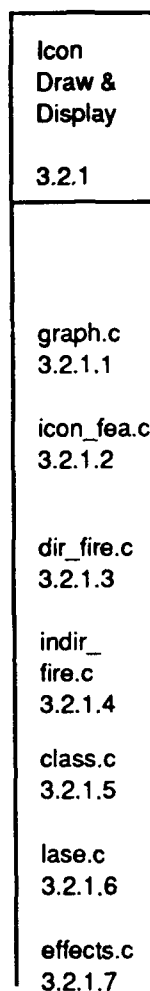


Figure 2.2.1-1: Icons--Icon Draw & Display CSC Structure.

2.2.1.1 graph.c

This CSU controls the graphics display of vehicle states for use with Plan View Display.

Table 2.2.1.1-1: Function Summary - set_distinguished()

This function is not used.

Table 2.2.1.1-2: Function Summary -clear_distinguished()

This function is not used.

Table 2.2.1.1-3: Function Summary - view_ground_vehicles()

This function is not used.

Table 2.2.1.1-4: Function Summary - view_platoons()

This function is not used.

Table 2.2.1.1-5 Function Summary - view_companies()

This function is not used.

Table 2.2.1.1-6: Function Summary - view_battalions()

This function is not used.

Table 2.2.1.1-7: Function Summary - view_sorties()

This function is not used.

Table 2.2.1.1-8: Function Summary - view_flights()

This function is not used.

Table 2.2.1.1-9 Function Summary - query_view_mode(veh_index)

This function is not used.

**Table 2.2.1.1-10 Function Summary - Update_Graphics_Display
(world_state, current_time)**

Parameters		
Parameter	Type	Where Typedef Declared
world_state	pointer to struct WORLD_STATE	world_state.h
current_time	int	standard

Calls	
Function	Where Described
heading	Measures.c
rel_turret	Measures.c
heading_fraction	Measures.c
get_turret_degrees	Text.c
mgrsi	Concurrent graphics library subroutine
setup_back_fb	New_zoom.c
mggetfb	Concurrent graphics library subroutine
mgibbgetview	Concurrent graphics library subroutine
mmgibbIt2	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgifbfbex	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
print_vehicle_id	Pvd.misc.c

2.2.1.2 icon_fea.c

This CSU draws the roaming unit symbols. (It is not used in the training sites, and not supported in training software.)

2.2.1.3 dir_fire.c

This CSU contains routines for processing impact PDUs and updating the event window with vehicle and ground impact information.

Table 2.2.1.3-1: Function Summary - do_impact(impact)

Parameters		
Parameter	Type	Where Typedef Declared
impact	pointer to ImpactVariant	p_sim.h

Calls	
Function	Where Described
find_icon	Icon.c
print_vehicle_id	Pvd.misc.c
save_impact_info	Dir_fire.c
show_firing_line	Dir_fire.c
set_text_window	indir_fire.c

Table 2.2.1.3-2: Function Summary - save_impact_info(attacker_id, impact_x, impact_y, attack_x, attack_y)

Parameters		
Parameter	Type	Where Typedef Declared
attacker_id	int	standard
impact_x, impact_y	float	standard
attack_x, attack_y	float	standard

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	failed

Table 2.2.1.3-3: Function Summary - show_firing_line(tank, impact_x, impact_y, attack_x, attack_y)

Parameters		
Parameter	Type	Where Typedef Declared
tank	ICONPTR	icon_dfn.h
impact_x, impact_y	float	standard
attack_x, attack_y	float	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.1.3-4: Function Summary - check_firing_duration

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.1.3-5: Function Summary - decode_ammo_type(burst)

Parameters		
Parameter	Type	Where Typedef Declared
burst	pointer to BurstDescriptor	basic.h

Calls	
Function	Where Described
decode specific ammo type	this file
decode generic ammo type	this file

Return Values		
Return Value	Type	Meaning
"ammo"	char	pointer to string

Table 2.2.1.3-6: Function Summary -
decode_generic_ammo_type(object_type)

Parameters		
Parameter	Type	Where Typedef Declared
object_type	ObjectType	p_sim.h

Table 2.2.1.3-7: Function Summary -
decode_specific_ammo_type(object_type)

Parameters		
Parameter	Type	Where Typedef Declared
object_type	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
M791 25mm AP	char	object type
M792 25mm	char	object type
M789 30mm HEDP	char	object type
M392A2 105mm APDS	char	object type
M456A1 105mm HEAT	char	object type
M329 107mm HE	char	object type
M107155mm HE	char	object type
M855 5.56mm bullet	char	object type
Mk82 500lb bomb	char	object type
TOW missile	char	missile
M47 Dragon missile	char	missile
Hellfire missile	char	missile
Maverick missile	char	missile
Sidewinder missile	char	missile
ADATS missile	char	missile
Stinger missile	char	missile
Tomahawk missile	char	missile
Styx C missile	char	Soviet missile
S-5 57mm rocket	char	Soviet missile

Table 2.2.1.3-8: Function Summary - decode_result(result)

Parameters		
Parameter	Type	Where Typedef Declared
result	int	standard

Return Values		
Return Value	Type	Meaning
nothing	char	nonImpact
ground	char	groundImpact
vehicle	char	vehicleImpact
illegal-target	char	default

2.2.1.4 indir_fire.c

This CSU contains routines for processing indirect fire PDUs and displaying various types of fire burst icons.

Table 2.2.1.4-1: Function Summary - init_bursts()

This function contains arrays to hold segments for burst icons.

Calls	
Function	Where Described
create_all_burst_segs	this file

Table 2.2.1.4-2: Function Summary - do_direct_fire (indir_fire_pkt)

Parameters		
Parameter	Type	Where Typedef Declared
indir_fire_pkt	pointer to IndirectFireVariant	p_sim.h

Calls	
Function	Where Described
report_ifire_to_screen	this file
do_firefinder	this file
draw_bursts	this file

Table 2.2.1.4-3: Function Summary - report_ifire_to_screen(i_pkt)

Parameters		
Parameter	Type	Where Typedef Declared
i_pkt	pointer to IndirectFireVariant	p_sim.h

Calls	
Function	Where Described
set text window	this file

Table 2.2.1.4-4: Function Summary - pr_ifire_map_coords(ipkt)

Parameters		
Parameter	Type	Where Typedef Declared
i_pkt	pointer to IndirectFireVariant	p_sim.h

Calls	
Function	Where Described
set text window	this file
status_stamp	
tdb_giv_xy_get_utm	see libtdb in MCC CSCI

Table 2.2.1.4-5: Function Summary - draw_bursts
(type_burst, i_pkt, color)

This function draws the string of bursts as specified by i_pkt and type_burst.

Parameters		
Parameter	Type	Where Typedef Declared
type_burst	pointer to long	standard
i_pkt	pointer to IndirectFireVariant	p_sim.h
color	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
set text window	this file
print_vehicle_id	Pvd.misc.c
status_stamp	
mgrxy	Concurrent graphics library subroutine
mqisegimm	Concurrent graphics library subroutine
mqisyncrb	Concurrent graphics library subroutine

Table 2.2.1.4-6: Function Summary - create_all_burst_segs(burst_scale)

Parameters		
Parameter	Type	Where Typedef Declared
burst_scale	real	standard

Calls	
Function	Where Described
create burst segment	this file

Table 2.2.1.4-7: Function Summary - create_burst_segment
(burst_array, scale_factor, burst_scale)

Parameters		
Parameter	Type	Where Typedef Declared
burst_array	pointer to long	standard
scale_factor	real	standard
burst_scale	real	standard

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mqieseg	Concurrent graphics library subroutine

Table 2.2.1.4-8: Function Summary - save_indir_fire_info(i_pkt)

This function stores the PDU information in the BurstDesc array.

Parameters		
Parameter	Type	Where Typedef Declared
i_pkt	pointer to IndirectFireVariant	p_sim.h

Return Values		
Return Value	Type	Meaning
YES	int	successful
NO	int	failed

Table 2.2.1.4-9: Function Summary - check_burst_duration()

Calls	
Function	Where Described
draw bursts	this file

Table 2.2.1.4-10: Function Summary - erase all bursts()

Calls	
Function	Where Described
draw_bursts	this file

Table 2.2.1.4-11: Function Summary - check_det(det)

Parameters		
Parameter	Type	Where Typedef Declared
det	pointer to IndirectFireDetonation	p_sim.h

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	failed

Table 2.2.1.4-12: Function Summary - set_firefinder()

This function sets the mode that turns on the firefinder feature.

2.2.1.5 class.c

This file contains routines for classifying objects into one of the display types that PVD can draw (such as vehicle, missile, building, or unknown object.)

Table 2.2.1.5-1: Function Summary - classify_object(object_type)

Given a Simnet object type, this routine returns a PVD draw type.

Parameters		
Parameter	Type	Where Typedef Declared
object_type	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
classify_vehicle	int	objectDomainVehicle
ICON_MISSILE	int	objectDomainMunition
ICON_BUILDING	int	objectDomainStructure
ICON_STEALTH	int	default
ICON_UNKNOWN	int	default

Calls	
Function	Where Described
classify_vehicle	this file

Table 2.2.1.5-2: Function Summary - classify_vehicle(object_type)

This function prints information from the LaserRangeVariant in the event window.

Parameters		
Parameter	Type	Where Typedef Declared
object_type	unsigned long	standard

Return Values		
Return Value	Type	Meaning
classify_ground_vehicle	int	vehicleEnvironmentGround
classify_air_vehicle	int	vehicleEnvironmentAir
classify_water_vehicle	int	vehicleEnvironmentWater
ICON_GROUND_VEHICLE	int	default

Calls	
Function	Where Described
classify_ground_vehicle	Class.c
classify_air_vehicle	Class.c
classify_water_vehicle	Class.c

Table 2.2.1.5-3: Function Summary - classify_ground_vehicle (object_type)

Parameters		
Parameter	Type	Where Typedef Declared
object_type	unsigned long	standard

Return Values		
Return Value	Type	Meaning
ICON_TANK	int	vehicleFunctionLightTank
ICON_IFV	int	vehicleFunctionPersonnel Carrier
ICON_ADA	int	vehicleFunctionAntiAircraft
ICON_GROUND_VEHICLE	int	default

Calls	
Function	Where Described
get_bfit_mode	

Table 2.2.1.5-4: Function Summary - classify_air_vehicle(object_type)

Parameters		
Parameter	Type	Where Typedef Declared
object_type	unsigned long	standard

Return Values		
Return Value	Type	Meaning
ICON FIXED WING	int	vehicleClassFixedWing
ICON ROTARY WING	int	vehicleClassRotaryWing
ICON FIXED WING	int	default

Table 2.2.1.5-5: Function Summary - classify_water_vehicle(object type)

Parameters		
Parameter	Type	Where Typedef Declared
object_type	unsigned long	standard

Return Values		
Return Value	Type	Meaning
ICON SMALL SHIP	int	vehicleFunctionCruiser
ICON MEDIUM SHIP	int	vehicleFunctionBattleship
ICON LARGE SHIP	int	vehicleFunctionCarrier
ICON SMALL SHIP	int	default

Table 2.2.1.5-6: Function Summary - knots_object(object)

This function returns 1 if this is a vehicle whose velocity should be reported in knots (ships and aircraft.)

Parameters		
Parameter	Type	Where Typedef Declared
object	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
0	int	other
1	int	This is a vehicle whose velocity should be reported in knots (aircraft and ships.)

2.2.1.6 lase.c

This file handles the receipt of a LaserRange PDU. It prints information from LaserRangeVariant in the event window.

Table 2.2.1.6-1: Function Summary - lase(l_variant)

Parameters		
Parameter	Type	Where Typedef Declared
l_variant	pointer to LaserRangeVariant	p_data.h

Calls	
Function	Where Described
display_lase	this file
draw_laser	this file

Table 2.2.1.6-2: Function Summary - display_lase(l_variant)

This function prints information from LaserRangeVariant in the event window.

Parameters		
Parameter	Type	Where Typedef Declared
l_variant	pointer to LaserRangeVariant	p_data.h

Calls	
Function	Where Described
tdb_giv_xy_get_utm	see libtdb in MCC CSCI
dist_3	Measures.c
set_text_window	indir_fire.c

Table 2.2.1.6-3: Function Summary - draw_laser(laser_packet)

Parameters		
Parameter	Type	Where Typedef Declared
laser_packet	pointer to LaserRangeVariant	p_data.h

Calls	
Function	Where Described
save_laser_info	this file
show_laser_beam	this file

Table 2.2.1.6-4: Function Summary - save_laser_info(laser_x, laser_y, muzzle_x, muzzle_y)

Parameters		
Parameter	Type	Where Typedef Declared
laser_x, laser_y	float	standard
muzzle_x, muzzle_y	float	standard

Return Values		
Return Value	Type	Meaning
0	int	failed
1	int	successful

Table 2.2.1.6-5: Function Summary - show_laser_beam(laser_x, laser_y, muzzle_x, muzzle_y)

Parameters		
Parameter	Type	Where Typedef Declared
laser_x, laser_y	float	standard
muzzle_x, muzzle_y	float	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.1.6-6: Function Summary - check_lasing_duration()

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

2.2.1.7 effects.c

This CSU provides a mechanism to interrupt icon processing to respond to user commands in a timely fashion. Update_Graphics_Display calls move_icon for each vehicle icon on the current map. This file also supports icon updating.

Table 2.2.1.7-1: Function Summary - init_effects()

Calls	
Function	Where Described
create_puff_seg	this file

Table 2.2.1.7-2: Function Summary - create_puff_seg()

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine

2.2.2 Icon Detail CSC Description

This CSC provides icon information in a separate display.

This CSC functionality is broken down into the CSUs shown in Figure 2.2.2-1.

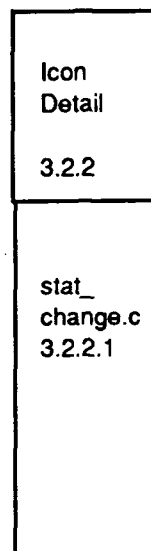


Figure 2.2.2-1: Icons--Icon Detail CSC Structure.

2.2.2.1 stat_change.c

This file contains routines for handling status change packets for vehicle icons. It prints messages in the event window indicating the arrival of a status change packet, containing information about type of vehicle, vehicle id, status (such as damaged, destroyed, or repaired), and damage cause.

**Table 2.2.2.1-1: Function Summary - display_status_change
(status_change)**

This function displays a message in the event window indicating the arrival of a status change packet.

Parameters		
Parameter	Type	Where Typedef Declared
status_change	pointer to StatusChangeVariant	p_data.h

Calls	
Function	Where Described
decode damage cause	this file
decode repair cause	this file
set text window	indir fire.c
print vehicle id	Pvd_misc.c

Table 2.2.2.1-2: Function Summary - decode_damage_cause(damage)

Parameters		
Parameter	Type	Where Typedef Declared
damage	int	standard

Return Values		
Return Value	Type	Meaning
cause	char	Prints cause: unspecified cause; random failure; collision with vehicle; crew error; direct fire; indirect fire; operator intervention; collision with terrain; unknown causes.

Table 2.2.2.1-3: Function Summary - decode_repair_cause(repair)

Parameters		
Parameter	Type	Where Typedef Declared
repair	int	standard

Return Values		
Return Value	Type	Meaning
cause	char	Prints cause: unspecified causes; crew action; operator intervention; simulated maintenance team; unknown causes.

2.2.3 Initial Icons CSC Description

This CSC is the main entry for Icons.

This CSC functionality is broken down into the CSUs shown in Figure 2.2.3-1.

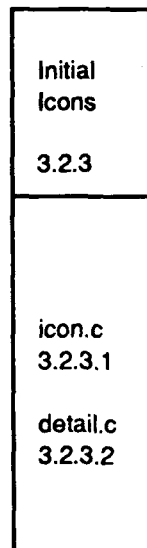


Figure 2.2.3-1: Icons--Initial Icons CSC Structure.

2.2.3.1 icon.c

This CSU includes the functions for creating, scaling, drawing and erasing all icons that appear on the map (representing vehicles, artillery impacts, etc.). The icon shapes are defined in icon_dfn.h.

Table 2.2.3.1-1: Function Summary - find_icon(vehicle_id)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to VehicleID	basic.h
Return Values		
Return Value	Type	Meaning
null	ICONPTR	no icon for given vehicle_id
tanks[vehicle_index]	ICONPTR	array of icon pointers
Calls		
Function	Where Described	
get_vehicle_index	Hash fn.c	

Table 2.2.3.1-2: Function Summary -get_icon_from_bumper(bumper_str)

Given the bumper number of an icon, this function returns the corresponding index.

Parameters		
Parameter	Type	Where Typedef Declared
bumper_str	pointer to char	standard

Table 2.2.3.1-3: Function Summary - create_icon(vehicle_index)

This function stores the vehicle id as well as the bumper number. If there is no bumper number, a string of the form "ID: vehicle id" will be stored.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_index	int	standard

Table 2.2.3.1-4: Function Summary - erase_old_tank(tank)

This function erases an old tank from the screen. It assumes in ABSOLUTE mode upon entering and leaving the routine.

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgrxy	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
erase_old_feature_icon	
display_attachment	this file
mgisegimm	Concurrent graphics library subroutine

Table 2.2.3.1-5: Function Summary - create_smoke_seg()

This is the segment for displaying the smoke vehicle. There will be only one segment, regardless of effect bits and size. This function performs scaling based on icon scale.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-6: Function Summary - create_minefield_flag()

This is the segment for displaying minefield flags. Breech lane flags are white, minefield flags are red.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-7: Function Summary - create_hl_seg()

This function creates a highlighting box such that the icon can fit inside it regardless of the icon's orientation.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgil1	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-8: Function Summary - create_bore_seg()

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgils	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-9: Function Summary - create_plane_segs()

A10 icon is created assuming in relative mode. Thus, the center of the icon is 0,0. Everything is calculated relative to the center. Dimensions used are: wingspan 60 feet, length 50 feet.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-10: Function Summary - create_hull_segs()

Hull is created assuming relative mode. Thus, the center of the hull is 0,0. Everything is calculated relative to the center. Hull is a 5 x 9 rectangle.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-11: Function Summary - create_turret_segs()

This function creates a segment for the turret. Cannon are drawn relative to the turret body. It assumes in relative mode upon entering the segment.

Calls	
Function	Where Described
mgipoly	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-12: Function Summary - create_m2_hull_segs()

Hull is created assuming relative mode. Thus, the center of the hull is 0,0. Everything is calculated relative to the center. Hull is a 4 x 7 rectangle.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-13: Function Summary - create_m2_turret_segs()

This function creates a segment for the turret. Cannon are drawn relative to turret body. It assumes in relative mode upon entering the segment.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine

Table 2.2.3.1-14: Function Summary - create_static_stripe_segs()

This function creates a segment for the stripe on MCC vehicles. The stripe is drawn relative to the hull body. It assumes in relative mode upon entering the segment.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-15: Function Summary - create_faad_turret_segs()

This function creates a segment for the turret. Cannon are drawn relative to the turret body. It assumes in relative mode upon entering the segment.

Calls	
Function	Where Described
mgibsegs	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-16: Function Summary - create_rwa_body_segs()

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-17: Function Summary - create_rwa_prop_segs()

This function creates a segment for the rwa prop and tail. It is drawn relative to the rwa body. It assumes in relative mode upon entering the segment.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgic	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-18: Function Summary - create_rwa_hl_segs()

This function calculates segments for the rwa highlight box. Since the RWA icon is asymmetric, its highlight box must be rotated based on its orientation.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgils	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-19: Function Summary - create_stealth_segs()

The Stealth icon will be an arrow pointing in the direction of travel. When it is attached, a circle will be displayed when the Stealth is attached to a specific vehicle.

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgewidth	Concurrent graphics library subroutine
mgill	Concurrent graphics library subroutine
mgixy	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-20: Function Summary - create_bfit_segs(model_type)

Hull is calculated assuming relative mode. Thus, the center of the hull is 0,0. Everything is calculated relative to the center. Hull is a 20 x 200 rectangle.

Parameters		
Parameter	Type	Where Typedef Declared
model_type	int	standard

Calls	
Function	Where Described
mgibseg	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgip	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-21: Function Summary - create_bfit_stripe_segs(model_type)

This function creates a segment for the stripe on vehicles. Stripe is drawn relative to the hull body. It assumes in relative mode upon entering the segment.

Parameters		
Parameter	Type	Where Typedef Declared
model_type	int	standard

Calls	
Function	Where Described
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-22: Function Summary - create_bfit_hl_segs(model_type)

This function calculates segments for the asymmetric highlight box. Since the icons are asymmetric, its highlight box must be rotated based on its orientation.

Parameters		
Parameter	Type	Where Typedef Declared
model_type	int	standard

Calls	
Function	Where Described
mgils	Concurrent graphics library subroutine
mgieseg	Concurrent graphics library subroutine

Table 2.2.3.1-23: Function Summary - display_attachment(hue)

The highlight for stealth attach is a circle of radius 9. This is just big enough to fit around the highlight box. It assumes current position is the center of the vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
hue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgewidth	Concurrent graphics library subroutine
mgitc	Concurrent graphics library subroutine

Table 2.2.3.1-24: Function Summary - create_attachment_icon(veh_ind)

Parameters		
Parameter	Type	Where Typedef Declared
veh_ind	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrrelxy	Concurrent graphics library subroutine

Table 2.2.3.1-25: Function Summary - delete_attachment_icon(veh_ind)

Parameters		
Parameter	Type	Where Typedef Declared
veh_ind	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrrelxy	Concurrent graphics library subroutine

Table 2.2.3.1-26: Function Summary - reinit_all_icons()

Calls	
Function	Where Described
erase_all	indir.fire.c
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.3.1-27: Function Summary - delete_icon(tank_ind)

Parameters		
Parameter	Type	Where Typedef Declared
tank_ind	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
erase_old tank	this file
erase whole trail	this file
highlight icon	this file
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.3.1-28: Function Summary - move_icon(tank_ind, type, appearance, force, hull_index, turret_index, x, y)

This function supports a vehicle changing its "alignment " (force) and appearance (guise) without deactivating. The change of guise can result from the PVD changing its view from the "distinguished" force.

Parameters		
Parameter	Type	Where Typedef Declared
tank_ind	int	standard
type	ObjectType	p_sim.h
appearance	unsigned long	standard
force	ForceID	basic.h
hull_index	int	standard
turret_index	int	standard
x, y	double	standard

Calls	
Function	Where Described
query view mode	Graph.c
move feature icon	
classify object	Class.c
mgiv	Concurrent graphics library subroutine
mgipin	Concurrent graphics library subroutine
highlight icon	this file
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.3.1-29: Function Summary - check_vehicle_type(tank)

This function checks the vehicle type to see if it has changed from a dead tank to an active one or vice versa. If there is a change, the function changes the color appropriately.

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h

Return Values		
Return Value	Type	Meaning
0	int	change
1	int	no change

Calls	
Function	Where Described
swap colors	this file

Table 2.2.3.1-30: Function Summary - swap_colors(tank)

This function swaps the hull and turret colors when the vehicle type changes. It also toggles the vehicle status.

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h

Table 2.2.3.1-31: Function Summary - assign_colors(tank, force)

This function initializes the colors of the hull and turret and initializes the status.

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h
force	ForceID	basic.h

Table 2.2.3.1-32: Function Summary - highlight_icon(icon, hue)

This function highlights a particular icon by drawing it in the color, hue. The highlighting box can be removed by using hue = 0.

Parameters		
Parameter	Type	Where Typedef Declared
icon	ICONPTR	icon_dfn.h
hue	long	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgrrelxy	Concurrent graphics library subroutine
highlight feature icon	
mgisyncrb	Concurrent graphics library subroutine
mgisegimm	Concurrent graphics library subroutine

Table 2.2.3.1-33: Function Summary -highlight_ selected_icon (vehicle_index)

This function is called by the user interface to let the icon routines know that the icon has been highlighted.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle index	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
icon height	
mgisegimm	Concurrent graphics library subroutine

**Table 2.2.3.1-34: Function Summary - unhighlight_selected_icon
(vehicle_index)**

This function is called by the user interface to let the icon routines know that highlighting has been turned off for the icon. This has the possibility of putting a streak through an icon if there is one nearby. Therefore, to minimize the possibility of this happening, do not erase highlighting box unless there definitely is one.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle index	int	standard

Return Values		
Return Value	Type	Meaning
yes	int	highlighting turned off

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
highlight icon	this file

Table 2.2.3.1-35: Function Summary - draw_tank(hull_orient, turret orient, tank)

Parameters		
Parameter	Type	Where Typedef Declared
hull_orient	int	standard
turret_orient	int	standard
tank	register ICONPTR	icon_dfn.h

Calls	
Function	Where Described
display_attachment	this file
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgisegimm	Concurrent graphics library subroutine

Table 2.2.3.1-36: Function Summary - init_icon_colors()

Calls	
Function	Where Described
make_color_array	
make_color_map	Color_map.c

Table 2.2.3.1-37: Function Summary - init_icons()

Calls	
Function	Where Described
calc_initial_icon_size	this file
calc_screen_scales	this file
create_rwa_hl_segs	this file
create_hull_segs	this file
create_turret_segs	this file
create_bore_seg	this file
create_plane_segs	this file
create_smoke_seg	this file
create_minefield_flag	this file
create_faad_turret_segs	this file
create_m2_hull_segs	this file
create_m2_turret_segs	this file
create_static_stripe_segs	this file
create_rwa_body_segs	this file
create_rwa_prop_segs	this file
create_stealth_segs	this file
create_bfit_segs	this file
create_bfit_stripe_segs	this file
create_bfit_hl_segs	this file
create_feature_icons	
show_icon_scale	Pvd_windows.c

Table 2.2.3.1-38: Function Summary - set_distinguished_colors()

This function is not used.

Table 2.2.3.1-39: Function Summary - set_other_colors()

This function is not used.

Table 2.2.3.1-40: Function Summary - make_color_array(color_array)

Parameters		
Parameter	Type	Where Typedef Declared
color_array	pointer to long	standard

Table 2.2.3.1-41: Function Summary - draw_trail(tank, force, x, y)

The trail for each tank is composed of an array of points. The arrays, x_trail and y_trail are arrays of floats which are the x and y coordinates for the trail of each tank. The indices of x_trail and y_trail indicate which tank's trail is being accessed. Each trail length is the number of points specified by NUM_POINTS in "icons_dfn.h."

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h
force	ForceID	basic.h
x, y	float	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
assign_colors	this file
init_trail	this file
mgihue	Concurrent graphics library subroutine
mgrp	Concurrent graphics library subroutine

Table 2.2.3.1-42: Function Summary - erase_trail(tank)

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgrp	Concurrent graphics library subroutine

Table 2.2.3.1-43: Function Summary - init_trail(tank, x_pos, y_pos)

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon_dfn.h
x_pos, y_pos	register float	standard

Table 2.2.3.1-44: Function Summary - zoom_icon()

Zoom_icon is called when the user is zooming in on a small section of terrain. The screen is rescaled and the icon segments must be recalculated to reflect the scaling. Each time zoom_icon is called, a new scale factor must be determined. A ratio of the starting viewport coordinates to the screen coordinates is stored. The default scale factor is set to DEFAULT_SCALE. The ratio of the Y_coordinates = ratio of the X_coordinates. Therefore, the ratio of the y_coordinates/DEFAULT_SCALE yields a value called Scale_ratio. The new scale ratio must always be $\leq$ its initial value in order for the icons and their orientation to be recognizable. Using the initial scale_ratio to determine the new scale factor for the tank, one can tell if the tank can be drawn using its actual dimensions or if it must first be magnified. A new set of segments are then recalculated relative to the zoom window.

Calls	
Function	Where Described
calc screen scales	this file
calc scale factor	this file
create symbolic icons	this file
zoomed in	this file
create hl seg	this file
create rwa hl segs	this file
create hull segs	this file
create turret segs	this file
create plane segs	this file
create smoke seg	this file
create minefield flag	this file
create faad turret segs	this file
create m2 hull segs	this file
create m2 turret segs	this file
create static stripe segs	this file
create rwa body segs	this file
create rwa prop segs	this file
create stealth segs	this file
create bfit segs	this file
create bfit stripe segs	this file
create bfit hl segs	this file
create all burst segs	Indir fire.c
set menu icon scale	Menu.c
force redraw of icons	this file

Table 2.2.3.1-45: Function Summary - top_level_zoom_icon()

Calls	
Function	Where Described
calc screen scales	this file
calc scale factor	this file
create symbolic icons	this file
menu_symbolic	Menu func.c
calc initial icon size	this file
create hl seg	this file
create rwa hl segs	this file
create hull segs	this file
create turret segs	this file
create plane segs	this file
create smoke seg	this file
create minfield flag	this file
create faad turret segs	this file
create m2 hull segs	this file
create m2 turret segs	this file
create static stripe segs	this file
create rwa body segs	this file
create rwa prop segs	this file
create stealth segs	this file
create bfit segs	this file
create bfit stripe segs	this file
create bfit hl segs	this file
create all burst segs	Indir fire.c
set menu icon scale	Menu.c
force redraw of icons	this file

Table 2.2.3.1-46: Function Summary - pre_zoom_icon()

Calls	
Function	Where Described
erase all bursts	Indir fire.c
erase all	indir fire.c

Table 2.2.3.1-47: Function Summary - calc_scale_factor()

This function determines the scale factor for the icons in a xoom environment. If the scale factor is greater than 1, the actual dimensions of the tank are used (scale = 1.0.)

Return Values		
Return Value	Type	Meaning
scale_ratio	pointer to real	actual dimensions of the tank

Table 2.2.3.1-48: Function Summary - scale_icon(scale_factor)

When called by the menu, this routine will change the scale factor for the tank icons and recalculates icon_coords.

Parameters		
Parameter	Type	Where Typedef Declared
scale_factor	int	standard

Calls	
Function	Where Described
init working cursor	Pvd iface.c
erase all tanks	this file
erase all bursts	Indir fire.c
change icon scaling	this file
create hl seg	this file
create rwa hl segs	this file
create hull segs	this file
create turret segs	this file
create plane segs	this file
create smoke seg	this file
create minefield flag	this file
create faad turret segs	this file
create m2 hull segs	this file
create m2 turret segs	this file
create static stripe segs	this file
create rwa body segs	this file
create rwa prop segs	this file
create stealth segs	this file
create bfit segs	this file
create bfit stripe segs	this file
create bfit hl segs	this file
create all burst segs	Indir fire.c
force redraw of icons	this file
show icon scale	Pvd windows.c
init cursor	Pvd iface.c

Table 2.2.3.1-49: Function Summary - change_icon_scaling(scale_factor)

Parameters		
Parameter	Type	Where Typedef Declared
scale_factor	real	standard

Table 2.2.3.1-50: Function Summary - calc_screen_scales()

For both x and y coordinates, the algorithm is the same. The ratio of x_real to size of viewport in x direction = ratio of x_integer to size of viewport in x direction in pixels. Similar for y. Thus, the scale factor being calculated is the dimensions of the integer window / dimensions of the real window since the mathematical calculations are using the real coordinates.

Calls	
Function	Where Described
mgrgetvcoor	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine

Table 2.2.3.1-51: Function Summary - erase_all_tanks()

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
erase old tank	this file
highlight icon	this file
mgisyncrb	Concurrent graphics library subroutine

Table 2.2.3.1-52: Function Summary - erase_all_trails()

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.2.3.1-53: Function Summary - erase_whole_trail(tank)

Parameters		
Parameter	Type	Where Typedef Declared
tank	ICONPTR	icon_dfn.h

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrp	Concurrent graphics library subroutine

Table 2.2.3.1-54: Function Summary - force_redraw_of_icons()

This routine causes all tanks to be redrawn on the screen. It is used when a zoom or scale change has occurred. All tanks are redrawn immediately.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgrrelxy	Concurrent graphics library subroutine
draw tank	this file
highlight icon	this file
mgisyncrb	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.2.3.1-55: Function Summary - redraw_trail(tank)

Parameters		
Parameter	Type	Where Typedef Declared
tank	register ICONPTR	icon dfn.h

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrp	Concurrent graphics library subroutine

Table 2.2.3.1-56: Function Summary - get_icon_scale()

Return Values		
Return Value	Type	Meaning
-1	int	symbolic scale mode
scale	int	normal scale mode

Calls	
Function	Where Described
check for target hit	

Table 2.2.3.1-57: Function Summary - calc_initial_icon_size()

Return Values		
Return Value	Type	Meaning
FALSE	int	old_max_scale==max_scale
TRUE	int	other

Calls	
Function	Where Described
mgrgetvcoor	Concurrent graphics library subroutine
menu 50x	Menu func.c
menu 100x	Menu func.c

Table 2.2.3.1-58: Function Summary - zoomed_in()

Return Values		
Return Value	Type	Meaning
FALSE	int	old_max_scale==max_scale
TRUE	int	other

Table 2.2.3.1-59: Function Summary - set_symbolic_mode(state)

Parameters		
Parameter	Type	Where Typedef Declared
state	int	standard

Calls	
Function	Where Described
create_symbolic_icons	this file

Table 2.2.3.1-60: Function Summary - create_symbolic_icons()

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
erase_all_tanks	this file
erase_all_bursts	Indir_fire.c
change_icon_scaling	this file
create_hl_seg	this file
create_rwa_hl_segs	this file
create_hull_segs	this file
create_turret_segs	this file
create_plane_segs	this file
create_smoke_seg	this file
create_minefield_flag	this file

(Table 2.2.3.1-60 is continued on the following page.)

Function	Where Described
create faad turret segs	this file
create m2 hull segs	this file
create m2 turret segs	this file
create static stripe segs	this file
create rwa body segs	this file
create rwa prop segs	this file
create stealth segs	this file
create bfit segs	this file
create bfit stripe segs	this file
create bfit hl segs	this file
create all burst segs	Indir fire.c
force redraw of icons	this file
show icon scale	Pvd windows.c
init_cursor	Pvd iface.c

Table 2.2.3.1-61: Function Summary - get_true_icon_scale()

Return Values		
Return Value	Type	Meaning
symbolic_scale	int	symbolic scale mode
scale	int	normal scale mode

Table 2.2.3.1-62: Function Summary - icon_get_total_scale
(scale_x, scale_y)

Parameters		
Parameter	Type	Where Typedef Declared
scale x, scale y	pointer to REAL	standard

Table 2.2.3.1-63: Function Summary - draw_marker(pkt)

Parameters		
Parameter	Type	Where Typedef Declared
pkt	pointer to MarkerVariant	p_sim.h

Calls	
Function	Where Described
mgipIn	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgrbox	Concurrent graphics library subroutine

Table 2.2.3.1-64: Function Summary - draw_lane(pkt)

This function shows the outline of the breached lane.

Parameters		
Parameter	Type	Where Typedef Declared
pkt	pointer to BreachedLaneVariant	p_sim.h

Calls	
Function	Where Described
mgipln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrls	Concurrent graphics library subroutine

Table 2.2.3.1-65: Function Summary - draw_minefield(pkt)

This function shows the outline of the minefield.

Parameters		
Parameter	Type	Where Typedef Declared
pkt	pointer to MinefieldVariant	p_data.h

Calls	
Function	Where Described
mgipln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgrls	Concurrent graphics library subroutine

2.2.3.2 detail.c

This CSU includes functions for describing the nearest tank and displays text in a window. Information may be included on tank identifiers (bumper #, vehicle id #); general description (tank, IFV, attack vehicle); status (operational, crippled, destroyed); appearance (on fire, smoke plume, engine smoke); team (A, B); country (US, Soviet, unknown); failures (of engine, drive train, communications and electronics); and capabilities (ammo supplier, fuel supplier, recovery or repair supplier.)

Table 2.2.3.2-1: Function Summary - detail_display(world_state, vehicle_index)

This function provides a text description of the nearest tank. Text is displayed in a window.

