

Software version : V1.32
Document version : V1.32
Original instructions(Korean)

API manual(V1.32)



M0609 | M0617 | M1013 | M1509 | H2017 |
H2515 | A0509 | A0509S | A0912 | A0912S

DOOSAN

© 2025 Doosan Robotics Inc.

Table of Contents

1	머리말	15
1.1	저작권	15
1.2	문서 제.개정 이력.....	15
2	소개(Introduction)	23
2.1	설치 가이드	23
2.1.1	헤더파일 사용하기	23
2.1.2	라이브러리 링크하기	23
2.1.3	권장 운용 사양.....	23
2.1.4	라이브러리 구성	24
2.2	프로그래밍 유의사항	25
2.2.1	로봇 연결/해제	25
2.2.2	로봇 초기화	25
2.2.3	제어권 관리	26
2.2.4	로봇 운용 모드.....	26
2.2.5	로봇 운용 상태.....	26
2.2.6	로봇 상태 천이.....	27
2.2.7	프로그램 실행 및 종료.....	29
2.2.8	로봇 안전 설정 기능 제한.....	30
3	정의(Definition)	31
3.1	상수 및 열거형 정의.....	31
3.1.1	enum.ROBOT_STATE.....	31
3.1.2	enum.ROBOT_CONTROL.....	32
3.1.3	enum.MONITORING_SPEED	33
3.1.4	enum.SPEED_MODE	33
3.1.5	enum.ROBOT_SYSTEM	33
3.1.6	enum.ROBOT_MODE	33
3.1.7	enum.ROBOT_SPACE	33
3.1.8	enum.SAFE_STOP_RESET_TYPE	34
3.1.9	enum.MANAGE_ACCESS_CONTROL.....	34

3.1.10	enum.MONITORING_ACCESS_CONTROL.....	34
3.1.11	enum.COORDINATE_SYSTEM	35
3.1.12	enum.JOG_AXIS.....	35
3.1.13	enum.JOINT_AXIS.....	36
3.1.14	enum.TASK_AXIS	36
3.1.15	enum.FORCE_AXIS	37
3.1.16	enum.MOVE_REFERENCE.....	37
3.1.17	enum.MOVE_MODE	37
3.1.18	enum.FORCE_MODE.....	38
3.1.19	enum.BLENDING_SPEED_TYPE.....	38
3.1.20	enum.STOP_TYPE	38
3.1.21	enum.MOVEB_BLENDING_TYPE.....	39
3.1.22	enum.SPLINE_VELOCITY_OPTION	39
3.1.23	enum.GPIO_CTRLBOX_DIGITAL_INDEX	39
3.1.24	enum.GPIO_CTRLBOX_ANALOG_INDEX.....	41
3.1.25	enum.GPIO_ANALOG_TYPE.....	41
3.1.26	enum.GPIO_TOOL_DIGITAL_INDEX.....	42
3.1.27	enum.MODBUS_REGISTER_TYPE.....	42
3.1.28	enum.DRL_PROGRAM_STATE.....	42
3.1.29	enum.PROGRAM_STOP_CAUSE	43
3.1.30	enum.PATH_MODE.....	43
3.1.31	enum.CONTROL_MODE	43
3.1.32	enum.DATA_TYPE	44
3.1.33	enum.VARIABLE_TYPE.....	44
3.1.34	enum.SUB_PROGRAM	44
3.1.35	enum.SINGULARITY_AVOIDANCE.....	45
3.1.36	enum.MESSAGE_LEVEL.....	45
3.1.37	enum.POPUP_RESPONSE	45
3.1.38	enum.MOVE_HOME	45
3.1.39	enum.BYTE_SIZE.....	46
3.1.40	enum.STOP_BITS.....	46
3.1.41	enum.PARITY_CHECK.....	46
3.1.42	enum.RELEASE_MODE.....	47
3.1.43	enum.SAFETY_MODE.....	47
3.1.44	enum.SAFETY_STATE	47

3.1.45	enum.SAFETY_MODE_EVENT.....	48
3.1.46	enum.CO_G_REFERENCE.....	49
3.1.47	enum.ADD_UP	49
3.1.48	enum.OUTPUT_TYPE.....	49
3.2	로그 및 알람 정의.....	50
3.2.1	LOG_LEVEL.....	50
3.2.2	LOG_GROUP	50
3.2.3	LOG_CODE.....	50
3.3	콜백 함수 정의.....	51
3.3.1	TOnMonitoringStateCB	51
3.3.2	TOnMonitoringDataCB	52
3.3.3	TOnMonitoringDataExCB.....	52
3.3.4	TOnMonitoringCtrlIOCB	53
3.3.5	TOnMonitoringCtrlIOExCB.....	54
3.3.6	TOnMonitoringModbusCB.....	55
3.3.7	TOnLogAlarmCB	56
3.3.8	TOnMonitoringAccessControlCB	57
3.3.9	TOnHommingCompletedCB	58
3.3.10	TOnTpInitializingCompletedCB	58
3.3.11	TOnMonitoringSpeedModeCB	59
3.3.12	TOnMasteringNeedCB	60
3.3.13	TOnProgramStoppedCB	60
3.3.14	TOnDisconnectedCB.....	61
3.3.15	TOnTpPopupCB	62
3.3.16	TOnTpLogCB	63
3.3.17	TOnTpGetUserInputCB	63
3.3.18	TOnTpProgressCB.....	64
3.3.19	TOnMonitoringRobotSystemCB	65
3.3.20	TOnMonitoringSafetyStateCB	65
3.4	구조체 정의	66
3.4.1	struct.SYSTEM_VERSION	68
3.4.2	struct.MONITORING_DATA.....	69
3.4.3	struct.MONITORING_DATA_EX	72
3.4.4	struct.MONITORING_CTRLIO.....	77
3.4.5	struct.MONITORING_CTRLIO_EX	79

3.4.6	struct.MONITORING_MODBUS	80
3.4.7	struct.LOG_ALARM.....	81
3.4.8	struct.MOVE_POSB	81
3.4.9	struct.ROBOT_POSE.....	82
3.4.10	struct.USER_COORDINATE.....	82
3.4.11	struct.MESSAGE_POPUP.....	82
3.4.12	struct.MESSAGE_INPUT.....	83
3.4.13	struct.MESSAGE_PROGRESS	83
3.4.14	struct.FLANGE_SERIAL_DATA.....	83
3.4.15	struct.FLANGE_SER_RXD_INFO	84
3.4.16	struct.READ_FLANGE_SERIAL.....	85
3.4.17	struct. INVERSE_KINEMATIC_RESPONSE.....	85
3.4.18	struct.UPDATE_SW_MODULE_RESPONSE.....	85
3.4.19	struct.CONFIG_JOINT_RANGE.....	86
3.4.20	struct.GENERAL_RANGE	86
3.4.21	struct.CONFIG_GENERAL_RANGE	86
3.4.22	struct.POINT_2D	86
3.4.23	struct.POINT_3D	87
3.4.24	struct.LINE	87
3.4.25	union.CONFIG_SAFETY_FUNCTION	87
3.4.26	struct._tStopCode	88
3.4.27	struct.CONFIG_INSTALL_POSE.....	88
3.4.28	struct.CONFIG_SAFETY_IO	88
3.4.29	struct.CONFIG_SAFETY_IO_EX	89
3.4.30	union.VIRTUAL_FENCE_OBJECT	89
3.4.31	struct.CONFIG_VIRTUAL_FENCE	89
3.4.32	struct.CONFIG_SAFE_ZONE.....	90
3.4.33	struct.ENABLE_SAFE_ZONE	90
3.4.34	struct.SAFETY_OBJECT_SPHERE	90
3.4.35	struct.SAFETY_OBJECT_CAPSULE	90
3.4.36	struct.SAFETY_OBJECT_CUBE.....	91
3.4.37	struct.SAFETY_OBJECT_OBB	91
3.4.38	struct.SAFETY_OBJECT_POLYPRISM.....	91
3.4.39	union.SAFETY_OBJECT_DATA	91
3.4.40	struct.SAFETY_OBJECT	92

3.4.41	struct.CONFIG_PROTECTED_ZONE	92
3.4.42	struct.CONFIG_COLLISION_MUTE_ZONE_PROPERTY	93
3.4.43	struct.CONFIG_COLLISION_MUTE_ZONE.....	93
3.4.44	struct.SAFETY_TOOL_ORIENTATION_LIMIT	93
3.4.45	struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE.....	94
3.4.46	struct.CONFIG_NUDGE	94
3.4.47	struct.CONFIG_COCKPIT_EX	94
3.4.48	struct.CONFIG_IDLE_OFF.....	95
3.4.49	struct.WRITE_MODBUS_DATA	95
3.4.50	struct.WRITE_MODBUS_RTU_DATA	95
3.4.51	struct.MODBUS_DATA	96
3.4.52	struct.MODBUS_DATA_LIST	96
3.4.53	struct.CONFIG_WORLD_COORDINATE.....	96
3.4.54	struct.CONFIG_CONFIGURABLE_IO.....	97
3.4.55	struct.CONFIG_CONFIGURABLE_IO_EX	97
3.4.56	struct.CONFIG_TOOL_SHAPE	97
3.4.57	struct.CONFIG_TOOL_SYMBOL.....	97
3.4.58	struct.CONFIG_TCP_SYMBOL	98
3.4.59	struct.CONFIG_TOOL_LIST	98
3.4.60	struct.CONFIG_TOOL_SHAPE_SYMBOL	98
3.4.61	struct.CONFIG_TCP_LIST	98
3.4.62	struct.CONFIG_TOOL_SHAPE_LIST	99
3.4.63	struct.SAFETY_CONFIGURATION_EX.....	99
3.4.64	struct.SAFETY_CONFIGURATION_EX2	101
3.4.65	struct.SAFETY_CONFIGURATION_EX2_V3.....	103
3.4.66	struct.ROBOT_VEL	105
3.4.67	struct.ROBOT_FORCE.....	105
3.4.68	struct.CONFIG_SAFETY_IO_OP	105
3.4.69	struct.MONITORING_CTRLIO_EX2	106
3.4.70	struct.ROBOT_WELDING_DATA	107
3.4.71	struct.WELDING_CHANNEL	108
3.4.72	struct.CONFIG_WELDING_INTERFACE	109
3.4.73	struct.CONFIG_WELD_SETTING	110
3.4.74	struct.GET_WELDING_SETTING_RESPONSE.....	111
3.4.75	struct.ADJUST_WELDING_SETTING	112

3.4.76	struct.CONFIG_TRAPEZOID_WEAVERS_SETTING	113
3.4.77	struct.CONFIG_ZIGZAG_WEAVERS_SETTING	114
3.4.78	struct.CONFIG_CIRCULE_WEAVERS_SETTING	115
3.4.79	struct.CONFIG_SINE_WEAVERS_SETTING	115
3.4.80	struct.CONFIG_WELDING_DETAIL_INFO	116
3.4.81	struct.CONFIG_ANALOG_WELDING_INTERFACE	117
3.4.82	struct.CONFIG_ANALOG_WELDING_SETTING	118
3.4.83	struct.ANALOG_WELDING_ADJUST_SETTING	119
3.4.84	struct.CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	120
3.4.85	struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS	121
3.4.86	struct.CONFIG_DIGITAL_WELDING_INTERFACE_MODE	122
3.4.87	struct.CONFIG_DIGITAL_WELDING_INTERFACE_TEST	122
3.4.88	struct.CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION	123
3.4.89	struct.CONFIG_DIGITAL_WELDING_INTERFACE_OPTION	124
3.4.90	struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2	125
3.4.91	struct.CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING	125
3.4.92	struct.CONFIG_DIGITAL_WELDING_INTERFACE_OTHER	126
3.4.93	struct.DIGITAL_WELDING_RESET	127
3.4.94	struct.CONFIG_DIGITAL_WELDING_MODE	127
3.4.95	struct.CONFIG_DIGITAL_WELDING_CONDITION	127
3.4.96	struct.CONFIG_DIGITAL_WELDING_ADJUST	130
3.4.97	struct.MEASURE_TCP_WELDING	131
3.4.98	struct.TACK_WELDING_SETTING	131
3.4.99	struct.DIGITAL_FORCE_WRITE_DATA	131
3.4.100	struct.ROBOT_DIGITAL_WELDING_DATA	132
3.4.101	struct.DIGITAL_WELDING_COMM_STATE	134

4 함수(Function) 136

4.1	로봇 연결 함수	136
4.1.1	CDRFLEX.open_connection	136
4.1.2	CDRFLEX.close_connection	136
4.2	로봇 속성 함수	137
4.2.1	CDRFLEX.get_system_version	137
4.2.2	CDRFLEX.get_library_version	138
4.2.3	CDRFLEX.get_robot_mode	138

4.2.4	CDRFLEX.set_robot_mode	139
4.2.5	CDRFLEX.get_robot_state.....	140
4.2.6	CDRFLEX.set_robot_control.....	141
4.2.7	CDRFLEX.set_robot_system.....	141
4.2.8	CDRFLEX.get_robot_speed_mode	142
4.2.9	CDRFLEX.set_robot_speed_mode.....	143
4.2.10	CDRFLEX.get_program_state	143
4.2.11	CDRFLEX.get_robot_system.....	144
4.2.12	CDRFLEX.set_safe_stop_reset_type	145
4.2.13	CDRFLEX.get_current_pose	146
4.2.14	CDRFLEX.get_current_posj	146
4.2.15	CDRFLEX.get_desired_posj	147
4.2.16	CDRFLEX.get_current_velj	147
4.2.17	CDRFLEX.get_current_posx	148
4.2.18	CDRFLEX.get_desired_posx	149
4.2.19	CDRFLEX.get_current_tool_flange_posx.....	149
4.2.20	CDRFLEX.get_current_velx	150
4.2.21	CDRFLEX.get_desired_velx	150
4.2.22	CDRFLEX.get_joint_torque	151
4.2.23	CDRFLEX.get_control_space	152
4.2.24	CDRFLEX.get_external_torque	152
4.2.25	CDRFLEX.get_tool_force	153
4.2.26	CDRFLEX.get_current_solution_space	153
4.2.27	CDRFLEX.get_last_alarm	154
4.2.28	CDRFLEX.get_solution_space	155
4.2.29	CDRFLEX.get_orientation_error	155
4.2.30	CDRFLEX.get_control_mode	156
4.2.31	CDRFLEX.get_current_rotm	156
4.2.32	CDRFLEX.get_safety_configuration	157
4.3	콜백함수 등록 함수	158
4.3.1	CDRFLEX.set_on_monitoring_state	158
4.3.2	CDRFLEX.set_on_monitoring_data	159
4.3.3	CDRFLEX.set_on_monitoring_data_ex.....	160
4.3.4	CDRFLEX.set_on_monitoring_ctrl_io	161
4.3.5	CDRFLEX.set_on_monitoring_ctrl_io_ex	161

4.3.6	CDRFLEX.set_on_monitoring_modbus.....	163
4.3.7	CDRFLEX.set_on_log_alarm	163
4.3.8	CDRFLEX.set_on_tp_popup	164
4.3.9	CDRFLEX.set_on_tp_log	165
4.3.10	CDRFLEX.set_on_tp_progress	166
4.3.11	CDRFLEX.set_on_tp_get_user_input.....	167
4.3.12	CDRFLEX.set_on_monitoring_access_control.....	167
4.3.13	CDRFLEX.set_on_homing_completed	168
4.3.14	CDRFLEX.set_on_tp_initializing_completed.....	169
4.3.15	CDRFLEX.set_on_monitoring_speed_mode.....	170
4.3.16	CDRFLEX.set_on_mastering_need	171
4.3.17	CDRFLEX.set_on_program_stopped	171
4.3.18	CDRFLEX.set_on_disconnected	172
4.3.19	CDRFLEX.set_on_monitoring_robot_system	173
4.3.20	CDRFLEX.set_on_monitoring_safety_state	174
4.4	제어권 관리 함수.....	174
4.4.1	CDRFLEX.manage_access_control	174
4.5	기본 제어 함수.....	175
4.5.1	CDRFLEX.jog	175
4.5.2	CDRFLEX.move_home	176
4.6	모션 제어 함수.....	177
4.6.1	CDRFLEX.movej	177
4.6.2	CDRFLEX.movesj.....	180
4.6.3	CDRFLEX.movesx	182
4.6.4	CDRFLEX.move_spiral	184
4.6.5	CDRFLEX.move_periodic	187
4.6.6	CDRFLEX.amovej.....	189
4.6.7	CDRFLEX.amovel	191
4.6.8	CDRFLEX.amevec	192
4.6.9	CDRFLEX.amovesj	194
4.6.10	CDRFLEX.amovesx	196
4.6.11	CDRFLEX.amoveb.....	198
4.6.12	CDRFLEX.ameve_spiral	200
4.6.13	CDRFLEX.ameve_periodic.....	203
4.6.14	CDRFLEX.stop	206

4.6.15	CDRFLEX.move_pause.....	206
4.6.16	CDRFLEX.move_resume.....	207
4.6.17	CDRFLEX.mwait	208
4.6.18	CDRFLEX.trans.....	209
4.6.19	CDRFLEX.fkin.....	210
4.6.20	CDRFLEX.ikin	210
4.6.21	CDRFLEX.set_ref_coord	211
4.6.22	CDRFLEX.check_motion	212
4.6.23	CDRFLEX.enable_alter_motion	213
4.6.24	CDRFLEX.alter_motion	214
4.6.25	CDRFLEX.disable_alter_motion.....	215
4.6.26	CDRFLEX.servoj	216
4.6.27	CDRFLEX.servol	218
4.6.28	CDRFLEX.speedj.....	219
4.6.29	CDRFLEX.speedl.....	220
4.6.30	CDRFLEX.moveb	221
4.6.31	CDRFLEX.ikin(Extension)	224
4.6.32	CDRFLEX.movejx.....	225
4.6.33	CDRFLEX.amove_spiral(Extension).....	227
4.6.34	CDRFLEX.movel	230
4.6.35	CDRFLEX.movec.....	233
4.7	로봇 설정 함수.....	236
4.7.1	CDRFLEX.add_tool	236
4.7.2	CDRFLEX.del_tool	237
4.7.3	CDRFLEX.set_tool	238
4.7.4	CDRFLEX.get_tool	239
4.7.5	CDRFLEX.add_tcp	240
4.7.6	CDRFLEX.del_tcp	241
4.7.7	CDRFLEX.set_tcp	241
4.7.8	CDRFLEX.get_tcp	242
4.7.9	CDRFLEX.set_tool_shape	243
4.7.10	CDRFLEX.get_workpiece_weight.....	243
4.7.11	CDRFLEX.reset_workpiece_weight.....	244
4.7.12	CDRFLEX.set_singularity_handling.....	245
4.7.13	CDRFLEX.setup_monitoring_version	246

4.7.14	CDRFLEX.config_program_watch_variable	246
4.7.15	CDRFLEX.set_user_home.....	247
4.7.16	CDRFLEX.servo_off	248
4.7.17	CDRFLEX.release_protective_stop.....	249
4.7.18	CDRFLEX.change_collision_sensitivity.....	249
4.7.19	CDRFLEX.set_safety_mode	250
4.7.20	CDRFLEX.set_auto_servo_off.....	251
4.7.21	CDRFLEX.set_workpiece_weight	251
4.8	I/O 제어 함수	253
4.8.1	CDRFLEX.set_tool_digital_output	253
4.8.2	CDRFLEX.get_tool_digital_input	254
4.8.3	CDRFLEX.get_tool_digital_output	255
4.8.4	CDRFLEX.set_digital_output.....	255
4.8.5	CDRFLEX.get_digital_input.....	256
4.8.6	CDRFLEX.get_digital_output.....	257
4.8.7	CDRFLEX.set_mode_analog_input.....	258
4.8.8	CDRFLEX.set_mode_analog_output	259
4.8.9	CDRFLEX.set_analog_output	259
4.8.10	CDRFLEX.get_analog_input	260
4.8.11	CDRFLEX.add_modbus_signal.....	261
4.8.12	CDRFLEX.del_modbus_signal.....	262
4.8.13	CDRFLEX.set_modbus_output.....	263
4.8.14	CDRFLEX.get_modbus_input	264
4.8.15	CDRFLEX.flange_serial_open	264
4.8.16	CDRFLEX.flange_serial_close	265
4.8.17	CDRFLEX.flange_serial_write	266
4.8.18	CDRFLEX.get_tool_analog_input.....	267
4.8.19	CDRFLEX.set_tool_digital_output_level.....	268
4.8.20	CDRFLEX.set_tool_digital_output_type	269
4.8.21	CDRFLEX.set_mode_tool_analog_input	270
4.9	프로그램 제어 함수	271
4.9.1	CDRFLEX.drl_start	271
4.9.2	CDRFLEX.drl_stop	272
4.9.3	CDRFLEX.drl_pause	273
4.9.4	CDRFLEX.drl_resume.....	273

4.9.5	CDRFLEX.change_operation_speed	274
4.9.6	CDRFLEX.save_sub_program	275
4.9.7	CDRFLEX.tp_popup_response	275
4.9.8	CDRFLEX.tp_get_user_input_response	276
4.10	힘/강성 제어 및 기타 사용자 편의 함수	277
4.10.1	CDRFLEX.parallel_axis.....	277
4.10.2	CDRFLEX.align_axis.....	279
4.10.3	CDRFLEX.is_done_bolt_tightening	280
4.10.4	CDRFLEX.task_compliance_ctrl.....	281
4.10.5	CDRFLEX.release_compliance_ctrl	282
4.10.6	CDRFLEX.set_stiffnessx	283
4.10.7	CDRFLEX.calc_coord	284
4.10.8	CDRFLEX.set_user_cart_coord	286
4.10.9	CDRFLEX.overwrite_user_cart_coord	288
4.10.10	CDRFLEX.get_user_cart_coord.....	289
4.10.11	CDRFLEX.set_desired_force	290
4.10.12	CDRFLEX.release_force	291
4.10.13	CDRFLEX.check_position_condition_abs	292
4.10.14	CDRFLEX.check_position_condition_rel.....	293
4.10.15	CDRFLEX.check_position_condition	294
4.10.16	CDRFLEX.check_force_condition	296
4.10.17	CDRFLEX.check_orientation_condition.....	297
4.10.18	CDRFLEX.coord_transform	300
4.10.19	CDRFLEX.set_palletizing_mode.....	301
4.10.20	CDRFLEX.query_modbus_data_list.....	302

5 실시간 외부 제어(Realtime Control) 303

5.1	실시간 외부 제어 소개.....	303
5.1.1	버전	303
5.2	통신 연결 함수.....	303
5.2.1	CDRFLEX.connect_rt_control	303
5.2.2	CDRFLEX.disconnect_rt_control.....	304
5.3	정보 조회 함수.....	305
5.3.1	CDRFLEX.get_rt_control_input_version_list	305
5.3.2	CDRFLEX.get_rt_control_output_version_list	305

5.3.3	CDRFLEX.get_rt_control_input_data_list.....	306
5.3.4	CDRFLEX.get_rt_control_output_data_list.....	307
5.4	설정 함수	311
5.4.1	CDRFLEX.set_rt_control_input.....	311
5.4.2	CDRFLEX.set_rt_control_output	312
5.5	운용 함수	313
5.5.1	CDRFLEX.start_rt_control.....	313
5.5.2	CDRFLEX.stop_rt_control	314
5.5.3	CDRFLEX.set_on_rt_monitoring_data	315
5.5.4	CDRFLEX.read_data_rt	316
5.5.5	CDRFLEX.write_data_rt	317
5.6	서브 모션 함수.....	318
5.6.1	알아두기	318
5.6.2	CDRFLEX.set_velj_rt.....	318
5.6.3	CDRFLEX.set_accj_rt	319
5.6.4	CDRFLEX.set_velx_rt.....	320
5.6.5	CDRFLEX.set_accx_rt	321
5.6.6	CDRFLEX.servoj_rt	321
5.6.7	CDRFLEX.servol_rt	324
5.6.8	CDRFLEX.speedj_rt	325
5.6.9	CDRFLEX.speedl_rt	328
5.6.10	CDRFLEX.torque_rt	329
6	어플리케이션 명령어(Application Command)	332
6.1	용접(Welding)	332
6.1.1	set_on_monitoring_welding_data.....	333
6.1.2	set_on_monitoring_analog_welding_data	334
6.1.3	set_on_monitoring_digital_welding_data	335
6.1.4	app_weld_weave_cond_trapezoidal	338
6.1.5	app_weld_weave_cond_zigzag	339
6.1.6	app_weld_weave_cond_circular	341
6.1.7	app_weld_weave_cond_sinusoidal	342
6.1.8	app_weld_enable_analog.....	343
6.1.9	app_weld_set_weld_cond_analog.....	347
6.1.10	app_weld_adj_welding_cond_analog	348

6.1.11	app_weld_set_interface_eip_m2r_process	350
6.1.12	app_weld_set_interface_eip_r2m_mode	353
6.1.13	app_weld_set_interface_eip_r2m_process	355
6.1.14	app_weld_set_interface_eip_r2m_test.....	357
6.1.15	app_weld_set_interface_eip_r2m_condition	359
6.1.16	app_weld_set_interface_eip_r2m_option.....	361
6.1.17	app_weld_set_interface_eip_m2r_process2	363
6.1.18	app_weld_set_interface_eip_m2r_monitoring.....	366
6.1.19	app_weld_set_interface_eip_m2r_other	368
6.1.20	app_weld_reset_interface	370
6.1.21	app_weld_enable_digital	371
6.1.22	app_weld_set_weld_cond_digital	371
6.1.23	app_weld_adj_welding_cond_digital	373
6.1.24	measure_welding_tcp	374
6.1.25	set_welding_cockpit_setting	375
6.1.26	set_digital_welding_monitoring_mode	376
6.1.27	app_weld_adj_motion_offset	377
6.1.28	set_welding_cockpit_setting_time_setting	378

1 머리말

두산로보틱스의 API는 프로그래밍 언어 C/C++ 개발되었으며, 두산 로봇 컨트롤러를 터치 패널이 아닌 별도의 사용자 어플리케이션에서 직접 제어하기 위해 사용할 수 있습니다. 본 API 매뉴얼은 위한 API 함수 및 기능에 대해 설명하고 있습니다.

본 매뉴얼의 내용은 작성된 시점을 기준으로 하며, 제품에 대한 정보는 사용자에게 사전 고지 없이 변경될 수 있습니다.

개정된 매뉴얼에 대한 상세 정보는 Robot LAB (<https://robotlab.doosanrobotics.com/>)에서 확인하십시오.

1.1 저작권

본 매뉴얼의 모든 내용과 도안에 대한 저작권 및 지적재산권은 두산로보틱스에 있습니다. 따라서 두산로보틱스의 서면 허가 없이 사용, 복사, 유포하는 어떠한 행위도 금지됩니다. 또한 특허권을 오용하거나 변용하는데 따르는 책임은 전적으로 사용자에게 있습니다.

본 매뉴얼은 신뢰할 수 있는 정보지만 오류 또는 오탈자로 인한 손실에 대해 어떠한 책임도 지지 않습니다. 제품 개선에 따라 매뉴얼에 포함된 정보는 예고 없이 변경될 수 있습니다.

© Doosan Robotics Inc., All rights reserved

1.2 문서 제.개정 이력

개정번호	제/개정 페이지 및 내용	개정 일자
1.0	최초 작성 및 배포	2018-06-29
1.1	신규 기능 추가	2020-05-13
1.11	함수 이름 변경(DRL Style)	2020-05-28
1.12	헤더 분할(CDRFLEx 추가)	2020-06-08
1.13	Flange_serial 기능 등 추가	2020-10-19
1.14	오타수정 / 일부 함수 수정 (movesj/set_safe_stop_reset_type/ close_connection 등)	2021-02-18

개정번호	제/개정 페이지 및 내용	개정 일자
1.15	누락 함수 추가 set_user_home flange_serial_read 파라미터 추가(timeout)	2021-03-19
1.16	누락 함수 추가 servo_off 함수 flange_serial_read 기능 추가	2021-03-24
1.17	버전정보 수정 : GL010110	2021-06-09
1.18	버전 정보 수정 : GL010111 신규 기능 추가 : get_override_speed	2021-09-15
1.19	버전 정보 수정 : GL010112 신규 기능 추가 .ikin(extension) 추가 .set_monitoring_robot_system 추가 .change_collision_detection 추가 .add_sw_module 추가 .del_sw_module 추가 .update_sw_module 추가 .release_protective_stop 추가 .set_on_monitoring_update_module 추가	2021-11-22
1.2	버전 정보 수정 : GL010113 신규 기능 추가 .실시간 제어 위한 함수 추가(별도 매뉴얼 참고) .set_safety_mode 추가 .set_on_monitoring_state 추가	2021-12-16
1.21	신규 기능 추가 .set_auto_servo_off	2021-12-19

개정번호	제/개정 페이지 및 내용	개정 일자
1.22	<p>버전 정보 수정 : GL010114</p> <p>get_robot_state 종복 변경(오기) -> get_program_state로 변경</p> <p>set_safe_stop_reset_type 예제에 enum 파라미터 수정 so 파일 관련 설명 수정(64bits POCO 라이브러리)</p> <p>신규 기능 추가</p> <ul style="list-style-type: none"> .servoj / servol / speedj / speedl .get_safety_configuration <p>안전 설정 반환값을 위한 신규 구조체 추가 : SAFETY_CONFIGURATION_EX</p>	2022-02-11
1.23	<p>3.3.21 : 인수 부분 TOnMonitoringSafetyState -> TOnMonitoringSafetyStateCB로 변경 strcut.MESSAGE_PROGRESS 추가</p> <p>3.7.14 : 인수 부분 TYPE_TYPE -> DATA_TYPE으로 변경 열거형 상수 SAFETY_MODE / SAFETY_EVENT 추가</p> <p>3.8.11 인수 부분에 enum -> enum.MODBUS_REGISTER_TYPE으로 수정 열거형 상수 COORDINATE_SYSTEM 추가</p> <p>열거형 상수 ROBOT_STATE 표기 오류 해결 get_override_speed 삭제</p> <p>실시간 제어 관련 내용 추가(chapter 4)</p>	2022-03-10

개정번호	제/개정 페이지 및 내용	개정 일자
1.24	<p>enum.SAFETY_MODE_EVENT 관련 내용 수정 set_safety_mode 예제의 SAFETY_MODE_EVENT 파라미터 관련 내용 수정</p> <p>flange_serial 관련 명령어에 port 파라미터 추가 (신규 플랜지에서 사용 가능함을 명시)</p> <p>버전 정보 변경 -> GL010115</p> <p>신규 함수 추가</p> <ul style="list-style-type: none"> - set_workpiece_weight - set_mode_tool_analog_input - set_tool_digital_output_type - set_tool_digital_output_level - get_tool_analog_input <p>enum.COG_REFERNCE 추가</p> <p>enum.ADD_UP 추가</p> <p>enum.OUTPUT_TYPE 추가</p>	2022-04-29
1.25	<p>flange_serial_open/close 함수 port 삭제</p> <p>set_tool / set_workpiece_weight 함수 경고 문구 추가</p>	2022-05-04

개정번호	제/개정 페이지 및 내용	개정 일자
1.26	enum.CONTROL_SPACE 오타 수정 set_on_monitoring_safety_state 오타 수정 MOVE_REFERENCE enum에 월드 / 사용자 좌표계 추가 change_operation_speed 값 범위 변경 TOnRobotSystemCB 수정 property 함수 10개 추가 get_current_posj get_control_space get_current_velj get_desired_posj get_current_tool_flange_posx get_current_velx get_desired_velx get_joint_torque get_external_torque get_tool_force enum추가 : ROBOT_SPACE struct 추가 : ROBOT_VEL /ROBOT_FORCE	2022-06-02
1.27	drl_stop 함수 파라미터 수정 : enum.STOP_TYPE -> unsigned char	2022-07-04
1.28	enum.ROBOT_STATE 설명 수정 enum.ROBOT_MODE 내용 추가 enum.ROBOT_SPACE 중복 부분 삭제 set_palletizing_mode 명령어 추가 move_home 주의사항 추가	2023-02-20
1.29	query_modbus_data_list 명령어 추가	2023-03-24

개정번호	제/개정 페이지 및 내용	개정 일자
1.30	<p>Arm64용 Drfl 바이너리 추가 Ubuntu 22.04 버전 지원 Realtime 성능 개선 Blending 모션 관련 이슈 개선 enum 추가 : MOVE_ORIENTATION, SPIRAL_DIR, ROT_DIR, DR_SERVOJ_TYPE struct 추가 : SAFETY_CONFIGURATION_EX2, CONFIG_SAFETY_IO_OP movej 각 축별 속도 및 가속도 조절 인자 추가 flange_serial_read : 리턴 값 변경 movejx : 확장 movejx 인자 추가 amovej : 확장 amovej 인자 추가 amovejx : 확장 amovejx 인자 추가 servoj : 확장 인자 추가 신규 함수 추가 <ul style="list-style-type: none"> • get_safety_configuration(Extension) • ikin(Add_iter_threshold) • ikin_norm • movec(Extension) • amovec(Extension) • move_spiral(Extension) • amove_spiral(Extension) • servoj (Extension) </p>	2024-11-26
1.31	<p>제어기(DRCF) 버전 선언 옵션 추가 (#define DRCF_VERSION) 신규 제어기 용 인터페이스 <ul style="list-style-type: none"> • MONITORING_CTRLIO_EX2 추가 • READ_CTRLIO_INPUT_EX2 추가 • READ_CTRLIO_OUTPUT_EX2 추가 • GPIO_CTRLBOX_DIGITAL_INDEX 확장 • SAFETY_CONFIGURATION_EX2_V3 추가 신규 제어기 용 API 추가 <ul style="list-style-type: none"> • get_safety_configuration_ex 삭제 • get_safety_configuration 신규 기능으로 업데이트 중복 함수 제거 <ul style="list-style-type: none"> • SetOnMonitoringData 함수 제거 (set_on_monitoring_data 와 중복) • SetOnMonitoringCtrlIO 함수 제거 (set_on_monitoring_ctrl_io 와 중복) </p>	2024-12-03

개정번호	제/개정 페이지 및 내용	개정 일자
1.32	<p>업데이트 기능</p> <ul style="list-style-type: none"> • 용접 관련 API 추가 <p>업데이트 정의</p> <ul style="list-style-type: none"> • struct.ROBOT_WELDING_DATA • struct.WELDING_CHANNEL • struct.CONFIG_WELDING_INTERFACE • struct.CONFIG_WELD_SETTING • struct.GET_WELDING_SETTING_RESPONSE • struct.ADJUST_WELDING_SETTING • struct.CONFIG_TRAPEZOID_WEAVERS_SETTING • struct.CONFIG_ZIGZAG_WEAVERS_SETTING • struct.CONFIG_CIRCULE_WEAVERS_SETTING • struct.CONFIG_SINE_WEAVERS_SETTING • struct.CONFIG_WELDING_DETAIL_INFO • struct.CONFIG_ANALOG_WELDING_INTERFACE • struct.CONFIG_ANALOG_WELDING_SETTING • struct.ANALOG_WELDING_ADJUST_SETTING • struct.CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA • struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS • struct.CONFIG_DIGITAL_WELDING_INTERFACE_MODE • struct.CONFIG_DIGITAL_WELDING_INTERFACE_TEST • struct.CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION • struct.CONFIG_DIGITAL_WELDING_INTERFACE_OPTION • struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2 • struct.CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING • struct.CONFIG_DIGITAL_WELDING_INTERFACE_OTHER • struct.DIGITAL_WELDING_RESET • struct.CONFIG_DIGITAL_WELDING_MODE • struct.CONFIG_DIGITAL_WELDING_CONDITION • struct.CONFIG_DIGITAL_WELDING_ADJUST • struct.MEASURE_TCP_WELDING • struct.TACK_WELDING_SETTING • struct.DIGITAL_FORCE_WRITE_DATA • struct.ROBOT_DIGITAL_WELDING_DATA • struct.DIGITAL_WELDING_COMM_STATE <p>업데이트 함수</p> <ul style="list-style-type: none"> • set_on_monitoring_welding_data • set_on_monitoring_analog_welding_data • set_on_monitoring_digital_welding_data 	2025-01-15

개정번호	제/개정 페이지 및 내용	개정 일자
	<ul style="list-style-type: none">• app_weld_weave_cond_trapezoidal• app_weld_weave_cond_zigzag• app_weld_weave_cond_circular• app_weld_weave_cond_sinusoidal• app_weld_enable_analog• app_weld_set_weld_cond_analog• app_weld_adj_welding_cond_analog• app_weld_set_interface_eip_m2r_process• app_weld_set_interface_eip_r2m_mode• app_weld_set_interface_eip_r2m_process• app_weld_set_interface_eip_r2m_test• app_weld_set_interface_eip_r2m_condition• app_weld_set_interface_eip_r2m_option• app_weld_set_interface_eip_m2r_process2• app_weld_set_interface_eip_m2r_monitoring	

2 소개(Introduction)

본 API는 두산 로봇 제어기를 T/P 어플리케이션(로봇 제어기에 탑재된 사용자 GUI 프로그램)이 아닌 별도의 사용자 어플리케이션에서 직접 제어하기 위한 함수(기능)으로 구성되어 있으며, C/C++ 프로그래밍 언어로 개발되어 있다.

2.1 설치 가이드

2.1.1 헤더파일 사용하기

라이브러리 함수 관련 헤더 파일(DRFL.h)을 #include 시 컴파일 옵션이 C++ 언어로 설정된 경우에는 CDRFLEx 클래스를 사용하여 프로그래밍 가능하나, C언어인 경우에는 “_함수명” 형식의 전역 함수를 사용하여 프로그래밍 해야 한다.

몇몇 함수는 제어기 버전에 따라 함수의 인터페이스 사용이 다를 수 있습니다. 헤더를 임포트하기 전 제어기 버전에 맞게 #define DRCF_VERSION 2 또는 #define DRCF_VERSION 3 을 선언해주세요.

2.1.2 라이브러리 링크하기

리눅스 라이브러리

/etc/ld.so.conf.d 디렉토리 아래에 .conf 확장자(예, DRFLib.conf)를 가진 파일 생성하고, 파일 안에 *.so 파일의 경로를 추가한 후 ldconfig 명령어를 실행하여 해당 라이브러리를 설정한다.

윈도우 라이브러리

윈도우 라이브러리는 Microsoft Visual C++ 2015(x86) 재배포 가능 패키지가 설치되어 있지 않은 경우, 정상적으로 동작하지 않을 수 있습니다.

C++ 프로젝트에서 본 API를 사용하기 위한 프로젝트 설정은 다음과 같다.

1. 헤더파일(include) 경로 설정
[구성 속성]->[C/C++]->[일반]->[추가 포함 디렉터리]
2. 라이브러리(lib) 경로 설정
[구성 속성]->[링커]->[일반]->[추가 라이브러리 디렉터리]

2.1.3 권장 운용 사양

본 라이브러리가 지원하는 권장 운용 사항은 다음과 같다.

구분	명칭	권장 사양	비고

윈도우	Operating System	Windows 7 / Windows 10	
	CPU Architecture	X86(64-bit & 32-bit)	
	Compiler	Mircrosoft Visual C++ 2015	
리눅스	Distros	Ubuntu	
	Kernel and Standard Libraries	Linux 3.x kernel Any GLIBC since 2.0	
	CPU Architecture	X86/Arm(64-bit)	
	Compiler	GNU GCC 4.x or higher	

2.1.4 라이브러리 구성

본 API는 3개의 C/C++ 헤더 파일과, 이와 관련된 라이브러리 파일 및 기타 지원(3rdparty) 라이브러리 파일로 구성되어 있습니다.

종류	파일명	설명	비고
헤더파일	DRFL.h	라이브러리 함수 정의 파일	
	DRFS.h	라이브러리 관련 구조체 정의 파일	
	DRFC.h	라이브러리 관련 상수 정의 파일	
	DRFLEx.h	라이브러리 함수 정의 파일	DRL Style
라이브러리 파일	libDRFL.a	리눅스 계열 라이브러리 파일	
	DRFLWin32.lib	윈도우 계열 라이브러리 파일	
	DRFLWin32.dll		
	DRFLWin64.lib		
	DRFLWin64.dll		

기타 지원파일	libPocoFoundation.so ¹	리눅스 계열 종속성 라이브러리 파일	ldConfig 함수를 이용하여 라이브러리 등록
	libPocoFoundation.so ² .50		
	libPocoFoundation.so ³ .62		
	libPocoNet.so ⁴		
	libPocoNet.so ⁵ .50		
	libPocoNet.so ⁶ .62		
	PocoFoundation.lib	윈도우 계열 종속성 라이브러리 파일	Microsoft Visual C++ 2015(x86) 재배포 가능 패키지 필요
	PocoFoundation.dll		
	PocoNet.lib		
	PocoNet.dll		

2.2 프로그래밍 유의사항

2.2.1 로봇 연결/해제

로봇 제어기와 본 API는 TCP/IP 통신을 통해서 연결됨으로 반드시 연결 수립 과정이 필요하며, 내부 연결 과정 중에 인증 절차가 포함되어 있음으로, 다른 기타 API나 소켓 관련 함수로 연결 시도시 정상적인 연결이 이루어지지 않음으로 반드시 본 API의 연결 관련 함수를 사용해야 한다.

또한 하나의 네트워크에 2개 이상의 로봇 제어기를 사용할 경우에는 T/P 어플리케이션에서 각각의 로봇제어기의 IP 주소를 겹치지 않게 변경하고, 각각의 로봇제어기에 연결 과정을 수행하면 정상적으로 제어할 수 있다.

그리고 본 API는 TCP/IP 통신을 사용함으로 사용자 어플리케이션이 수행되는 컴퓨터의 성능이나, 네트워크의 부하에 따라 성능이 저하나 기능 오류가 발생할 수 있다.

2.2.2 로봇 초기화

로봇 제어기는 로봇 구동에 필요한 각종 정보를 T/P 어플리케이션의 초기화 과정을 통해 전달받아 처리함으로, 본 API 사용하여 구성된 사용자 어플리케이션은 로봇 운용 상태 정보에서 초기화 완료 여부를 반드시 확인 후 로봇 제어를 시작해야 한다. 초기화 완료 여부는 로봇 연결 관련 함수를 사용하여 정상 접속시 콜백 함수인 TOnMonitoringStateCB 함수나 사용자 호출 함수인 get_robot_state 함수를 통해서 확인할 수 있다.

¹ <http://libPocoFoundation.so>

² <http://libPocoFoundation.so>

³ <http://libPocoFoundation.so>

⁴ <http://libPocoNet.so>

⁵ <http://libPocoNet.so>

⁶ <http://libPocoNet.so>

2.2.3 제어권 관리

로봇 제어기는 안전상 하나의 어플리케이션에만 제어할 수 있도록 구성되어 있음으로, 로봇 초기화 완료 후 제어권 관련 함수인 ManageAccessControl(=manage_access_control) 함수와 TOnChangingAccessControlCB 콜백 함수를 사용하여 제어권 획득 및 이양에 관련된 로직을 사용자 어플리케이션에 구현해야 하며, 제어권 소유시만에 제어 명령을 전달해야 한다. 제어권 없이 제어 명령을 전달할 경우, 모든 제어 명령은 무시되어 처리되지 않는다.

2.2.4 로봇 운용 모드

로봇 제어기 운용모드는 자동모드와 수동모드 2개가 지원되며, set/get_robot_mode 함수를 통해서 설정 및 모드를 확인할 수 있다. 자동모드는 당사가 제공하는 로봇 프로그래밍 언어인 DRL로 구성된 프로그램을 자동으로 실행하기 위한 모드이며, 수동모드는 단일 동작(예, 조그 동작)을 수행하기 위한 모드로 안전을 위하여 로봇의 끝단의 TCP 속도가 250mm/sec 제한되는 모드이다. 이와 관련하여 조인트 공간에 대한 로봇 움직임 제어 함수인 movej 명령어를 수동모드로 설정하고 최대 속도로 제어하고자 하는 경우에는, 속도 초과 오류가 발생하고, 로봇의 동작이 정지될 수 있음으로 운용 모드 설정에 유의해야 한다.

2.2.5 로봇 운용 상태

로봇 제어기의 운용 상태 정보는 다음과 같이 총 15가지 상태가 있으며, 예약 사용(11번~14번)을 제외한 모든 상태는 OnMonitoringState 콜백함수나 사용자 호출 함수인 get_robot_state 함수를 통해서 확인할 수 있다.

순번	로봇 운용 상태	설명
0	STATE_INITIALIZING	T/P 어플리케이션에 의해서 자동으로 진입하는 상태로 각종 파라미터 설정을 위한 초기화 상태임. 초기화 완료 시 자동으로 지령 대기 상태로 전환됨.
1	STATE_STANDBY	운용 가능한 기본 상태로, 지령 대기 상태임
2	STATE_MOVING	지령 대기 상태에서 지령 수신 후 로봇 동작 시 자동으로 전환되는 지령 동작 상태임. 동작 완료 시 자동 지령 대기 상태로 전환됨.
3	STATE_SAFE_OF_F	일반적인 Servo Off 흑인 기능 및 동작 오류로 인한 로봇 정지 모드로, 서보 오프 상태(제어 정지 후 모터 및 브레이크 전원을 차단한 상태)임
4	STATE_TEACHING	직접교시 상태
5	STATE_SAFE_STOP	기능 및 동작 오류로 인한 로봇 정지 모드로, 안전 정지 상태(제어 정지만 수행한 상태, 자동 모드인 경우 프로그램 일시 정지 상태) - 로봇 상태에 따라 STATE_STANBY or STATE_MOVING으로 표시됨

순번	로봇 운용 상태	설명
6	STATE_EMERGE NCY_STOP:	비상 정지 상태
7	STATE_HOMMING	홈링 모드 상태. (로봇을 하드 웨어적으로 정렬하는 상태)
8	STATE_RECOVERY	로봇 구동 범위 초과 등과 같은 오류로 인한 로봇 정지 시, 구동 범위 이내로 로봇을 이동시키기 위한 복구 모드 상태
9	STATE_SAFE_STOP2	STATE_SAFE_STOP 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태
10	STATE_SAFE_OF_F2	STATE_SAFE_OFF 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태
11	STATE_RESERVE_D1	예약 사용
12	STATE_RESERVE_D2	예약 사용
13	STATE_RESERVE_D3	예약 사용
14	STATE_RESERVE_D4	예약 사용
15	STATE_NOT_READY	로봇 제어기 부트업 이후 초기화를 위한 대기 상태임. T/P 어플리케이션에 의해 초기화 상태로 전환됨.

2.2.6 로봇 상태 천이

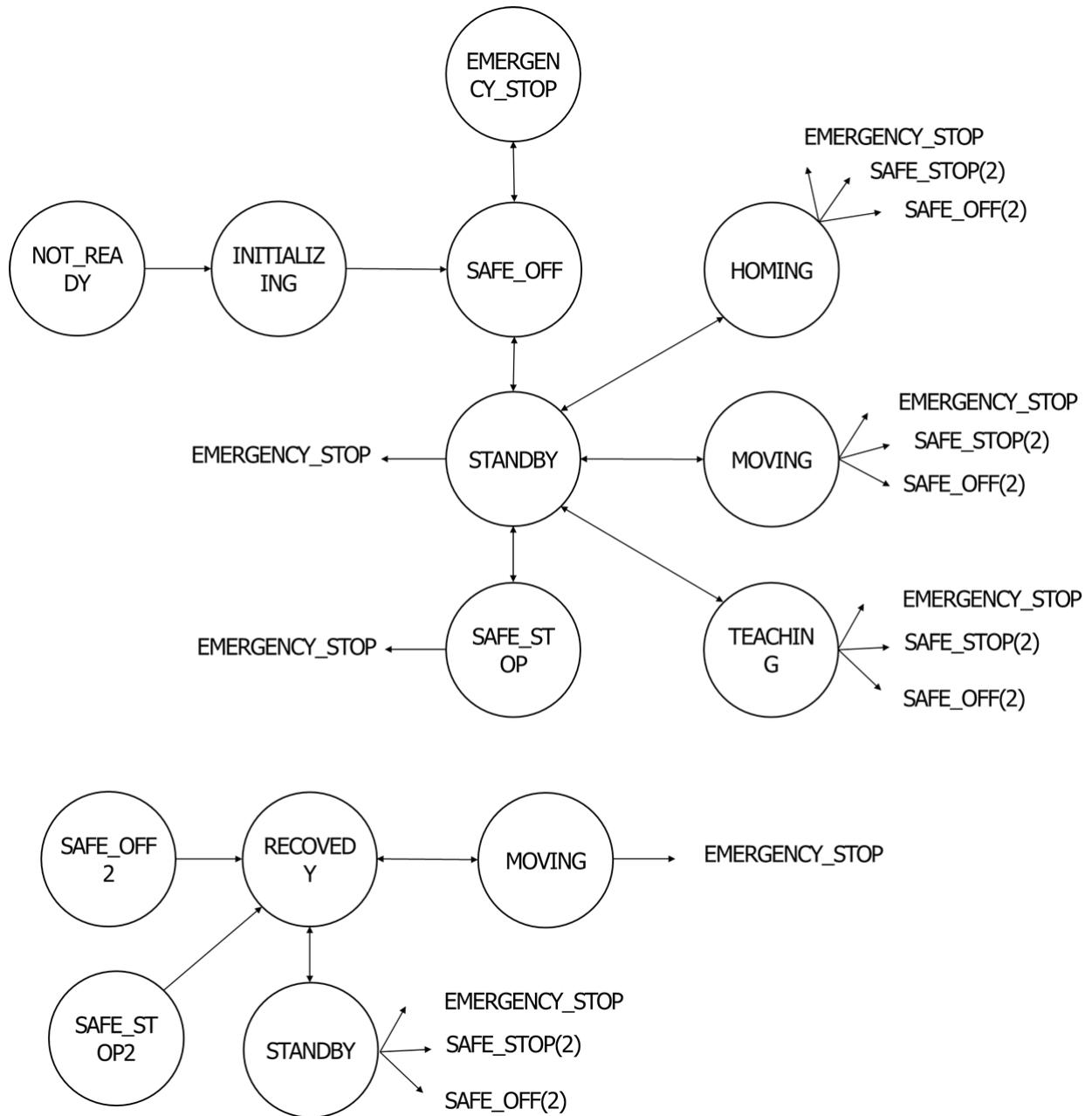
지령 대기 상태(STATE_STANDBY)가 로봇 제어를 위한 기본 준비 상태이며, 사용자로부터 제어 명령 수신시 자동으로 STATE_HOMING 상태나, STATE_MOVING 상태 혹은 STATE_TEACHING 상태로 전환하면서 동작을 수행하고, 오류없이 동작이 완료되면 다시 STATE_STANDBY 상태로 전환하여 사용자 명령을 대기하게 된다.