Parameters		
Parameter	Type	Where Typedef Declared
world_state	pointer to WORLD_STATE struct	world_state.h
vehicle_index	pointer to int	standard

Calls	
Function	Where Described
get_elevation	Get_elev.c
get_velocity	Text.c
get_object_from_guise	this file
decode_appearance	this file
decode_team	this file
decode_capabilities	this file
kph	Text.c
map_tank_index	
decode_guises	this file
word_print	this file
tdb_giv_xy_get_utm	see libtdb in MCC CSCI
knots_object	Lase.c

Table 2.2.3.2-2: Function Summary - word_print(string, window_width)

This function prints out a string to fit a narrow window, assuming only one white space character between words.

Parameters		
Parameter	Type	Where Typedef Declared
string	pointer to char	standard
window_width	int	standard

Table 2.2.3.2-3: Function Summary - get_object_from_guise(guise)

Parameters		
Parameter	Type	Where Typedef Declared
guise	pointer to VehicleGuises	basic.h

Return Values		
Return Value	Type	Meaning
guise ->distinguished	ObjectType	value of Distinguished
guise ->other	ObjectType	other

Table 2.2.3.2-4: Function Summary - decode_guises(guise, model, country, description)

Parameters		
Parameter	Type	Where Typedef Declared
guise	pointer to VehicleGuises	basic.h
model	pointer to char	standard
country	pointer to char	standard
description	pointer to char	standard

Calls	
Function	Where Described
get_vehicle_model	this file
get_vehicle_country	this file
get_vehicle_description	this file

Table 2.2.3.2-5: Function Summary - get_vehicle_model(vehicle)

This function returns a pointer to a static string representing the "model" of the passed vehicle object.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
A-10	char	vehicle US
AH-64	char	vehicle US
OH-58D	char	vehicle US
F-14D	char	vehicle US
F-15E	char	vehicle US
F-20	char	vehicle US
F-4S	char	vehicle US
F-5F	char	vehicle US
M1	char	vehicle US
M2	char	vehicle US
M3	char	vehicle US
M113	char	vehicle US
M577	char	vehicle US
M106	char	vehicle US
M109	char	vehicle US
M88	char	vehicle US
M35	char	vehicle US
M977	char	vehicle US
M978	char	vehicle US
LHD-1	char	vehicle US
LHD-2	char	vehicle US
DD-963	char	vehicle US

(Table 2.2.3.2-5 is continued on the following page.)

Return Value	Type	Meaning
DD-964	char	vehicle US
DD-976	char	vehicle US
CG-47	char	vehicle US
CG-52	char	vehicle US
CVN-68	char	vehicle US
CVN-69	char	vehicle US
CVN-70	char	vehicle US
BB-71	char	vehicle US
BB-72	char	vehicle US
BB-73	char	vehicle US
BB-74	char	vehicle US
PHM-1	char	vehicle US
PHM-2	char	vehicle US
CG-26	char	vehicle US
CG-16	char	vehicle US
CG-17	char	vehicle US
CGN-38	char	vehicle US
CGN-36	char	vehicle US
CGN-37	char	vehicle US
CGN-35	char	vehicle US
CGN-25	char	vehicle US
CGN-9	char	vehicle US
CV-63	char	vehicle US
CV-64	char	vehicle US
CV-65	char	vehicle US
CV-59	char	vehicle US
CV-41	char	vehicle US
CV-42	char	vehicle US
CVN-65	char	vehicle US
DDG-52	char	vehicle US
DDG-993	char	vehicle US
DDG-994	char	vehicle US
DDG-2	char	vehicle US
FF-1052	char	vehicle US
FF-1040	char	vehicle US
SSBN-726	char	vehicle US
SSBN-727	char	vehicle US
SSBN-728	char	vehicle US
SSBN-640	char	vehicle US
SSN-637	char	vehicle US
SSN-585	char	vehicle US
SSN-590	char	vehicle US
SSN-688	char	vehicle US
SSN-689	char	vehicle US
SSN-753	char	vehicle US
SS-580	char	vehicle US
AD-41	char	vehicle US
AD-42	char	vehicle US

(Table 2.2.3.2-5 is continued on the following page.)

Return Value	Type	Meaning
AFS-1	char	vehicle US
Su-25	char	vehicle USSR
MiG23	char	vehicle USSR
MiG27	char	vehicle USSR
Mi-24	char	vehicle USSR
Mi-28	char	vehicle USSR
M-1943	char	vehicle USSR
T-72	char	vehicle USSR
BREM1	char	vehicle USSR
BMP-1	char	vehicle USSR
BMP-1K	char	vehicle USSR
2S1	char	vehicle USSR
BMP-2	char	vehicle USSR
ZSU-23-4M	char	vehicle USSR
GAZ-66	char	vehicle USSR
Ural-375F	char	vehicle USSR
Alpha	char	vehicle USSR
unknown	char	default

Calls	
Function	Where Described
get_bfit_mode	

Table 2.2.3.2-6: Function Summary - decode_appearance(appearance, object)

This function returns a pointer that is effective until the next call to this routine.

Parameters		
Parameter	Type	Where Typedef Declared
appearance	unsigned long	standard
object	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
out_string	char	value of MaxAppearance

Table 2.2.3.2-7: Function Summary - decode_team(force)

This function returns a static string indicating on which team the vehicle is playing.

Parameters		
Parameter	Type	Where Typedef Declared
force	int	standard

Table 2.2.3.2-8: Function Summary - decode_capabilities(abilities)

Parameters		
Parameter	Type	Where Typedef Declared
abilities	VehicleCapabilities	basic.h

Table 2.2.3.2-9: Function Summary - get_vehicle_country(vehicle)

This function returns a static string indicating the country of manufacture (or design) of the vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p_sim.h

Return Values		
Return Value	Type	Meaning
US	char	vehicleCountry
Soviet	char	vehcileCountry
unknown	char	default

**Table 2.2.3.2-10: Function Summary - get_vehicle_description
(vehicle, short_desc)**

This function returns a pointer to a static string that provides a general description of the type of vehicle (tank, IFV, APC, etc. It also modifies short_desc to point to as "short" no more than 5-character description.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p sim.h
short_desc	pointer to pointer to char	standard

Return Values		
Return Value	Type	Meaning
Munition	char	objectDomain
Structure	char	objectDomain
Life form	char	objectDomain
Unknown domain!	char	objectDomain
get_ground_vehicle_description	char	vehicleEnvironment
get air vehicle description	char	vehicleEnvironment
get water vehicle description	char	vehicleEnvironment
satellite	char	vehicleEnviornment
unknown object	char	default

Calls	
Function	Where Described
get ground vehicle description	this file
get air vehicle description	this file
get water vehicle description	this file

**Table 2.2.3.2-11: Function Summary - get_ground_vehicle_description
(vehicle, short_desc)**

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p_sim.h
short_desc	pointer to pointer to char	standard

Return Values		
Return Value	Type	Meaning
Tank	char	vehicleFunction
Light tank	char	vehicleFunction
Armored personnel carrier	char	vehicleFunction
Command post	char	vehicleFunction
Howitzer	char	vehicleFunction
Mortar	char	vehicleFunction
Rocket launcher	char	vehicleFunction
Recon vehicle	char	vehicleFunction
Recovery vehicle	char	vehicleFunction
Supply truck	char	vehicleFunction
Tank destroyer	char	vehicleFunction
Air defense vehicle	char	vehicleFunction
Ground vehicle	char	default

Calls	
Function	Where Described
get_bfit_mode	

**Table 2.2.3.2-12: Function Summary - get_air_vehicle_description
(vehicle, short_desc)**

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p_sim.h
short_desc	pointer to pointer to char	standard

Return Values		
Return Value	Type	Meaning
Fixed wing aircraft	char	vehicle Class
Helicopter	char	vehicleClass
Lighter than air craft	char	vehicleClass
Aircraft	char	default

**Table 2.2.3.2-13: Function Summary - get_water_vehicle_description
(vehicle, short_desc)**

Parameters		
Parameter	Type	Where Typedef Declared
vehicle	ObjectType	p_sim.h
short_desc	pointer to pointer to char	standard

Return Values		
Return Value	Type	Meaning
Ammunition supply ship	char	vehicleFunction
Amphibious assault ship	char	vehicleFunction
Amphibious cargo ship	char	vehicleFunction
Amphibious command ship	char	vehicleFunction
Amphibious transport ship	char	vehicleFunction
Attack submarine	char	vehicleFunction
Ballistic missile submarine	char	vehicleFunction
Battleship	char	vehicleFunction
Aircraft carrier	char	vehicleFunction
Combat stores ship	char	vehicleFunction
Cruiser	char	vehicleFunction
Destroyer tender	char	vehicleFunction
Destroyer	char	vehicleFunction
Dock landing ship	char	vehicleFunction
Fast combat support ship	char	vehicleFunction
Fleet oiler	char	vehicleFunction
Frigate	char	vehicleFunction
Hydrofoil	char	vehicleFunction
Mine countermeasures ship	char	vehicleFunction
Ocean minesweeper	char	vehicleFunction
Repair ship	char	vehicleFunction
Replenishment ship	char	vehicleFunction
Salvage ship	char	vehicleFunction
Ship	char	vehicleFunction
Submarine tender	char	vehicleFunction
Tank landing ship	char	vehicleFunction
ship	char	default

Calls	
Function	Where Described
decode damage cause	this file
decode repair cause	this file
set text window	indir_fire.c

2.3 MAP HANDLING CSC DESCRIPTION

This CSC supports a color-coded display of the terrain showing various geographic details, such as roads, rivers, and terrain relief. Shading of the geographic display is performed by an algorithm which assumes a sun elevation of 45 degrees from the horizon and a sun shining from the top right-hand corner of the display. This CSC also allows a user to vary map features of zoom, grid, and contour lines.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-3.

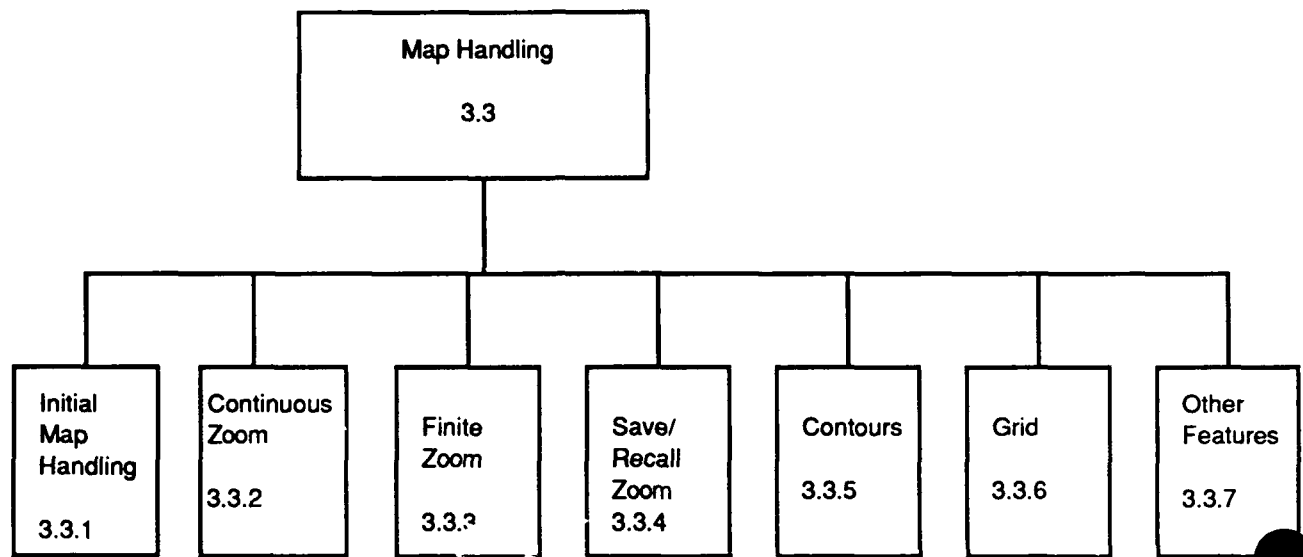


Figure 2-3: Map Handling CSC Structure.

2.3.1 Initial Map Handling CSC Description

This CSC is the main entry for Map Handling.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.1-1.

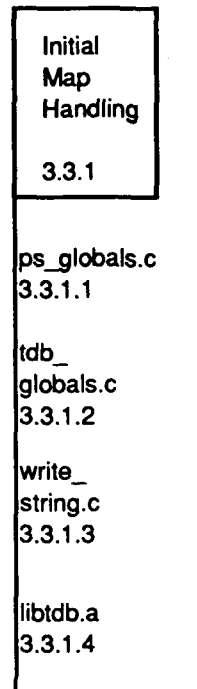


Figure 2.3.1-1: Map Handling--Initial Map Handling CSC Structure.

2.3.1.1 ps_globals.c CSU Description (/simnet/pvd/lib/paintlib.a)

This file contains globals for the paint system.

2.3.1.2 tdb_globals.c CSU Description (/simnet/pvd/lib/paintlib.a)

This file contains globals for the terrain database.

2.3.1.3 write_string.c CSU Description (/simnet/pvd/lib/paintlib.a)

This CSU writes a string to the screen using graphics fonts, on particular plan, using a particular color.

Table 2.3.1.3-1: - Function Summary - write string(window, hues, string, x, y, clear, font)

Parameters		
Parameter	Type	Where Typedef Declared
window	int	standard
planes	int	standard
hue	int	standard
string	pointer to char	standard
x, y	int	standard
clear	int	standard
font	int	standard
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgipIn	Concurrent graphics library subroutine	
mgigf	Concurrent graphics library subroutine	
mgiclearpIn	Concurrent graphics library subroutine	
mgihue	Concurrent graphics library subroutine	

2.3.1.4 libtdb.a

This CSU provides access to the terrain database.

2.3.2 Continuous Zoom CSC Description

This CSC provides the continuous zoom optional map feature. This feature supports arbitrary scale map drawing.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.1-1.

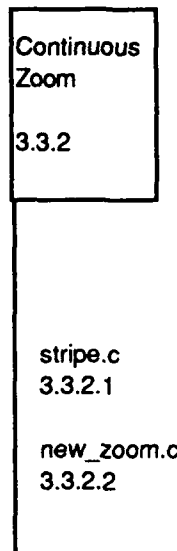


Figure 2.3.1-1: Map Handling--Continuous Zoom CSC Description.

2.3.2.1 stripe.c

This CSU draws a horizontal color stripe, starting at the given location on the current window's coordinates.

Table 2.3.2.1-1: Function Summary - display_stripe(start_x, start_y, box_edge, initial_color, color_increment, num_boxes)

Parameters		
Parameter	Type	Where Typedef Declared
start_x, start_y	int	standard
box_edge	int	standard
initial_color	int	standard
color_increment	int	standard
num_boxes	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

2.3.2.2 new_zoom.c

This CSU provides general purpose, arbitrary scale map drawing.

Note: Function names beginning with **mg** are Concurrent graphics library subroutines.

Table 2.3.2.2-1: new_zoom.c Information

This file defines or declares these global variables.

Global Variables	Type and Meaning	Typedef Source	Comments
Map	MAP_DATA		Declared by: init_paint_system; poly_zoom; unified_paint_square_region; pnav_paint_region; pnav_zoom_to_scale; paint_square_region; zoom_to_scale; pan; setup_back_fb; copy_map_back; zoom_out
Abort_Cmd			1 if ^B used to cancel command
patch_indices	INT 4*		read in by tdb_init_cache

External Static Variables	Type and Meaning	Typedef Source	Comments
dont_draw_overlays	int		if no overlays(roads & rivers) are to be drawn with the patches, because zoomed too far out
Stripe_Buffer	pointer to char		Get set to segment of cache memory fill with a stripe of patches from disk
Patch_Guards	pointer to type PATCH_GUARD		for intervisibility, etc
map_source	BB_DESC		For copy_map_back
map_destination	BB_DESC		For copy_map_back
no_map_on_paint	int		For set_map_on_paint() & set_no_map_on_paint; If 1 paint_square_region will NOT draw any element of the map (ground, roads & rivers, features)
tdb_info	TDB_INFO		declared by init_terrain_db; terrain description field of a database header.

Table 2.3.2.2-2: Function Summary - init_terrain_db(database_file, cached_patches, include_patch_guards)

This function initializes access to the terrain data base. Database_file contains the pathname to the database. Cached_patches contains the number of patches to cache. If the value of include_patch_guards is 1, it reads in patch guards for intervisibility.

Parameters		
Parameter	Type	Where Typedef Declared
database_file	pointer to char	standard
cached_patches	int	standard
include_patch_guards	int	standard
Return Values		
Return Value	Type	Meaning
0,	int	unable to initialize the database
1	int	function successful
Calls		
Function	Where Described	
tdb_init_cache	see libtdb in MCC CSCI	
tdb_error	see libtdb in MCC CSCI	
init_intervisibility		

Table 2.3.2.2-3: Function Summary - reinit_db(db_name, include_patch_guards)

This function switches to a new database. Db_name contains the name of the new database, such as "Graf" or "Knox."

Parameters		
Parameter	Type	Where Typedef Declared
db_name	pointer to char type	standard
include_patch_guards	int	standard
Return Values		
Return Value	Type	Meaning
0	int	Unable to change the database
1	int	Reopening old database

Calls	
Function	Where Described
make_path_name	init_env.c
get_data_directory	init_env.c
tdb_terminate	lib.tdb
init_terrain_db	this file
reset_views	Prev_view.c
init_intervisibility	
init_contour	init_contour.c
clear_stored_bitmaps	finite_zoom.c
init_discrete_zoom	finite_zoom.c
handle_update_display	handle_input.c

Table 2.3.2.2-4: Function Summary - init_paint_system (map_window, x0, y0, width, height, min_map_scale, pixels_per_inch, colors, finite_zoom_options, graphics_state)

This function places the map window and stores global variables describing it. Map_window is the window in which the map is to be drawn. x0 and y0 define the lower left corner of the map in raw pixel coordinates. Width and height determine the size of the map window. 1::min_map_scale defines the greatest zoom allowed. Pixels per inch is used for the graphics system. Colors contains various planes and colors. Finite_zoom_options determines what to draw when a new bitblt is created.

Parameters		
Parameter	Type	Where Typedef Declared
map_window	int	standard
x0, y0	int	standard
width, height	int	standard
min_map_scale	float	standard
pixels_per_inch	float	standard
colors	pointer to COLOR_SYSTEM	paint_system.h
finite_zoom_options	pointer to TERRAIN_FEATURES	paint_system.h
graphics_state	pointer to GRAPHICS	paint_system.h
Calls		
Function	Where Described	
mgigetiswdef	Concurrent graphics library subroutine	
mgidefw	Concurrent graphics library subroutine	
mgipw	Concurrent graphics library subroutine	
init_contour	init_contour.c	
init_canopy_pattern	this file	

Table 2.3.2.2-5: Function Summary - poly_zoom(x0, y0, x1, y1)

This function changes the current map zoom. It works by drawing the relevant patch of polygons. It uses bit zoom instead of painting, if appropriate. Input coordinates are in meters with (0,0) being lower left corner and (1,1) the upper right. Assumes input coordinates form a square, otherwise forces region to be square.

Parameters		
Parameter	Type	Where Typedef Declared
int x0, y0	int	standard
int x1, y1	int	standard
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
mgrsi	Concurrent graphics library subroutine	
force_square	this file	
zoom_to_scale	this file	
unified_paint_square_region	this file	

Table 2.3.2.2-6: Function Summary - unified_paint_square_region(x0_m, y0_m, x1_m, y1_m)

This function checks whether there is a shaded discrete zoom for this map scale. It rounds to the nearest integer in the scale and adjusts corner points.

Parameters		
Parameter	Type	Where Typedef Declared
float x0_m, y0_m	float	standard
float x1_m, y1_m	float	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	aborted
Calls		
Function	Where Described	
get_discrete_zoom_levels	init env.c	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
blt_paint_square_region	this file	
paint_square_region	this file	
shade_desired	Unshade.c	

Table 2.3.2.2-7: Function Summary - pnav_paint_region(x0_m, y0_m, x1_m, y1_m)

This function uses bit zoom instead of painting if appropriate. Input coordinates are in meters with (0,0) being the lower left corner. This function assumes input coordinates form a square.

Parameters		
Parameter	Type	Where Typedef Declared
float x0_m, y0_m	float	standard
float x1_m, y1_m	float	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	aborted
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
blt_paint_square	this file	
paint_square	this file	

Table 2.3.2.2-8: Function Summary - pnav_zoom_to_scale(scale, x, y)

This function paints a map in the given scale at the passed map center. The scale of the map will be 1::scale, with x,y being the center of the map in meters. The scale is rounded to the nearest integer.

Parameters		
Parameter	Type	Where Typedef Declared
scale	int	standard
x,y	float	standard
Calls		
Function	Where Described	
pnav_paint_region	this file	

Table 2.3.2.2-9: Function Summary - paint_square_region(x0_m, y0_m, x1_m, y1_m)

This function paints a square sub-region of the map. Input coordinates are in meters with (0,0) being the lower left corner. It assumes input coordinates form a square.

This function also converts indices into screen pixel coordinates, and converts new window meters into GROUND_map index, and scales the window appropriately.

This function computes patches involved. Data in patch file is stored left to right, bottom to top.

In addition, it checks whether shade is desired.

Parameters		
Parameter	Type	Where Typedef Declared
x0_m, y0_m	float	standard
x1_m, y1_m	float	standard
Return Values		
Return Value	Type	Meaning
0	int	aborted
1	int	successful
Function		Where Described
mgrrsi		Concurrent graphics library subroutine
setup_back_fb		FOR DEBUGGING ONLY
mgclearpln		Concurrent graphics library subroutine
mgrvcoor		Concurrent graphics library subroutine
mgihue		Concurrent graphics library subroutine
mgipln		Concurrent graphics library subroutine
mgrbox		Concurrent graphics library subroutine
mgisyncrb		Concurrent graphics library subroutine
tdb_cache_disable		see libtdb in MCC CSCI
copy_map_back		this file
copy_nonmap_back		this file
display_back_fb		this file
draw_stripe		this file
tdb_cache_enable		see libtdb in MCC CSCI
silent_shade_on		Unshade.c
silent_shade_off		Unshade.c
zoom_overlay		Overlayif.c
zoom_icon		Icon.c

Table 2.3.2.2-10: Function Summary - draw_polygons(first_poly, end_of_polys, vertex_list)

This function draws the polygons from first_poly through just before end_of_polys.

Parameters		
Parameter	Type	Where Typedef Declared
*first_poly	pointer to POLYGON_DESCRIPTOR	terrain.h
end_of_polys	pointer to POLYGON_DESCRIPTOR	terrain.h
vertex_list	pointer to TERRAIN_POINT	terrain.h
Calls		
Function	Where Described	
P_POLY_DUPLICATE	terrain.h	
GET_POLY_NUM_VERTS	terrain.h	
mgisyncrb	Concurrent graphics library subroutine	
mgrpoly	Concurrent graphics library subroutine	
mgihue	Concurrent graphics library subroutine	
GET_POLY_SHADING	terrain.h	

Table 2.3.2.2-11: Function Summary - draw_non_ground_polygons (first_poly, end_of_polys, vertex_list)

This function draws only those polygons that do not have one of the ground hues (e.g. roads and rivers).

Parameters		
Parameter	Type	Where Typedef Declared
first_poly	pointer to POLYGON_DESCRIPTOR	terrain.h
end_of_poly	pointer to POLYGON_DESCRIPTOR	terrain.h
vertex_list	pointer to TERRAIN_POINT	terrain.h
Calls		
Function	Where Described	
mgisyncrb	Concurrent graphics library subroutine	
GET_POLY_SHADING	terrain.h	
mgihue	Concurrent graphics library subroutine	
mgrpoly	Concurrent graphics library subroutine	
P_POLY_DUPLICATE	terrain.h	
MAX_FEATURE_VALUE	paint_sys.h	
GET_POLY_NUM_VERTS	terrain.h	

Table 2.3.2.2-12: Function Summary - draw_patch(patch_offset)

This function draws the immutable terrain, including trees, treelines, and canopies, in an entire patch of Delta terrain. A terrain patch is made up of a number of polygons, as defined by the internal variable, *num_polygons*. Patch_offset is the number of a patch, counting from 0, to draw.

Parameters		
Parameter	Type	Where Typedef Declared
patch_offset	int	standard
Calls		
Function	Where Described	
draw_polygons	this file	
draw_non_ground_polygons	this file	
GET POLY PRIORITY	terrain.h	
GET POLY SHADING	terrain.h	
GET POLY_NUM_VERTS	terrain.h	
mgihue	Concurrent graphics library subroutine	
mgrpoly	Concurrent graphics library subroutine	
mgrl	Concurrent graphics library subroutine	
mgrfc	Concurrent graphics library subroutine	
GET CANOPY POLY_NUM_VERTS	terrain.h	
mgrscalex	Concurrent graphics library subroutine	
mgrscaley	Concurrent graphics library subroutine	
mgistippoly	Concurrent graphics library subroutine	
draw_treeline	this file	
mgisyncrb	Concurrent graphics library subroutine	
shade_desired	Unshade.c	

Table 2.3.2.2-13: Function Summary -draw_stripe(p_left, p_right, row)

This function draws a row of patches. It assumes that terrain planes have been cleared. Tdb_get_stripe returns a pointer to a char.

Parameters		
Parameter	Type	Where Typedef Declared
p_left	int	standard
p_right	int	standard
row	int	standard
Calls		
Function	Where Described	
draw_patch	this file	
tdb_get_stripe	see libtdb in MCC CSCI	

Table 2.3.2.2-14: Function Summary - zoom_to_scale(scale, marked_x, marked_y)

This function zooms map display to scale factor. The scale factor of the new map will be 1:scale. Marked_x, marked_y defines the center of the zoomed region.

Parameters		
Parameter	Type	Where Typedef Declared
scale	int	standard
marked_x, marked_y	int	standard
Return Values		
Return Value	Type	Meaning
unified_paint_square_region (rxl, ryb, rxr, ryt)	int	Coordinates of the new window
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
mgjunscalexy	Concurrent graphics library subroutine	
unified_paint_square_region	this file	

Table 2.3.2.2-15: Function Summary - pan (marked_x, marked_y)

This function pans the display by repainting. Panning is restricted to remaining on the map. Marked_x, marked_y is a marked point which determines pan direction and distance.

Parameters		
Parameter	Type	Where Typedef Declared
marked_x, marked_y	int	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	aborted
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
mgrsi	Concurrent graphics library subroutine	
unified_paint_square_region	this file	

Table 2.3.2.2-16: Function Summary - zoom_out(linear_factor)

This function zooms away from the center of the current field by a given factor. If the zoom out exceeds the top level, the top level map is displayed.

Parameters		
Parameter	Type	Where Typedef Declared
linear factor	float	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	aborted
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgimode	Concurrent graphics library subroutine	
mgrsi	Concurrent graphics library subroutine	
display_top_map	display_top.c	
unified_paint_square_region	this file	

Table 2.3.2.2-17: Function Summary - setup_back_fb(clear_contour_plane)

Parameters		
Parameter	Type	Where Typedef Declared
clear_contour_plane	int	standard
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
mgigetfb	Concurrent graphics library subroutine	
mgifb	Concurrent graphics library subroutine	
mgisyncrb	Concurrent graphics library subroutine	
mgiclearpin	Concurrent graphics library subroutine	

Table 2.3.2.2-18: Function Summary - backout_of_change()

Calls	
Function	Where Described
mgigetfb	Concurrent graphics library subroutine
mgifb	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
copy_map_back	this file

Table 2.3.2.2-19: Function Summary - display_back_fb()

This function sets the currently displayed frame buffer to the current back buffer.

Calls	
Function	Where Described
mgigetfb	Concurrent graphics library subroutine
mgifb	Concurrent graphics library subroutine
mgigetfb	DEBUG ONLY

Table 2.3.2.2-20: Function Summary - copy_map_back()

This function copies the map planes and map area to the back frame buffer.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgibblt2	Concurrent graphics library subroutine

Table 2.3.2.2-21: Function Summary - copy_nonmap_back()

This function copies everything not on the map planes to the back frame buffer.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgifbfbex	Concurrent graphics library subroutine

Table 2.3.2.2-22: Function Summary - print_bb_desc(desc)

This function prints out the contents of a bitblt descriptor.

Parameters		
Parameter	Type	Where Typedef Declared
desc	pointer to BB_DESC	libgpdefs.h

Table 2.3.2.2-23: Function Summary - init_pixel_map_descriptors()

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgibbgetview	Concurrent graphics library subroutine

Table 2.3.2.2-24: Function Summary - force_square(x0, y0, x1, y1)

This function shrinks the upper or right side of the passed rectangle to make it square.

Parameters		
Parameter	Type	Where Typedef Declared
x0, y0	pointer to int	standard
x1, y1	pointer to int	standard
Return Values		
Return Value	Type	Meaning
1	int	resultant square has 0 area.
0	int	square has area other than 0

Table 2.3.2.2-25: Function Summary - init_canopy_pattern()

Calls	
Function	Where Described
mgloadstippat	Concurrent graphics library subroutine
mgjusestippat	Concurrent graphics library subroutine

Table 2.3.2.2-26: Function Summary - set_no_map_on_paint()

This function controls the assignment of setting modes for repainting the map when the screen is redrawn.

Table 2.3.2.2-27: Function Summary - draw_treeline(current_treeline)

This function draws a treeline, given a pointer to a treeline header. It assumes the vertices for the treeline immediately follow the header. The drawing context, including "hue" should be set correctly before calling this routine.

Parameters		
Parameter	Type	Where Typedef Declared
current_treeline	pointer to TREELINE_HEADER	terrain.h
Calls		
Function	Where Described	
mgisyncrb	Concurrent graphics library subroutine	
mgris	Concurrent graphics library subroutine	

Table 2.3.2.2-28: Function Summary - highlight_altitude_hazard(relative hazard_altitude)

Parameters		
Parameter	Type	Where Typedef Declared
relative hazard altitude	float	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	aborted
Calls		
Function	Where Described	
get_elevation	Get_elev.c	
mgrrsi	Concurrent graphics library subroutine	
mgiv	Concurrent graphics library subroutine	
mgipln	Concurrent graphics library subroutine	
mgihue	Concurrent graphics library subroutine	
draw_x	Intervis.c	
tdb_cache_enable	see libtdb in MCC CSCI	
draw_hazards_in_stripe	this file	

Table 2.3.2.2-29: Function Summary - draw_hazard_polygons(first_poly, end_of_polys, vertex_list, hazard_altitude)

This function draws the polygons from first_poly through just before end_of_polys.

Parameters		
Parameter	Type	Where Typedef Declared
first_poly	pointer to POLYGON_DESCRIPTOR	terrain.h
end_of_polys	pointer to POLYGON_DESCRIPTOR	terrain.h
vertex_list	pointer to TERRAIN_POINT	terrain.h
hazard_altitude	float	standard
Calls		
Function	Where Described	
P_POLY_DUPLICATE	terrain.h	
GET_POLY_NUM_VERTS	terrain.h	
mgisyncrb	Concurrent graphics library subroutine	
mgrscalex	Concurrent graphics library subroutine	
mgrscaley	Concurrent graphics library subroutine	
mgrpoly	Concurrent graphics library subroutine	
mgistippoly	Concurrent graphics library subroutine	

**Table 2.3.2.2-30: Function Summary - draw_hazards_in_patch
(patch_offset, hazard_altitude, patch)**

This function draws the polygons in a patch of terrain that have a vertex exceeding hazard_altitude.

Parameters		
Parameter	Type	Where Typedef Declared
patch_offset	int	standard
hazard_altitude	float	standard
patch	int	standard
Calls		
Function	Where Described	
mgrscalex	Concurrent graphics library subroutine	
mgrscaley	Concurrent graphics library subroutine	
mgistippoly	Concurrent graphics library subroutine	
mgisyncrb	Concurrent graphics library subroutine	
draw_hazard_polygons	this file	

**Table 2.3.2.2-31: Function Summary - draw_hazards_in_stripe
(p_left, p_right, row, hazard_altitude)**

This function assumes terrain planes have been cleared.

Parameters		
Parameter	Type	Where Typedef Declared
p_left	int	standard
p_right	int	standard
row	int	standard
hazard_altitude	float	standard
Calls		
Function	Where Described	
tdb_get_stripe	see libtdb in MCC CSCI	
draw_hazards_in_patch	this file	

2.3.3 Finite Zoom CSC Description

This CSC provides the finite zoom optional map feature. This feature supports map drawing at fixed scales from pre-computed bitblt (bit block transfer) files.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.3-1. Additionally, this file uses the file `new_zoom.c` (Section 2.3.2.2).

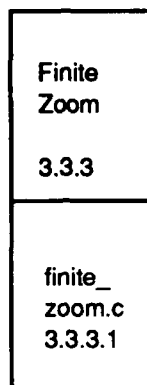


Figure 2.3.3-1: Map Handling--Finite Zoom CSC Structure.

2.3.3.1 `finite_zoom.c`

This file contains routines that implement rapid drawing of the map with pre-computed bitmaps.

Some Abbreviations:

'blt'=block transfer; 'bitblt'=block transfer, bitwise; 'fb'=frame buffer; 'lru'=least recently used.

Performance notes and tradeoffs:

In `blt_paint_square_region`, there is no significant speed up by blting to only the back frame buffer (~6sec) instead of to both (~7sec)!

Performance with various numbers of screens in a bitmap: (4 MBy memory)

- 1 screens per bitmap:	1 panel:	2.4 sec
	2 panels:	4 sec
	4 panels:	7 sec
- 4 screens per bitmap:	1 panel:	6 sec
	2 panels:	13 sec
	4 panels:	20 sec

Why is the 4 screen per bitmap case so much slower? A likely explanation:

A significant time is taken to read in the bitmap from the file -- and each bitmap is 4x larger in this case. Note that the times for 4 panels (1 screen/bmap) and 1 panel (4 screens/bmap) are nearly identical.

This file defines or declares these global variables:

Global Variables	Type and Meaning	Typedef Source	Comments
Map	MAP_DATA		explicitly declared in these functions: •init_discrete_zoom •create_file •create_bitmap •blt_paint_square_region

This file defines these external static variables.

External Static Variables	Type and Meaning	Typedef Source	Comments
current_zoom	int		Reciprocal of scale factor for current bitmap; 0 means no resident bitmap
num_bitmaps	int		Number of bitmaps in memory at current zoom
stored_bitmaps [MAX_BITMAPS_STORED]	array of structs of type BITMAP_CACHE		Sorted on y, then x
file_header	struct of type FILE_HEADER		bit blt
map_window	BB_DESC		descriptor for the visible map on the screen
World_Meters_Per_Pixel	float		meters on map represented by one pixel
use_counter	int		Numbers the bitmaps as they are used, for finding LRUed

Table 2.3.3.1-1: Function Summary: init_discrete_zoom()

This function is called once when the plan view starts up. This function creates the bitmaps needed and initializes the list of supported bitmaps.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgibbgetview	Concurrent graphics library subroutine
count_bits_set	this file
get_db_directory	Init_env.c
make_path_name	Init_env.c
create_file	this file

Table 2.3.3.1-2: Function Summary: clear_stored_bitmaps()

This function resets the finite_zoom software module, clearing all stored bitmaps.

Calls	
Function	Where Described
mgibbfree	Concurrent graphics library subroutine

Table 2.3.3.1-3: Function Summary: init_stored_bitmaps(map_fd, scale)

This function reads the file header and stores this information in external statics. It returns 1 if successful, 0 otherwise. Scale is the reciprocal of map_scale. Map_fd is the file descriptor of the file containing bitmaps of interest.

Parameters		
Parameter	Type	Where Typedef Declared
scale	int	standard
map fd	int	standard
Return Values		
Return Value	Type	Meaning
0	int	file I/O failed
1	int	function successful
Errors		
Error	Reason for Error	
init_stored_bitmaps /lseek	system message	
init_stored_bitmaps /read	system message	

Table 2.3.3.1-4: Function Summary: create_file(scale, map_file)

This function returns the file descriptor of a newly created bitmap file in the following format:

```
*int bitmaps = num_x * num_y
int num_x      bit maps in x direction
int num_y      bit maps in y direction
float x_spacing edge length of bitmap in meters (less in last bitmaps)
float y_spacing
int index[ bitmaps ];  where to lseek to find bmap desc.
BB_DESC desc          \ Repeated bitmaps times char[] bb_data    /
```

Scale is the reciprocal of the scale of the map to create. Map_file is the name of the file to create.

Parameters		
Parameter	Type	Where Typedef Declared
scale	int	standard
map_file	pointer to char	standard

Return Values		
Return Value	Type	Meaning
map_fd	internal integer	returns file descriptor of newly created bitmap file
-1	int	indicates I/O failure
Errors		
Error	Reason for Error	
create_file/open	system message	
create_file/write	system message	
Calls		
Function	Where Described	
mgiv	Concurrent graphics library subroutine	
set no map on paint	new_zoom.c	
set map on paint	new_zoom.c	
mgibbfree	Concurrent graphics library subroutine	
file bitmap	this file	

Table 2.3.3.1-5: Function Summary: create_bitmap()

This function creates a host bitmap for discrete zooming. Since the graphics library can only draw to screen memory, the host bitmap image is constructed out of a series of screen images that are bit blted into host memory.

This function sets up new contour and grid state; allocates memory for bitmap; sets the bitmap frame buffer to the currently displayed frame buffer; loops over all screens, painting the map to the display screen (with grid or contours if requested) and then blting it from the display screen memory to the bitmap in host memory; sets contour spacing; and restores Contour and Grid State.

Return Values		
Return Value	Type	Meaning
bitmap	see Global Variables	returns bitmap
Calls		
Function	Where Described	
get grid spacing	Grid.c	
get contour interval	init_contour.c	
mgibbdesc1	Concurrent graphics library subroutine	
mgibballocc	Concurrent graphics library subroutine	
count_bits_set	this file	
paint_square_region	new_zoom.c	
mgisyncrb	Concurrent graphics library subroutine	
mgibblt2	Concurrent graphics library subroutine	

Table 2.3.3.1-6: Function Summary: file_bitmap(fd, bitmap, x_index, y_index, index_location)

This function passes file index and file pointer. Fd is the file descriptor of the file to write to. This file is assumed to be open and correctly positioned. Bitmap refers to the bitmap to be written to the file. x_index and y_index are the indices of the bitmaps in the file. Index_location is updated as a side effect of this function.

Parameters		
Parameter	Type	Where Typedef Declared
fd	int.	standard
bitmap	BB_DESC	libgpdefs.h
x_index	int.	standard
y_index	int	standard
index_location	pointer to pointer to int	standard
Return Values		
Return Value	Type	Meaning
-1	int	error
Errors		
Error	Reason for Error	
file_bitmap/write(#1)	system message	
file_bitmap/write(#2)	system message	
file_bitmap/write(#3)	system message	
file_bitmap/lseek(#2)	system message	
file_bitmap/lseek	system message	

Table 2.3.3.1-7: Function Summary - count_bits_set(long_word)

This function counts the number of bits set in the passed integer.

Parameters		
Parameter	Type	Where Typedef Declared
long_word	int	standard
Return Values		
Return Value	Type	Meaning
count	internal integer	number of bits in long word that were '1'

Table 2.3.3.1-8: Function Summary - get_shift(long_word)

Returns the number of least significant bits of 0 in the passed 32-bit word. Returns 32 if there are no 1s in the long_word.

Parameters		
Parameter	Type	Where Typedef Declared
long_word	int	standard

Return Values		
Return Value	Type	Meaning
32	int	32 means there were no 1s in long word
bit	int	number of least significant bits of 0 in the passed 32-bit word.

Table 2.3.3.1-9: Function Summary - blt_paint_square_region (x0_m, y0_m, x1_m, y1_m)

This function paints a section of the map into the map window using presaved bitmaps. The section of the map to be painted is specified as a lower left and upper right corner in world coordinates (real world meters measured from the lower left corner) of the database. If the current bitmap file is at the requested scale, the appropriate section of the map is blt'ed to the screen. If not, an attempt is made to open an existing bitmap file at the requested scale, and if this is successful, drawing proceeds as before. Otherwise, a bitmap file is created at the requested scale. (This is a very slow process, involving drawing the entire database at the requested scale, one screen at a time). If even this fails, the requested map is painted directly from the database, using paint_square_region.

Parameters		
Parameter	Type	Where Typedef Declared
x0_m, y0_m	float	standard
x1_m, y1_m	float	standard
Return Values		
Return Value	Type	Meaning
1	int	successful
Calls		
Function	Where Described	
clear_stored_bitmaps	this file	
make_path_name	Init_env.c	
paint_square_region	new_zoom.c	
init_stored_bitmaps	this file	
mgiv	Concurrent graphics library subroutine	
mgrsi	Concurrent graphics library subroutine	
setup_back_fb	New_zoom.c	
mgiclearpln	Concurrent graphics library subroutine	
mgigetfb	Concurrent graphics library subroutine	
mgrvcoor	Concurrent graphics library subroutine	
mgipln	Concurrent graphics library subroutine	
mgihue	Concurrent graphics library subroutine	
mgrbox	Concurrent graphics library subroutine	
display_bitmap	this file	
mgisynrb	Concurrent graphics library subroutine	
zoom_overlay	Overlayif.c	
zoom_icon	Icon.c	
display_back_fb	New_zoom.c	
copy_map_back	New_zoom.c	
get_db_directory	Init_env.c	

Table 2.3.3.1-10: Function Summary - display_bitmap(map_fd, x_index, y_index, start_x, start_y)

This function first checks to see whether bitmap is in memory. If not, the bitmap is loaded; if not, it performs a bitblt into the appropriate location. Map_fd is the file pointer to the necessary bitmap file. Start_x, start_y defines where the lower left corner of the bitmap should be, in screen coordinates.

Parameters		
Parameter	Type	Where Typedef Declared
map_fd	int	standard
x_index, y_index	int	standard
start_x, start_y	int	standard
Calls		
Function	Where Described	
mgibbfree	Concurrent graphics library subroutine	
mgibblt2	Concurrent graphics library subroutine	
mgisyncrb	Concurrent graphics library subroutine	

Table 2.3.3.1-11: Function Summary - get_bitmap(map_fd, x_index, y_index)

This function returns a pointer to a bitmap descriptor containing the bitmap with bitmap coordinates (x_index, y_index) in the file, map_fd. A cache is used to minimize file access. Map_fd refers to the file containing bitmaps for the current zoom.

This function fills an empty cache slot. It replaces the least recently used bitmap in the cache, first freeing old memory.

Parameters		
Parameter	Type	Where Typedef Declared
map_fd	int	standard
x_index	int	standard
y_index	int	standard
Return Values		
Return Value	Type	Meaning
lru_bitmap	internal pointer to type BB_DESC	requested bitmap
Calls		
Function	Where Described	
readin_bitmap	this file	
get_lru_bitmap	this file	
mgibbfree	Concurrent graphics library subroutine	

Table 2.3.3.1-12: Function Summary - get_lru_bitmap()

This function gets the least recently used cache entry.

Return Values		
Return Value	Type	Meaning
least_used	internal integer	array index at lru bitmap in stored bitmaps

**Table 2.3.3.1-13: Function Summary - readin_bitmap
(map_fd, x_index, y_index, bb_desc)**

This function reads in the bitmap with bitmap coordinates (x_index, y_index) in file map_fd, into the passed bitmap descriptor.

Parameters		
Parameter	Type	Where Typedef Declared
map_fd	int	standard
x_index, y_index	int	standard
bb_desc	BB_DESC*	libgpdefs.h
Calls		
Function	Where Described	
mgibballoc	Concurrent graphics library subroutine	
mgibbsize	Concurrent graphics library subroutine	

2.3.4 Save/Recall Zoom CSC Description

This CSC enables a zoom optional map feature to be saved and recalled.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.4-1.

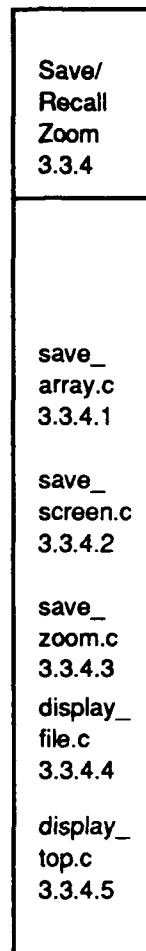


Figure 2.3.4-1: Map Handling--Save/Recall Zoom CSC Structure.

2.3.4.1 save_array.c

This CSU manages the storage and recall of an array to and from a file.

Table 2.3.4.1-1: Function Summary - save_array(array, length, file_name)

This function saves an array to a file. The file is headed by a longword which is the length of the array in bytes.

Parameters		
Parameter	Type	Where Typedef Declared
array	pointer to int	standard
length	int	standard
file_name	pointer to char	standard

Return Values		
Return Value	Type	Meaning
1	int	successful
-1	int	error creating file
-2	int	error in writing file

Table 2.3.4.1-2: Function Summary - recall_array(array, length, file_name)

This function recalls an array from a file created by store_array(...). The file is a binary dump of the array preceeded with a longword header giving the array length in bytes. The array is calloc'ed from heap storage on a longword boundary.

Parameters		
Parameter	Type	Where Typedef Declared
array	pointer to pointer to int	standard
length	pointer to int	standard
file_name	pointer to char	standard

Return Values		
Return Value	Type	Meaning
1	int	successful
-1	int	unable to open file
-2	int	unable to read from file
-3	int	unable to allocate memory

2.3.4.2 save_screen.c

This CSU saves a specified rectangular region of the screen to a file.

Table 2.3.4.2-1: Function Summary - save_screen(x0, y0, x1, y1, plane_mask, file_name)

This function saves a specified rectangular region of the screen to a file. Only the specified planes are saved. Passed coordinates are in window 0 (raw screen) coordinates.

Parameters		
Parameter	Type	Where Typedef Declared
x0, y0	int	standard
x1, y1	int	standard
plane_mask	int	standard
file_name	pointer to char	standard

Return Values		
Return Value	Type	Meaning
-1	int	error occurred in create
-2	int	error occurred in write
1	int	successful

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgigetfbdata	Concurrent graphics library subroutine

2.3.4.3 save_zoom.c

This CSU saves the current terrain map to a file.

Table 2.3.4.3-1: Function Summary - save_zoom()

This function saves the current terrain map to a file.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
get_db_directory	Init env.c
save_screen	this file
mgrsi	Concurrent graphics library subroutine

Table 2.3.4.3-2: Function Summary - recall_zoom()

Calls	
Function	Where Described
get_db_directory	Init_env.c
mgiv	Concurrent graphics library subroutine
mgrvcoor	Concurrent graphics library subroutine
setup_back_fb	New_zoom.c
backout_of_change	New_zoom.c
display_back_fb	New_zoom.c
copy_map_back	New_zoom.c

2.3.4.4 display_file.c

This CSU displays the contents of a file created by save_screen.c.

Table 2.3.4.4-1: Function Summary - display_file(file_name)

This function displays the contents of a file created by save_screen.c. The file's contents are written to the currently displayed frame buffer.

Parameters		
Parameter	Type	Where Typedef Declared
file_name	pointer to char	standard

Return Values		
Return Value	Type	Meaning
-1	int	open failed
-2	int	read failed
1	int	successful

Calls	
Function	Where Described
mgigetfb	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgigetwloc	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgiloadfbdata	Concurrent graphics library subroutine

2.3.4.5 display_top.c

This CSU draws the largest scale map.

Table 2.3.4.5-1: Function Summary - display_top_map()

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	failed or aborted

Calls	
Function	Where Described
setup back fb	New zoom.c
mgipln	Concurrent graphics library subroutine
mgrvcoor	Concurrent graphics library subroutine
make path name	Init env.c
get db directory	Init env.c
display file	Display file.c
zoom overlay	Overlayif.c
display back fb	New zoom.c
copy map back	New zoom.c
top level zoom icon	Icon.c
mgiv	Concurrent graphics library subroutine

2.3.5 Contours CSC Description

This CSC provides the optional contour lines map feature and supports interval setting between contour lines.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.5-1.

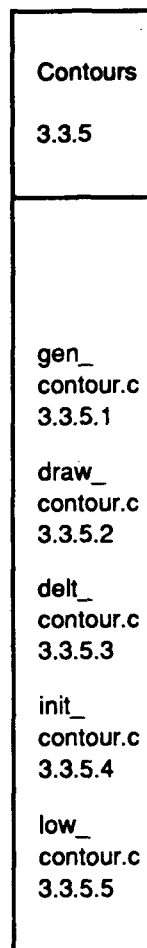


Figure 2.3.5-1: Map Handling--Contours CSC Structure.

2.3.5.1 gen_contour.c

This CSU clears the contour from both frame buffers.

Table 2.3.5.1-1: Function Summary - clear_contour()

This function clears the contour from both frame buffers.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipin	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

2.3.5.2 draw_contour.c

This CSU draws contours in the map window based on current contour levels and zoom.

Table 2.3.5.2-1: Function Summary - draw_contour()

This function draws contours in the MAP_WINDOW based on current contour levels and zoom.

Return Values		
Return Value	Type	Meaning
0	int	contours were aborted
1	int	successful

Calls	
Function	Where Described
return_delta_contour	

Table 2.3.5.2-2: Function Summary - draw_top_contours()

This function draws the contours for the top level map.

Calls	
Function	Where Described
make_path_name	Init env.c
get_db_directory	Init env.c
draw_contour	this file
save_screen	Save screen.c

Table 2.3.5.2-3: Function Summary - add_contours(grid_present)

This function adds contours at the appropriate spacing to the screen.

Parameters		
Parameter	Type	Where Typedef Declared
grid_present	int	standard

Calls	
Function	Where Described
top_level	this file
draw top contours	this file
draw contour	this file
draw grid lines	Grid.c

Table 2.3.5.2-4: Function Summary - top_level()

Return Values		
Return Value	Type	Meaning
1	int	at top level of map display
0	int	at other than top level

Calls	
Function	Where Described
mgmsi	Concurrent graphics library

2.3.5.3 delt_contour.c CSU Description (/simnet/pvd/lib/paintlib.a)

This CSU contours the entire screen from global memory. Contour map is drawn on contour_plane. It draws contours for an individual Delta graphics patch.

Table 2.3.5.3-1: Function Summary - delta_contour(contour_plane)

This function contours the entire screen from global memory. Contour map is drawn on contour_plane.

Parameters		
Parameter	Type	Where Typedef Declared
contour_plane	int	standard

Return Values		
Return Value	Type	Meaning
1	int	contours are completed
0	int	contours are aborted

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgmsi	Concurrent graphics library subroutine
tdb cache disable	see libtdb in MCC CSCI
tdb cache enable	see libtdb in MCC CSCI
draw contour stripe	this file

**Table 2.3.5.3-2: Function Summary - draw_contour_stripe
(left, right, row)**

Parameters		
Parameter	Type	Where Typedef Declared
left	int	standard
right	int	standard
row	int	standard

Calls	
Function	Where Described
contour_patch	this file

Table 2.3.5.3-3: Function Summary - contour_patch(patch_index)

This function draws contours for an individual delta graphics patch.

Parameters		
Parameter	Type	Where Typedef Declared
patch index	int	standard

Calls	
Function	Where Described
GET POLY NUM VERTS	terrain.h
GET POLY PRIORITY	terrain.h
contour_poly	Low contour.c
mgihue	Concurrent graphics library subroutine
draw_delta_poly	Low contour.c

2.3.5.4 init_contour.c CSU Description (/simnet/pvd/lib/paintlib.a)

This CSU sets the initial contour levels. It reads elevation limits and sets contour intervals.

Table 2.3.5.4-1: Function Summary - init_contour()

This function sets the initial contour levels.

Calls	
Function	Where Described
read_elevation_limits	this file
set_contour_interval	this file

Table 2.3.5.4-2: Function Summary - new_contour_interval(interval)

Parameters		
Parameter	Type	Where Typedef Declared
interval	int	standard

Calls	
Function	Where Described
set_contour_interval	this file

Table 2.3.5.4-3: Function Summary - set_contour_interval(start_z, spacing, num_z)

Parameters		
Parameter	Type	Where Typedef Declared
start_z	float	standard
spacing	float	standard
num_z	int	standard

Table 2.3.5.4-4: Function Summary - read_elevation_limits

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	otherwise

Table 2.3.5.4-5: Function Summary - get_contour_interval()

Return Values		
Return Value	Type	Meaning
Map.contour_levels.heights[1] -Map.contour_levels.heights[0]	float	calculation of interval between contour lines in meters

2.3.5.5 low_contour.c

This CSU contours an individual Delta graphics polygon and encodes threshold patterns.

Table 2.3.5.5-1: Function Summary - contour_poly(num_vertex, vertex, vertex_list, z_levels, num_z_levels)

This function contours an individual Delta graphics polygon. The polygon is assumed to be convex, coplaner and 3 or 4 sided.

Parameters		
Parameter	Type	Where Typedef Declared
num_vertex	int	standard
vertex	unsigned short	standard
vertex_list	pointer to TERRAIN_POINT	terrain.h
z_levels	float	standard
num_z_levels	int	standard

Calls	
Function	Where Described
encode_contour	this file
mgisyncrb	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine

Table 2.3.5.5-2: Function Summary - encode_contour(threshold, poly, num_points, vertex_list, edge_1, edge_2)

This function computes the index of the starting points of the 2 edges that the contour intersects.

Parameters		
Parameter	Type	Where Typedef Declared
threshold	register float	standard
poly	unsigned short	standard
num_points	int	standard
vertex_list	pointer to TERRAIN_POINT	terrain.h
edge_1	pointer to int	standard
edge_2	pointer to int	standard

Return Values		
Return Value	Type	Meaning
0	int	no intersection
1	int	otherwise

**Table 2.3.5.5-3: Function Summary - draw_delta_poly
(num_vertex, vertex, vertex_list)**

This function erases an individual Delta graphics polygon.

Parameters		
Parameter	Type	Where Typedef Declared
num_vertex	int	standard
vertex	unsigned char	standard
vertex_list	pointer to TERRAIN_POINT	terrain.h

Calls	
Function	Where Described
mgrpoly	Concurrent graphics library

2.3.6 Grid CSC Description

This CSC provides the optional UTM grid lines map feature and supports interval setting between grid lines.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.6-1.

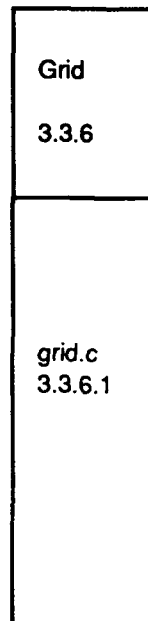


Figure 2.3.6-1: Map Handling--Grid CSC Structure.

2.3.6.1 grid.c CSU Description (/simnet/pvd/lib/paintlib.a)

This CSU provides functions for overlaying UTM (universal terrain meridian) grid lines.

**Table 2.3.6.1-1: Function Summary - draw_horiz_grid_lines
(rxl, ryb, rxr, ryt)**

Parameters		
Parameter	Type	Where Typedef Declared
rxl	float	standard
ryb	float	standard
rxr	float	standard
ryt	float	standard

Calls	
Function	Where Described
tdb_giv_xy_get_utm	see libtdb in MCC CSCI
calc_first_position	this file
mgrl	Concurrent graphics library subroutine
mgrscaley	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

**Table 2.3.6.1-2: Function Summary - draw_vert_grid_lines
(rxl, ryb, rxr, ryt)**

Parameters		
Parameter	Type	Where Typedef Declared
rxl	float	standard
ryb	float	standard
rxr	float	standard
ryt	float	standard

Calls	
Function	Where Described
tdb_giv_xy_get_utm	see libtdb in MCC CSCI
calc_first_position	this file
mgrl	Concurrent graphics library subroutine
mgrscalex	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.3.6.1-3: Function Summary - draw_grid_lines()

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgrsi	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgidash	Concurrent graphics library subroutine
draw_vert_grid_lines	this file
draw_horiz_grid_lines	this file

Table 2.3.6.1-4: Function Summary - calc_first_position(bound)

When grid lines are drawn, the lines must always appear at the same map meter points as they would on a full-scale map. Therefore, the x or y coordinate of the first line must be determined based on the current viewport coordinates. This is done by taking the lower x or y bound, dividing by the grid_spacing, adding 1 (to round up) and multiplying again by the grid_spacing.

Parameters		
Parameter	Type	Where Typedef Declared
bound	float	standard

Return Values		
Return Value	Type	Meaning
start_loc + (MAX_SPACING - start_loc)	static int	algorithm for calculating first position for grid spacing
start_loc	static int	bound / grid_spacing + 1
!(val%MAX_SPACING)	static int	value of MAX_SPACING

Table 2.3.6.1-5: Function Summary - change_grid_spacing(new_val)

Parameters		
Parameter	Type	Where Typedef Declared
new_val	int	standard

Table 2.3.6.1-6: Function Summary - get_grid_spacing()

Return Values		
Return Value	Type	Meaning
grid_spacing	int	value of grid_spacing

2.3.7 Other Features CSC Description

This CSC provides other map features, such as terrain shade and color. It also accesses a previous map view.

This CSC functionality is broken down into the CSUs shown in Figure 2.3.7-1.

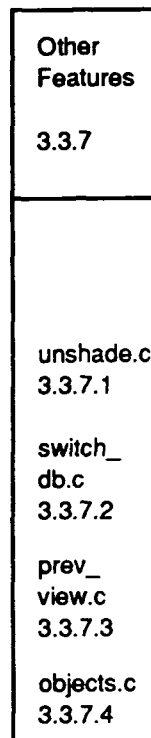


Figure 2.3.7-1: PVD-Level--Initial PVD-Level CSC Structure.

2.3.7.1 unshade.c

This CSU turns map shading on or off.

Table 2.3.7.1-1: Function Summary - shade_desired()

Return Values		
Return Value	Type	Meaning
shade	static int	returns shade

Table 2.3.7.1-2: Function Summary - toggle_shade()

Calls	
Function	Where Described
shading menu entry off	Edit_menu.c
shading menu entry on	Edit_menu.c
handle repaint	handle_input.c

Table 2.3.7.1-3: Function Summary - silent_shade_off()

This function assigns map shading off.

Table 2.3.7.1-4: Function Summary - silent_shade_on()

This function assigns map shading on.

2.3.7.2 switch_db.c

This CSU switches databases.

Table 2.3.7.2-1: Function Summary - switch_database(entry_num)

Parameters		
Parameter	Type	Where Typedef Declared
entry_num	int	standard

Calls	
Function	Where Described
get_db_names	Init_env.c
check mark	Menu_fea.c
reinit db	New_zoom.c

2.3.7.3 prev_view.c

This CSU displays previous and next map views; manages storage of views in buffers.

Table 2.3.7.3-1: Function Summary - previous_view()

This function displays the previous map view.

Return Values		
Return Value	Type	Meaning
0	int	aborted or no previous view

Calls	
Function	Where Described
ring inc	this file
ring dec	this file
display top map	Display top.c
unified paint square region	New zoom.c
display views	this file

Table 2.3.7.3-2: Function Summary - test_front()

Return Values		
Return Value	Type	Meaning
1	int	buffer preceeding the front doesn't match the current viewport real coordinates
0	int	successful

Calls	
Function	Where Described
ring dec	this file
mgrsi	Concurrent graphics library subroutine

Table 2.3.7.3-3: Function Summary - next_view()

This function display the "next" view, if any.

Return Values		
Return Value	Type	Meaning
0	int	no more views to display

Calls	
Function	Where Described
display top map	Display top.c
unified paint square region	New zoom.c
ring inc	this file
display views	this file

Table 2.3.7.3-4: Function Summary - add_view()

This function is called after a new view is created.

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
ring inc	this file
display views	this file

Table 2.3.7.3-5: Function Summary - ring_dec(ring_var, ring_size)

Parameters		
Parameter	Type	Where Typedef Declared
ring_var	int	standard
ring_size	int	standard

Return Values		
Return Value	Type	Meaning
ring_var	int	integer value of ring_var

Table 2.3.7.3-6: Function Summary - ring_inc(ring_var, ring_size)

Parameters		
Parameter	Type	Where Typedef Declared
ring_var	int	standard
ring_size	int	standard

Return Values		
Return Value	Type	Meaning
ring_var	int	integer value of ring_var

Table 2.3.7.3-7: Function Summary - get_cached_bitmap(xl, yb, xr, yt)

This function returns a file descriptor for any cached bitmap matching the current coordinates.

Parameters		
Parameter	Type	Where Typedef Declared
xll	float	standard
yb	float	standard
yr	float	standard
yf	float	standard

Return Values		
Return Value	Type	Meaning
-1	int	failed open

Table 2.3.7.3-8: Function Summary - create_cached_bitmap(xl, yb, xr, yt)

Parameters		
Parameter	Type	Where Typedef Declared
xl	float	standard
yb	float	standard
xy	float	standard
yt	float	standard

Table 2.3.7.3-9: Function Summary - print_buffer_status()

This function prints the current state of the view ring_buffer. It displays values for view_ring_oldest, view_ring_newest, and view_ring_front.

Table 2.3.7.3-10: Function Summary - reset_views()

This function sets the number of saved views to zero. It is used when changing databases and all old views are invalid.

Calls	
Function	Where Described
display_views	this file

Table 2.3.7.3-11: Function Summary - display_views()

This function displays a graphical depiction of the ring buffer as an aid to the user.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
ring_inc	this file
mgibox	Concurrent graphics library subroutine

2.3.7.4 objects.c

This CSU enables printing a description of a moused map object. This file interprets the user's gestures with the mouse and invokes the appropriate actions.

Table 2.3.7.4-1: Function Summary - print_nearest_object(map_x, map_y)

Parameters		
Parameter	Type	Where Typedef Declared
map_x	int	standard
map_y	int	standard

Calls	
Function	Where Described
mgijunscalexy	Concurrent graphics library subroutine
tdb_close_thing	see libtdb in MCC CSCI
tdb_thing_string	see libtdb in MCC CSCI

2.4 PVD-LEVEL PROCESSING

This CSC provides shared data types, definitions, shared procedures and libraries.

The main loop of the PVD includes functions that get the next available packet from the network, process the packet, storing data and/or undertaking actions. Positions of graphic icons on the screen are updated, based on packets received. The main loop also processes user commands.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-4.

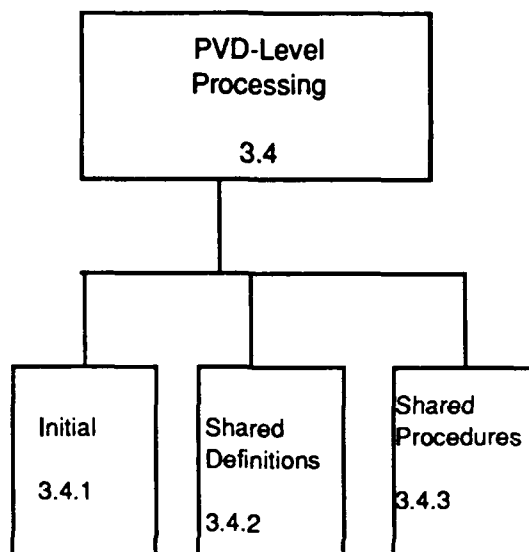


Figure 2-4: PVD-Level CSC Structure.

2.4.1 Initial PVD-Level CSC Description

This CSC is the main entry for PVD-level processing.

This CSC functionality is broken down into the CSUs shown in Figure 2.4.1-1.

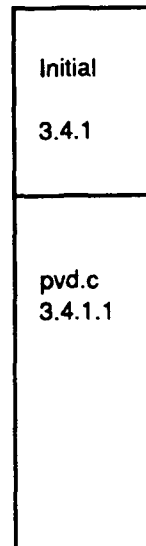


Figure 2.4.1-1: PVD-Level--Initial PVD-Level CSC Structure.

2.4.1.1 pvd.c

This CSU provides the main loop of the PVD. It contains routines to do the following things:

- Get the next available packet from the network.
- Process the packet, store data and/or undertake actions.
- Update positions of graphics icons on the screen.
- Process user commands.

Table 2.4.1.1-1: Function Summary - exit_gracefully()

Calls	
Function	Where Described
thats all folks	this file

Table 2.4.1.1-2: Function Summary - exit_disgracefully()

This function operates like `exit_gracefully`, but it does a core dump rather than an exit (if CATC is being used) after cleaning up.

Calls	
Function	Where Described
thats_all_folks	this file
catc_on	

Table 2.4.1.1-3: Function Summary - thats_all_folks()

Calls	
Function	Where Described
mgiflush	Concurrent graphics library subroutine
mgiclearpln	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
setup back fb	New zoom.c
display back fb	New zoom.c
mgicm	Concurrent graphics library subroutine
close user interface	Pvd iface.c
clear screen	text.c
net get statistics	libnetif.a in MCC CSCI
close c sect server	terrain_pro.c
table_uninit	

**Table 2.4.1.1-4: Function Summary - process_nonappearance_packets
(process_special_packets)**

Parameters		
Parameter	Type	Where Typedef Declared
process_special_packets	pointer to int	standard

Calls	
Function	Where Described
get_next_packet	Virtual net.c
Collect World State	Pvd_misc.c

2.4.2 Shared Definitions CSC Description

This CSC provides shared definitions, including protocols, for PVD Procedures.

This CSC functionality is broken down into the CSUs shown in Figure 2.4.2-1.



Figure 2.4.2-1: PVD-Level--Initial PVD-Level CSC Structure.

2.4.2.1 p_assoc.h

This is a commonly-maintained protocol file used with Plan View Display and other SIMNET applications.

2.4.3 Shared Procedures CSC Description

This CSC contains procedures and libraries used by more than one CSC in the PVD.

This CSC functionality is broken down into the CSUs shown in Figure 2.4.3-1.

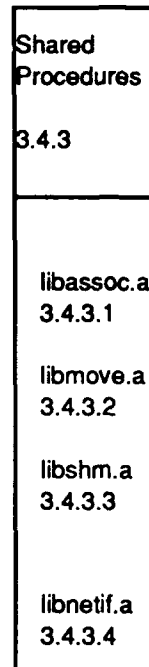


Figure 2.4.3-1: PVD-Level--Initial PVD-Level CSC Structure.

2.4.3.1 libassoc.a

This CSU implements the SIMNET association protocol.

2.4.3.2 libmove.a

This CSU is used by libnetif.a

2.4.3.3 libshm.a

This CSU is used by libnetif.a. It provides access to shared memory.

2.4.3.4 libnetif.a

This CSU provides a low level interface to the network.

2.5 UTILITIES

This CSC includes commonly-used routines that support text input/output, command and options processing, map and icon display processing, and debugging.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-5.

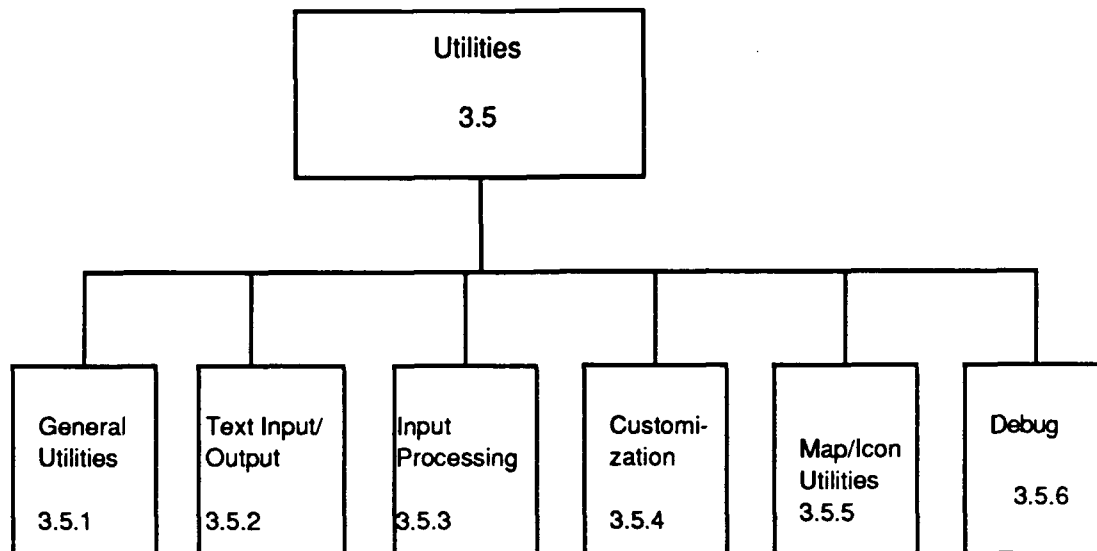


Figure 2-5: Utilities CSC Structure.

2.5.1 General Utilities CSC Description

This CSC provides general PVD-level utilities.

This CSC functionality is broken down into the CSUs shown in Figure 2.5.1-1.

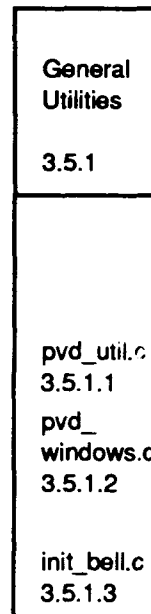


Figure 2.5.1-1: Utilities--General Utilities CSC Structure.

2.5.1.1 pvd_util.c

This CSU forces an integer variable to lie within a specified range.

Table 2.5.1.1-1: Function Summary - force_to_range(i, lower, upper)

This function forces an integer variable to lie within a specified range. "i" is the value to be forced within the range; "lower" and "upper" specify the lower and upper limits of the range.

Parameters		
Parameter	Type	Where Typedef Declared
i	pointer to int	standard
lower, upper	int	standard

2.5.1.2 pvd_windows.c CSU Description (/simnet/pvd/lib/pvdface.a)

This CSU initializes and manages the windows on the PVD screen.

Table 2.5.1.2-1: Function Summary - set_menu_window_top(y)

This function sets the value of the menu window top.