EMERGENCY_STOP 상태는 초기화(STATE_INITIALIZING) 상태를 제외한 어느 상태에서도 E/M 버튼에 의해서 전환되어 로봇이 정지하게 되며, 로봇 제어기 내부적으로 기능이나 동작 오류 발생시에도 SAFE_OFF 상태(모터 및 브레이크 전원 차단)나, SAFE_STOP 상태(제어 정지)로 자동 전환되어 로봇이 정지하게 된다.

이외에 안전상 사용자에 의해서 직접 상태가 전환되어야 하는 기능은 다음과 같으며, SetRobotControl 함수에 의해서 이 기능을 수행할 수 있다.

순번	로봇 상태 제어 명령	설명
0	CONTROL_INIT_CONFIG	STATE_NOT_READY 상태에서 STATE_INITIALIZING 상태로 전환하는 기능을 수행하며, T/P 어플리케이션만이 이 기능을 수행함.
1	CONTROL_ENABLE_OPERATION	STATE_INITIALIZING 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행하며 T/P 어플리케이션만이 이 기능을 수행함
2	CONTROL_RESET_SAFET_STOP	STATE_SAFE_STOP 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함. 자동모드 일 경우, 프로그램 재시작 여부를 설정할 수 있음.
3	CONTROL_RESET_SAFE_OFF	STATE_SAFE_OFF 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함,
4	CONTROL_RECOVERY_SAFE_STOP	STATE_SAFE_STOP2 상태에서 S/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함.
5	CONTROL_RECOVERY_SAFE_OFF	STATE_SAFE_OFF2 상태에서 S/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함.
6	CONTROL_RECOVERY_BACKDRIVE	STATE_SAFE_OFF2 상태에서 H/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함. STATE_STANDBY 상태로 전환할 수 없고 로봇 제어기 전원을 재부팅해야 함.
7	CONTROL_RESET_RECOVERY	STATE_RECOVERY 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함.

SAFE_OFF상태는 일반적으로 서보온에 해당하는 RESET_SAFE_OFF 명령에 의해 지령대기 상태(STATE_STANDBY)로 전환되며, SAFE_STOP 상태는 RESET_SAFE_STOP 사용자 명령에 의해 지령대기 상태(STATE_STANDBY)로 전환된다. 또한 로봇의 제한 한계치를 넘어서는 오류가 발생할 경우, SAFE_OFF2 상태(모터 및 브레이크 전원 차단)나, SAFE_STOP2 상태(제어 정지)로 전환되는데, 이경우에는 RECOVERY 모드로 전환하여 제한 한계치안으로 로봇을 이동 후 RESET_RECOVERY 명령으로 지령대기 상태(STATE_STANDBY)로 전환해야 오류 발생 없이 정상적으로 로봇 제어를 수행할 수 있다.



2.2.7 프로그램 실행 및 종료

프로그램이 내/외부 오류로 인한 종료 및 정상 종료인 경우에도, 반드시 프로그램 정지(drl_stop) 명령을 수행해야 하며, 프로그램이 내부적으로 완전히 종료 및 정리되는데 다소 시간이 필요함으로, 프로그램 종료에 관한 콜백 함수(TOnProgramStoppedCB)에서 프로그램의 종료 여부를 반드시 확인한 후 필요시 프로그램을 재시작해야 한다.

또한, 프로그램을 가상 로봇 시스템으로 설정하고 프로그램 실행 후 종료하면 자동적으로 실제 로봇 시스템으로 변경됨으로 이에 유의해야 한다.

2.2.8 로봇 안전 설정 기능 제한

로봇 동작(TOOL, TCP 등) 및 안전(안전영역, 안전 정지, 안전 I/O 등)과 관련된 설정 기능은 본 API에서 제공하지 않으며, T/P 어플리케이션에서 직접 설정해야 한다. 로봇 제어기 부팅시 초기화 과정이 T/P 어플리케이션에서만 수행되는 것으로 제한되어 있음으로 로봇 안전 설정은 반드시 T/P 어플리케이션에서 설정해야 한다.

예외적으로, 모션 제어 명령에 사용되는 TOOL이나 TCP 설정은 본 API의 기능으로 제공하고 있으나, 이는 T/P 어플리케이션과 연동되지 않음으로, 로봇 제어기 재부팅시나 프로그램 재기동시 반드시 다시 재등록 후 사용해야 함으로, T/P 어플리케이션에서 설정 후 사용하는 것을 권장한다.

이와 관련하여 로봇 설정은 다음과 같은 로봇 정지 상태(STATE_INITIALIZING, STATE_STANDBY, STATE_SAFE_OFF, STATE_EMERGENCY)에서만 가능하며, 이외에 상태에서 설정을 시도하면 오류가 발생하여 로봇 동작이 정지하면서 그에 관한 로그 및 알람 메시지가 콜백함수(TOnLogAlarmCB)에서 생성된다.

3 정의(Definition)

3.1 상수 및 열거형 정의

3.1.1 enum.ROBOT_STATE

로봇 제어기의 운용 상태 정보를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	STATE_INITIALIZING	T/P 어플리케이션에 의해서 자동으로 진입하는 상태로 각종 파라미터 설정을 위한 초기화 상태임.
1	STATE_STANDBY	운용 가능한 기본 상태 지령 대기 상태임
2	STATE_MOVING	지령 대기 상태에서 지령 수신 후 동작시 자동으로 전환되는 지령 동작 상태임. 동작 완료시 자동 지령 대기 상태로 전환됨.
3	STATE_SAFE_OFF	기능 및 동작 오류로 인한 로봇 정지 모드로, 서보 오프 상태(제어 정지 후 모터 및 브레이크 전원을 차단한 상태)임
4	STATE_TEACHING	직접교시 상태
5	STATE_SAFE_STOP	기능 및 동작 오류로 인한 로봇 정지 모드로, 안전 정지 상태(제어 정지만 수행한 상태, 자동모드인 경우 프로그램 일시 정지 상태)
6	STATE_EMERGENCY_STOP:	비상 정지 상태
7	STATE_HOMMING	홈링 모드 상태(로봇을 하드웨어적으로 정렬하는 상태).
8	STATE_RECOVERY	로봇 구동 범위 초과 등과 같은 오류로 인한 로봇 정지시, 구동 범위 이내로 이동시키기 위한 복구 모드 상태임.
9	eSTATE_SAFE_STOP2	eSTATE_SAFE_STOP 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태
10	STATE_SAFE_OFF2	eSTATE_SAFE_OFF 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태
11	STATE_RESERVED1	예약 사용

순번	상수명	설명
12	STATE_RESERVED2	예약 사용
13	STATE_RESERVED3	예약 사용
14	STATE_RESERVED4	예약 사용
15	STATE_NOT_READY	로봇제어기 부트업이후 초기화를 위한 대기 상태임. T/P 어플리케이션에 의해 초기화 상태로 전환됨.

3.1.2 enum.ROBOT_CONTROL

로봇 제어기의 운용 상태를 전환 및 변경시킬 수 있는 열거형 상수로, 다음과 같이 정의되어 있다

순번	상수명	설명
0	CONTROL_INIT_CONFIG	STATE_NOT_READY 상태에서 STATE_INITIALIZING 상태로 전환하는 기능을 수행하며, T/P 어플리케이션만이 이 기능을 수행함.
1	CONTROL_ENABLE_OPERATION	STATE_INITIALIZING 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행하며 T/P 어플리케이션만이 이 기능을 수행함
2	CONTROL_RESET_SAFET_STOP	STATE_SAFE_STOP 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함. 자동모드 일 경우, 프로그램 재시작 여부를 설정할 수 있음.
3	CONTROL_RESET_SAFET_OFF	STATE_SAFE_OFF 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함,
4	CONTROL_RECOVER_Y_SAFE_STOP	STATE_SAFE_STOP2 상태에서 S/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함.
5	CONTROL_RECOVER_Y_SAFE_OFF	STATE_SAFE_OFF2 상태에서 S/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함.
6	CONTROL_RECOVER_Y_BACKDRIVE	STATE_SAFE_OFF2 상태에서 H/W 적으로 STATE_RECOVERY 상태로 전환하는 기능을 수행함. STATE_STANDBY 상태로 전환할 수 없고 로봇 제어기 전원을 재부팅해야 함.
7	CONTROL_RESET_COVERY	STATE_RECOVERY 상태에서 STATE_STANDBY 상태로 전환하는 기능을 수행함.

3.1.3 enum.MONITORING_SPEED

로봇 제어기의 속도 모드를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SPEED_NORMAL_MODE	정상 속도 모드
1	SPEED_REDUCED_MODE	감속 속도 모드

3.1.4 enum.SPEED_MODE

MONITORING_SPEED 열거형 상수와 동일하게 정의되어 있다.

3.1.5 enum.ROBOT_SYSTEM

로봇 제어기의 로봇 운용 시스템을 의미하는 열거형 상수로 다음과 같이 정의되어 있다.

순번	상수명	설명
0	ROBOT_SYSTEM_REAL	실제 로봇 시스템
1	ROBOT_SYSTEM_VIRTUAL	가상 로봇 시스템

3.1.6 enum.ROBOT_MODE

로봇 제어기의 로봇 운용 모드를 의미하는 열거형 상수로 다음과 같이 정의되어 있다.

순번	상수명	설명
0	ROBOT_MODE_MANUAL	수동 모드
1	ROBOT_MODE_AUTONOMOUS	자동 모드
2	ROBOT_MODE_MEASURE	측정 모드(현재는 지원하지 않음)

3.1.7 enum.ROBOT_SPACE

로봇 제어기에서 로봇을 제어하는 좌표 공간을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	ROBOT_SPACE_JOINT	관절 공간
1	ROBOT_SPACE_TASK	작업 공간

3.1.8 enum.SAFE_STOP_RESET_TYPE

로봇 제어기의 운용 상태가 STATE_SAFE_STOP일 경우, 이를 해제하고, 이후 일련의 동작을 정의하기 위한 열거형 상수로 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SAFE_STOP_RESET_TYPE_DEFAULT	단순 상태 해제(수동모드)
	SAFE_STOP_RESET_TYPE_PROGRAM_STOP	프로그램 종료(자동모드)
1	SAFE_STOP_RESET_TYPE_PROGRAM_RESUME	프로그램 재시작(자동모드)

3.1.9 enum.MANAGE_ACCESS_CONTROL

로봇 제어기의 제어권을 획득 및 변경할 수 열거형 상수로 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MANAGE_ACCESS_CONTROL_FORCE_REQUEST	제어권 강제 회수 메시지 송신
1	MANAGE_ACCESS_CONTROL_REQUEST,	제어권 이양 요청 메시지 송신
2	MANAGE_ACCESS_CONTROL_RESPONSE_YES	제어권 이양 승락 메시지 송신
3	MANAGE_ACCESS_CONTROL_RESPONSE_NO	제어권 이양 거절 메시지 송신

3.1.10 enum.MONITORING_ACCESS_CONTROL

로봇 제어기에서 제어권 변경시 이를 확인할 수 있는 열거형 상수로 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MONITORING_ACCESS_CONTROL_REQUEST,	제어권 이양 요청 메시지 수신
1	MONITORING_ACCESS_CONTROL_DENY	제어권 이양 거절 메시지 수신

순번	상수명	설명
2	MONITORING_ACCESS_CONTROL_GRANT	제어권 획득 메시지 수신
3	MONITORING_ACCESS_CONTROL_LOSS	제어권 손실 메시지 수신

3.1.11 enum.COORDINATE_SYSTEM

로봇 제어기에서 좌표계를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	COORDINATE_SYSTEM_BASE	베이스 좌표계
1	COORDINATE_SYSTEM_TOOL	툴 좌표계
2	COORDINATE_SYSTEM_WORLD	월드 좌표계
101	COORDINATE_SYSTEM_USER_MIN	사용자 좌표계(101~200)
200	COORDINATE_SYSTEM_USER_MAX	사용자 좌표계(101~200)

3.1.12 enum.JOG_AXIS

로봇 제어기에서 조그 제어를 수행할 각 축을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	JOG_AXIS_JOINT_1	로봇의 1번 관절 혹은 축
1	JOG_AXIS_JOINT_2	로봇의 2번 관절 혹은 축
2	JOG_AXIS_JOINT_3	로봇의 3번 관절 혹은 축
3	JOG_AXIS_JOINT_4	로봇의 4번 관절 혹은 축
4	JOG_AXIS_JOINT_5	로봇의 5번 관절 혹은 축
5	JOG_AXIS_JOINT_6	로봇의 6번 관절 혹은 축
6	JOG_AXIS_TASK_X	로봇 TCP의 X 축

순번	상수명	설명
7	JOG_AXIS_TASK_Y	로봇 TCP의 Y 축
8	JOG_AXIS_TASK_Z	로봇 TCP의 Z 축
9	JOG_AXIS_TASK_RX	로봇 TCP의 RX 축
10	JOG_AXIS_TASK_RY	로봇 TCP의 RY 축
11	JOG_AXIS_TASK_RZ	로봇 TCP의 RZ 축

3.1.13 enum.JOINT_AXIS

로봇 제어기에서 관절 공간 좌표계 기준으로 로봇의 각 축을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	JOINT_AXIS_1	로봇의 1번 관절 혹은 축
1	JOINT_AXIS_2	로봇의 2번 관절 혹은 축
2	JOINT_AXIS_3	로봇의 3번 관절 혹은 축
3	JOINT_AXIS_4	로봇의 4번 관절 혹은 축
4	JOINT_AXIS_5	로봇의 5번 관절 혹은 축
5	JOINT_AXIS_6	로봇의 6번 관절 혹은 축

3.1.14 enum.TASK_AXIS

로봇 제어기에서 작업 공간 좌표계 기준으로 로봇의 각 축을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	TASK_AXIS_X	로봇 TCP의 X 축
1	TASK_AXIS_X	로봇 TCP의 Y 축
2	TASK_AXIS_X	로봇 TCP의 Z 축

3.1.15 enum.FORCE_AXIS

로봇 제어기에서 힘 제어를 수행할 경우, 좌표 참조 기준에 대한 정의 기준을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	FORCE_AXIS_X	x축
1	FORCE_AXIS_Y	y축
2	FORCE_AXIS_Z	z축
10	FORCE_AXIS_A	x축 회전
11	FORCE_AXIS_B	y축 회전
12	FORCE_AXIS_C	z축 회전

3.1.16 enum.MOVE_REFERENCE

로봇제어기에서 작업 공간 기준으로 모션 제어를 수행할 경우, 이동할 위치에 대한 정의 기준을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MOVE_REFERENCE_BASE	로봇 베이스 기준
1	MOVE_REFERENCE_TOOL	로봇 TCP 기준
2	MOVE_REFERENCE_WORLD	로봇 월드 좌표계 기준
101	MOVE_REFERENCE_USER_MIN	사용자 좌표계(101~200)
200	MOVE_REFERENCE_USER_MAX	사용자 좌표계(101~200)

3.1.17 enum.MOVE_MODE

로봇 제어기에서 모션 제어를 수행할 경우, 이동할 위치에 대한 표시 방식을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MOVE_MODE_ABSOLUTE	절대 좌표
1	MOVE_MODE_RELATIVE	상대 좌표

3.1.18 enum.FORCE_MODE

로봇 제어기에서 힘 제어를 수행할 경우, 힘 제어를 수행할 방법에 대한 표시 방식을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	FORCE_MODE_ABSOLUTE	절대 좌표
1	FORCE_MODE_RELATIVE	상대 좌표

3.1.19 enum.BLENDING_SPEED_TYPE

로봇 제어기에서 모션 제어 수행 시, 각 경유 지점에 대한 블렌딩 속도 타입을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	BLENDING_SPEED_TYPE_DUPLICATE	선행 모션의 속도와 후행 모션의 속도를 중첩하여 처리
1	BLENDING_SPEED_TYPE_OVERRIDE	선행 모션의 속도를 후행 모션의 속도로 오버라이딩 처리

3.1.20 enum.STOP_TYPE

로봇 제어기에서 모션 제어 수행 중 이를 정지 시킬 수 있는 모션 정지 타입을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	STOP_TYPE_QUICK_STO,	내부 예약 사용
1	STOP_TYPE_QUICK	빠른 정지(모션 궤적 유지)
2	STOP_TYPE_SLOW	느린 정지(모션 궤적 유지)

순번	상수명	설명
3	STOP_TYPE_HOLD	긴급 정지
	STOP_TYPE_EMERGENCY	긴급 정지

3.1.21 enum.MOVEB_BLENDING_TYPE

로봇 제어기에서 MoveB 모션 제어 수행 시, 각 경유 지점에 대한 블렌딩 모션 타입을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MOVEB_BLENDING_TYPE_LINE	직선
1	MOVEB_BLENDING_TYPE_CIRCLE	원호

3.1.22 enum.SPLINE_VELOCITY_OPTION

로봇 제어기에서 Spline 모션 제어 수행 시, 각 경유 지점의 속도 제어 옵션을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SPLINE_VELOCITY_OPTION_DEFAULT	변속 모션
1	SPLINE_VELOCITY_OPTION_CONST	등속 모션

3.1.23 enum.GPIO_CTRLBOX_DIGITAL_INDEX

로봇 제어기의 컨트롤 박스에 장착된 GPIO 디지털 입출력 단자를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

v2 제어기 : 0-15

v3 제어기 : 0-31

순번	상수명	설명
0	GPIO_CTRLBOX_DIGITAL_INDEX_1	컨트롤 박스 GPIO 1번 입출력 포트
1	GPIO_CTRLBOX_DIGITAL_INDEX_2	컨트롤 박스 GPIO 2번 입출력 포트

순번	상수명	설명
2	GPIO_CTRLBOX_DIGITAL_INDEX_3	컨트롤 박스 GPIO 3번 입출력 포트
3	GPIO_CTRLBOX_DIGITAL_INDEX_4	컨트롤 박스 GPIO 4번 입출력 포트
4	GPIO_CTRLBOX_DIGITAL_INDEX_5	컨트롤 박스 GPIO 5번 입출력 포트
5	GPIO_CTRLBOX_DIGITAL_INDEX_6	컨트롤 박스 GPIO 6번 입출력 포트
6	GPIO_CTRLBOX_DIGITAL_INDEX_7	컨트롤 박스 GPIO 7번 입출력 포트
7	GPIO_CTRLBOX_DIGITAL_INDEX_8	컨트롤 박스 GPIO 8번 입출력 포트
8	GPIO_CTRLBOX_DIGITAL_INDEX_9	컨트롤 박스 GPIO 9번 입출력 포트
9	GPIO_CTRLBOX_DIGITAL_INDEX_10	컨트롤 박스 GPIO 10번 입출력 포트
10	GPIO_CTRLBOX_DIGITAL_INDEX_11	컨트롤 박스 GPIO 11번 입출력 포트
11	GPIO_CTRLBOX_DIGITAL_INDEX_12	컨트롤 박스 GPIO 12번 입출력 포트
12	GPIO_CTRLBOX_DIGITAL_INDEX_13	컨트롤 박스 GPIO 13번 입출력 포트
13	GPIO_CTRLBOX_DIGITAL_INDEX_14	컨트롤 박스 GPIO 14번 입출력 포트
14	GPIO_CTRLBOX_DIGITAL_INDEX_15	컨트롤 박스 GPIO 15번 입출력 포트
15	GPIO_CTRLBOX_DIGITAL_INDEX_16	컨트롤 박스 GPIO 16번 입출력 포트
16	GPIO_CTRLBOX_DIGITAL_INDEX_17	컨트롤 박스 GPIO 17번 입출력 포트
17	GPIO_CTRLBOX_DIGITAL_INDEX_18	컨트롤 박스 GPIO 18번 입출력 포트
18	GPIO_CTRLBOX_DIGITAL_INDEX_19	컨트롤 박스 GPIO 19번 입출력 포트
19	GPIO_CTRLBOX_DIGITAL_INDEX_20	컨트롤 박스 GPIO 20번 입출력 포트
20	GPIO_CTRLBOX_DIGITAL_INDEX_21	컨트롤 박스 GPIO 21번 입출력 포트
21	GPIO_CTRLBOX_DIGITAL_INDEX_22	컨트롤 박스 GPIO 22번 입출력 포트

순번	상수명	설명
22	GPIO_CTRLBOX_DIGITAL_INDEX_23	컨트롤 박스 GPIO 23번 입출력 포트
23	GPIO_CTRLBOX_DIGITAL_INDEX_24	컨트롤 박스 GPIO 24번 입출력 포트
24	GPIO_CTRLBOX_DIGITAL_INDEX_25	컨트롤 박스 GPIO 25번 입출력 포트
25	GPIO_CTRLBOX_DIGITAL_INDEX_26	컨트롤 박스 GPIO 26번 입출력 포트
26	GPIO_CTRLBOX_DIGITAL_INDEX_27	컨트롤 박스 GPIO 27번 입출력 포트
27	GPIO_CTRLBOX_DIGITAL_INDEX_28	컨트롤 박스 GPIO 28번 입출력 포트
28	GPIO_CTRLBOX_DIGITAL_INDEX_29	컨트롤 박스 GPIO 29번 입출력 포트
29	GPIO_CTRLBOX_DIGITAL_INDEX_30	컨트롤 박스 GPIO 30번 입출력 포트
30	GPIO_CTRLBOX_DIGITAL_INDEX_31	컨트롤 박스 GPIO 31번 입출력 포트
31	GPIO_CTRLBOX_DIGITAL_INDEX_32	컨트롤 박스 GPIO 32번 입출력 포트

3.1.24 enum.GPIO_CTRLBOX_ANALOG_INDEX

로봇 제어기의 컨트롤 박스에 장착된 GPIO 아날로그 입출력 단자를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	GPIO_CTRLBOX_ANALOG_INDEX_1	컨트롤 박스 GPIO 1번 입출력 포트
1	GPIO_CTRLBOX_ANALOG_INDEX_2	컨트롤 박스 GPIO 2번 입출력 포트

3.1.25 enum.GPIO_ANALOG_TYPE

로봇 제어기의 컨트롤 박스에 장착된 GPIO 아날로그 입출력 단자의 입출력 타입을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	GPIO_ANALOG_TYPE_CURRENT	전류 입출력

순번	상수명	설명
1	GPIO_ANALOG_TYPE_VOLTAGE	전압 입출력

3.1.26 enum.GPIO_TOOL_DIGITAL_INDEX

로봇 끝단에 장착된 GPIO 디지털 입출력 단자를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	GPIO_TOOL_DIGITAL_INDEX_1	로봇 끝단의 GPIO 1번 입출력 포트
1	GPIO_TOOL_DIGITAL_INDEX_2	로봇 끝단의 GPIO 2번 입출력 포트
2	GPIO_TOOL_DIGITAL_INDEX_3	로봇 끝단의 GPIO 3번 입출력 포트
3	GPIO_TOOL_DIGITAL_INDEX_4	로봇 끝단의 GPIO 4번 입출력 포트
4	GPIO_TOOL_DIGITAL_INDEX_5	로봇 끝단의 GPIO 5번 입출력 포트
5	GPIO_TOOL_DIGITAL_INDEX_6	로봇 끝단의 GPIO 6번 입출력 포트

3.1.27 enum.MODBUS_REGISTER_TYPE

로봇 제어기에서 등록할 수 있는 모드버스 레지스터 타입에 관한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MODBUS_REGISTER_TYPE_DISCRETE_INPUTS	Dircrete Input
1	MODBUS_REGISTER_TYPE_COILS	Coils
2	MODBUS_REGISTER_TYPE_INPUT_REGISTER	Input Register
3	MODBUS_REGISTER_TYPE_HOLDING_REGISTER	Holding Register

3.1.28 enum.DRL_PROGRAM_STATE

로봇 제어기에서 프로그램의 실행 상태를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	DRL_PROGRAM_STATE_PLAY	프로그램 실행 상태
1	DRL_PROGRAM_STATE_STOP	프로그램 정지 상태
2	DRL_PROGRAM_STATE_HOLD	프로그램 일시 정지 상태

3.1.29 enum.PROGRAM_STOP_CAUSE

로봇 제어기에서 프로그램 종료 시 종료 이유를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	PROGRAM_STOP_CAUSE_NORMAL	정상 프로그램 종료
1	PROGRAM_STOP_CAUSE_FORCE	강제 프로그램 종료
2	PROGRAM_STOP_CAUSE_ERROR	내/외부 오류로 인한 프로그램 종료

3.1.30 enum.PATH_MODE

경로 수정 모드 사용 시 경로 모드를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	PATH_MODE_DPOS	누적량
1	PATH_MODE_DVEL	증분량

3.1.31 enum.CONTROL_MODE

로봇 제어기 제어 모드를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
3	CONTROL_MODE_POSITION	위치 제어 모드
4	CONTROL_MODE_TORQUE	토크 제어 모드

3.1.32 enum.DATA_TYPE

로봇 제어기에서 모니터링 할 변수의 자료형을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	DATA_TYPE_BOOL	boolean
1	DATA_TYPE_INT	integer
2	DATA_TYPE_FLOAT	float
3	DATA_TYPE_STRING	string
4	DATA_TYPE_POSJ	posj
5	DATA_TYPE_POSX	posx
6	DATA_TYPE_UNKNOWN	알 수 없는 타입

3.1.33 enum.VARIABLE_TYPE

로봇 제어기에서 모니터링 할 변수의 타입을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	VARIABLE_TYPE_INSTALL	설치 변수
1	VARIABLE_TYPE_GLOBAL	일반 변수

3.1.34 enum.SUB_PROGRAM

로봇 제어기에서 서브 프로그램의 작업을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SUB_PROGRAM_DELETE	서브 프로그램 삭제
1	SUB_PROGRAM_SAVE	서브 프로그램 저장

3.1.35 enum.SINGULARITY_AVOIDANCE

특이점 회피 방법을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SINGULARITY_AVOIDANCE_AVOID	자동 회피 모드
1	SINGULARITY_AVOIDANCE_STOP	감속 / warning / task 종료
2	SINGULARITY_AVOIDANCE_VEL	속도 가변

3.1.36 enum.MESSAGE_LEVEL

사용자에게 제공할 메시지의 종류를 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MESSAGE_LEVEL_INFO	정보 알림
1	MESSAGE_LEVEL_WARN	경고 알림
2	MESSAGE_LEVEL_ALARM	에러 알림

3.1.37 enum.POPUP_RESPONSE

팝업 메시지에 대한 사용자 상호작용을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	POPUP_RESPONSE_STOP	현재 구동 중인 Task 종료
1	POPUP_RESPONSE_RESUME	현재 구동 중인 Task 계속해서 실행

3.1.38 enum.MOVE_HOME

Homming 시 참조할 좌표계에 관한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	MOVE_HOME_MECHANIC	기계적 홈 위치[0,0,0,0,0,0] 기준

순번	상수명	설명
1	MOVE_HOME_USER	사용자 지정 홈 위치 기준

3.1.39 enum.BYTE_SIZE

Serial 통신 시 전송할 데이터의 사이즈에 관한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	BYTE_SIZE_FIVEBITS	5 bits
1	BYTE_SIZE_SIXBITS	6 bits
2	BYTE_SIZE_SEVENBITS	7 bits
3	BYTE_SIZE_EIGHTBITS	8 bits

3.1.40 enum.STOP_BITS

Serial 통신 시 프레임의 끝을 알리는 Stopbits에 관한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	STOPBITS_ONE	1 bit
1	STOPBITS_TWO	2 bits

3.1.41 enum.PARITY_CHECK

Serial 통신 시 Parity 체크에 관한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	PARITY_CHECK_NONE	None check
1	PARITY_CHECK_EVEN	Even check
2	PARITY_CHECK_ODD	Odd check

3.1.42 enum.RELEASE_MODE

Protective stop 시 이후 작업을 설정하기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	RELEASE_MODE_STOP	프로그램 정지
1	RELEASE_MODE_RESUME	프로그램 재개
2	RELEASE_MODE_RELEASE	안전 정지 해제
3	RELEASE_MODE_RESET	Interlock Reset

3.1.43 enum.SAFETY_MODE

제어기의 현재 안전 상태를 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SAFETY_MODE_MANUAL	수동 모드
1	SAFETY_MODE_AUTONOMOUS	자동 모드
2	SAFETY_MODE_RECOVERY	복구 모드
3	SAFETY_MODE_BACDRIVE	백드라이브 모드
4	SAFETY_MODE_MEASURE	측정 모드
5	SAFETY_MODE_INITIALIZE	초기화 모드(부팅 시)

3.1.44 enum.SAFETY_STATE

안전 보드의 현재 상태를 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SAFETY_STATE_BP_START	부트업 시작
1	SAFETY_STATE_BP_INIT	부트업 초기화

순번	상수명	설명
2	SAFETY_STATE_VD_STO	모터 정지
3	SAFETY_STATE_VD_SOS	정지
4	SAFETY_STATE_JH_SOS	jog & homing 정지
5	SAFETY_STATE_JH_MOVE	jog & homing 동작
6	SAFETY_STATE_HG_MOVE	직접 교시 동작
7	SAFETY_STATE_RV_SOS	복구 모드 정지
8	SAFETY_STATE_RV_MOVE	복구 모드 동작
9	SAFETY_STATE_RV_BACK	백드라이브 모드
10	SAFETY_STATE_RV_HG_MOVE	복구 모드 직접 교시
11	SAFETY_STATE_SW_SOS	정상 모드 정지
12	SAFETY_STATE_SW_RUN	정상 모드 동작
13	SAFETY_STATE_CW_SOS	협업 모드 정지
14	SAFETY_STATE_CW_RUN	협업 모드 동작
15	SAFETY_STATE_CM_RUN	충돌 무효 영역 동작
16	SAFETY_STATE_AM_RUN	자동 측정 동작
17	SAFETY_STATE_DRL_JH_SOS	DRL jog & homing 정지
18	SAFETY_STATE_DRL_HG_MOVE	DRL 직접 교시 동작

3.1.45 enum.SAFETY_MODE_EVENT

안전 보드의 현재 상태를 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	SAFETY_MODE_EVENT_ENTER	진입
1	SAFETY_MODE_EVENT_MOVE	동작
2	SAFETY_MODE_EVENT_STOP	정지

3.1.46 enum.CO_G_REFERENCE

플랜지의 무게 중심 위치 기준 좌표계를 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	CO_G_REFERENCE_TCP	TCP 기준
1	CO_G_REFERENCE_FLANGE	FLANGE 기준

3.1.47 enum.ADD_UP

플랜지의 작업물 상태를 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	ADD_UP_REPLACE	작업물 교체
1	ADD_UP_ADD	작업물 추가
2	ADD_UP_REMOVE	작업물 제거

3.1.48 enum.OUTPUT_TYPE

플랜지에 설치 된 digital output의 출력 타입을 나타내기 위한 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	OUTPUT_TYPE_PNP	PNP
1	OUTPUT_TYPE_NPN	NPN

3.2 로그 및 알람 정의

3.2.1 LOG_LEVEL

로봇 제어기에서 알람 레벨을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	LOG_LEVEL_RESERVED	내부 예약 상태
1	LOG_LEVEL_SYSINFO	단순 기능 및 동작 오류에 대한 정보용 메시지
2	LOG_LEVEL_SYSWARN	단순 기능 및 동작 오류로 인한 로봇이 정지된 상태
3	LOG_LEVEL_SYSERROR	안전 이슈나 장치 오류로 인한 로봇이 정지된 상태

3.2.2 LOG_GROUP

로봇 제어기에서 알람 그룹별을 의미하는 열거형 상수로, 다음과 같이 정의되어 있다.

순번	상수명	설명
0	LOG_GROUP_RESERVED	
1	LOG_GROUP_SYSTEMFMK	하위 제어기(프레임워크)
2	eLOG_GROUP_MOTIONLIB,	하위 제어기(알고리즘)
3	LOG_GROUP_SMARTTP	상위 제어기 프로그램(GUI)
4	LOG_GROUP_INVERTER	로봇 인버터 보드
5	LOG_GROUP_SAFETYCONTROLLER	안전 보드(Safety Controller)

3.2.3 LOG_CODE

로그 및 알람 코드는 DRSC.h 파일에 enum 변수로 정의되어 있으며, 이에 대한 설명은 별도의 로그 및 알람 코드 정의서 문서 참조 요망.

3.3 콜백 함수 정의

3.3.1 TOnMonitoringStateCB

기능

로봇제어기에서 운용 상태 정보 변경시 이를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eState	enum.ROBOT_STATE	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnMonitoringStateCB(const ROBOT_STATE eState)
2 {
3     switch((unsigned char)eState)
4     {
5         case STATE_SAFE_OFF:
6             // 로봇 제어기 서보온
7             drfl.SetRobotControl(CONTROL_RESET_SAFET_OFF);
8             break;
9         default:
10            break;
11    }
12 }
13
14 int main()
15 {
16     drfl.SetOnMonitoringState(OnMonitoringStateCB);
17 }
```

3.3.2 TOnMonitoringDataCB

기능

로봇 제어기의 현재 위치와 같은 로봇 운용 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
*pData	struct. MONITORING_DATA	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnMonitoringDataCB(const LPMONITORING_DATA pData)
2 {
3     // 조인트 공간 로봇 위치 데이터 표시
4     cout << "joint data "
5     << pData->_tCtrl._tJoint._fActualPos[0]
6     << pData->_tCtrl._tJoint._fActualPos[1]
7     << pData->_tCtrl._tJoint._fActualPos[2]
8     << pData->_tCtrl._tJoint._fActualPos[3]
9     << pData->_tCtrl._tJoint._fActualPos[4]
10    << pData->_tCtrl._tJoint._fActualPos[5] << endl;
11 }
12
13 int main()
14 {
15     drfl.SetOnMonitoringData(OnMonitoringDataCB);
16 }
```

3.3.3 TOnMonitoringDataExCB

기능

로봇 제어기의 현재 위치와 같은 로봇 운용 확장 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(100msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
*pData	struct. MONITORING_DATA_EX	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnMonitoringDataExCB(const LPMONITORING_DATA_EX pData)
2 {
3     // 조인트 공간 로봇 위치 데이터 표시
4     cout << "joint data "
5     << pData->_tCtrl._tJoint._fActualPos[0]
6     << pData->_tCtrl._tJoint._fActualPos[1]
7     << pData->_tCtrl._tJoint._fActualPos[2]
8     << pData->_tCtrl._tJoint._fActualPos[3]
9     << pData->_tCtrl._tJoint._fActualPos[4]
10    << pData->_tCtrl._tJoint._fActualPos[5] << endl;
11 }
12
13 int main()
14 {
15     Drfl.SetOnMonitoringExData(OnMonitoringDataCB);
16 }
```

3.3.4 TOnMonitoringCtrlIOCB

기능

로봇 제어기의 Control Box에 설치되어 있는 I/O 상태 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생 시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
*pCtrlIO	struct. MONITORI NG_CTRLIO	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnMonitoringCtrlIOCB(const LPMONITORING_CTRLIO pCtrlIO)
2 {
3     // 디지털 입력 GPIO 상태 데이터 표시
4     cout << "gpio data" << endl;
5     for (int i = 0; i < NUM_DIGITAL; i++)
6         cout << "DI#" << i << ":" << pCtrlIO->_tInput._iActualDI[i] <<
7         endl;
8 }
9 int main()
10 {
11     Drfl.SetOnMonitoringCtrlIO(OnMonitoringCtrlIOCB)
12 }
```

3.3.5 TOnMonitoringCtrlIOExCB

기능

로봇 제어기의 Control Box에 설치되어 있는 I/O 상태 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
*pCtrlIO	struct. MONITORING_CTRLIO_E X	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnMonitoringCtrlIOExCB(const LPMONITORING_CTRLIO_EX pCtrlIO)
2 {
3     // 디지털 입력 GPIO 상태 데이터 표시
4     cout << "gpio data" << endl;
5     for (int i = 0; i < NUM_DIGITAL; i++)
6         cout << "DI#"<< i << ":" << pCtrlIO->_tInput._iActualDI[i] <<
7         endl;
8 }
9
10 int main()
11 {
12     Drfl.SetOnMonitoringCtrlIOEx(OnMonitoringCtrlIOExCB)
13 }
```

3.3.6 TOnMonitoringModbusCB**기능**

로봇 제어기에 등록된 모드버스 I/O 상태 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
*pModbus	struct. MONITORING_MODBUS	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnMonitoringModbusCB(const LPMONITORING_MODBUS pModbus)
2 {
3     // 모드버스 IO 상태 데이터 표시
4     cout << "modbus data" << endl;
```

```

5     for (int i = 0; i < pModbus->_iRegCount; i++)
6         cout << pModbus->_tRegister[i]._szSymbol << ":" "
7             << pModbus->_tRegister[i]._iValue<< endl;
8     }
9
10    int main()
11    {
12        Drfl.SetOnMonitoringModbus(OnMonitoringModbusCB)
13    }

```

3.3.7 TOnLogAlarmCB

기능

로봇 제어기에서 발생하는 모든 알람 및 로그 정보 데이터를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
* pLogAlarm	struct.LOG_ALARM	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnLogAlarm(LPLOG_ALARM pLogAlarm)
2 {
3     switch(pLogAlarm->_iGroup)
4     {
5         case LOG_GROUP_SYSTEMFMK:
6             switch(pLogAlarm->_iLevel)
7             {
8                 case LOG_LEVEL_SYSINFO:
9                     cout << "index(" << pLogAlarm->_iIndex << "), ";
10                    cout << "param(" << pLogAlarm->_szParam[0]<< ", ";
11                    cout << "param(" << pLogAlarm->_szParam[1]<< ", ";
12                    cout << "param(" << pLogAlarm->_szParam[2]<< ")" << endl;
13                     break;
14                 default:
15                     break;
16             }
}

```

```

17     break;
18     default:
19         break;
20     }
21 }
22
23 int main()
24 {
25     Drfl.SetOnLogAlarm(OnLogAlarmCB)
26 }
```

3.3.8 TOnMonitoringAccessControlCB

기능

로봇 제어기의 제어권 상태(요청/승락/거절) 변경시 이를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eAccCtrl	enum. MONITORING_ACCESS_CONTROL	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnMonitoringAccessControlCB(const MONITORING_ACCESS_CONTROL eAccCtrl)
2 {
3     // 제어권 이양 메시지 수신
4     case MONITORING_ACCESS_CONTROL_REQUEST:
5         // 제어권 이양 거부
6         drfl.ManageAccessControl(MANAGE_ACCESS_CONTROL_RESPONSE_NO);
7         break;
8     default:
9         break;
10 }
11
12 int main()
13 {
14     Drfl.SetOnMonitoringAccessControl (OnMonitoringAccessControlCB);
```

15 }

3.3.9 TOnHommingCompletedCB

기능

로봇 제어기가 홈밍 제어 모드에 있을 경우, 홈밍 완료 여부를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

없음

리턴

없음

예제

```

1 void OnHommingCompletedCB()
2 {
3     // 홈밍 완료 메시지 생성
4     cout << "homming completed" << endl;
5     drfl.Homme(False)
6 }
7
8 int main()
9 {
10    Drfl.SetOnHommingCompleted(OnHommingCompletedCB);
11 }
```

3.3.10 TOnTpInitializingCompletedCB

기능

로봇 제어기 부팅후, T/P 어플리케이션에 의해 로봇 제어기가 초기화 과정을 수행할 경우, 초기화 완료 여부 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

없음

리턴

없음

예제

```

1 void OnTpInitializingCompletedCB()
2 {
3     // Tp 초기화 여부 확인후 제어권 요청.
4     cout << "tp initializing completed" << endl;
5     drfl.ManageAccessControl(MANAGE_ACCESS_CONTROL_REQUEST);
6 }
7
8 int main()
9 {
10    Drfl.SetOnTpInitializingCompleted(OnTpInitializingCompletedCB);
11 }
```

3.3.11 TOnMonitoringSpeedModeCB**기능**

로봇 제어기의 현재 속도 모드를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eSpeedMode	enum.MONITORING_SPEED	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnMonitoringSpeedModeCB(const MONITORING_SPEED eSpdMode)
2 {
3     // 속도 모드 표시
4     cout << "speed mode: " << (int)eSpeedMode << endl;
5 }
6 }
```

```

7 int main()
8 {
9     Drfl.SetOnMonitoringSpeedMode(OnMonitoringSpeedModeCB);
10 }

```

3.3.12 TOnMasteringNeedCB

기능

로봇 제어기에서 외부 충격으로 인해 로봇의 축이 틀어졌을 경우, 이를 확인하기 위한 콜백함수이다. Home 명령을 통해 로봇의 축을 다시 정렬할 수 있으며, SetOnHommingCompleted 콜백 함수를 통해서 축 정렬이 완료되었음을 확인할 수 있다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

없음

리턴

없음

예제

```

1 void OnMasteringNeedCB()
2 {
3     //로봇 축 정렬을 위한 홈밍 모드 시작
4     drfl.Homme(True)
5 }
6
7 int main()
8 {
9     Drfl.SetOnMasteringNeedCB(TOnMasteringNeedCB);
10 }

```

3.3.13 TOnProgramStoppedCB

기능

로봇 제어기에서 자동모드로 프로그래밍 실행 도중 오류나 사용자 명령에 의해서 종료되었을 경우, 프로그램 실행이 완전히 종료되었음을 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eStopCause	enum. PROGRAM_STOP_CAUSE	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnProgramStoppedCB(const PROGRAM_STOP_CAUSE eStopCause)
2 {
3     //프로그램을 재시작 가능 상태 표시
4 }
5
6 int main()
7 {
8     Drfl.SetOnProgramStopped(OnProgramStoppedCB);
9 }
```

3.3.14 TOnDisconnectedCB**기능**

로봇 제어기와 연결이 외부 요인이나 혹은 사용자에 의해서 종료되었을 경우, 이를 확인하기 위한 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

없음

리턴

없음

예제

```
1 void OnDisconnectedCB()
```

```

2 {
3     //로봇 제어기 재접속 및 오류 처리 필요
4 }
5
6 int main()
7 {
8     Drfl.SetOnDisconnected (OnDisconnectedCB);
9 }
```

3.3.15 TOnTpPopupCB

기능

로봇 제어기에서 사용자 팝업 기능을 사용하였을 때 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
tPopup	struct.MESSAGE_POPUP	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnTpPopupCB(LPMESSAGE_POPUP tPopup)
2 {
3     //사용자 팝업 생성 시 수행
4 }
5
6 int main()
7 {
8     Drfl.SetOnTpPopup(OnTpPopupCB);
9 }
```

3.3.16 TOnTpLogCB

기능

로봇 제어기에서 사용자 로그 기능을 사용하였을 때 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
strLog	문자열	-	256바이트 문자열

리턴

없음

예제

```

1 void OnTpLogCB(const char* strLog)
2 {
3     //사용자 로그 출력 시 수행
4 }
5
6 int main()
7 {
8     Drfl.SetOnTpLog(OnTpLogCB);
9 }
```

3.3.17 TOnTpGetUserInputCB

기능

로봇 제어기에서 사용자 입력 기능을 사용하였을 때 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
tInput	struct.MESSAGE_INPU T	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnTpGetUserInputCB(LPMESSAGE_INPUT tInput)
2 {
3     //사용자 입력 받을 시 수행
4 }
5
6 int main()
7 {
8     Drfl.SetOnTpGetUserInput(OnTpGetUserInputCB);
9 }
```

3.3.18 TOnTpProgressCB**기능**

로봇 제어기에서 실행 단계의 정보를 출력하였을 때 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
tProgress	struct.MESSAGE_PROGRESS	-	구조체 정의 참조

리턴

없음

예제

```

1 void OnTpProgressCB(LPMESSAGE_PROGRESS tProgress)
2 {
3     //실행 단계 정보 출력 시 수행
4 }
5
6 int main()
7 {
8     Drfl.SetOnTpProgress(OnTpProgressCB);
```

9 }

3.3.19 TOnMonitoringRobotSystemCB

기능

로봇 제어기에서 로봇 운용 상태가 변경되었을 때 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eRobotState	enum.ROBOT_STATE	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnMonitoringRobotSystemCB(ROBOT_STATE eRobotState)
2 {
3     //로봇 운용 상태 변경 시 수행
4 }
5
6 int main()
7 {
8     Drfl.set_on_monitroing_robot_system(OnMonitoringRobotSystemCB);
9 }
```

3.3.20 TOnMonitoringSafetyStateCB

기능

로봇 제어기에서 설치/삭제 명령 이후, 현재 Safety 상태를 자동으로 확인하기 위해 호출되는 콜백함수이다. 콜백 함수는 특정 이벤트 발생시 자동으로 실행되므로, 콜백 함수 내에서 과도한 실행 시간(50msec 이내)을 요구하는 코드를 작성해서는 안 된다.

인수

인수명	자료형	기본값	설명
eSafetyState	enum.SafetyState	-	상수 및 열거형 정의 참조

리턴

없음

예제