Parameters		
Parameter	Type	Where Typedef Declared
y	int	standard

Table 2.5.1.2-2: Function Summary - get_menu_window_top()

Return Values		
Return Value	Type	Meaning
window menu top	int	value of menu window top

Table 2.5.1.2-3: Function Summary - init_windows()

Calls	
Function	Where Described
init_info_window	this file
init_event_window	this file
init_legend_window	this file
init_utm_window	this file

Table 2.5.1.2-4: Function Summary - init_info_window()

Calls	
Function	Where Described
mgidew	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
init_info_text_window	this file

Table 2.5.1.2-5: Function Summary - init_info_text_window()

Calls	
Function	Where Described
mgimktwgw	Concurrent graphics library subroutine
draw_window_border	Pvd_windows.c
set_text_window	T_windows.c
mgicm	Concurrent graphics library subroutine

Table 2.5.1.2-6: Function Summary - init_event_window()

Calls	
Function	Where Described
mgidfw	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
init_event_text_window	this file

Table 2.5.1.2-7: Function Summary - init_event_text_window()

Calls	
Function	Where Described
mgimktwgw	Concurrent graphics library subroutine
draw_window_border	Pvd iface.c

Table 2.5.1.2-8: Function Summary - reinit_event_text_window()

Calls	
Function	Where Described
mgimktwgw	Concurrent graphics library subroutine
draw_window_border	Pvd windows.c

Table 2.5.1.2-9: Function Summary - erase_event_window_border()

Calls	
Function	Where Described
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine

Table 2.5.1.2-10: Function Summary - init_time_window()

Calls	
Function	Where Described
mgidfw	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine

Table 2.5.1.2-11: Function Summary - init_legend_window()

Calls	
Function	Where Described
mgidfw	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
draw_window_border	Pvd windows.c
display_key_text	Pvd windows.c

Table 2.5.1.2-12: Function Summary - display_map_key()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
display_stripe	Stripe.c
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
display_map key text	this file

Table 2.5.1.2-13: Function Summary - display_map_key_text()

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-14: Function Summary - default_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-15: Function Summary - freedraw_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-16: Function Summary - resize_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-17: Function Summary - move_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-18: Function Summary - abort_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-19: Function Summary - catc_place_tgt_mvr_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-20: Function Summary - catc_place_tgt_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

**Table 2.5.1.2-21: Function Summary -
catc_move_tgt_pickup_mouse_help()**

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-22: Function Summary - catc_move_tgt_place_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-23: Function Summary - catc_remove_tgt_mouse_help()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-24: Function Summary - display_mouse_key()

Calls	
Function	Where Described
mouse info	

Table 2.5.1.2-25: Function Summary - set_mouse_text(func)

Calls	
Function	Where Described
display_mouse_key	this file

Table 2.5.1.2-26: Function Summary - display_intervis_key()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
display_intervis_key_text	this file

Table 2.5.1.2-27: Function Summary - display_intervis_key_text()

Calls	
Function	Where Described
get view height	Get view.c
get target height	Get view.c
get view range	Get view.c
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-28: Function Summary - display_icon_key()

Calls	
Function	Where Described
mgiclearpln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgixy	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgisegimm	Concurrent graphics library subroutine

Table 2.5.1.2-29: Function Summary - display_icon_key_text()

Calls	
Function	Where Described
mgipln	Concurrent graphics library subroutine

Table 2.5.1.2-30: Function Summary - draw_fixed_faad
(hull_hue, turret_hue)

Parameters		
Parameter	Type	Where Typedef Declared
hull hue	int	standard
turret hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.5.1.2-31: Function Summary - draw_fixed_fwa(hull_hue)

Parameters		
Parameter	Type	Where Typedef Declared
hull hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

**Table 2.5.1.2-32: Function Summary - draw_fixed_m2
(hull_hue, turret_hue)**

Parameters		
Parameter	Type	Where Typedef Declared
hull hue	int	standard
turret hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

**Table 2.5.1.2-33: Function Summary - draw_fixed_tank
(hull_hue, turret_hue)**

Parameters		
Parameter	Type	Where Typedef Declared
hull hue	int	standard
turret hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

**Table 2.5.1.2-34: Function Summary - draw_fixed_static
(hull_hue, turret_hue)**

Parameters		
Parameter	Type	Where Typedef Declared
hull hue	int	standard
turret hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.5.1.2-35: Function Summary - draw_fixed_burst(hue)

Parameters		
Parameter	Type	Where Typedef Declared
hue	int	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgixy	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine

Table 2.5.1.2-36: Function Summary - display_key_text()

Calls	
Function	Where Described
display_map_key	Pvd_windows.c
display_intervis_key	Pvd_windows.c

Table 2.5.1.2-37: Function Summary - init_utm_window()

Calls	
Function	Where Described
mgidew	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
draw_window_border	Pvd_windows.c
init_zoom_scale	Pvd_windows.c
init_icon_scale	Pvd_windows.c
init_exercise	Pvd_windows.c

Table 2.5.1.2-38: Function Summary - get_exercise_time(num_packets)

Parameters		
Parameter	Type	Where Typedef Declared
num_packets	int	standard

Return Values		
Return Value	Type	Meaning
time	char	prints time in hrs, mins, secs

Table 2.5.1.2-39: Function Summary - display_time(start_time)

Parameters		
Parameter	Type	Where Typedef Declared
start time	int	standard

Table 2.5.1.2-40: Function Summary - init_zone_window()

Calls	
Function	Where Described
mgidew	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
draw zone	

Table 2.5.1.2-41: Function Summary - draw_window_border
(top, bottom, left, right)

Parameters		
Parameter	Type	Where Typedef Declared
top	int	standard
bottom	int	standard
left	int	standard
right	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine

Table 2.5.1.2-42: Function Summary - init_zoom_scale()

Calls	
Function	Where Described
calc_zoom_Scale	this file

Table 2.5.1.2-43: Function Summary - calc_zoom_scale()

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.1.2-44: Function Summary - init_icon_scale()

Calls	
Function	Where Described
write string	Write string.c

Table 2.5.1.2-45: Function Summary - show_icon_scale(icon_scale)

Parameters		
Parameter	Type	Where Typedef Declared
icon scale	long	standard

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.1.2-46: Function Summary - init_exercise()

Calls	
Function	Where Described
change_exercise	this file

Table 2.5.1.2-47: Function Summary - change_exercise()

Calls	
Function	Where Described
mgimode	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.1.2-48: Function Summary - change_intervis_key_text()

Calls	
Function	Where Described
display_intervis_key	Pvd_windows.c

2.5.1.3 init_bell.c

This CSU sets up the bell for the Masscomp graphics terminal.

2.5.2 Text Input/Output CSC Description

This CSC provides text input/output capability.

This CSC functionality is broken down into the CSUs shown in Figure 2.5.1-1.

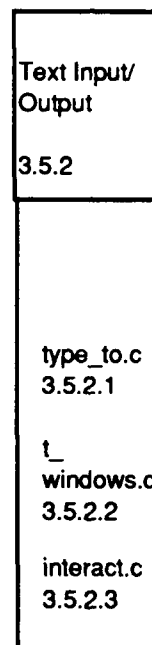


Figure 2.5.2-1: Utilities--Text Input/Output CSC Structure.

2.5.2.1 type_to.c

This CSU outputs a string to a particular device.

Table 2.5.2.1-1: Function Summary - type_to_device(echo_string, device)

Parameters		
Parameter	Type	Where Typedef Declared
echo_string	pointer to char	standard
device	pointer to char	standard

2.5.2.2 t_windows.c

This CSU sets the text window.

Table 2.5.2.2-1: Function Summary - set_text_window(t_window)

Parameters		
Parameter	Type	Where Typedef Declared
t_window	pointer to char	standard

2.5.2.3 interact.c

This CSU manages interactive keyboard input. It gives prompt strings and receives input.

Table 2.5.2.3-1: Function Summary - get_typed_input(prompt)

Parameters		
Parameter	Type	Where Typedef Declared
prompt	pointer to char	standard

Return Values		
Return Value	Type	Meaning
user_input	char	pointer to user input

Table 2.5.2.3-2: Function Summary - get_typed_input_float
(prompt, default_val)

Parameters		
Parameter	Type	Where Typedef Declared
prompt	pointer to char	standard
default_val	float	standard

Return Values		
Return Value	Type	Meaning
user_input	float	pointer to user input

**Table 2.5.2.3-3: Function Summary - get_typed_input_str
(prompt, default_str)**

Parameters		
Parameter	Type	Where Typedef Declared
prompt	pointer to char	standard
default_str	pointer to char	standard

Return Values		
Return Value	Type	Meaning
user_buffer	char	pointer to user buffer

**Table 2.5.2.3-4: Function Summary - input_name(prompt,
user_buffer, default_str)**

Parameters		
Parameter	Type	Where Typedef Declared
prompt	pointer to char	standard
user_buffer	pointer to char	standard
default_str	pointer to char	standard

2.5.3 Input Processing CSC Description

This CSC processes commands and command options and other forms of user input.

This CSC functionality is broken down into the CSUs shown in Figure 2.5.3-1.

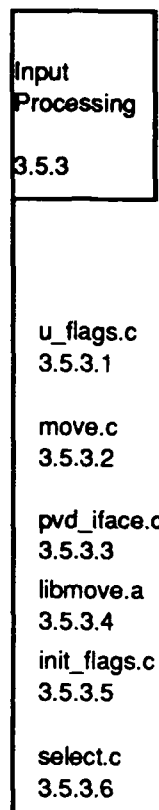


Figure 2.5.3-1: Utilities--Input Processing CSC Structure.

2.5.3.1 u_flags.c

This CSU stores the information from the command line.

**Table 2.5.3.1-1: Function Summary - process_unix_flags
(argc, argv, unix_flags)**

This function stores the information from the command line.

Parameters		
Parameter	Type	Where Typedef Declared
argc	int	standard
argv	pointer to char	standard
unix_flags	pointer to struct U_FLAGS	u_flags.h

Return Values		
Return Value	Type	Meaning
0	int	successful

2.5.3.2 move.c CSU Description (/simnet/pvd/lib/pvdface.a)

This CSU enables selecting a region of terrain with select_region. This file interprets the user's gestures with the mouse and invoke the appropriate actions.

Table 2.5.3.2-1: Function Summary - move_window(w)

Parameters		
Parameter	Type	Where Typedef Declared
w	int	standard

Calls	
Function	Where Described
get_window_num	Pop_windows.c
mgiv	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgigetkursxy	Concurrent graphics library subroutine
mgietwloc	Concurrent graphics library subroutine
mgrgetwloc	Concurrent graphics library subroutine
constrain_to_screen	this file
mgifb	Concurrent graphics library subroutine
mgipin	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgifbfbex	Concurrent graphics library subroutine
mgimw	Concurrent graphics library subroutine

Table 2.5.3.2-2: Function Summary - clear_window(w)

Parameters		
Parameter	Type	Where Typedef Declared
w	int	standard

Calls	
Function	Where Described
mgigetwloc	Concurrent graphics library subroutine
mgrgetwloc	Concurrent graphics library subroutine
mguhue	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.3.2-3: Function Summary - select_region(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x,y	int	standard

Calls	
Function	Where Described
set mark	pvd iface.c
mgigetkursxy	Concurrent graphics library subroutine
set first point	pvd iface.c
set point	pvd iface.c

Table 2.5.3.2-4: Function Summary - constrain_to_screen(xl, yb, xr, yt, screen_xl, screen_yb, screen_xr, screen_yt)

This function forces a rectangular region to be within the screen. It assumes the region will fit on the screen. The region is translated the smallest possible distance to put it on the screen.

Parameters		
Parameter	Type	Where Typedef Declared
xl, yb, xr, yt	pointer to int	standard
screen_xl	int	standard
screen_yb	int	standard
screen_xr	int	standard
screen_yt	int	standard

Calls	
Function	Where Described
get window num	Pop windows.c

2.5.3.3 pvd_iface.c

This CSU enables a point to be selected. This file interprets the user's gestures with the mouse and invokes the appropriate actions.

Table 2.5.3.3-1: Function Summary - init_user_interface()

Calls	
Function	Where Described
mgibuttonint	Concurrent graphics library subroutine
mgewidth	Concurrent graphics library subroutine
init_fonts	this file
init_menus	Menu.c
mgicursmode	Concurrent graphics library subroutine
draw_setup_menu	Menu.c
init_working_cursor	this file
init_windows	Pvd_windows.c
make_path_name	init_env.c
get_db_directory	init_env.c

Table 2.5.3.3-2: Function Summary - init_fonts()

Calls	
Function	Where Described
mgifetchgf	Concurrent graphics library subroutine
mgideltf	Concurrent graphics library subroutine
mgifgf	Concurrent graphics library subroutine

Table 2.5.3.3-3: Function Summary - close_user_interface()

Calls	
Function	Where Described
mgicursmode	Concurrent graphics library subroutine
mgisynrb	Concurrent graphics library subroutine

Table 2.5.3.3-4: Function Summary - init_cursor()

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library subroutine

Table 2.5.3.3-5: Function Summary - init_working_cursor()

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library

Table 2.5.3.3-6: Function Summary - init_cross_cursor()

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library subroutine

Table 2.5.3.3-7: Function Summary - button(x, y, s)

Parameters		
Parameter	Type	Where Typedef Declared
x	unsigned int	standard
y	unsigned int	standard
s	unsigned int	standard

Table 2.5.3.3-8: Function Summary - handle_mouse(mouse)

Parameters		
Parameter	Type	Where Typedef Declared
mouse	int	standard

Calls	
Function	Where Described
handle_popup windows	Pop_windows.c
handle menu	Menu.c
select_region	Move.c
add_selected_point	this file
handle_highlight_point	handle_input.c
print_nearest_object	Objects.c

Table 2.5.3.3-9: Function Summary - set_mark(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Calls	
Function	Where Described
toggle_selected	this file

Table 2.5.3.3-10: Function Summary - set_first_point(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Calls	
Function	Where Described
toggle_selected	this file

Table 2.5.3.3-11: Function Summary - set_point(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Calls	
Function	Where Described
toggle_selected	this file

Table 2.5.3.3-12: Function Summary - toggle_selected(hue)

Parameters		
Parameter	Type	Where Typedef Declared
hue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimodfunc	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine

Table 2.5.3.3-13: Function Summary - get_coord(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Calls	
Function	Where Described
get_elevation	Get_elev.c
pixel to user	this file
tdb giv xy get utm	see libtdb in MCC CSCI
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.3.3-14: Function Summary - xy_get_coord(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Calls	
Function	Where Described
get_elevation	Get elev.c
pixel_to_user	this file
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.3.3-15: Function Summary - user_to_pixel(px, py, rx, ry)

Parameters		
Parameter	Type	Where Typedef Declared
px	pointer to int	standard
py	pointer to int	standard
rx	float	standard
ry	float	standard

Calls	
Function	Where Described
mgrscalexy	Concurrent graphics library subroutine

Table 2.5.3.3-16: Function Summary - pixel_to_user(px, py, rx, ry)

Parameters		
Parameter	Type	Where Typedef Declared
px	int	standard
py	int	standard
rx	pointer to float	standard
ry	pointer to float	standard

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine

Table 2.5.3.3-17: Function Summary - add_selected_point(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	int	standard
y	int	standard

Table 2.5.3.3-18: Function Summary - current_loc(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	float	standard
y	float	standard

Calls	
Function	Where Described
mgrscalexy	Concurrent graphics library subroutine
tdb_giv_xy_get_utm	see libtdb in MCC CSCI
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.5.3.3-19: Function Summary - xy_current_loc(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x	real	standard
y	real	standard

Calls	
Function	Where Described
mgrscalexy	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

2.5.3.4 libmove.a

This is a common library used by PVD and other SIMNET applications.

2.5.3.5 init_flags.c

This CSU initializes the global flags used to communicate between the main loop and the menu system.

2.5.3.6 select.c CSU Description (/simnet/pvd/lib/pvdiface.a)

This CSU contains routines for selecting vehicles.

Table 2.5.3.6-1: Function Summary - tank_selected(id)

Parameters		
Parameter	Type	Where Typedef Declared
id	int	standard

Return Values		
Return Value	Type	Meaning
1	int	passed tank id selected
0	int	not selected

Table 2.5.3.6-2: Function Summary - toggle_nearest_tank(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x,	int	standard
y	int	standard

Return Values		
Return Value	Type	Meaning
1	int	tank was nearby and its highlight state was toggled
0	int	not the above

Calls	
Function	Where Described
get true icon scale	Icon.c
get world state	
mgrscalexy	Concurrent graphics library subroutine
mgrrsi	Concurrent graphics library subroutine
add selected tank	this file
detail display	Detail.c
remove Selected tank	this file

Table 2.5.3.6-3: Function Summary - remove_selected_tank(vehicle_index)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_index	int	standard

Calls	
Function	Where Described
unhighlight tank	Highlight.c

Table 2.5.3.6-4: Function Summary - add_selected_tank(vehicle_index)

This function adds the vehicle indicated by the passed vehicle_index to the list of selected vehicles.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_index	int	standard

Calls	
Function	Where Described
highlight tank	Highlight.c

Table 2.5.3.6-5: Function Summary - my_min(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x,	int	standard
y	int	standard

Return Values		
Return Value	Type	Meaning
x	int	x<y
y	int	value of y

Table 2.5.3.6-6: Function Summary - my_max(x, y)

Parameters		
Parameter	Type	Where Typedef Declared
x,	int	standard
y	int	standard

Return Values		
Return Value	Type	Meaning
x	int	x>y
y	int	value of y

Table 2.5.3.6-7: Function Summary - vehicle_in_region(x, y, region)

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard
region	struct REGION_FLAG	pvd_flags.h

Return Values		
Return Value	Type	Meaning
yes	int	vehicle not in specified region
no	int	vehicle in specified region

Calls	
Function	Where Described
my_min	this file
my_max	this file
mgrscalexy	Concurrent graphics library subroutine

Table 2.5.3.6-8: Function Summary - clear_selected_vehicles()

Calls	
Function	Where Described
unhighlight tank	Highlight.c

Table 2.5.3.6-9: Function Summary - select_vehicle_by_name()

This function causes the entered vehicle to become selected. If the entered string is a vehicle id, vehicles are searched for a matching id. If no match, the passed string is compared to vehicle markings.

Calls	
Function	Where Described
get typed input str	interact.c
get world state	
is_vehicle id string	this file
vehicle id cmp	this file
add selected tank	this file
detail display	Detail.c

Table 2.5.3.6-10: Function Summary - is_vehicle_id_string
(input_string, vehicle_id)

Parameters		
Parameter	Type	Where Typedef Declared
input_string	pointer to char	standard
vehicle_id	pointer to VehicleID	basic.h

Return Values		
Return Value	Type	Meaning
1	int	The passed string could represent a vehicle id. Fills in the passed vehicle_id accordingly.
0	int	Otherwise than above

**Table 2.5.3.6-11: Function Summary - vehicle_id_cmp
(vehicle_id_1, vehicle_id_2)**

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id_1	pointer to VehicleID	basic.h
vehicle_id_2	pointer to VehicleID	basic.h

Return Values		
Return Value	Type	Meaning
1	int	Contents of the passed vehicle ids are equal
0	int	Otherwise than above

2.5.4 Customization CSC Description

This CSC provides customizing parameters for PVD functions.

This CSC functionality is broken down into the CSUs shown in Figure 2.5.4-1.

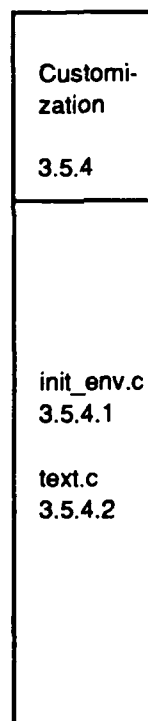


Figure 2.5.4-1: Utilities--Customization CSC Structure.

2.5.4.1 init_env.c

This CSU contains routines for initializing the environment in PVD.

Table 2.5.4.1-1: Function Summary - init_environment()

This function reads environment variables from a file.

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	failed

Calls	
Function	Where Described
get stealth address	this file
flag stealth menu	Menu.c
flag catc menu	Menu.c
set default menu	Menu.c
set time out msec	Purge.c
set exercise	Pvd misc.c
set firefinder	Indir fire.c
set bfit mode	
set nlos mode	

Table 2.5.4.1-2: Function Summary - get_data_directory()

Return Values		
Return Value	Type	Meaning
ENVVIRON.data_directory	pointer to char	value of data_directory

Table 2.5.4.1-3: Function Summary - get_vr_host()

Return Values		
Return Value	Type	Meaning
ENVIRON.vr_host	pointer to char	value of vr_host

Table 2.5.4.1-4: Function Summary - get_pt_host()

Return Values		
Return Value	Type	Meaning
ENVIRON.pt_host	pointer to char	value of pt_host

Table 2.5.4.1-5: Function Summary - get_rc_host()

Return Values		
Return Value	Type	Meaning
ENVIRON.rc_host	pointer to char	value of rc_host

Table 2.5.4.1-6: Function Summary - get_dp_address()

Return Values		
Return Value	Type	Meaning
ENVIRON.dp_address	pointer to unsigned char	value of dp_address

Table 2.5.4.1-7: Function Summary - get_db_directory()

Return Values		
Return Value	Type	Meaning
ENVIRON.db_directory	pointer to char	value of db_directory

**Table 2.5.4.1-8: Function Summary - make_path_name
(directory, file_name, path)**

This function copies the supplied directory name and file name into the specified string.

Parameters		
Parameter	Type	Where Typedef Declared
directory	pointer to char	standard
file_name	pointer to char	standard
path	pointer to char	standard

Table 2.5.4.1-9: Function Summary - get_discrete_zoom_levels()

This function returns a pointer to an array of desired discrete zoom levels.

Return Values		
Return Value	Type	Meaning
ENVIRON.discrete_zoom_levels	pointer to int	value of discrete_zoom_levels

Table 2.5.4.1-10: Function Summary - get_patches_to_cache()

Return Values		
Return Value	Type	Meaning
ENVIRON.patches_to_cache	pointer to char	value of patches_to_cache

Table 2.5.4.1-11: Function Summary - get_stealth_address()

Return Values		
Return Value	Type	Meaning
&ENVIRON.stealth_address	pointer to SimulationAddress	pointer to stealth_address

Table 2.5.4.1-12: Function Summary - get_db_names(entry_num)

Parameters		
Parameter	Type	Where Typedef Declared
entry_num	int	standard

Return Values		
Return Value	Type	Meaning
&(ENVIRON.database_names [entry_num])	pointer to DB_NAMES	pointer to db_names

2.5.4.2 text.c

This CSU contains routines that control the text display of vehicle states in PVD.

Table 2.5.4.2-1: Function Summary - text_display(world_state)

Parameters		
Parameter	Type	Where Typedef Declared
world_state	pointer to struct WORLD_STATE	world_state.h

Calls	
Function	Where Described
find_closest	
get_screen_size	this file
print_header	this file
display_page_num	this file
move_cursor_to	this file
print_vehicle_id	Pvd.misc.c
out_of_range	Pvd_misc.c
prepare_line	this file
print_vehicle	this file

Table 2.5.4.2-2: Function Summary - clear_screen()

This function clears the screen.

Table 2.5.4.2-3: Function Summary - home()

This function returns the cursor to the upper left-hand corner of the screen.

Table 2.5.4.2-4: Function Summary - move_cursor_to(line, column)

This function moves the cursor to a given line and column on ANSI compatible terminals.

Parameters		
Parameter	Type	Where Typedef Declared
line	int	standard
column	int	standard

Table 2.5.4.2-5: Function Summary - prepare_line(line_number)

This function clears the specified line on the screen and leaves the cursor at the beginning. The first line on the screen is line 1.

Parameters		
Parameter	Type	Where Typedef Declared
line_number	int	standard

Table 2.5.4.2-6: Function Summary - print_header(line_num)

This function prints the column headers for the vehicle display. It does not output a newline.

Parameters		
Parameter	Type	Where Typedef Declared
line_num	int	standard

Calls	
Function	Where Described
prepare_line	this file

Table 2.5.4.2-7: Function Summary - print_vehicle(appearancePDU, statusPDU)

This function prints a one-line description of the passed vehicle's state. Line is not terminated with a newline.

Parameters		
Parameter	Type	Where Typedef Declared
appearancePDU	pointer to SimulationPDU	p_sim.h
statusPDU	pointer to DataCollectionPDU	p_data.h

Calls	
Function	Where Described
kph	this file
get velocity	this file
get turret degrees	this file
get gun degrees	this file
heading	Measures.c
rel turret	measures.c
rel gun	Measures.c
get 4 char type	this file
head_deg	
rad_to_deg	

Table 2.5.4.2-8: Function Summary - kph(vel_array)

This function converts an array of three component velocities (in meters per second) to kilometers per hour.

Parameters		
Parameter	Type	Where Typedef Declared
vel_array	float	standard

Return Values		
Return Value	Type	Meaning
$(3.6 * \sqrt{vel\_array[0]^2 + vel\_array[1]^2 + vel\_array[2]^2})$	float	value of calculation

Table 2.5.4.2-9: Function Summary - display_int_at(row, col, format, value)

This function displays an integer at the specified row and column coordinates. The int is displayed by printf as specified in the passed string.

Parameters		
Parameter	Type	Where Typedef Declared
row	int	standard
col	int	standard
format	pointer to char	standard
value	int	standard

Table 2.5.4.2-10: Function Summary - delete_text_tank(index)

This function clears all lines below a deleted tank.

Parameters		
Parameter	Type	Where Typedef Declared
index	int	standard

Calls	
Function	Where Described
move_cursor to	this file

**Table 2.5.4.2-11: Function Summary - display_page_num
(current_page, total_pages)**

This function displays the current page number in the upper right corner of the screen.

Parameters		
Parameter	Type	Where Typedef Declared
current_page	int	standard
total_pages	int	standard

Calls	
Function	Where Described
move_cursor to	this file

Table 2.5.4.2-12: Function Summary - get_screen_size(lines, cols)

This function returns the number of columns and lines the terminal can display. NOTE: it must handle the graphics screen (tty4) as a special case, since its characteristics are not entered in the environment variable "TERMCAP."

Parameters		
Parameter	Type	Where Typedef Declared
lines	pointer to int	standard
cols	pointer to int	standard

Table 2.5.4.2-13: Function Summary - get_velocity(a_variant)

This function returns the speed of the vehicle in kilometers pers hour.

Parameters		
Parameter	Type	Where Typedef Declared
a_variant	pointer to VehicleAppearanceVariant	p_sim.h

Return Values		
Return Value	Type	Meaning
0.0	float	0
kph	float	specific.simple.velocity
kph	float	specific.tank.velocity

Calls	
Function	Where Described
print_vehicle_id	Pvd_misc.c

Table 2.5.4.2-14: Function Summary - get_turret_degrees(a_variant)

This function returns the direction of the turret with relation to the front of the hull in compass degrees. If the vehicle has no turret, it returns 0.0.

Parameters		
Parameter	Type	Where Typedef Declared
a_variant	pointer to VehicleAppearanceVariant	p_sim.h

Return Values		
Return Value	Type	Meaning
0.0	float	vehicle has no turret
rad_to_deg	float	turret position in compass degrees

Table 2.5.4.2-15: Function Summary - get_gun_degrees(a_variant)

Parameters		
Parameter	Type	Where Typedef Declared
a_variant	pointer to VehicleAppearanceVariant	p_sim.h

Return Values		
Return Value	Type	Meaning
0.0	float	vehicleClassStatic
0.0	float	vehicleClassSimple
rad_to_deg	float	gun elevation

2.5.5 Map/Icon CSC Description

This CSC provides commonly-used Map/Icon utilities. This CSC functionality is broken down into the CSUs shown in Figure 2.5.5-1.

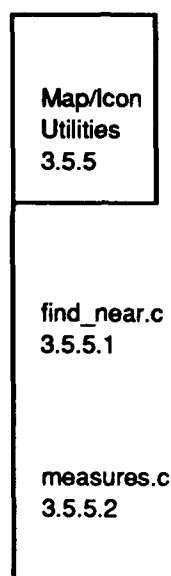


Figure 2.5.5-1: Map/Icon -CSC Structure.

2.5.5.1 find_near.c

This CSU finds the nearest point on a route.

Table 2.5.5.1-1: Function Summary - find_nearest_route_pt(route, point, end_points_only, test_x, test_y)

This function finds the nearest point on a route. It returns a pointer to the route and a pointer to the point. If end_points_only is nonzero, it only considers the two endpoints of each route. A nil route indicates no route found.

Note: triangles are assumed always to be the same size in pixels.

Parameters		
Parameter	Type	Where Typedef Declared
route	pointer to pointer to ROUTE	overlay.h
point	pointer to pointer to ENDPT	overlay.h
end_points_only	int	standard
test_x, test_y	float	standard

Calls	
Function	Where Described
mgrrsi	Concurrent graphics library

2.5.5.2 measures.c

This CSU contains routines that compute tank headings, turret orientations, and gun elevations in compass radians.

Table 2.5.5.2-1: Function Summary - heading(hull_to_world)

This function computes the tank's heading in radians. Heading is expressed in "compass radians", that is, radians measured clockwise from due "north" The returned values range from $-\pi$ to $+\pi$.

Parameters		
Parameter	Type	Where Typedef Declared
hull to world	float	standard

Return Values		
Return Value	Type	Meaning
radians	real	range of $-\pi$ to $+\pi$

Table 2.5.5.2-2: Function Summary - heading_fraction(hull_to_world)

This function returns the heading of the vehicle as a 32-bit binary fraction of a circle (0 is N, increasing clockwise.)

Parameters		
Parameter	Type	Where Typedef Declared
hull to world	float	standard

Table 2.5.5.2-3: Function Summary - rel_turret(turret_az)

This function returns the orientation of the turret with relation to the hull in compass radians. It returns a value in the range 0 to $2 * \pi$.

Parameters		
Parameter	Type	Where Typedef Declared
turret az	unsigned long	standard

Return Values		
Return Value	Type	Meaning
$2.0 * \pi - ((\text{float})\text{turret\_az}) / (\text{TWO TO THE } 32) * 2.0 * \pi$	real	range of 0 to $2 * \pi$

Table 2.5.5.2-4: Function Summary - rel_gun(gun_el)

This function returns the elevation of the gun above the hull plane in radians. It returns a value in the range -PI to PI.

Parameters		
Parameter	Type	Where Typedef Declared
gun_el	unsigned long	standard

Table 2.5.5.2-5: - Function Summary - angle_normalize(rads)

This functions puts an angle in radians within the range -PI to PI.

Parameters		
Parameter	Type	Where Typedef Declared
rads	float	standard

Return Values		
Return Value	Type	Meaning
rads	float	range of - pi to +pi

Table 2.5.5.2-6: Function Summary - dist2(from_x, from_y, to_x, to_y)

This function returns a two-dimensional distance.

Parameters		
Parameter	Type	Where Typedef Declared
from x, from y	float	standard
to x, to y	float	standard

Return Values		
Return Value	Type	Meaning
sqrt(delta_x * delta_x + delta_y * delta_y)	float	two-dimensional distance

Table 2.5.5.2-7: Function Summary - dist_3(x0, y0, z0, x1, y1, z1)

This function returns a three-dimensional distance.

Parameters		
Parameter	Type	Where Typedef Declared
x0, y0, z0	float	standard
x1, y1, z1	float	standard

Return Values		
Return Value	Type	Meaning
sqrt(dx * dx + dy * dy + dz * dz)	float	three-dimensional distance

Table 2.5.5.2-8: Function Summary - cmc_msec(sec)

This function converts seconds to cmc "milliseconds."

Parameters		
Parameter	Type	Where Typedef Declared
sec	int	standard

Return Values		
Return Value	Type	Meaning
(sec * 1000)	int	milliseconds

2.5.6 Debug CSC Description

This CSC provides debugging routines.

This CSC functionality is broken down into the CSUs shown in Figure 2.5.6-1.

**Figure 2.5.6-1: Utilities--Debug CSC Structure.**

2.5.6.1 memory.c

This CSU contains routines for determining the amount of free memory available.

Table 2.5.6.1-1: Function Summary - get_all_memory(last_chunk)

This function returns an estimate of the total amount of memory available, and, in 'last_chunk', a tolerance on this estimate. It attempts to allocate all of dynamic memory in order to see how much there is.

Parameters		
Parameter	Type	Where Typedef Declared
last_chunk	pointer to int	standard

Return Values		
Return Value	Type	Meaning
sum	int	Number of bytes in last chunk of memory allocated; 0 means got it all

Table 2.5.6.1-2: Function Summary - get_biggest_memory_chunk(chunk)

This function returns the size in bytes of the single largest chunk of memory that could be malloc'ed. The int pointed to by the argument is set to the address of this chunk.

Parameters		
Parameter	Type	Where Typedef Declared
chunk	pointer to int	standard

Return Values		
Return Value	Type	Meaning
current	int	Number of bytes of largest chunk of memory that can be malloced.
0	int	error

2.5.6.2 tell.c

This CSU displays in the event window the screen coordinates of the map window, map coordinates, and mouse locations. (Used for debugging purposes.)

Table 2.5.6.2-1: Function Summary - tell_all()

Calls	
Function	Where Described
mgigetwloc	Concurrent graphics library subroutine
mgrgetwloc	Concurrent graphics library subroutine

Table 2.5.6.2-2: Function Summary - tell_real()

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine

Table 2.5.6.2-3: Function Summary - tell_integer()

Calls	
Function	Where Described
mgigetvcoor	Concurrent graphics library subroutine

2.6 NETWORK PROCESSING

This CSC initializes and sends packets based on user inputs. It also receives, filters, and stores information from vehicle appearance packets into PVD shared memory. This CSC detects and eliminates vehicles that have timed out, and accesses basic network services.

This CSC is designed to hide the details of network communications from other CSC functions. This CSC functionality is broken down into the lower-level CSCs shown in Figure 2-6.

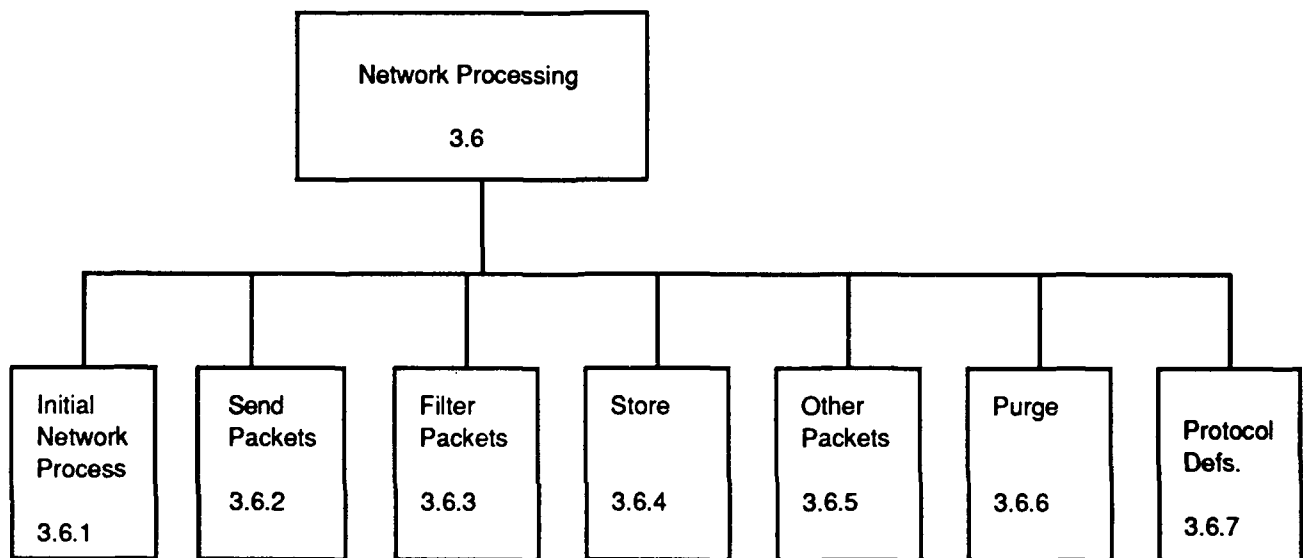


Figure 2-6: Network Processing CSC Structure.

2.6.1 Initial Network Process CSC Description

This CSC provides the main entry for network processing.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.1-1.

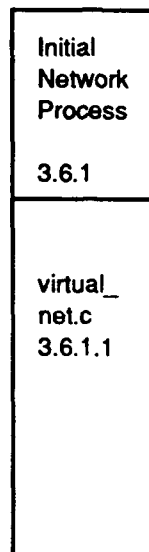


Figure 2.6.1-1: Network Processing—Initial Network Process CSC Structure.

2.6.1.1 virtual_net.c

This CSU involves a network interface. It contains functions that initialize and receive packets from the simulation network via the association layer (libassoc.a.)

Table 2.6.1.1-1: Function Summary - init_virtual_net()

This function initializes a connection to the network through the association layer.

Calls	
Function	Where Described
table_init	
net_alive	libnetif.a in MCC CSCI

Table 2.6.1.1-2: Function Summary - get_next_packet(buf_ptr, length, group, protocol, originator)

This function gets the next packet from the association layer.

Parameters		
Parameter	Type	Where Typedef Declared
buf_ptr	pointer to pointer to char	standard
length	pointer to int	standard
group	pointer to MulticastGroupID	p_assoc.h
protocol	pointer to AssociationUserProtocol	p_assoc.h
originator	pointer to SimulationAddress	address.h

Return Values		
Return Value	Type	Meaning
1	int	data available
0	int	no more packets in the rings; or read failed

Calls	
Function	Where Described
PointToPointReceivePDU	see libassoc in Vehicle Simulation CSCI
get_next_packet	this file

2.6.2 Send Packets CSC Description

This CSC initializes and sends packets based on menu selections.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.2-1.

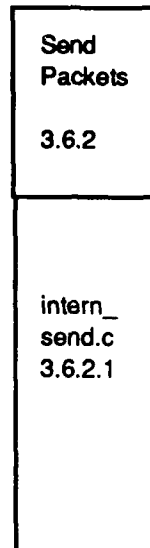


Figure 2.6.2-1: Network Processing--Send Packets CSC Structure.

2.6.2.1 intern_send.c

This CSU contains routines for simulating sending packets to oneself.

Table 2.6.2.1-1: Function Summary - fake_deactivate(vehicle_id)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	int	standard

Calls	
Function	Where Described
get world state	
tank number exists	

Table 2.6.2.1-2: Function Summary - init_internal_ring_buffers()

Calls	
Function	Where Described
initialize_ptr_array	

Table 2.6.2.1-3: Function Summary - send_buffer_to_self(net_buffer)

Parameters		
Parameter	Type	Where Typedef Declared
net_buffer	pointer to NetworkBuffer	network.h

Calls	
Function	Where Described
fake_filter	this file
enqueue_buffer_into_fake_ring	this file

Table 2.6.2.1-4: Function Summary - fake_filter(buffer)

Fake filter simulates the effect of CMC crad filtering on exercise ID for those buffers which come from some other source than the network (e.g., the internal CATC functions.)

Parameters		
Parameter	Type	Where Typedef Declared
buffer	pointer to NetworkBuffer	network.h

Return Values		
Return Value	Type	Meaning
1	int	accept all packets in this mode
0	int	accept all non-Sim packets

Calls	
Function	Where Described
get_exercise	Pvd_misc.c

Table 2.6.2.1-5: Function Summary - enqueue_buffer_into_fake_ring (buffer)

This function copies buffer into a fake ring buffer for later processing.

Parameters		
Parameter	Type	Where Typedef Declared
buffer	pointer to char	standard

Return Values		
Return Value	Type	Meaning
0	int	successful
1	int	otherwise

Table 2.6.2.1-6: Function Summary - process_internal_ring_buffer()

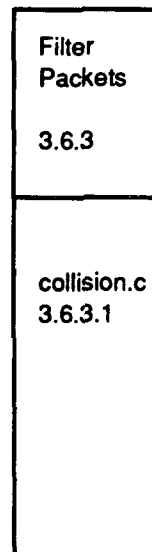
This function processes buffers form the internal ring buffer.

Calls	
Function	Where Described
CollectWorldState	Pvd_misc.c
get_world_state	

2.6.3 Filter Packets CSC Description

This CSC receives and filters packets based on menu selections.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.3-1.

**Figure 2.6.3-1: Network Processing--Filter Packets CSC Structure.**

2.6.3.1 collision.c

This CSU handles collision PDUs. When a collision occurs, it displays information in the event window, such as the vehicle IDs of the involved vehicles.

Table 2.6.3.1-1: Function Summary - do_collision(coll_variant)

Parameters		
Parameter	Type	Where Typedef Declared
coll_variant	pointer to CollisionVariant	p_sim.h

Calls	
Function	Where Described
set_text_window	T_windows.c
print_vehicle_id	Pvd.misc.c

2.6.4 Store CSC Description

This CSC stores status information in shared memory.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.4-1. Additionally, this CSC makes use of the CSU, libshm.a (Section 2.4.3.3), and pvd_misc.c. (Section 2.6.5.1).

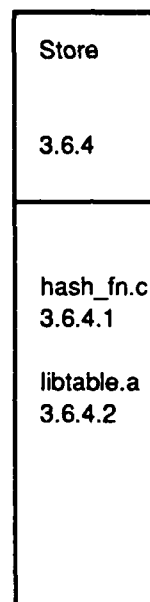


Figure 2.6.4-1: Network Processing--Store CSC Structure.

2.6.4.1 hash_fn.c

This CSU executes the hash function used in the tabling code. It stores the numbers used to access vehicle appearance packets.

Table 2.6.4.1-1: Function Summary - Hash_Vehicle_ID(vehicle_id)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to register unsigned short	standard

Return Values		
Return Value	Type	Meaning
index	int	index to hash table

Table 2.6.4.1-2: Function Summary - matching_vehicle_IDs (vehicle_id, pdu)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to VehicleID	basic.h
pdu	pointer to SimulationPDU	p_sim.h

Return Values		
Return Value	Type	Meaning
equal vehicle IDs	int	vehicleIDs are equal

Calls	
Function	Where Described
equal_vehicle_IDs	this file

Table 2.6.4.1-3: Function Summary - equal_vehicle_IDs (vehicle_id1, vehicle_id2)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id1	pointer to VehicleID	basic.h
vehicle_id2	pointer to VehicleID	basic.h

Return Values		
Return Value	Type	Meaning
1	int	passed vehicleIDs are equal
0	int	other

Table 2.6.4.1-4: Function Summary - get_vehicle_index(id)

This function returns the vehicle index corresponding to the passed vehicle id. It returns -1 if there is no such vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
id	pointer to register VehicleID	basic.h

Return Values		
Return Value	Type	Meaning
table_entry	int	vehicle index corresponding to passed vehicle id
-1	int	no such vehicle

Calls	
Function	Where Described
Hash Vehicle ID	this file
table_get	

2.6.4.2 libtable.a

This is a common library shared by PVD and other SIMNET applications.

2.6.5 Other Packets CSC Description

This CSC routes filtered packets to other CSC functions.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.5-1.

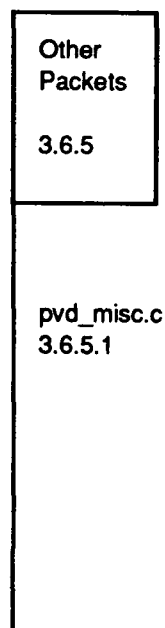


Figure 2.6.5-1: Network Processing--Other Packets CSC Structure.

2.6.5.1 pvd_misc.c

This CSU stores status packets, and dispatches functions to handle the various other PDUs relevant to the PVD. The key function in this file is Collect_World_State.

Table 2.6.5.1-1: Function Summary - Collect_World_State(buf, length, group, protocol, originator, world_state)

Parameters		
Parameter	Type	Where Typedef Declared
buf	pointer to register char	standard
length	long int	standard
group	MultiCastGroupID	p_assoc.h
protocol	AssociationUserProtocol	p_assoc.h
originator	pointer to SimulationAddress	address.h
world_state	pointer to register struct WORLD_STATE	world_state.h

Calls	
Function	Where Described
construct sim from stealth	Place.c
matching vehicle IDs	Hash fn.c
print simulation address	this file
current exercise	
Hash Vehicle ID	Hash fn.c
table get	
print vehicle id	this file
make new vehicle index	this file
PRO SIM APPEARANCE SIZE	
table insert	
spin	this file
net_gettime	libnetif.a in MCC CSCI
create icon	
do impact	Dir fire.c
do direct fire	Indir fire.c
do collision	Collision.c
get vehicle index	Hash fn.c
purge vehicle	Purge.c
process minefield marker pdu	
process breached lane pdu	
process minefield pdu	
print data header	
display status change	Stat change.c
modify last status packet	Pvd misc.c
lase	Lase.c
display event flag	Pvd misc.c
handle terrain protocol	
handle log rtc	Pvd rtc.c
handle stealth error	Place.c
handle attached packet	Place.c

Table 2.6.5.1-2: Function Summary - UpdateText_Display(world_state)

Parameters		
Parameter	Type	Where Typedef Declared
world_state	pointer to struct WORLD_STATE	world_state.h

Return Values		
Return Value	Type	Meaning
Text_Display	int	value of world_state

Table 2.6.5.1-3: Function Summary - out_of_range(index, bound)

Parameters		
Parameter	Type	Where Typedef Declared
index	int	standard
bound	int	standard

Return Values		
Return Value	Type	Meaning
1	int	passed array index is outside the specified bounds
-1	int	0 to bound
0	int	otherwise

Table 2.6.5.1-4: Function Summary - init_ascii()

This function sets up an ASCII terminal.

Calls	
Function	Where Described
clear_screen	Text.c
home	Text.c

Table 2.6.5.1-5: Function Summary - make_new_vehicle_index()

This function is called only when a new vehicle is being created.

Return Values		
Return Value	Type	Meaning
EXISTING_tanks++	int	returns new_vehicle_index

Table 2.6.5.1-6: Function Summary - refresh_timestamps()

This function sets the time of the last packet received for all existing tanks to the current time. For example, it is used after updates have been turned off during a logger freeze.

Calls	
Function	Where Described
get world state	
net_gettime	libnetif.a in MCC CSCI

Table 2.6.5.1-7: Function Summary - clear_net_input()

This function marks all routines for erasure and clears buffers of packets waiting for processing.

Calls	
Function	Where Described
get world state	
net_flush	libnetif.a in MCC CSCI

Table 2.6.5.1-8: Function Summary - spin()

This function wastes cycles, for example, while waiting for the CMC card to clear a flag in shared memory.

Return Values		
Return Value	Type	Meaning
dilatory - dilatory	int	wastes time

Table 2.6.5.1-9: Function Summary - get_spin_count()

Return Values		
Return Value	Type	Meaning
spin count	int	number of times spin was called

Table 2.6.5.1-10: Function Summary - init_text_exercises()

This function sets up the card to read all exercises.

Calls	
Function	Where Described
AssocSubscribe	libassoc.a in MCC CSCI

Table 2.6.5.1-11: Function Summary - set_exercise(exercise_id)

This function sets the exercise ID to be viewed. Valid ID numbers are 1 to 255.

Parameters		
Parameter	Type	Where Typedef Declared
exercise_id	int	standard

Calls	
Function	Where Described
AssocUnsubscribe	Protocol
AssocSubscribe	Protocol
set stealth exercise	
clear net input	Pvd misc.c

Table 2.6.5.1-12: Function Summary - current_exercise(exercise_id)

This function returns 1 if the passed exercise ID is acceptable, given what exercises are being accepted.

Parameters		
Parameter	Type	Where Typedef Declared
exercise_id	int	standard

Return Values		
Return Value	Type	Meaning
0	int	Exercise ID not current exercise
1	int	Exercise ID = current exercise

Table 2.6.5.1-13: Function Summary -get_exercise()

This function returns the exercise currently being displayed.

Return Values		
Return Value	Type	Meaning
CURRENT_exercise	int	pointer to CURRENT_exercise

Table 2.6.5.1-14: Function Summary - get_vehicle_exercise(vehicle_id)

This function assumes vehicle_id is valid. Given a valid id, it returns the exercise.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to VehicleID	basic.h

Return Values		
Return Value	Type	Meaning
world_state->v_appearances[] ->exercise	int	returns exercise

Calls	
Function	Where Described
get_world_state	

Table 2.6.5.1-15: Function Summary - get_dynamic_appearance_data (vehicle_index, guises, appearance, location, velocity)

This function fills in data for key changing components of an appearance packet.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_index	int	standard
guises	Pointer to VehicleGuises	basic.h
appearance	Pointer to unsigned long	standard
location	Pointer to float	standard
velocity	Pointer to float	standard

Calls	
Function	Where Described
get_world_state	

Table 2.6.5.1-16: Function Summary - print_vehicle_id(vehicle_id)

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	Pointer to VehicleID	basic.h

Calls	
Function	Where Described
fprint_vehicle_id	this file

Table 2.6.5.1-17: Function Summary - fprint_vehicle_id(fd, vehicle_id)

This function prints the vehicle id.

Parameters		
Parameter	Type	Where Typedef Declared
fd	Pointer to FILE	stdio.h
vehicle_id	Pointer to VehicleID	basic.h

**Table 2.6.5.1-18: Function Summary - print_simulation_address
(simulation_address)**

This function calls fprintf_simulation_address.

Parameters		
Parameter	Type	Where Typedef Declared
simulation_address	Pointer to SimulationAddress	address.h

Calls	
Function	Where Described
fprintf_simulation_address	this file

**Table 2.6.5.1-19: Function Summary - fprintf_simulation_address
(fd, simulation_address)**

This function prints the Site name and Host name.

Parameters		
Parameter	Type	Where Typedef Declared
fd	Pointer to FILE	stdio.h
simulation_address	Pointer to SimulationAddress	address.h

Table 2.6.5.1-20: Function Summary - display_event_flag(e_variant)

This function prints the data from an event flag packet in the event window.

Parameters		
Parameter	Type	Where Typedef Declared
e_variant	Pointer to EventFlagVariant	p_data.h

Calls	
Function	Where Described
set_text_window	T_windows.c
print_vehicle_id	this file

**Table 2.6.5.1-21: Function Summary - modify_last_status_packet
(status_change)**

This function updates the last status packet for the vehicle to account for the change in status reported by the passed Status Change PDU.

Parameters		
Parameter	Type	Where Typedef Declared
status_change	Pointer to StatusChangeVariant	p_data.h

2.6.6 Purge CSC Description

This CSC purges vehicles based on time stamp packet information.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.6-1.

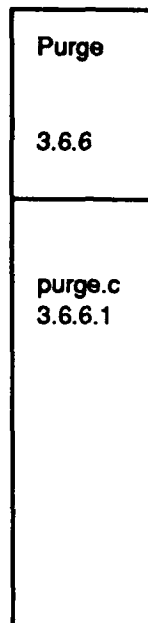


Figure 2.6.6-1: Network Processing--Purge CSC Structure.

2.6.6.1 purge.c

This CSU deletes all tanks that have not broadcast an appearance packet within the time-out period. It remaps the globally available list of currently selected tanks to account for the deletion of a tank. It also adjusts the list of tanks tagged for vehicle to vehicle intervisibility display to account for evaporation (due to cessation of appearance packets) of a tank.

Table 2.6.6.1-1: Function Summary - remap_selected_tanks(dead_id)

This function remaps the globally available linked list of currently selected tanks to account for the deletion of a tank ID.

Parameters		
Parameter	Type	Where Typedef Declared
dead id	int	standard

Table 2.6.6.1-2: Function Summary - remap_tagged_tanks(dead_id)

This function adjusts the list of tanks tagged for vehicle-to-vehicle intervisibility display to account for the evaporation of a tank due to the cessation of appearance packets.

Parameters		
Parameter	Type	Where Typedef Declared
dead_id	int	standard

Calls	
Function	Where Described
get tags	Vehicle to v.c
erase rays	Vehicle to v.c
set tags	Vehicle to v.c

Table 2.6.6.1-3: Function Summary - purge_expired_vehicles(time)

This function updates the timestamps on all vehicles and purges any vehicles that have not received an appearance packet within the time-out period.

Parameters		
Parameter	Type	Where Typedef Declared
time	int	standard

Calls	
Function	Where Described
get world state	
print vehicle id	Pvd misc.c
purge vehicle	this file

Table 2.6.6.1-4: Function Summary - set_time_out_msec(milli_sec)

This function determines the number of milliseconds before a vehicle times out. This should normally be set to protocolVehicleDisappearanceTime * 1000.

Parameters		
Parameter	Type	Where Typedef Declared
milli_sec	int	standard

Table 2.6.6.1-5: Function Summary - purge_vehicle(vehicle_index)

This function eliminates a vehicle that has timed out or has been deactivated.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_index	int	standard

Calls	
Function	Where Described
get world state	
table delete	
print vehicle id	Pvd_misc.c
spin	this file
table free	
delete icon	Icon.c
remap selected tanks	this file
remap tagged tanks	this file
delete text tank	Text.c
delete vehicle id	

2.6.7 Protocol Definitions CSC Description

This CSC provides network protocol definitions.

This CSC functionality is broken down into the CSUs shown in Figure 2.6.7-1.

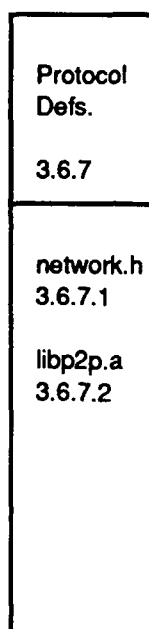


Figure 2.6.7-1: Network Processing—Protocol Definitions CSC Structure.

2.6.7.1. network.h

This is a commonly-maintained file containing network protocol definitions used by Plan View Display and other SIMNET applications.

2.6.7.2 libp2p.a

This is a common library shared by PVD and other SIMNET applications.

2.7 GRAPHICS CSC DESCRIPTION

This CSC provides graphics support for graphics input and display.

This CSC functionality is broken down into the CSCs shown in Figure 2-7.

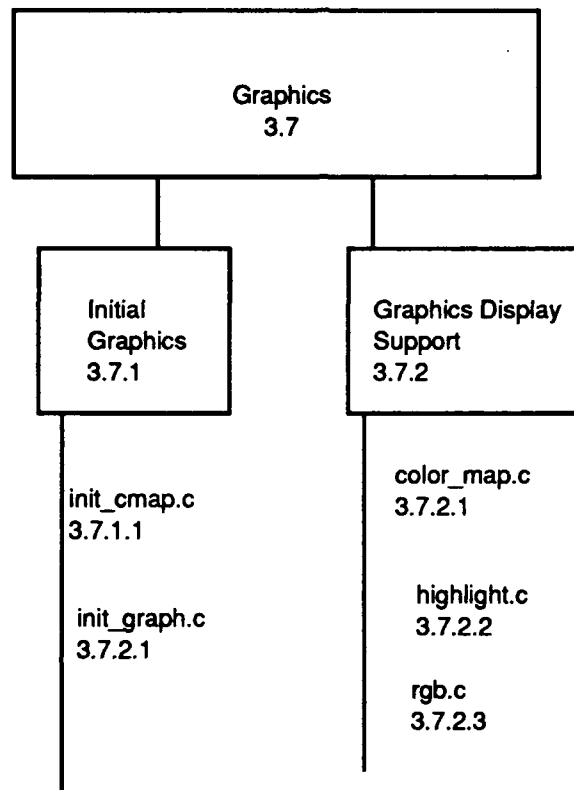


Figure 2-7: Graphics CSC Structure.

2.7.1 Initial Graphics

This CSC initializes graphics support.

This CSC functionality is broken down into the CSUs shown in Figure 2.7.1-1.

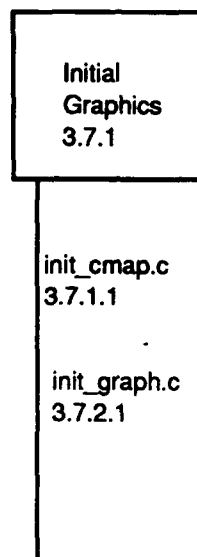


Figure 2.7.1-1: Graphics--Initial Graphics CSC Structure.

2.7.1.1 init_cmap.c

This CSU initializes the terrain color map. There are 15 color slots to assign; black is hardwired.

Table 2.7.1.1-1: Function Summary - flag_map_colors()

This function initializes the map_colors.

Table 2.7.1.1-2: Function Summary - init_terrain_cmap(shift)

Parameters		
Parameter	Type	Where Typedef Declared
shift	int	standard

Calls	
Function	Where Described
rgb	Rgb.c
mgfcns	Concurrent graphics library
make color map	Color map.c

Table 2.7.1.1-3: Function Summary -mapcolor(index, red, green, blue, shift)

Parameters		
Parameter	Type	Where Typedef Declared
index	int	standard
red, green, blue	int	standard
shift	int	standard

Calls	
Function	Where Described
rgb	rgb.c

2.7.1.2 init_graph.c

This CSU initializes Plan View Display on a graphics terminal.

Table 2.7.1.2-1: Function Summary - init_graph()

This function initializes Plan View Display on a graphics terminal.

Calls	
Function	Where Described
mgiasngp	Concurrent graphics library subroutine
mgixitmode	Concurrent graphics library subroutine
mgiclearpln	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
mgidefw	Concurrent graphics library subroutine
mgipw	Concurrent graphics library subroutine
count bits set	Finite_zoom.c
get shift	Finite_zoom.c
init_paint system	new_zoom.c
init_working_cursor	Pvd_iface.c
mgifb	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgiocm	Concurrent graphics library subroutine
init_terrain_cmap	Init_cmap.c
rgb	Rgb.c
make_color_map	Color_map.c
init_window_colors	init_windows.c
init_icon_colors	Icon.c
mgrvcoor	Concurrent graphics library subroutine
init_pixel_map_descriptors	New_zoom.c
init_user_interface	Pvd_iface.c
init_working_cursor	Pvd_iface.c
init_icons	Icon.c

(Table 2.7.1.2-1 is continued on the following page.)

Function	Where Described
init bursts	indir fire.c
init effects	effects.c
toggle shade	Unshade.c
display top map	Display_top.c
add view	Prev view.c
init working cursor	Pvd iface.c
init popup	Pop windows.c
init discrete zoom	Finite zoom.c
init cursor	Pvd iface.c

2.7.2 Graphics Display Support

This CSC supports graphics display.

This CSC functionality is broken down into the CSUs shown in Figure 2.7.2-1.

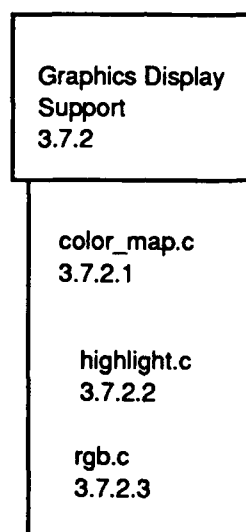


Figure 2.7.2-1: Graphics--Graphics Display Support CSC Structure.

2.7.2.1 color_map.c

This CSU contains routines for creating the color map for Plan View Display, including overlaying graphics windows and adding tank icons, trails, contour lines, and terrain.

Table 2.7.2.1-1: Function Summary - make_color_map(plane_mask, RGBvals)

This routine determines which type of entry—icons, contour lines, or terrain—is being made to the color map.

Parameters		
Parameter	Type	Where Typedef Declared
plane_mask	int	standard
RGBvals	long	standard

Calls	
Function	Where Described
make_terrain_colors	this file
make_contour_colors	this file
make_icon_colors	this file
make_window_colors	this file

Table 2.7.2.1-2: Function Summary - make_window_colors(RGBvals)

This function assigns the passed window colors to the lower two planes of the color map. Color 00B is always transparent.

Parameters		
Parameter	Type	Where Typedef Declared
RGBvals	long	standard

Calls	
Function	Where Described
mgicm	Concurrent graphics library

Table 2.7.2.1-3: Function Summary - make_icon_colors(RGBvals)

This function assigns all entries with lower two bits equal to 00B (i.e., transparent) and those with the next three bits not equal to 000B.

Parameters		
Parameter	Type	Where Typedef Declared
RGBvals	long	standard

Calls	
Function	Where Described
mgicm	Concurrent graphics library

Table 2.7.2.1-4: Function Summary - make_terrain_colors(RGBvals)

This function assigns all the entries of the color map with the correct upper four bits to the color specified by the RGBvals entry. The entry has the lower six bits as 000000B. Otherwise the colors specified by the lower six bits will prevail.

Parameters		
Parameter	Type	Where Typedef Declared
RGBvals	long	standard

Calls	
Function	Where Described
mgicm	Concurrent graphics library

Table 2.7.2.1-5: Function Summary - make_contour_colors(RGBvals)

This function assigns all the entries of the color map with the lower five bits set to 00000B and the sixth bit set to 1 to the color specified by the RGBvals entry. The entry with the lower six bits as 000000B allows control to go to the upper four bits. Otherwise, the colors specified by the lower six bits will prevail.

Parameters		
Parameter	Type	Where Typedef Declared
RGBvals	long	standard

Calls	
Function	Where Described
mgicm	Concurrent graphics library

2.7.2.2 highlight.c CSU Description (/simnet/pvd/lib/pvdiface.a)

This CSU highlights the selected tank icon.

Table 2.7.2.2-1: Function Summary - highlight_tank(index)

This function prints the tank index to be highlighted.

Parameters		
Parameter	Type	Where Typedef Declared
index	int	standard

Calls	
Function	Where Described
highlight selected icon	Icon.c

Table 2.7.2.2-2: Function Summary - unhighlight_tank(index)

This function prints the tank index to be unselected.

Parameters		
Parameter	Type	Where Typedef Declared
index	int	standard

Calls	
Function	Where Described
unhighlight_selected_icon	Icon.c

2.7.2.3 rgb.c

This CSU converts 4 bit (0 to 15) RGB color intensities into values suitable for insertion into a color map.

Table 2.7.2.3-1: Function Summary - rgb(red, green, blue)

Parameters		
Parameter	Type	Where Typedef Declared
red, green, blue	int	standard

Return Values		
Return Value	Type	Meaning
color map entry	int	32-bit integer value

2.8 OVERLAYS CSC DESCRIPTION

This CSC provides for the creation of map overlays, consisting of military unit symbols, free draw objects, and control points. This CSC also provides menu options for specifying these map features—contour, pan, zoom, and grid. This CSC supports saving and retrieving overlays in files.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-8.

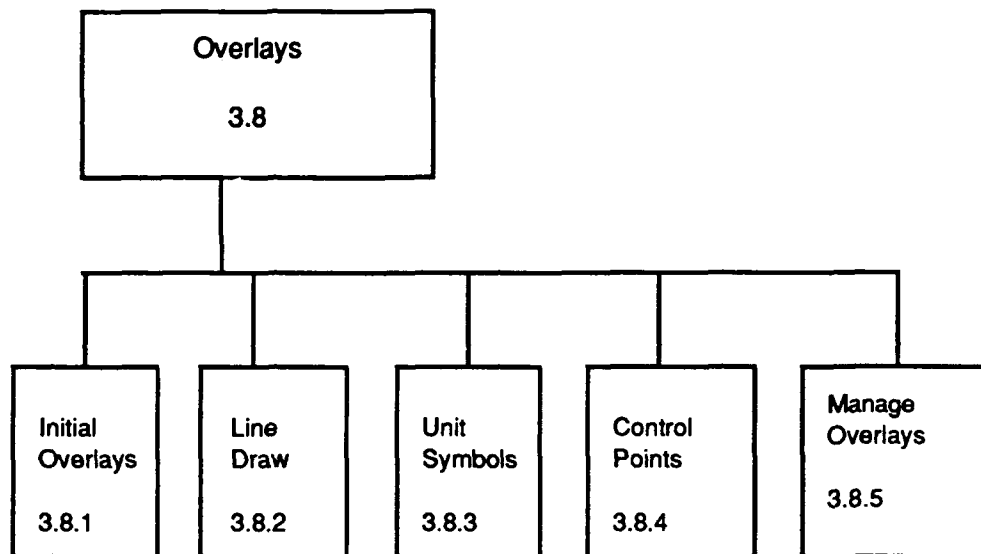


Figure 2-8: Overlays CSC Structure.

2.8.1 Initial Overlays CSC Description

This CSC provides the main entry for Overlays.

This CSC functionality is broken down into the CSUs shown in Figure 2.8.1-1.

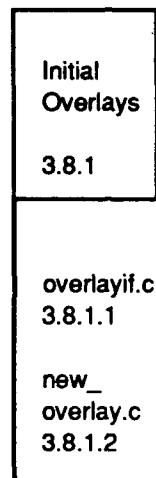


Figure 2.8.1-1: Overlays--Initial Overlays CSC Structure.

2.8.1.1 overlayif.c

This CSU contains routines that handle common overlay menu functions.

Table 2.8.1.1-1: overlay_mouse()

This function handles mouse events while the PVD is in Overlay Mode. The right button chooses a menu option; the center button selects an object ; and the left button prints information about an object.

Return Values		
Return Value	Type	Meaning
1	int	OverlaysOn

Calls	
Function	Where Described
handle_menu	Menu.c
select_overobj	This file
select_region	

Table 2.8.1.1-2: Function Summary - zoom_overlay()

This function redraws everything that was created through the Overlays Menu and is now included in the Overlay Table.

Calls	
Function	Where Described
get shift	Finite zoom.c
calc zoom ratio	New overlay.c
zoom overline	Overline.c
zoom controlpt	Controlpt.c
zoom symbol	symbol.c

Table 2.8.1.1-3: Function Summary - erase_overlay()

This function erases all the items currently active in the object tables. It does not clear the object tables.

Calls	
Function	Where Described
calc zoom ratio	New overlay.c
erase overline	Overline.c
erase controlpt	Controlpt.c
erase symbol	symbol.c
mgrsi	Concurrent graphics library subroutine
init working cursor	Pvd iface.c
unified paint square region	New zoom.c
init cursor	Pvd iface.c

Table 2.8.1.1-4: Function Summary - read_over()

This function loads and displays an overlay in the permanent set of planes.

Calls	
Function	Where Described
reset overlay	this file
get overlay	this file
draw overlay menu	Menu.c

Table 2.8.1.1-5: Function Summary - get_overlay()

This function prompts the user for a file, then loads and displays the overlay objects contained in that file.

Calls	
Function	Where Described
load overlay	Over files c
zoom overlay	This file

Table 2.8.1.1-6: Function Summary - save_over()

This function saves the current overlay.

Calls	
Function	Where Described
get_typed_input_str	interact.c
save_overline	Over_files.c
save_controlpt	Over_files.c
save_symbol	Over_files.c

Table 2.8.1.1-7: Function Summary - delete_over_file()

This function deletes overlay files.

Table 2.8.1.1-8: Function Summary - edit_over()

This function loads and displays an overlay in the EDIT set of planes.

Calls	
Function	Where Described
reset_overlay	This file
get_overlay	this file
draw_overlay_menu	menu.c

Table 2.8.1.1-9: Function Summary - reset_overlay()

This function erases all the items currently active in the object tables and clears the object tables.

Calls	
Function	Where Described
calc_zoom_ratio	New_overlay.c
erase_overline	Overline.c
erase_controlpt	Controlpt.c
erase_symbol	symbol.c
clear_overlays	New_overlay.c
mgrsi	Concurrent graphics library subroutine
init_working_cursor	Pvd_iface.c
unified_paint_square_region	New_zoom.c
init_cursor	Pvd_iface.c

Table 2.8.1.1-10: Function Summary - select_overobj()

This function tries to select the object currently pointed at by the mouse. It first checks for unit symbols, then for free draw lines, and finally for control points. This ordering affects selection of overlapping objects, i.e., it is possible NOT to select an object due to the proximity of higher priority objects.

Calls	
Function	Where Described
uncolor_selected	symbolif.c
uncolor_controlpt	Controlptif.c
select_overline	Overlineif.c
select_controlpt	Controlptif.c
string_error_box	Error_box.c
kill_error_window	Error_box.c

Table 2.8.1.1-11: Function Summary - delete_overobj()

This function deletes overlay objects.

Calls	
Function	Where Described
controlpt_delete	Controlptif.c
overline_delete	Overlineif.c
symbol_delete	Symbolif.c

Table 2.8.1.1-12: Function Summary - reduce_overobj()

This function resizes selected overlay objects.

Calls	
Function	Where Described
controlpt_reduce	Controlptif.c
symbol_reduce	Symbolif.c

Table 2.8.1.1-13: Function Summary - move_overobj()

This function moves selected overlay objects.

Calls	
Function	Where Described
move_controlpt	Controlptif.c
symbol_move	Symbolif.c

Table 2.8.1.1-14: Function Summary - addtext_overobj()

This function adds text to a selected overlay object.

Calls	
Function	Where Described
controlpt_addtext	Controlptif.c
overline_addtext	Overlineif.c
symbol_addtext	Symbolif.c

Table 2.8.1.1-15: Function Summary - init_erasing_cursor()

This function changes the cursor into a unified square.

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library

Table 2.8.1.1-16: Function Summary - init_drawing_cursor()

This function changes the cursor into a solid box.

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library subroutine

Table 2.8.1.1-17: Function Summary - init_size_cursor()

This function changes the cursor into a small filled square.

Calls	
Function	Where Described
mgiloadcurs	Concurrent graphics library

Table 2.8.1.1-18: Function Summary - gen_add_text(Text, Window, Planes, Hue)

Parameters		
Parameter	Type	Where Typedef Declared
Text	pointer to SYMLABEL	symbol.h
Window	int	standard
Planes	int	standard
Hue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
add free text	Overdraw.c
get_string	this file
mggetcursmode	Concurrent graphics library subroutine
mgicursmode	Concurrent graphics library subroutine
display_cursor loc	
mgimodfunc	Concurrent graphics library subroutine
move_mouse help	
mgigetvcoor	Concurrent graphics library subroutine
pixel to user	Pvd_iface.c
default_mouse help	Pvd_windows.c

Table 2.8.1.1-19: Function Summary - get_string(StrInput)

This routine displays a prompt for the user to input a string which is then stored in StrInput.

Parameters		
Parameter	Type	Where Typedef Declared
StrInput	pointer to char	standard

2.8.1.2 new_overlay.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines that handle common overlay functions.

Table 2.8.1.2-1: Function Summary - overlay_init()

This routine initializes all overlay tables.

Calls	
Function	Where Described
symbol init	symbol.c
overline init	Overline.c
controlpt init	Controlpt.c
init over files	Over_files.c

Table 2.8.1.2-2: Function Summary - display_overlays(Planes, PencilHue, FoeHue, FriendHue, ClearHue)

This routine redraws everything that was created through the OVERLAYS menu and is now included in the Overlay Table.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
PencilHue	int	standard
FoeHue	int	standard
FriendHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
calc_zoom_ratio	this file
zoom_overline	Overline.c
zoom_controlpt	Controlpt.c
zoom_symbol	symbol.c

Table 2.8.1.2-3: Function Summary - clear_overlays()

This routine clears the contents of the Overlay Tables.

Calls	
Function	Where Described
overline_reset	Overline.c
controlpt_reset	Controlpt.c
symbol_reset	symbol.c

Table 2.8.1.2-4: Function Summary - calc_zoom_ratio()

This function is called when symbols are about to be drawn onto the ICON_PLANES.

Calls	
Function	Where Described
mgrgetvcoor	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine

Table 2.8.1.2-5: Function Summary - over_user_to_pixel(px, py, rx, ry)

Parameters		
Parameter	Type	Where Typedef Declared
px, py	pointer to int	standard
rx, ry	float	standard

Calls	
Function	Where Described
mgrscalexy	Concurrent graphics library subroutine

Table 2.8.1.2-6: Function Summary - over_point_in_box(x, y, box)

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard
box	pointer to struct RECTANGULAR REGION	pop_win.h

Return Values		
Return Value	Type	Meaning
1	int	point is in box
0	int	point is not in box

2.8.2 Line Draw CSC Description

This CSC supports free drawing options for use with lines and boundaries such as phase lines and defended areas.

This CSC functionality is broken down into the CSUs shown in Figure 2.8.2-1.

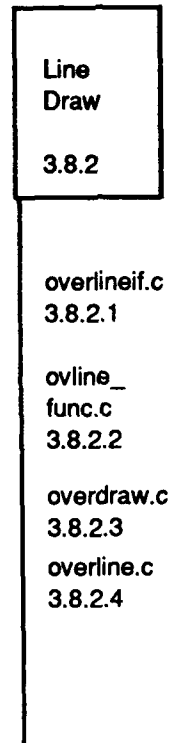


Figure 2.8.2-1: Overlays—Line Draw CSC Structure.

2.8.2.1 overlineif.c

This CSU contains high-level functions that call routines for displaying the popup window and managing the menu choices for free drawing.

Table 2.8.2.1-1: Function Summary – overline_create()

This function displays the pop_window for creating an overline.