```

1 void OnMonitoringSafetyStateCB (SafetyState iState)
2 {
3     //Safety State 업데이트 시 수행
4 }
5
6 int main()
7 {
8     Drfl.set_on_monitoring_safety_state(OnMonitoringSafetyStateCB);
9 }
```

3.4 구조체 정의

- struct.SYSTEM_VERSION(p. 68)
- struct.MONITORING_DATA(p. 69)
- struct.MONITORING_DATA_EX(p. 72)
- struct.MONITORING_CTRLIO(p. 77)
- struct.MONITORING_CTRLIO_EX(p. 79)
- struct.MONITORING_MODBUS(p. 80)
- struct.LOG_ALARM(p. 81)
- struct.MOVE_POSB(p. 81)
- struct.ROBOT_POSE(p. 82)
- struct.USER_COORDINATE(p. 82)
- struct.MESSAGE_POPUP(p. 82)
- struct.MESSAGE_INPUT(p. 83)
- struct.MESSAGE_PROGRESS(p. 83)
- struct.FLANGE_SERIAL_DATA(p. 83)
- struct.FLANGE_SER_RXD_INFO(p. 84)
- struct.READ_FLANGE_SERIAL(p. 85)
- struct.INVERSE_KINEMATIC_RESPONSE(p. 85)
- struct.UPDATE_SW_MODULE_RESPONSE(p. 85)
- struct.CONFIG_JOINT_RANGE(p. 86)

- [struct.GENERAL_RANGE\(p. 86\)](#)
- [struct.CONFIG_GENERAL_RANGE\(p. 86\)](#)
- [struct.POINT_2D\(p. 86\)](#)
- [struct.POINT_3D\(p. 87\)](#)
- [struct.LINE\(p. 87\)](#)
- [union.CONFIG_SAFETY_FUNCTION\(p. 87\)](#)
- [struct._tStopCode\(p. 88\)](#)
- [struct.CONFIG_INSTALL_POSE\(p. 88\)](#)
- [struct.CONFIG_SAFETY_IO\(p. 88\)](#)
- [struct.CONFIG_SAFETY_IO_EX\(p. 89\)](#)
- [union.VIRTUAL_FENCE_OBJECT\(p. 89\)](#)
- [struct.CONFIG_VIRTUAL_FENCE\(p. 89\)](#)
- [struct.CONFIG_SAFE_ZONE\(p. 90\)](#)
- [struct.ENABLE_SAFE_ZONE\(p. 90\)](#)
- [struct.SAFETY_OBJECT_SPHERE\(p. 90\)](#)
- [struct.SAFETY_OBJECT_CAPSULE\(p. 90\)](#)
- [struct.SAFETY_OBJECT_CUBE\(p. 91\)](#)
- [struct.SAFETY_OBJECT_OBB\(p. 91\)](#)
- [struct.SAFETY_OBJECT_POLYPRISM\(p. 91\)](#)
- [union.SAFETY_OBJECT_DATA\(p. 91\)](#)
- [struct.SAFETY_OBJECT\(p. 92\)](#)
- [struct.CONFIG_PROTECTED_ZONE\(p. 92\)](#)
- [struct.CONFIG_COLLISION_MUTE_ZONE_PROPERTY\(p. 93\)](#)
- [struct.CONFIG_COLLISION_MUTE_ZONE\(p. 93\)](#)
- [struct.SAFETY_TOOL_ORIENTATION_LIMIT\(p. 93\)](#)
- [struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE\(p. 94\)](#)
- [struct.CONFIG_NUDGE\(p. 94\)](#)
- [struct.CONFIG_COCKPIT_EX\(p. 94\)](#)
- [struct.CONFIG_IDLE_OFF\(p. 95\)](#)
- [struct.WRITE_MODBUS_DATA\(p. 95\)](#)
- [struct.WRITE_MODBUS_RTU_DATA\(p. 95\)](#)
- [struct.MODBUS_DATA\(p. 96\)](#)
- [struct.MODBUS_DATA_LIST\(p. 96\)](#)
- [struct.CONFIG_WORLD_COORDINATE\(p. 96\)](#)
- [struct.CONFIG_CONFIGURABLE_IO\(p. 97\)](#)
- [struct.CONFIG_CONFIGURABLE_IO_EX\(p. 97\)](#)
- [struct.CONFIG_TOOL_SHAPE\(p. 97\)](#)
- [struct.CONFIG_TOOL_SYMBOL\(p. 97\)](#)
- [struct.CONFIG_TCP_SYMBOL\(p. 98\)](#)
- [struct.CONFIG_TOOL_LIST\(p. 98\)](#)
- [struct.CONFIG_TOOL_SHAPE_SYMBOL\(p. 98\)](#)
- [struct.CONFIG_TCP_LIST\(p. 98\)](#)
- [struct.CONFIG_TOOL_SHAPE_LIST\(p. 99\)](#)

- struct.SAFETY_CONFIGURATION_EX(p. 99)
- struct.SAFETY_CONFIGURATION_EX2(p. 101)
- struct.SAFETY_CONFIGURATION_EX2_V3(p. 103)
- struct.ROBOT_VEL(p. 105)
- struct.ROBOT_FORCE(p. 105)
- struct.CONFIG_SAFETY_IO_OP(p. 105)
- struct.MONITORING_CTRLIO_EX2(p. 106)
- struct.ROBOT_WELDING_DATA(p. 107)
- struct.WELDING_CHANNEL(p. 108)
- struct.CONFIG_WELDING_INTERFACE(p. 109)
- struct.CONFIG_WELD_SETTING(p. 110)
- struct.GET_WELDING_SETTING_RESPONSE(p. 111)
- struct.ADJUST_WELDING_SETTING(p. 112)
- struct.CONFIG_TRAPEZOID_WEAVERS_SETTING(p. 113)
- struct.CONFIG_ZIGZAG_WEAVERS_SETTING(p. 114)
- struct.CONFIG_CIRCULE_WEAVERS_SETTING(p. 115)
- struct.CONFIG_SINE_WEAVERS_SETTING(p. 115)
- struct.CONFIG_WELDING_DETAIL_INFO(p. 116)
- struct.CONFIG_ANALOG_WELDING_INTERFACE(p. 117)
- struct.CONFIG_ANALOG_WELDING_SETTING(p. 118)
- struct.ANALOG_WELDING_ADJUST_SETTING(p. 119)
- struct.CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA(p. 120)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS(p. 121)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_MODE(p. 122)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_TEST(p. 122)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION(p. 123)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_OPTION(p. 124)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2(p. 125)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING(p. 125)
- struct.CONFIG_DIGITAL_WELDING_INTERFACE_OTHER(p. 126)
- struct.DIGITAL_WELDING_RESET(p. 127)
- struct.CONFIG_DIGITAL_WELDING_MODE(p. 127)
- struct.CONFIG_DIGITAL_WELDING_CONDITION(p. 127)
- struct.CONFIG_DIGITAL_WELDING_ADJUST(p. 130)
- struct.MEASURE_TCP_WELDING(p. 131)
- struct.TACK_WELDING_SETTING(p. 131)
- struct.DIGITAL_FORCE_WRITE_DATA(p. 131)
- struct.ROBOT_DIGITAL_WELDING_DATA(p. 132)
- struct.DIGITAL_WELDING_COMM_STATE(p. 134)

3.4.1 struct.SYSTEM_VERSION

로봇 제어기에서 상세 버전 정보를 확인하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	버전 정보	char	-	상위제어기 정보 (32byte)
32	버전 정보	char	-	하위제어기 정보 (32byte)
64	버전 정보	char	-	DRL 버전 번호 (32bytes)
96	버전 정보	char	-	인버터 정보 (32byte)
128	버전 정보	char	-	Safety Baord 정보(32byte)
160	버전 정보	char	-	로봇 시리얼 번호 (32byte)
192	버전 정보	char	-	로봇 모델 번호 (32byte)
224	버전 정보	char		JTS Board번호 (32byte)
256	버전 정보	char		Flange Board 번호 (32byte)

3.4.2 struct.MONITORING_DATA

로봇 제어기에서 로봇 운용 데이터 정보를 확인하기 위한 구조체 정보로 다음과 같은 5개의 운용 정보로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	운용 정보(#1)	-	-	제어관련 정보
2	운용 정보(#2)	-	-	관절공간(Joint Space) 정보
146	운용 정보(#3)	-	-	작업공간(Task Space) 정보
327	운용 정보(#4)	-	-	토크 및 외력 관련 정보
423	운용 정보(#5)	-	-	기타 로봇 관련 입력 정보

운용 정보(#1)은 다음과 같이 현재 로봇의 움직임에 관한 제어 모드 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	제어 모드	uchar	0x00~0x01	위치제어: 0 토크제어: 1
1	제어 공간	uchar	0x00~0x01	관절각도 공간: 0 작업좌표 공간: 1

운용 정보(#2)는 다음과 같이 관절공간(로봇의 조인트 기반)에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	위치 정보	float	-	6개의 현재 조인트 각도 정보
24	위치 정보	float	-	6개의 현재 조인트 각도 정보 (Absolute Encoder)
48	속도 정보	float	-	6개의 현재 조인트 속도 정보
72	오차 정보	float	-	6개의 현재 조인트 오차 정보
96	위치 정보	float	-	6개의 타켓 조인트 위치 정보
120	위치 정보	float	-	6개의 타켓 조인트 속도 정보

운용 정보(#3)은 다음과 같이 작업공간(로봇의 Flange에 장착된 Tool 기반)에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	위치 정보 (Tool)	float	-	6개의 현재 Tool 위치 정보

BYTE#	필드명	데이터 형	값	비고
24	위치 정보 (Flange)	float	-	6개의 현재 Flange 위치 정보
48	속도 정보	float	-	6개의 현재 Tool 속도 정보
72	오차 정보	float	-	6개의 현재 Tool 오차 정보
96	위치 정보	float	-	6개의 타겟 Tool 위치 정보
120	속도 정보	float	-	6개의 타겟 Tool 속도 정보
144	자세 정보	uchar	0x00~0x07	로봇의 자세 정보

- 자세 정보는 로봇이 한 지점을 가리킬 수 있는 8가지 자세 정보 중 하나이다.
- 운용 정보(#3)의 속도 정보는 X, Y, Z, Rx, Ry, Rz의 현재 및 타겟 속도 정보이다.

운용정보(#4)는 다음과 같이 토크제어 모드에서 로봇의 움직임에 관한 토크 및 외력 등의 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	토크정보	float	-	6개의 현재 계산된 다이나믹 토크 정보
24	토크정보	float	-	6개의 현재 측정된 JTS 센서 정보
48	외력정보	float	-	6개의 현재 Joint 축별 외력 정보
72	외력정보	float	-	6개의 현재 Tool 기반 외력 정보

운용정보(#5)은 다음과 같이 기타 로봇 관련 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	시간 정보	double	-	내부 클락 카운터 정보
8	I/O 정보(#1)	uchar	-	6개의 Digital On/Off정보
14	I/O정보(#2)	uchar	-	6개의 Digital On/Off정보
20	브레이크 정보	uchar	-	6개의 브레이크 상태 정보
26	버튼 정보	uint	-	5개의 로봇 버튼 정보
46	전류 정보	float	-	6개 모터의 전류 소모량 정보
70	온도 정보	float	-	6개 인버터 온도 정보

- 브레이크 정보는 0이면 풀린 상태이며, 1이면 잠긴 상태로 로봇을 운용할 수 없는 상태이다.(현재는 지원되지 않는다)
- I/O 정보에 대한 설명은 다음과 같다.

I/O 정보	설명
I/O 정보(#1)	로봇 끝단에 부착되어 있는 6의 Digital Input 정보
I/O 정보(#2)	로봇 끝단에 부착되어 있는 6의 Digital Output 정보

- 버튼 정보는 6축에 부착되어 있는 5개의 푸쉬 버튼의 On/Off 상태를 의미한다.

3.4.3 struct.MONITORING_DATA_EX

로봇 제어기에서 로봇 운용 데이터 정보를 확인하기 위한 구조체 정보로 다음과 같은 7개의 운용 정보로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	운용 정보(#1)	-	-	제어관련 정보
2	운용 정보(#2)	-	-	관절공간(Joint Space) 정보
146	운용 정보(#3)	-	-	작업공간(Task Space) 정보
327	운용 정보(#4)	-	-	토크 및 외력 관련 정보

BYTE#	필드명	데이터 형	값	비고
439	운용 정보(#5)	-	-	월드 좌표 관련 정보
643	운용 정보(#6)	-	-	유저 좌표 관련 정보
825	운용 정보(#7)	-	-	기타 로봇 관련 입력 정보
919	예약 공간(#1)	-	-	120 바이트 예약 공간
1039	예약 공간(#2)	-	-	120 바이트 예약 공간

운용 정보(#1)은 다음과 같이 현재 로봇의 움직임에 관한 제어 모드 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	제어 모드	uchar	0x00~0x01	위치제어: 0 토크제어: 1
1	제어 공간	uchar	0x00~0x01	관절각도 공간: 0 작업좌표 공간: 1

운용 정보(#2)는 다음과 같이 관절공간(로봇의 조인트 기반)에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	위치 정보	float	-	6개의 현재 조인트 각도 정보
24	위치 정보	float	-	6개의 현재 조인트 각도 정보 (Absolute Encoder)
48	속도 정보	float	-	6개의 현재 조인트 속도 정보
72	오차 정보	float	-	6개의 현재 조인트 오차 정보
96	위치 정보	float	-	6개의 타켓 조인트 위치 정보

BYTE#	필드명	데이터 형	값	비고
120	위치 정보	float	-	6개의 타겟 조인트 속도 정보

운용 정보(#3)은 다음과 같이 작업공간(로봇의 Flange에 장착된 Tool 기반)에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	위치 정보 (Tool)	float	-	6개의 현재 Tool 위치 정보
24	위치 정보 (Flange)	float	-	6개의 현재 Flange 위치 정보
48	속도 정보	float	-	6개의 현재 Tool 속도 정보
72	오차 정보	float	-	6개의 현재 Tool 오차 정보
96	위치 정보	float	-	6개의 타겟 Tool 위치 정보
120	속도 정보	float	-	6개의 타겟 Tool 속도 정보
144	자세 정보	uchar	0x00~0x07	로봇의 자세 정보
145	회전 행렬	float	-	3x3 행렬 정보

- 자세 정보는 로봇이 한 지점을 가리킬 수 있는 8가지 자세 정보 중 하나이다.
- 운용 정보(#3)의 속도 정보는 X, Y, Z, Rx, Ry, Rz의 현재 및 타겟 속도 정보이다.

운용정보(#4)는 다음과 같이 토크제어 모드에서 로봇의 움직임에 관한 토크 및 외력 등의 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	토크정보	float	-	6개의 현재 계산된 다이나믹 토크 정보
24	토크정보	float	-	6개의 현재 측정된 JTS 센서 정보
48	외력정보	float	-	6개의 현재 Joint 축별 외력 정보
72	외력정보	float	-	6개의 현재 Tool 기반 외력 정보

운용정보(#5)는 다음과 같이 월드 좌표계 공간에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	World 좌표계와 Base 좌표계간 관계	float	-	6개의 위치 편차 정보
24	위치 정보 (Tool)	float	-	6개의 현재 Tool 위치 정보
48	위치 정보 (Flange)	float	-	6개의 현재 Flange 위치 정보
72	속도 정보	float	-	6개의 현재 Tool 속도 정보
96	외력 정보	float	-	6개의 현재 Tool 기반 외력 정보
120	위치 정보	float	-	6개의 타겟 Tool 위치 정보
144	속도 정보	float	-	6개의 타겟 Tool 속도 정보
168	회전 행렬	float	-	3x3 행렬 정보

운용정보(#6)은 다음과 같이 유저 좌표계 공간에서 로봇의 움직임에 관한 제어 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	User 좌표계 ID	float	-	좌표계 ID 정보(101~200)

BYTE#	필드명	데이터 형	값	비고
1	User 좌표계의 Parent 좌표계	float	-	Base 좌표: 0 World 좌표: 2
2	위치 정보 (Tool)	float	-	6개의 현재 Tool 위치 정보
26	위치 정보 (Flange)	float	-	6개의 현재 Flange 위치 정보
50	속도 정보	float	-	6개의 현재 Tool 속도 정보
74	외력 정보	float	-	6개의 현재 Tool 기반 외력 정보
98	위치 정보	float	-	6개의 타겟 Tool 위치 정보
122	속도 정보	float	-	6개의 타겟 Tool 속도 정보
146	회전 행렬	float	-	3x3 행렬 정보

운용정보(#7)은 다음과 같이 기타 로봇 관련 입출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	시간 정보	double	-	내부 클락 카운터 정보
8	I/O 정보(#1)	uchar	-	6개의 Digital On/Off 정보
14	I/O 정보(#2)	uchar	-	6개의 Digital On/Off 정보
20	브레이크 정보	uchar	-	6개의 브레이크 상태 정보
26	버튼 정보	uint	-	5개의 로봇 버튼 정보
46	전류 정보	float	-	6개 모터의 전류 소모량 정보
70	온도 정보	float	-	6개 인버터 온도 정보

예약 공간 (#1)에는 운용정보의 확장을 위한 예비 공간이다. 버튼 정보는 운용정보(#7)의 버튼 정보와 동일하나, 자료형 및 버튼 크기가 변경된 정보를 포함하고 있다.

BYTE#	필드명	데이터 형	값	비고
0	버튼 정보	uchar	-	6개의 로봇 버튼 정보

버튼 정보는 운용정보(#7)의 버튼 정보와 동일하나, 자료형 및 버튼 크기가 변경된 정보를 포함하고 있다.

예약 공간(#2)에는 소형 모델 관련하여 다음과 같은 입출력 정보가 제공된다.

BYTE#	필드명	데이터 형	값	비고
0	토크 정보	float	-	6개의 현재 측정된 FTS 센서 정보
24	토크 정보	float	-	6개의 현재 측정된 CS 센서 정보
48	속도 정보	float	-	3개의 현재 측정된 가속도 센서 정보
60	특이점 정보	uchar	-	Minimum Singular Value

- 브레이크 정보는 0이면 풀린 상태이며, 1이면 잠긴 상태로 로봇을 운용할 수 없는 상태이다.(현재는 지원되지 않는다)
- I/O 정보에 대한 설명은 다음과 같다.

I/O 정보	설명
I/O 정보(#1)	로봇 끝단에 부착되어 있는 6의 Digital Input 정보
I/O 정보(#2)	로봇 끝단에 부착되어 있는 6의 Digital Output 정보

- 버튼 정보는 6축에 부착되어 있는 5개의 푸쉬 버튼의 On/Off 상태를 의미한다.

3.4.4 struct.MONITORING_CTRLIO

로봇 제어기에서 컨트롤 박스에 장착되어 있는 I/O의 현재 상태 정보를 확인하기 위한 구조체 정보는 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O 정보(#1)	uchar	-	16개의 Digital On/Off정보

BYTE#	필드명	데이터 형	값	비고
16	I/O 정보(#2)	float		2개의 Analog 수치 정보
24	I/O 정보(#3)	uchar	-	3개의 Switch On/Off 정보
27	I/O 정보(#4)	uchar		2개의 Safety On/Off 정보
29	I/O 정보(#5)	uchar		2개의 Encorder On/Off 정보
31	I/O 정보(#6)	uint		2개의 Encorder 수치 정보
39	I/O 정보(#7)	uchar	-	16개의 Digital On/Off정보
55	I/O 정보(#8)	float		2개의 Analog 수치 정보

I/O 정보에 대한 설명은 다음과 같다.

I/O 정보	설명
I/O 정보(#1)	컨트롤 박스에 부착되어 있는 16개의 Digital Input 정보
I/O 정보(#2)	컨트롤 박스에 부착되어 있는 2개의 Analog Input 정보
I/O 정보(#3)	직접교시 버튼 등과 같은 컨트롤 박스 및 T/P에 부착되어 있는 3개의 스위치 상태 정보
I/O 정보(#4)	컨트롤 박스에 부착되어 있는 Safety 관련2개의 Input 정보
I/O 정보(#5)	컨트롤 박스에 부착되어 있는 Encorder 관련2개의 Input 정보
I/O 정보(#6)	컨트롤 박스에 부착되어 있는 Encorder 관련2개의 raw data 정보
I/O 정보(#7)	컨트롤 박스에 부착되어 있는 16개의 Digital Output 정보
I/O 정보(#8)	컨트롤 박스에 부착되어 있는 2개의 Analog Output 정보

3.4.5 struct.MONITORING_CTRLIO_EX

로봇 제어기에서 컨트롤 박스에 장착되어 있는 I/O의 현재 상태 정보를 확인하기 위한 구조체 정보는 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O 정보(#1)	-	-	I/O 신호 입력 정보
31	I/O 정보(#2)	-	-	I/O 신호 출력 정보
57	I/O 정보(#3)	-	-	Encoder 신호 데이터 정보
69	예약 공간	-	-	24바이트 예약 공간

I/O 정보(#1)은 다음과 같이 컨트롤 박스 내 Safety Board에 부착된 I/O 입력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Digital 신호	uchar	0x00~0x01	16개의 Digital On/Off 정보
16	Analog 신호	float	-	2개의 Analog 수치 정보
24	Switch 신호	uchar	0x00~0x01	3개의 Switch On/Off 정보
27	Safety 신호	uchar	0x00~0x01	2개의 Safety On/Off 정보
29	Analog 모드	uchar	0x00~0x01	2개의 Analog 모드 정보 전류: 0 전압: 1

Switch 신호 정보 직접교시 버튼 등과 같은 컨트롤 박스 및 T/P에 부착되어 있는 3개의 스위치 상태 정보이며, Safety 신호는 컨트롤 박스에 부착되어 있는 Safety Emergency 입력 신호와 Protective-Stop 신호 2개의 입력 상태 정보이다.

운용 정보(#2)는 컨트롤 박스 내 Safety Board에 부착된 I/O 출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Digital 신호	uchar	0x00~0x01	16개의 Digital On/Off 정보
16	Analog 신호	float	-	2개의 Analog 수치 정보

BYTE#	필드명	데이터 형	값	비고
24	Analog 모드	uchar	0x00~0x01	2개의 Analog 모드 정보 전류: 0 전압: 1

운용 정보(#3)은 다음과 같이 컨트롤 박스 내 Safety Board에 부착된 Encoder 데이터 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Strobe 신호 정보	uchar	0x00~0x01	2개의 Encoder On/Off 정보
2	RAW 데이터 정보	uint	-	2개의 Encoder 수치 정보
10	Reset 신호 정보	uchar	0x00~0x01	2개의 Encoder Reset 정보

예약 공간에는 소형 모델 관련하여 다음과 같은 입출력 정보가 제공된다.

BYTE#	필드명	데이터 형	값	비고
0	프로세스 버튼의 Digital 입력 신호	uchar	0x00~0x01	4개의 Digital On/Off 정보

3.4.6 struct.MONITORING_MODBUS

로봇 제어기에서 등록된 모드버스 I/O 정보가 있을 경우, 모드버스 I/O의 현재 상태 정보를 확인하기 위한 구조체로, 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O 개수	ushort	-	Modbus I/O 신호 개수
2	I/O 정보(#1)	-	-	Modbus I/O 신호 정보
...	Modbus I/O 신호 정보
2+(100*34)	I/O 정보(#100)	-	-	Modbus I/O 신호 정보

- I/O 정보(#N)는 모드버스 I/O 정보를 식별하기 위한 다음과 같이 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	I/O 이름	char	-	Modbus I/O 신호 이름 (32bytes)

BYTE#	필드명	데이터 형	값	비고
2	I/O 상태	ushort	-	Modbus I/O 신호 값

3.4.7 struct.LOG_ALARM

로봇 제어기에서 내/외부적 요인에 의해서 로봇 동작에 관한 기능 및 동작 오류 발생 시 이를 확인하기 위한 구조체로, 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	로그 레벨	uchar	-	Info: 0 Warning: 1 Alarm: 2
1	분류 번호	uchar	-	분류 코드
2	로그 번호	uint	-	메시지 번호
6	파라미터(#1)	char	-	예약 공간: 256
262	파라미터(#2)	char	-	예약 공간: 256
518	파라미터(#3)	char	-	예약 공간: 256

분류 및 오류 메시지는 사전에 정의된 내용을 번호를 통해서 전달하며, 필요 시 관련 파라미터를 함께 송부하며 자세한 설명은 로그 및 알람 정의 부분 참조 요망.

3.4.8 struct.MOVE_POSB

로봇 제어기에서 MoveB 모션 제어 수행 시, 경유점 정보를 설정하기 구조체로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	위치 정보(#1)	float	-	6개의 Task Space 정보
24	위치 정보(#2)	float	-	6개의 Task Space 정보
48	모션 종류	uchar	0x00~0x03	1st모션과 2nd 모션의 조합 라인: 0 원호: 1

BYTE#	필드명	데이터 형	값	비고
49	곡선 곡률	float		반지름 정보(mm 단위)

- MoveB의 경우 라인모션과 원호 모션의 조합으로 이루어진다. 라인 모션의 경우 시작점을 제외한 1개의 경유 지점이, 원호 모션의 경우, 시작점을 제외한 2개의 경유 지점이 필요하다. 즉, 라인 모션인 경우 위치 정보(#2)는 무시된다.
- 참조 기준은 베이스 프레임 기준이면 Base 좌표계를, 좌표계 기준이면 Tool을 선택한다.

3.4.9 struct.ROBOT_POSE

로봇 제어기에서 get_current_pose 명령어에 대한 현재 위치정보를 나타내기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
위치 정보	위치 정보	float	6개의 위치 정보	위치 정보

3.4.10 struct.USER_COORDINATE

로봇 제어기에서 get_user_cart_coord 명령어에 대한 현재 사용자 좌표계 정보를 나타내기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	요청 ID	uchar	101~120	지정 ID: 101~120
1	참조 기준	uchar	0x00, 0x02	Base 기준 : 0 World 기준 : 2
2	좌표계 정보	float	-	6개의 Task Space 정보

3.4.11 struct.MESSAGE_POPUP

로봇 제어기에서 유형화 어플리케이션을 통해 만들어진 프로그램 실행 시, 유형화 메시지(Popup) 관련 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	메시지 정보	char	-	256 바이트 문자열

BYTE#	필드명	데이터 형	값	비고
256	메시지 레벨	uchar	-	Message : 0 Warning : 1 Alarm : 2
257	버튼 타입	uchar	-	Resume&Stop 표시 : 0 Stop 표시 : 1

3.4.12 struct.MESSAGE_INPUT

로봇 제어기에서 DRL 프로그램 실행 시 사용자 입력을 받아 처리해야 할 경우, 사용자 입력 관련 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	메시지 정보	char	-	256 바이트 문자열
256	자료형	uchar	-	int : 0 float : 1 string : 2 bool : 3

3.4.13 struct.MESSAGE_PROGRESS

로봇 제어기에서 DRL 프로그램 실행 시 현재 진행 상태에 대한 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	현재 진행 상태	uchar	-	진행 상태 정보
1	총 진행 상태	uchar	-	진행 상태 정보

3.4.14 struct.FLANGE_SERIAL_DATA

시리얼 통신 정보는 공용체로 구성되어 있으며, 시리얼 통신 설정 정보와 시리얼 통신 데이터 정보에 대한 구조체로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	명령 정보	uchar	-	0 : Open 1 : Close 2 : Send 3 : Recv
1	시리얼 통신 정보	-	-	시리얼 통신 정보

시리얼 통신 정보에 대한 공용체는 다음과 같이 정의되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	시리얼 통신 설정 정보	-	-	10Bytes 구조체
	시리얼 통신 데이터 정보	-	-	34bytes 구조체

시리얼 통신 설정 정보는 다음과 같이 정의되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	Baudrate	uchar	-	7bytes baudrate 값
7	데이터 크기	uchar	-	통신 바이트 크기
8	Parity	uchar	-	페러티 체크
9	정지 비트	uchar	-	정지 비트

시리얼 통신 데이터 정보는 다음과 같이 정의되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	데이터 길이	ushort	-	전송 데이터 길이
2	전송 데이터	uchar	-	32바이트 데이터

3.4.15 struct.FLANGE_SER_RXD_INFO

Flange Serial 사용 시 전송된 시리얼 데이터 값을 수신하기 위한 구조체로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	데이터 길이	short	-	전송 데이터 길이
2	전송 데이터	uchar	-	256바이트 데이터

3.4.16 struct.READ_FLANGE_SERIAL

Flange Serial 사용 시 수신 가능한 데이터가 버퍼에 존재하는지 확인하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	데이터 여부	uchar	-	0 : non-receive 1 : received

3.4.17 struct.INVERSE_KINEMATIC_RESPONSE

역기구학의 계산 반환값을 전달하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	로봇 자세 정보	float[6]	-	6개의 Joint Space 정보
32	가능 여부 반환	int	-	0: 조인트 값 정상 반환 1: 동작 영역을 벗어남 2: 손목 축 특이점

3.4.18 struct.UPDATE_SW_MODULE_RESPONSE

SW 모듈의 설치/삭제 명령 이후 현재 SW Module의 내역을 전달하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	수행 내역	uchar	-	0: 설치 1: 삭제
8	상세 SW Module 내역에 대한 문자열 정보	char	-	최대 2048 바이트 고정 문자열

3.4.19 struct.CONFIG_JOINT_RANGE

관절공간에서의 제한을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	일반 모드	struct.JOINT_RANGE	-	구조체 정의 참조
72	감속 모드	struct.JOINT_RANGE	-	구조체 정의 참조

3.4.20 struct.GENERAL_RANGE

힘 제한을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	최대 힘	float	-	-
4	최대 파워	float	-	-
8	최대 속도	float	-	-
12	최대 모멘텀	float	-	-

3.4.21 struct.CONFIG_GENERAL_RANGE

힘 제한을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	일반 모드	struct.GENERAL_RANGE	-	구조체 정의 참조
4	감속 모드	struct.GENERAL_RANGE	-	구조체 정의 참조

3.4.22 struct.POINT_2D

2차원 좌표를 표현하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	x 좌표 정보	float	-	x 좌표값

BYTE#	필드명	데이터 형	값	비고
4	y 좌표 정보	float	-	y 좌표값

3.4.23 struct.POINT_3D

3차원 좌표를 표현하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	x축 정보	float	-	x 좌표값
4	y축 정보	float	-	y 좌표값
8	z축 정보			z 좌표값

3.4.24 struct.LINE

두 2차원 point 간 거리를 표현하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	x축 정보	float	-	x 좌표값
4	y축 정보	float	-	y 좌표값
8	z축 정보	float		z 좌표값

3.4.25 union.CONFIG_SAFETY_FUNCTION

Safety 기능을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	정지 코드	tStopCode[17]	-	하기 구조체 정의 참조

BYTE#	필드명	데이터 형	값	비고
34	정지 기능	unsigned char[17]	-	SF05. Emergency Stop SF06. Protective Stop SF07. StadnStill Monitoring SF08. Joint Angle Monitoring SF09. Joint Speed Monitoring SF10. Joint Torque Monitoring SF11. Collision Detection SF12. TCP Position Monitoring SF13. TCP Orientation Monitoring SF14. TCP Speed Monitoring SF15. TCP Force Monitoring SF16. Momentum Monitoring SF17. Power Mon.

3.4.26 struct._tStopCode

BYTE#	필드명	데이터 형	값	비고
0	일반 작업 공간	unsigned char	4	-
34	협동 작업 공간	unsigned char	4	-

3.4.27 struct.CONFIG_INSTALL_POSE

설치 자세를 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	기울기	float	-	로봇 기울기 값
4	회전	float	-	로봇 회전 값

3.4.28 struct.CONFIG_SAFETY_IO

안전 I/O를 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	IO 포트	unsigned char[2][8]	-	컨트롤러 IO 16개 port

3.4.29 struct.CONFIG_SAFETY_IO_EX

안전 I/O를 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	IO 포트	unsigned char[2][8]	-	컨트롤러 IO 16개 port
16	트리거 레벨	unsigned char[2][8]	-	트리거 레벨

3.4.30 union.VIRTUAL_FENCE_OBJECT

작업 공간을 제한하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	사각형 작업 공간	_CUBE	-	하기 구조체 정의 참조
20	다각형 작업 공간	_POLYGON	-	하기 구조체 정의 참조
125	원기둥 작업 공간	_CYLINDER	-	하기 구조체 정의 참조
137	예약 공간	unsigned char[10]	-	예약공간 10bytes

3.4.31 struct.CONFIG_VIRTUAL_FENCE

작업 공간 제한을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	참조 기준	unsigned char	-	BASE : 0 TOOL : 1 101~200 : 사용자 좌표 계
1	펜스 타입	unsigned char	-	0: cube

BYTE#	필드명	데이터 형	값	비고
2	작업공간 정보	struct.VIRTUAL_FENCE_OBJECT	-	구조체 정의 참조

3.4.32 struct.CONFIG_SAFE_ZONE

안전 공간을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	참조 기준	unsigned char	-	BASE : 0 TOOL : 1 101~200 : 사용자 좌표계
1	선 정보	struct.LINE[2]	-	두 선 정보
33	좌표 정보	struct POINT_2D[3]	-	세 점 정보

3.4.33 struct.ENABLE_SAFE_ZONE

안전 공간을 활성화하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	안전 공간 활성화 정보	unsigned char[3]	-	활성화 정보

3.4.34 struct.SAFETY_OBJECT_SPHERE

안전 공간을 구 형태로 설정하기 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	반지름	float	-	구 반지름 정보
4	3차원 좌표	struct.POINT_3D	-	3차원 좌표 정보

3.4.35 struct.SAFETY_OBJECT_CAPSULE

안전 공간을 캡슐 형태로 설정하기 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	반지름	float	-	구 반지름 정보
4	3차원 좌표	struct.POINT_3D[2]	-	두 개의 3차원 좌표

3.4.36 struct.SAFETY_OBJECT_CUBE

안전 공간을 큐브 형태로 설정하기 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	3차원 좌표	struct.POINT_3D[3]	-	세 개의 3차원 좌표

3.4.37 struct.SAFETY_OBJECT_OBB

안전 공간을 오각형 형태로 설정하기 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	3차원 좌표	struct.POINT_3D[4]	-	네 개의 3차원 좌표

3.4.38 struct.SAFETY_OBJECT_POLYPRISM

안전 공간을 다각 기둥 형태로 설정하기 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	각의 개수	unsigned char	-	다각 기둥 각의 개수
1	2차원 좌표계	struct.POINT_2D[10]	-	10개의 2차원 좌표
81	Z축 제한	float	-	Z축 제한(하방)
85	Z축 제한	float	-	Z축 제한(상방)

3.4.39 union.SAFETY_OBJECT_DATA

안전 공간 설정을 위한 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
16	구형 오브젝트	struct.SAFETY_OBJECT_SPHERE	-	구조체 정의 참조
28	캡슐 오브젝트	struct.SAFETY_OBJECT_CAPSULE	-	구조체 정의 참조
36	큐브 오브젝트	struct.SAFETY_OBJECT_CUBE	-	구조체 정의 참조
48	오각 오브젝트	struct.SAFETY_OBJECT_OBB	-	구조체 정의 참조
89	다각 오브젝트	struct.SAFETY_OBJECT_POLYPRISM	-	구조체 정의 참조
100	예약 공간	unsigned char	-	예약 공간 100bytes

3.4.40 struct.SAFETY_OBJECT

안전 설정 오브젝트 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	참조 기준	unsigned char	-	BASE : 0 TOOL : 1 101~200 : 사용자 좌표계
1	오브젝트 종류	unsigned char	-	0 : 구 1 : 캡슐 2 : 큐브 3 : 오각 기둥 4 : 다각 기둥
2	안전 공간 오브젝트 정보	union.SAFETY_OBJECT_DATA	-	구조체 정의 참조

3.4.41 struct.CONFIG_PROTECTED_ZONE

안전 공간 설정을 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	유효성	unsigned char[10]	-	필드 활성화 여부
10	오브젝트 종류	struct.SAFETY_OBJECT	-	구조체 정의 참조

3.4.42 struct.CONFIG_COLLISION_MUTE_ZONE_PROPERTY

충돌 감시 무효 공간을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	ID	char[32]	-	필드 ID
32	활성화 여부	unsigned char	-	필드 활성화 여부
33	충돌 민감도	float	-	충돌 민감도(0~100%)
37	오브젝트 종류	struct.SAFETY_OBJECT	-	구조체 정의 참조

3.4.43 struct.CONFIG_COLLISION_MUTE_ZONE

충돌 감시 무효 공간을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	유효성	unsigned char[10]	-	필드 활성화 여부
10	충돌 감시 무효 공간 정보	struct.CONFIG_COLLISION_MUTE_ZONE_PROPERTY	-	구조체 정의 참조

3.4.44 struct.SAFETY_TOOL_ORIENTATION_LIMIT

툴 회전 제한을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	3차원 좌표계	struct.POINT_3D	-	구조체 정의 참조
12	목표 각도	float	-	목표 각도

3.4.45 struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE

툴 회전 제한 영역을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	유효성	unsigned char[10]	-	필드 활성화 여부
10	안전 설정 오브젝트	struct.SAFETY_OBJECT[10]	-	10개의 안전 설정 오브젝트
1030	툴 회전 제한	struct.SAFETY_TOOL_ORIENTATION_LIMIT[10]		10개의 툴 회전 제한

3.4.46 struct.CONFIG_NUDGE

Nudge 영역을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	활성화	unsigned char	-	활성화 여부
1	입력 힘	float	-	입력된 힘 값
4	지연 시간	float	-	총 지연시간

3.4.47 struct.CONFIG_COCKPIT_EX

콕핏 설정을 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	활성화	unsigned char	-	활성화 여부
1	버튼	unsigned char[2]	-	0 : 직접 교시 1: TCP-Z 2: 회전 3: 위치
3	복구 교시	unsigned char	-	복구 교시 활성화

3.4.48 struct.CONFIG_IDLE_OFF

자동 Servo Off 기능을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	기능 활성화	unsigned char	-	활성화 여부
1	자동 해제 시간	float	-	자동 해제 대기 시간(sec)

3.4.49 struct.WRITE_MODBUS_DATA

모드버스 TCP 데이터를 쓰기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	신호 이름	char[32]	-	32바이트 식별 이름
32	tcp 주소	char[16]	-	TCP 주소
52	포트 번호	unsigned short	-	포트 번호
56	Slave ID	int	-	1~255
60	레지스터 타입	unsgiendo char	-	모드버스 레지스터 타입
61	신호 번호	unsigned short	-	신호 번호
65	값	unsigned short	-	값(0~65535)

3.4.50 struct.WRITE_MODBUS_RTU_DATA

모드버스 RTU 데이터를 쓰기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	신호 이름	char[32]	-	32바이트 식별 이름
32	tcp 주소	char[16]	-	TCP 주소
52	포트 번호	unsigned short	-	포트 번호

BYTE#	필드명	데이터 형	값	비고
56	Slave ID	int	-	1~255
60	레지스터 타입	unsgnied char	-	모드버스 레지스터 타입
61	신호 번호	unsigned short	-	신호 번호
65	값	unsigned short	-	값(0~65535)

3.4.51 struct.MODBUS_DATA

모드버스 데이터를 저장하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	신호 타입	unsigned char	-	0: TCP 1: RTU
1	모드버스 객체	_tData	-	하기 공용체 정의 참조

3.4.52 struct.MODBUS_DATA_LIST

다수의 모드버스 데이터를 저장하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	데이터 개수	unsigned short	-	modbus 데이터 수
2	모드버스 데이터	struct.MODBUS_DATA	-	구조체 정의 참조

3.4.53 struct.CONFIG_WORLD_COORDINATE

월드 좌표계를 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	설정 타입	unsigned char	-	0: world2base 1: base2ref 2: world2ref

BYTE#	필드명	데이터 형	값	비고
2	목표 위치	float[6]	-	목표 위치

3.4.54 struct.CONFIG_CONFIGURABLE_IO

I/O입력을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O정보	unsigned char[2][16]	-	Safety I/O 정보

3.4.55 struct.CONFIG_CONFIGURABLE_IO_EX

I/O입력을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O정보	unsigned char[2][16]	-	Safety I/O 정보
2	레벨	unsigned char[2][16]	-	트리거 레벨 정보

3.4.56 struct.CONFIG_TOOL_SHAPE

툴 형상을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	유효성	unsigned char[5]	-	필드 활성화 여부
5	안전 설정 오브젝트	struct.SAFETY_OBJECT[5]	-	5개의 안전 설정 오브젝트

3.4.57 struct.CONFIG_TOOL_SYMBOL

툴 이름을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	이름	char[32]	-	32bytes 식별 이름

BYTE#	필드명	데이터 형	값	비고
32	툴 설정 오브젝트	struct.CONFIG_TOOL	-	툴 설정 오브젝트

3.4.58 struct.CONFIG_TCP_SYMBOL

TCP 이름을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	이름	char[32]	-	32bytes 식별 이름
32	TCP 설정 오브젝트	struct.CONFIG_TCP	-	TCP 설정 오브젝트

3.4.59 struct.CONFIG_TOOL_LIST

다수의 툴 이름을 설정하기 위한 구조체 정보로 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	개수	int	-	툴 개수
4	툴 이름 오브젝트	struct.CONFIG_TOOL_SYMBOL[50]	-	툴 이름 오브젝트(최대 50개)

3.4.60 struct.CONFIG_TOOL_SHAPE_SYMBOL

툴 형상 이름을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	이름	char[32]	-	32bytes 식별 이름
32	툴 형상 설정 오브젝트	struct.CONFIG_TOOL_SHAPE	-	툴 형상 설정 오브젝트

3.4.61 struct.CONFIG_TCP_LIST

다수의 TCP 이름을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	개수	int	-	툴 개수

BYTE#	필드명	데이터 형	값	비고
4	TCP 이름 오브젝트	struct.CONFIG_TCP_SYMBOL[50]	-	TCP 이름 오브젝트(최대 50개)

3.4.62 struct.CONFIG_TOOL_SHAPE_LIST

다수의 툴 형상 이름을 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	개수	int	-	툴 개수
4	툴 형상 이름 오브젝트	struct.CONFIG_TOOL_SHAPE_SYMBOL[50]	-	툴 형상 이름 오브젝트(최대 50개)

3.4.63 struct.SAFETY_CONFIGURATION_EX

현재 안전 설정 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

i 해당 구조체는 Deprecated 타입입니다. get_safety_configuration() 함수에서 더 이상 사용되지 않습니다.

BYTE#	필드명	데이터 형	값	비고
0	버전	unsigned int	-	데이터 버전
4	관절 공간 범위	struct.CONFIG_JOINT_RANGE	-	관절 공간 범위
232	작업 공간 범위	struct.CONFIG_GENERAL_RANGE	-	작업 공간 범위
280	충돌 민감도	float	-	충돌 민감도
284	안전 기능	union.CONFIG_SAFETY_FUNCTION	-	안전 기능
318	툴 이름 설정	struct.CONFIG_TOOL_SYMBOL	-	툴 이름 설정
390	TCP 이름 설정	struct.CONFIG_TCP_SYMBOL	-	TCP 이름 설정
446	설치 자세 설정	struct.CONFIG_INSTALL_POSE	-	설치 자세 설정

BYTE#	필드명	데이터 형	값	비고
454	안전 I/O 설정	struct.CONFIG_SAFETY_IO	-	안전 I/O 설정
470	작업 공간 설정	struct.CONFIG_VIRTUAL_FENCE	-	작업 공간 설정
582	안전 공간 설정	struct.CONFIG_SAFE_ZONE	-	안전 공간 설정
639	안전 공간 활성화	struct.ENABLE_SAFE_ZONE	-	안전 공간 활성화
642	보호 공간 설정	struct.CONFIG_PROTECTED_ZONE	-	보호 공간 설정
1672	충돌 감시 무효 공간 설정	struct.CONFIG_COLLISION_MUTE_ZONE	-	충돌 감시 무효 공간 설정
3072	툴 회전 제한 공간 설정	struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE	-	툴 회전 제한 공간 설정
4262	툴 형상 설정	struct.CONFIG_TOOL_SHAPE	-	툴 형상 설정
4777	Nudge 설정	struct.CONFIG_NUDGE	-	Nudge 설정
4786	콕핏 설정	struct.CONFIG_COCKPIT_EX	-	콕핏 설정
4790	자동 Servo Off 설정	struct.CONFIG_IDLE_OFF	-	자동 Servo Off 설정
4795	TCP 리스트 설정	struct.CONFIG_TCP_LIST	-	TCP 리스트 설정
7599	툴 리스트 설정	struct.CONFIG_TOOL_LIST	-	툴 리스트 설정
11203	툴 형상 리스트 설정	struct.CONFIG_TOOL_SHAPE_LIST	-	툴 형상 리스트 설정
38557	활성화 TCP 목록	char[32]	-	활성화 TCP 목록
38589	활성화 툴 목록	char[32]	-	활성화 툴 목록
38621	활성화 툴 형상 목록	char[32]	-	활성화 툴 형상 목록
38653	모드버스 리스트	struct.MODBUS_DATA_LIST	-	모드버스 리스트

BYTE#	필드명	데이터 형	값	비고
45755	월드 좌표계 설정	struct.CONFIG_WORLD_COORDINATE	-	월드 좌표계 설정
45780	속도 Cws	float	-	속도 Cws
45784	속도 Io	float	-	속도 Io
45788	IO 설정	struct.CONFIG_CONFIGURABLE_IO	-	IO 설정

3.4.64 struct.SAFETY_CONFIGURATION_EX2

현재 안전 설정 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	_iDataVersion	unsigned int	-	데이터 버전
4	_tJointRange	struct.CONFIG_JOINT_RANGE	-	관절 공간 범위
232	_tGeneralRange	struct.CONFIG_GENERAL_RANGE	-	작업 공간 범위
280	_fCollisionSensitivity	float	-	충돌 민감도
284	_tSafetyFunc	union.CONFIG_SAFETY_FUNCTION	-	안전 기능
318	_tTool	struct.CONFIG_TOOL_SYMBOL	-	툴 이름 설정
390	_tTcp	struct.CONFIG_TCP_SYMBOL	-	TCP 이름 설정
446	_tInstallPose	struct.CONFIG_INSTALL_POSE	-	설치 자세 설정
454	_tSafetyIO	struct.CONFIG_SAFETY_IO_OPTION	-	안전 I/O 설정
520	_tSafetySpaceVF	struct.CONFIG_VIRTUAL_FENCE	-	작업 공간 설정

BYTE#	필드명	데이터 형	값	비고
632	_tSafetySpaceSZ	struct.CONFIG_SAFE_ZONE	-	안전 공간 설정
689	_tSafetySpaceESZ	struct.ENABLE_SAFE_ZONE	-	안전 공간 활성화
692	_tSafetySpacePZ	struct.CONFIG_PROTECTED_ZONE	-	보호 공간 설정
1722	_tSafetySpaceCM	struct.CONFIG_COLLISION_MUTE_ZONE	-	충돌 감시 무효 공간 설정
3122	_tSafetySpaceTO	struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE	-	툴 회전 제한 공간 설정
4312	_tSafetySpaceTS	struct.CONFIG_TOOL_SHAPE	-	툴 형상 설정
4827	_tConfigNudge	struct.CONFIG_NUDGE	-	Nudge 설정
4836	_tCockPit	struct.CONFIG_COCKPIT_EX	-	콕핏 설정
4840	_tIdleOff	struct.CONFIG_IDLE_OFF	-	자동 Servo Off 설정
4845	_tConfigTCP	struct.CONFIG_TCP_LIST	-	TCP 리스트 설정
7649	_tConfigTool	struct.CONFIG_TOOL_LIST	-	툴 리스트 설정
11253	_tConfigToolShape	struct.CONFIG_TOOL_SHAPE_LIST	-	툴 형상 리스트 설정
38607	_szActiveTcp	char[32]	-	활성화 TCP 목록
38639	_szActiveTool	char[32]	-	활성화 툴 목록
38671	_szActiveToolShape	char[32]	-	활성화 툴 형상 목록
38703	_tModbusList	struct.MODBUS_DATA_LIST	-	모드버스 리스트
45805	_tWorld2BaseRelation	struct.CONFIG_WORLD_COORDINATE	-	월드 좌표계 설정
45830	m_CwsSpeedRatio	float	-	속도 Cws

BYTE#	필드명	데이터 형	값	비고
45834	속도 Io	float	-	속도 Io
45838	IO 설정	struct.CONFIG_CONFIGURABLE_IO	-	IO 설정

3.4.65 struct.SAFETY_CONFIGURATION_EX2_V3

현재 안전 설정 정보를 제공하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

통합 제어기용 인터페이스이다.

BYTE#	필드명	데이터 형	값	비고
0	_iDataVersion	unsigned int	-	데이터 버전
4	_tJointRange	struct.CONFIG_JOINT_RANGE	-	관절 공간 범위
232	_tGeneralRange	struct.CONFIG_GENERAL_RANGE	-	작업 공간 범위
280	_fCollisionSensitivity	float	-	충돌 민감도
284	_tSafetyFunc	union.CONFIG_SAFETY_FUNCTION	-	안전 기능
318	_tTool	struct.CONFIG_TOOL_SYMBOL	-	툴 이름 설정
390	_tTcp	struct.CONFIG_TCP_SYMBOL	-	TCP 이름 설정
446	_tInstallPose	struct.CONFIG_INSTALL_POSE	-	설치 자세 설정
454	_tSafetyIO	struct.CONFIG_SAFETY_IO_OP	-	안전 I/O 설정
520	_tSafetySpaceVF	struct.CONFIG_VIRTUAL_FENCE	-	작업 공간 설정
632	_tSafetySpaceSZ	struct.CONFIG_SAFE_ZONE	-	안전 공간 설정

BYTE#	필드명	데이터 형	값	비고
689	_tSafetySpaceESZ	struct.ENABLE_SAFE_ZONE	-	안전 공간 활성화
692	_tSafetySpacePZ	struct.CONFIG_PROTECTED_ZONE	-	보호 공간 설정
1722	_tSafetySpaceCM	struct.CONFIG_COLLISION_MUTE_ZONE	-	충돌 감시 무효 공간 설정
3122	_tSafetySpaceTO	struct.CONFIG_TOOL_ORIENTATION_LIMIT_ZONE	-	툴 회전 제한 공간 설정
4312	_tSafetySpaceTS	struct.CONFIG_TOOL_SHAPE	-	툴 형상 설정
4827	_tConfigNudge	struct.CONFIG_NUDGE	-	Nudge 설정
4836	_tCockPit	struct.CONFIG_COCKPIT_EX	-	콕핏 설정
4840	_tIdleOff	struct.CONFIG_IDLE_OFF	-	자동 Servo Off 설정
4845	_tConfigTCP	struct.CONFIG_TCP_LIST	-	TCP 리스트 설정
7649	_tConfigTool	struct.CONFIG_TOOL_LIST	-	툴 리스트 설정
11253	_tConfigToolShape	struct.CONFIG_TOOL_SHAPE_LIST	-	툴 형상 리스트 설정
38607	_szActiveTcp	char[32]	-	활성화 TCP 목록
38639	_szActiveTool	char[32]	-	활성화 툴 목록
38671	_szActiveToolShape	char[32]	-	활성화 툴 형상 목록
38703	_tModbusList	struct.MODBUS_DATA_LIST	-	모드버스 리스트
45805	_tWorld2BaseRelation	struct.CONFIG_WORLD_COORDINATE	-	월드 좌표계 설정
45830	m_CwsSpeedRatio	float	-	속도 Cws
45834	속도 Io	float	-	속도 Io

BYTE#	필드명	데이터 형	값	비고
45838	_iSafetyZoneCount	int	-	Zone 배열 길이
45842	_tSafetyZone	struct.CONFIG_SAFETY_ZONE	-	Safety Zone 정보
53942	_iUserCoordCount	int	-	Coord 배열 길이
53946	_tUserCoordinates	struct.CONFIG_USER_COORDINATE_EX2	-	Coord 정보
57946	_tConfigurableIO	struct.CONFIG_CONFIGURABLE_IO_EX	-	IO 설정

3.4.66 struct.ROBOT_VEL

로봇 제어기에서 현재 속도 정보를 나타내기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	속도 정보	float	6개의 속도 정보	속도 정보

3.4.67 struct.ROBOT_FORCE

로봇 제어기에서 현재 외력 정보를 나타내기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	외력 정보	float	6개의 외력 정보	외력 정보

3.4.68 struct.CONFIG_SAFETY_IO_OP

안전 I/O를 설정하기 위한 구조체 정보로 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	_iIO	unsigned char[2][16]	-	컨트롤러 IO 16개 port
32	_iTBI_Op	unsigned char	-	TBSFT 입력 옵션
33	_iReserved	unsigned char	-	예약 공간

BYTE#	필드명	데이터 형	값	비고
34	_iIO_Op	unsigned char[2][16]	-	컨트롤러 IO 16개 port 옵션

3.4.69 struct.MONITORING_CTRLIO_EX2

로봇 제어기에서 컨트롤 박스에 장착되어 있는 I/O의 현재 상태 정보를 확인하기 위한 구조체 정보는 다음과 같은 필드로 구성되어 있다.

BYTE#	필드명	데이터 형	값	비고
0	I/O 정보(#1)	-	-	I/O 신호 입력 정보
48	I/O 정보(#2)	-	-	I/O 신호 출력 정보
90	I/O 정보(#3)	-	-	Encoder 신호 데이터 정보
102	예약 공간	-	-	24바이트 예약 공간

I/O 정보(#1)은 다음과 같이 컨트롤 박스 내 Safety B'd에 부착된 I/O 입력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Digital 신호	uchar	0x00~0x01	32개의 Digital On/Off 정보
32	Analog 신호	float	-	2개의 Analog 수치 정보
40	Switch 신호	uchar	0x00~0x01	3개의 Switch On/Off 정보
43	Safety 신호	uchar	0x00~0x01	2개의 Safety On/Off 정보
46	Analog 모드	uchar	0x00~0x01	2개의 Analog 모드 정보 전류: 0 전압: 1

Switch 신호 정보 직접교시 버튼 등과 같은 컨트롤 박스 및 T/P에 부착되어 있는 3개의 스위치 상태 정보이며, Safety 신호는 컨트롤 박스에 부착되어 있는 Safety Emergency입력 신호와 Protective-Stop 신호 2개의 입력 상태 정보이다.

운용 정보(#2)는 컨트롤 박스 내 Safety B'd에 부착된 I/O 출력 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Digital 신호	uchar	0x00~0x01	32개의 Digital On/Off 정보
32	Analog 신호	float	-	2개의 Analog 수치 정보
40	Analog 모드	uchar	0x00~0x01	2개의 Analog 모드 정보 전류: 0 전압: 1

운용 정보(#3)은 다음과 같이 컨트롤 박스 내 Safety Board에 부착된 Encorder 데이터 정보로 구성된다.

BYTE#	필드명	데이터 형	값	비고
0	Strobe 신호 정보	uchar	0x00~0x01	2개의 Encorder On/Off 정보
2	RAW 데이터 정보	uint	-	2개의 Encorder 수치 정보
10	Reset 신호 정보	uchar	0x00~0x01	2개의 Encorder Reset 정보

예약 공간에는 소형 모델 관련하여 다음과 같은 입출력 정보가 제공된다.

BYTE#	필드명	데이터 형	값	비고
0	프로세스 버튼의 Digital 입력 신호	uchar	0x00~0x01	4개의 Digital On/Off 정보

3.4.70 struct.ROBOT_WELDING_DATA

이 구조체는 로봇의 용접 관련 데이터를 실시간으로 모니터링하고 제어하는 데 사용됩니다.

각 멤버 변수는 로봇 용접 시스템의 특정 상태 또는 설정 값을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_iAdjAvail	unsigned char	0 또는 1	조정 가능 상태
1	_fTargetVol	float		목표 전압
5	_fTargetCur	float		목표 전류

BYTE#	멤버 변수	데이터 타입	값	설명
9	_fTargetVel	float		목표 속도
13	_fActualVol	float		실제 전압
17	_fActualCur	float		실제 전류
21	_fOffsetY	float		위빙 Y 오프셋
25	_fOffsetZ	float		위빙 Z 오프셋
29	_iArcOnDO	unsigned char	0 또는 1	아크 상태 (0: Off, 1: On)
30	_iGasOnDO	unsigned char	0 또는 1	가스 상태 (0: Off, 1: On)
31	_iInchPDO	unsigned char	0 또는 1	인칭 플러스 상태 (0: Off, 1: On)
32	_iInchNPO	unsigned char	0 또는 1	인칭 마이너스 상태 (0: Off, 1: On)
33	_iStatus	unsigned char	0 또는 1	용접 상태 (0: 시작, 1: 종료)

3.4.71 struct.WELDING_CHANNEL

이 구조체는 용접 채널의 설정 정보를 담고 있습니다.

- _bTargetCh 는 용접에 사용할 채널 번호를 지정합니다. 0은 채널을 사용하지 않음을 의미합니다.
- _bTargetAT 는 채널의 타입을 지정합니다. 0은 전류 채널, 1은 전압 채널을 의미합니다.
- _ConstValue 는 용접 채널에 적용할 상수 값을 저장하는 배열입니다.
- _fMinValue 와 _fMaxValue 는 용접 채널의 최소값과 최대값을 지정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bTargetCh	unsigned char	0, 1 또는 2	대상 채널 번호 (0: 사용 안 함, 1: 채널 1, 2: 채널 2)
1	_bTargetAT	unsigned char	0 또는 1	채널 타입 (0: 전류, 1: 전압)

BYTE#	멤버 변수	데이터 타입	값	설명
2	_ConstValue	float[2]		상수 값 배열 (A, B)
10	_fMinValue	float		최소값
14	_fMaxValue	float		최대값

3.4.72 struct.CONFIG_WELDING_INTERFACE

이 구조체는 용접 인터페이스의 설정 정보를 담고 있습니다.

- _bEnable 은 용접 인터페이스의 활성화 여부를 나타냅니다.
- _tChOut 은 용접 출력 채널 (전압, 전류)에 대한 설정 정보를 담고 있는 WELDING_CHANNEL 구조체 배열입니다.
- _tChIn 은 용접 입력 채널 (전압, 전류)에 대한 설정 정보를 담고 있는 WELDING_CHANNEL 구조체 배열입니다.
- _iArcOnDO , _iGasOnDO , _iInchPDO , _iInchNDO 는 각각 아크 시작, 가스 시작, 인칭 플러스, 인칭 마이너스에 사용할 DO 번호를 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bEnable	unsigned char	0 또는 1	용접 인터페이스 활성화 여부 (0: 비활성화, 1: 활성화)
1	_tChOut	WELDING_CHANNEL[2]	출력 채널 설정 (전압, 전류)	
19	_tChIn	WELDING_CHANNEL[2]	입력 채널 설정 (전압, 전류)	
37	_iArcOnDO	unsigned char	0 ~ 15	아크 시작 DO 번호
38	_iGasOnDO	unsigned char	0 ~ 15	가스 시작 DO 번호
39	_iInchPDO	unsigned char	0 ~ 15	인칭 플러스 DO 번호
40	_iInchNDO	unsigned char	0 ~ 15	인칭 마이너스 DO 번호

3.4.73 struct.CONFIG_WELD_SETTING

이 구조체는 용접 설정 정보를 담고 있습니다.

- `_bVirtualMode` 는 가상 용접 모드 활성화 여부를 나타냅니다.
- `_fTargetVol`, `_fTargetCur`, `_fTargetVel` 은 각각 목표 전압, 목표 전류, 목표 속도를 나타냅니다.
- `_fTargetMinVel`, `_fTargetMaxVel` 은 각각 최소 목표 속도, 최대 목표 속도를 나타냅니다.
- `_tDetail` 은 용접 세부 설정 정보를 담고 있는 구조체입니다.
 - `_fRs`, `_fTss`, `_fTas`, `_fTwc`, `_fRf`, `_fTaf`, `_fTsf` 는 각각 시작 비율, 보호 가스 방출 시간, 시작 전류 시간, 용접 조건 변경 시간, 종료 비율, 종료 전류 시간, 종료 보호 가스 방출 시간을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bVirtualMode</code>	<code>unsigned char</code>	0 또는 1	가상 용접 모드 활성화 여부 (0: 비활성화, 1: 활성화)
1	<code>_fTargetVol</code>	<code>float</code>	목표 전압	
5	<code>_fTargetCur</code>	<code>float</code>	목표 전류	
9	<code>_fTargetVel</code>	<code>float</code>	목표 속도	
13	<code>_fTargetMinVel</code>	<code>float</code>	최소 목표 속도	
17	<code>_fTargetMaxVel</code>	<code>float</code>	최대 목표 속도	
21	<code>_tDetail._fRs</code>	<code>float</code>	시작 비율	
25	<code>_tDetail._fTs</code>	<code>float</code>	보호 가스 방출 시간	
29	<code>_tDetail._fTas</code>	<code>float</code>	시작 전류 시간	

BYTE#	멤버 변수	데이터 타입	값	설명
33	<code>_tDetail._fTw c</code>	float	용접 조건 변경 시간	
37	<code>_tDetail._fRf</code>	float	종료 비율	
41	<code>_tDetail._fTa f</code>	float	종료 전류 시간	
45	<code>_tDetail._fTs f</code>	float	종료 보호 가스 방출 시간	

3.4.74 struct.GET_WELDING_SETTING_RESPONSE

이 구조체는 용접 설정 정보 응답을 담고 있습니다.

- `_bVirtualMode` 는 가상 용접 모드 활성화 여부를 나타냅니다.
- `_fTargetVol`, `_fTargetCur`, `_fTargetVel` 은 각각 목표 전압, 목표 전류, 목표 속도를 나타냅니다.
- `_fTargetMinVel`, `_fTargetMaxVel` 은 각각 최소 목표 속도, 최대 목표 속도를 나타냅니다.
- `_tDetail` 은 용접 세부 설정 정보를 담고 있는 구조체입니다.
 - `_fRs`, `_fTss`, `_fTas`, `_fTwc`, `_fRf`, `_fTaf`, `_fTsf` 는 각각 시작 비율, 보호 가스 방출 시간, 시작 전류 시간, 용접 조건 변경 시간, 종료 비율, 종료 전류 시간, 종료 보호 가스 방출 시간을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bVirtualMode</code>	unsigned char	0 또는 1	가상 용접 모드 활성화 여부 (0: 비활성화, 1: 활성화)
1	<code>_fTargetVol</code>	float	목표 전압	
5	<code>_fTargetCur</code>	float	목표 전류	
9	<code>_fTargetVel</code>	float	목표 속도	
13	<code>_fTargetMinVel</code>	float	최소 목표 속도	

BYTE#	멤버 변수	데이터 타입	값	설명
17	_fTargetMaxVel	float	최대 목표 속도	
21	_tDetail._fRs	float	시작 비율	
25	_tDetail._fTs	float	보호 가스 방출 시간	
29	_tDetail._fTas	float	시작 전류 시간	
33	_tDetail._fTwc	float	용접 조건 변경 시간	
37	_tDetail._fRf	float	종료 비율	
41	_tDetail._fTaf	float	종료 전류 시간	
45	_tDetail._fTsf	float	종료 보호 가스 방출 시간	

3.4.75 struct.ADJUST_WELDING_SETTING

이 구조체는 용접 설정을 조정하기 위한 정보를 담고 있습니다.

- _bRealTime 은 실시간 조정 여부를 나타냅니다. 0이면 프리셋 모드, 1이면 실시간 조정 모드입니다.
- _bResetFlag 는 리셋 플래그를 나타냅니다. 0이면 현재 값을 유지하고, 1이면 설정 값으로 리셋합니다.
- _fTargetVol, _fTargetCur, _fTargetVel 은 각각 목표 전압, 목표 전류, 목표 속도를 나타냅니다.
- _fOffsetY, _fOffsetZ 는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 나타냅니다.
- _fWidthRate 는 위빙 폭 비율을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bRealTime	unsigned char	0 또는 1	실시간 조정 여부 (0: 프리셋, 1: 실시간)

BYTE#	멤버 변수	데이터 타입	값	설명
1	_bResetFlag	unsigned char	0 또는 1	리셋 플래그 (0: 현재 값 유지, 1: 설정 값으로 리셋)
2	_fTargetVol	float	목표 전압	
6	_fTargetCur	float	목표 전류	
10	_fTargetVel	float	목표 속도	
14	_fOffsetY	float	위빙 Y 오프셋	
18	_fOffsetZ	float	위빙 Z 오프셋	
22	_fWidthRate	float	위빙 폭 비율	

3.4.76 struct.CONFIG_TRAPEZOID_WEAVERS_SETTING

이 구조체는 트래피조이드 위빙 설정 정보를 담고 있습니다.

- _fOffsetY, _fOffsetZ 는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 나타냅니다.
- _fGradient 는 위빙 기울기를 나타냅니다.
- _fwPT1, _fwPT2 는 각각 위빙 wPT1, 위빙 wPT2 좌표를 나타냅니다.
- _fwT1, _fwT2 는 각각 위빙 wT1 시간, 위빙 wT2 시간을 나타냅니다.
- _fwTAcc1, _fwTAcc2 는 각각 위빙 wTAcc1 가속 시간, 위빙 wTAcc2 가속 시간을 나타냅니다.
- _fwTTD1, _fwTTD2 는 각각 위빙 wTTD1 감속 시간, 위빙 wTTD2 감속 시간을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_fOffsetY	float	위빙 Y 오프셋	
4	_fOffsetZ	float	위빙 Z 오프셋	
8	_fGradient	float	위빙 기울기	
12	_fwPT1[0]	float	위빙 wPT1 X좌표	

BYTE#	멤버 변수	데이터 타입	값	설명
16	_fwPT1[1]	float	위빙 wPT1 Y좌표	
20	_fwPT2[0]	float	위빙 wPT2 X좌표	
24	_fwPT2[1]	float	위빙 wPT2 Y좌표	
28	_fwT1	float	위빙 wT1 시간	
32	_fwT2	float	위빙 wT2 시간	
36	_fwTAcc1	float	위빙 wTAcc1 가속 시간	
40	_fwTAcc2	float	위빙 wTAcc2 가속 시간	
44	_fwTTD1	float	위빙 wTTD1 감속 시간	
48	_fwTTD2	float	위빙 wTTD2 감속 시간	

3.4.77 struct.CONFIG_ZIGZAG_WEAVERS_SETTING

이 구조체는 지그재그 위빙 설정 정보를 담고 있습니다.

- _fOffsetY, _fOffsetZ 는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 나타냅니다.
- _fGradient 는 위빙 기울기를 나타냅니다.
- _fWeavingWidth 는 위빙 폭을 나타냅니다.
- _fWeavingCycle 는 위빙 주기를 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_fOffsetY	float	위빙 Y 오프셋	
4	_fOffsetZ	float	위빙 Z 오프셋	
8	_fGradient	float	위빙 기울기	

BYTE#	멤버 변수	데이터 타입	값	설명
12	_fWeavingWidth	float	위빙 폭	
16	_fWeavingCycle	float	위빙 주기	

3.4.78 struct.CONFIG_CIRCULE_WEAVERS_SETTING

이 구조체는 원형 위빙 설정 정보를 담고 있습니다.

- _fOffsetY, _fOffsetZ 는 각각 위빙 Y 오프셋, 위빙 Z 오프셋(mm)을 나타냅니다.
- _fGradient 는 위빙 기울기(deg)를 나타냅니다.
- _fwWdt 는 위빙 폭(x,y 방향)을 나타냅니다.
- _fwT 는 위빙 주기(x,y 방향 주기)를 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_fOffsetY	float		위빙 Y 오프셋(mm)
4	_fOffsetZ	float		위빙 Z 오프셋(mm)
8	_fGradient	float		위빙 기울기(deg)
12	_fwWdt[2]	float		위빙 폭(x,y 방향)
20	_fwT[2]	float		위빙 주기(x,y 방향)

3.4.79 struct.CONFIG_SINE_WEAVERS_SETTING

이 구조체는 사인파 위빙 설정 정보를 담고 있습니다.

- _fOffsetY, _fOffsetZ 는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 나타냅니다.
- _fGradient 는 위빙 기울기를 나타냅니다.
- _fWeavingWidth 는 위빙 폭을 나타냅니다.
- _fWeavingCycle 는 위빙 주기를 나타냅니다.

소스 및 관련 콘텐츠

BYTE#	멤버 변수	데이터 타입	값	설명
0	_fOffsetY	float	위빙 Y 오프셋	
4	_fOffsetZ	float	위빙 Z 오프셋	
8	_fGradient	float	위빙 기울기	
12	_fWeavingWidth	float	위빙 폭	
16	_fWeavingCycle	float	위빙 주기	

3.4.80 struct.CONFIG_WELDING_DETAIL_INFO

이 구조체는 용접 채널의 세부 정보를 설정하기 위해 사용됩니다.

- _iChannel 멤버 변수는 설정할 채널 번호를 지정합니다. 0은 채널이 미지정되었음을 의미합니다.
- _iChannelType 멤버 변수는 채널의 타입을 지정합니다. 0은 전류 채널, 1은 전압 채널을 의미합니다.
- _iRealMinOut 멤버 변수는 실제 최소 출력값을 나타냅니다.
- _iMinOut 멤버 변수는 최소 출력값을 나타냅니다.
- _iRealMaxOut 멤버 변수는 실제 최대 출력값을 나타냅니다.
- _iMaxOut 멤버 변수는 최대 출력값을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_iChannel	unsigned char	0, 1 또는 2	채널 번호 (0: 미지정, 1: 채널 1, 2: 채널 2)
1	_iChannelType	unsigned char	0 또는 1	채널 타입 (0: 전류, 1: 전압)
2	_iRealMinOut	float	실제 최소 출력값	
6	_iMinOut	float	최소 출력값	
10	_iRealMaxOut	float	실제 최대 출력값	
14	_iMaxOut	float	최대 출력값	

3.4.81 struct.CONFIG_ANALOG_WELDING_INTERFACE

이 구조체는 아날로그 용접 인터페이스 설정 정보를 담고 있습니다.

- `_bMode` 멤버 변수는 아날로그 용접 인터페이스 모드를 설정합니다. 0은 정지, 1은 시작을 의미합니다.
- `_tTargetVoltage`, `_tFeedingSpeed`, `_tWeldingVoltage`, `_tWeldingCurrent` 멤버 변수는 각각 목표 전압, 이송 속도, 용접 전압, 용접 전류에 대한 세부 설정 정보를 담고 있는 `CONFIG_WELDING_DETAIL_INFO` 구조체입니다.
- `_iArcOnDO`, `_iGasOnDO`, `_iInchPDO`, `_iInchNDO` 멤버 변수는 각각 아크 시작, 가스 시작, 인칭 플러스, 인칭 마이너스에 사용할 DO 번호를 나타냅니다.
- `_iBlowOutValue` 멤버 변수는 블로우 아웃 값을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bMode</code>	<code>unsigned char</code>	0 또는 1	아날로그 용접 인터페이스 모드 (0: 정지, 1: 시작)
1	<code>_tTargetVoltage</code>	<code>CONFIG_WELDING_DETAIL_INFO</code>	목표 전압 설정	
17	<code>_tFeedingSpeed</code>	<code>CONFIG_WELDING_DETAIL_INFO</code>	이송 속도 설정	
33	<code>_tWeldingVoltage</code>	<code>CONFIG_WELDING_DETAIL_INFO</code>	용접 전압 설정	
49	<code>_tWeldingCurrent</code>	<code>CONFIG_WELDING_DETAIL_INFO</code>	용접 전류 설정	
65	<code>_iArcOnDO</code>	<code>unsigned char</code>	0 ~ 16	아크 시작 DO 번호
66	<code>_iGasOnDO</code>	<code>unsigned char</code>	0 ~ 16	가스 시작 DO 번호
67	<code>_iInchPDO</code>	<code>unsigned char</code>	0 ~ 16	인칭 플러스 DO 번호
68	<code>_iInchNDO</code>	<code>unsigned char</code>	0 ~ 16	인칭 마이너스 DO 번호

BYTE#	멤버 변수	데이터 타입	값	설명
69	_iBlowOutValue	unsigned char	0 ~ 16	블로우 아웃 값

3.4.82 struct.CONFIG_ANALOG_WELDING_SETTING

이 구조체는 아날로그 용접 설정 정보를 담고 있습니다.

- `_iVirtualWelding` 멤버 변수는 가상 용접 모드를 설정합니다. 0은 실제 용접, 1은 가상 용접을 의미합니다.
- `_fTargetVoltage`, `_fTargetCurrent`, `_fTargetVel` 멤버 변수는 각각 목표 전압, 목표 전류, 목표 속도를 설정합니다.
- `_fMinVel`, `_fMaxVel` 멤버 변수는 각각 최소 속도, 최대 속도를 설정합니다.
- `_tDetail` 멤버 변수는 용접 세부 설정 정보를 담고 있는 구조체입니다.
 - `_fRs`, `_fTss`, `_fTas`, `_fTwc`, `_fRf`, `_fTaf`, `_fTsf`는 각각 시작 비율, 보호 가스 방출 시간, 시작 전류 시간, 용접 조건 변경 시간, 종료 비율, 종료 전류 시간, 종료 보호 가스 방출 시간을 나타냅니다.
 - `_fStartVoltage`, `_fEndVoltage`는 각각 시작 전압 조건, 종료 전압 조건을 나타냅니다.
- `_fTargetFeedingSpeed` 멤버 변수는 목표 이송 속도를 설정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_iVirtualWelding</code>	unsigned char	0 또는 1	가상 용접 모드 (0: 실제 용접, 1: 가상 용접)
1	<code>_fTargetVoltage</code>	float	목표 전압 (V)	
5	<code>_fTargetCurrent</code>	float	목표 전류 (A)	
9	<code>_fTargetVel</code>	float	목표 속도 (mm/sec)	
13	<code>_fMinVel</code>	float	최소 속도 (mm/sec)	
17	<code>_fMaxVel</code>	float	최대 속도 (mm/sec)	
21	<code>_tDetail._fRs</code>	float	시작 비율	

BYTE#	멤버 변수	데이터 타입	값	설명
25	<code>_tDetail._fTss</code>	<code>float</code>	보호 가스 방출 시간 (초)	
29	<code>_tDetail._fTas</code>	<code>float</code>	시작 전류 시간 (초)	
33	<code>_tDetail._fTwc</code>	<code>float</code>	용접 조건 변경 시간 (초)	
37	<code>_tDetail._fRf</code>	<code>float</code>	종료 비율	
41	<code>_tDetail._fTaf</code>	<code>float</code>	종료 전류 시간 (초)	
45	<code>_tDetail._fTsf</code>	<code>float</code>	종료 보호 가스 방출 시간 (초)	
49	<code>_tDetail._fStar</code> <code>tVoltage</code>	<code>float</code>	시작 전압 조건 (V)	
53	<code>_tDetail._fEndV</code> <code>oltage</code>	<code>float</code>	종료 전압 조건 (V)	
57	<code>_fTargetFeeding</code> <code>Speed</code>	<code>float</code>	목표 이송 속도	

3.4.83 struct.ANALOG_WELDING_ADJUST_SETTING

이 구조체는 아날로그 용접 설정을 조정하기 위한 정보를 담고 있습니다.

- `_bRealTime` 멤버 변수는 실시간 조정 여부를 나타냅니다. 0이면 프리셋 모드, 1이면 실시간 조정 모드입니다.
- `_bResetFlag` 멤버 변수는 리셋 플래그를 나타냅니다. 0이면 현재 값을 유지하고, 1이면 설정 값으로 리셋합니다.
- `_fTargetVol` 멤버 변수는 목표 전압을 설정합니다.
- `_fFeedingVel` 멤버 변수는 이송 속도를 설정합니다.
- `_fTargetVel` 멤버 변수는 목표 속도를 설정합니다.
- `_fOffsetY`, `_fOffsetZ` 멤버 변수는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 설정합니다.
- `_fWidthRate` 멤버 변수는 위빙 폭 비율을 설정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bRealTime	unsigned char	0 또는 1	실시간 조정 여부 (0: 프리셋, 1: 실시간)
1	_bResetFlag	unsigned char	0 또는 1	리셋 플래그 (0: 현재 값 유지, 1: 설정 값으로 리셋)
2	_fTargetVol	float	목표 전압 (V)	
6	_fFeedingVel	float	이송 속도	
10	_fTargetVel	float	목표 속도 (mm/sec)	
14	_fOffsetY	float	위빙 Y 오프셋	
18	_fOffsetZ	float	위빙 Z 오프셋	
22	_fWidthRate	float	위빙 폭 비율	

3.4.84 struct.CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA

이 구조체는 디지털 용접 인터페이스의 매핑 데이터를 설정하는 데 사용됩니다.

- _bEnable 멤버 변수는 해당 매핑 데이터의 사용 여부를 나타냅니다. 0은 사용 안 함, 1은 사용을 의미합니다.
- _nDataType 멤버 변수는 데이터 타입을 지정합니다. 0은 On/Off 신호, 1은 값을 의미합니다.
- _nPositionalNumber 멤버 변수는 데이터의 위치 번호를 지정합니다. 0은 1, 1은 0.1, 2는 0.001을 의미합니다.
- _fMinData, _fMaxData 멤버 변수는 각각 최소 데이터, 최대 데이터를 나타냅니다.
- _nByteOffset, _nBitOffset 멤버 변수는 각각 바이트 오프셋, 비트 오프셋을 나타냅니다.
- _nCommDataType 멤버 변수는 통신 데이터 타입을 지정합니다.
- _nMaxDigitSize 멤버 변수는 최대 디지털 값을 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bEnable	unsigned char	0 또는 1	사용 여부 (0: 사용 안 함, 1: 사용)

BYTE#	멤버 변수	데이터 타입	값	설명
1	_nDataType	unsigned char	0 또는 1	데이터 타입 (0: On/Off, 1: 값)
2	_nPositionalNumber	unsigned char	0, 1 또는 2	위치 번호 (0: 1, 1: 0.1, 2: 0.001)
3	_fMinData	float	최소 데이터	
7	_fMaxData	float	최대 데이터	
11	_nByteOffset	unsigned char	바이트 오프셋	
12	_nBitOffset	unsigned char	비트 오프셋	
13	_nComnDataType	unsigned char	0 ~ 7	통신 데이터 타입 (0: 1비트 (Disable Low), 1: 1비트 (Disable High), 2: 2비트, 3: 4비트, 4: 8비트, 5: 15바이트, 6: 16비트, 7: 32비트)
14	_nMaxDigitSize	unsigned char	최대 디지털 값	

3.4.85 struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS

이 구조체는 디지털 용접 인터페이스의 프로세스 관련 설정 정보를 담고 있습니다.

- _tWeldingStart, _tRobotReady, _tErrorReset 멤버 변수는 각각 용접 시작, 로봇 준비 상태, 오류 리셋에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tWeldingStart	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	용접 시작 설정	

BYTE#	멤버 변수	데이터 타입	값	설명
15	_tRobotReady	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	로봇 준비 상태 설정	
30	_tErrorReset	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	오류 리셋 설정	

3.4.86 struct.CONFIG_DIGITAL_WELDING_INTERFACE_MODE

이 구조체는 디지털 용접 인터페이스의 모드 관련 설정 정보를 담고 있습니다.

- _tWeldingMode, _t2T2TSpecial, _tPulseMode, _tWOpt1 멤버 변수는 각각 용접 모드, 2T2T 특수 모드, 펄스 모드, WM 옵션 1에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tWeldingMode	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	용접 모드 설정	
15	_t2T2TSpecial	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	2T2T 특수 모드 설정	
30	_tPulseMode	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	펄스 모드 설정	
45	_tWOpt1	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	WM 옵션 1 설정	

3.4.87 struct.CONFIG_DIGITAL_WELDING_INTERFACE_TEST

이 구조체는 디지털 용접 인터페이스의 테스트 관련 설정 정보를 담고 있습니다.

- _tGasTest, _tInchingP, _tInchingM, _tBlowOutTorch, _tSimulation, _tTSopt1, _tTSopt2 멤버 변수는 각각 가스 테스트, 인칭 P, 인칭 M, 블로우 아웃 토치, 시뮬레이션, TS 옵션 1, TS 옵션 2에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tGasTest	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	가스 테스트 설정	
15	_tInchingP	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	인칭 P 설정	
30	_tInchingM	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	인칭 M 설정	
45	_tBlowOutTorch	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	블로우 아웃 토치 설정	
60	_tSimulation	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	시뮬레이션 설정	
75	_tTSopt1	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	TS 옵션 1 설정	
90	_tTSopt2	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	TS 옵션 2 설정	

3.4.88 struct.CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION

이 구조체는 디지털 용접 인터페이스의 조건 관련 설정 정보를 담고 있습니다.

- _tJobNumber, _tSynergicID, _tWireFeedSpeed, _tArcLengthCorrection, _tDynamicCorrection 멤버 변수는 각각 작업 번호, 시너직 ID, 와이어 이송 속도, 아크 길이 보정, 동적 보정에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tJobNumber	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	작업 번호 설정	

BYTE#	멤버 변수	데이터 타입	값	설명
15	_tSynergicID	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	시너직 ID 설정	
30	_tWireFeedSpeed	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	와이어 이송 속도 설정	
45	_tArcLengthCorrection	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	아크 길이 보정 설정	
60	_tDynamicCorrection	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	동적 보정 설정	

3.4.89 struct.CONFIG_DIGITAL_WELDING_INTERFACE_OPTION

이 구조체는 디지털 용접 인터페이스의 옵션 관련 설정 정보를 담고 있습니다.

- _tArcOnDelay, _tArcOffDelay, _tCraterOnDelay, _tCraterOffDelay 멤버 변수는 각각 아크 시작 지연, 아크 종료 지연, 크레이터 시작 지연, 크레이터 종료 지연에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tArcOnDelay	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	아크 시작 지연 설정	
15	_tArcOffDelay	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	아크 종료 지연 설정	
30	_tCraterOnDelay	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	크레이터 시작 지연 설정	
45	_tCraterOffDelay	CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA	크레이터 종료 지연 설정	

3.4.90 struct.CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2

이 구조체는 디지털 용접 인터페이스의 프로세스2 관련 설정 정보를 담고 있습니다.

- `_tCurrentFlow`, `_tProcessActive`, `_tMainCurrent`, `_tMachineReady`, `_tCommReady` 멤버 변수는 각각 전류 흐름, 프로세스 활성화, 메인 전류, 장비 준비 상태, 통신 준비 상태에 대한 설정 정보를 담고 있는 `CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA` 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_tCurrentFlow</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	전류 흐름 설정	
15	<code>_tProcessActive</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	프로세스 활성화 설정	
30	<code>_tMainCurrent</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	메인 전류 설정	
45	<code>_tMachineReady</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	장비 준비 상태 설정	
60	<code>_tCommReady</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	통신 준비 상태 설정	

3.4.91 struct.CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING

이 구조체는 디지털 용접 인터페이스의 모니터링 관련 설정 정보를 담고 있습니다.

- `_tActualWeldingCurrent`, `_tActualWeldingVoltage`, `_tActualWireFeederSpeed`, `_tActualArcVoltage` 멤버 변수는 각각 실제 용접 전류, 실제 용접 전압, 실제 와이어 이송 속도, 실제 아크 전압에 대한 설정 정보를 담고 있는 `CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA` 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_tActualWeldingCurrent</code>	<code>CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA</code>	실제 용접 전류 설정	

BYTE#	멤버 변수	데이터 타입	값	설명
15	_tActualWeldingVoltage	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	실제 용접 전압 설정	
30	_tActualWireFeederSpeed	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	실제 와이어 이송 속도 설정	
45	_tActualArcVoltage	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	실제 아크 전압 설정	

3.4.92 struct.CONFIG_DIGITAL_WELDING_INTERFACE_OTHER

이 구조체는 디지털 용접 인터페이스의 기타 설정 정보를 담고 있습니다.

- _tArcOnSignal, _tArcError, _tWireError, _tMTmode, _tOTmode 멤버 변수는 각각 아크 시작 신호, 아크 에러, 와이어 에러, MT 모드, OT 모드에 대한 설정 정보를 담고 있는 CONFIG_DIGITAL_WELDING_IF_MAPPING_DATA 구조체입니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_tArcOnSignal	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	아크 시작 신호 설정	
15	_tArcError	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	아크 에러 설정	
30	_tWireError	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	와이어 에러 설정	
45	_tMTmode	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	MT 모드 설정	
60	_tOTmode	CONFIG_DIGITAL_WELDING_I F_MAPPING_DATA	OT 모드 설정	

3.4.93 struct.DIGITAL_WELDING_RESET

이 구조체는 디지털 용접 인터페이스를 리셋하기 위한 정보를 담고 있습니다.

- `_bReset` 멤버 변수는 리셋 신호를 나타냅니다. 0은 리셋하지 않음, 1은 리셋을 의미합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bReset</code>	<code>unsigned char</code>	0 또는 1	리셋 신호 (0: 리셋 안 함, 1: 리셋)

3.4.94 struct.CONFIG_DIGITAL_WELDING_MODE

이 구조체는 디지털 용접 모드 설정 정보를 담고 있습니다.

- `_bMode` 멤버 변수는 디지털 용접 모드를 설정합니다. 0은 정지, 1은 시작을 의미합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bMode</code>	<code>unsigned char</code>	0 또는 1	디지털 용접 모드 (0: 정지, 1: 시작)

3.4.95 struct.CONFIG_DIGITAL_WELDING_CONDITION

이 구조체는 디지털 용접 조건 설정 정보를 담고 있습니다.

- `_cVirtualWelding` 멤버 변수는 가상 용접 모드를 설정합니다. 0은 실제 용접, 1은 가상 용접을 의미합니다.
- `_fTargetVel` 멤버 변수는 목표 속도를 설정합니다.
- `_fMinVelLimit`, `_fMaxVelLimit` 멤버 변수는 각각 최소 속도 제한, 최대 속도 제한을 설정합니다.
- `_nWeldingMode`, `_n2t2tSpecial`, `_nPulseMode`, `_nWMopt1` 멤버 변수는 각각 용접 모드, 2T2T 특수 모드, 필스 모드, WM 옵션 1을 설정합니다.
- `_cSimulation` 멤버 변수는 시뮬레이션 모드를 설정합니다. 0은 비활성화, 1은 활성화를 의미합니다.
- `_cTSopt1`, `_cTSopt2` 멤버 변수는 TS 옵션 1, TS 옵션 2를 설정합니다.
- `_nJobNumber`, `_nSynergicID` 멤버 변수는 각각 작업 번호, 시너직 ID를 설정합니다.
- `_fWireFeedSpeed`, `_fArcLengthCorrection`, `_fDynamicCorrection` 멤버 변수는 각각 와이어 이송 속도, 아크 길이 보정, 동적 보정을 설정합니다.
- `_fOption1`부터 `_fOption15` 까지의 멤버 변수는 추가적인 옵션을 설정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_cVirtualWelding</code>	<code>unsigned char</code>	0 또는 1	가상 용접 모드 (0: 실제 용접, 1: 가상 용접)

BYTE#	멤버 변수	데이터 타입	값	설명
1	_fTargetVel	float	목표 속도 (mm/sec)	
5	_fMinVelLimit	float	최소 속도 제한 (mm/sec)	
9	_fMaxVelLimit	float	최대 속도 제한 (mm/sec)	
13	_nWeldingMode	unsigned int	용접 모드	
17	_n2t2tSpecial	unsigned int	2T2T 특수 모드	
21	_nPulseMode	unsigned int	펄스 모드	
25	_nWModt1	unsigned int	WM 옵션 1	
29	_cSimulation	unsigned char	0 또는 1	시뮬레이션 모드 (0: 비활성화, 1: 활성화)
30	_cTSopt1	unsigned char	TS 옵션 1	
31	_cTSopt2	unsigned char	TS 옵션 2	
32	_nJobNumber	unsigned int	작업 번호	
36	_nSynergicID	unsigned int	시너직 ID	
40	_fWireFeedSpeed	float	와이어 이송 속도	
44	_fArcLengthCorrection	float	아크 길이 보정	

BYTE#	멤버 변수	데이터 타입	값	설명
48	_fDynamicCorrection	float	동적 보정	
52	_fOption1	float	옵션 1	
56	_fOption2	float	옵션 2	
60	_fOption3	float	옵션 3	
64	_fOption4	float	옵션 4	
68	_fOption5	float	옵션 5	
72	_fOption6	float	옵션 6	
76	_fOption7	float	옵션 7	
80	_fOption8	float	옵션 8	
84	_fOption9	float	옵션 9	
88	_fOption10	float	옵션 10	
92	_fOption11	float	옵션 11	
96	_fOption12	float	옵션 12	
100	_fOption13	float	옵션 13	
104	_fOption14	float	옵션 14	
108	_fOption15	float	옵션 15	

3.4.96 struct.CONFIG_DIGITAL_WELDING_ADJUST

이 구조체는 디지털 용접 조건을 조정하기 위한 정보를 담고 있습니다.

- `_bRealTime` 멤버 변수는 실시간 조정 여부를 나타냅니다. 0이면 프리셋 모드, 1이면 실시간 조정 모드입니다.
- `_bResetFlag` 멤버 변수는 리셋 플래그를 나타냅니다. 0이면 현재 값을 유지하고, 1이면 설정 값으로 리셋합니다.
- `_fTargetVel` 멤버 변수는 목표 속도를 설정합니다.
- `_fOffsetY`, `_fOffsetZ` 멤버 변수는 각각 위빙 Y 오프셋, 위빙 Z 오프셋을 설정합니다.
- `_fWidthRate` 멤버 변수는 위빙 폭 비율을 설정합니다.
- `_fDynamicCor`, `_fVoltageCor` 멤버 변수는 각각 동적 보정, 전압 보정을 설정합니다.
- `_nJobNumber`, `_nSynergicID` 멤버 변수는 각각 작업 번호, 시너직 ID를 설정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bRealTime</code>	<code>unsigned char</code>	0 또는 1	실시간 조정 여부 (0: 프리셋, 1: 실시간)
1	<code>_bResetFlag</code>	<code>unsigned char</code>	0 또는 1	리셋 플래그 (0: 현재 값 유지, 1: 설정 값으로 리셋)
2	<code>_fTargetVel</code>	<code>float</code>	목표 속도 (mm/sec)	
6	<code>_fOffsetY</code>	<code>float</code>	위빙 Y 오프셋	
10	<code>_fOffsetZ</code>	<code>float</code>	위빙 Z 오프셋	
14	<code>_fWidthRate</code>	<code>float</code>	위빙 폭 비율	
18	<code>_fDynamicCor</code>	<code>float</code>	동적 보정	
22	<code>_fVoltageCor</code>	<code>float</code>	전압 보정	
26	<code>_nJobNumber</code>	<code>unsigned int</code>	작업 번호	
30	<code>_nSynergicID</code>	<code>unsigned int</code>	시너직 ID	

3.4.97 struct.MEASURE_TCP_WELDING

이 구조체는 용접 TCP 측정에 필요한 정보를 담고 있습니다.

- `_iMode` 멤버 변수는 측정 모드를 설정합니다. 0은 측정하지 않음, 1은 측정을 의미합니다.
- `_fStickout` 멤버 변수는 스틱 아웃 값을 mm 단위로 설정합니다.
- `_fTargetPos` 멤버 변수는 9개 조인트에 대한 목표 위치를 X, Y, Z 좌표로 설정합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_iMode</code>	<code>unsigned char</code>	0 또는 1	측정 모드 (0: 측정 안 함, 1: 측정)
1	<code>_fStickout</code>	<code>float</code>	스틱 아웃 값 (mm)	
5		<code>_fTargetPos</code>	<code>float[9]</code> [NUMBER_OF_JOINT]	9개 조인트에 대한 목표 위치 (X, Y, Z 좌표)

3.4.98 struct.TACK_WELDING_SETTING

이 구조체는 택 용접 설정 정보를 담고 있습니다.

- `_bEnable` 멤버 변수는 택 용접 활성화 여부를 나타냅니다. 0은 비활성화, 1은 활성화를 의미합니다.
- `_bWeldingType` 멤버 변수는 용접 타입을 나타냅니다. 0은 아날로그 용접, 1은 디지털 용접을 의미합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	<code>_bEnable</code>	<code>unsigned char</code>	0 또는 1	택 용접 활성화 여부 (0: 비활성화, 1: 활성화)
1	<code>_bWeldingType</code>	<code>unsigned char</code>	0 또는 1	용접 타입 (0: 아날로그, 1: 디지털)

3.4.99 struct.DIGITAL_FORCE_WRITE_DATA

이 구조체는 디지털 용접 신호를 강제로 출력하기 위한 데이터를 담고 있습니다.

- `_bForceWrite` 멤버 변수는 강제 출력 사용 여부를 나타냅니다. 0은 사용 안 함, 1은 사용을 의미합니다.
- `_iForceData` 멤버 변수는 강제 출력할 데이터 값을 나타냅니다.
- `_iTargetOutput` 멤버 변수는 강제 출력할 대상을 지정합니다. 1부터 4까지는 디지털 출력 (DO), 5부터 6까지는 아날로그 출력 (AO), 7은 시너직 신호, 8은 오류 신호를 나타냅니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_bForceWrite	unsigned char	0 또는 1	강제 출력 여부 (0: 사용 안 함, 1: 사용)
1	_iForceData	unsigned char	0 또는 1	강제 출력 데이터
2	_iTargetOutput	unsigned char	1 ~ 8	출력 대상 (1: DO1, 2: DO2, 3: DO3, 4: DO4, 5: AO1, 6: AO2, 7: 시너직 신호, 8: 오류 신호)

3.4.100 struct.ROBOT_DIGITAL_WELDING_DATA

이 구조체는 로봇의 디지털 용접 관련 데이터를 실시간으로 모니터링하고 제어하는 데 사용됩니다.

- 각 멤버 변수는 로봇 디지털 용접 시스템의 특정 상태 또는 설정 값을 나타냅니다.
- _iAdjAvail 값이 1인 경우, 용접 조건을 조정할 수 있습니다.
- _iWeldingStatus 값은 용접 작업의 시작과 종료를 나타냅니다.

소스 및 관련 콘텐츠

BYTE#	멤버 변수	데이터 타입	값	설명
0	_iAdjAvail	unsigned char	0 또는 1	조정 가능 상태
1	_fTargetVol	float	목표 전압 (V)	
5	_fTargetCur	float	목표 전류 (A)	
9	_fTargetVel	float	목표 속도 (mm/sec)	
13	_fActualVol	float	실제 전압 (V)	
17	_fActualCur	float	실제 전류 (A)	
21	_fTargetVoltOut	float	목표 전압 출력 (V)	

BYTE#	멤버 변수	데이터 타입	값	설명
25	_fTargetCurOut	float	목표 전류 출력 (A)	
29	_fWeavingOffset	float	위빙 오프셋	
33	_iArcOn	unsigned char	0 또는 1	아크 상태 (0: Off, 1: On)
34	_iGasOn	unsigned char	0 또는 1	가스 상태 (0: Off, 1: On)
35	_iInchP	unsigned char	0 또는 1	인칭 플러스 상태 (0: Off, 1: On)
36	_iInchN	unsigned char	0 또는 1	인칭 마이너스 상태 (0: Off, 1: On)
37	_iWeldingStatus	unsigned char	0 또는 1	용접 상태 (0: 시작, 1: 종료)
38	_fActualFeeding Speed	float	실제 이송 속도	
42	_iErrorNumber	int	오류 번호	
46	_fWireStick	float	와이어 스틱	
50	_iError	int	오류	
54	_fOption1	float	옵션 1	
58	_fOption2	float	옵션 2	
62	_fOption3	float	옵션 3	

BYTE#	멤버 변수	데이터 타입	값	설명
66	_fOption4	float	옵션 4	
70	_fOption5	float	옵션 5	
74	_fOption6	float	옵션 6	
78	_fOption7	float	옵션 7	
82	_fOption8	float	옵션 8	
86	_fOption9	float	옵션 9	
90	_fOption10	float	옵션 10	
94	_iCurrentFlow	unsigned char	0 또는 1	전류 흐름 상태 (0: Off, 1: On)
95	_iProcessActive	unsigned char	0 또는 1	프로세스 활성화 상태 (0: 비활성화, 1: 활성화)
96	_iMachineryRead y	unsigned char	0 또는 1	장비 준비 상태 (0: 미준비, 1: 준비)
97	_fVoltageCorrection	float	전압 보정	
101	_fDynamicCorrection	float	동적 보정	

3.4.101 struct.DIGITAL_WELDING_COMM_STATE

이 구조체는 디지털 용접 통신 상태 정보를 담고 있습니다.

- _iCommState 멤버 변수는 디지털 용접 통신 상태를 나타냅니다. 0은 연결 끊김, 1은 연결됨을 의미합니다.

BYTE#	멤버 변수	데이터 타입	값	설명
0	_iCommState	unsigned char	0 또는 1	통신 상태 (0: 연결 끊김, 1: 연결됨)

4 함수(Function)

4.1 로봇 연결 함수

4.1.1 CDRFLEEx.open_connection

기능

로봇 제어기와 TCP/IP 통신을 사용하여 연결을 수립하기 위한 함수이다. TCP/IP 포트는 내부 고정임으로 별도로 지정할 필요는 없으며, 2개의 이상의 로봇 제어기를 사용할 경우, T/P 어플리케이션에서 IP 주소를 변경해야 한다.

인수

인수명	자료형	기본값	설명
strIpAddr	string	“192.168.137.100”	제어기 IP

리턴

값	설명
0	오류
1	성공

예제

```

1 CDRFLEEx drfl;
2 bool bConnected = drfl.open_connection("192.168.137.100");
3 if (bConnected) {
4     SYSTEM_VERSION tSysVerion = {'\0', };
5     Drfl.get_system_version(&tSysVerion)
6     cout << "System version: " << tSysVerion._szController << endl;
7 }
```

4.1.2 CDRFLEEx.close_connection

기능

로봇제어기와 통신 연결을 해제하기 위한 함수이다.

인수

없음

리턴

값	설명
1	성공

예제

1	<code>Drfl.close_connection();</code>
---	---------------------------------------

4.2 로봇 속성 함수

4.2.1 CDRFLEx.get_system_version

기능

로봇 제어기를 구성하고 있는 각각의 서브 시스템에 대한 버전 정보를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
*pVersion	struct.SYSTEM_VERSIO N	-	구조체 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 SYSTEM_VERSION tSysVerion = {'\0', };
2 Drfl.get_system_version(&tSysVerion);
3 cout << "version: " << tSysVerion._szController << endl;

```

4.2.2 CDRFLEEx.get_library_version**기능**

본 API의 버전 정보를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
문자열(최대 32바이트)	버전 정보(예, GL:010105)

예제

```

1 char *lpszLibVersion = get_library_version();
2 cout << "LibVersion: " << lpszLibVersion << endl;

```

4.2.3 CDRFLEEx.get_robot_mode**기능**

로봇 제어기의 현재 운용 모드를 확인하기 위한 함수이다. 자동모드는 일련의 순서로 구성된 동작(프로그램)을 자동으로 수행하기 위한 모드이며, 수동모드는 조그와 같은 단일 동작을 수행하기 위한 모드이다.

인수

없음

리턴

값	설명
enum.ROBOT_MODE	상수 및 열거형 정의 참조

예제

```

1 string strDrlProgram = "\r\n\
2 loop = 0\r\n\
3 while loop < 3:\r\n\
4     movej(posj(10,10.10,10,10.10), vel=60, acc=60)\r\n\
5     movej(posj(00,00.00,00,00.00), vel=60, acc=60)\r\n\
6     loop+=1\r\n\
7     movej(posj(10,10.10,10,10.10), vel=60, acc=60)\r\n";
8
9 if (drfl.get_robot_state() == eSTATE_STANDBY) {
10
11     if (drfl.get_robot_mode() == ROBOT_MODE_MANUAL) {
12         // 수동 모드
13         drfl.jog(JOG_AXIS_JOINT_3, MOVE_REFERENCE_BASE, 60.f);
14         sleep(2);
15         drfl.jog(JOG_AXIS_JOINT_3, MOVE_REFERENCE_BASE, 0.f);
16     }
17     else {
18         // 자동모드
19         ROBOT_SYSTEM eTargetSystem = ROBOT_SYSTEM_VIRTUAL;
20         Drfl.drl_start(eTargetSystem, strDrlProgram)
21     }
22 }
```

4.2.4 CDRFLE.set_robot_mode

기능

로봇 제어기의 현재 운용 모드 정보를 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eMode	enum.ROBOT_MODE	-	상수 및 열거형 정의 참조

리턴

값	설명
0	성공
1	오류

예제

```

1 if (Drfl.get_robot_mode() == ROBOT_MODE_MANUAL) {
2     // 자동모드 전환
3     Drfl.set_robot_mode(ROBOT_MODE_AUTONOMOUS);
4 }
```

4.2.5 CDRFLEEx.get_robot_state

기능

TOnMonitoringStateCB 콜백 함수와 더불어 로봇 제어기의 현재 운용 상태 정보를 확인하기 위한 함수로, 안전상 운용 상태에 따라 사용자가 set_robot_control 함수를 이용하여 운용 상태를 전환해야 한다.

인수

없음

리턴

값	설명
enum.ROBOT_STATE	상수 및 열거형 정의 참조

예제

```

1 if (Drfl.get_robot_state() == STATE_STANDBY) {
2     if (Drfl.get_robot_mode() == ROBOT_MODE_MANUAL) {
3         // 수동모드
4         Drfl.jog (JOG_AXIS_JOINT_3, MOVE_REFERENCE_BASE, 60.f);
5         sleep(2);
6         Drfl.jog(JOG_AXIS_JOINT_3, MOVE_REFERENCE_BASE, 0.f);
7     }
}
```

8 }

4.2.6 CDRFLEx.set_robot_control

기능

로봇 제어기에서 현재 운용 상태를 사용자가 직접 설정 및 전환할 수 있는 함수이다.