Calls	
Function	Where Described
mouse to option	option.c
del line window	this file
draw line window	this file
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
display_block	option.c
mgihue	Concurrent graphics library subroutine
mark box	option.c
phase action	ovline func.c
solid action	ovline func.c
handle popup windows	pop windows.c
init drawing cursor	Overlayif.c
freedraw mouse help	Pvd windows.c
calc zoom ratio	New overlay.c
display cursor loc	
mgigetvcoor	Concurrent graphics library subroutine
overline loop	this file
insert overline	Overline.c
overline features	Overdraw.c
size loop	this file
init cursor	Pvd iface.c

Table 2.8.2.1-2: Function Summary – overline_addtext()

This function allows the user to place a string bound to a particular overline.

Calls	
Function	Where Described
gen add text	Overlayif.c
default mouse help	Pvd windows.c
uncolor line	this file
draw overline	Overdraw.c
zoom overlay	Overlayif.c

Table 2.8.2.1-3: Function Summary – size_loop(Overline)

This function lets the user place a size icon on a displayed line.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Table 2.8.2.1-4: Function Summary - oveline_delete()

This function takes the selected line and deletes it.

Calls	
Function	Where Described
uncolor line	this file
draw overline	Overdraw.c
zoom overlay	Overlayif.c

Table 2.8.2.1-5: Function Summary - overline_newfeatures()

This function allows the user to modify the features of a selected line.

Calls	
Function	Where Described
mouse to option	option.c
del line window	this file
get free overline	Overline.c
uncolor line	this file
draw line window	this file
mgive	Concurrent graphics library
mgipln	Concurrent graphics library
display_block	option.c
mgihue	Concurrent graphics library
mark box	option.c
show overline	this file
copy overline	Overline.c
handle_popup_windows	pop_windows.c
draw overline	Overdraw.c
size loop	this file
init cursor	Pvd_iface.c
zoom overlay	Overlayif.c

Table 2.8.2.1-6: Function Summary - show_overline(Overline)

This function displays Overline in a popup window.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mark box	option.c
draw name	this file
draw time	this file
solid action	ovline func.c
dashed action	Ovline func.c
lgeneric type	Ovline func.c
draw blabel1	this file
draw blabel2	this file
lgeneric_size	Overline_func.c

Table 2.8.2.1-7: Function Summary – overline_newshape()

This function allows the user to modify the shape of a selected line.

Calls	
Function	Where Described
calc zoom ratio	New overlay.c
get free overline	Overline.c
copy overline	Overline.c
uncolor line	this file
overline features	Overdraw.c
mgihue	Concurrent graphics library
mgewidth	Concurrent graphics library
over user to pixel	New overlay.c
mgittl	Concurrent graphics library
freedraw mouse help	Pvd windows.c
overline loop	this file
draw overline	Overdraw.c
zoom overlay	Overlayif.c
init cursor	Pdv iface.c
default mouse help	Pvd windows.c

Table 2.8.2.1-8: Function Summary – select_overline()

This function selects a displayed overline by placing the cursor at any end point and pressing the left button.

Return Values		
Return Value	Type	Meaning
1	int	overline selected
0	int	overline not selected

Calls	
Function	Where Described
uncolor line	this file
mouse to overline	Overline.c
color line	Overlineif.c

Table 2.8.2.1-9: Function Summary – overline_loop(Overline)

This function polls the position of the mouse. If the variations in either Y or X direction are bigger than some threshold, then a new offset is recorded.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
calc zoom ratio	New overlay.c
mgewidth	Concurrent graphics library
mgiv	Concurrent graphics library
mgipin	Concurrent graphics library
init drawing cursor	Overlayif.c
overline features	Overdraw.c
overline pencil on	this file
init erasing cursor	Overlayif.c
overline erase	this file
default mouse help	Pvd windows.c

Table 2.8.2.1-10: Function Summary – overline_pencil_on(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
get angle	Overdraw.c
calc zoom ratio	New overlay.c
pixel to user	Pvd iface.c
mggetcursxy	Concurrent graphics library
over user to pixel	New overlay.c
add segment	Overdraw.c

Table 2.8.2.1-11: Function Summary – overline_erase(Overline)

This function erases an overline.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
calc zoom ratio	New overlay.c
erase segment	Overdraw.c
zoom overlay	Overlayif.c

Table 2.8.2.1-12: Function Summary – draw_line_window()

This function draws the pop window to create and edit lines.

Calls	
Function	Where Described
create window	Pop_windows.c
line reg def	this file
del line window	this file

Table 2.8.2.1-13: Function Summary – del_line_window()

This function erases the pop window for overlines.

Calls	
Function	Where Described
reset region	option.c
delete window	pop_windows.c

Table 2.8.2.1-14: Function Summary – line_reg_def(PopWindow)

This function defines blocks in a LineRegion.

Parameters		
Parameter	Type	Where Typedef Declared
PopWindow	pointer to struct POP_WIN	pop_win.h

Calls	
Function	Where Described
define block	option.c
option reg define	option.c

Table 2.8.2.1-15: Function Summary – color_line(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
draw overline	Overdraw.c

Table 2.8.2.1-16: Function Summary – uncolor_line(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
draw_overline	Overdraw.c

Table 2.8.2.1-17: Function Summary – draw_name(Overline)

This function places the line label at each end of the overline, under the line. It draws on the LineRegion pop window.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgigf	Concurrent graphics library
mgigfs	Concurrent graphics library

Table 2.8.2.1-18: Function Summary – draw_time(Overline)

This function places the time string which describes when the line is effective.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgigf	Concurrent graphics library
mgigfs	Concurrent graphics library

Table 2.8.2.1-19: Function Summary – draw_line(Overline)

This function draws the line in the pop window, taking into consideration things like dashed, thickness, etc.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgwidth	Concurrent graphics library
mgitl	Concurrent graphics library

Table 2.8.2.1-20: Function Summary – draw_lsize(Overline)

This function draws the unit size corresponding to the boundary after redrawing the line under it.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
room for lsize	this file
get free symbol	symbol.c
calc basic dim	symbol.c
draw size	syndraw.c

Table 2.8.2.1-21: Function Summary – room_for_lsize(Overline, Mode)

This function erases or redraws the section of teh overline that coincides with the size icons.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Mode	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgwidth	Concurrent graphics library subroutine
mgitl	Concurrent graphics library subroutine
mgidash	Concurrent graphics library subroutine

Table 2.8.2.1-22: Function Summary – draw_ltype(Overline)

This function makes the changes necessary to draw the overline type.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
draw blabel1	this file
draw blabel2	this file
draw endlabels	this file
draw midlabel	this file

Table 2.8.2.1-23: Function Summary – draw_blabel2(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.1-24: Function Summary – draw_blabel1(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.1-25: Function Summary – draw_endlabels(Overline, String)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
String	pointer to char	standard

Calls	
Function	Where Described
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
String	pointer to char	standard

Calls	
Function	Where Described
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

2.8.2.2 ovline_func.c

This CSU contains routines included in overline option blocks.

**Table 2.8.2.2-1: Function Summary - name_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

Table 2.8.2.2-2: Function Summary - name_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_name	Overlineif.c
get_string	Overlayif.c

**Table 2.8.2.2-3: Function Summary - time_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

Table 2.8.2.2-4: Function Summary – time_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_time	Overlayif.c
get_string	Overlayif.c

Table 2.8.2.2-5: Function Summary – unit1_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

Table 2.8.2.2-6: Function Summary – unit1_action
(Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_blabel1	Overlineif.c
get_string	Overlayif.c

**Table 2.8.2.2-7: Function Summary - unit2_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	initiates labeling

Table 2.8.2.2-8: Function Summary - unit2_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	opron.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw blabel2	Overlineif.c
get_string	Overlayif.c

**Table 2.8.2.2-9: Function Summary - solid_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	initiates labeling function

Table 2.8.2.2-10: Function Summary - solid_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
dashed action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw line	Overlineif.c
draw lsize	Overlineif.c

**Table 2.8.2.2-11: Function Summary – dashed_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	initiates labeling

Table 2.8.2.2-12: Function Summary – dashed_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
solid action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw line	Overlineif.c
draw lsize	Overlineif.c

**Table 2.8.2.2-13: Function Summary – thin_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	initiates labeling

Table 2.8.2.2-14: Function Summary – thin_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
thick action	Ovline func.c
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw line	Overlineif.c
draw lsize	Overlineif.c
draw name	Overlineif.c
draw time	Overlayif.c

Table 2.8.2.2-15: Function Summary – thick_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	initiates labeling

Table 2.8.2.2-16: Function Summary – thick_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
thin action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw line	Overlineif.c
draw lsize	Overlineif.c
draw name	Overlineif.c
draw time	Overlineif.c

Table 2.8.2.2-17: Function Summary - lgeneric_size (Size, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Size	int	standard
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw lsize	Overlineif.c

Table 2.8.2.2-18: Function Summary - lsquad_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric size	this file

Table 2.8.2.2-19: Function Summary - lsection_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric size	this file

Table 2.8.2.2-20: Function Summary – lplatoon_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_lsize	Overlineif.c
lgeneric_size	this file

Table 2.8.2.2-21: Function Summary – ltroop_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_size	this file

Table 2.8.2.2-22: Function Summary – lbattalion_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_size	this file

Table 2.8.2.2-23: Function Summary – lgroup_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_size	this file

Table 2.8.2.2-24: Function Summary – lgeneric_type (Type, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Type	int	standard
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
lgeneric_size	this file
draw_lsize	Overlineif.c
room_for_lsize	Overlineif.c
draw_ltype	Overlineif.c

Table 2.8.2.2-25: Function Summary – boundary_label (LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

Table 2.8.2.2-26: Function Summary – boundary_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
unit1_action	this file
unit2_action	this file
lgeneric_size	this file
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_lsize	Overlineif.c
room_for_lsize	Overlineif.c
lgeneric_type	this file

**Table 2.8.2.2-27: Function Summary – ldeparture_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

**Table 2.8.2.2-28: Function Summary – ldeparture_action
(Region, Overline)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_type	this file

**Table 2.8.2.2-29: Function Summary – phase_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

Table 2.8.2.2-30: Function Summary – phase_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_ltype	Overlineif.c
lgeneric_type	this file

**Table 2.8.2.2-31: Function Summary - pdeployment_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

**Table 2.8.2.2-32: Function Summary - pdeployment_action
(Region, Overline)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_type	this file

**Table 2.8.2.2-33: Function Summary - nofire_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	initiates labeling

**Table 2.8.2.2-34: Function Summary – nofire_action
(Region, Overline)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_type	this file

**Table 2.8.2.2-35: Function Summary – delay_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw text label	int	inititates labeling

Table 2.8.2.2-36: Function Summary – delay_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric_type	this file

**Table 2.8.2.2-37: Function Summary – free_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
draw_text_label	int	inititates labeling

Table 2.8.2.2-38: Function Summary – free_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
lgeneric type	this file

Table 2.8.2.2-39: Function Summary – ldone_create_action (Region, Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
reset region	option.c
delete window	pop_windows.c

2.8.2.3 overdraw.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU draws overlay lines from menu choices for free drawing.

Table 2.8.2.3-1: Function Summary – draw_overline(Overline, Window, Planes, PencilHue, ClearHue, Mode)

This function draws an overline.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Window	int	standard
Planes	int	standard
PencilHue	int	standard
ClearHue	int	standard
Mode	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgimodfunc	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
calc_zoom_ratio	New overlay.c
mgewidth	Concurrent graphics library subroutine
mgrscalexy	Concurrent graphics library subroutine
over_user_to_pixel	New overlay.c
mgittl	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
overline features	this file
add_size	this file

Table 2.8.2.3-2: Function Summary – add_segment(Overline, Offset)

This function draws a segment from the current end of line to the new offset.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Offset	pointer to OFFSET	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over_user_to_pixel	New overlay.c
mgittl	Concurrent graphics library subroutine

Table 2.8.2.3-3: Function Summary – erase_segment(Overline, EndPoint)

This function erases the last segment in the line and cleans up.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Offset	pointer to OFFSET	overline.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over_user_to_pixel	New overlay.c
mgittl	Concurrent graphics library subroutine

Table 2.8.2.3-4: Function Summary – add_size(Overline, OffsetNum, Planes, PencilHue, ClearHue)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
OffsetNum	int	standard
Planes	int	standard
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
mgewidth	Concurrent graphics library subroutine
get angle	Overdraw.c
get need	this file
over user to pixel	New overlay.c
size on line	this file

Table 2.8.2.3-5: Function Summary – overline_features (Overline, Planes, PencilHue)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Planes	int	standard
PencilHue	int	standard

Calls	
Function	Where Described
get angle	Overdraw.c
mgiv	Concurrent graphics library subroutine
mgimodfunc	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
add name	this file
add time	this file
add type	this file
add free text	this file

Table 2.8.2.3-6: Function Summary – add_name(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
over_user_to_pixel	New_overlay.c
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.3-7: Function Summary – add_time(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
over_user_to_pixel	New_overlay.c
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.3-8: Function Summary – add_type(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Calls	
Function	Where Described
add_endlabels	this file
add_midlabel	this file

**Table 2.8.2.3-9: Function Summary – add_free_text
(Text, Window, Planes, Hue)**

Parameters		
Parameter	Type	Where Typedef Declared
Text	pointer to SYMLABEL	symbol.h
Window	int	standard
Planes	int	standard
Hue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
over user to pixel	New overlay.c
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.3-10: Function Summary – add_endlabels(Overline, Type)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Type	pointer to char	standard

Calls	
Function	Where Described
over user to pixel	New overlay.c
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

**Table 2.8.2.3-11: Function Summary – add_midlabel
(Overline, String, Side)**

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
String	pointer to char	standard
Side	int	standard

Calls	
Function	Where Described
over user to pixel	New overlay.c
get angle	this file
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.3-12: Function Summary – size on line(Overline, Angle, NumIcons, LineWidth, StartX, StartY, PencilHue, ClearHue)

This function draws the size on boundary lines.

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
Angle	double	standard
NumIcons	int	standard
LineWidth	int	standard
StartX, StartY	int	standard
PencilHue, ClearHue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mgewidth	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
mgitl	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine

Table 2.8.2.3-13: Function Summary – find_offset (Overline, ChosenX, ChosenY)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h
ChosenX, ChosenY	int	standard

Return Values		
Return Value	Type	Meaning
i	int	returns offset number of closest overlay line
-1	int	value is outside limit

Calls	
Function	Where Described
calc zoom ratio	New overlay.c
over user to pixel	New overlay.c

Table 2.8.2.3-14: Function Summary - get_need(Overline)

Parameters		
Parameter	Type	Where Typedef Declared
Overline	pointer to OVERLINE	overline.h

Return Values		
Return Value	Type	Meaning
NumIcons	int	displays number icons

Calls	
Function	Where Described
over user to pixel	New overlay.c
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.8.2.3-15: Function Summary - get_angle(OffsetX, OffsetY)

Parameters		
Parameter	Type	Where Typedef Declared
OffsetX, OffsetY	short	standard

Return Values		
Return Value	Type	Meaning
Angle	double	value of angle

2.8.2.4 overline.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines that handle overlines, including boundaries and control lines. It manages OverlineTable.

Table 2.8.2.4-1: Function Summary - overline_init()

This function initializes the Overline Table.

Table 2.8.2.4-2: Function Summary - overline_reset()

This function reinitializes the Overline Table.

Table 2.8.2.4-3: Function Summary – get_free_overline()

This function allocates an overline structure and returns a pointer to the newly allocated overline.

Return Values		
Return Value	Type	Meaning
Line	pointer to OVERLINE	pointer to newly allocated overline

Table 2.8.2.4-4: Function Summary – copy_overline(From, To)

This function creates a copy of an overline.

Parameters		
Parameter	Type	Where Typedef Declared
From, To	pointer to OVERLINE	overline.h

**Table 2.8.2.4-5: Function Summary – zoom_overline
(Planes, PencilHue, ClearHue)**

This function redraws the active lines in the Overline Table at the appropriate zoom level and in the given planes.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
PencilHue, ClearHue	int	standard

Calls	
Function	Where Described
draw overline	Overdraw.c

Table 2.8.2.4-6: Function Summary – erase_overline(Planes, ClearHue)

This function erases all the free draw objects in the Overline Table.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
draw overline	Overdraw.c

Table 2.8.2.4-7: Function Summary – insert_overline(NewLine)

This function inserts NewLine into the Overline Table.

Parameters		
Parameter	Type	Where Typedef Declared
NewLine	pointer to OVERLINE	overline.h

Table 2.8.2.4-8: Function Summary – mouse_to_overline(MouseX, MouseY)

This function finds (and returns a pointer to) the overline whose end points are closest to the given set of mouse coordinates.

Parameters		
Parameter	Type	Where Typedef Declared
MouseX, MouseY	int	x, y coordinates

Return Values		
Return Value	Type	Meaning
CurOverline	pointer to OVERLINE	returns pointer to overline whose endpoints are close to mouse coordinates
NULL	pointer to OVERLINE	no matching overline found

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
over user to pixel	New overlay.c
over point in box	New overlay.c

2.8.3 Unit Symbols CSC Description

This CSC enables military unit symbols to be created and edited.

This CSC functionality is broken down into the CSUs shown in Figure 2.8.3-1.

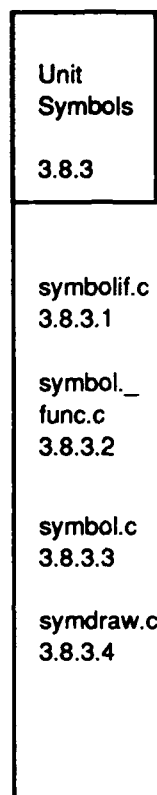


Figure 2.8.3-1: Overlays--Unit Symbols CSC Structure.

2.8.3.1 symbolif.c

This CSU manages military unit symbols. It contains routines for displaying popup windows and for creating and modifying the symbols.

Table 2.8.3.1-1: Function Summary - symbol_create()

This function displays the pop window for creating military symbols.

Calls	
Function	Where Described
draw create window	this file
get free symbol	symbol.c
mgiv	Concurrent graphics library subroutine
mgipin	Concurrent graphics library subroutine
display block	option.c
mgihue	Concurrent graphics library subroutine
mark box	option.c
calc basic dim	symbol.c
friend action	symbol func.c
actual action	symbol func.c
unit action	symbol func.c
platoon action	symbol func.c
mouse to option	option.c
handle popup windows	pop windows.c
mggetcursmode	Concurrent graphics library subroutine
display cursor loc	
mgimodfunc	Concurrent graphics library subroutine
draw symbol	symbol.c
mgigetvcoor	Concurrent graphics library subroutine
default mouse help	

Table 2.8.3.1-2: Function Summary - draw_create_window()

This function creates a pop_window and defines the SymbolRegion if it is the first time being draw. It also draws the window.

Calls	
Function	Where Described
create window	pop windows.c
del create window	this file
sym reg def	this file
draw window	pop windows.c

Table 2.8.3.1-3: Function Summary - sym_reg_def(PopWindow)

This function creates a symbol region to fit inside a window with the characteristics of the opo window given.

Parameters		
Parameter	Type	Where Typedef Declared
PopWindow	pointer to struct PopWindow	symbolif.h

Calls	
Function	Where Described
option reg define	option.c
define block	option.c

Table 2.8.3.1-4: Function Summary - del_create_window()

This function erases the pop window, resets all the options in SymbolRegion, and resets SymbolCreateIsUp (that is, it sets it to 0.)

Calls	
Function	Where Described
reset region	option.c
delete_window	pop_windows.c

Table 2.8.3.1-5: Function Summary - select_symbol()

This function selects one of the symbols displayed by placing the cursor inside it and pressing the left button. Returns 1 if the symbol was selected and 0 otherwise. Selected symbols experience a color change.

Calls	
Function	Where Described
uncolor selected	symbolif.c
mouse to symbol	symbol.c
color selected	symbolif.c

Return Values		
Return Value	Type	Meaning
1	int	symbol selected
0	int	otherwise

Table 2.8.3.1-6: Function Summary - symbol_edit()

This function takes the selected symbol and modified it.

Calls	
Function	Where Described
get free symbol	symbol.c
copy symbol	symbol.c
draw create window	this file
calc basic dim	symbol.c
mgiv	Concurrent graphics library subroutine
get window num	pop_windows.c
mgipin	Concurrent graphics library subroutine
display block	option.c
mgihue	Concurrent graphics library subroutine
mark box	option.c
show symbol	this file
mouse to option	option.c
handle popup windows	pop_windows.c
calc zoom ratio	new_overlay.c
uncolor selected	symbolif.c
draw symbol	symbol.c

Table 2.8.3.1-7: Function Summary - symbol_move()

This function takes the selected symbol and moves it.

Calls	
Function	Where Described
mggetcursmode	Concurrent graphics library subroutine
mgicursmode	Concurrent graphics library subroutine
calc zoom ratio	new_overlay.c
calc basic dim	symbol.c
uncolor selected	symbolif.c
draw symbol	symbol.c
mgimodfunc	Concurrent graphics library subroutine
move mouse help	
display cursor loc	
mgigetvcoor	Concurrent graphics library subroutine
pixel to user	pvdiface.c
add free text	overdraw.c
default mouse help	
zoom overlay	overlayif.c

Table 2.8.3.1-8: Function Summary - symbol_delete()

This function deletes the selected symbol.

Calls	
Function	Where Described
calc zoom ratio	new_overlay.c
calc basic dim	symbol.c
uncolor selected	symbolif.c
mgimodfunc	Concurrent graphics library subroutine
draw symbol	symbol.c
zoom overlay	overlayif.c

Table 2.8.3.1-9: Function Summary - show_symbol(Symbol)

This function draws the current symbol in the pop window by selecting the appropriate options. Assumes that the symbol region has been reset and thus none of the options are marked Selected.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mark box	option.c
draw designation	symdraw.c
draw hi_eche	symdraw.c
friend action	symbol_func.c
foe action	symbol_func.c
draw box	symdraw.c
draw proposed	symdraw.c
mgidash	Concurrent graphics library subroutine
generic box	symbol_func.c
generic size	symbol_func.c
generic supply	symbol_func.c
generic branch	symbol_func.c
generic weapon	symbol_func.c

Table 2.8.3.1-10: Function Summary - symbol_addtext()

This function tries to add text to an active symbol. If there are more text strings attached to the symbol, it will announce the error and abort the operation.

Calls	
Function	Where Described
gen add text	overlayif.c
uncolor selected	symbolif.c
draw symbol	symbol.c
zoom overlay	overlayif.c

Table 2.8.3.1-11: Function Summary - color_selected(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw symbol	symbol.c

Table 2.8.3.1-12: Function Summary - uncolor_selected(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw symbol	symbol.c

Table 2.8.3.1-13: Function Summary - symbol_reduce()

Calls	
Function	Where Described
resize mouse help	
resize symbol	this file
default mouse help	
zoom overlay	overlayif.c

Table 2.8.3.1-14: Function Summary - resize_symbol(Flag)

Parameters		
Parameter	Type	Where Typedef Declared
Flag	int	standard

Calls	
Function	Where Described
uncolor selected	this file
draw symbol	symbol.c

2.8.3.2 symbol_func.c

This CSU contains routines for selecting symbols and labeling them on the screen.

**Table 2.8.3.2-1: Function Summary - desig_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Designation"

Table 2.8.3.2-2: Function Summary - desig_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_designation	syndraw.c

**Table 2.8.3.2-3: Function Summary - hi_eche_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Higher echelons of command"

Table 2.8.3.2-4: Function Summary - hi_eche_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_hi_eche	symdraw.c

Table 2.8.3.2-5: Function Summary - friend_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Friendly symbol"

Table 2.8.3.2-6: Function Summary - friend_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
foe_action	this file
mgihue	Concurrent graphics library subroutine
mark_box	option.c

**Table 2.8.3.2-7: Function Summary - foe_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Enemy symbol"

Table 2.8.3.2-8: Function Summary - foe_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
friend action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c

**Table 2.8.3.2-9: Function Summary - actual_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Placed on actual location"

Table 2.8.3.2-10: Function Summary -actual_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
proposed action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw proposed	symdraw.c
mgidash	Concurrent graphics library subroutine
draw_box	symdraw.c

**Table 2.8.3.2-11: Function Summary - proposed_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Placed on proposed location"

Table 2.8.3.2-12: Function Summary - proposed_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
actual action	this file
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw box	symdraw.c
mgidash	Concurrent graphics library subroutine
draw_proposed	symdraw.c

Table 2.8.3.2-13: Function Summary - generic_box(Shape, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Shape	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_box	symdraw.c
mgidash	Concurrent graphics library subroutine

**Table 2.8.3.2-14: Function Summary - unit_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Unit"

Table 2.8.3.2-15 Function Summary - unit_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_box	symdraw.c
mgidash	Concurrent graphics library subroutine
generic_box	this file

Table 2.8.3.2-16: Function Summary - headq_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Headquarters"

Table 2.8.3.2-17: Function Summary - headq_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic box	this file

Table 2.8.3.2-18: Function Summary - css_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Combat service support"

Table 2.8.3.2-19: Function Summary - css_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_box	this file

Table 2.8.3.2-20: Function Summary - generic_size(Size, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Size	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_size	symdraw.c

**Table 2.8.3.2-21: Function Summary - squad_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Squad"

Table 2.8.3.2-22: Function Summary - squad_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic size	this file

**Table 2.8.3.2-23: Function Summary - section_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text_label	option.c

Return Values		
Return Value	Type	Meaning
draw text_label	int	"Section"

Table 2.8.3.2-24: Function Summary - section_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic size	this file

**Table 2.8.3.2-25: Function Summary - platoon_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text_label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Platoon, Detachment"

Table 2.8.3.2-26: Function Summary - platoon_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw size	symdraw.c
generic size	this file

Table 2.8.3.2-27: Function Summary - troop_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Company, Battery, Troop"

Table 2.8.3.2-28: Function Summary - troop_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic size	this file

**Table 2.8.3.2-29: Function Summary - battalion_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Battalion, Squadron"

Table 2.8.3.2-30: Function Summary - battalion_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_size	this file

**Table 2.8.3.2-31: Function Summary - group_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Group, Regiment"

Table 2.8.3.2-32: Function Summary - group_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_size	this file

Table 2.8.3.2-33: Function Summary - generic_branch(Branch, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Branch	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_branch	symdraw.c

Table 2.8.3.2-34: Function Summary - air_def_art_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Air Defense Artillery"

**Table 2.8.3.2-35: Function Summary - air_def_art_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

**Table 2.8.3.2-36: Function Summary - armor_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Armor"

Table 2.8.3.2-37: Function Summary - armor_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

**Table 2.8.3.2-38: Function Summary - nbc_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"NBC"

Table 2.8.3.2-39: Function Summary - nbc_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

**Table 2.8.3.2-40: Function Summary - cavalry_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Cavalry, Reconnaissance"

Table 2.8.3.2-41: Function Summary - cavalry_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_branch	this file

**Table 2.8.3.2-42: Function Summary - field_art_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Field artillery"

Table 2.8.3.2-43: Function Summary - field_art_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_branch	this file

**Table 2.8.3.2-44: Function Summary - infantry_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Infantry"

Table 2.8.3.2-45: Function Summary - infantry_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

**Table 2.8.3.2-46: Function Summary - medical_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Medical"

Table 2.8.3.2-47: Function Summary - medical_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

Table 2.8.3.2-48: Function Summary - airborne_label
(LimitX, StartY, StartX, Tallest)

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Airborne"

Table 2.8.3.2-49: Function Summary - airborne_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic branch	this file

**Table 2.8.3.2-50: Function Summary - antitank_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Antitank"

Table 2.8.3.2-51: Function Summary - antitank_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_branch	this file

**Table 2.8.3.2-52: Function Summary - army_avi_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Army aviation"

Table 2.8.3.2-53: Function Summary - army_avi_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_branch	this file

**Table 2.8.3.2-54: Function Summary - repair_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Repair and maintenance"

Table 2.8.3.2-55: Function Summary - repair_action(Region, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_branch	this file

Table 2.8.3.2-56: Function Summary - generic_supply(Supply, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Supply	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_supply	syndraw.c

**Table 2.8.3.2-57: Function Summary - ammo_all_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"All types"

**Table 2.8.3.2-58: Function Summary - ammo_all_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_supply	syndraw.c
ammo_art_action	this file
ammo_rock_action	this file
ammo_small_action	this file
ammo_specd_action	this file
ammo_conv_action	this file

**Table 2.8.3.2-59: Function Summary - ammo_art_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label		"Artillery"

**Table 2.8.3.2-60: Function Summary - ammo_art_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_supply	this file

**Table 2.8.3.2-61: Function Summary - ammo_rock_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Rocket and guided missile"

**Table 2.8.3.2-62: Function Summary - ammo_rock_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_supply	this file

**Table 2.8.3.2-63: Function Summary - ammo_small_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Small arms"

**Table 2.8.3.2-64: Function Summary - ammo_small_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_supply	this file

**Table 2.8.3.2-65: Function Summary - ammo_spec_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Special"

**Table 2.8.3.2-66: Function Summary - ammo_spec_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_supply	this file

**Table 2.8.3.2-67: Function Summary - ammo_conv_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Conventional"

**Table 2.8.3.2-68: Function Summary - ammo_conv_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic supply	this file

**Table 2.8.3.2-69: Function Summary - generic_weapon
(Weapon, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Weapon	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_weapon	syndraw.c

**Table 2.8.3.2-70: Function Summary - weap_auto_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Automatic infantry weapon"

**Table 2.8.3.2-71: Function Summary - weap_auto_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-72: Function Summary - weap_mortar_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Mortar"

**Table 2.8.3.2-73: Function Summary - weapon_mortar_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-74: Function Summary - weap_airdef_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Air defense machine gun"

**Table 2.8.3.2-75: Function Summary - weap_airdef_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic weapon	this file

**Table 2.8.3.2-76: Function Summary - weap_antit_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Antitank rocket launcher"

**Table 2.8.3.2-77: Function Summary - weap_antit_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-78: Function Summary - weap_gun_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Gun or gun/howitzer"

**Table 2.8.3.2-79: Function Summary - weap_gun_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-80: Function Summary - weap_howit_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Howitzer"

**Table 2.8.3.2-81: Function Summary - weap_howit_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic weapon	this file

**Table 2.8.3.2-82: Function Summary - weap_antig_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Gun in antitankrole"

**Table 2.8.3.2-83: Function Summary - weap_antig_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-84: Function Summary - weap_recoil_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Recoilless rifle"

**Table 2.8.3.2-85: Function Summary - weap_recoil_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-86: Function Summary - weap_airgun_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Gun in air defense role"

**Table 2.8.3.2-87: Function Summary - weap_airgun_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-88: Function Summary - weap_miss_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Missile or rocket"

**Table 2.8.3.2-89: Function Summary - weap_miss_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-90: Function Summary - weap_airmiss_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Missile in air defense role"

**Table 2.8.3.2-91: Function Summary - weap_antmiss_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-92: Function Summary - weap_antmiss_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Missile in antitank role"

**Table 2.8.3.2-93: Function Summary - weap_antmiss_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

**Table 2.8.3.2-94: Function Summary - armor_tank_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Tank"

**Table 2.8.3.2-95: Function Summary - armor_tank_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic_weapon	this file

Table 2.8.3.2-96: Function Summary - generic_weapsiz(Size, Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Size	int	standard
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark_box	option.c
draw_weapon	syndraw.c

**Table 2.8.3.2-97: Function Summary - weap_light_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Light"

**Table 2.8.3.2-98: Function Summary - weap_light_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
mark box	option.c
draw weapon	symdraw.c
weap med action	this file
weap heavy action	this file

**Table 2.8.3.2-99: Function Summary - weap_med_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"Medium"

**Table 2.8.3.2-100: Function Summary - weap_med_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
generic weapsize	this file

**Table 2.8.3.2-101: Function Summary - weap_heavy_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Medium"

**Table 2.8.3.2-102: Function Summary - weap_heavy_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"Heavy"

**Table 2.8.3.2-103: Function Summary - done_create_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"DONE"

**Table 2.8.3.2-104: Function Summary - done_create_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
reset region	option.c
delete_window	pop_windows.c

**Table 2.8.3.2-105: Function Summary - next_create_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw text label	option.c

Return Values		
Return Value	Type	Meaning
draw text label	int	"NEXT PAGE"

**Table 2.8.3.2-106: Function Summary - next_create_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
erase_region	option.c
display_block	option.c

**Table 2.8.3.2-107: Function Summary - prev_create_label
(LimitX, StartY, StartX, Tallest)**

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartY	int	standard
StartX	pointer to int	standard
Tallest	pointer to int	standard

Calls	
Function	Where Described
draw_text_label	option.c

Return Values		
Return Value	Type	Meaning
draw_text_label	int	"PREVIOUS PAGE"

**Table 2.8.3.2-108: Function Summary - prev_create_action
(Region, Symbol)**

Parameters		
Parameter	Type	Where Typedef Declared
Region	pointer to OPTION_REGION	option.h
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
erase_region	option.c
display_block	option.c

2.8.3.3 symbol.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines for managing SymbolTable. It initializes and reinitializes SymbolTable, allocates space and inserts symbols and erases symbols currently active in SymbolTable. It also finds and returns a pointer to a symbol that encloses a set of mouse coordinates, and displays all symbols that would be visible under the current zoom level.

Table 2.8.3.3-1: Function Summary - symbol_init()

This function initializes the SymTable.

Table 2.8.3.3-2: Function Summary - symbol_reset()

This function reinitializes the SymTable.

Table 2.8.3.3-3: Function Summary - get_free_symbol()

This function allocates space for SYMBOL and initializes it with the appropriate defaults.

Return Values		
Return Value	Type	Meaning
NewSymbol	pointer to SYMBOL	allocates space for SYMBOL

Table 2.8.3.3-4: Function Summary - insert_symbol(NewSymbol)

This function inserts this symbol into the SymbolTable.

Parameters		
Parameter	Type	Where Typedef Declared
NewSymbol	pointer to SYMBOL	symbolif.h

**Table 2.8.3.3-5: Function Summary - draw_symbol
(Symbol, Window, Planes, Hue)**

This function takes a SYMBOL and draws it. It examines each of the characteristic fields in the symbol. Drawing is done in relative mode and in real coordinates. This routine will be able to write in color using the ICON planes or in monochrome in pop_window planes.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h
Window	int	standard
Planes	int	standard
Hue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine
calc zoom ratio	new overlay.c
calc basic dim	this file
draw designation	symdraw.c
draw hi_eche	symdraw.c
draw size	symdraw.c
draw proposed	symdraw.c
mgidash	Concurrent graphics library subroutine
draw box	symdraw.c
draw supply	symdraw.c
draw branch	symdraw.c
draw weapon	symdraw.c
draw symtext	symdraw.c

Table 2.8.3.3-6: Function Summary - calc_basic_dim(Symbol)

This function sets global variables for drawing routines. Assumes MAX_HALF_W will be less than POP_WINDOWS.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

**Table 2.8.3.3-7: Function Summary - mouse_to_symbol
(MouseX, MouseY)**

This function finds and returns a pointer to the symbol that encloses the given set of mouse coordinates. If no symbol encloses them, return is NULL.

Parameters		
Parameter	Type	Where Typedef Declared
MouseX, MouseY	int	standard

Calls	
Function	Where Described
mgrrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	new overlay.c
calc basic dim	this file
over user to pixel	new overlay.c
over point in box	new Overlay.c

Return Values		
Return Value	Type	Meaning
NULL	pointer to SYMBOL	mouse clicked outside of all symbols
CurSymbol	pointer to SYMBOL	symbol encloses mouse coordinates

Table 2.8.3.3-8: Function Summary - erase_symbol(Planes, ClearHue)

This function erases all the symbols currently active in the symbol table. It does NOT clear the table.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	new overlay.c
draw symbol	this file

**Table 2.8.3.3-9: Function Summary - zoom_symbol
(Planes, FriendHue, FoeHue)**

This function displays all the symbols that would be visible under the current zoom level.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
FriendHue	int	standard
FoeHue	int	standard

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	new overlay.c
calc basic dim	this file
draw symbol	this file

**Table 2.8.3.3-10: Function Summary - copy_symbol
(FromSymbol, ToSymbol)**

Parameters		
Parameter	Type	Where Typedef Declared
FromSymbol	pointer to SYMBOL	symbolif.h
ToSymbol	pointer to SYMBOL	symbolif.h

2.8.3.4 symdraw.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU examines various fields in SYMBOL and dispatches appropriate drawing routines for them.