인수

인수명	자료형	기본값	설명
eControl	enum.ROBOT CONTR OL	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1  if (Drfl.get_robot_state () == STATE_SAFE_OFF) {
2      // 서보온
3      Drfl.set_robot_control(CONTROL_RESET_SAFE_OFF);
4  }
5  else if (Drfl.get_robot_state() == STATE_SAFE_OFF2) {
6      // 복구 모드 진입
7      Drfl.set_robot_control(CONTROL_RECOVERY_SAFE_OFF);
8  }

```

4.2.7 CDRFLEx.set_robot_system

기능

로봇 제어기에서 현재 운용 로봇 시스템을 설정 및 변경하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eRobotSystem	enum.ROBOT_SYSTEM	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 ROBOT_SYSTEM eRobotSystem = Drfl.GetRobotSystem();
2 if(eRobotSystem != ROBOT_SYSTEM_REAL) {
3     //자동모드 전환
4     Drfl.set_robot_system(ROBOT_SYSTEM_REAL);
5 }
6 else {
7     //do somting ...
8 }
```

4.2.8 CDRFLEEx.get_robot_speed_mode

기능

TOnMonitoringSpeedModeCB 콜백 함수와 더불어 로봇 제어기에서 현재 속도 모드(정상 모드, 감속 모드) 정보를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
enum.SPEED_MODE	상수형 및 열거형 정의 참조

예제

```

1  if (Drfl.get_robot_speed_mode() == SPEED_REDUCED_MODE){
2      //감속모드 이면 정속 속도 모드로 변경
3      Drfl.set_robot_speed_mode(SPEED_NORMAL_MODE);
4 }
```

4.2.9 CDRFLEEx.set_robot_speed_mode**기능**

로보 제어기에서 현재 운용 중인 속도 모드를 설정 및 변경하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eSpeedMode	enum.SPEED_MODE	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1  if (Drfl.get_robot_speed_mode() == SPEED_REDUCED_MODE){
2      //감속모드 이면 정속 속도 모드로 변경
3      Drfl.set_robot_speed_mode(SPEED_NORMAL_MODE);
4 }
```

4.2.10 CDRFLEEx.get_program_state**기능**

로봇 제어기에서 현재 자동모드로 실행중인 프로그램 실행 상태 정보를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
enum.DRL_PROGRAM_STATE	상수 및 열거형 정의 참조

예제

```

1 if (Drfl.get_program_state() == DRL_PROGRAM_STATE_PLAY){
2     // 프로그램 실행 중지
3     Drfl.drl_stop(STOP_TYPE_SLOW)
4 }
5 else if (Drfl.get_program_state() == DRL_PROGRAM_STATE_HOLD) {
6     // 프로그램 실행 재개
7     Drfl.drl_resume()
8 }
```

4.2.11 CDRFLEx.get_robot_system

기능

로봇 제어기에서 현재 운용 로봇 시스템(가상 로봇, 실제 로봇) 정보를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
enum.ROBOT_SYSTEM	상수 및 열거형 정의 참조

예제

```

1 // 현재 로봇 시스템 확인
2 ROBOT_SYSTEM eRobotSystem = Drfl.get_robot_system();
```

```

3   if(eRobotSystem != ROBOT_SYSTEM_REAL) {
4       Drfl.set_robot_system(ROBOT_SYSTEM_REAL);
5   }
6   else {
7       //do somting ...
8   }
9 }
```

4.2.12 CDRFLEx.set_safe_stop_reset_type

기능

로봇 제어기의 운용 상태 정보가 SAFE_STOP 일 경우, set_robot_control 함수를 이용하여 상태 전환 후 이후 자동으로 실행되는 일련의 동작을 정의하기 위한 함수이다. 로봇 운용 모드가 자동일 경우, 프로그램의 재실행 여부를 정의 및 설정할 수 있으며, 수동모드일 경우에는 이 설정은 무시된다.

인수

인수명	자료형	기본값	설명
eResetType	enum.SAFE_STOP_RESET_TYPE	SAFE_STOP_RESET_TYPE_DEFAULT	상수 및 열거형 정의 참조

리턴

값	설명
1	오류

예제

```

1 void OnMonitoringStateCB(const ROBOT_STATE eState)
2 {
3     switch((unsigned char)eState)
4     {
5         case eSTATE_SAFE_STOP:
6             if (Drfl.get_robot_mode(ROBOT_MODE_AUTONOMOUS) {
7                 // 자동모드 이면 상태 전환 후 프로그램 재실행
8                 Drfl.set_safe_stop_reset_type(SAFE_STOP_PROGRAM_RESUME);
9                 Drfl.set_robot_control(eCONTROL_RESET_SAFET_STOP);
10            }
11            else {
12                // 수동모드이면 상태만 STATE_STANDBY로 전환
13                Drfl.set_robot_control(eCONTROL_RESET_SAFET_STOP);
14            }
15            break;
}
```

16

//...

4.2.13 CDRFLEx.get_current_pose

기능

로봇 제어기에서 좌표계(관절 공간 또는 작업공간)에 따른 로봇의 각 축별 현재 위치 정보를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eSpaceType	enum.ROBOT_SPACE	ROBOT_SPACE_JOINT	상수 및 열거형 정의 참조

리턴

값	설명
struct.ROBOT_POSE	구조체 정의 참조

예제

```

1 LPROBOT_POSE lpPose = drfl.get_current_pose(ROBOT_SPACE_JOINT);
2 // 조인트 공간의 현재 로봇 위치 표시
3 for (int k = 0; k < NUM_JOINT; k++ )
4     cout << lpPose->_fPosition[k] << endl;

```

4.2.14 CDRFLEx.get_current_posj

기능

로봇 제어기에서 로봇의 현재 관절 각도를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_POSE	구조체 정의 참조

예제

1	LPROBOT_POSE lpPose = drfl.get_current_posj();
---	--

4.2.15 CDRFLEx.get_desired_posj**기능**

로봇 제어기에서 현재의 목표 관절각을 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_POSE	구조체 정의 참조

예제

1	LPROBOT_POSE lpPose = drfl.get_desired_posj();
---	--

4.2.16 CDRFLEx.get_current_velj**기능**

로봇 제어기에서 로봇의 현재 관절 속도를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_VEL	구조체 정의 참조

예제

```
1 LPROBOT_VEL lpVel = drfl.get_current_velj();
```

4.2.17 CDRFLEx.get_current_posx

기능

현재 태스크 좌표계의 자세와 solution space를 반환한다. 이 때 자세는 eTargetRef를 기준으로 한다.

인수

인수명	자료형	기본값	설명
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
struct.ROBOT_TASK_POSE	구조체 정의 참조

예제

```
1 ROBOT_TASK_POSE* result = Drfl.get_current_posx();
2 float* pos = new float[NUM_TASK];
3 int sol = result->_iTargSol;
4 memcpy(pos, result->_fTargetPos, sizeof(float) * NUM_TASK);
```

4.2.18 CDRFLEx.get_desired_posx

기능

현재 툴의 목표(target) 자세를 반환한다. 이 때 자세는 eTargetRef를 기준으로 한다.

인수

인수명	자료형	기본값	설명
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
enum.ROBOT_POSE	상수 및 열거형 정의 참조

예제

```

1 ROBOT_POSE* result = Drfl.get_desired_posx();
2 for(int i = 0; i < NUM_TASK; i++)
3 {
4     cout << result->fPosition[i] << endl;
5 }
```

4.2.19 CDRFLEx.get_current_tool_flange_posx

기능

로봇 제어기에서 입력된 기준 좌표계에 해당하는 현재 로봇의 툴 플랜지 포즈를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
struct.ROBOT_POSE	구조체 정의 참조

예제

```

1 LPROBOT_POSE lpPose1 =
2 drfl.get_current_tool_flange_posx(COORDINATE_SYSTEM_BASE);
LPROBOT_POSE lpPose2 = drfl.get_desired_posj(COORDINATE_SYSTEM_TOOL);
```

4.2.20 CDRFLEx.get_current_velx

기능

로봇 제어기에서 입력된 기준좌표계에 해당하는 로봇의 현재 툴 속도를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_VEL	구조체 정의 참조

예제

```
1 LPROBOT_VEL lpVel = drfl.get_current_velx();
```

4.2.21 CDRFLEx.get_desired_velx

기능

로봇 제어기에서 입력된 기준좌표계에 해당하는 로봇의 목표 툴 속도를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
struct.ROBOT_VEL	구조체 정의 참조

예제

1	LPROBOT_VEL lpVel = drfl.get_desired_velx();
---	--

4.2.22 CDRFLEx.get_joint_torque**기능**

로봇 제어기에서 로봇의 조인트 센서 토크 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_FORCE	구조체 정의 참조

예제

1	LPROBOT_FORCE lpForce = drfl.get_joint_torque();
---	--

4.2.23 CDRFLEx.get_control_space

기능

로봇 제어기에서 로봇의 현재 제어 공간을 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
enum.ROBOT_SPACE	상수 및 열거형 정의 참조

예제

```
1 ROBOT_SPACE eSpace = drfl.get_control_space();
```

4.2.24 CDRFLEx.get_external_torque

기능

로봇 제어기에서 로봇의 각 관절에서 외력에 의해 발생하는 토크 값을 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
struct.ROBOT_FORCE	구조체 정의 참조

예제

```
1 LPROBOT_FORCE lpETT = drfl.get_external_torque();
```

4.2.25 CDRFLEx.get_tool_force

기능

로봇 제어기에서 입력된 기준 좌표계에서의 현재 툴에 작용하는 외력 값을 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BA SE	상수 및 열거형 정의 참조

리턴

값	설명
struct.ROBOT_FORCE	구조체 정의 참조

예제

```
1 LPROBOT_FORCE lpForce = drfl.get_tool_force();
```

4.2.26 CDRFLEx.get_current_solution_space

기능

로봇 제어기에서 로봇의 자세 정보를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
unsigned char(0~7)	로봇 자세 정보

예제

```

1 unsigned char iSolutionSpace = Drfl.get_current_solution_space();
2 // 로봇의 현재 자세를 유지하면 MoveJ이동
3 float point[6] = { 10, 10, 10, 10, 10, 10 };
4 Drfl.movejx(point, iSolutionSpace, 30, 30);

```

4.2.27 CDRFLEx.get_last_alarm

기능

로봇 제어기에서 가장 최근에 발생한 로그 및 알람 코드를 확인하기 위한 함수이다.

인수

없음

리턴

값	설명
Struct.LOG_ALARM*	구조체 정의 참조

예제

```

1 LPLOG_ALARM pLogAlarm= Drfl.get_last_alarm();
2 switch(pLogAlarm->_iGroup)
3 {
4     case LOG_GROUP_SYSTEMFMK:
5         switch(pLogAlarm->_iLevel)
6         {
7             case LOG_LEVEL_SYSINFO:
8                 cout << "index(" << pLogAlarm->_iIndex << ")", " ;
9                 cout << "param(" << pLogAlarm->_szParam[0]<< ", ";
10                cout << "param(" << pLogAlarm->_szParam[1]<< ", ";
11                cout << "param(" << pLogAlarm->_szParam[2]<< ")" << endl;
12                break;
13            default:
14                break;
15        }
16        break;
17    default:
18        break;
19    }

```

4.2.28 CDRFLEx.get_solution_space

기능

solution space의 값을 구한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개의 Joint Space 정보

리턴

값	설명
unsigned char(0~7)	자세 정보

예제

```

1 float p1[6] = {0, 0, 0, 0, 0, 0};
2 int sol = Drfl.get_solution_space(p1)
3 cout << sol << endl;

```

4.2.29 CDRFLEx.get_orientation_error

기능

축 방향 eTaskAxis에 대한 임의의 pose fPosition1과 fPosition2 사이의 Orientation error 값을 반환한다.

인수

인수명	자료형	기본값	설명
fPosition1	float[6]	-	6개의 Task Space 정보
fPosition2	float[6]		6개의 Task Space 정보
eTaskAxis	enum.TASK_AXI S		상수 및 열거형 정의 참조

리턴

값	설명
float	Orientation Error 값

예제

```

1 float x1[6] = {0, 0, 0, 0, 0, 0};
2 float x2[6] = {10, 20, 30, 40, 50, 60};
3 float diff = Drfl.get_orientation_error(x1, x2, TASK_AXIS_X);
4 cout << diff << endl;

```

4.2.30 CDRFLEx.get_control_mode**기능**

현재 제어 모드를 반환한다.

인수

없음

리턴

값	설명
CONTROL_MODE	상수 및 열거형 정의 참조

예제

```

1 CONTROL_MODE mode =Drfl.get_control_mode();
2 cout << mode << endl;

```

4.2.31 CDRFLEx.get_current_rotm**기능**

입력된 기준좌표계(eTargetRef)에 해당하는 현재 툴의 회전 행렬을 반환한다.

인수

인수명	자료형	기본값	설명
eTarget Ref	enum.COORDINATE_SYSTEM	CORODINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
float[3][3]	Rotation Matrix

예제

```

1 float(*result)[3] = Drfl.get_current_rotm();
2 for (int i=0; i<3; i++)
3 {
4     for (int j=0; j<3; j++)
5     {
6         cout << result[i][j] ;
7     }
8     cout << endl;
9 }
```

4.2.32 CDRFLEX.get_safety_configuration

기능

현재 안전 설정 값을 반환한다.

인수

없음

리턴

값	설명
struct.SAFETY_CONFIGURATION_EX2	구조체 정의 참조
struct.SAFETY_CONFIGURATION_EX2_V3	구조체 정의 참조

예제

```

1 LPSAFETY_CONFIGURATION_EX2 tParam = drfl.get_safety_configuration(); //  
DRCF_VERSION == 2 로 선언된 경우  
2 LPSAFETY_CONFIGURATION_EX2_V3 tParam =  
drfl.get_safety_configuration(); // DRCF_VERSION == 3 으로 선언된 경우

```

4.3 콜백함수 등록 함수

4.3.1 CDRFLEx.set_on_monitoring_state

기능

로봇제어기의 운용 상태 정보 변경시 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 데이터 변경시 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringStateCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringStateCB(const ROBOT_STATE eState)
2 {
3     switch((unsigned char)eState)
4     {
5         case STATE_SAFE_OFF:
6             // 로봇 제어기 서보온
7             drfl.set_robot_control(CONTROL_RESET_SAFET_OFF);
8             break;
9         default:
10            break;
11    }
12 }
13
14 int main()

```

```

15
16     drfl.set_on_monitoring_state(OnMonitoringStateCB);
17 }
```

4.3.2 CDRFLEx.set_on_monitoring_data

기능

로봇 제어기의 현재 위치와 같은 로봇 운용 데이터 정보를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 데이터 변경시 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringDataCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringDataCB(const LPMONITORING_DATA pData)
2 {
3     // 조인트 공간 로봇 위치 데이터 표시
4     cout << "joint data "
5     << pData->_tCtrl._tJoint._fActualPos[0]
6     << pData->_tCtrl._tJoint._fActualPos[1]
7     << pData->_tCtrl._tJoint._fActualPos[2]
8     << pData->_tCtrl._tJoint._fActualPos[3]
9     << pData->_tCtrl._tJoint._fActualPos[4]
10    << pData->_tCtrl._tJoint._fActualPos[5] << endl;
11 }
12
13 int main()
14 {
15     drfl.set_on_monitoring_data(OnMonitoringDataCB);
16 }
```

4.3.3 CDRFLEx.set_on_monitoring_data_ex

기능

로봇 제어기의 현재 위치와 같은 로봇 운용 데이터 확장 정보를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 데이터 변경시 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

setup_monitoring_version 함수를 이용해 모니터링 데이터 버전을 1로 변경하였을 때 활성화된다.

해당 함수는 M2.5 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringDataExCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringDataExCB(const LPMONITORING_DATA_EX pData)
2 {
3     // 사용자 좌표계 로봇 위치 데이터 표시
4     cout << "joint data "
5     << pData->_tCtrl._tUser._fActualPos[0][0]
6     << pData->_tCtrl._tUser._fActualPos[0][1]
7     << pData->_tCtrl._tUser._fActualPos[0][2]
8     << pData->_tCtrl._tUser._fActualPos[0][3]
9     << pData->_tCtrl._tUser._fActualPos[0][4]
10    << pData->_tCtrl._tUser._fActualPos[0][5] << endl;
11 }
12
13 int main()
14 {
15     Drfl.set_on_monitoring_data_ex(OnMonitoringDataExCB);
16 }
```

4.3.4 CDRFLEEx.set_on_monitoring_ctrl_io

기능

로봇 제어기의 Control Box에 장착되어 있는 I/O의 현재 상태 정보를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringCtrlIOCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringCtrlIOCB(const LPMONITORING_CTRLIO pCtrlIO)
2 {
3     // 디지털 입력 GPIO 상태 데이터 표시
4     cout << "gpio data" << endl;
5     for (int i = 0; i < NUM_DIGITAL; i++)
6         cout << "DI#" << i << ":" << pCtrlIO->_tInput._iActualDI[i] <<
7 endl;
8 }
9
10 int main()
11 {
12     drfl.set_on_monitoring_ctrl_io(OnMonitoringCtrlIOCB)
13 }
```

4.3.5 CDRFLEEx.set_on_monitoring_ctrl_io_ex

● 해당 함수는 V2 / V3 제어기에 따라 인터페이스가 다릅니다. 사용하는 제어기 버전에 맞게 #define DRCF_VERSION 2 또는 #define DRCF_VERSION 3 지정 후 DRFL을 임포트 해주세요.

기능

로봇 제어기의 Control Box에 장착되어 있는 I/O의 현재 상태 정보를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

이 기능은 M2.5 버전 이상에서만 사용할 수 있습니다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringCtrlIOExCB	-	콜백 함수 정의 참조 (DRCF_VERSION 2 인 경우)
pCallbackFunc	TOnMonitoringCtrlIOEx2CB	-	콜백 함수 정의 참조 (DRCF_VERSION 3 인 경우)

리턴

없음

예제

```

1 void OnMonitoringCtrlIOExCB(const LPMONITORING_CTRLIO_EX pCtrlIO) // DRCF
2 VERSION 2 인 경우
3 {
4     // 디지털 입력 GPIO 상태 데이터 표시
5     cout << "gpio data" << endl;
6     for (int i = 0; i < NUM_DIGITAL; i++)
7         cout << "DI#" << i << ":" << pCtrlIO->_tInput._iActualDI[i] <<
8         endl;
9 }
10
11 void OnMonitoringCtrlIOExCB(const LPMONITORING_CTRLIO_EX2 pCtrlIO) // DRCF
12 VERSION 3 인 경우
13 {
14     // 디지털 입력 GPIO 상태 데이터 표시
15     cout << "gpio data" << endl;
16     for (int i = 0; i < NUM_DIGITAL; i++)
17         cout << "DI#" << i << ":" << pCtrlIO->_tInput._iActualDI[i] <<
18         endl;
19 }
20
21 int main()
22 
```

```

18
19     drfl.set_on_monitoring_ctrl_io_ex(OnMonitoringCtrlIOCBEx)
20 }
```

4.3.6 CDRFLEEx.set_on_monitoring_modbus

기능

로봇 제어기에 등록되어 있는 모드버스 I/O의 현재 상태 정보를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringModbusCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringModbusCB(const LPMONITORING_MODBUS pModbus)
2 {
3     // 모드버스 IO 상태 데이터 표시
4     cout << "modbus data" << endl;
5     for (int i = 0; i < pModbus->_iRegCount; i++)
6         cout << pModbus->_iRegCount[i]._szSymbol << ":" "
7             << pModbus->_iRegCount[i]._iValue<< endl;}
8
9     int main()
10    {
11        drfl.set_on_monitoring_modbus(OnMonitoringModbusCB)
12    }
```

4.3.7 CDRFLEEx.set_on_log_alarm

기능

로봇 제어기에서 발생하는 모든 알람 및 로그 정보 데이터를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnLogAlarmCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnLogAlarm(LPLOG_ALARM pLogAlarm)
2 {
3     switch(pLogAlarm->_iGroup)
4     {
5         case LOG_GROUP_SYSTEMFMK:
6             switch(pLogAlarm->_iLevel)
7             {
8                 case LOG_LEVEL_SYSINFO:
9                     cout << "index(" << pLogAlarm->_iIndex << ") , ";
10                cout << "param(" << pLogAlarm->_szParam[0]<< ", ";
11                cout << "param(" << pLogAlarm->_szParam[1]<< ", ";
12                cout << "param(" << pLogAlarm->_szParam[2]<< ")" << endl;
13                break;
14            default:
15                break;
16            }
17            break;
18        default:
19            break;
20        }
21    }
22
23 int main()
24 {
25     drfl.set_on_log_alarm(OnLogAlarmCB)
26 }
```

4.3.8 CDRFLE.set_on_tp_popup

기능

DRL에서 tp_popup 명령을 사용했을 경우, 팝업 메시지를 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnTpPopupCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnTpPopup(LPMESSAGE_POPUP tPopup)
2 {
3     cout << "Popup Message: " << tPopup->_szText << endl;
4     cout << "Message Level: " << tPopup->_iLevel << endl;
5     cout << "Button Type: " << tPopup->_iBtnType << endl;
6 }
7 int main()
8 {
9     drfl.set_on_tp_popup(OnTpPopupCB)
10 }
```

4.3.9 CDRFLE.set_on_tp_log**기능**

DRL에서 tp_log명령을 사용했을 경우, 로그 메시지를 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnTpLogCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnTpLog(const char* strLog)
2 {
3     cout << "Log Message: " << strLog << endl;
4 }
5 int main()
6 {
7     drfl.set_on_tp_log(OnTpLogCB)
8 }
```

4.3.10 CDRFLEx.set_on_tp_progress

기능

DRL에서 tp_progress명령을 사용했을 경우, 실행 단계의 정보를 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnTpProgressC B	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnTpProgress(LPMESSAGE_PROGRESS tProgress)
2 {
3     cout << "Progress cnt : " << tProgress->_iCurrentCount << endl;
4     cout << "Current cnt : " << tProgress->_iCurrentCount << endl;
5 }
6 int main()
7 {
8     drfl.set_on_tp_progress(OnTpProgressCB)
9 }
```

4.3.11 CDRFLEx.set_on_tp_get_user_input

기능

DRL에서 tp_get_user_input명령을 사용했을 경우, 사용자 입력을 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnTpGetUserInputCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnTpGetuserInput(LPMESSAGE_INPUT tInput)
2 {
3     cout << "User Input : " << tInput->_szText << endl;
4     cout << "Data Type : " << tInput->_iType << endl;
5 }
6 int main()
7 {
8     drfl.set_on_tp_get_user_input(OnTpGetUserInputCB)
9 }
```

4.3.12 CDRFLEx.set_on_monitoring_access_control

기능

로봇 제어기의 제어권 상태(요청/승락/거절 인한) 변경시 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringAccessControlCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringAccessControlCB(const MONITORING_ACCESS_CONTROL eAccCtrl)
2 {
3     // 제어권 이양 메시지 수신
4     case MONITORING_ACCESS_CONTROL_REQUEST:
5         // 제어권 이양 거부
6         drfl.ManageAccessControl(MANAGE_ACCESS_CONTROL_RESPONSE_NO);
7         break;
8     default:
9         break;
10 }
11
12 int main()
13 {
14     drfl.set_on_monitoring_access_control(OnMonitoringAccessControlCB);
15 }
```

4.3.13 CDRFLEX.set_on_homming_completed

기능

로봇 제어기가 홈밍 제어 모드에 있을 경우, 홈밍 완료 여부를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnHommingCompletedCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnHommingCompletedCB()
2 {
3     // 흡밍 완료 메시지 생성
4     cout << "homming completed" << endl;
5     drfl.Homme(False)
6
7 }
8
9 int main()
10 {
11     drfl.set_on_homming_completed(OnHommingCompletedCB);
12 }
```

4.3.14 CDRFLEX.set_on_tp_initializing_completed

기능

로봇 제어기 부팅후, T/P 어플리케이션에 의해 로봇 제어기가 초기화 과정을 수행할 경우, 초기화 완료 여부를 여부를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnTpInitializingCompletedC B	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnTpInitializingCompletedCB()
2 {
3     // Tp 초기화 여부 확인후 제어권 요청.
4     cout << "tp initializing completed" << endl;
```

```

5     drfl.manage_access_control(MANAGE_ACCESS_CONTROL_REQUEST);
6 }
7
8 int main()
9 {
10    drfl.set_on_tp_initializing_completed(OnTpInitializingCompletedCB);
11 }
```

4.3.15 CDRFLEX.set_on_monitoring_speed_mode

기능

로봇 제어기의 현재 속도 모드를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringSpeedModeC B	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringSpeedModeCB(const MONITORING_SPEED eSpdMode)
2 {
3     // 속도 모드 표시
4     cout << "speed mode: " << (int)eSpeedMode << endl;
5 }
6
7 int main()
8 {
9     drfl.set_on_monitoring_speed_mode(OnMonitoringSpeedModeCB);
10 }
```

4.3.16 CDRFLEx.set_on_mastering_need

기능

로봇 제어기에서 외부 충격으로 인해 로봇의 축이 틀어졌을 경우, 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMasteringNeedCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMasteringNeedCB()
2 {
3     //로봇 축 정렬을 위한 홈링 모드 시작
4     drfl.Home(True)
5 }
6
7 int main()
8 {
9     drfl.set_on_mastering_need(TOnMasteringNeedCB);
10 }
```

4.3.17 CDRFLEx.set_on_program_stopped

기능

로봇 제어기에서 자동모드로 프로그래밍 실행 도중 오류나 사용자 명령에 의해서 종료되었을 경우, 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnProgramStoppedCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnProgramStoppedCB(const PROGRAM_STOP_CAUSE eStopCause)
2 {
3     //프로그램을 재시작 가능 상태 표시
4 }
5
6 int main()
7 {
8     drfl.set_on_program_stopped(OnProgramStoppedCB);
9 }
```

4.3.18 CDRFLEX.set_on_disconnected

기능

로봇 제어기와 연결이 외부 요인이나 혹은 사용자에 의해서 종료되었을 경우, 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnDisconnectedCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnDisconnectedCB()
2 {
3     //로봇 제어기 재접속 및 오류 처리 필요
4 }
5
6 int main()
7 {
8     drfl.set_on_disconnected(OnDisconnectedCB);
9 }
```

4.3.19 CDRFLEx.set_on_monitoring_robot_system

기능

로봇 제어기의 운용 시스템이 변경되었을 경우, 이를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다.
자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringRobotSystemC B	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringRobotSystemCB(ROBOT_STATE eRobotState)
2 {
3     //로봇 운용 상태 변경 시 수행
4 }
5
6 int main()
7 {
8     Drfl.set_on_monitroing_robot_system(OnMonitoringRobotSystemCB);
9 }
```

4.3.20 CDRFLEx.set_on_monitoring_safety_state

기능

로봇 제어기의 안전 상태를 자동으로 확인하기 위한 콜백함수를 등록하기 위한 함수이다. 자동으로 실행되어야 하는 기능을 작성할 때 유용하다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringSafetyState	-	콜백 함수 정의 참조

리턴

없음

예제

```

1 void OnMonitoringSafetyStateCB (SafetyState iState)
2 {
3     //Safety State 업데이트 시 수행
4 }
5
6 int main()
7 {
8     Drfl.set_on_monitoring_safety_state(OnMonitoringSafetyStateCB);
9 }
```

4.4 제어권 관리 함수

4.4.1 CDRFLEx.manage_access_control

기능

로봇 제어기의 제어권을 요청 메시지를 송신하거나 제어권 요청 메시지 수신시 이에 대한 사용자 응답을 처리하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eAccessControl	enum.MANAGE_ACCESS_CONTROL	MANAGE_ACCESS_CONTROL_REQUEST	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 void OnMonitoringAccessControlCB(const MONITORING_ACCESS_CONTROL eAccCtrl)
2 {
3     // 제어권 이양 거부 수신
4     case MONITORING_ACCESS_CONTROL_DENY:
5         // 제어권 없음 표시
6         break;
7     default:
8         break;
9 }
10
11 int main()
12 {
13     drfl.set_on_monitoring_access_control(OnMonitoringAccessControlCB);
14     // 제어권 이양 요청
15     drfl.manage_access_control(MANAGE_ACCESS_CONTROL_REQUEST);
16 }
17 }
```

4.5 기본 제어 함수

4.5.1 CDRFLEx.jog

기능

로봇 제어기에서 로봇의 각 축에 대한 조그 움직임 제어를 수행하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eJointAxis	enum.JOG_AXIS	-	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE	-	상수 및 열거형 정의 참조
fVelocity	float	-	조그 속도(%단위) +: 양의 방향 0: 정지 -: 음의 방향

리턴

값	설명
0	오류
1	성공

예제

```

1 // 로봇베이스를 기준으로 + 방향으로 60% 속도로 조그
2 drfl.jog(JOG_AXIS_JOINT_1, MOVE_REFERENCE_BASE, 60.f);
3 // 정지
4 drfl.jog(JOG_AXIS_JOINT_1, MOVE_REFERENCE_BASE, 0.f);
5 // 로봇베이스를 기준으로 - 방향으로 60% 속도로 조그
6 drfl.jog(JOG_AXIS_JOINT_1, MOVE_REFERENCE_BASE, -60.f);
7 // 정지
8 drfl.jog(JOG_AXIS_JOINT_1, MOVE_REFERENCE_BASE, 0.f);

```

4.5.2 CDRFLEX.move_home

기능

로봇 제어기에서 내/외부 오류로 인하여 로봇의 각 축이 틀어졌을 경우, 로봇의 각 축을 정렬하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eMode	MOVE_HOME	MOVE_HOME_MECHANIC	상수 및 열거형 정의 참조
bRun	unsigned char	1	0: 정지 1: 동작

리턴

값	설명
0	오류
1	성공

예제

```

1 // Homing 동작 수행
2 drfl.move_home()
3 // Homing 동작 정지
4 drfl.move_home(MOVE_HOME_MECHANIC, (unsigned char)0);

```

⚠ 주의

Homing 완료 후 Move 명령어 사용 시, Homming 이후에 wait 명령어 사용하기를 권장함

4.6 모션 제어 함수

4.6.1 CDRFLEx.movej

기능

로봇제어기에서 로봇을 현재 관절위치에서 목표 관절위치까지 이동시키기 위한 함수이다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목표 관절 위치
fTargetVel	float	-	속도
fTargetAcc	float	-	가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
fBlendingRadius	float	0.f	blending radius
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조

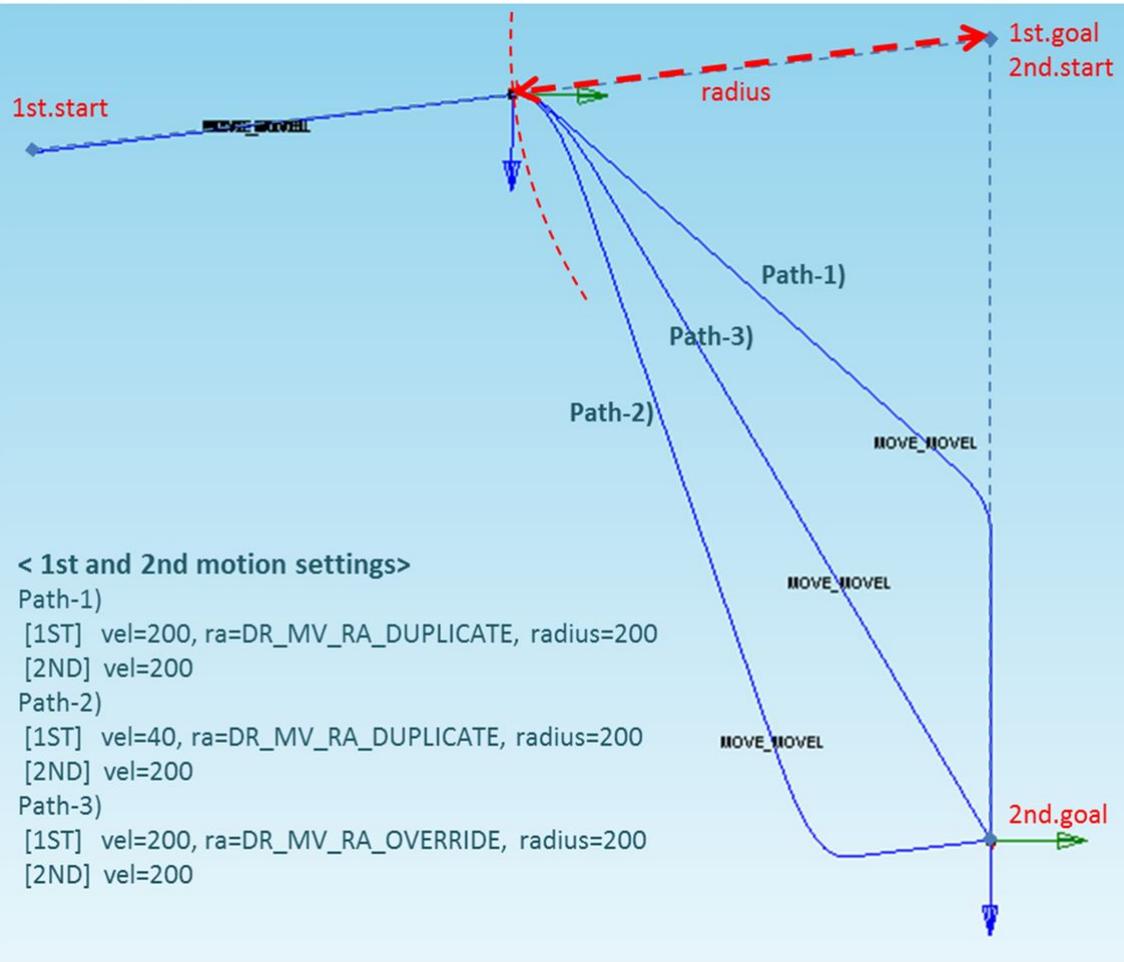
● 알아두기

- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.

⚠ 주의

eBlendingType이 BLENDING_SPEED_TYPE_DUPLICATE이고 fBlendingRadius 가 0보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간 보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 참고 하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



리턴

값	설명
0	오류
1	성공

예제

```

1 // CASE 1
2 float q0[6] = { 0, 0, 90, 0, 90, 0 };
3 float jvel=10;
4 float jacc=20;

```

```

5 drfl.movej(q0, jvel, jacc);
6 # 속도 10(deg/sec), 가속도 20(deg/sec2)로 q0 관절각으로 이동
7
8 // CASE 2
9 float q0[6] = { 0, 0, 90, 0, 90, 0 };
10 float jTime=5;
11 drfl.movej(q0, 0, 0, jTime)
12 # q0 관절각까지 5초의 도착시간을 가지고 이동
13
14 // CASE 3
15 float q0[6] = { 0, 0, 90, 0, 90, 0 };
16 float q1[6] = {90, 0, 90, 0, 90, 0 };
17 float jvel=10;
18 float jacc=20;
19 float blending_radius=50;
20 drfl.movej(q0, jvel, jacc, 0, MOVE_MODE_ABSOLUTE, blending_radius);
21 // q0 관절각으로 이동하며 q0 관절각과 일치하는 위치로부터 50mm의 거리
22 // 가 될 때 다음 모션을 수행하도록 설정
23 drfl.movej(q1, jvel, jacc, 0, MOVE_MODE_ABSOLUTE, 0,
24 BLENDING_SPEED_TYPE_DUPLICATE));
    // 직전모션과 Blending되어 q1 관절각으로 이동

```

4.6.2 CDRFLEx.moves

기능

로봇 제어기에서 로봇을 현재 위치에서 관절공간의 경유점들을 거쳐 목표위치(마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동시키기 위한 함수이다. 입력된 속도/가속도는 경로 중 최대 속도/가속도를 의미하며 입력되는 경유점의 위치에 따라 모션 중의 감속 또는 가속이 결정된다.

인수

인수명	자료형	기본값	설명
fTargetPos	Float[MAX_SPLINE_POINT][6]	-	최대 100개까지의 경유점 리스트
nPosCount	unsigned char	-	유효 경유점 개수
fTargetVel	float/float[6]	-	속도
fTargetAcc	float/float[6]	-	가속도
fTargetTime	float	0.f	도달 시간 [sec]

인수명	자료형	기본값	설명
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조

❶ 알아두기

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 position list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다. (position list=[q1, q2, ..., q(n-1), q(n)]로 이루어질 때 q1은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // CASE 1 : 절대각도 입력 (mod=MOVE_MODE_ABSOLUTE)
2 float jpos[4][6];
3 float jvel=10;
4 float jacc=10;
5 int jposNum = 4;
6 jpos[0][0]=0; jpos[0][1]=0; jpos[0][2]=-30; jpos[0][3]=0; jpos[0][4]=-30;
7 jpos[0][5]=0;
8 jpos[1][0]=90; jpos[1][1]=0; jpos[1][2]=0; jpos[1][3]=0; jpos[1][4]=0;
9 jpos[1][5]=0;
10 jpos[2][0]=0; jpos[2][1]=0; jpos[2][2]=-30; jpos[2][3]=0; jpos[2][4]=-30;
11 jpos[2][5]=0;
12 jpos[3][0]=-90; jpos[3][1]=0; jpos[3][2]=0; jpos[3][3]=0; jpos[3][4]=0;
13 jpos[3][5] = 0;
14 drfl.movesj(jpos, jposNum, jvel, jacc);
15 // jpos에 정의된 절대경유점 집합을 연결하는 스플라인 곡선을 최대속도
16 // 10(deg/sec), 최대가속도 10(deg/sec2)로 움직임

// CASE 2 : 상대각도 입력 (mod=MOVE_MODE_RELATIVE)
float jpos[4][6];
float jvel=10;

```

```

17 float jacc=10;
18 int jposNum = 4;
19 jpos[0][0]=0; jpos[0][1]=0; jpos[0][2]=-30; jpos[0][3]=0; jpos[0][4]=-30;
20 jpos[0][5]=0;
// 시작위치 + jpos[0]
21 jpos[1][0]=90; jpos[1][1]=0; jpos[1][2]=30; jpos[1][3]=0; jpos[1][4]=30;
jpos[1][5]=0;
// 시작위치 + jpos[0] + jpos[1]
22 jpos[2][0]=-90; jpos[2][1]=0; jpos[2][2]=-30; jpos[2][3]=0; jpos[2][4]=-30;
jpos[2][5]=0;
// 시작위치 + jpos[0] + jpos[1] + jpos[2]
23 jpos[3][0]=-90; jpos[3][1]=0; jpos[3][2]=30; jpos[3][3]=0; jpos[3][4]=30;
jpos[3][5]=0;
// 시작위치 + jpos[0] + jpos[1] + jpos[2] + jpos[3]
24 drfl.movesj(jpos, jposNum, jvel, jacc, time, MOVE_MODE_RELATIVE);
// jpos에 정의된 상대경유점 집합을 연결하는 스플라인 곡선을 최대속도
25 // 10(deg/sec), 최대가속도 10(deg/sec2)로 움직임
26
27
28
29

```

4.6.3 CDRFLEx.movesx

기능

로봇 제어기에서 로봇을 현재 위치에서 작업공간의 경유점들을 거쳐 목표위치(마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동시키기 위한 함수이다. 입력된 속도/가속도는 경로 중 최대 속도/가속도이며 정속모션 옵션을 선택할 경우 조건에 따라 입력한 속도로 정속도의 모션을 수행합니다.

인수

인수명	자료형	기본값	설명
fTargetPos	float [MAX_SPLINE_POINT][6]	-	최대 100개까지의 경유점 정보
nPosCount	unsigned char	-	유효 경유점 개수
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.f	도달 시간 [sec]

인수명	자료형	기본값	설명
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조
eVelOpt	enum.SPLINE_VELOCITY_OPTION	SPLINE_VELOCITY_OPTION_DEFAULT	상수 및 열거형 정의 참조

❶ 알아두기

- fTargetVel에 하나의 인자를 입력한 경우(예를들어, fTargetVel =[30, 0]) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를들어, fTargetAcc =[60, 0]) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 position list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다. (position list=[q1, q2,...,q(n-1), q(n)])로 이루어질 때 q1은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

eVelOpt을 SPLINE_VELOCITY_OPTION_CONST 옵션(등속모션)을 선택할 경우 입력된 경유점 간 거리와 속도 조건에 따라 등속모션을 사용할 수 없을 수 있으며, 이 경우에 변속모션 (eVelOpt =SPLINE_VELOCITY_OPTION_DEFAULT)으로 자동 전환됩니다.

리턴

값	설명
0	오류
1	성공

예제

```
1 // CASE 1 : 절대좌표계 기준 이동 (mod= MOVE_MODE_ABSOLUTE)
2 float xpos[4][6];
```

```

3   int xposNum = 4;
4   float tvel={ 50, 100 };
5   float tacc={ 50, 100 };
6   xpos[0][0]=559; xpos[0][1]=434.5; xpos[0][2]=651.5;
7   xpos[0][3]=0; xpos[0][4]=180; xpos[0][5]=0;
8   xpos[1][0]=559; xpos[1][1]=434.5; xpos[1][2]=251.5;
9   xpos[1][3]=0; xpos[1][4]=180; xpos[1][5]=0;
10  xpos[2][0]=559; xpos[2][1]=234.5; xpos[2][2]=251.5;
11  xpos[2][3]=0; xpos[2][4]=180; xpos[2][5]=0;
12  xpos[3][0]=559; xpos[3][1]=-234.5; xpos[3][2]=451.5;
13  xpos[3][3]=0; xpos[3][4]=180; xpos[3][5]=0;
14  drfl.movesx(xpos, xposNum, tvel, tacc, 0, MOVE_MODE_ABSOLUTE);
15  // 현재위치에서 시작하여 xpos에 정의된 경유점 집합을 연결하는 스플라인
16  // 곡선을 최대속도 50, 50(mm/sec, deg/sec), 최대가속도 100, 100(mm/sec2,
17  // deg/sec2)로 움직임
18
19 // CASE 2 : 상대좌표계 기준 이동 (mod= MOVE_MODE_RELATIVE)
20 float xpos[4][6];
21 int xposNum = 4;
22 float tvel={ 50, 100 };
23 float tacc={ 50, 100 };
24 xpos[0][0]=0; xpos[0][1]=400; xpos[0][2]=0;
25 xpos[0][3]=0; xpos[0][4]=0; xpos[0][5]=0;
26 // 시작위치에서 상대좌표 xpos[0]만큼의 동차변환 → x0
27 xpos[1][0]=0; xpos[1][1]=0; xpos[1][2]=-400;
28 xpos[1][3]=0; xpos[1][4]=0; xpos[1][5]=0;
29 // x0에서 상대좌표 xpos[1]만큼의 동차변환 → x1
30 xpos[2][0]=0; xpos[2][1]=-200; xpos[2][2]=0;
31 xpos[2][3]=0; xpos[2][4]=0; xpos[2][5]=0;
32 // x1에서 상대좌표 xpos[2]만큼의 동차변환 → x2
33 xpos[3][0]=0; xpos[3][1]=0; xpos[3][2]=200;
34 xpos[3][3]=0; xpos[3][4]=0; xpos[3][5]=0;
35 // x3에서 상대좌표 xpos[3]만큼의 동차변환 → x3
36 drfl.movesx(xpos, xposNum, tvel, tacc, 0, MOVE_MODE_RELATIVE);
37 // 현재위치에서 시작하여 xpos에 상대정의된 경유점 집합을 연결하는
38 // 스플라인 곡선을 최대속도 50, 50(mm/sec, deg/sec), 최대가속도 100,
39 // 100(mm/sec2, deg/sec2)로 움직임

```

4.6.4 CDRFLEx.move_spiral

기능

로봇제어기에서 방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축 방향으로 병행하면서 이동시키기 위한 함수이다. 현재 위치에서 eMoveReference로 지정한 좌표계 상의 axis 방향에 수직인 평면에서의 나선궤적과 axis 방향으로의 직선궤적을 동시에 따라 이동한다.

인수

인수명	자료형	기본값	범위	설명
eTaskAxis	enum.TASK_AXIS	-	-	상수 및 열거형 정의 참조
fRevolution	float	-	rev > 0	총 회전수 [revolution]
fMaximumRadius	float	-	rmax > 0	spiral 최종 반경 [mm]
fMaximumLength	float	-		axis 방향으로 이동하는 거리 [mm]
fTargetVel	float[2]	-		선속도, 각속도
fTargetAcc	float[2]	-		선가속도, 각가속도
fTargetTime	float	0.0	time ≥ 0	총 수행시간 <sec>
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_TOOL		상수 및 열거형 정의 참조

● 알아두기

- fRevolution 는 spiral 모션의 총 회전수를 의미합니다.
- fMaximumRadius 는 spiral 모션의 최대 반경을 의미합니다.
- fMaximumLength 는 모션 동안 axis 방향으로 병진하는 거리를 의미합니다. 단, 음수인 경우 -axis 방향 병진합니다.
- fTargetVel 은 spiral 모션의 이동 속도를 의미합니다.
- fTargetAcc 는 spiral 모션의 이동 가속도를 의미합니다.
- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eTaskAxis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- eMoveReference 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다. 이 경우 fTargetVel, fTargetAcc 또는 fTargetTime 값을 작게 조정하는 것을 권장합니다.

리턴

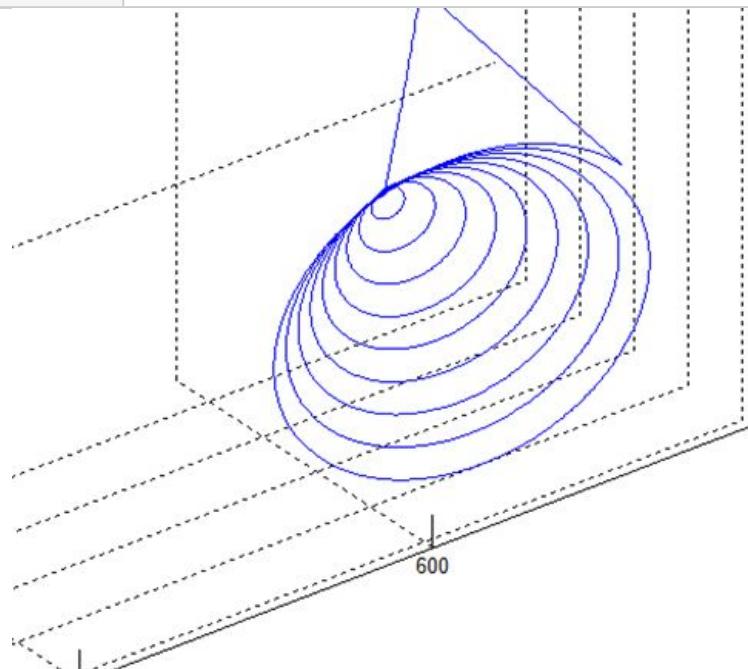
값	설명
0	오류
1	성공

예제

```

1 float rev = 3;
2 float rmax = 50;
3 float lmax = 50;
4 float tvel = { 50, 50 };
5 float tacc = { 100, 100 };
6 Drfl.move_spiral(TASK_AXIS_Z, rev, rmax, lmax, tvel, tacc);
7 // 현재위치에서 시작하여 중심에서 최대 50mm 반경까지 나선형 웨직임
8 // 속도 50, 50(mm/sec, deg/sec), 가속도 100, 100(mm/sec2, deg/sec2)를
9 // 유지하며, 동시에 Tool z방향으로 50mm까지 이동

```



4.6.5 CDRFLEx.move_periodic

기능

로봇제어기에서 현재 위치에서 시작하는 상대 모션으로 입력된 기준 좌표계(eMoveReference)의 각 축(병진 및 회전)에 대한 Sine 함수 기반으로 주기 모션을 수행합니다. 각 axis 별 모션의 특성은 fAmplitude와 fPeriodic에 의해 결정되고, 가감속 시간과 총 모션 시간은 주기, 반복, 횟수에 의해 설정됩니다.

인수

인수명	자료형	기본값	범위	설명
fAmplitude	float[6]	-	>=0	Amplitude(-amp에서 +amp 사이 모션) [mm] or [deg]
fPeriodic	float[6]	-	>=0	period(1주기 소요 시간)[sec]
fAccelTime	float	-	>=0	Acc-, dec- time [sec]
nRepeat	unsigned char	-	> 0	반복 횟수
eMoveReference	enum	MOVE_REFERENCE_TOOL		상수 및 열거형 정의 참조

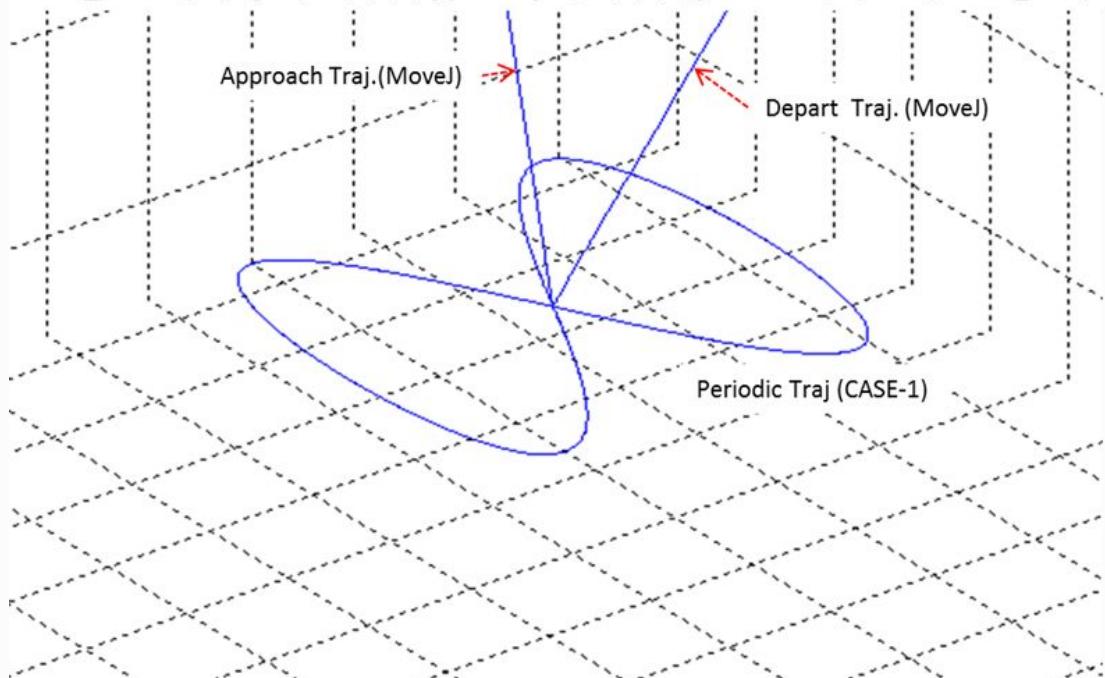
알아두기

- fAmplitude는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 amp를 값으로 하는 6개 원소의 list 형태로 입력해야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 amp를 0으로 입력해야 합니다.
- fPeriodic는 해당 방향 모션의 1회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 period를 값으로 하는 총 6개 원소의 list 형태로 입력하거나 대표값을 입력해야합니다.
- fAccelTime은 주기모션의 시작과 끝의 가속 및 감속 시간을 의미합니다. 입력된 가감속시간과 최대주기*1/4 중 큰 값이 적용됩니다. 입력된 가감속 시간이 전체모션시간의 1/2을 초과하는 경우 에러가 발생합니다.
- nRepeat은 가장 큰 period 값을 가지는 축(기준 축)의 반복 횟수를 정의하며, 이에 따라 총 모션 시간이 결정됩니다. 나머지 축의 반복 횟수는 모션 시간에 따라 자동결정됩니다. 모션이 정상 종료되는 경우 종료 위치가 시작 위치와 일치하게 하도록 나머지 축 모션은 기준 축 모션이 종료되기 전에 멈춥니다.

저 종료될 수 있습니다. 모든 축의 모션이 동시 종료되지 않는 경우 감속구간에서의 경로는 이전 경로에서 벗어나게 됩니다. 관련한 사항은 아래 이미지를 참조하십시오

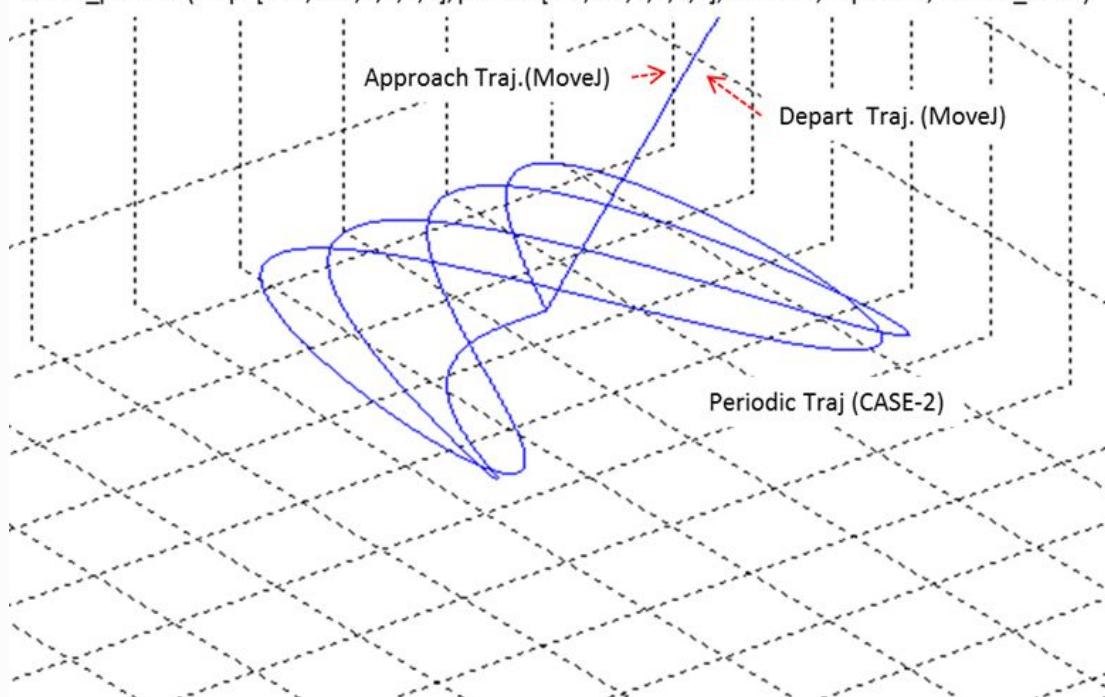
CASE-1) All-axis motions end at the same time

```
move_periodic(amp=[100,100,0,0,0,0], period=[3.2,1.6,0,0,0,0], atime=3.1, repeat=2, ref=DR_BASE)
```



CASE-2) Diff-axis motions end individually

```
move_periodic(amp=[100,100,0,0,0,0], period=[3.2,1.5,0,0,0,0], atime=0, repeat=2, ref=DR_BASE)
```



- eMoveReference 는 반복 모션의 기준 좌표계를 의미합니다.

- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.

최대속도=진폭(amp)*2*pi(3.14)/주기(period)

(예, 진폭=10mm, 주기=1초인 경우 최대속도=62.83mm/sec)

- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 <#1>
2 float fAmplitude[NUM_TASK] = {10,0,0,0,30,0};
3 float fPeriod[NUM_TASK] = { 1, 1, 1, 1, 1, 1 };
4 drfl.move_periodic(fAmplitude, fPeriod, 0.2, 5, MOVE_REFERENCE_TOOL);
// Tool 좌표계 x축(10mm 진폭, 1초 주기) 모션과 y회전축(진폭 30deg, 1초 주기)
// 모션이 총 5회 반복 수행
5 <#2>
6 float fAmplitude[NUM_TASK] = {10,0,20,0,0.5,0};
7 float fPeriod[NUM_TASK] = { 1,0,1.5,0,0,0 };
8 drfl.move_periodic (fAmplitude, fPeriod, 0.5, 3, MOVE_REFERENCE_BASE);
// BASE 좌표계 x축(10mm 진폭, 1초 주기), z축(20mm 진폭, 1.5초 주기) 모션이
// 총 3회 반복 수행됨, y회전축 모션은 period가 ‘zero(0)’이므로 미수행
// z축 모션의 주기가 크므로 총 모션 시간은 약 5.5초(1.5초*3회 + 가감속 1초)
9 // 이며, x축은 4.5회 반복 수행
10
11
12
13
14

```

4.6.6 CDRFLEx.amovej

기능

비동기 방식의 movej로 블렌딩을 위한 fBlendingRadius 인자를 갖지 않는 점을 제외하고는 movej 함수와 동일하게 동작한다. 그러나 해당 명령어는 비동기 방식의 특성상 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목표 관절 위치
fTargetVel	float/float[6]	-	속도
fTargetAcc	float/float[6]	-	가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조

알아두기

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- 옵션 eBlendingType 및 fTargetVel / fTargetAcc 에 따른 blending 시의 경로는 movej() 모션 설명을 참조할 것

리턴

값	설명
0	오류
1	성공

예제

```

1 float q0[6] = { 0, 0, 90, 0, 90, 0 };
2 float q1[6] = { 90, 0, 90, 0, 90, 0 };
3 float q99[6] = { 0, 0, 0, 0, 0, 0 };

```

```

4   float jvel=10;
5   float jacc=20;
6   drfl.amovej(q0, jvel, jacc);
7   Sleep(3000);
8
9   Drfl.amovej (q1, jvel, jacc);
10  drfl.mwait(); // 모션이 종료할 때까지 프로그램 일시 중지
11
12  Drfl.movej(q99, jvel, jacc);

```

4.6.7 CDRFLEx.amovel

기능

비동기 방식의 move 모션으로 블렌딩을 위한 fBlendingRadius 인자를 갖지 않는 점을 제외하고 move와 동일하게 작동한다. 그러나 해당 명령어는 비동기 방식의 특성상 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목적 TCP 위치
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	-	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
eMoveMode	enum.MOVE_MODE_E	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE_E	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조

알아두기

- fTargetVel에 하나의 인자를 입력한 경우(예를 들어, fTargetVel = {30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를 들어, fTargetAcc = {60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각각속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다
- 옵션 eBlendingType 및 fTargetVel / fTargetAcc에 따른 blending 시의 경로는 MoveL() 모션 설명을 참조할 것

리턴

값	설명
0	오류
1	성공

예제

```

1 // x1으로 모션시작 후 2초 후에 D-Out
2 float x1[6] = { 559, 434.5, 651.5, 0, 180, 0 };
3 float tvel = { 50, 50 };
4 float tacc = { 100, 100 };
5 drfl.amovel(x1, tvel, tacc);
6 Sleep(2000);
7 drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.6.8 CDRFLEEx.amovec

기능

비동기 방식의 movec모션으로 블렌딩을 위한 fBlendingRadius 인자를 갖지 않는 점을 제외하고 movec와 동일하게 작동한다. 그러나 해당 명령어는 비동기 방식의 특성상 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다.

인수

인수명	자료형	기본값	설명
fTargetPos[0]	float[6]	-	경유 지점

인수명	자료형	기본값	설명
fTargetPos[1]	float[6]		목표 위치
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE_E	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE_E	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조
fTargetAngle2	float	0.f	angle1
fTargetAngle2	float	0.f	angle2
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조

❶ 알아두기

- fTargetVel에 하나의 인자를 입력한 경우(예를들어, fTargetVel =[30, 0]) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를들어, fTargetAcc =[60, 0]) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 fTargetPos[0] 과 fTargetPos[1] 는 각각 앞 선 위치값에 대한 상대좌표로 정의됩니다. (fTargetPos[0]은 시작점 대비 상대좌표, fTargetPos[1]는 fTargetPos[0]대비 상대좌표)
- fTargetAngle1이 0보다 크고, fTargetAngle2이 0인 경우 fTargetAngle1은 Circular path 상의 총 회전각이 적용됩니다.
- fTargetAngle1과 fTargetAngle2가 0보다 큰 경우, fTargetAngle1은 circular path 상에서 정속으로 이동하는 총 회전각을, fTargetAngle2는 가속과 감속을 위한 회전 구간의 회전각을 의미합니다. 이 때 총 이동각은 fTargetAngle1+ 2 X fTargetAngle2만큼 circular path 상을 움직입니다.

- 옵션 eBlendingType 와 fTargetVel / fTargetAcc 에 따른 블렌딩 상태의 경로는 movej() 모션 설명을 참조하십시오

리턴

값	설명
0	오류
1	성공

예제

```

1 // x1의 두 점을 경유하는 원호모션 시작 후 3초 후에 D-Out
2 float x1[2][6] = { { 559, 434.5, 651.5, 0, 180, 0 }, { 559, 434.5, 251.5,
3 0, 180, 0 } };
4 float tvel = { 50, 50 }; // 태스크 속도를 50(mm/sec), 50(deg/sec)로 설정
5 float tacc = { 100, 100 }; // 태스크 가속도를 100(mm/sec2), 100(deg/sec2)로 설정
6
7 drfl.amovec(x1, tvel, tacc);
Sleep(3000);
drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.6.9 CDRFLEEx.amovesj

기능

비동기 방식의 movesj모션으로 비동기 처리 방식외에는 movesj()와 동일하게 동작하며, 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다. amovesj()에 의한 모션이 종료되기 전에 발생하는 새로운 모션 명령어는 안전상의 이유로 오류를 발생시킨다. 따라서 amovesj()와 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amovesj()에 의한 모션이 종료된 것을 확인한 후 새로운 모션 명령어가 시작되도록 해야한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[MAX_SPLINE_POINT][6]	-	최대 100개까지의 경유점 리스트
nPosCount	unsigned char	-	유효 경유점 개수
fTargetVel	float/float[6]	-	속도

인수명	자료형	기본값	설명
fTargetAcc	float/float[6]	-	가속도
fTargetTime	float	0.0	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조

❶ 알아두기

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 position list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다. (position list=[q1, q2, ..., q(n-1), q(n)]로 이루어질 때 q1은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블랜딩을 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // jpos의 모든 점을 경유하는 스플라인모션 시작 후 3초 후에 D-Out
2 float jpos[4][6];
3 float jvel=10;
4 float jacc=10;
5 int jposNum = 4;
6 jpos[0][0]=0; jpos[0][1]=0; jpos[0][2]=-30; jpos[0][3]=0; jpos[0][4]=-30;
7 jpos[0][5]=0;
8 jpos[1][0]=90; jpos[1][1]=0; jpos[1][2]=0; jpos[1][3]=0; jpos[1][4]=0;
jpos[1][5]=0;
9 jpos[2][0]=0; jpos[2][1]=0; jpos[2][2]=-30; jpos[2][3]=0; jpos[2][4]=-30;
jpos[2][5]=0;

```

```

9   jpos[3][0]=-90; jpos[3][1]=0; jpos[3][2]=0; jpos[3][3]=0; jpos[3][4]=0;
10  jpos[3][5] = 0;
11  drfl.movesj(jpos, jposNum, jvel, jacc);
12  Sleep(3000);
13  drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.6.10 CDRFLEx.amovesx

기능

비동기 방식의 movesx모션으로 비동기 처리 이외에는 movesx()와 동일하게 동작하며 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다, amovesx()에 의한 모션이 종료되기 전에 발생하는 새로운 모션 명령어는 안전상의 이유로 오류를 발생시킨다. 따라서 amovesx 함수()와 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amovesx()에 의한 모션이 종료된 것을 확인한 후 새로운 모션 명령어가 시작되도록 해야한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float [MAX_SPLINE_POINT][6]	-	최대 100개까지의 경유점 정보
nPosCount	unsigned char	-	유효 경유점 개수
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.0	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조

인수명	자료형	기본값	설명
eVelOpt	enum.SPLINE_VELOCITY_OPTION	SPLINE_VELOCITY_OPTION_DEFAULT	상수 및 열거형 정의 참조

❶ 알아두기

- fTargetVel에 하나의 인자를 입력한 경우(예를들어, fTargetVel =[30, 0]) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를들어, fTargetAcc =[60, 0]) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 position list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다. (position list=[q1, q2, ..., q(n-1), q(n)]로 이루어질 때 q1은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

eVelOpt을 SPLINE_VELOCITY_OPTION_CONST 옵션(등속모션)을 선택할 경우 입력된 경유점 간 거리와 속도 조건에 따라 등속모션을 사용할 수 없을 수 있으며, 이 경우에 변속모션 (eVelOpt =SPLINE_VELOCITY_OPTION_DEFAULT)으로 자동 전환됩니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // xpos의 모든 점을 경유하는 스플라인모션 시작 후 3초 후에 D-out
2 float xpos[4][6];
3 int xposNum = 4;
4 float tvel={ 50, 100 };
5 float tacc={ 50, 100 };
6 xpos[0][0]=559; xpos[0][1]=434.5; xpos[0][2]=651.5;
7 xpos[0][3]=0; xpos[0][4]=180; xpos[0][5]=0;
8 xpos[1][0]=559; xpos[1][1]=434.5; xpos[1][2]=251.5;
9 xpos[1][3]=0; xpos[1][4]=180; xpos[1][5]=0;
10 xpos[2][0]=559; xpos[2][1]=234.5; xpos[2][2]=251.5;

```

```

11  xpos[2][3]=0; xpos[2][4]=180; xpos[2][5]=0;
12  xpos[3][0]=559; xpos[3][1]= 234.5; xpos[3][2]=451.5;
13  xpos[3][3]=0; xpos[3][4]=180; xpos[3][5]=0;
14  drfl.amovesx(xpos, xposNum, tvel, tacc, 0, MOVE_MODE_ABSOLUTE);
15  drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.6.11 CDRFLEX.amoveb

기능

비동기 방식의 moveb모션으로 비동기 처리 외에는 moveb() 함수와 동일하게 동작하며 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다. amoveb() 함수에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킨다. 따라서 amoveb() 함수와 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amoveb()에 의한 모션이 종료된 것을 확인한 후 새로운 모션 명령어가 시작되도록 해야한다

인수

인수명	자료형	기본값	설명
tTargetPos	struct MOVE_POS B [MAX_MOV EB_POINT]	-	최대 25개까지의 경로 정보
nPosCount	unsigned char	-	유효 경로 정보 개수
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOV E_MODE	MOVE_MO DE_ ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOV E_REFEREN CE	MOVE_REF ERENCE_B ASE	상수 및 열거형 정의 참조

알아두기

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 posb list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

- tTargetPos 에서 blending radius가 0인 경우, 사용자 입력 오류가 나타납니다.
- 연속된 Line-Line segment가 같은 방향을 가질 경우 Line의 중복입력으로 사용자 입력 오류가 나타납니다.
- 블렌딩 구간에서 조건에 따라 급격하게 방향전환이 발생하게 되는 경우 급가속을 방지하기 위해 사용자 입력오류가 나타납니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // xb의 모든 경로를 따르는 모션 시작 후 3초 후에 D-Out
2 MOVE_POSB xb[4];
3 memset(xb, 0x00, sizeof(xb));
4 int segNum = 4;
5 float tvel = { 50, 50 };
6 float tacc = { 100, 100 };
7 xb[0]._iBlendType = 0; // line
8 xb[0]._fBlendRad = 50;
9 xb[0]._fTargetPos[0][0] = 559;
10 xb[0]._fTargetPos[0][1] = 234.5;
11 xb[0]._fTargetPos[0][2] = 651.5;
12 xb[0]._fTargetPos[0][3] = 0;
13 xb[0]._fTargetPos[0][4] = 180;
14 xb[0]._fTargetPos[0][5] = 0;
15 xb[1]._iBlendType = 1; // circle
16 xb[1]._fBlendRad = 50;
17 xb[1]._fTargetPos[0][0] = 559;
18 xb[1]._fTargetPos[0][1] = 234.5;

```

```

19 xb[1]._fTargetPos[0][2] = 451.5;
20 xb[1]._fTargetPos[0][3] = 0;
21 xb[1]._fTargetPos[0][4] = 180;
22 xb[1]._fTargetPos[0][5] = 0;
23 xb[1]._fTargetPos[1][0] = 559;
24 xb[1]._fTargetPos[1][1] = 434.5;
25 xb[1]._fTargetPos[1][2] = 451.5;
26 xb[1]._fTargetPos[1][3] = 0;
27 xb[1]._fTargetPos[1][4] = 180;
28 xb[1]._fTargetPos[1][5] = 0;
29 xb[2]._iBlendType = 0; // line
30 xb[2]._fBlendRad = 50;
31 xb[2]._fTargetPos[0][0] = 559;
32 xb[2]._fTargetPos[0][1] = 434.5;
33 xb[2]._fTargetPos[0][2] = 251.5;
34 xb[2]._fTargetPos[0][3] = 0;
35 xb[2]._fTargetPos[0][4] = 180;
36 xb[2]._fTargetPos[0][5] = 0;
37 xb[3]._iBlendType = 0; // line
38 xb[3]._fBlendRad = 50;
39 xb[3]._fTargetPos[0][0] = 559;
40 xb[3]._fTargetPos[0][1] = 234.5;
41 xb[3]._fTargetPos[0][2] = 251.5;
42 xb[3]._fTargetPos[0][3] = 0;
43 xb[3]._fTargetPos[0][4] = 180;
44 xb[3]._fTargetPos[0][5] = 0;
45 drfl.amoveb(xb, segNum, tvel, tacc);
46 Sleep(3000);
47 drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.6.12 CDRFLEX.amove_spiral

기능

비동기 방식의 move_spiral 모션으로 비동기 처리 외에는 move_spiral() 함수와 동일하게 동작하며 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다.

amove_spiral() 함수에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킨다. 따라서 amove_spiral() 함수과 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amove_spiral()에 의한 모션이 종료된 것을 확인한 후 새로운 모션명령어가 시작되도록 해야한다.

본 명령은 방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축방향으로 병행하는 모션을 수행합니다. 현재 위치에서 기준 좌표계(eMoveReference) 지정한 좌표계 상의 축 방향에 수직인 평면에서의 나선궤적과 축 방향으로의 직선궤적을 동시에 따라 이동한다.

인수

인수명	자료형	기본값	범위	설명
eTaskAxis	enum.TASK_AXI_S	-	-	상수 및 열거형 정의 참조
fRevolution	float	-	rev > 0	총 회전수 [revolution]
fMaximumRadius	float	-	rmax > 0	spiral 최종 반경 [mm]
fMaximumLength	float	-		axis 방향으로 이동하는 거리 [mm]
fTargetVel	float[2]	-		선속도, 각속도
fTargetAcc	float[2]	-		선가속도, 각가속도
fTargetTime	float	0.0	time ≥ 0	총 수행시간 <sec>
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_TOOL		상수 및 열거형 정의 참조

알아두기

- fRevolution 는 spiral 모션의 총 회전수를 의미합니다.
- fMaximumRadius 는 spiral 모션의 최대 반경을 의미합니다.
- fMaximumLength 는 모션 동안 axis 방향으로 병진하는 거리를 의미합니다. 단, 음수인 경우 -axis 방향 병진합니다.
- fTargetVel 은 spiral 모션의 이동 속도를 의미합니다.
- fTargetAcc 는 spiral 모션의 이동 가속도를 의미합니다.
- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eTaskAxis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- eMoveReference 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다.
이 경우 fTargetVel, fTargetAcc 또는 fTargetTime 값을 작게 조정하는 것을 권장합니다.

리턴

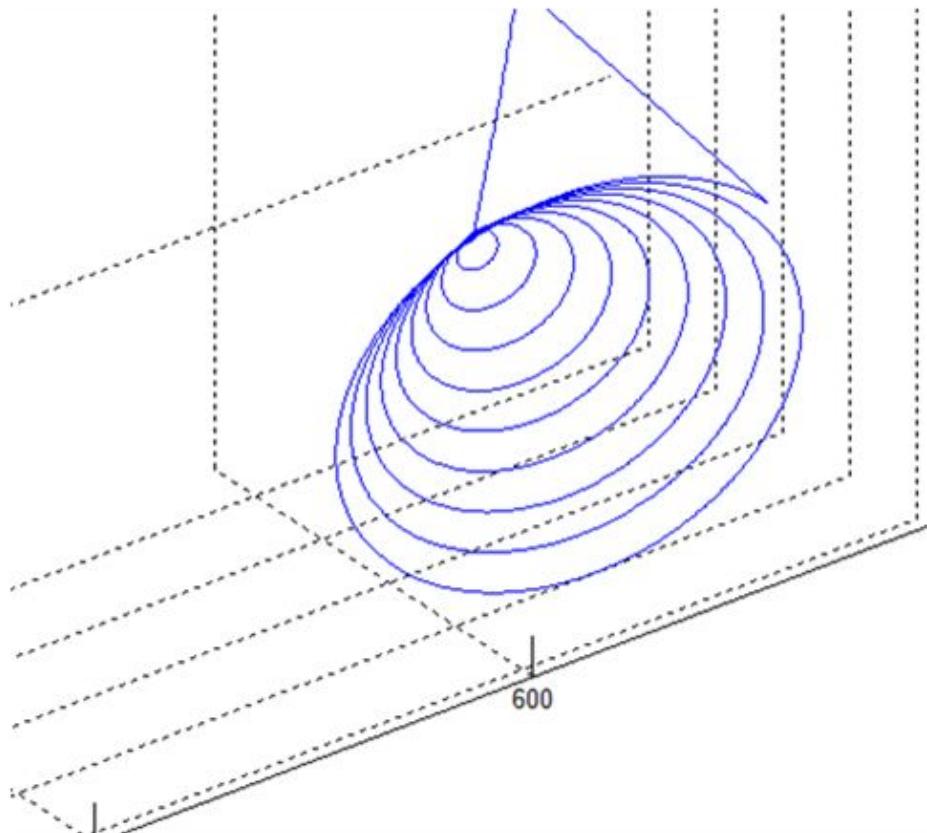
값	설명
0	오류
1	성공

예제

```

1 // 나선형 모션 시작 후 3초 후에 D-Out
2 float rev = 3;
3 float rmax = 50;
4 float lmax = 50;
5 float tvel = { 50, 50 };
6 float tacc = { 100, 100 };
7 Drfl.amove_spiral(TASK_AXIS_Z, rev, rmax, lmax, tvel, tacc);
8 Sleep(3000);
9 drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```



4.6.13 CDRFLEx.amove_periodic

기능

비동기 방식의 move_periodic 모션으로 비동기 처리 외에 move_periodic() 함수와 동일하게 동작하며 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다. move_periodic() 함수에 의한 모션이 종료되기 전에 발생하는 새로운 모션 명령어는 안전상의 이유로 오류를 발생시킨다. 따라서 amove_periodic() 함수와 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amove_periodic()에 의한 모션이 종료된 것을 확인한 후 새로운 모션명령어가 시작되도록 해야한다.

이 명령어는 현재위치에서 시작하는 상대 모션으로 입력된 기준 좌표계(eMoveReferecne)의 각 축(병진 및 회전)에 대한 Sine함수 기반으로 주기모션을 수행합니다. 각 축 별 모션의 특성은 진폭과 주기에 의해 결정되고 가감속 시간과 총 모션 시간은 주기, 반복 및 횟수에 의해 설정된다.

인수

인수명	자료형	기본값	범위	설명
fAmplitude	float[6]	-	>=0	Amplitude(-fAmplitude에서 +fAmplitude 사이 모션) [mm] or [deg]

인수명	자료형	기본값	범위	설명
fPeriodic	float[6]	-	≥ 0	period(1주기 소요 시간)[sec]
fAccelTim e	float	-	≥ 0	Acc-, dec- time [sec]
nRepeat	unsigned char	-	> 0	반복 횟수
eMoveRef erence	enum.MOVE_ REFERENCE	MOVE_REFERENCE_TOOL	-	상수 및 열거형 정의 참조

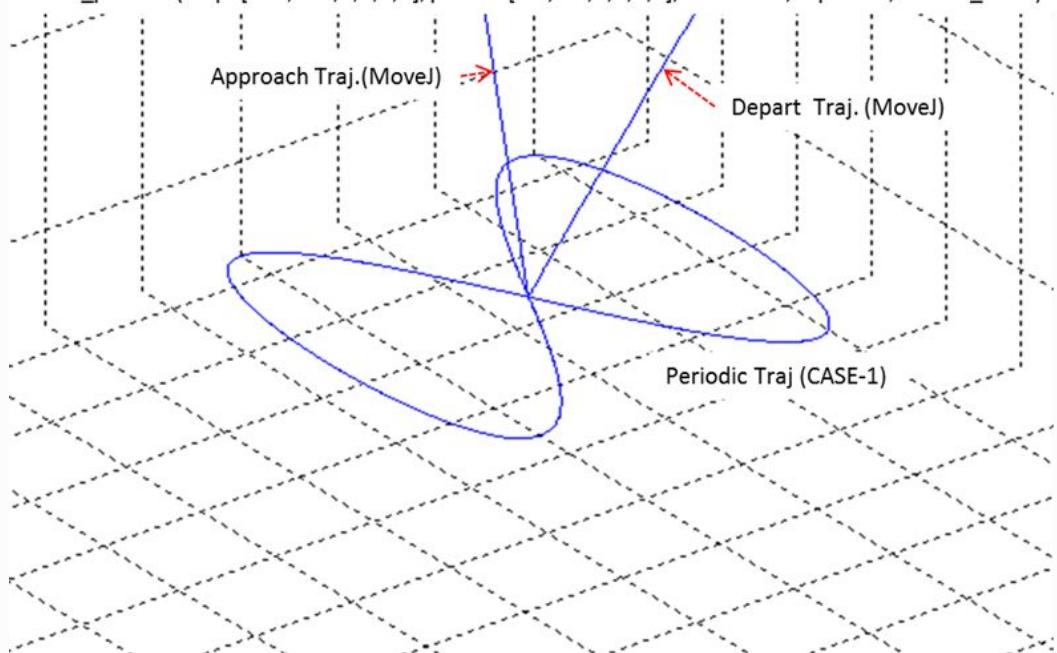
❶ 알아두기

- fAmplitude 는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 6개가 입력되어야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 fAmplitude 를 0으로 입력해야 합니다.
- fPeriodic 는 해당 방향 모션의 1회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 6개가 입력되어야 합니다.
- fAccelTime 은 주기모션의 시작과 끝의 가속 및 감속 시간을 의미합니다. 입력된 가감속시간과 최대주기*1/4 중 큰 값이 적용됩니다. 입력된 가감속 시간이 전체모션시간의 1/2을 초과하는 경우 에러가 발생합니다.
- nRepeat 은 가장 큰 period 값을 가지는 축(기준 축)의 반복 횟수를 정의하며, 이에 따라 총 모션 시간이 결정됩니다. 나머지 축의 반복 횟수는 모션 시간에 따라 자동 결정됩니다.
- 모션이 정상 종료되는 경우 종료 위치가 시작 위치와 일치하게 하도록 나머지 축 모션은 기준 축 모션이 종료되기 전에 먼저 종료될 수 있습니다. 모든 축의 모션이 동시에 종료되지 않는 경우 감속구간

에서의 경로는 이전 경로에서 벗어나게 됩니다. 관련한 사항은 아래 이미지를 참조하십시오

CASE-1) All-axis motions end at the same time

move_periodic(amp=[100,100,0,0,0,0], period=[3.2,1.6,0,0,0,0], atime=3.1, repeat=2, ref=DR_BASE)



- eMoveReference 는 반복 모션의 기준 좌표계를 의미합니다.
- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.
최대속도=진폭(fAmplitude)*2*pi(3.14)/주기(fPeriodic)
(예, 진폭=10mm, 주기=1초인 경우 최대속도=62.83mm/sec)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // Tool좌표계 x축(10mm진폭, 1초 주기)모션과 y회전축(진폭 0.5deg, 1초 주기)
2 // 모션을 총 5회 반복 수행
3 // periodic 모션을 시작하고 1초 후에 Digital_Output 채널1번을 SET(1) 한다.
4 float fAmplitude[NUM_TASK] = {10, 0, 0, 0, 0.5, 0};
5 float fPeriod[NUM_TASK] = { 1, 1, 1, 1, 1, 1};
6 Drfl.amove_periodic(fAmplitude, fPeriod, 0.5, 5, MOVE_REFERENCE_TOOL);
7 Sleep(1000);

```

```

8 DrfL.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);
9 DrfL.mwait();

```

4.6.14 CDRFLEx.stop

기능

로봇제어기에서 수행 중인 모션을 정지시키기 위한 함수이다. 인자로 받는 eStopType에 따라 다르게 정지하며 Estop을 제외한 모든 Stop 모드는 현재 수행하고 있는 구간의 모션을 정지한다

인수

인수명	자료형	기본값	설명
eStopType	enum.STOP_TYPE	STOP_TYPE_QUICK	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 # x1으로 이동 시작 2초 후에 Soft Stop으로 모션 종료
2 float x1[NUM_TASK] = {784, 543, 970, 0, 180, 0};
3 drfL.moveL(x1, 100, 200)           // x1으로 모션 및 즉시 다음 명령 수행
4 Sleep(2000)                      // 2초간 프로그램 일시 중지
5 drfL.stop(STOP_TYPE_SLOW)        // Soft Stop하여 모션 정지

```

4.6.15 CDRFLEx.move_pause

기능

로봇제어기에서 현재 진행중인 로봇의 모션을 감속시켜 일시중지시키기 위한 함수이다. 진행중인 로봇 모션이 없는 경우에는 무시된다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 float j00[NUM_JOINT] = {0, 0, 90, 0, 90,};
2 drfl.amovej(j00, 20, 40); // 비동기모션 시작
3 while (1) {
4     // 현재 관절각도 확인
5     LPPOSITION pPose = drfl.get_current_pose(ROBOT_POSE_JOINT);
6     if (pPose->_fTargetPos[2] >= 45) { // 3축 각도가 45도 이상이면,
7         drfl.move_pause();           // 모션 일시정지
8         Sleep(5000);              // 5초간 기다림
9         break;                   // while문 종료
10    }
11 }
12 drfl.move_resume();           // 모션 재개

```

4.6.16 CDRFLEX.move_resume

기능

로봇제어기에서 move_pause 함수로 일시 중지된 로봇의 모션을 재개하기 위한 함수이다. 진행중인 로봇 경로모션이 없는 경우에는 무시된다.

인수

없음

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

1   float j00[NUM_JOINT] = {0, 0, 90, 0, 90,};
2   drfl.amovej(j00, 20, 40); // 비동기모션 시작
3   while (1) {
4       // 현재 관절각도 확인
5       LPPOSITION pPose = drfl.get_current_pose(ROBOT_POSE_JOINT);
6       if (pPose->_fTargetPos[2] >= 45) { // 3축 각도가 45도 이상이면,
7           drfl.move_pause();           // 모션 일시정지
8           Sleep(5000);             // 5초간 기다림
9           break;                  // while문 종료
10      }
11  }
12  drfl.move_resume();           // 모션 재개

```

4.6.17 CDRFLEx.mwait

기능

로봇제어기에서 선행된 모션명령어의 동작이 종료되기를 기다리기 위한 함수이다. 비동기 모션 명령어와 본 함수를 결합하여 사용하면 동기 모션 명령어와 동일한 동작을 수행할 수 있다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1   float point[6] = { 30, 30, 30, 30, 30, 30 };
2   drfl.amovej(point, 60, 120);

```

3	drfl.mwait();
---	---------------

4.6.18 CDRFLEx.trans

기능

eSourceRef 좌표계 기준으로 정의된 Sourcepos를 동일 좌표계를 기준으로 offset만큼 이동/회전하여 eTargetRef 좌표계 기준으로 변환한 후 반환한다.

인수

인수명	자료형	기본값	설명
fSourcePos	float[6]	-	6개의 Joint Space 정보
fOffset	float[6]	-	6개의 오프셋 정보
eSourceRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
ROBOT_POSE	상수 및 열거형 정의 참조

예제

1	float point[6] = { 30, 30, 30, 30, 30, 30 };
2	float offset[6] = { 100, 100, 100, 100, 100, 100};
3	LPROBOT_POSE res = Drfl.trans(point, offset);
4	for (int i=0; i<6; i++){
5	cout << res->_fPosition[i] << endl;
6	}

4.6.19 CDRFLEx.fkin

기능

조인트 공간에서 조인트 각도(fSourcePos)를 입력받아 eTargetRef좌표계 기준의 TCP의 포즈(태스크 공간의 위치 및 자세)를 반환한다.

인수

인수명	자료형	기본값	설명
fSourcePos	float[6]	-	6개의 Joint Space 정보
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
ROBOT_POSE	상수 및 열거형 정의 참조

예제

```

1 float q1[6] = {0,0,90,0,90,0};
2 Drfl.movej(q1, 60, 30);
3 float q2[6] = {30, 0, 90, 0, 90, 0};
4 LPROBOT_POSE res = Drfl.fkin(q2, COORDINATE_SYSTEM_WORLD);
5 float vel[2] = {100, 100};
6 float acc[2] = {200, 200};
7 float x2[6] = {0,0,0,0,0,0};
8 for(int i=0; i<6; i++){
9     x2[i] = res->_fPosition[i];
10 }
11 Drfl.MoveL(x2, vel, acc);

```

4.6.20 CDRFLEx.ikin

기능

작업 공간 내 기준 좌표계(eTargetRef)의 로봇 포즈에 상응하는 8개의 관절 형상 중 지정한 관절 형상 iSolutionSpace에 해당하는 관절 각도를 반환한다.

인수

인수명	자료형	기본값	설명
fSourcePos	float[6]	-	6개의 Task Space 정보
iSolutionSpace	uchar	-	solution space
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
ROBOT_POSE	상수 및 열거형 정의 참조

예제

```

1 float x1[6] = {370.9, 719.7, 651.5, 90, -180, 0};
2 LPROBOT_POSE res = Drfl.ikin(x1, 2);
3 float q1[6] = {0,};
4 for(int i=0; i<6; i++){
5     q1[i] = res->_fPosition[i];
6 }
7 Drfl.movej(q1, 60, 30);

```

4.6.21 CDRFLEx.set_ref_coord

기능

기준 좌표계를 설정한다.

인수

인수명	자료형	기본값	설명
eTargetCoordSystem	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 float p1[6] = {0, 0, 90, 0, 90, 0};
2 Drfl.movej(p1, 60, 30);
3 float vec[2][3] = {{-1, 1, 1}, {1, 1, 0}};
4 float org[3] = {370.9, -419.7, 651.5};
5
6 int user = Drfl.set_user_cart_coord(vec, org);
7 Drfl.set_ref_coord((COORDINATE_SYSTEM)user);

```

4.6.22 CDRFLEx.check_motion**기능**

현재 진행 중인 모션의 상태를 확인한다.

인수

없음

리턴

값	설명
0	수행 중인 모션이 없음
1	모션 연산 중
2	모션이 수행 중

예제

```
1 float q0[6] = {0, 0, 90, 0, 90, 0};
```

```

2   float q99[6] = {0, 0, 0, 0, 0, 0};
3   Drfl.amovej(q0, 60, 30); //q0로 모션 및 즉시 다음명령 수행
4   while(true)
5   {
6     if(Drfl.check_motion() == 0) // 모션이 완료된 경우
7     {
8       Drfl.movej(q99, 60, 30); //q99로 조인트 모션
9       break;
10    }
11 }

```

4.6.23 CDRFLEx.enable.Alter_motion

기능

경로 수정 기능을 활성화 한다. 경로 생성의 단위 주기는 100msec이며 입력 인자 n을 설정하여 경로 생성 주기 ($n * 100\text{msec}$)를 변경 할 수 있다. 입력 인자 ePathMode를 통해 alter_motion의 입력값의 의미를 2가지 모드(누적량 모드, 증분량 모드) 중 하나로 선택하여 사용 할 수 있다. 누적량 모드의 경우 현재의 모션경로에 대한 절대적 증분위치/자세만큼 경로 수정량이 반영된다. 증분량 모드의 경우 바로 현재의 절대적 증분위치/자세에 입력된 증분위치/자세만큼 경로 수정량이 추가되어 반영된다. 입력 인자 eTargetRef를 통해 기준 좌표계를 설정할 수 있다. 입력 인자 fLimitDpos, fLimitDposPer를 통해 각각 누적량, 증분량의 한계치를 설정 할 수 있다. 한계치를 벗어나는 위치 값의 한계치에 수렴한 값으로 경로 수정량이 재 조정된다.

해당 함수는 M2.4 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
iCycleTime	int	-	경로 생성 주기
ePathMode	Int	-	경로 수정 모드
eTargetRef	int	-	상수 및 열거형 정의 참조
fLimitDpos	float[2]	-	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]
fLimitDposPer	float[2]	-	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]

● 알아두기

- alter_motion은 사용자 thread 내에서만 동작한다.

- eTargetRef가 None인 경우, global coordinate 적용(global coordinate초기값은 COORDINATE_SYSTEM_BASE이며, set_ref_coord 명령에 의해 설정 가능)

리턴

값	설명
0	오류
1	성공

예제

```

1 float limit_dPOS[2] = {50, 90};
2 float limit_dPOS_per[2] = {50, 50};
3 Drfl.enable_alter_motion(5, PATH_MODE_DPOS, COORDINATE_SYSTEM_BASE,
limit_dPOS, limit_dPOS_per);

```

4.6.24 CDRFLEx.alter_motion

기능

입력 인자 pos에 해당하는 양만큼 경로 수정을 진행한다.

해당 함수는 M2.4 버전 이상에서만 사용 가능하다.

⚠ 주의

- alter_motion은 사용자 thread 내에서만 동작한다.

ℹ 알아두기

- alter_motion은 enable_alter_motion을 통해 경로보정기능이 활성화 된 경우에만 유효하다.
- enable_alter_motion의 설정 값 fLimitPos, fLimitPosPer에 따라 경로 수정량은 조정될 수 있다.
- pos의 방향값은 Fixed XYZ로 설정하여야 한다.

인수

인수명	자료형	기본값	설명
pos	float[6]	-	position list

리턴

값	설명
0	오류
1	성공

예제

```

1  DWORD WINAPI ThreadFunc(void *arg){
2      while(true){
3          float pos[6] = {10, 0, 0, 10, 0, 0};
4          Drfl.alter_motion(pos);
5      }
6  }
...
8  int _tmain (int argc, _TCHAR* argv[]){
9      HANDLE hThread;
10     DWORD dwThreadID;
11     hThread = CreateThread(NULL, 0, ThreadFunc, NULL, 0, &dwThreadID);
12     if (hThread == 0) {
13         printf("Thread Error\n");
14         return 0;
15     }
16     float limit_dPOS[2] = {50, 90};
17     float limit_dPOS_per[2] = {50, 50};
18     Drfl.enable_alter_motion(5, PATH_MODE_DPOS, COORDINATE_SYSTEM_BASE,
19     limit_dPOS, limit_dPOS_per);
}
```

4.6.25 CDRFLEEx.disable_alter_motion

기능

경로 수정 기능을 비활성화 한다.

해당 함수는 M2.4 버전 이상에서만 사용 가능하다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1  DWORD WINAPI ThreadFunc(void *arg){
2      while(true){
3          float pos[6] = {10, 0, 0, 10, 0, 0};
4          Drfl.alter_motion(pos);
5      }
6  }
7  ...
8  int _tmain (int argc, _TCHAR* argv[]){
9      HANDLE hThread;
10     DWORD dwThreadID;
11     hThread = CreateThread(NULL, 0, ThreadFunc, NULL, 0, &dwThreadID);
12     if (hThread == 0) {
13         printf("Thread Error\n");
14         return 0;
15     }
16     float limit_dPOS[2] = {50, 90};
17     float limit_dPOS_per[2] = {50, 50};
18     Drfl.enable_alter_motion(5, PATH_MODE_DPOS, COORDINATE_SYSTEM_BASE,
19     limit_dPOS, limit_dPOS_per);
20     Drfl.disable_alter_motion();
}

```

4.6.26 CDRFLEx.servoj

기능

비동기 방식의 모션 명령어로 모션 시작과 동시에 다음 명령어를 수행한다. 연속적으로 전달되는 명령 중 가장 최근의 목표 관절 위치를 최대 속도, 가속도 내에서 추종하는 모션이다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목표 관절 위치
fLimitVel	float[6]	-	6개 축에 대한 최대 속도
fLimitAcc	float[6]	-	6개 축에 대한 최대 가속도
fTargetTime	float	-	도달 시간 [sec]
eTargetMod	enum.DR_SERVOJ_TYPE	DR_SERVO_OVERRIDE	상수 및 열거형 정의 참조

● 알아두기

- fTargetTime이 입력된 경우 최대 속도/가속도 조건에 의해 도달 시간을 지킬 수 없는 경우 자동 조정하고 알림 메시지를 띄웁니다.

⚠ 주의

- 현재는 change_operation_speed 함수의 작동 속도 조절 기능과 연동되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float q0[6] = { 0, 0, 90, 0, 90, 0 };
2 float jvel[6]={10, 10, 10, 10, 10, 10};
3 float jacc[6]= {10, 10, 10, 10, 10, 10};
4

```

5	drfl.servoj(q0, jvel, jacc, -10000);
---	--------------------------------------

4.6.27 CDRFLEx.servol

기능

비동기 방식의 모션 명령어로 모션 시작과 동시에 다음 명령어를 수행한다. 연속적으로 전달되는 명령 중 가장 최근의 목표 위치를 최대 속도, 가속도 내에서 추종하는 모션이다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목표 관절 위치
fTargetVel	float[2]	-	속도
fTargetAcc	float[2]	-	가속도
fTargetTime	float	-	도달 시간 [sec]

➊ 알아두기

- fTargetTime이 입력된 경우 최대 속도/가속도 조건에 의해 도달 시간을 지킬 수 없는 경우 자동 조정하고 알림 메시지를 띄웁니다.

⚠ 주의

- 현재는 change_operation_speed 함수의 작동 속도 조절 기능과 연동되지 않습니다.
- 현재는 singularity 처리 옵션 중 DR_VAR_VEL 옵션과 연동되지 않습니다. DR_VAR_VEL 옵션으로 설정 시 자동으로 DR_AVOID옵션으로 변경하고 알림 메시지를 띄웁니다.
- 현재는 힘/강성 제어 명령어와 연동되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float q0[6] = { 368, 34.5, 442, 50, -180, 50 };
2 float jvel[2]={10, 10};
3 float jacc[2]={20, 20};
4
5 drfl.servol(q0, jvel, jacc, -10000);

```

4.6.28 CDRFLEx.speedj

기능

비동기 방식의 모션 명령어로 모션 시작과 동시에 다음 명령어를 수행한다. 연속적으로 전달되는 명령 중 가장 최근의 목표 관절 속도를 설정한 최대 속도, 가속도 내에서 추종하는 모션이다.

인수

인수명	자료형	기본값	설명
fTargetVel	float[6]	-	속도
fTargetAcc	float[6]	-	가속도
fTargetTime	float	-	도달 시간 [sec]

❶ 알아두기

- fTargetTime이 입력된 경우 최대 속도/가속도 조건에 의해 도달 시간을 지킬 수 없는 경우 자동 조정하고 알림 메시지를 띄웁니다.
- 속도가 있는 상태에서 정상적으로 정지하고 싶은 경우에는, vel을 [0,0,0,0,0,0]으로 입력하거나 stop 명령어를 활용합니다.
- 안전을 위하여 이동 중1 [sec] 동안 새로운 speedj 명령이 전달되지 않으면, 에러 메시지를 띄우고 정지합니다.

⚠ 주의

- 현재는 change_operation_speed 함수의 작동 속도 조절 기능과 연동되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```
float jvel={10, 10, 10, 10, 10, 10};
float jacc={20, 20, 20, 20, 20, 20};

drfl.speedj(jvel, jacc, -10000);
```

4.6.29 CDRFLEx.speedl

기능

비동기 방식의 모션 명령어로 모션 시작과 동시에 다음 명령어를 수행한다. 연속적으로 전달되는 명령 중 가장 최근의 목표 위치를 최대 속도, 가속도 내에서 추종하는 모션이다.

인수

인수명	자료형	기본값	설명
fTargetVel	float[6]	-	속도
fTargetAcc	float[2]	-	가속도
fTargetTime	float	-	도달 시간 [sec]

● 알아두기

- fTargetTime이 입력된 경우 최대 속도/가속도 조건에 의해 도달 시간을 지킬 수 없는 경우 자동 조정하고 알림 메시지를 띄웁니다.
- 안전을 위하여 이동 중1 [sec] 동안 새로운 speedl 명령이 전달되지 않으면, 에러를 발생시키고 정지합니다.

⚠ 주의

- 현재는 change_operation_speed 함수의 작동 속도 조절 기능과 연동되지 않습니다.
- 현재는 singularity 처리 옵션 중 DR_VAR_VEL 옵션과 연동되지 않습니다. DR_VAR_VEL 옵션으로 설정 시 자동으로 DR_AVOID 옵션으로 변경하고 알림 메시지를 띄웁니다.
- 현재는 힘/강성 제어 명령어와 연동되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float jvel={10, 10, 10, 10, 10, 10};
2 float jacc={20, 20};
3
4 drfl.speedl(jvel, jacc, -10000)

```

4.6.30 CDRFLEx.moveb

기능

로봇 제어기에서 하나 이상의 라인 또는 원호 구성된 경로 정보를 받아 로봇을 경로 정보에 설정된 blending radius로 블렌딩하여 등속으로 이동시키기 위한 함수이다.

인수

인수명	자료형	기본값	설명
tTargetPos	struct MOVE_POSB [MAX_MOVEB_POI NT]	-	최대 25개까지의 경로 정보
nPosCount	unsigned char	-	유효 경로 정보 개수

인수명	자료형	기본값	설명
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE_E	MOVE_MODE_ABSOLUTE	상수 및 열거형 상수 참조
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_BASE	상수 및 열거형 상수 참조
eAppType	enum.DR_MV_APP	DR_MV_APP_NONE	application 사용 여부



알아두기

- fTargetVel에 하나의 인자를 입력한 경우(예를들어, fTargetVel =[30, 0]) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를들어, fTargetAcc =[60, 0]) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 posb list의 각 pos는 앞 선 pos에 대한 상대좌표로 정의됩니다.



주의

- posb에서 blending radius가 0인 경우, 사용자 입력 오류가 나타납니다.
- 연속된 Line-Line segment가 같은 방향을 가질 경우 Line의 중복입력으로 사용자 입력 오류가 나타납니다.
- 블렌딩 구간에서 조건에 따라 급격하게 방향전환이 발생하게 되는 경우 급가속을 방지하기 위해 사용자 입력오류가 나타납니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

MOVE_POSB xb[4];
memset(xb, 0x00, sizeof(xb));
int segNum = 4;
float tvel = { 50, 50 };
float tacc = { 100, 100 };
xb[0]._iBlendType = 0; // line
xb[0]._fBlendRad = 50;
xb[0]._fTargetPos[0][0] = 559;
xb[0]._fTargetPos[0][1] = 234.5;
xb[0]._fTargetPos[0][2] = 651.5;
xb[0]._fTargetPos[0][3] = 0;
xb[0]._fTargetPos[0][4] = 180;
xb[0]._fTargetPos[0][5] = 0;
xb[1]._iBlendType = 1; // circle
xb[1]._fBlendRad = 50;
xb[1]._fTargetPos[0][0] = 559;
xb[1]._fTargetPos[0][1] = 234.5;
xb[1]._fTargetPos[0][2] = 451.5;
xb[1]._fTargetPos[0][3] = 0;
xb[1]._fTargetPos[0][4] = 180;
xb[1]._fTargetPos[0][5] = 0;
xb[1]._fTargetPos[1][0] = 559;
xb[1]._fTargetPos[1][1] = 434.5;
xb[1]._fTargetPos[1][2] = 451.5;
xb[1]._fTargetPos[1][3] = 0;
xb[1]._fTargetPos[1][4] = 180;
xb[1]._fTargetPos[1][5] = 0;
xb[2]._iBlendType = 0; // line
xb[2]._fBlendRad = 50;
xb[2]._fTargetPos[0][0] = 559;
xb[2]._fTargetPos[0][1] = 434.5;
xb[2]._fTargetPos[0][2] = 251.5;
xb[2]._fTargetPos[0][3] = 0;
xb[2]._fTargetPos[0][4] = 180;
xb[2]._fTargetPos[0][5] = 0;
xb[3]._iBlendType = 0; // line
xb[3]._fBlendRad = 50;
xb[3]._fTargetPos[0][0] = 559;

```

```

xb[3]->_fTargetPos[0][1] = 234.5;
xb[3]->_fTargetPos[0][2] = 251.5;
xb[3]->_fTargetPos[0][3] = 0;
xb[3]->_fTargetPos[0][4] = 180;
xb[3]->_fTargetPos[0][5] = 0;
drfl.moveb(xb, segNum, tvel, tacc);
    // 현재위치에서 시작하여 LINE→CIRCLE→LINE→LINE으로 이루어진 궤적을
    // 속도 50, 50(mm/sec, deg/sec), 가속도 100, 100(mm/sec2, deg/sec2)
    // 유지하며 각 구간의 끝점에서 50mm 거리에 도달하면 다음 segment로
    // blending이 시작됨

```

4.6.31 CDRFLEx.ikin(Extension)

기능

작업 공간 내 기준 좌표계(eTargetRef)의 로봇 포즈에 상응하는 8개의 관절 형상 중 지정한 관절 형상 iSolutionSpace에 해당하는 관절 각도 및 이동 가능한 공간 여부를 반환한다.

인수

인수명	자료형	기본값	설명
fSourceP os	float[6]	-	6개의 Task Space 정보
iSolution Space	uchar	-	solution space
eTargetR ef	enum.COORDINAT E_SYSTEM	COORDINATE_SYS TEM_BASE	상수 및 열거형 정의 참조
iRefPosO pt	uchar	-	[iRefPosOpt] 멀티턴 해 생성 시 옵션에 따른 기준 위치에 가까운 해 선택 0: posj(0,0,0,0,0,0) 위치 기준 1: 현재 조인트 각도 위치 기준

리턴

값	설명
INVERSE_KINEMATIC _RESPONSE	구조체 정의 참조

예제

```

float x1[6] = {370.9, 719.7, 651.5, 90, -180, 0};
LPINVERSE_KINEMATIC_RESPONSE res = Drfl.ikin(x1, 2, COORDINATE_BASE, 0);
float q1[6] = {0,};
for(int i=0; i<6; i++){
    q1[i] = res->_fPosition[i];
}
Drfl.movej(q1, 60, 30);

```

4.6.32 CDRFLEX.movejx

기능

로봇제어기에서 로봇을 관절 공간 안에서 목표 위치로 이동시키기 위한 함수이다. 목표 위치는 작업공간 상의 위치임으로 movej과 동일하게 이동하지만 로봇의 모션은 관절공간에서 이루어지기 때문에 목표 위치까지 직선경로가 보장되지 않는다. 추가적으로 하나의 작업공간좌표에 대응하는 8가지의 관절조합형태(robot configuration)중 하나를 iSolutionSpace(solution space)에 지정해야 한다. iSolutionSpace에 255를 지정할 시, 8가지의 관절조합형태 중, 현재 robot configuration과 관절 공간 상에서 가장 가까운 (2-5축의 관절 공간의 L2 norm이 가장 작음) Solution Space로 이동한다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목적 TCP 위치
iSolutionSpace	unsigned char	-	관절조합형태(아래 설명 참조)
fTargetVel	float/float[6]	-	속도
fTargetAcc	float/float[6]	-	가속도
fTargetTime	float	0.f	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조

인수명	자료형	기본값	설명
fBlendingRadius	Float	0.f	blending radius
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조



알아두기

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리된다.
- 상대모션으로 입력하는 경우(eMoveMode = MOVE_MODE_ABSOLUTE), 선행모션에 블렌딩을 사용하는 경우 에러가 발생하므로 movej() 또는 MoveL()을 이용하여 블렌딩하는 것을 권장한다.
- 옵션 eBlendingType 및 fTargetVel / fTargetAcc 에 따른 블렌딩을 수행할 경우 movej(), MoveL() 설명을 참조하십시오.

Robot configuration (형태 vs. solution space)

Solution space	Binary	Shoulder	Elbow	Wrist
0	000	Lefty	Below	No Flip
1	001	Lefty	Below	Flip
2	010	Lefty	Above	No Flip
3	011	Lefty	Above	Flip
4	100	Righty	Below	No Flip
5	101	Righty	Below	Flip
6	110	Righty	Above	No Flip
7	111	Righty	Above	Flip
255	Auto Calculation (the smallest L2 norm in the joint space of axes 2-5)			

리턴

값	설명
0	오류
1	성공

예제

```
// CASE 1
float x1[6] = { 559, 34.5, 651.5, 0, 180, 0 };
float sol=2;
float jvel=10;
float jacc=20;
drfl.movejx(x1, sol, jvel, jacc);
    // 속도 10(deg/sec), 가속도 20(deg/sec2)으로 x1 및 configuration과 일치하는 관절각으로 이동

// CASE 2
float x1[6] = { 559, 34.5, 651.5, 0, 180, 0 };
float sol=2;
float jTime=5;
drfl.movejx(x1, sol, 0, 0, jTime);
    // x1 및 configuration과 일치하는 관절각까지 5초의 도착시간을 가지고 이동