Table 2.8.3.4-1: Function Summary - draw_box(Symbol)

This function examines various fields in SYMBOL and dispatches to the appropriate drawing routines.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw_unit	this file
add_headq	this file

Table 2.8.3.4-2: Function Summary - draw_size(Symbol)

This function draws the size symbol on top of the basic box.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw_squad	this file
draw_section	this file
draw_platoon	this file
draw_troop	this file
draw_battalion	this file
draw_group	this file

Table 2.8.3.4-3: Function Summary - draw_branch(Symbol)

This function draws the branch inside the basic box. The branch could be the combination of two branches, e.g., INFANTRY and ARMOR are equivalent to "Mechanized Infantry."

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw air def	this file
draw armor	this file
draw nbc	this file
draw cavalry	this file
draw field	this file
draw infantry	this file
draw medical	this file
draw airborne	this file
draw antitank	this file
draw aviation	this file
draw repair	this file

Table 2.8.3.4-4: Function Summary - draw_hi_eche(Symbol)

This function draws information regarding Higher Echelons of command separated by a lash. Each echelon has an optional size and branch.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4- 5: Function Summary - draw_designation(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-6: Function Summary - draw_syntext(Symbol, Planes, Hue)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h
Planes	int	standard
Hue	int	standard

Calls	
Function	Where Described
add free text	overdraw.c

Table 2.8.3.4-7: Function Summary - draw_proposed(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-8: Function Summary - draw_supply(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw ammo	this file
add ammo art	this file
add ammo rock	this file
add ammo small	this file
add ammo spec	this file
add ammo conv	this file

Table 2.8.3.4-9: Function Summary - draw_weapon(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw weap auto	this file
add high trayec	this file
add airdef	this file
add flat trayec	this file
add launcher	this file
draw weap gun	this file
draw weap miss	this file
draw armor tank	this file

Table 2.8.3.4-10: Function Summary - draw_weapsize(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
draw weap_med	this file
draw weap_heavy	this file

Table 2.8.3.4-11: Function Summary - draw_unit(Symbol)

Draws a rectangle in relative mode starting at a given set of center coordinates.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mqirelxy	Concurrent graphics library subroutine
mqil	Concurrent graphics library subroutine
mqimode	Concurrent graphics library subroutine

Table 2.8.3.4-12 Function Summary - add_headq(Symbol)

Adds a staff at the lower left-hand corner of the basic box.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-13: Function Summary - draw_css(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgic	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-14: Function Summary - draw_squad(Symbol)

This function adds a squad size symbol.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-15: Function Summary - draw_section(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-16: Function Summary - draw_platoon(Symbol)

This function adds a platoon size symbol.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-17: Function Summary - draw_troop(Symbol)

This function adds a troop size symbol.

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-18: Function Summary - draw_battalion(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-19: Function Summary - draw_group(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-20: Function Summary - draw_air_def(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-21: Function Summary - draw_armor(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-22: Function Summary - draw_nbc(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-23: Function Summary - draw_cavalry(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-24: Function Summary - draw_field(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-25: Function Summary - draw_infantry(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-26: Function Summary - draw_medical(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-27: Function Summary - draw_airborne(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-28: Function Summary - draw_antitank(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-29: Function Summary - draw_aviation(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgifa	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-30: Function Summary - draw_repair(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-31: Function Summary - draw_ammo(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgifa	Concurrent graphics library subroutine
mgipoly	Concurrent graphics library subroutine
mgimodfunc	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-32: Function Summary - add_ammo_art(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-33: Function Summary - add_ammo_rock(Symbol)

This function is not implemented.

Table 2.8.3.4-34: Function Summary - add_ammo_small(Symbol)

This function is not implemented.

Table 2.8.3.4-35: Function Summary - add_ammo_spec(Symbol)

This function is not implemented.

Table 2.8.3.4-36: Function Summary - add_ammo_conv(Symbol)

This function is not implemented.

Table 2.8.3.4-37: Function Summary - draw_weap_auto(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-38: Function Summary - add_high_trayec(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgic	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-39: Function Summary - add_airdef(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-40: Function Summary - add_flat_trayec(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-41: Function Summary - add_launcher(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-42: Function Summary - draw_weap_gun(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-43: Function Summary - draw_weap_miss(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgia	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-44: Function Summary - draw_armor_tank(Symbol)

Parameters		
Parameter	Type	Where Typedef Declared
Symbol	pointer to SYMBOL	symbolif.h

Calls	
Function	Where Described
mgirelxy	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
mgimode	Concurrent graphics library subroutine

Table 2.8.3.4-45: Function Summary - draw_weap_med(Symbol)

This function is not implemented.

Table 2.8.3.4-46: Function Summary - draw_weap_heavy(Symbol)

This function is not implemented.

2.8.4 Control Points CSC Description

This CSC enables several types of control points to be created. These include: check, contact, release/start, coordinate, linkup, and passage points.

This CSC functionality is broken down into the CSUs shown in Figure 2.8.4-1.

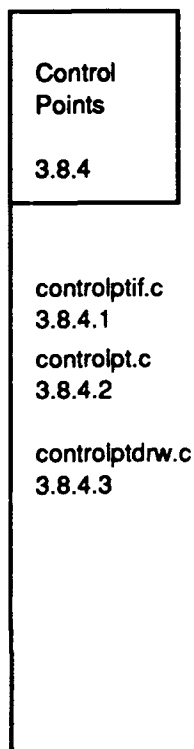


Figure 2.8.4-1: Overlays--Control Points CSC Structure.

2.8.4.1 controlptif.c

This CSU contains routines for selecting control points. It interprets mouse gestures and invokes appropriate actions.

Table 2.8.4.1-1: Function Summary – select_controlpt()

This function selects a control point.

Return Values		
Return Value	Type	Meaning
0	int	no point selected
1	int	point selected

Calls	
Function	Where Described
uncolor_controlpt	this file
mouse to controlpt	Controlpt.c

Table 2.8.4.1-2: Function Summary – color_controlpt(point)

This function draws the selected point with SELECT_HUE.

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h

Calls	
Function	Where Described
draw_controlpt	Controlpt.c

Table 2.8.4.1-3: Function Summary – uncolor_controlpt(point)

This function draws the selected point with CLEAR hue.

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h

Calls	
Function	Where Described
draw_controlpt	Controlpt.c

Table 2.8.4.1-4: Function Summary – controlpt_check()

This function creates a check point icon.

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-5: Function Summary – controlpt_coord()

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-6: Function Summary – controlpt_con()

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-7: Function Summary – controlpt_link()

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-8: Function Summary – controlpt_rel()

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-9: Function Summary – controlpt_pass()

Calls	
Function	Where Described
get free controlpt	Controlpt.c
place controlpt	this file
insert controlpt	Controlpt.c

Table 2.8.4.1-10: Function Summary – move_controlpt()

Calls	
Function	Where Described
uncolor controlpt	this file
draw controlpt	Controlpt.c
place controlpt	this file
add free text	Overdraw.c

Table 2.8.4.1-11: Function Summary – place_controlpt(point)

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h

Calls	
Function	Where Described
mgigetcursmode	Concurrent graphics library subroutine
mgicursmode	Concurrent graphics library subroutine
display cursor loc	
mgimodfunc	Concurrent graphics library subroutine
draw controlpt	Controlpt.c
move mouse help	
mgigetvcoor	Concurrent graphics library subroutine
pixel to user	Pvd iface.c
default mouse help	Pvd windows.c

Table 2.8.4.1-12: Function Summary – controlpt_delete()

Calls	
Function	Where Described
draw_controlpt	Controlpt.c
zoom_overlay	Overlayif.c

Table 2.8.4.1-13: Function Summary – controlpt_reduce()

Calls	
Function	Where Described
resize mouse help	
resize controlpt	this file
default mouse help	Pvd_windows.c
zoom_overlay	Overlayif.c

Table 2.8.4.1-14: Function Summary – resize_controlpt(Flag)

Parameters		
Parameter	Type	Where Typedef Declared
Flag	int	standard

Calls	
Function	Where Described
uncolor controlpt	this file
draw controlpt	Controlpt.c

Table 2.8.4.1-15: Function Summary – controlpt_addtext()

Calls	
Function	Where Described
gen add text	Overlayif.c
default mouse help	Pvd_windows.c
uncolor controlpt	this file
draw controlpt	Controlpt.c
zoom_overlay	Overlayif.c

2.8.4.2 controlpt.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines for managing PointTable; it initializes and reinitializes PointTable. It selects a control point, inserts a point into PointTable, displays all control points that would be visible under the current zoom level, and erases a control point.

Table 2.8.4.2-1: Function Summary – controlpt_init()

This function initializes PointTable.

Table 2.8.4.2-2: Function Summary – controlpt_reset()

This function reinitializes PointTable and frees all previously allocated points.

**Table 2.8.4.2-3: Function Summary – mouse_to_controlpt
(MouseX, MouseY)**

This function tries to select a control point and returns NULL if it cannot.

Parameters		
Parameter	Type	Where Typedef Declared
MouseX, MouseY	int	standard

Return Values		
Return Value	Type	Meaning
CurPoint	pointer to SYMPT	controlpt selected
NULL		cannot select a point

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
over user to pixel	New overlay.c
over point in box	New overlay.c

Table 2.8.4.2-4 Function Summary – get_free_controlpt()

This function returns a SYMPT structure.

Return Values		
Return Value	Type	Meaning
point	pointer to SYMPT	pointer to SYMPT structure

Table 2.8.4.2-5: Function Summary – insert_controlpt(point)

This function tries to insert point into the PointTable.

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h

**Table 2.8.4.2-6: Function Summary – zoom_controlpt
(Planes, PencilHue, ClearHue)**

This function displays all the control points that would be visible under the current zoom level.

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
draw_controlpt	this file

Table 2.8.4.2-7: Function Summary – erase_controlpt(Planes, ClearHue)

This function erases every control point currently displayed

Parameters		
Parameter	Type	Where Typedef Declared
Planes	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgrsi	Concurrent graphics library subroutine
mgigetvcoor	Concurrent graphics library subroutine
calc zoom ratio	New overlay.c
draw_controlpt	this file

2.8.4.3 controlptdrw.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines for calculating and drawing control points.

Table 2.8.4.3-1: Function Summary – draw_controlpt(point, Window, Planes, PencilHue, ClearHue)

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
Window	int	standard
Planes	int	standard
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
calc_zoom_ratio	New_overlay.c
draw_ptshape	this file
draw_pttext	this file

**Table 2.8.4.3-2: Function Summary – draw_pttext
(point, Planes, PencilHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
Planes	int	standard
PencilHue	int	standard

Calls	
Function	Where Described
add_free_text	Overdraw.c

**Table 2.8.4.3-3: Function Summary – draw_ptshape
(point, Planes, PencilHue, ClearHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
Planes	int	standard
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
draw_circlept	this file
add_cross	this file
draw_squarept	this file
add_dot	this file
draw_solidpt	this file

**Table 2.8.4.3-4: Function Summary – draw_circlept
(point, PencilHue, ClearHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over user to pixel	New overlay.c
mgifc	Concurrent graphics library subroutine

**Table 2.8.4.3-5: Function Summary – draw_squarept
(point, PencilHue, ClearHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over user to pixel	New overlay.c
mgibox	Concurrent graphics library subroutine
mgewidth	Concurrent graphics library subroutine
mgitl	Concurrent graphics library subroutine

**Table 2.8.4.3-6: Function Summary – draw_solidpt
(point, PencilHue, ClearHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over user to pixel	New overlay.c
mgifc	Concurrent graphics library subroutine

**Table 2.8.4.3-7: Function Summary - add_cross
(point, PencilHue, ClearHue)**

Parameters		
Parameter	Type	Where Typedef Declared
point	pointer to SYMPT	controlpt.h
PencilHue	int	standard
ClearHue	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine
over user to pixel	New overlay.c
mgifc	Concurrent graphics library subroutine

2.8.5 Manage Overlays CSC Description

This CSC provides management of overlays, including editing, storage and retrieval.

This CSC functionality is broken down into the CSUs shown in Figure 2.8.5-1.

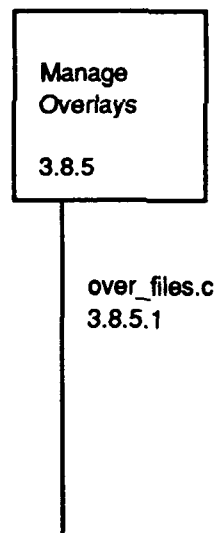


Figure 2.8.5-1: Overlays--Manage Overlays CSC Structure.

2.8.5.1 over_files.c CSU Description (/simnet/pvd/lib/liboverlay.a)

This CSU contains routines for setting path to overlay directory, loading overlay objects, saving overlines in OverlineTable, saving control points in PointTable, saving symbols in SymTable, loading and displaying overlines, loading control points into PointTable, and loading unit symbols into SymTable.

Table 2.8.5.1-1: Function Summary – init_over_files()

This function sets the path to the overlay directory.

Calls	
Function	Where Described
make_path_name	init_env.c
get_data_directory	init_env.c

Table 2.8.5.1-2: Function Summary – load_overlay(fp)

This function loads all the overlay objects contained in file fp.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h

Calls	
Function	Where Described
read_overlines	this file
read_controlpts	this file
read_symbols	this file

Table 2.8.5.1-3: Function Summary – save_overline(fp)

This function saves overlines in OverlineTable.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h

Table 2.8.5.1-4: Function Summary – save_controlpt(fp)

This function saves control points in PointTable.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h

Table 2.8.5.1-5: Function Summary – save_symbol(fp)

This function saves symbols in SymTable.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h

Table 2.8.5.1-6: Function Summary – read_overlines(fp, num)

This function loads and displays overlines.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h
num	int	standard

Calls	
Function	Where Described
get free overline	Overline.c
insert overline	Overline.c

Table 2.8.5.1-7: Function Summary – read_controlpts(fp, num)

This function loads control points into PointTable.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h
num	int	standard

Calls	
Function	Where Described
get free controlpt	Controlpt.c
insert controlpt	Controlpt.c

Table 2.8.5.1-8: Function Summary – read_symbols(fp, num)

This function loads unit symbols into SymTable.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h
num	int	standard

Calls	
Function	Where Described
get free symbol	symbol.c
insert symbol	symbol.c

Table 2.8.5.1-9: Function Summary – list_over_files()

This function lists existing overlay files on the screen.

2.9 POPUP WINDOWS CSC DESCRIPTION

This CSC provides moveable popup windows with kill boxes.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-9.

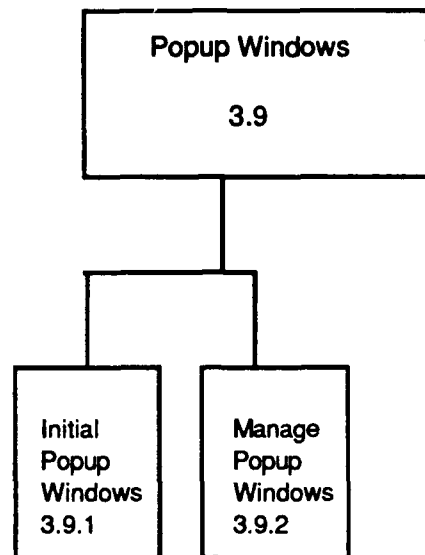


Figure 2-9: Popup Windows CSC Structure

2.9.1 Initial Popup Windows CSC Description

This CSC is the main entry for Popup Windows.

This CSC functionality is broken down into the CSUs shown in Figure 2.9.1-1.

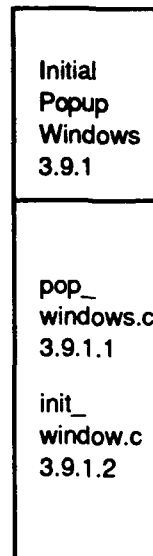


Figure 2.9.1-1: Popup Windows--Initial Popup Windows CSC Structure.

2.9.1.1 pop_windows.c

This CSU manipulates popup windows. It interprets the user's gestures with the mouse and invokes the appropriate actions. This file supports movable popup windows with kill boxes.

Table 2.9.1.1-1: Function Summary – pop_windows_present()

This function returns the number of popup windows on the screen.

Return Values		
Return Value	Type	Meaning
windows.active	int	number of windows on screen

Table 2.9.1.1-2: Function Summary – init_popup()

This function initializes popup windows.

Calls	
Function	Where Described
mgidew	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgibank	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgibbgetview	Concurrent graphics library subroutine
dump_windows	this file

Table 2.9.1.1-3: Function Summary – setup_new_window
 (title, user_x_size, user_y_size, place_x, place_y,
 real_xl, real_yb, real_xr, real_yt)

This function returns a new window pointer, or 0 if it fails.

Parameters		
Parameter	Type	Where Typedef Declared
title	pointer to char	standard
user x size, user y size	int	standard
place x, place y	int	standard
real xl, real yb	float	standard
real xr, real yt	float	standard

Return Values		
Return Value	Type	Meaning
window_ptr	pointer to struct POP_WIN	returns new window pointer

Calls	
Function	Where Described
create_window	Pop_windows.c
mgrvcoor	Concurrent graphics library subroutine
draw_window	this file

Table 2.9.1.1-4: Function Summary – delete_window(window_ptr)

This function deletes a popup window.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Return Values		
Return Value	Type	Meaning
-1	int	cannot delete window; all windows are free

Calls	
Function	Where Described
erase window	Pop windows.c
dump windows	this file

**Table 2.9.1.1-5: Function Summary – translate_window
(window_ptr, x_offset, y_offset)**

This function offsets the internal location of the database and moves the user's MassComp window.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h
x_offset, y_offset	int	standard

Calls	
Function	Where Described
translate_region	this file
mgimw	Concurrent graphics library subroutine

Table 2.9.1.1-6: Function Summary – translate_region(region, x, y)

Parameters		
Parameter	Type	Where Typedef Declared
region	pointer to struct RECTANGULAR_REGION	pop_window.h
x, y	int	standard

Table 2.9.1.1-7: Function Summary – create_window(title, user_x_size, user_y_size, place_x, place_y, death_function)

This function creates a popup window.

Parameters		
Parameter	Type	Where Typedef Declared
title	pointer to char	standard
user_x_size, user_y_size	int	standard
place_x, place_y	int	standard
death_function	int	standard

Return Values		
Return Value	Type	Meaning
0	int	window not created
new_window	pointer to struct POP_WIN	window created

Calls	
Function	Where Described
check_placability	this file
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
place_new_window	this file
retile_screen	this file
mgipw	Concurrent graphics library subroutine
dump_windows	this file

Table 2.9.1.1-8: Function Summary – erase_user_box(window_ptr)

This function sets the user portion of the window to background color.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.9.1.1-9: Function Summary – erase_window(window_ptr)

This function erases a popup window.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

Table 2.9.1.1-10: Function Summary – big_font_data(height, width, offset, hpad, vpad)

Parameters		
Parameter	Type	Where Typedef Declared
height	pointer to int	standard
width	pointer to int	standard
offset	pointer to int	standard
hpad	pointer to int	standard
vpad	pointer to int	standard

Calls	
Function	Where Described
mgigetginfo	Concurrent graphics library subroutine

Table 2.9.1.1-11: Function Summary – draw_window(window_ptr)

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgil	Concurrent graphics library subroutine
big_font_data	this file
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.9.1.1-12: Function Summary – copy_array(from, to, num longs)

This function copies arrays.

Parameters		
Parameter	Type	Where Typedef Declared
from	array of int	standard
to	array of int	standard
num longs	int	standard

Table 2.9.1.1-13: Function Summary – get_window_num(window_ptr)

This function maps a window pointer into a MassComp window number.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Return Values		
Return Value	Type	Meaning
window_ptr->window_num	int	window number

Table 2.9.1.1-14: Function Summary – handle_popup_windows(mouse)

This function returns 1 if the mouse still needs to be handled, 0 if successfully handled here.

Parameters		
Parameter	Type	Where Typedef Declared
mouse	pointer to struct MOUSE_EVENT	pvd_flags.h

Return Values		
Return Value	Type	Meaning
0	int	mouse handled here
1	int	mouse still needs to be handled

Calls	
Function	Where Described
point_in_box	this file
move_pop_window	this file

Table 2.9.1.1-15: Function Summary – point_in_box(x, y, box)

This function returns 1 if the passed point is in the indicated box.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard
box	pointer to struct RECTANGULAR_REGION	pop_window.h

Return Values		
Return Value	Type	Meaning
1	int	passed point is in the box
0	int	passed point not in box

**Table 2.9.1.1-16: Function Summary – move_pop_window
(win, initial_x, initial_y)**

This function moves popup windows by copying via the back frame buffer.

Parameters		
Parameter	Type	Where Typedef Declared
win	pointer to struct POP_WIN	pop_window.h
initial_x, initial_y	int	standard

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgimodfunc	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
offset_box	this file
mggetcursxy	Concurrent graphics library subroutine
check_placability	this file
move via back fb	this file

**Table 2.9.1.1-17: Function Summary – check_placability
(x, y, x_minus, y_minus, x_plus, y_plus, win)**

This function returns 1 if a window may be placed at the passed location.

Parameters		
Parameter	Type	Where Typedef Declared
window_ptr	pointer to struct POP_WIN	pop_window.h

Return Values		
Return Value	Type	Meaning
0	int	window does not exist
1	int	window may be placed at the passed location

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
point in box	this file
box_overlaps_window	this file

Table 2.9.1.1-18: Function Summary - offset_box
(x, y, minus_x, minus_y, plus_x, plus_y)

This function draws a box relative to the passed point.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard
minus_x, minus_y	int	standard
plus_x, plus_y	int	standard

Calls	
Function	Where Described
mgil	Concurrent graphics library subroutine

Table 2.9.1.1-19: Function Summary - box_overlaps_window
(xl, yb, xr, yt, win)

Parameters		
Parameter	Type	Where Typedef Declared
xl, yb, xr, yt	int	standard
win	pointer to struct POP WIN	pop_window.h

Return Values		
Return Value	Type	Meaning
1	int	box overlaps window
0	int	box does not overlap window

Calls	
Function	Where Described
point_in_box	this file

Table 2.9.1.1-20: Function Summary - move_via_back_fb
(win, x_offset, y_offset)

This function relocates a window. The old location of the window is erased, and underlying windows are not repainted.

Parameters		
Parameter	Type	Where Typedef Declared
win	pointer to struct POP WIN	pop_window.h
x_offset, y_offset	int	standard

Calls	
Function	Where Described
copy_nonmap_back	New_zoom.c
mgibank	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgifb	Concurrent graphics library subroutine
erase_window	this file
mgibblt2	Concurrent graphics library subroutine
translate_window	this file
copy_planes_back	this file

Table 2.9.1.1-21: Function Summary – copy_planes_back(plane_mask)

Parameters		
Parameter	Type	Where Typedef Declared
plane_mask	int	standard

Calls	
Function	Where Described
mgibank	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgibblt2	Concurrent graphics library subroutine
mgifbfbex	Concurrent graphics library subroutine

Table 2.9.1.1-22: Function Summary – place_new_window
(x, y, width, height)

This function finds an acceptable place for a new window of the passed size. It first scans the screen in WINDOW_STEP increments. If it fails to find a place, it retiles the screen and tries again.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	pointer to int	standard
width, height	int	standard

Return Values		
Return Value	Type	Meaning
1	int	successful
0	int	not successful

Calls	
Function	Where Described
check_placability	this file

Table 2.9.1.1-23: Function Summary – retile_screen()

This function sorts the windows on the screen by area and rearranges them so as to allow maximum area for new windows.

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgibank	Concurrent graphics library subroutine
mgigetfb	Concurrent graphics library subroutine
copy_planes_back	this file
mgifb	Concurrent graphics library subroutine
mgiclearpln	Concurrent graphics library subroutine
place_new_window	this file
mgibblt2	Concurrent graphics library subroutine
translate_window	this file

Table 2.9.1.1-24: Function Summary – dump windows()

This function prints out the current window structure on standard out.

2.9.1.2 init_window.c

This CSU sets up the three colors available to the window system.

Table 2.9.1.2-1: Function Summary – init_window_colors()

This function sets up the three colors available to the window system.

Calls	
Function	Where Described
make_color_map	Color_map.c

2.9.2 Manage Popup Windows CSC Description

This CSC manages popup windows, which are used to contain certain menu options. This CSC functionality is broken down into the CSUs shown in Figure 2.9.2-1.

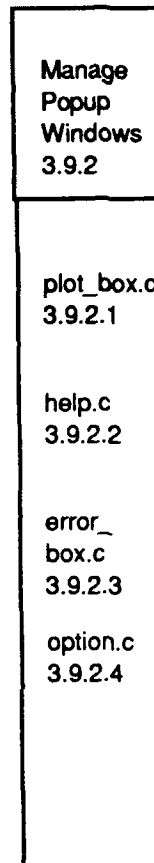


Figure 2.9.2-1: Popup Windows--Manage Popup Windows CSC Structure.

2.9.2.1 plot_box.c

This CSU sets up a plotting window, with axes.

Table 2.9.2.1-1: Function Summary – plot_box
(window, back_color, pen_color, place, margins, axes,
homogeneous, x_tic, y_tic, x_label, y_label)

This function sets up a plotting window, with axes. It assumes a window has been defined but not placed.

Parameters		
Parameter	Type	Where Typedef Declared
window	int	standard
back_color	int	standard
pen_color	int	standard
place	pointer to struct int_quad	this file
margins	pointer to struct quad	this file
axes	pointer to struct quad	this
homogeneous	int	standard
x_tic, y_tic	float	standard
x_label, y_label	pointer to char	standard

Calls	
Function	Where Described
mgrvcoor	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
mgrl1	Concurrent graphics library subroutine
label_point	
mgisyncrb	Concurrent graphics library subroutine

2.9.2.2 help.c

This CSU displays and removes a popup window for displaying Help information on the screen.

Table 2.9.2.2-1: Function Summary – help()

This function displays the contents of an ascii file in the help window.

Calls	
Function	Where Described
help_menu_entry_on	Edit_menu.c
delete_window	Pop_windows.c
create_window	Pop_windows.c
get_window_num	Pop_windows.c
make_path_name	Init_env.c
get_data_directory	Init_env.c
display_window_text	this file
help_menu_entry_off	Edit_menu.c

Table 2.9.2.2-2: Function Summary – kill_help_window()

This function deletes the help window.

Calls	
Function	Where Described
help_menu_entry_on	Edit_menu.c
delete_window	Pop_windows.c

Table 2.9.2.2-3: Function Summary – display_window_text (text_file, title, window)

This function displays text in the help window. It first opens the text file, then replaces the window to fit the text.

Parameters		
Parameter	Type	Where Typedef Declared
text_file	pointer to char	standard
title	pointer to char	standard
window	pointer to struct POP_WIN	pop_window.h

Calls	
Function	Where Described
find_max_chars	this file
mgigetvcoor	Concurrent graphics library subroutine
get_window_num	Pop_windows.c
draw_window	Pop_windows.c
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine
strip_char	this file

Table 2.9.2.2-4: Function Summary – display_window_string (msg_text, window, ignore_nl)

This function prints the specified string in the specified window.

Parameters		
Parameter	Type	Where Typedef Declared
msg_text	pointer to char	standard
window	int	standard
ignore_nl	int	standard

Calls	
Function	Where Described
mgigetvcoor	Concurrent graphics library subroutine
draw_window	Pop_windows.c
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgigl	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

Table 2.9.2.2-5: Function Summary – strip_char(string, character)

This function removes any single type of character from a string.

Parameters		
Parameter	Type	Where Typedef Declared
string	pointer to char	standard
character	point to char	standard

Table 2.9.2.2-6: Function Summary – find_max_chars(fp)

This function finds the maximum width of an open file, including tabs, etc.

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h

Calls	
Function	Where Described
check_help	
help_menu_entry_on	Edit_menu.c

2.9.2.3 error_box.c

This CSU displays and removes error messages in a popup window on the screen.

Table 2.9.2.3-1: Function Summary – error_box(text_file, error_title)

This function puts an error message on the screen in a popup window.

Parameters		
Parameter	Type	Where Typedef Declared
text_file	pointer to char	standard
error_title	pointer to char	standard

Return Values		
Return Value	Type	Meaning
-1	int	can't open error window; already exists

Calls	
Function	Where Described
create_window	Pop_windows.c
make_path_name	Init_env.c
get_data_directory	Init_env.c
display_window_text	Help.c

Table 2.9.2.3-2: Function Summary – kill_error_window()

This function removes an error popup window from the screen.

Calls	
Function	Where Described
delete_window	Pop_windows.c

Table 2.9.2.3-3: Function Summary – error_window_exists()

Return Values		
Return Value	Type	Meaning
error_window != 0	int	error window exists

Table 2.9.2.3-4: Function Summary – check_error_window()

This function checks whether an error window exists.

Calls	
Function	Where Described
error_window_exists	this file
kill_error_window	this file

**Table 2.9.2.3-5: Function Summary – string_error_box
(message_text, window_title)**

This function puts an error message on the screen in a popup window, taking a string argument for the message text and allowing specification of the window bar title.

Parameters		
Parameter	Type	Where Typedef Declared
message_text	pointer to char	standard
window_title	pointer to char	standard

Return Values		
Return Value	Type	Meaning
-1	int	error window exists

Calls	
Function	Where Described
create_window	Pop_windows.c
display_window_string	Help.c

2.9.2.4 option.c

This CSU contains routines for defining regions within the popup window, displaying options labels with corresponding select boxes, and for providing for mouse selection of options.

**Table 2.9.2.4-1: Function Summary – option_reg_define
(PopWindow, Xl, Yb, Xr, Yt)**

This function defines a region within a popup window in which to display input options.

Parameters		
Parameter	Type	Where Typedef Declared
PopWindow	pointer to struct POP_WIN	pop_window.h
Xl, Yb, Xr, Yt	int	standard

Return Values		
Return Value	Type	Meaning
OptionRegion	pointer to struct OPTION_REGION	set up option with no options blocks defined or displayed

**Table 2.9.2.4-2: Function Summary – define_block
(OptionRegion, OptionList, Title)**

This function takes a previously defined OptionRegion, a list of options, and a title string and returns a relocatable option block if it fits in the given OptionRegion.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
OptionList	pointer to struct OPTION	option.h
Title	pointer to char	standard

Return Values		
Return Value	Type	Meaning
NULL	int	block too big for this region
OptionBlock	pointer to struct OPTION_BLOCK	option block successfully defined

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
get_window_num	Pop_windows.c

**Table 2.9.2.4-3: Function Summary – display_block
(OptionRegion, OptionBlock)**

This function displays a previously defined block. It activates the block and displays the option labels with their corresponding select boxes.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
OptionBlock	pointer to struct OPTION_BLOCK	option.h

Return Values		
Return Value	Type	Meaning
0	int	not enough room left for block
OptionRegion -> FreeYLine - OptionRegion -> Area.yb	int	height in number of pixels left in this input region

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
get_window_num	Pop_windows.c
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
display_title	this file
mark_box	this file

**Table 2.9.2.4-4: Function Summary – erase_block
(OptionRegion, OptionBlock)**

This function erases the area corresponding to the width of the option region and the height of the given OptionBlock, including title. It also deactivates the block.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
OptionBlock	pointer to struct OPTION_BLOCK	option.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
get_window_num	Pop windows.c
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgibox	Concurrent graphics library subroutine

**Table 2.9.2.4-5: Function Summary – display_title
(OptionRegion, OptionBlock)**

This function displays the title corresponding to the given option block on top center of the already display labels.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
OptionBlock	pointer to struct OPTION_BLOCK	option.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
get_window_num	Pop windows.c
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

**Table 2.9.2.4-6: Function Summary – mouse_to_option
(OptionRegion, PosX, PosY)**

This function returns the option displayed closest to coordinates PosX, PosY.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
PosX, PosY	int	standard

Return Values		
Return Value	Type	Meaning
NULL	pointer to struct OPTION	coordinates do not correspond to a valid option
MinOption	pointer to struct OPTION	looks through all active option blocks to find one that encloses the chosen option

**Table 2.9.2.4-7: Function Summary – draw text label
(Text, Font, LimitX, StartY, StartX, Tallest)**

This function draws a select box and the label Text to the right of the box.

Parameters		
Parameter	Type	Where Typedef Declared
Text	pointer to char	standard
FontNum	int	standard
LimitX, StartY	int	standard
StartX, Tallest	pointer to int	standard

Return Values		
Return Value	Type	Meaning
0	int	cannot draw box and text starting at StartX within LimitX
1	int	can draw box and text

Calls	
Function	Where Described
mgigetginfo	Concurrent graphics library subroutine
draw_select_box	this file
mgigf	Concurrent graphics library subroutine
mgigfs	Concurrent graphics library subroutine

**Table 2.9.2.4-8: Function Summary – display_from_blk
(OptionRegion, StartBlock)**

This function displays as many blocks as it can, starting with StartBlock, until it cannot fit any more or until it has displayed all the blocks after StartBlock.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h
StartBlock	pointer to struct OPTION_BLOCK	option.h

Return Values		
Return Value	Type	Meaning
CurBlock	int	assigns value of start block to CurBlock

Calls	
Function	Where Described
erase_region	this file
display_block	this file

Table 2.9.2.4-9: Function Summary – erase_region(OptionRegion)

This function deletes all the blocks currently being displayed. It does not reset the options in the region.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h

Calls	
Function	Where Described
erase_block	this file

Table 2.9.2.4-10: Function Summary – reset_region(OptionRegion)

This function is called when the region is no longer used; it unselects all options.

Parameters		
Parameter	Type	Where Typedef Declared
OptionRegion	pointer to struct OPTION_REGION	option.h

Calls	
Function	Where Described
erase_region	this file

**Table 2.9.2.4-11: Function Summary – draw_select_box
(LimitX, StartX, StartY)**

This function draws a box with coordinates StartX, StartY as the upper left corner.

Parameters		
Parameter	Type	Where Typedef Declared
LimitX	int	standard
StartX, StartY	int	standard

Return Values		
Return Value	Type	Meaning
0	int	width of box larger than LimitX
BOX_WIDTH	int	width of box

Calls	
Function	Where Described
mgils	Concurrent graphics library subroutine

Table 2.9.2.4-12: Function Summary – mark_box(Block, Index)

This function places an X in the select box that corresponds to option in Options[Index].

Parameters		
Parameter	Type	Where Typedef Declared
Block	pointer to OPTION_BLOCK	option.h
Index	int	Concurrent graphics library

Calls	
Function	Where Described
mgil	Concurrent graphics library subroutine

2.10 TOOLS CSC DESCRIPTION

This CSC provides map tools that include ruler, intervisibility options, and terrain cross-section.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-10.

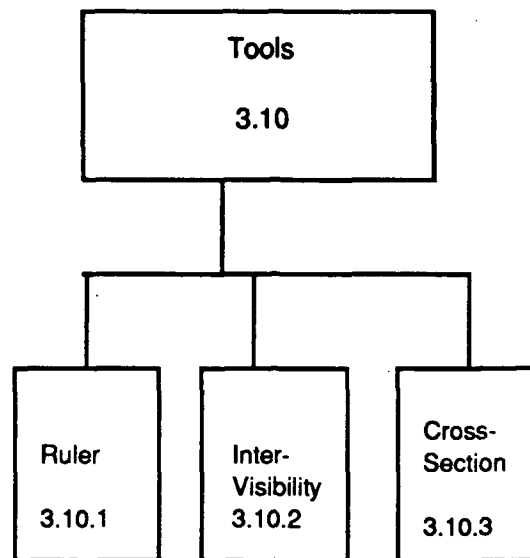


Figure 2-10: Tools CSC Structure.

2.10.1 Ruler CSC Description

This CSC measures the distance between locations.

This CSC functionality is broken down into the CSUs shown in Figure 2.10.1-1.

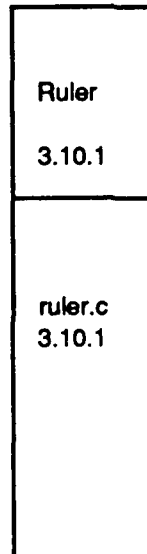


Figure 2.10.1-1: Tools--Ruler CSC Structure.

2.10.1.1 ruler.c

This CSU supports measuring distances on the map.

Table 2.10.1.1-1: Function Summary – text_ruler(x0, y0, x1, y1)

This function displays the distance in meters between two passed points.

Parameters		
Parameter	Type	Where Typedef Declared
x0, y0	int	standard
x1, y1	int	standard

Calls	
Function	Where Described
mgjunscalexy	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgifc	Concurrent graphics library subroutine

2.10.2 Intervisibility CSC Description

This CSC performs several types of intervisibility calculations based on user-supplied input. Types of intervisibility are area from point, point to vehicle, vehicle to vehicle, and point to point. This CSC also calculates intervisibility to enemy vehicles. It uses target and view heights and view range as inputs.

This CSC functionality is broken down into the CSUs shown in Figure 2.10.2-1.

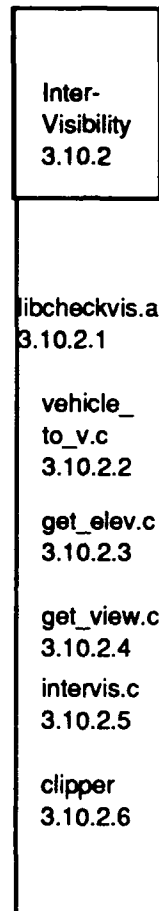


Figure 2.10.2-1: Tools—Intervisibility CSC Structure.

2.10.2.1 libcheckvis.a CSU Description (/simnet/lib)

This library supports intervisibility calculations. It is documented in "Vehicles CSCI Software Design Document."

2.10.2.2 vehicle_to_v.c

This CSU manages startup and shutdown of vehicle to vehicle intervisibility tagged to a moving vehicle. It updates intervisibility lines.

**Table 2.10.2.2-1: Function Summary – do_vehicle_to_vehicle
(selected_vehicles)**

This function handles startup or shutdown of vehicle to vehicle intervisibility tagged to a moving vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
selected_vehicles	SELECTED_VEHICLES	select.h

Calls	
Function	Where Described
erase_rays	this file
encode_force	this file
intervis_vehicles	Intervis.c
get_world_state	
get_view_height	Get view.c
get_target_height	Get view.c

**Table 2.10.2.2-2: Function Summary – update_vehicle_to_vehicle
(current_time)**

Parameters		
Parameter	Type	Where Typedef Declared
current_time	int	standard

Calls	
Function	Where Described
erase_rays	this file
encode_force	this file
intervis_vehicles	Intervis.c
get_view_height	Get view.c
get_target_height	Get View.c

Table 2.10.2.2-3: Function Summary – erase_rays(orgin, num_rays, ends)

Parameters		
Parameter	Type	Where Typedef Declared
orgin	pointer to POINT	v_vis.h
num_rays	int	standard
ends	array of POINT	v_vis.h

Calls	
Function	Where Described
mgiv	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgipIn	Concurrent graphics library subroutine
mgri	Concurrent graphics library subroutine

Table 2.10.2.2-4: Function Summary – get_tags()

Return Values		
Return Value	Type	Meaning
tags	pointer to struct V_V_VISIBILITY	local static variable

Table 2.10.2.2-5: Function Summary – set_tags(new_tags)

Parameters		
Parameter	Type	Where Typedef Declared
new_tags	pointer to struct V_V_VISIBILITY	v_vis.h

Table 2.10.2.2-6: Function Summary – clear_all_veh_to_veh()

This function turns off vehicle to vehicle intervisibility for all vehicles.

Calls	
Function	Where Described
erase_rays	this file

Table 2.10.2.2-7: Function Summary – encode_force(force)

This function returns the forces to view given that user is of the passed force.

Parameters		
Parameter	Type	Where Typedef Declared
force	int	standard

Return Values		
Return Value	Type	Meaning
0xA	int	distinguishedForceID
0x9	int	otherForceID
0x8	int	observerForceID
0x3	int	targetForceID
0xF	int	default ALL_VEHICLES

Table 2.10.2.2-8: Function Summary – view_all_align()

This function assigns visibility mode to ALL_VEHICLES.

Table 2.10.2.2-9: Function Summary – view_opposing_align()

This function assigns visibility mode to OPPOSING_VEHICLES.

2.10.2.3 get_elev.c

This CSU returns elevation in meters of any point on the map.

Table 2.10.2.3-1: Function Summary – get_elevation(x, y)

This function returns the elevation in meters of any point on the map.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	float	standard

Return Values		
Return Value	Type	Meaning
0.0	float	error
location.z	float	returns location

Calls	
Function	Where Described
tdb_get_z	see libtdb in MCC CSCI
tdb_error	see libtdb in MCC CSCI

2.10.2.4 get_view.c

This CSU gets the height of the top of the target vehicle above the ground.

Table 2.10.2.4-1: Function Summary – get_view_height()

Return Values		
Return Value	Type	Meaning
view_height	float	returns height of eye level above ground

Table 2.10.2.4-2: Function Summary – get_target_height()

Return Values		
Return Value	Type	Meaning
target_height	float	returns height of top of target vehicle above ground

Table 2.10.2.4-3: Function Summary – set_view_height(height)

Parameters		
Parameter	Type	Where Typedef Declared
height	float	standard

Table 2.10.2.4-4: Function Summary - set_target_height(height)

Parameters		
Parameter	Type	Where Typedef Declared
height	float	standard

Table 2.10.2.4-5: Function Summary - set_view_range(range)

Parameters		
Parameter	Type	Where Typedef Declared
range	float	standard

Table 2.10.2.4-6: Function Summary - get_view_range()

Return Values		
Return Value	Type	Meaning
view_range	float	returns maximum visibility range

2.10.2.5 intervis.c

This CSU supports the various intervisibility menu options.

Table 2.10.2.5-1: Function Summary - intervisibility(from_x, from_y, to_x, to_y, view_height, target_height)

This function determines intervisibility between two points, using straight line (not segmented) visibility. The result is printed in the info window ("Not visible," etc.) and a line segment of the appropriate color is drawn on the map.

Parameters		
Parameter	Type	Where Typedef Declared
from_x, from_y	int	standard
to_x, to_y	int	standard
view_height	float	standard
target_height	float	standard

Calls	
Function	Where Described
mgjunscalexy	Concurrent graphics library subroutine
pve checkvis	
mgiv	Concurrent graphics library subroutine
mgipin	Concurrent graphics library subroutine
set ray color	this file
print visibility code	this file
mgrl	Concurrent graphics library subroutine
draw x	this file

Table 2.10.2.5-2: Function Summary – init_intervisibility

This function initializes the intervisibility function.

Calls	
Function	Where Described
get_patch_guards	

Table 2.10.2.5-3: Function Summary – new_intervis (from_x, from_y, to_x, to_y, view_height, target_height)

This function determines point to point intervisibility, using segmented rays.

Parameters		
Parameter	Type	Where Typedef Declared
from x, from y	int	standard
to x, to y	int	standard
view height	float	standard
target height	float	standard

Calls	
Function	Where Described
mgjunscalexy	Concurrent graphics library subroutine
get_elevation	Get_elev.c
mgiv	Concurrent graphics library subroutine
mgjpln	Concurrent graphics library subroutine
pve_checkvis	
print_visibility_code	this file
set_ray_color	this file
draw_x	this file
draw_coded_line	this file

Table 2.10.2.5-4: Function Summary – draw_coded_line (segments, num_segments, final_x, final_y)

Parameters		
Parameter	Type	Where Typedef Declared
segments	array of PV CHANGE PT	pvis_call.h
num_segments	int	standard
final_x, final_y	float	standard

Calls	
Function	Where Described
set_ray_color	this file
mgrl	Concurrent graphics library subroutine

Table 2.10.2.5-5: Function Summary – set_ray_color (visibility_code)

This function sets the default color to the appropriate one for the visibility code.

Parameters		
Parameter	Type	Where Typedef Declared
visibility_code	int	standard

Calls	
Function	Where Described
mgihue	Concurrent graphics library subroutine

Table 2.10.2.5-6: Function Summary – intervis_360
 (from_x, from_y, num_rays, max_range,
 view_height, target_height)

This function draws intervisibility lines out from the passed point to points on a circle of radius 3500 meters. The lines show the possibility of seeing an object "target_height" above the ground.

Parameters		
Parameter	Type	Where Typedef Declared
from_x, from_y	int	standard
num_rays	int	standard
max_range	float	standard
view_height	float	standard
target_height	float	standard

Calls	
Function	Where Described
mgjunscalexy	Concurrent graphics library subroutine
mgrrsi	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
rayvis	
pve checkvis	
mgrl	Concurrent graphics library subroutine

Table 2.10.2.5-7: Function Summary – real_intervis_360(from_x, from_y, num_rays, max_range, view_height, target_height)

This function draws area intervisibility rays out from the passed point. The lines show the possibility of seeing an object's "target_height" above the ground.

Parameters		
Parameter	Type	Where Typedef Declared
from_x, from_y	int	standard
num_rays	int	standard
max_range	float	standard
view_height	float	standard
target_height	float	standard

Calls	
Function	Where Described
get_patch_guards	
mgjunscalexy	Concurrent graphics library subroutine
mgrrsi	Concurrent graphics library subroutine
get_elevation	Get elev.c
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
pve_checkvis	
draw_coded_line	this file

Table 2.10.2.5-8: Function Summary – intervis_vehicles (from_xm, from_ym, from_zm, world_state, view_height, target_height, side_code, tags, print)

This function draws intervisibility lines from the selected point to all existing vehicles. The view height is relative to the vehicle's coordinate origin.

Parameters		
Parameter	Type	Where Typedef Declared
from_xm, from_ym, from_zm	float	standard
world_state	pointer to struct WORLD_STATE	world_state.h
view_height	float	standard
target_height	float	standard
side_code	int	standard
tags	pointer to struct V V VISIBILITY	v_vis.h
print	int	standard

Calls	
Function	Where Described
get_patch_guards	
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
pve_checkvis	
mgihue	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
draw_x	this file

**Table 2.10.2.5-9: Function Summary – emb_attenuate
(from_x, from_y, to_y, view_height, target_height)**

This function determines intervisibility between two points, using straight line (not segmented) visibility. The result is printed in the info window ("Not visible," etc.) and a line segment of the appropriate color is drawn on the map.

Parameters		
Parameter	Type	Where Typedef Declared
from_x, from_y	int	standard
to_x, to_y	int	standard
view_height	float	standard
target_height	float	standard

Calls	
Function	Where Described
mgijunscalexy	Concurrent graphics library subroutine
get_elevation	Get_elev.c
pve_checkvis	
mgiv	Concurrent graphics library subroutine
mgipln	Concurrent graphics library subroutine
set_ray_color	this file
print_visibility_code	this file
mgrl	Concurrent graphics library subroutine
draw_x	this file
attenuation_factor	this file

Table 2.10.2.5-10: Function Summary – draw_x(x, y)

This function marks a spot with an x.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard

Calls	
Function	Where Described
mgil	Concurrent graphics library

**Table 2.10.2.5-11: Function Summary – attenuation_factor
(distance, frequency)**

This function computes the attenuation factor for radio transmissions partially blocked by terrain.

Parameters		
Parameter	Type	Where Typedef Declared
distance, frequency	double	standard

Return Values		
Return Value	Type	Meaning
1.0	static double	wavelength too close; distance too small
$4.0 + \text{distance} * 195$	static double	distance < .01
$5.12 + \text{distance} * 83.5$	static double	distance < .09
$5.86 + \text{distance} * 75.23$	static double	distance < 1.0
$2.47 + \text{distance} * 78.629$		

**Table 2.10.2.5-12: Function Summary – print_visibility_code
(fp, visibility_code)**

Parameters		
Parameter	Type	Where Typedef Declared
fp	pointer to FILE	stdio.h
visibility_code	int	standard

2.10.2.6 clipper.c

This CSU contains routines for determining the clipping edges of a map.

Table 2.10.2.6-1: Function Summary – clip_to_map(inside_pt, test_pt)

Parameters		
Parameter	Type	Where Typedef Declared
inside_pt	array of double	standard
test_pt	array of double	standard

Calls	
Function	Where Described
clip_to_rect	this file
dist2	Measures.c

Table 2.10.2.6-2: Function Summary – clip_to_rect(edge_code, start_pt, end_pt, x_min, x_max, y_min, y_max, x_inter, y_inter)

This function determines if a point is outside a clipping edge, where the clipping edge is one of the boundary edges of a rectangular 2-D window. If so, it computes the intersection.

Parameters		
Parameter	Type	Where Typedef Declared
edge_code	int	standard
start_pt, end_pt	array of float	standard
x_min, x_max	float	standard
y_min, y_max	float	standard
x_inter, y_inter	pointer to float	standard

Return Values		
Return Value	Type	Meaning
0	int	line does not cross window boundary
1	int	line crosses window boundary

2.10.3 CrossSection CSC Description

This CSC supports toggling to display or remove a plot of the elevation between two points in the terrain.

This CSC functionality is broken down into the CSUs shown in Figure 2.10.3-1.

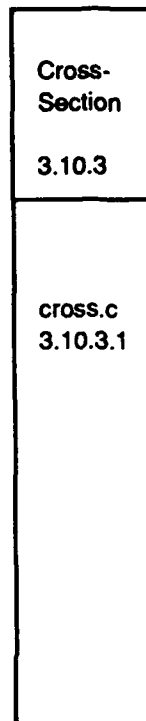


Figure 2.10.3-1: Tools--Cross-Section CSC Structure.

2.10.3.1 cross.c

This CSU plots a cross-section through the terrain in a pop-up window.

Table 2.10.3.1-1: Function Summary – cross_section(x, y, x_, y_)

This function plots a cross section through the terrain. The cross section is normalized in elevation and the vertical scale exaggeration is integral.

Parameters		
Parameter	Type	Where Typedef Declared
x, y	int	standard
x_, y_	int	standard

Calls	
Function	Where Described
cross menu entry on	Edit menu.c
delete window	Pop windows.c
error box	Error box.c
create window	Pop windows.c
cross menu entry off	Edit menu.c
mgrsi	Concurrent graphics library subroutine
compute elevation segs	this file
mgipln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
get window num	Pop windows.c
draw window	Pop windows.c
mgrvcoor	Concurrent graphics library subroutine
plot box	Plot box.c
mgihue	Concurrent graphics library subroutine
mgrls	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgidash	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine
get target height	Get view.c
get view height	Get view.c

**Table 2.10.3.1-2: Function Summary – colin_cross
(x0, y0, z0, x1, y1, z1)**

Parameters		
Parameter	Type	Where Typedef Declared
x0, y0, z0, x1, y1, z1	float	standard

Calls	
Function	Where Described
cross menu entry on	Edit menu.c
delete window	Pop windows.c
create window	Pop windows.c
cross menu entry off	Edit menu.c
mgrsi	Concurrent graphics library subroutine
compute elevation segs	this file
mgipln	Concurrent graphics library subroutine
mgiv	Concurrent graphics library subroutine
get window num	Pop windows.c
draw window	Pop windows.c
mgrvcoor	Concurrent graphics library subroutine
mgihue	Concurrent graphics library subroutine
mgris	Concurrent graphics library subroutine
mgisyncrb	Concurrent graphics library subroutine
mgidash	Concurrent graphics library subroutine
mgrl	Concurrent graphics library subroutine

Table 2.10.3.1-3: Function Summary – kill_cross_window()

This function deletes a cross section window.

Calls	
Function	Where Described
cross menu entry on	edit menu.c
delete window	Pop windows.c

Table 2.10.3.1-4: Function Summary – compute_elevation_segs
 (from_x, from_y, to_x, to_y, num_lines, x_array,
 z_array, min_elevation, max_elevation)

This function returns an array of elevation points along a given segment.

Parameters		
Parameter	Type	Where Typedef Declared
from_x, from_y	float	standard
to_x, to_y	float	standard
num_lines	pointer to int	standard
x_array	array of float	standard
z_array	array of float	standard
min_elevation	pointer to float	standard
max_elevation	pointer to float	standard

Calls	
Function	Where Described
get_elevation	Get_elev.c

2.11 REMOTE DEVICES INTERFACES

This CSC sets up the communications data structures for interfacing with the Data Logger and Stealth over the SIMNET network. For the Data Logger it also sends and receives messages. This CSC uses logger protocol to communicate with the Data Logger and Stealth protocol to initialize and process Stealth packets. This CSC accesses basic SIMNET network services to mediate communication with the Stealth and the Logger.

Event Waypoints

The PVD provides a set of menu options, 'event waypoints,' for annotating an exercise playback. During an exercise, an analyst may want to annotate the recording of the exercise with event type information not described by any of the available SIMNET message formats. Selection of an event waypoint causes a text string message, called an event flag Protocol Data Unit (PDU), to be recorded in sequence with the other PDUs being sent during the exercise. A technician may reprogram event waypoints to send other text strings by editing the file, events.def. Event flag PDUs are defined by the SIMNET data collection protocol as described in "SIMNET Network and Protocols," BBN Report No. 1702.

This CSC is designed to hide the details of remote device communications from other CSC functions.

Logger

The PVD controls communications with the Data Logger and its exercise playback operations.

Stealth

The PVD controls communications with the Stealth and its operations. Initially, the technician must reconfigure the PVD system to make the Stealth menu options—which include Teleport and Attach—available.

This CSC functionality is broken down into lower-level CSCs shown in Figure 2-11.

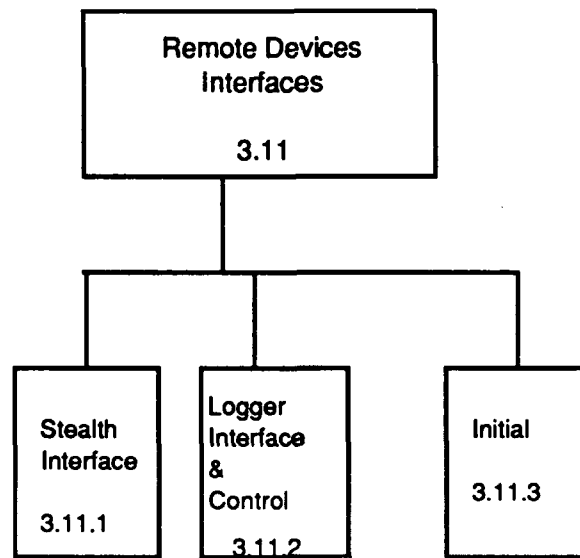


Figure 2-11: Remote Devices Interfaces CSC Structure.

2.11.1 Stealth Interface CSC Description

This CSC initializes, dispatches, processes, and stores Stealth messages. This CSC also makes use of the functions in the Network Processing CSC (Section 2.6).

This CSC functionality is broken down into the CSUs shown in Figure 2.11.1-1.

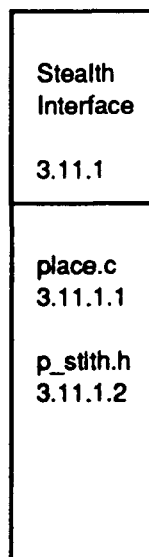


Figure 2.11.1-1: Remote Devices Interfaces--Stealth Interface CSC Structure.

2.11.1.1 place.c

This CSU provides control of the stealth vehicle.

**Table 2.11.1.1-1: Function Summary – send_event_flag
(vehicle_id, code, flag_string)**

This function broadcasts an event_flag packet.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to VehicleID	basic.h
code	int	standard
flag_string	pointer to char	standard

Calls	
Function	Where Described
get_vehicle_exercise	
AssocSendDatagram	libassoc.a in MCC CSCI
print_vehicle_id	pvd_misc.c

**Table 2.11.1.1-2: Function Summary – teleport_stealth
(new_x, new_y, new_z, azimuth)**

This function sends a teleport message point to point to the stealth vehicle specified by the ethernet address in the pvd_init.dat file.

Parameters		
Parameter	Type	Where Typedef Declared
new_x, new_y, new_z	float	standard
azimuth	Angle	basic.h

Calls	
Function	Where Described
PointToPointSendPDU	see libassoc in Vehicle Simulation CSCI
get_stealth_address	init_env.c

Table 2.11.1.1-3: Function Summary – attach_stealth(vehicle_id)

This function sends a packet commanding the stealth vehicle to become attached to the specified vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	pointer to VehicleID	basic.h

Calls	
Function	Where Described
PointToPointSendPDU	see libassoc in Vehicle Simulation CSCI
get_stealth_address	init_env.c
print_vehicle_id	pvd_misc.c

Table 2.11.1.1-4: Function Summary – mimic_vehicle (vehicle_id, channel)

This function sends a mimic packet (point to point) to the stealth vehicle specified in the pvd_init.dat file. The mimic packet causes the stealth vehicle to become co-located with the vehicle with the passed vehicle_id.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	int	standard
channel	int	standard

Calls	
Function	Where Described
get_stealth_net_address	
net_write	libnetif.a in MCC CSCI

Table 2.11.1.1-5: Function Summary – set_stealth_exercise(exercise_id)

This function assigns the stealth vehicle to a particular exercise. Upon receipt of the packet, Stealth should only show vehicles in from that exercise.

Parameters		
Parameter	Type	Where Typedef Declared
exercise_id	int	standard

Calls	
Function	Where Described
PointToPointSendPDU	see libassoc in Vehicle Simulation CSCI
get_stealth_address	init_env.c

Table 2.11.1.1-6: Function Summary – handle_stealth_error(error_packet)

This function prints error messages received from the stealth vehicle.

Parameters		
Parameter	Type	Where Typedef Declared
error_packet	pointer to StealthErrorVariant	p_stlth.h

Table 2.11.1.1-7: Function Summary – handle_attached_packet (attached_packet)

Parameters		
Parameter	Type	Where Typedef Declared
attached_packet	pointer to AttachedVariant	p_stlth.h

Calls	
Function	Where Described
get_vehicle_index	
print_vehicle_id	pvd_misc.c
delete_attachment_icon	
create_attachment_icon	

Table 2.11.1.1-8: Function Summary – get_location(vehicle_id, x, y, z)

This function determines the current x, y, and z for a given vehicle ID.

Parameters		
Parameter	Type	Where Typedef Declared
vehicle_id	int	standard
x, y, z	pointer to float	standard

Table 2.11.1.1-9: Function Summary – construct_sim_from_stealth (stealth_pdu)

This function creates a vehicle appearance packet in the simulation protocol corresponding to the passed stealth appearance packet.

Parameters		
Parameter	Type	Where Typedef Declared
stealth_pdu	pointer to StealthPDU	p_stlth.h

Return Values		
Return Value	Type	Meaning
&sim_pdu	int	pointer to sim_pdu

2.11.1.2 p_stlth.h

This is a protocol file for the stealth vehicle. It is a commonly-maintained file used by Plan View Display and other SIMNET applications.

2.11.2 Logger Interface and Control CSC Description

This CSC sets up and verifies a separate network connection with the Logger. This CSC also initializes and dispatches Logger data and command messages. In addition, this CSC processes and stores playback exercise data. This CSC also makes use of the functions in the Network Processing CSC (Section 2.6).

This CSC functionality is broken down into the CSUs shown in Figure 2.11.2-1.

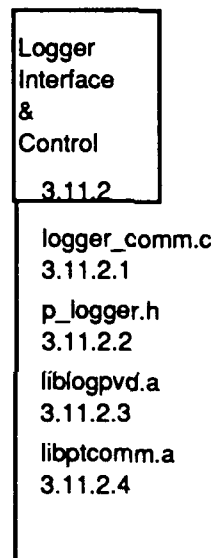


Figure 2.11.2-1: Remote Devices Interfaces--Logger Interface CSC Structure.

2.11.2.1 logger_comm.c

This CSU provides control of the logger.

Table 2.11.1.2-1: Function Summary – init_connections()

This function initializes logger functions.

Calls	
Function	Where Described
pvd phantom tank init	
get pt host	Init env.c
change phantom status	
init dp	
change dataprobe status	
pvd logger init	
get vr host	Init env.c
change logger status	
dl connected	

Table 2.11.1.2-2: Function Summary – get_loggerAvailReplys (buf, length, group, protocol, originator)

Parameters		
Parameter	Type	Where Typedef Declared
buf	pointer to LoggerPDU	p_logger.h
length	int	standard
group	MulticastGroupID	p_assoc.h
protocol	AssociationUserProtocol	p_assoc.h
originator	pointer to SimulationAddress	address.h

Return Values		
Return Value	Type	Meaning
1	int	handles the packet
0	int	left to someone else

Calls	
Function	Where Described
get vr host	Init env.c
print simulation address	pvd_misc.c

Table 2.11.1.2-3: Function Summary – find_loggers()

This function returns 1 if any loggers were found, 0 otherwise.

Return Values		
Return Value	Type	Meaning
1	int	loggers found
0	int	no loggers found

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
do_avail_request	Pvdrtc.c
send_rtc	send_rtc.c
set_text_window	T_windows.c
get_vr_host	Init_env.c
net_gettime	libnetif.a in MCC CSCI
process_nonappearance_packets	
init_cursor	Pvd_iface.c

**Table 2.11.1.2-4: Function Summary – get_loggerConnectAck
(buf, length, group, protocol, originator)**

This function processes replies to connection requests. It returns 0 if the packet is not handled.

Parameters		
Parameter	Type	Where Typedef Declared
buf	pointer to LoggerPDU	p_logger.h
length	int	standard
group	MulticastGroupID	p_assoc.h
protocol	AssociationUserProtocol	p_assoc.h
originator	pointer to SimulationAddress	address.h

Return Values		
Return Value	Type	Meaning
0	int	packet not handled
1	int	packet handled

Table 2.11.1.2-5: Function Summary – connect_to_logger()

This function connects to the requested logger.

Calls	
Function	Where Described
init_working_cursor	Pvd_iface.c
init_pdu	Pvdrtc.c
send_rtc	Pvdrtc.c
net_gettime	libnetif.a in MCC CSCI
process_nonappearance_packets	
change_logger_status	
init_cursor	Pvd_iface.c

Table 2.11.1.2-6: Function Summary – disconnect_dl()

This function disconnects from the current datalogger.

Calls	
Function	Where Described
do_disconnect	Pvdrtc.c
send_rtc	Pvdrtc.c
net_zeroaddr	libnetif.a in MCC CSCI
change_logger_status	

Table 2.11.1.2-7: Function Summary – play_logger()

This function provides initialization data to the logger for playback.

Calls	
Function	Where Described
get_typed_input_str	Interact.c
do_start_playback_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-8: Function Summary – record_logger()

This function moves the logger into the recording state.

Calls	
Function	Where Described
get_exercise	
get_current_time	
do_start_record_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-9: Function Summary – continue_logger()

This function sends a "Continue" command to the logger.

Calls	
Function	Where Described
do_continue_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-10: Function Summary – logger_freeze()

This function sends the "Suspend" command to the logger.

Calls	
Function	Where Described
do_suspend_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-11: Function Summary – logger_stop()

This function sends the "Stop" command to the logger.

Calls	
Function	Where Described
do_stop_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-12: Function Summary – logger_speed(speed)

Parameters		
Parameter	Type	Where Typedef Declared
speed	int	standard

Calls	
Function	Where Described
do_play_speed_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-13: Function Summary – logger_seek(seconds, relative_to_current_loc)

Parameters		
Parameter	Type	Where Typedef Declared
seconds	int	standard
relative_to_current_loc	int	standard

Calls	
Function	Where Described
do_seek_relative_command	Pvdrtc.c
send_rtc	Pvdrtc.c

Table 2.11.1.2-14: Function Summary – get_logger_address()

This function returns the logger's address.

Return Values		
Return Value	Type	Meaning
&Logger_S_Address	SimulationAddress	pointer to logger simulation address

Table 2.11.1.2-15: Function Summary – get_logger_name()

This function returns the logger's name.

Return Values		
Return Value	Type	Meaning
Logger_Name	pointer to char	logger name

Table 2.11.1.2-16: Function Summary – get_logger_name_from_address(addr)

Given a simulation address, this function returns the name of the corresponding logger (or the string "Unknown").

Parameters		
Parameter	Type	Where Typedef Declared
addr	pointer to SimulationAddress	address.h

Return Values		
Return Value	Type	Meaning
Logger_Names[logger_index]	pointer to char	returns name of logger
"Unknown"	char	returns string "Unknown"

Table 2.11.1.2-17: Function Summary – get_logger_broadcast_address()

Return Values		
Return Value	Type	Meaning
&LoggerBroadcastAddr	pointer to SimulationAddress	pointer to logger broadcast address

2.11.2.2 p_logger.h

This is a protocol file for the logger. It is a commonly-maintained file used by Plan View Display and other SIMNET applications.

2.11.2.3 liblogpvd.a CSU Description (/simnet/pvd/libsrc)

This library supports the logger interface and is used only by the PVD. These are pvd-specific sources for liblogpvd.a: `cmc.c`; `cmc.h`; `filter.c`; `filter.h`; `global.h`; `logger.c`; `logger.h`; `logger.old.c`; `logpvd.c`; `logpvd.h`; `ltime.c`; `ltime.h`; `pro_logger.h`; `pvd.c`; (Section 2.4.1.1); `pvdrtc.c` (Section 3.11.3.1); `pvdrtc.h`; `queue.c`; `queue.h`; `racal.c`; `racal.h`; `rtc.c`; `rth.c`; `rtc.h`; `user.c`; `user.h`; `user.old.c`; `util.c`; `util.h`.

2.11.2.4 libptcomm.a CSU Description (/simnet/pvd/libsrc/libcomm)

The only function from libptcomm.a that is called from this library is `pvd_logger_exit()`, from `pvd2logger.c`.

Table 2.11.2.4-1: libptcomm.a Information.

External Static Variables		
Variable	Type	Where Typedef Declared
<code>logger_skt</code> ;	int	
<code>buffer[BUFFER_LENGTH]</code>	char	

Table 2.11.2.4-2: pvd_logger_exit() Information.

This function does two kinds of closes on a socket that is not open.

Return Values		
Return Value	Type	Meaning
int		

Calls	
Function	Where Described
<code>close_network</code>	

2.11.3 Initial Remote Interfaces CSC Description

This CSC is the main entry for Remote Devices Interfaces.

This CSC functionality is broken down into the CSUs shown in Figure 2.11.3-1.

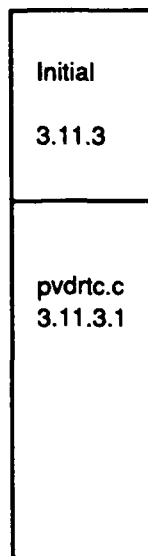


Figure 2.11.3-1: Remote Devices Interfaces--Initial Remote Interfaces CSC Structure.

2.11.3.1 pvdrtc.c

This CSU handles real-time commands to and from the Data Logger.

Table 2.11.3.1-1: Function Summary – handle_log_rtc(rtc)

This function handles processing of runtime messages from the Logger.

Parameters		
Parameter	Type	Where Typedef Declared
rtc	pointer to LoggerPDU	p_logger.h

Table 2.11.3.1-2: Function Summary – send_rtc(rtc, length)

This function places the rtc on the CMC network.

Parameters		
Parameter	Type	Where Typedef Declared
rtc	pointer to LoggerPDU	p_logger.h
length	int	standard

Calls	
Function	Where Described
AssocSendDatagram	libassoc.a in MCC CSCI

Table 2.11.3.1-3: Function Summary – init_pdu(packet, kind, destination)

Parameters		
Parameter	Type	Where Typedef Declared
packet	pointer to LoggerPDU	p_logger.h
kind	int	standard
destination	pointer to SimulationAddress	address.h

Table 2.11.3.1-4: Function Summary – avail_reply_handler(inrtc)

Parameters		
Parameter	Type	Where Typedef Declared
inrtc	pointer to LoggerPDU	p_logger.h

Table 2.11.3.1-5: Function Summary – connect_ack_nak_handler(buf)

Parameters		
Parameter	Type	Where Typedef Declared
buf	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
get_logger_address	Logger_comm.c
get_logger_name	Logger_comm.c
print_simulation_address	pvd_misc.c

Table 2.11.3.1-6: Function Summary – information_handler(inrtc)

Parameters		
Parameter	Type	Where Typedef Declared
inrtc	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
get_logger_name	Logger_comm.c

Table 2.11.3.1-7: Function Summary – clock_tick_handler(inrtc)

Parameters		
Parameter	Type	Where Typedef Declared
inrtc	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
offset datetime	
datetime to strings	

Table 2.11.3.1-8: Function Summary – status_reply_handler(inrtc)

Parameters		
Parameter	Type	Where Typedef Declared
inrtc	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
get logger name from address	Logger comm.c

Table 2.11.3.1-9: Function Summary – do_avail_request(pdu)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
init pdu	Pvdrtc.c
get logger broadcast address	Logger comm.c

Table 2.11.3.1-10: Function Summary – do_connect_request(pdu, destination)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h

Calls	
Function	Where Described
init pdu	Pvdrtc.c

Table 2.11.3.1-11: Function Summary – do_disconnect(pdu, destination)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

Table 2.11.3.1-12: Function Summary – do_start_recording_command (pdu, destination, starttime, is_contig, is_looping, is_promiscuous, racal_present, medium, nvolumes, nexercises, sizes, exercise_ids, file_nms, projid, comment)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h
starttime	pointer to LoggerTime	p_logger.h
is_contig	BOOL	sim_types.h
is_looping	BOOL	sim_types.h
is_promiscuous	BOOL	sim_types.h
racal_present	BOOL	sim_types.h
medium	int	standard
nvolumes	int	standard
nexercises	int	standard
sizes	array of int	standard
exercise_ids	array of int	standard
file_names	array of pointer to char	standard
projid	pointer to char	standard
comment	pointer to char	standard

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

Table 2.11.3.1-13: Function Summary – do_start_playback_command
 (pdu, destination, is_looping, racal_present, medium,
 nvolumes, size, exerciseid, file_nms, filter_out_stealth)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h
is_looping	BOOL	sim_types.h
racal_present	BOOL	sim_types.h
medium	int	standard
nvolumes	int	standard
size	int	standard
exerciseid	int	standard
file_nms	array of pointer to char	standard
filter_out_stealth	int	standard

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

Table 2.11.3.1-14: Function Summary – do_continue_command
 (pdu, destination)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

Table 2.11.3.1-15: Function Summary – do_suspend_command
 (pdu, destination)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

**Table 2.11.3.1-16: Function Summary - do_stop_command
(pdu, destination)**

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

**Table 2.11.3.1-17: Function Summary - do_seek_relative_command
(pdu, dnation, seconds, rel_curloc)**

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h
seconds	int	standard
rel_curloc	BOOL	sim_types.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

**Table 2.11.3.1-18: Function Summary - do_play_speed_command
(pdu, destination, factor)**

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h
destination	pointer to SimulationAddress	address.h
factor	int	standard

Calls	
Function	Where Described
init_pdu	Pvdrtc.c

Table 2.11.3.1-19: Function Summary - do_status_request(pdu)

Parameters		
Parameter	Type	Where Typedef Declared
pdu	pointer to LoggerPDU	p_logger.h

Calls	
Function	Where Described
init_pdu	Pvdrtc.c