// CASE 3
float x1[6] = { 559, 34.5, 651.5, 0, 180, 0 };
float x2[6] = { 559, 434.5, 651.5, 0, 180, 0 };
float sol=2;
float jvel=10;
float jacc=20;
float blending_radius=50;
drfl.movejx(x1, sol, jvel, jacc, 0, MOVE_MODE_ABSOLUTE, blending_radius);
    // x1 및 configuration과 일치하는 관절각으로 이동하며 x1 위치로부터 50mm
    // 의 거리가 될 때 다음 모션을 수행하도록 설정
drfl.movejx(x2, sol, jvel, jacc, 0, MOVE_MODE_ABSOLUTE, 0,
BLENDING_SPEED_TYPE_DUPLICATE);
    // 직전모션과 Blending되어 x2 및 configuration과 일치하는 관절각으로 이동
```

4.6.33 CDRFLEx.amove_spiral(Extension)

기능

비동기 방식의 move_spiral모션으로 비동기 처리 외에는 move_spiral() 함수와 동일하게 동작하며 모션 종료를 기다리지 않고 모션 시작과 동시에 리턴하여 다음 라인을 실행한다.

amove_spiral() 함수에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킨다. 따라서 amove_spiral() 함수과 이어지는 새로운 모션명령어 사이에는 mwait() 함수를 사용하여 amove_spiral()에 의한 모션이 종료된 것을 확인한 후 새로운 모션명령어가 시작되도록 해야한다.

본 명령은 방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축방향으로 병행하는 모션을 수행합니다. 현재 위치에서 기준 좌표계(eMoveReference) 지정한 좌표계 상의 축 방향에 수직인 평면에서의 나선궤적과 축 방향으로의 직선궤적을 동시에 따라 이동한다.

인수

인수명	자료형	기본값	범위	설명
eTaskAxis	enum.TASK_AXI S	-	-	상수 및 열거형 정의 참조
fRevolution	float	-	rev > 0	총 회전수 [revolution]
fTargetPos	float[3]	-	-	목표점
fTargetVel	float[2]	-		선속도, 각속도
fTargetAcc	float[2]	-		선가속도, 각가속도
fTargetTime	float	0.0	time \geq 0	총 수행시간 <sec>
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_TOOL		상수 및 열거형 정의 참조
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE		상수 및 열거형 정의 참조
eSpiralDir	enum.SPIRAL_DIR	DR_SPIRAL_OUTWARD		상수 및 열거형 정의 참조
eRotDir	enum.ROT_DIR	DR_ROT_FORWARD		상수 및 열거형 정의 참조

● 알아두기

- fRevolution 는 spiral 모션의 총 회전수를 의미합니다.
- fTargetVel 은 spiral 모션의 이동 속도를 의미합니다.
- fTargetAcc 는 spiral 모션의 이동 가속도를 의미합니다.

- fTargetTime 을 지정할 경우 fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eTaskAxis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- eMoveReference 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다.
이 경우 fTargetVel, fTargetAcc 또는 fTargetTime 값을 작게 조정하는 것을 권장합니다.

리턴

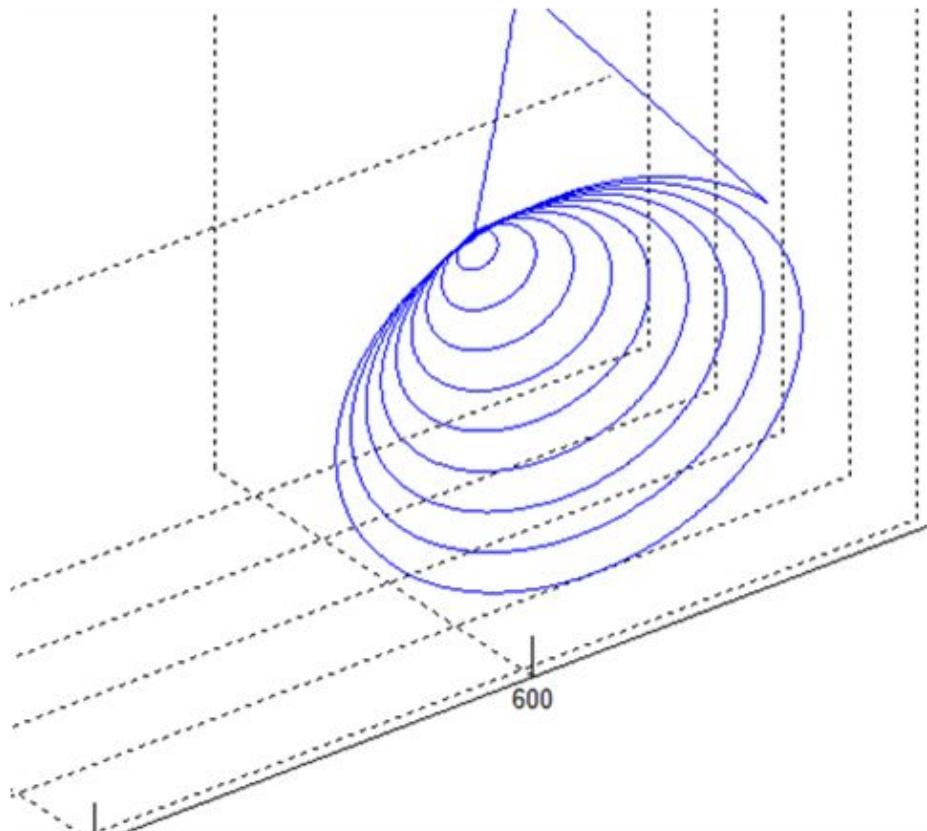
값	설명
0	오류
1	성공

예제

```

1 // 나선형 모션 시작 후 3초 후에 D-Out
2 float rev = 3;
3 float rmax = 50;
4 float lmax = 50;
5 float tvel = { 50, 50 };
6 float tacc = { 100, 100 };
7 Drfl.amove_spiral(TASK_AXIS_Z, rev, rmax, lmax, tvel, tacc);
8 Sleep(3000);
9 drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```



4.6.34 CDRFLEx.movel

기능

로봇제어기에서 로봇을 작업 공간 안에서 목표 위치(pos)로 직선을 따라 이동시키기 위한 함수이다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	6개 축에 대한 목적 TCP 위치
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도
fTargetTime	float	0.f	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리

인수명	자료형	기본값	설명
eMoveMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조
fBlendingRadius	float	0.f	blending시 radius
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조
eAppType	enum.DRMV_APP	DRMV_APP_NONE	application 타입 설정



알아두기

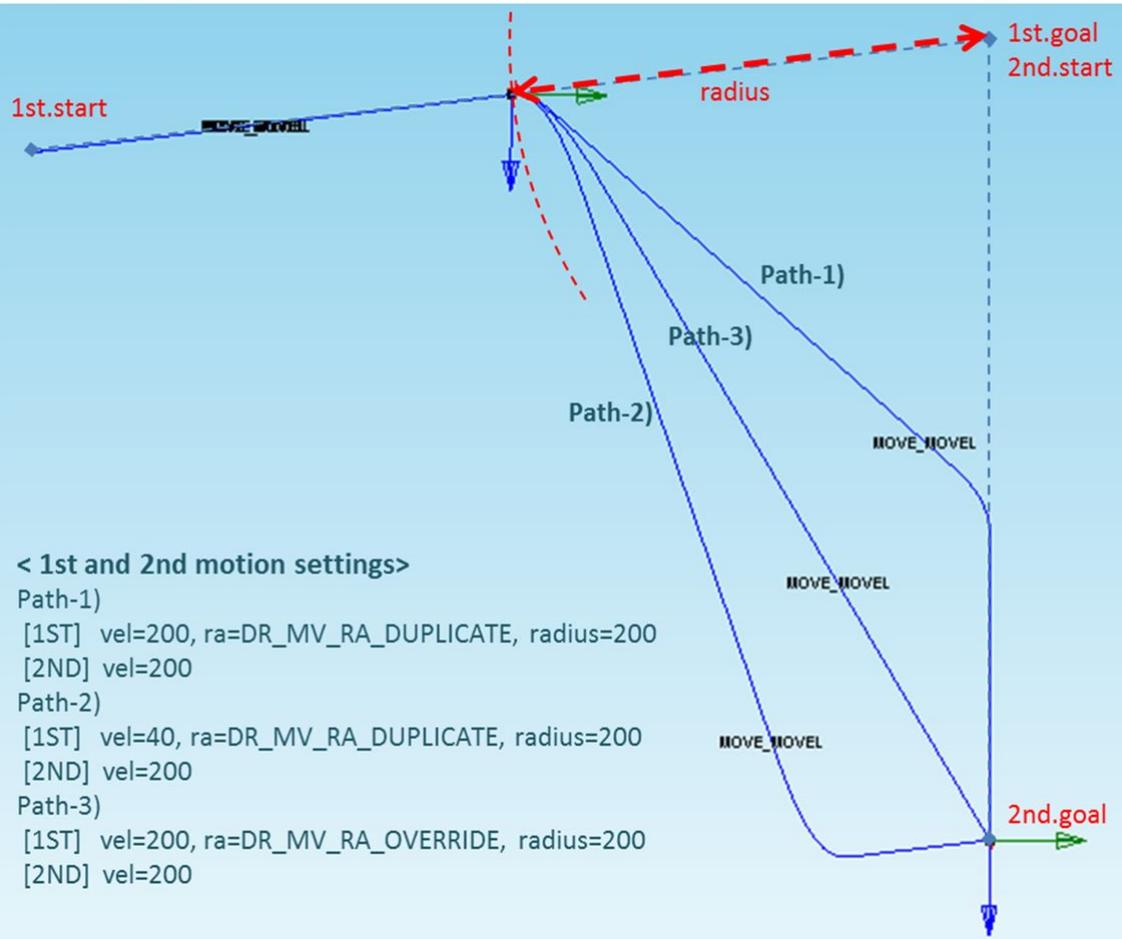
- fTargetVel에 하나의 인자를 입력한 경우(예를들어, fTargetVel = {30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를들어, fTargetAcc = {60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.



주의

eBlendingType이 BLENDING_SPEED_TYPE_DUPLICATE이고 fBlendingRadius 가 0보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간 보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 참고 하십시오.

<(Example) Path differences accord. to 1st and 2nd motion settings>



리턴

값	설명
0	오류
1	성공

예제

```
// CASE 1
float x1[6] = { 559, 434.5, 651.5, 0, 180, 0 };
float tvel = { 50, 50 };
```

```

float tacc = { 100, 100 };
drfl.movvel(x1, tvel, tacc);
// 속도 50(mm/sec), 가속도 100(mm/sec2)로 x1 위치로 이동

// CASE 2
float x1[6] = { 559, 434.5, 651.5, 0, 180, 0 };
float tTime = 5;
drfl.movvel(x1, 0, 0, tTime);
// x1 위치로 5초의 도착시간을 가지고 이동

// CASE 3
float x1[6] = { 559, 434.5, 651.5, 0, 180, 0 };
float tvel = 50;
float tacc = 100;
drfl.movvel(x1, tvel, tacc, 0, MOVE_MODE_RELATIVE, MOVE_REFERENCE_TOOL);
// 시작위치에서 Tool 좌표계 기준으로 x1 만큼의 상대위치로 이동
float x1[6] = { 559, 434.5, 651.5, 0, 180, 0 };
float x2[6] = { 559, 434.5, 251.5, 0, 180, 0 };
float tvel = 50;
float tacc = 100;
float blending_radius = 100;
drfl.movvel(x1, tvel, tacc, 0, MOVE_MODE_ABSOLUTE, MOVE_REFERENCE_BASE,
blending_radius);

```

4.6.35 CDRFLEx.movec

기능

로봇 제어기에서 작업공간을 기준으로 로봇이 현재 위치에서 경유 지점을 지나 목표 위치까지 원호 또는 지정한 각도로 원호를 따라 이동시키기 위한 함수이다.

인수

인수명	자료형	기본값	설명
fTargetPos[0]	float[6]	-	경유 지점
fTargetPos[1]	float[6]		목표 위치
fTargetVel	float[2]	-	선속도, 각속도
fTargetAcc	float[2]	-	선가속도, 각가속도

인수명	자료형	기본값	설명
fTargetTime	float	0.0	도달 시간 [sec]
eMoveMode	enum.MOVE_MODE_E	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eMoveReference	enum.MOVE_REFERENCE_E	MOVE_REFERENCE_BASE	상수 및 열거형 정의 참조
fTargetAngle1	float	0.0	angle1
fTargetAngle2	float	0.0	angle2
fBlendingRadius	float	0.0	blending 시 radius
eBlendingType	enum.BLENDING_SPEED_TYPE	BLENDING_SPEED_TYPE_DUPLICATE	상수 및 열거형 정의 참조
eAppType	enum.DR_MV_APP_P	DR_MV_APP_NONE	application 타입 설정



알아두기

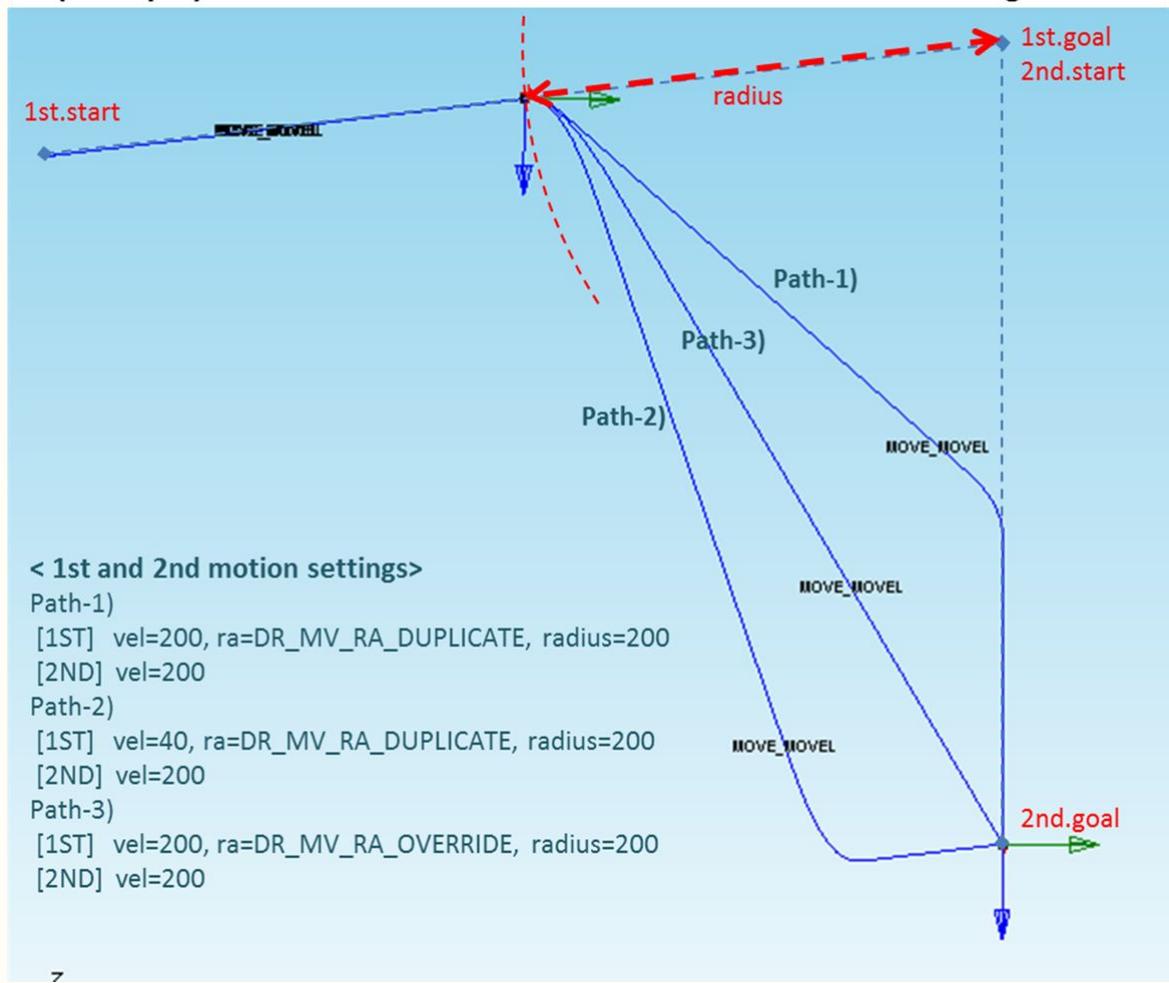
- fTargetVel에 하나의 인자를 입력한 경우(예를 들어, fTargetVel =[30, 0]) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- fTargetAcc에 하나의 인자를 입력한 경우(예를 들어, fTargetAcc =[60, 0]) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- fTargetTime 지정 시, fTargetVel, fTargetAcc 를 무시하고 fTargetTime 기준으로 처리됩니다.
- eMoveMode 가 MOVE_MODE_RELATIVE 인 경우 fTargetPos[0] 과 fTargetPos[1] 는 각각 앞 선 위치값에 대한 상대좌표로 정의됩니다. (fTargetPos[0]은 시작점 대비 상대좌표, fTargetPos[1]는 fTargetPos[0]대비 상대좌표)
- fTargetAngle1이 0보다 크고, fTargetAngle2이 0인 경우 fTargetAngle1은 Circular path 상의 총 회전각이 적용됩니다.
- fTargetAngle1과 fTargetAngle2가 0보다 큰 경우, fTargetAngle1은 circular path 상에서 정속으로 이동하는 총 회전각을, fTargetAngle2는 가속과 감속을 위한 회전 구간의 회전각을 의미합니다. 이 때 총 이동각은 fTargetAngle1+ 2 X fTargetAngle2만큼 circular path 상을 움직입니다.



주의

eBlendingType이 BLENDING_SPEED_TYPE_DUPLICATE이고 fBlendingRadius가 0보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간 보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 참고 하십시오.

<(Example) Path differences accord. to 1st and 2nd motion settings>



리턴

값	설명
0	오류
1	성공

예제

```
// CASE 1
float x1[2][6] = {{559,434.5,651.5,0,180,0}, {559,434.5,251.5,0,180,0}};
float tvel = {50,50}; # 태스크 속도를 50(mm/sec), 50(deg/sec)로 설정
float tacc = {100,100}; # 태스크 가속도를 100(mm/sec2), 100(deg/sec2)로 설정
drfl.movec(x1, tvel, tacc);
// 속도 50(mm/sec), 가속도 100(mm/sec2)로 x1[0]을 경유하여 x1[1]에 이르는
// 원호궤적을 따라 이동

// CASE 2
float x1[2][6] = {{559,434.5,651.5,0,180,0},{559,434.5,251.5,0,180,0}};
float tTime = 5;
drfl.movec(x1, 0, 0, tTime);
// x1[0]을 경유하여 x1[1]까지 5초의 도착시간을 가지고 원호궤적으로 이동

// CASE 3
float x1[2][6] = {{559,434.5,651.5,0,180,0},{559,434.5,251.5,0,180,0}};
float x2[2][6] = {{559,234.5,251.5,0,180,0 },{559,234.5,451.5,0,180,0}};
float tvel = {50,50};
float tacc = {100,100};
float blending_radius = 50;
drfl.movec(x1, tvel, tacc, 0, MOVE_MODE_ABSOLUTE, MOVE_REFERENCE_BASE, 0, 0,
blending_radius);
drfl.movec(x2, tvel, tacc, 0, MOVE_MODE_ABSOLUTE, MOVE_REFERENCE_BASE, 0, 0, 0,
BLENDING_SPEED_TYPE_DUPLICATE);
```

4.7 로봇 설정 함수

4.7.1 CDRFLEEx.add_tool

기능

로봇 끝단에 장착될 Tool 정보를 안전상 사전에 등록하여 사용하기 위한 함수이다. 본 함수를 이용하여 등록된 Tool 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능하다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	Tool 이름
fCog	float[3]	-	무게 중심

인수명	자료형	기본값	설명
fWeight	Float	-	툴 무게
fInertia	float[6]	-	관성 정보

리턴

값	설명
0	오류
1	성공

예제

```

1 float fCog[3] = {10, 10, 10};
2 float finertia[6] = { 0, 0, 0, 0, 0, 0 };
3 // Tool 등록
4 drfl.add_tool("tool#1", 5.3f, fCog, finertia);
5
6 // 현재 장착된 Tool 선택
7 drfl.set_tool("tool#1");
8
9 // Tool 등록 해제
10 drfl.del_tool("tool#1");

```

4.7.2 CDRFLEx.del_tool

기능

로봇 제어기에 사전에 등록된 Tool 정보를 삭제하기 위한 함수이다

인수

인수명	자료형	기본값	설명
strSymbol	string	-	Tool 이름

리턴

값	설명
0	오류
1	성공

예제

```

1 float fCog[3] = {10, 10, 10};
2 float finertia[6] = { 0, 0, 0, 0, 0, 0 };
3 // Tool 등록
4 drfl.add_tool("tool#1", 5.3f, fCog, finertia);
5
6 // 현재 Tool 선택
7 drfl.set_tool("tool#1");
8
9 // Tool 등록 해제
10 drfl.del_tool("tool#1");

```

4.7.3 CDRFLEx.set_tool

기능

로봇 제어기에 사전에 등록되어 있는 Tool 정보 중 현재 장착된 Tool에 대한 정보를 설정하는 함수이다. 현재 장착된 Tool이 없을 경우, 빈 문자열을 전달하면 현재 설정되어 있는 정보가 초기화된다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	Tool 이름

⚠ 주의

- set_tool, set_workpiece_weight 함수를끼리 연달아 사용하는 경우에는 transition_time 만큼 wait(transition_time) 함수를 사이에 넣어서 사용해야 합니다. 그렇지 않으면 무게 변경에 오류가 생길 수 있습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float fCog[3] = {10, 10, 10};
2 float finertia[6] = { 0, 0, 0, 0, 0, 0 };
3 // Tool 등록
4 drfl.add_tool("tool#1", 5.3f, fCog, finertia);
5
6 // 현재 Tool 선택
7 drfl.set_tool("tool#1");
8
9 // Tool 등록 해제
10 drfl.del_tool("tool#1");

```

4.7.4 CDRFLEx.get_tool

기능

로봇 제어기에서 현재 설정된 Tool 정보를 가져오는 함수이다. 설정된 Tool 정보가 없을 경우, 빈 문자열이 반환된다.

인수

없음

리턴

값	설명
string	Tool 이름

예제

```

1 string strTool = drfl.get_tool();
2 // 현재 장착되어 있는 Tool 정보 확인
3 if (strTool != "tool#1) {

```

```

4 drfl.set_tool("tool#1);
5 }
```

4.7.5 CDRFLE.add_tcp

기능

로봇 TCP 정보를 안전상 사전에 등록하여 사용하기 위한 함수이다. 본 함수를 이용하여 등록된 TCP 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능하다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	TCP 이름
fPosition	float[6]	-	TCP 정보

리턴

값	설명
0	오류
1	성공

예제

```

1 float fTCP[6] = { 10, 10, 10, 0, 0, 0 };
2 // TCP 등록
3 drfl.add_tcp("tcp#1", fTCP);
4
5 // 현재 TCP 설정
6 drfl.set_tcp("tcp#1");
7
8 // TCP 등록 해제
9 drfl.del_tcp("tcp#1");
```

4.7.6 CDRFLEEx.del_tcp

기능

로봇 제어기에 사전에 등록된 TCP 정보를 삭제하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	TCP 이름

리턴

값	설명
0	오류
1	성공

예제

```

1 float fTCP[6] = { 10, 10, 10, 0, 0, 0 };
2 // TCP 등록
3 drfl.add_tcp("tcp#1", fTCP);
4
5 // 현재 TCP 설정
6 drfl.set_tcp("tcp#1");
7
8 // TCP 등록 해제
9 drfl.ConfigDelete("tcp#1");

```

4.7.7 CDRFLEEx.set_tcp

기능

로봇 제어기에 사전에 등록되어 있는 TCP 정보 중 현재 장착된 TCP에 대한 정보를 설정하는 함수이다. 현재 장착된 TCP가 없을 경우, 빈 문자열을 전달하면 현재 설정되어 있는 정보가 초기화된다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	Tool 이름

리턴

값	설명
0	오류
1	성공

예제

```

1 drfl.set_tcp("tcp#1");
2
3 float point[6] = { 756.6f, 511.6f, 876.0f, 44.5f, 86.7f, 55.7f };
4 float vel[2] = {30, 0 };
5 float acc[2] = {30, 0 };
6
7 drfl.MoveL(point, vel, vel);

```

4.7.8 CDRFLEEx.get_tcp**기능**

로봇 제어기에서 현재 설정된 TCP 정보를 가져오는 함수이다. 설정된 Tool 정보가 없을 경우, 빈 문자열이 반환된다.

인수

없음

리턴

값	설명
string	TCP 이름

예제

```

1 string strTCP = drfl.get_tcp();
2 // 현재 장착되어 있는 TCP 정보 확인
3 if (strTCP != "tcp#1") {
4     drfl.set_tcp("tcp#1");
5 }
```

4.7.9 CDRFLEEx.set_tool_shape

기능

로봇 제어기에 사전에 등록되어 있는 ToolShape 정보 중 현재 장착된 ToolShape에 대한 정보를 설정하는 함수이다.
현재 장착된 Tool이 없을 경우, 빈 문자열을 전달하면 현재 설정되어 있는 정보가 초기화된다.

해당 함수는 M2.4 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	ToolShape 이름

리턴

값	설명
0	오류
1	성공

예제

1	Drfl.set_tool_shape("tool_shape1") //TP에서 등록된 "tool_shape1"의 정보를 활성화한다.
---	---

4.7.10 CDRFLEEx.get_workpiece_weight

기능

작업물의 무게를 측정하여 반환한다.

인수

없음

리턴

값	설명
float	측정 무게 값

예제

```
1 float weight = Drfl.get_workpiece_weight();
```

4.7.11 CDRFLEx.reset_workpiece_weight

기능

소재의 무게를 측정하기 전 알고리즘의 초기화를 위해 소재의 무게 정보를 초기화한다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```
1 Drfl.reset_workpiece_weight()
```

4.7.12 CDRFLEx.set_singularity_handling

기능

task motion에서 특이점의 영향으로 path deviation이 발생할 경우 대응 정책을 사용자가 선택할 수 있도록 하다. mode의 설정은 아래와 같은 설정이 가능하다.

- 자동회피 모드(Default) : SINGULARITY_AVOIDANCE_AVOID
- 경로 우선 : SINGULARITY_AVOIDANCE_STOP
- 속도 가변 : SINGULARITY_AVOIDANCE_VEL

기본 설정은 자동회피 모드이며, 이 설정의 경우 특이점으로 인한 불안정성을 감소시키지만 path tracking 정확도가 감소한다.

경로 우선 설정의 경우 singularity 의 영향으로 불안정성이 발생할 가능성이 있는 경우, 감속 후 warning 메시지를 출력하고 해당 Task를 종료한다.

속도 가변 설정의 경우 특이점으로 인한 불안정을 감소시키면서 path tracking 정확도를 높인다. 하지만 특이점 구간에서 TCP 속도 변경이 발생한다.

인수

인수명	자료형	기본값	설명
eMode	SINGULARITY_AVOIDANCE	-	열거형 및 상수 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 float p1[6] = {400, 500, 800, 0, 180, 0};
2 float p2[6] = {400, 500, 500, 0, 180, 0};
3 float p3[6] = {400, 500, 200, 0, 180, 0};
4 Drfl.set_singularity_handling(SINGULARITY_AVOIDANCE_AVOID); // 특이점 자동회피
모드
5 Drfl.MoveL(p1, 10, 20);
6 Drfl.set_singularity_handling(SINGULARITY_AVOIDANCE_STOP); // Task 모션 경로
우선
7 Drfl.MoveL(p2, 30, 60);

```

8	Drfl.set_singularity_handling(SINGULARITY_AVOIDANCE_VEL); // 특이점 속도 가변 모드
---	---

4.7.13 CDRFLEX.setup_monitoring_version

기능

로봇 제어기에서 전송되는 모니터링 정보의 버전 정보를 설정하기 위한 함수이다.

해당 함수는 M2.5 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
iVersion	int	-	버전 정보 0: 버전 v0 1: 버전 v1

리턴

값	설명
0	오류
1	성공

예제

1	Drfl.setup_monitoring_version(1); //모니터링 데이터 버전 정보를 1로 설정
---	---

4.7.14 CDRFLEX.config_program_watch_variable

기능

로봇 제어기에서 프로그램 실행 시 프로그램 내부 변수를 모니터링 하기 위하여 모니터링할 변수 명을 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eDivision	VARIABLE_TYPE	-	상수 및 열거형 정의 참조
eType	TYPE_TYPE	-	상수 및 열거형 정의 참조
strName	string	-	128바이트 변수명 문자열
strData	string	-	128바이트 데이터 문자열

리턴

값	설명
0	오류
1	성공

예제

```
1 Drfl.config_program_watch_variable(VARIABLE_TYPE_INSTALL, DATA_TYPE_FLOAT,
"var", "1.22"); // 설치변수 “var”을 float 1.22로 저장
```

4.7.15 CDRFLEx.set_user_home

기능

로봇 제어기에서 사용자 홈 위치를 설정하기 위한 함수이다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.set_user_home();
2 Drfl.move_home(MOVE_HOME_USER, (unsigned char)1);

```

4.7.16 CDRFLEx.servo_off

기능

로봇 제어기에서 모터 및 브레이크 전원을 설정(로봇 전원 off)하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eStopType	enum.STOP_TYPE	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.servo_off(STOP_TYPE_QUICK);

```

4.7.17 CDRFLEx.release_protective_stop

기능

로봇 제어기에서 안전 정지 상태를 해제하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eReleaseMode	enum.RELEASE_MODE	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```
1 Drfl.release_protective_stop(RELEASE_MODE_RELEASE);
```

4.7.18 CDRFLEx.change_collision_sensitivity

기능

로봇 제어기에서 충돌 민감도를 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
fSensitivity	float	-	충돌 민감도(0~100)

리턴

값	설명
0	오류
1	성공

예제

```
1 Drfl.change_collision_sensitivity(20);
```

4.7.19 CDRFLEx.set_safety_mode**기능**

로봇 제어기에서 안전 모드를 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eSafetyMode	enum.SAFE TY_MODE	-	상수 및 열거형 정의 참조
eSafetyEvent	enum.SAFE TY_EVENT	-	상수 및 열거형 정의 참조

리턴

값	설명
0	실패
1	성공

예제

1	Drfl.set_safety_mode(SAFETY_MODE_MANUAL, SAFETY_EVENT_MOVE); //안전 모드를 수동-동작으로 설정
---	--

4.7.20 CDRFLEx.set_auto_servo_off

기능

로봇 제어기에서 일정 시간 경과 시 동작되는 자동 Safe-Off 상태 전환 기능을 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
bFuncEnable	bool	-	미사용 : 0 사용 : 1
fElapseTime	float	-	분 단위 설정 값

리턴

값	설명
0	실패
1	성공

예제

1	Drfl.set_auto_servo_off(1, 3);
---	--------------------------------

4.7.21 CDRFLEx.set_workpiece_weight

기능

로봇 끝단의 툴 무게/무게 중심에 추가하여 작업물의 무게/무게 중심, 기타 정보를 설정한다. 전체 페이로드의 무게 및 무게 중심은 설정된 툴 무게/무게 중심과 작업물의 무게/무게 중심을 종합하여 반영된다. 작업물의 종류가 다양하거나 동적으로 무게가 변경되어야 하는 Application에서 사용 가능한 함수이다.

⚠ 주의

- 작업물 무게 변경은 Auto Mode 중 Collision Detection과 TCP SLF Violation 검사가 모두 무효화된 경우에만 허용됩니다.
- 현재 버전에서, Collision Detection은 Collision Sensitivity를 0으로 Override 한 경우, TCP SLF Violation은 TCP SLF Limit을 최대로 Override 한 경우를 기능이 무효화된 상태로 간주합니다. 이 무효화는 Collision Sensitivity Reduction Zone, Custom Zone 등을 이용해서 설정할 수 있습니다.
- 그 외의 경우에 작업물 무게가 0으로 설정되어 있지 않으면 SS1 보호 정지를 발생시킵니다.
- 에러 발생으로 로봇이 정지하여 매뉴얼로 복구하여야 하는 경우 복구 모드에서 로봇을 원하는 위치에 놓고, Auto Mode에서 해당 Zone들이 활성화된 상태에서 Servo On, I/O 조작 등을 통해 작업물을 언로딩할 수 있습니다.
- 설정된 툴 무게를 변경하는 경우에는 작업물 무게가 0으로 초기화 됩니다.
- set_safety_mode 명령어를 사용하여 안전 모드 설정을 직접 수행해야 합니다(예제 참조)
- set_tool, set_workpiece_weight 함수를끼리 연달아 사용하는 경우에는 transition_time 만큼 wait(transition_time) 함수를 사이에 넣어서 사용해야 합니다. 그렇지 않으면 무게 변경에 오류가 생길 수 있습니다.

인수

인수명	자료형	기본값	설명
fWeight	float	0	무게 [kgf]
fCog	float[3]	[0, 0, 0]	무게 중심 위치(x, y, z) [mm]
eCogRef	enum.COG_REFERENCE	COG_REFERENCE_TCP	상수 및 열거형 정의 참조
eAddUp	enum.ADD_UP	ADD_UP_REPLACE	상수 및 열거형 정의 참조
fStartTime	float	-10000	전달 시간 이후 작업물 무게 변경 시작[sec]
fTransitionTime	float	-10000	전달 시간 동안 작업물 무게 점진적 변경[sec]

리턴

값	설명
0	실패
1	성공

예제

```

1 Drfl.set_safety_mode(SAFETY_MODE_AUTONOMOUS, SAFETY_MODE_EVENT_MOVE);
2 Drfl.set_workpiece_weight();
3 Drfl.set_safety_mode(SAFETY_MODE_AUTONOMOUS, SAFETY_MODE_EVENT_STOP);

```

4.8 I/O 제어 함수

4.8.1 CDRFLEEx.set_tool_digital_output

기능

로봇 제어기에서 로봇 끝단에 장착된 디지털 접점에 신호를 출력하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioIndex	enum.GPIO_TOOL - DIGITAL_INDEX	-	상수 및 열거형 정의 참조
bOnOff	bool	-	출력하고자 하는 데이터 <ul style="list-style-type: none"> • ON: 1 • OFF: 0

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

1 //로봇 암의 디지털 1번 출력 접점 출력
2 drfl.set_tool_digital_output(GPIO_TOOL_DIGITAL_INDEX_1, 1);

```

4.8.2 CDRFLE.get_tool_digital_input

기능

로봇 제어기에서 로봇 끝단에 장착된 디지털 접점의 신호를 확인하기 함수이다.

인수

인수명	자료형	기본값	설명
eGpioldex	enum.GPIO_TOOL_DIGITAL_INDEX	-	상수 및 열거형 정의 참조

리턴

값	설명
0	OFF
1	ON

예제

```

1 //로봇 암의 디지털 1번 입력 접점 확인
2 bool bSignal = drfl.get_tool_digital_input(GPIO_TOOL_DIGITAL_INDEX_1);
3 if (bSignal == True) {
4     // do something
5 }

```

4.8.3 CDRFLEEx.get_tool_digital_output

기능

로봇 제어기에서 로봇 끝단에 장착된 디지털 접점의 신호를 확인하기 함수이다.

인수

인수명	자료형	기본값	설명
eGpiodex	enum.GPIO_TOOL — DIGITAL_INDEX	-	상수 및 열거형 정의 참조

리턴

값	설명
0	OFF
1	ON

예제

```

1 //로봇 암의 디지털 1번 출력 접점 확인
2 bool bSignal = drfl.get_tool_digital_output(GPIO_TOOL_DIGITAL_INDEX_1);
3 if (bSignal == True) {
4     // do something
5 }
```

4.8.4 CDRFLEEx.set_digital_output

기능

로봇 제어기에서 컨트롤 박스에 장착된 디지털 접점에 신호를 출력하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioidex	enum.GPIO_CTRLBOX — DIGITAL_INDEX	-	상수 및 열거형 정의 참조
bOnOff	bool	-	출력하고자 하는 데이터 • ON: 1 • OFF: 0

리턴

값	설명
0	오류
1	성공

예제

```

1 // 컨트롤박스 디지털 1번 출력 접점 출력
2 drfl.set_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1, 1);

```

4.8.5 CDRFLEEx.get_digital_input

기능

로봇 제어기에서 컨트롤 박스에 장착된 디지털 접점의 신호를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioidex	enum.GPIO_CTRLBO X_ DIGITAL_INDEX	-	상수 및 열거형 정의 참조

리턴

값	설명
0	OFF
1	ON

예제

```

1 //컨트롤 박스의 디지털 1번 입력 접점 체크
2 bool bSignal = drfl.get_digital_input(GPIO_CTRLBOX_DIGITAL_INDEX_1);
3 if (bSignal ==TRUE) {
4     // do something
5 }
```

4.8.6 CRDFLEx.get_digital_output

기능

로봇 제어기에서 컨트롤 박스에 장착된 디지털 접점의 신호를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioldex	enum.GPIO_CTRLB OX_ DIGITAL_INDEX	-	상수 및 열거형 정의 참조

리턴

값	설명
0	OFF
1	ON

예제

```

1 // 컨트롤 박스의 디지털 1번 출력 접점 체크
2 bool bSignal = drfl.get_digital_output(GPIO_CTRLBOX_DIGITAL_INDEX_1);
3 if (bSignal == TRUE) {
4     // do something
5 }
```

4.8.7 CDRFLEx.set_mode_analog_input

기능

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 입력 접점에 대한 채널 모드를 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioidex	enum.GPIO_CTRLBOX_ANALOG_INDEX_	-	상수 및 열거형 정의 참조
mod	enum.GPIO_ANALOG_TYPE	GPIO_ANALOG_TYPE_CURRENT	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 // 컨트롤 박스의 아날로그 1번 입력 접점을 전류 모드로 설정함
2 drfl.set_mode_analog_input(GPIO_CTRLBOX_ANALOG_INDEX_1,
3 GPIO_ANALOG_TYPE_CURRENT);
4
5 // 컨트롤 박스의 아날로그 2번 입력 접점을 전압 모드로 설정함.
6 drfl.set_mode_analog_input(GPIO_CTRLBOX_ANALOG_INDEX_2,
7 GPIO_ANALOG_TYPE_VOLTAGE);
```

4.8.8 CDRFLEEx.set_mode_analog_output

기능

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 출력 접점에 대한 채널 모드를 설정하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpioidex	enum.GPIO_CTRLBOX_ANA LOG_INDEX_	-	상수 및 열거형 정의 참조
mod	enum.GPIO_ANALOG_TYPE	GPIO_ANALOG_TYPE _CURRENT	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 // 컨트롤박스의 아날로그 1번 출력 접점을 전류 모드로 설정함
2 drfl.set_mode_analog_output(pCtrl, GPIO_CTRLBOX_ANALOG_INDEX_1,
3                             GPIO_ANALOG_TYPE_CURRENT);
4
5 // 컨트롤박스의 아날로그 2번 출력 접점을 전압 모드로 설정함
6 drfl.set_mode_analog_output(pCtrl, GPIO_CTRLBOX_ANALOG_INDEX_2,
7                             GPIO_ANALOG_TYPE_VOLTAGE);

```

4.8.9 CDRFLEEx.set_analog_output

기능

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 접점에 신호를 출력하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpiodex	enum.GPIO_CTRLBOX_ANALOG_INDEX_	-	상수 및 열거형 정의 참조
fValue	float	-	아날로그 신호 출력 <ul style="list-style-type: none"> 전류 모드인 경우: 4.0~20.0 [mA] 전압 모드인 경우: 0~10.0 [V]

리턴

값	설명
0	오류
1	성공

예제

```

1 // 컨트롤 박스의 아날로그 1번 출력 접점을 전류 모드로 설정
2 drfl.set_mode_analog_output(GPIO_CTRLBOX_ANALOG_INDEX_1,
3   GPIO_ANALOG_TYPE_CURRENT);
4
5 // 컨트롤 박스의 아날로그 1번 출력 접점에 5.2mA 출력
6 drfl.set_analog_output(GPIO_CTRLBOX_ANALOG_INDEX_1, 5.2);

```

4.8.10 CDRFLE.get_analog_input

기능

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 접점의 신호를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eGpiodex	enum.GPIO_CTRLBOX_ANALOG_INDEX_	-	상수 및 열거형 정의 참조

리턴

값	설명
아날로그 신호 입력	<ul style="list-style-type: none"> 전류 모드인 경우: 4.0~20.0 [mA] 전압 모드인 경우: 0~10.0 [V]

예제

```

1 // 컨트롤 박스의 아날로그 1번 입력 점접을 전류 모드로 설정
2 drfl.set_mode_analog_output(GPIO_CTRLBOX_ANALOG_INDEX_1,
3   GPIO_ANALOG_TYPE_CURRENT);
4 // 컨트롤 박스의 아날로그 1번 입력 점접 전류값 확인
5 float fCurrent = drfl.get_analog_input(GPIO_CTRLBOX_ANALOG_INDEX_1);

```

4.8.11 CDRFLEX.add_modbus_signal

기능

로봇 제어기에서 Modbus의 I/O 신호 사전에 등록하여 사용하기 위한 함수이다. 본 함수를 이용하여 등록된 Modbus I/O 신호 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능하다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	modbus signal 이름
strIpAddress	string	-	modbus 모듈 ip 주소
nPort	unsigned short	-	modbus 모듈 port
eRegType	enum	-	상수 및 열거형 정의 참조
iRegIndex	unsigned short	-	Modbus signal의 index

인수명	자료형	기본값	설명
nRegValue	unsigned short	0	type이 MODBUS_REGISTER_TYPE_COILS 또는 MODBUS_REGISTER_TYPE_HOLDING_REGISTER 일 때 출력값 (그 외 경우에는 무시됩니다.)
nSlaveId	unsigned short	255	ModbusTCP 모듈의 Slave ID(0 or 1-247 or 255)

리턴

값	설명
0	오류
1	성공

예제

```

1  /*
2   * Modbus IO를 연결하고 접점을 할당하는 예제
3   * Modbus IO IP: 192.168.127.254
4   * input 2점: "di1", "di2"
5   */
6
7 // set <modbus> input : "di1", "di2"
8 drfl.add_modbus_signal("di1", "192.168.127.254" , 502,
9 MODBUS_REGISTER_TYPE_DISCRETE_INPUTS, 0, 0);
drfl.add_modbus_signal("di2", "192.168.127.254" , 502,
MODBUS_REGISTER_TYPE_DISCRETE_INPUTS, 0, 0);

```

4.8.12 CDRFLEX.del_modbus_signal

기능

로봇 제어기에 사전에 등록된 Modbus I/O 신호 정보를 삭제하기 위한 함수이다

인수

인수명	자료형	기본값	설명
strSymbol	string	-	등록된 modbus 신호의 이름

리턴

값	설명
0	오류
1	성공

예제

```

1  /*
2   * Modbus IO 신호가 "di1", "do1"로 등록되어 있는데,
3   * 이 신호 등록을 삭제하고자 하는 경우.
4   */
5   drfl.del_modbus_signal("di1")    // "di1" 접점 등록 삭제
6   drfl.del_modbus_signal("do1")    // "do1" 접점 등록 삭제

```

4.8.13 CDRFLEEx.set_modbus_output

기능

로봇 제어기에서 Modbus I/O 신호 접점에 신호를 출력하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	modbus 이름
nValue	unsigned short	-	<ul style="list-style-type: none"> Modbus digital I/O 인 경우: 0 or 1 Modbus analog I/O 인 경우: 데이터

리턴

값	설명
0	오류
1	성공

예제

```

1 //Modbus digital I/O가 연결되어 있고, 신호가 “do1”, “do2”로 등록되어 있는 경우
2 drfl.set_modbus_output("do1",1);
3 drfl.set_modbus_output("do2",0);

```

4.8.14 CDRFLEX.get_modbus_input

기능

로봇 제어기에서 Modbus I/O 신호 접점의 신호를 확인하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
strSymbol	string	-	modbus 이름

리턴

값	설명
unsigned short	<ul style="list-style-type: none"> Modbus Digital I/O 인 경우: 0 or 1 Modbus Analog 모듈인 경우: 데이터

예제

```

1 //Modbus digital I/O가 연결되어 있고, 신호가 “di1”, “di2”로 등록되어 있는 경우
2 unsigned short signal1 = drfl.get_modbus_input("di1");
3 unsigned short signal2 = drfl.get_modbus_input("di2");
4 if ( signal1 == 1 && signal2 == 1) {
5
6     // do something...
7 }

```

4.8.15 CDRFLEX.flange_serial_open

기능

로봇 제어기에서 Flange Serial 통신 포트를 열기 위한 명령어이다.

해당 명령어는 신규 플랜지 버전(v2)에서 사용 불가

인수

인수명	자료형	기본값	설명
baudrate	Int	115200	Baudrate 2400, 4800, 9600, 19200, 38400, 57600, 115200 etc
bytesize	enum.BYTE_SIZE	BYTE_SIZE_EIGHTBITS	상수 및 열거형 정의 참조
parity	enum.PARITY_CHECK	PARITY_CHECK_NONE	상수 및 열거형 정의 참조
stopbits	enum.STOP_BITS	STOPBITS_ONE	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.flange_serial_open(115200);
2 Drfl.flange_serial_write(8, "test1234");
3 LPFLANGE_SER_RXD_INFO temp = Drfl.flange_serial_read(4.5);
4 cout << "serial read : " << temp->_cRxd << endl;
5 cout << "serial cnt : " << (int)temp->_iSize << endl;
6 Drfl.flange_serial_close();

```

4.8.16 CDRFLEEx.flange_serial_close

기능

로봇 제어기에서 Flange Serial 통신 포트를 닫기 위한 명령어이다.

해당 명령어는 신규 플랜지 버전(v2)에서 사용 불가

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.flange_serial_open(115200);
2 Drfl.flange_serial_write(8, "test1234");
3 LPFLANGE_SER_RXD_INFO temp = Drfl.flange_serial_read(4.5);
4 cout << "serial read : " << temp->_cRxd << endl;
5 cout << "serial cnt : " << (int)temp->_iSize << endl;
6 Drfl.flange_serial_close();

```

4.8.17 CDRFLEX.flange_serial_write

기능

로봇 제어기에서 Flange Serial 통신 포트를 통해 데이터를 전송하기 위한 함수이다

port 파라미터는 신규 플랜지 버전(v2)에서만 사용 가능

인수

인수명	자료형	기본값	설명
nSize	Int	-	데이터 크기
pSendData	Char	-	데이터 값(최대 256 바이트)
nPort	int	1	port 값 1 : X1 2 : X2(M/H시리즈에서만 사용 가능)

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.flange_serial_open(115200);
2 Drfl.flange_serial_write(8, "test1234");
3 LPFLANGE_SER_RXD_INFO temp = Drfl.flange_serial_read(4.5);
4 cout << "serial read : " << temp->_cRxd << endl;
5 cout << "serial cnt : " << (int)temp->_iSize << endl;
6 Drfl.flange_serial_close();

```

4.8.18 CDRFLEEx.get_tool_analog_input

기능

로봇 플랜지에서 아날로그 입력에 해당하는 채널의 값을 불러온다.

해당 명령어는 신규 플랜지 버전(v2)에서만 사용 가능

인수

인수명	자료형	기본값	설명
nCh	int	-	1 : channel 1 2 : channel 2 3 : channel 3 (M/H 모델만 지원) 4 : channel 4 (M/H 모델만 지원)

리턴

값	설명
float	해당 channel의 analog input 값 ▪ 전류 모드인 경우: 4.0~20.0 [mA] 전압 모드인 경우: 0~10.0 [V]

예제

```

1 Drfl.set_mode_tool_analog_input(1, GPIO_ANALOG_TYPE_CURRENT);
2 Drfl.set_mode_tool_analog_input(2, GPIO_ANALOG_TYPE_VOLTAGE);
3 float cur = Drfl.get_tool_analog_input(1);
4 float vol = Drfl.get_tool_analog_input(2);

```

4.8.19 CDRFLEx.set_tool_digital_output_level

기능

로봇 플랜지에서 디지털 출력에 대한 출력 강도를 설정한다.

해당 명령어는 신규 플랜지 버전(v2)에서만 사용 가능

인수

인수명	자료형	기본값	설명
nLv	int	12	Flange I/O 에 설치된 Digital output에 출력하고자 하는 전압의 세기 ▪ 0, 12, 24 [V]

리턴

값	설명
0	오류
1	성공

예제

1	Drfl.set_tool_digital_output_level(24);
---	---

4.8.20 CDRFLEx.set_tool_digital_output_type

기능

플랜지에서 디지털 출력에 대한 출력 타입을 설정한다.

해당 명령어는 신규 플랜지 버전(v2)에서만 사용 가능

인수

인수명	자료형	기본값	설명
nPort	int	1	port 값 1 : X1 2 : X2(M/H시리즈에서만 사용 가능)
eOutputType	enum.OUTPUT_TYPE	OUTPUT_TYPE_PNP	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

1	Drfl.set_tool_digital_output_type(1, OUTPUT_TYPE_PNP);
---	--

4.8.21 CDRFLEx.set_mode_tool_analog_input

기능

플랜지에서 아날로그 입력에 대한 입력 타입을 설정한다.

해당 명령어는 신규 플랜지 버전(v2)에서만 사용 가능

인수

인수명	자료형	기본값	설명
nCh	int	-	1 : channel 1 2 : channel 2 3 : channel 3 (M/H 모델만 지원) 4 : channel 4 (M/H 모델만 지원)
eAnalogType	enum.GPIO_ANALOG_TYPE		상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.set_mode_tool_analog_input(1, GPIO_ANALOG_TYPE_CURRENT);
2 Drfl.set_mode_tool_analog_input(2, GPIO_ANALOG_TYPE_VOLTAGE);
3 float cur = Drfl.get_tool_analog_input(1);
4 float vol = Drfl.get_tool_analog_input(2);

```

4.9 프로그램 제어 함수

4.9.1 CDRFLEx.drl_start

기능

로봇 제어기에서 DRL 언어로 구성된 프로그램(태스크)을 실행하기 위한 함수이다.

인수

인수명	자료형	기본값	설명
eRobotSystem	enum.ROBOT_SYSTEM		상수 및 열거형 정의 참조
strDrlProgram	string		실행 시킬 DRL 프로그램 문자열

리턴

값	설명
0	오류
1	성공

예제

```

1  string strDrlProgram = "loop = 0\nwhile loop < 3:\n"
2    movej(posj(10,10.10,10,10.10), vel=60, acc=60)\n
3    movej(posj(00,00.00,00,00.00), vel=60, acc=60)\n      loop+=1\n
4    movej(posj(10,10.10,10,10.10), vel=60, acc=60)";\n
5    if (Drfl.get_robot_state() == STATE_STANDBY) {\n
6      Drfl.set_robot_mode(ROBOT_MODE_AUTONOMOUS);\n
7      if (Drfl.get_robot_mode() == ROBOT_MODE_AUTONOMOUS) {\n
8        // 자동모드\n
9        ROBOT_SYSTEM eTargetSystem = ROBOT_SYSTEM_VIRTUAL;\n
10       Drfl.drl_start(eTargetSystem, strDrlProgram);\n
11     }\n
12   }

```

● 알아두기

- 로봇 운용 상태가 지령 대기상태(STATE_STANDBY)이어야 하며, 로봇 모드가 자동모드일 때 사용해야 정상 동작한다.
- DRL 프로그램 작성은 별로 Programming Manual 문서를 참조해서 작성해야 합니다.

4.9.2 CDRFLEEx.drl_stop

기능

로봇 제어기에서 현재 실행중인 DRL 프로그램(태스크)을 정지하기 위한 함수입니다. 인자로 받는 iStopType에 따라 다르게 정지하며, 현재 수행하고 있는 구간의 모션을 정지합니다.

인수

인수명	자료형	기본값	설명
iStopType	unsigned char	1	0 : Slow Stop 1 : Quick Stop

리턴

값	설명
0	오류
1	성공

예제

```

1 DRL_PROGRAM_STATE eProgramState = drfl.get_program_state();
2 if ((eProgramState == DRL_PROGRAM_STATE_PLAY) ||
3     (eProgramState == DRL_PROGRAM_STATE_HOLD)) {
4
5     drfl.drl_stop(1);
6     //...
7 }
```

알아두기

- 프로그램이 정상 및 에러로 종료된 경우에도, 반드시 프로그램 정지 명령을 수행해야 한다.

4.9.3 CDRFLEEx.drl_pause

기능

로봇제어기에서 현재 실행 중인 DRL 프로그램(태스크)을 일시 정지하기 위한 함수이다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1  if (drfl.get_program_state() == DRL_PROGRAM_STATE_PLAY) {
2      drfl.drl_pause();
3  }
```

4.9.4 CDRFLEEx.drl_resume

기능

로봇 제어기에서 현재 일시 정지된 DRL 프로그램(태스크)을 재개하기 위한 함수이다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 if (drfl.get_program_state() == DRL_PROGRAM_STATE_HOLD) {
2     bool bResult = drfl.drl_resume();
3     //...
4 }
```

4.9.5 CDRFLEx.change_operation_speed

기능

DRL로 작성된 프로그램의 작동 속도를 조절한다. 인수는 현재 설정된 속도에 대한 상대적인 비율을 백분율로 환산한 값으로, 1에서 100까지의 값을 가진다.

인수

인수명	자료형	기본값	설명
speed	float	-	operation speed(10~100)(%)

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.change_operation_speed(10);
2 string strDrlProgram = "loop = 0\nwhile loop < 3:\n"
3     movej(posj(10,10.10,10,10.10), vel=60, acc=60)\n
4     movej(posj(00,00.00,00,00.00), vel=60, acc=60)\n    loop+=1\n
5     movej(posj(10,10.10,10,10.10), vel=60, acc=60)";
6
7 if (Drfl.get_robot_state() == STATE_STANDBY) {
8     Drfl.set_robot_mode(ROBOT_MODE_AUTONOMOUS);
9     if (Drfl.get_robot_mode() == ROBOT_MODE_AUTONOMOUS) {
10        // 자동모드
11        ROBOT_SYSTEM eTargetSystem = ROBOT_SYSTEM_VIRTUAL;
12        Drfl.drl_start(eTargetSystem, strDrlProgram);
13    }
14 }
```

11 }

4.9.6 CDRFLEEx.save_sub_program

기능

작성한 DRL언어로 구성된 프로그램(태스크)를 서브 프로그램으로 저장한다.

인수

인수명	자료형	기본값	설명
iTargetSystem	SUB_PROGRAM	-	상수 및 열거형 정의 참조
strFileName	string	-	256바이트 파일 이름 정보
strDrlProgram	string	-	N개의 문자열 버퍼

리턴

값	설명
0	오류
1	성공

예제

```

1 string drl_string= "movej([0,0,0,0,0,0], 60, 30)";
2 Drfl.save_sub_program(SUB_PROGRAM_SAVE, "sub_test", drl_string.c_str());

```

4.9.7 CDRFLEEx.tp_popup_response

기능

DRL프로그램 동작 시 유형화 메시지(Popup) 출력 후 사용자 응답에 따른 다음 동작(일시 정지된 프로그램의 중지 및 재개)을 제어한다.

인수

인수명	자료형	기본값	설명
eRes	POPUP_RESPONSE_NSE	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 // DRL 명령어 tp_popup 실행 시
2 Drfl.tp_popup_response(POPUP_RESPONSE_RESUME);
```

4.9.8 CDRFLEEx.tp_get_user_input_response

기능

DRL프로그램 동작 시 사용자 입력 요구 시 사용자 입력 정보를 전달한다.

인수

인수명	자료형	기본값	설명
strUserInput	string	-	256바이트 사용자 입력 문자열

리턴

값	설명
0	오류
1	성공

예제

```

1 // DRL 명령어 tp_get_user_input 실행 시
2 Drfl.tp_get_user_input_response("tp get user input response");

```

4.10 힘/강성 제어 및 기타 사용자 편의 함수

4.10.1 CDRFLEx.parallel_axis

CDRFLEx.parallel_axis(fTargetPos1, fTargetPos2, fTargetPos3, eTaskAxis, eSourceRef)

기능

입력된 기준 좌표계(eSourceRef) 기준의 3개의 포즈(fTargetPos1, fTargetPos2, fTargetPos3)가 이루는 평면의 normal vector 방향에 Tool 좌표계의 지정축의 방향(eTaskAxis)을 일치시킨다. 이 때 로봇의 TCP 위치는 현재 위치를 유지한다.

인수

인수명	자료형	기본값	설명
fTargetPos1	float[6]	-	6개의 Task Space 정보
fTargetPos2	float[6]	-	6개의 Task Space 정보
fTargetPos3	float[6]	-	6개의 Task Space 정보
eTaskAxis	enum.TASK_AXIS	-	상수 및 열거형 정의 참조
eSourceRef	enum.COORDINATE_SYSTEM	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 float x0[6] = {0, 0, 90, 0, 90, 0};
2 Drfl.movej(x0, 60, 30);
3 float x1[6] = {0, 500, 700, 30, 0, 90};
4 float x2[6] = {500, 0, 700, 0, 0, 45};
5 float x3[6] = {300, 100, 500, 45, 0, 45};
6 Drfl.parallel_axis(x1, x2, x3, TASK_AXIS_X);

```

CDRFLEEx.parallel_axis(eSourceVec, eTaskAxis, eTargetRef)

기능

입력된 기준 좌표계(eTargetRef) 기준의 벡터(fSourceVec) 방향에 Tool 좌표계의 지정축의 방향을 일치시킨다. 이 때 로봇의 TCP 위치는 현재 위치를 유지한다.

인수

인수명	자료형	기본값	설명
eSourceVec	float[3]	-	3개의 XYZ 정보
eTaskAxis	enum.TASK_AXIS	-	상수 및 열거형 정의 참조
eTargetRef	enum.COORDINATE_SYSTEM	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 float v[3] = {1000, 700, 300};
2 Drfl.parallel_axis(v, TASK_AXIS_X, COORDINATE_SYSTEM_BASE);

```

4.10.2 CDRFLEEx.align_axis

`CDRFLEEx.align_axis(fTargetPos1, fTargetPos2, fTargetPos3, fSourceVec, eTaskAxis, eTargetRef)`

기능

입력된 기준 좌표계(eSourceRef) 기준의 3개의 포즈(fTargetPos1, fTargetPos2, fTargetPos3)가 이루는 평면의 normal vector 방향에 Tool 좌표계의 지정축의 방향(eTaskAxis)을 일치시킨다. 이 때 로봇의 TCP 위치는 현재 위치를 유지한다.

인수

인수명	자료형	기본값	설명
fTargetPos1	float[6]	-	6개의 Task Space 정보
fTargetPos2	float[6]		6개의 Task Space 정보
fTargetPos3	float[6]		6개의 Task Space 정보
fSourceVec	float[3]		3개의 XYZ 정보
eTaskAxis	enum.TASK_AXIS	-	상수 및 열거형 정의 참조
eTargetRef	enum.COORDINATE_SYSTEM	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 float x1[6] = {0, 500, 700, 30, 0, 0};
2 float x2[6] = {500, 0, 700, 0, 0, 0};
3 float x3[6] = {300, 100, 500, 0, 0, 0};
4 float pos[3] = {400, 400, 500};

```

5	Drfl.align_axis(x1, x2, x3, pos, TASK_AXIS_X, COORDINATE_SYSTEM_BASE);
---	--

CDRFLEEx.align_axis(eTargetVec, eSourceVec, eTaskAxis, eTargetRef)

기능

입력된 기준 좌표계(eTargetRef) 기준의 벡터(fSourceVec) 방향에 Tool 좌표계의 지정축의 방향을 일치시킨다. 이 때 로봇의 TCP 위치는 fTargetVec 위치로 이동시킨다.

인수

인수명	자료형	기본값	설명
eTargetVec	float[3]	-	3개의 XYZ 정보
eSourceVec	float[3]	-	3개의 XYZ 정보
eTaskAxis	enum.TASK_AXIS	-	상수 및 열거형 정의 참조
eTargetRef	enum.COORDINATE_SYSTEM	-	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

1	float v1[3] = {400, 400, 500};
2	float v2[3] = {350, 37, 430};
3	Drfl.align_axis(v1, v2, TASK_AXIS_X, COORDINATE_SYSTEM_BASE);

4.10.3 CDRFLEEx.is_done_bolt_tightening

기능

툴의 조임 토크를 모니터링하여 주어진 시간 내에 설정된 토크(m)에 도달한 경우 True를 반환하고, 주어진 시간을 초과한 경우 False를 반환한다.

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	상수 및 열거형 정의 참조
fTargetTor	float	0	Target torque
fTimeout	float	0	Monitoring duration [sec]

리턴

값	설명
0	오류
1	성공

예제

```

1 float p0[6] = {0,0,90,0,90,0};
2 Drfl.movej(p0, 60, 30);
3 float stx[6] = {3000, 3000, 3000, 200, 200, 200};
4 Drfl.task_compliance_ctrl(stx);
5 float x1[6] = {559, 34.5, 651.5, 0, 180, 60};
6 float velx[2] = {50, 50};
7 float accx[2] = {50, 50};
8 Drfl.amovel(x1, velx, accx);
9 bool res = Drfl.is_done_bolt_tightening(FORCE_AXIS_Z, 10, 5);
10 int x = 0;
11 if(res){
12     x = 1;
13 }
14 else{
15     x = 2;
16 }
```

4.10.4 CDRFLEx.task_compliance_ctrl

기능

기준에 설정한 기준 좌표계를 기준으로 태스크 Compliance control을 시작한다.

인수

인수명	자료형	기본값	설명
fTargetStiffness	float[6]	[3000, 3000, 3000, 200, 200, 200]	3개의 Translational 강성 정보 3개의 회전 강성 정보
eForceReference	enum.COORDINATE_SYSTEM_TOOL	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조
fTargetTime	float	0	강성 변화 시간 [sec] . 범위 0 ~ 1.0 . 주어진 시간 동안 linear transition

리턴

값	설명
0	오류
1	성공

예제

```

1 float p0[6] = {0,0,90,0,90,0};
2 Drfl.movej(p0, 60, 30);
3 float stx[6] = {3000, 3000, 3000, 200, 200, 200};
4 Drfl.task_compliance_ctrl(stx);

```

4.10.5 CDRFLEx.release_compliance_ctrl

기능

Compliance control을 종료하고 현재 위치에서 위치 제어를 시작한다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 float p0[6] = {0, 0, 90, 0, 90, 0};
2 Drfl.movej(p0, 60, 30);
3 float stx[6] = {3000, 3000, 3000, 200, 200, 200};
4 Drfl.task_compliance_ctrl(stx);
5 float stx2[6] = {1, 2, 3, 4, 5, 6};
6 Drfl.set_stiffnessx(stx2);
7 Drfl.release_compliance_ctrl();

```

4.10.6 CDRFLEx.set_stiffnessx

기능

전역으로 설정된 좌표계(set_ref_coord 참조) 기준으로 강성값을 설정한다. 현재 강성 또는 기본값으로부터 STX로 주어진 time값 동안 linear transition한다. Translation 강성의 사용자 범위는 0 ~ 20000N/m, Rotational 강성의 사용자 범위는 0 ~ 400NM/rad이다.

인수

인수명	자료형	기본값	설명
fTargetStiffness	float[6]	-	3개의 Translational 강성 정보 3개의 회전 강성 정보
eForceReference	enum.COORDINATE_SYSTEM_TOOL	COORDINATE_SYS	상수 및 열거형 정의 참조
fTargetTime	float	0	강성 변화 시간 [sec] . 범위 0 ~ 1.0 . 주어진 시간 동안 linear transition

리턴

값	설명
0	오류
1	성공

예제

```

1 float p0[6] = {0, 0, 90, 0, 90, 0};
2 Drfl.movej(p0, 60, 30);
3 float stx[6] = {3000, 3000, 3000, 200, 200, 200};
4 Drfl.task_compliance_ctrl(stx);
5 float stx2[6] = {1, 2, 3, 4, 5, 6};
6 Drfl.set_stiffnessx(stx2);
7 Drfl.release_compliance_ctrl();

```

4.10.7 CDRFLEx.calc_coord

기능

지정한 좌표계(ref) 기준의 최대 4개의 입력점(x1~x4) 및 입력 모드(mod)를 기반으로 새로운 직교 좌표계를 계산한다. 여기서 입력 모드는 입력점의 개수가 2개인 경우에만 유효하다.

입력점의 개수가 1개인 경우, x1의 위치와 회전으로 좌표계가 계산된다.

입력점의 개수가 2개인 경우 입력모드가 0일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 현재의 Tool-z방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산된다.

입력점의 개수가 2개인 경우 입력모드가 1일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 x1의 z축방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산된다.

입력점의 개수가 3개인 경우, x1에서 x2로 향하는 벡터가 x방향으로 정의되며, x1에서 x3으로 향하는 벡터를 v라고 하였을 경우 z방향은 오른손법칙에 따라 x방향 벡터 곱하기 v로 정의되며, x1의 위치가 원점이 되도록 좌표계가 계산된다.

입력점의 개수가 4개인 경우, 입력점의 개수가 3개인 경우와 축의 방향은 동일하며 원점의 위치가 x4의 위치가 되도록 좌표계가 계산된다.

해당 함수는 M2.5 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
nCnt	unsigned short	-	입력점 개수
nInputMode	unsigned short	-	입력 모드 (입력점개수가 2개인 경우에만 유효함) <ul style="list-style-type: none"> • 0: 현재 Tool-z방향 기준으로 사용자 좌표계의 z방향 정의 • 1: x1의 z방향 기준으로 사용자 좌표계의 z방향 정의
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조
fTargetPos1	float[6]		6개의 Task Space 정보
fTargetPos2	float[6]		6개의 Task Space 정보
fTargetPos3	float[6]		6개의 Task Space 정보
fTargetPos4	float[6]		6개의 Task Space 정보

리턴

값	설명
enum.ROBOT_POSE	상수 및 열거형 정의 참조

예제

```

1 float pos1[6] = {500, 30, 500, 0, 0, 0};
2 float pos2[6] = {400, 30, 500, 0, 0, 0};
3 float pos3[6] = {500, 30, 600, 45, 180, 45};
4 float pos4[6] = {500, -30, 600, 0, 180, 0};
5 ROBOT_POSE* pose_user1 = Drfl.calc_coord(4, 0, COORDINATE_SYSTEM_BASE,
6 pos1, pos2, pos3, pos4);
  for (int i=0; i<NUM_TASK; i++)

```

```

7   {
8     cout << pose_user1->_fPosition[i] << endl;
9   }

```

4.10.8 CDRFLEx.set_user_cart_coord

[CDRFLEx.set_user_cart_coord\(iReqId, fTargetPos, eTargetRef\)](#)

기능

기준 좌표계(eTargetRef) 기반의 새로운 사용자 좌표계를 설정할 수 있다. 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계 설정이 불가하다.

인수

인수명	자료형	기본값	설명
iReqId	int	-	식별자(ID) 0 : 자동 생성 101 ~ 120 : 지정 생성
fTargetPos	float[6]	-	6개의 TaskSpace 정보
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
int	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 120)

예제

```

1 float pos1[6] = {500, 30, 500, 0, 0, 0};
2 int id = Drfl.set_user_cart_coord(0, pos1, COORDINATE_SYSTEM_BASE);

```

CDRFLEx.set_user_cart_coord(fTargetPos, fTargetOrg, eTargetRef)**기능**

사용자가 입력 좌표계(eTargetRef) 기준의 포즈 fTargetPos[0], fTargetPos[1], fTargetPos[2]를 이용하여 새로운 직교 좌표계를 설정할 수 있다. 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계 설정이 불가하다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[3][6]	-	6개의 Task Space 정보 #1 6개의 Task Space 정보 #2 6개의 Task Space 정보 #3
fTargetOrg	float[3]	-	사용자 원점 정보
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
int	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 120)

예제

```

1 float x1[6] = {0,500,700,0,0,0};
2 float x2[6] = {500,0,700,0,0,0};
3 float x3[6] = {300,100,500,0,0,0};
4 float org[3] = {10, 20, 30};
5 float x[3][6] = {{0,500,700,0,0,0}, {500,0,700,0,0,0},
{300,100,500,0,0,0}};
6 Drfl.set_user_cart_coord(x, org, COORDINATE_SYSTEM_BASE);

```

[CDRFLEEx.set_user_cart_coord\(fTargetVec, fTargetOrg, eTargetRef\)](#)**기능**

입력좌표계 기준의 벡터 fTargetVec[0]와 fTargetVec[1]을 사용하여 새로운 직교 좌표계를 설정한다. 직교 좌표계의 원점은 입력 좌표계(eTargetRef) 기준의 fTargetOrg에 위치한다. 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계 설정이 불가하다.

해당 함수는 M2.5 hotfix 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
fTargetVec	float[2][3]	-	3개의 XYZ 정보 #1 3개의 XYZ 정보 #2
fTargetOrg	float[3]	-	사용자 원점 정보
eTargetRef	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

리턴

값	설명
int	설정된 Coordinate ID (101 ~ 120)

예제

```

1 float vec[2][3] = {{-1, 1, 1}, {1, 1, 0}};
2 float org[3] = {370.9, -419.7, 651.5};
3
4 int user = Drfl.set_user_cart_coord(vec, org);

```

[4.10.9 CDRFLEEx.overwrite_user_cart_coord](#)**기능**

요청하는 ID의 사용자 좌표계의 좌표 위치(fTargetPos), 기준 좌표계(eTargetRef) 정보를 변경한다.

해당 함수는 M2.5 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
bTargetUpdate	bool	-	0 : 좌표계 미변경 1 : 좌표계 변경
iReqId	int	-	식별자
fTargetPos	float[6]	-	6개의 Task Space 정보
eTargetRef	enum.COORDINATE_SYSTEM_BASE	COORDINATE_SYSTEM_BASE	상수 및 열거형 정의 참조

좌표계 변경(bTargetUpdate) 변수가 0일 경우에는 프로그램 실행시에만 유저 좌표계가 변경되어 유지되어야 하며, 1일 경우에는 상위제어기의 저장된 유저 좌표계 정보 자체가 변경되어야 한다.

리턴

값	설명
int	변경된 Coordinate의 ID(101~120)

예제

```

1 float pos1[6] = {30, 40, 50, 0, 0, 0};
2 int pose_user1 = Drfl.set_user_cart_coord(0, pos1,
COORDINATE_SYSTEM_BASE);
3
4 float pos2[6] = {100, 150, 200, 45, 180, 0};
5 int result = Drfl.overwrite_user_cart_coord(0, pose_user1, pos2);
6
7 cout << result << endl;

```

4.10.10 CDRFLEEx.get_user_cart_coord

기능

해당하는 ID(iReqId)의 사용자 좌표계의 정보인 참조 기준 및 위치 정보를 조회한다.

해당 함수는 M2.5 버전 이상에서만 사용 가능하다.

인수

인수명	자료형	기본값	설명
iReqId	int	-	식별자

리턴

값	설명
USER_COORDINATE	변경된 Coordinate의 ID, 참조 기준 및 위치정보

예제

```

1 float pos[6] = {10, 20, 30, 0, 0, 0};
2 int id = Drfl.set_user_cart_coord(0, pos);
3 USER_COORDINATE *temp = Drfl.get_user_cart_coord(id);

```

4.10.11 CDRFLEX.set_desired_force**기능**

전역으로 설정된 좌표계(set_ref_coord 참조) 기준으로 힘 제어 목표값, 힘 제어 방향, 힘 목표값, 외력참조모드를 설정한다.

인수

인수명	자료형	기본값	설명
fTargetForce	float[6]	-	3개의 Translational 힘 성분 3개의 Rotational 모멘트 성분
iTargetDirection	unsigned char[6]	-	1이면 해당 방향 힘 제어 0이면 해당 방향 compliance 제어
eForceReference	enum.COORDINATE_SYSTEM_TOOL	COORDINATE_SYS TEM_TOOL	상수 및 열거형 정의 참조
fTargetTime	float	0	힘을 증가시키는 데 소요되는 시간[sec] 범위 : 0~1.0

인수명	자료형	기본값	설명
eForceMode	enum.FORCE_MODE	FORCE_MODE_ABOLUTE	상수 및 열거형 정의 참조

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.set_ref_coord(COORDINATE_SYSTEM_TOOL);
2 float x0[6] = {0, 0, 90, 0, 90, 0};
3 Drfl.movej(x0, 60, 30);
4 float stx[6] = {500, 500, 500, 100, 100, 100};
5 Drfl.task_compliance_ctrl(stx);
6 float fd[6] = {0, 0, 0, 0, 0, 10};
7 unsigned char fctrl_dir[6] = {0, 0, 1, 0, 0, 1};
8 Drfl.set_desired_force(fd, fctrl_dir);

```

4.10.12 CDRFLEx.release_force

기능

힘 제어 목표값을 time 값 동안 0으로 줄이고 작업 공간을 순응 제어로 반환한다.

인수

인수명	자료형	기본값	설명
fTargetTime	float	0	힘을 감소시키는데 소요되는 시간 범위 0~1.0

리턴

값	설명
0	오류
1	성공

예제

```

1 Drfl.set_ref_coord(COORDINATE_SYSTEM_TOOL);
2 float x0[6] = {0, 0, 90, 0, 90, 0};
3 Drfl.movej(x0, 60, 30);
4 float stx[6] = {500, 500, 500, 100, 100, 100};
5 Drfl.task_compliance_ctrl(stx);
6 float fd[6] = {0, 0, 0, 0, 0, 10};
7 unsigned char fctrl_dir[6] = {0, 0, 1, 0, 0, 1};
8 Drfl.set_desired_force(fd, fctrl_dir);
9 float x1[6] = {0, 500, 700, 0, 180, 0};
10 float velx[2] = {60, 60};
11 float accx[2] = {30, 30};
12 Drfl.MoveL(x1, velx, accx);
13 Drfl.release_force(0.5);
14 Drfl.release_compliance_ctrl();

```

4.10.13 CDRFLEEx.check_position_condition_abs

기능

주어진 위치 상태를 확인한다. while 혹은 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다. fTargetPos는 입력좌표계(eTargetRef) 기준의 축 방향 및 포즈를 의미한다.

입력 좌표계가(eTargetRef)가 COORDINATE_SYSTEM_TOOL인 경우 입력 위치(fTargetPos)는 BASE 좌표계 기준의 값을 입력해야 한다.

→ 상대 이동 기준은 check_position_condition_rel 함수를 참조

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	축 방향

인수명	자료형	기본값	설명
fTargetMin	float	-	최솟값
fTargetMax	float	-	최댓값
eForceReference	enum.COORDINATE_SYSTEM_TOOL	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참
0	조건이 거짓

예제

```

1 bool CON1 = Drf1.check_position_condition_abs(FORCE_AXIS_X, -5, 0,
COORDINATE_SYSTEM_WORLD);
2 bool CON2 = Drf1.check_position_condition_abs(FORCE_AXIS_Y, -10000, 700);
3 bool CON3 = Drf1.check_position_condition_abs(FORCE_AXIS_Z, -10, -5);
4 bool CON4 = Drf1.check_position_condition_abs(FORCE_AXIS_Z, 30, -10000);
5 bool CON5 = Drf1.check_position_condition_abs(FORCE_AXIS_Z, -10, -5,
COORDINATE_SYSTEM_BASE);
6 bool CON6 = Drf1.check_position_condition_abs(FORCE_AXIS_Z, -10, -5);

```

4.10.14 CDRFLEX.check_position_condition_rel

기능

주어진 위치 상태를 확인한다. while 혹은 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다. eTaskAxis, fTargetPos는 입력좌표계(eTargetRef) 기준의 축 방향 및 포즈를 의미한다.

입력 좌표계가(eTargetRef) COORDINATE_SYSTEM_TOOL인 경우 입력 위치(fTargetPos)는 BASE 좌표계 기준의 값을 입력해야 한다.

→ 절대 이동 기준은 check_position_condition_abs 함수를 참조

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	축 방향
fTargetMin	float	-	최솟값
fTargetMax	float	-	최댓값
fTargetPos	float[6]	-	6개의 Task Space 정보
eForceReference	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참
0	조건이 거짓

예제

```

1 float posx1[6] = {400, 500, 800, 0, 180, 0};
2 bool CON7 = Drf1.check_position_condition_rel(FORCE_AXIS_Z, -10, -5,
posx1, COORDINATE_SYSTEM_TOOL);

```

4.10.15 CDRFLEX.check_position_condition

기능

주어진 위치 상태를 확인한다. while 혹은 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다. eTaskAxis, fTargetPos는 입력좌표계(eTargetRef) 기준의 축 방향 및 포즈를 의미한다.

입력 좌표계가(eTargetRef)가 COORDINATE_SYSTEM_TOOL인 경우 입력 위치(fTargetPos)는 BASE 좌표계 기준의 값을 입력해야 한다.

eMode가 MOVE_MODE_RELATIVE일 경우 check_position_condition_rel을, MOVE_MODE_ABSOLUTE일 경우 check_position_condition_abs를 호출한다.

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	축 방향
fTargetMin	float	-	최솟값
fTargetMax	float	-	최댓값
fTargetPos	float[6]	-	6개의 Task Space 정보
eMode	enum.MOVE_MODE	MOVE_MODE_ABSOLUTE	상수 및 열거형 정의 참조
eForceReference	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참
0	조건이 거짓

예제

```

1 float posx1[6] = {400, 500, 800, 0, 180, 0};
2 bool CON7 = Drfl.check_position_condition(FORCE_AXIS_Z, -10, -5, posx1,
MOVE_MODE_ABSOLUTE);

```

4.10.16 CDRFLEX.check_force_condition

기능

주어진 힘 상태를 확인한다. 단, 힘의 방향은 고려하지 않고 크기로만 비교한다. while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다. 힘 측정 시 eForceAxis는 입력 좌표계(eTargetRef) 기준의 축 방향이고 모멘트 측정 시 eForceAxis는 툴 좌표계 기준의 축 방향이다.

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	상수 및 열거형 정의 참조
fTargetMin	float	-	최솟값
fTargetMax	float	-	최댓값
eForceReference	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참
0	조건이 거짓

예제

```

1  bool fcon1 = Drfl.check_force_condition(FORCE_AXIS_Z, 5, 10,
2   COORDINATE_SYSTEM_WORLD);
3  bool fcon2;
4  bool pcon1;
5  while(true){
6   fcon2 = Drfl.check_force_condition(FORCE_AXIS_C, 30, -10000);
7   pcon1 = Drfl.check_position_condition_abs(FORCE_AXIS_X, 0, 0.1);
8   if(fcon2 && pcon1)
9   {
10    break;
11 }

```

11 }

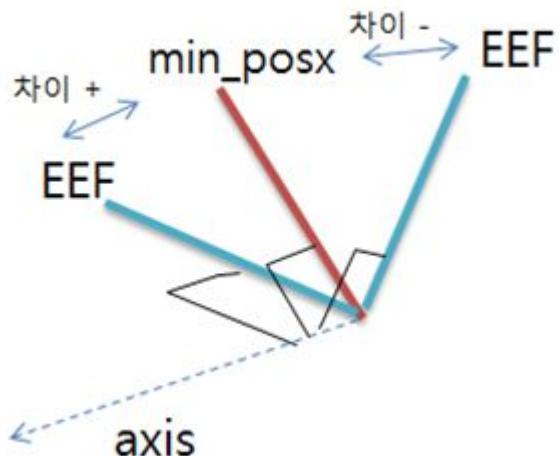
4.10.17 CDRFLEEx.check_orientation_condition

`CDRFLEEx.check_orientation_condition(eForceAxis, fTargetMin, fTargetMax, eForceReference)`

기능

현재 로봇 엔드이펙터의 자세 정보와 주어진 위치 자세 간 차이의 상태를 확인한다. 현재 자세와 주어진 자세 간의 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴한다. 차이가 + 값이면 true를, - 값이면 false를 리턴한다. 현재 자세를 기준으로, 주어진 position보다 차이가 +인지 -인지 확인할 때 사용한다. 사용 예로, 직접교시 position을 이용하여 현재 위치와 차이가 + 방향인지, - 방향인지를 판단한 후 orientation limit에 대한 조건을 만들 수 있다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 반대면 False
- Max만 설정 시: max 차이가 -이면 True, +이면 False



인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	축 방향
fTargetMin	float[6]	-	6개의 최소값 정 보
fTargetMax	float[6]	-	6개의 최대값 정 보

인수명	자료형	기본값	설명
eForceReference	enum.COORDINATE_SYSTEM	COORDINATE_SYSTEM_TOOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참
0	조건이 거짓

예제

```

1 float posx1[6] = {400, 500, 800, 0, 180, 30};
2 float posx2[6] = {400, 500, 500, 0, 180, 60};
3 bool con1 = Drf1.check_orientation_condition(FORCE_AXIS_C, posx1, posx2);

```

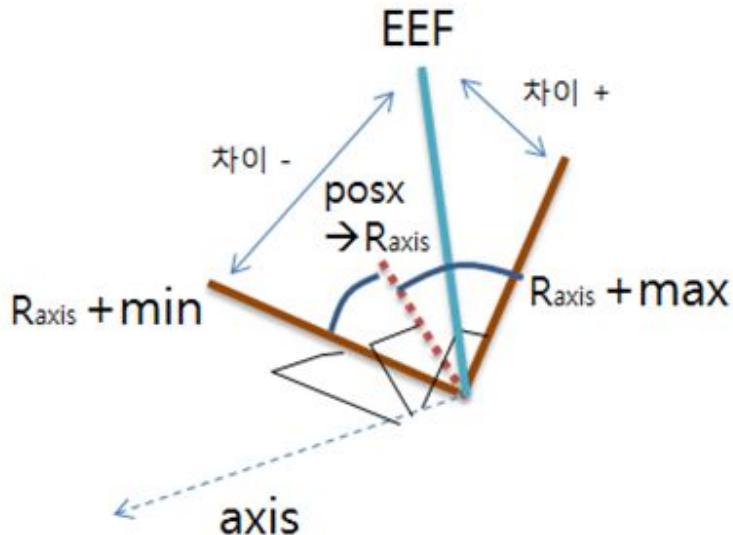
`CDRFLEEx.check_orientation_condition(eForceAxis, fTargetMin, fTargetMax, fTargetPos, eForceReference)`

기능

현재 로봇 엔드이펙터의 자세 정보와 회전각 범위 차이에 대한 상태를 확인한다. 현재 자세와 회전각 범위에 대한 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴한다. 차이가 + 값이면 true를, -값이면 false를 리턴한다. 현재 자세를 기준으로, 주어진 position과 회전각 범위 차이가 +인지 -인지 확인할 때 사용한다. 사용 예로, 어떤 기준이 되는 position에서 min, max로 회전각 범위를 설정하여, 현재 위치와 차이가 + 방향인지, - 방향인지 판단한 후 orientation limit에 대한 조건을 만들 수 있다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 그 반대이면 False

- Max만 설정 시: max 차이가 -이면 True, +이면 False



i 알아두기

회전각 범위: 주어진 position에서 설정된 axis를 기준으로, 상대적인 각도 범위(min, max)를 말한다. 인자 ref에 따라 주어진 position의 기준 좌표계가 정해진다.

인수

인수명	자료형	기본값	설명
eForceAxis	enum.FORCE_AXIS	-	축 방향
fTargetMin	float	-	최소값
fTargetMax	float	-	최대값
fTargetPos	float[6]	-	6개의 Task Space 정보
eForceReference	enum.COORDINATE_SYSTEM_T OOL	COORDINATE_SYSTEM_T OOL	상수 및 열거형 정의 참조

리턴

값	설명
1	조건이 참

값	설명
0	조건이 거짓

예제

1	float posx1[6] = {400, 500, 800, 0, 180, 30};
2	bool con1 = Drfl.check_orientation_condition(FORCE_AXIS_C, 0, 5, posx1);

4.10.18 CDRFLEX.coord_transform**기능**

'eInCoordSystem' 기준 좌표계에서 표현되는 'fTargetPos' Task 좌표를 'eOutCoordSystem' 기준 좌표계에서 표현되는 Task 좌표로 변환하여, 출력한다. 아래의 경우에 대한 좌표변환 계산을 지원한다.

- (eInCoordSystem) 월드 기준 좌표계 → (eOutCoordSystem) 월드 기준 좌표계
- (eInCoordSystem) 월드 기준 좌표계 → (eOutCoordSystem) 베이스 기준 좌표계
- (eInCoordSystem) 월드 기준 좌표계 → (eOutCoordSystem) 툴 기준 좌표계
- (eInCoordSystem) 월드 기준 좌표계 → (eOutCoordSystem) 사용자 기준 좌표계
- (eInCoordSystem) 베이스 기준 좌표계 → (eOutCoordSystem) 월드 기준 좌표계
- (eInCoordSystem) 베이스 기준 좌표계 → (eOutCoordSystem) 베이스 기준 좌표계
- (eInCoordSystem) 베이스 기준 좌표계 → (eOutCoordSystem) 툴 기준 좌표계
- (eInCoordSystem) 베이스 기준 좌표계 → (eOutCoordSystem) 사용자 기준 좌표계
- (eInCoordSystem) 툴 기준 좌표계 → (eOutCoordSystem) 월드 기준 좌표계
- (eInCoordSystem) 툴 기준 좌표계 → (eOutCoordSystem) 베이스 기준 좌표계
- (eInCoordSystem) 툴 기준 좌표계 → (eOutCoordSystem) 툴 기준 좌표계
- (eInCoordSystem) 툴 기준 좌표계 → (eOutCoordSystem) 사용자 기준 좌표계
- (eInCoordSystem) 사용자 기준 좌표계 → (eOutCoordSystem) 월드 기준 좌표계
- (eInCoordSystem) 사용자 기준 좌표계 → (eOutCoordSystem) 베이스 기준 좌표계
- (eInCoordSystem) 사용자 기준 좌표계 → (eOutCoordSystem) 툴 기준 좌표계
- (eInCoordSystem) 사용자 기준 좌표계 → (eOutCoordSystem) 사용자 기준 좌표계

인수

인수명	자료형	기본값	설명
fTargetpos	float[6]	-	6개의 Task Space 정보
eInCoordSystem	float	-	최솟값

인수명	자료형	기본값	설명
eOutCoordSystem	float	-	최댓값

리턴

값	설명
float[6]	6개의 Task Space 정보

예제

```

1 float base_pos[6] = {400, 500, 800, 0, 180, 15};
2 ROBOT_POSE* tool_pos;
3 tool_pos = Drfl.coord_transform(base_pos, COORDINATE_SYSTEM_BASE,
COORDINATE_SYSTEM_TOOL);

```

4.10.19 CDRFLEEx.set_palletizing_mode

기능

팔레타이징 응용 모션에서 wrist 특이점 근방에서 설정한 위치와 속도를 정확히 지킬 수 있는 모드입니다. 모션 지령 중 B방향 성분은 0deg 혹은 180deg로 설정한 상태에서 사용할 때 wrist 특이점 근방에서 안정적으로 사용이 가능합니다.

인수

인수명	자료형	기본값	설명
iMode	unsigned char	-	0: 미활성화 1: 활성화

리턴

값	설명
0	실패
1	성공

예제

```

1 Drfl.set_palletizing_mode(1);
2 Drfl.set_palletizing_mode(0);

```

4.10.20 CDRFLEX.query_modbus_data_list

기능

현재 등록된 모든 모드버스 데이터(TCP/RTU) 정보를 조회한다.

인수

없음

리턴

값	설명
struct.MODBUS_DATA_LIST	구조체 정의 참조

예제

```

LPMODBUS_DATA_LIST tParam = Drfl.query_modbus_data_list();
for (int i = 0; i < tParam->_nCount; i++) {
    if (tParam->_tRegister[i]._iType == 0) // TCP : 0
    {
        cout << tParam->_tRegister[i]._tData._tcp._iRegValue << endl;
    }
    else { // RTU : 1
        cout << tParam->_tRegister[i]._tData._rtu._iRegValue << endl;
    }
}

```

5 실시간 외부 제어(Realtime Control)

5.1 실시간 외부 제어 소개

실시간 외부 제어는 외부 PC에서 두산 제어기를 통해서 로봇을 직접 제어하고 싶어하는 사용자들을 위한 UDP/IP 기반의 통신 api입니다. 기존 TCP/IP 기반의 통신 api와 독립적으로 운용되며, 최대 1kHz까지 실시간 제어에 필요한 입력 데이터(외부 제어기 → 로봇 제어기)와 출력 데이터(로봇 제어기 → 외부 제어기)를 각각 설정한 주기로 주고 받고, 이와 별도로 서보 제어 명령(servoj_rt, servol_rt, speedj_rt, speedl_rt, torque_rt)을 보낼 수 있습니다.

5.1.1 버전

실시간 외부 제어 버전은 입/출력 데이터 구조체 버전에 따라서 관리됩니다. SW 버전이 올라가면서 입/출력 데이터 구조체 각각이 변경될 수 있지만, 외부 제어기를 개발한 입/출력 데이터 버전을 맞춰주면 하위 호환성을 지원합니다.

5.2 통신 연결 함수

5.2.1 CDRFLEEx.connect_rt_control

기능

외부 제어기에서 로봇 제어기로 연결합니다.

인수

인수명	자료형	기본값	설명
strIPAddr	String	192.168.137.100	IP Address
usPort	unsigned int	12345	Port Number

❶ 알아두기

- Real-time external control은 UDP/IP 통신을 사용한다.
- 해당 통신 채널은 기존 TCP/IP 기반 api 및 제어권과 독립적으로 동작합니다.
- 통합제어기(v3) 인 경우 v3.2.2 이하 버전은 192.168.137.50 으로 연결 가능합니다.

⚠ 주의

- 현재 버전에서는 1:1 연결만 지원합니다.

리턴

값	설명
0	오류
1	성공

예제

```
Drfl.connect_rt_control(); //connect udp, if your controller version is below v3.2.2,  
ip address will be 192.168.137.50.
```

5.2.2 CDRFLEEx.disconnect_rt_control

기능

실시간 외부 제어 연결을 해제합니다.

❶ 알아두기

- 연결 해제 시에 모든 설정을 초기화 합니다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```
Drfl.disconnect_rt_control(); //disconnect udp
```

5.3 정보 조회 함수

5.3.1 CDRFLEEx.get_rt_control_input_version_list

기능

실시간 외부 제어에서 지원하는 입력 데이터 (외부 제어기 → 로봇 제어기) 버전 리스트를 반환합니다.

인수

없음

리턴

값	설명
string	입력 데이터 버전 리스트. 콤마(,)로 구분.

버전 리스트

입력 데이터 버전	제어기 버전
v1.0	V2.9

예제

```

1 string version_list = Drfl.get_rt_control_input_version_list();
2 cout << version_list << endl;

```

5.3.2 CDRFLEEx.get_rt_control_output_version_list

기능

실시간 외부 제어에서 지원하는 출력 데이터 (로봇 제어기 → 외부 제어기) 버전 리스트를 반환합니다.

인수

없음

리턴

값	설명
String	출력 데이터 버전 리스트. 콤마(,)로 구분.

버전 리스트

입력 데이터 버전	버전
v1.0	V2.9

예제

```

1 string version_list = Drfl.get_rt_control_output_version_list();
2 cout << version_list << endl;

```

5.3.3 CDRFLEEx.get_rt_control_input_data_list

기능

특정 버전에서 지원하는 입력 데이터 리스트를 반환합니다.

인수

인수명	자료형	기본값	설명
strVersion	String	-	입력 데이터 버전

리턴

값	설명
string	입력 데이터 리스트. 콤마(,)로 구분.

입력 데이터 리스트

인수명	자료형	설명	최초 탑재 버전
external_force_torque	float[6]	external force/torque sensor	v1.0
external_digital_input	uint16	external digital input(16 channel)	v1.0
external_digital_output	uint16	external digital output(16 channel)	v1.0
external_analog_input	float[6]	external analog input(6 channel)	v1.0
external_analog_output	float[6]	external analog output(6 channel)	v1.0

예제

```

1 string data_list = Drfl.get_rt_control_input_data_list();
2 cout << version_list << endl;

```

5.3.4 CDRFLEx.get_rt_control_output_data_list

기능

특정 버전에서 지원하는 출력 데이터 리스트를 반환합니다.

인수

인수명	자료형	기본값	설명
strVersion	string	-	출력 데이터 버전

리턴

값	설명
string	출력 데이터 리스트. 콤마(,)로 구분.

출력 데이터 리스트

인수명	자료형	설명	최초 탑재 버전
time_stamp	double	timestamp at the data of data acquisition [s]	v1.0
actual_joint_position	float[6]	actual joint position from incremental encoder at motor side(used for control) [deg]	v1.0
actual_joint_position_ab_s	float[6]	actual joint position from absolute encoder at link side (used for exact link position) [deg]	v1.0
actual_joint_velocity	float[6]	actual joint velocity from incremental encoder at motor side [deg/s]	v1.0
actual_joint_velocity_ab_s	float[6]	actual joint velocity from absolute encoder at link side [deg/s]	v1.0
actual_tcp_position	float[6]	actual robot tcp position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler ZYZ notation [mm, deg]	v1.0
actual_tcp_velocity	float[6]	actual robot tcp velocity w.r.t. base coordinates [mm, deg/s]	v1.0
actual_flange_position	float[6]	actual robot flange position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler ZYZ notation [mm, deg]	v1.0
actual_flange_velocity	float[6]	robot flange velocity w.r.t. base coordinates [mm, deg/s]	v1.0
actual_motor_torque	float[6]	actual motor torque applying gear ratio = gear_ratio * current2torque_constant * motor current [Nm]	v1.0
actual_joint_torque	float[6]	estimated joint torque by robot controller [Nm]	v1.0
raw_joint_torque	float[6]	calibrated joint torque sensor data	v1.0

인수명	자료형	설명	최초 탑재 버전
raw_force_torque	float[6]	calibrated force torque sensor data w.r.t. flange coordinates [N, Nm]	v1.0
external_joint_torque	float[6]	estimated joint torque [Nm]	v1.0
external_tcp_force	float[6]	estimated tcp force w.r.t. base coordinates [N, Nm]	v1.0
target_joint_position	float[6]	target joint position [deg]	v1.0
target_joint_velocity	float[6]	target joint velocity [deg/s]	v1.0
target_joint_acceleration	float[6]	target joint acceleration [deg/s^2]	v1.0
target_motor_torque	float[6]	target motor torque [Nm]	v1.0
target_tcp_position	float[6]	target tcp position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler XYZ notation [mm, deg]	v1.0
target_tcp_velocity	float[6]	target tcp velocity w.r.t. base coordinates [mm, deg/s]	v1.0
jacobian_matrix	float[6][6]	jacobian matrix=J(q) w.r.t. base coordinates	v1.0
gravity_torque	float[6]	gravity torque=g(q) [Nm]	v1.0
coriolis_matrix	float[6][6]	coriolis matrix=C(q) [Nm.s]	v1.0
mass_matrix	float[6][6]	mass matrix=M(q)+B [Nm.s^2]	v1.0
solution_space	uint8	robot configuration	v1.0
singularity	float	minimum singular value	v1.0
operation_speed_rate	float	current operation speed rate(1~100 %)	v1.0
joint_temperature	float[6]	joint temperature(celsius)	v1.0

인수명	자료형	설명	최초 탑재 버전
controller_digital_input	uint16	controller digital input(16 channel)	v1.0
controller_digital_output	uint16	controller digital output(16 channel)	v1.0
controller_analog_input_type	uint8	controller analog input type(2 channel)	v1.0
controller_analog_input	float[2]	controller analog input(2 channel)	v1.0
controller_analog_output_type	uint8	controller analog output type(2 channel)	v1.0
controller_analog_output	float[2]	controller analog output(2 channel)	v1.0
flange_digital_input	uint8	flange digital input (A-Series: 2 channel, M/H-Series: 6 channel)	v1.0
flange_digital_output	uint8	flange digital output (A-Series: 2 channel, M/H-Series: 6 channel)	v1.0
flange_analog_input	float[4]	flange analog input (A-Series: 2 channel, M/H-Series: 4 channel)	v1.0
external_encoder_strobe_count	uint8[2]	strobe count (increased by 1 when detecting setting edge)	v1.0
external_encoder_count	uint32[2]	external encoder count	v1.0
goal_joint_position	float[6]	final goal joint position (reserved)	v1.0
goal_tcp_position	float[6]	final goal tcp position (reserved)	v1.0
robot_mode	uint8	ROBOT_MODE_MANUAL(0), ROBOT_MODE_AUTONOMOUS(1), ROBOT_MODE_MEASURE(2)	v1.0

인수명	자료형	설명	최초 탑재 버전
robot_state	uint8	STATE_INITIALIZING(0), STATE_STANDBY(1), STATE_MOVING(2), STATE_SAFE_OFF(3), STATE_TEACHING(4), STATE_SAFE_STOP(5), STATE_EMERGENCY_STOP(6), STATE_HOMMING(7), STATE_RECOVERY(8), STATE_SAFE_STOP2(9), STATE_SAFE_OFF2(10)	v1.0
control_mode	uint16	position control mode, torque mode	v1.0

예제

```

1 string data_list = Drfl.get_rt_control_input_data_list();
2 cout << version_list << endl;

```

5.4 설정 함수

5.4.1 CDRFLEEx.set_rt_control_input

기능

실시간 외부 제어에서 지원하는 입력 데이터 (외부 제어기 → 로봇 제어기) 통신 구성을 설정합니다.

인수

인수명	자료형	기본값	설명
strVersion	string	-	입력 데이터 버전
fPeriod	float	-	통신 주기 (sec). 범위: 0.001~1 sec
nLossCnt	int	-	연속해서 입력 데이터 또는 서보 제어 명령어가 설정한 카운트를 초과하여 손실되면 실시간 제어 연결을 해제합니다. 단, -1로 설정할 경우 Loss Count를 체크하지 않습니다.

● 알아두기

- fPeriod는 현재 0.001~1 sec를 지원합니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // Connect and configure input data
2 CDRFLEX drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 drfl.set_rt_control_input(version, period, losscount);

```

5.4.2 CDRFLEX.set_rt_control_output

기능

실시간 외부 제어에서 지원하는 출력 데이터 (로봇 제어기 → 외부 제어기) 통신 구성을 설정합니다.

인수

인수명	자료형	기본값	설명
strVersion	string	-	출력 데이터 버전
fPeriod	float	-	통신 주기 (sec). 범위: 0.001~1 sec
nLossCnt	int	-	Loss Count

● 알아두기

- fPeriod는 현재 0.001~1 sec를 지원합니다.
- nLossCnt는 현재 사용되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // Connect and configure output data
2 CDRFLEX drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 drfl.set_rt_control_output(version, period, losscount);

```

5.5 운용 함수

5.5.1 CDRFLEX.start_rt_control

기능

설정한 입/출력 데이터의 송/수신을 시작합니다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 // Connect and start
2 CDRFLEX drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 drfl.set_rt_control_output(version, period, losscount);
8 drfl.start_rt_control();

```

5.5.2 CDRFLEX.stop_rt_control

기능

설정한 입/출력 데이터의 송/수신을 종료합니다.

인수

없음

리턴

값	설명
0	오류
1	성공

예제

```

1 // Connect and start/stop
2 CDRFLEX drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 drfl.set_rt_control_output(version, period, losscount);
8 drfl.start_rt_control();
9 // do something
10 drfl.stop_rt_control();

```

5.5.3 CDRFLEx.set_on_rt_monitoring_data

기능

외부 제어기에서 로봇 제어기 출력 데이터를 수신했을 때 불리는 Callback 함수입니다.

알아두기

- 이 함수 내부에서 printf와 같은 non real-time 함수를 30Hz보다 빠르게 호출하면 안됩니다.

인수

없음

리턴

없음

예제

```

1 // callback function
2 void OnRTMonitoringData(LPRT_OUTPUT_DATA_LIST tData)
3 {
4     return;
5
6     static int td = 0;
7     if (td++==1000) {
8         td = 0;
9         printf("timestamp : %.3f\n", tData->time_stamp);
10        printf("actual_joint_position : %f %f %f %f %f %f\n",
11               tData->actual_joint_position[0], tData->actual_joint_position[1],
12               tData->actual_joint_position[2], tData->actual_joint_position[3],
13               tData->actual_joint_position[4], tData->actual_joint_position[5]);
14        printf("actual_motor_torque : %f %f %f %f %f %f\n",
15               tData->actual_motor_torque[0], tData->actual_motor_torque[1],
16               tData->actual_motor_torque[2], tData->actual_motor_torque[3],
17               tData->actual_motor_torque[4], tData->actual_motor_torque[5]);
18        printf("actual_grav_torque : %f %f %f %f %f %f\n",
19               tData->gravity_torque[0], tData->gravity_torque[1],
20               tData->gravity_torque[2], tData->gravity_torque[3],
21               tData->gravity_torque[4], tData->gravity_torque[5]);
22        printf("target torque : %f %f %f %f %f %f\n",
23               tData->target_motor_torque[0], tData->target_motor_torque[1],
24               tData->target_motor_torque[2], tData->target_motor_torque[3],
25               tData->target_motor_torque[4], tData->target_motor_torque[5]);
26    }
27 }
28

```

```

17 // main.cpp
18 CDRFLEEx drfl;
19 drfl.connect_rt_control();
20 string version = "v1.0";
21 float period = 0.001; // 1 msec
22 int losscount = 4;
23 Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
24 drfl.set_rt_control_output(version, period, losscount);
25 drfl.start_rt_control();

```

5.5.4 CDRFLEEx.read_data_rt

기능

외부 제어기에서 수신한 출력 데이터(로봇 제어기 → 외부 제어기)를 읽는 함수이다.

인수

없음

리턴

값	설명
LPRT_OUTPUT_DATA_LIST	출력 데이터 리스트 참조.

예제

```

1 // main.cpp
2 CDRFLEEx drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
8 drfl.set_rt_control_output(version, period, losscount);
9 drfl.start_rt_control();
10
11 float q[NUMBER_OF_JOINT] = {0.0, };
12 float q_dot[NUMBER_OF_JOINT] = {0.0, };
13 float trq_g[NUMBER_OF_JOINT] = {0.0, };
14 while (1)
15 {
16     time=(++count)*st;
17
18     memcpy(q, drfl.read_data_rt()->actual_joint_position, sizeof(float)*6)
19 ;

```

```

19     memcpy(q_dot, drfl.read_data_rt()->actual_joint_velocity, sizeof(float)
20 *6;
21     memcpy(trq_g, drfl.read_data_rt()->gravity_torque, sizeof(float)*6);
22     memcpy(trq_d, trq_g, sizeof(float)*6);
23
24     drfl.torque_rt(trq_d, 0);
25
26     if(time > plan1.time)
27     {
28         time=0;
29         Drfl.stop(STOP_TYPE_SLOW);
30         break;
31     }
32     rt_task_wait_period(NULL);
}

```

5.5.5 CDRFLEx.write_data_rt

기능

외부 제어기에서 송신할 입력 데이터(외부 제어기 → 로봇 제어기)를 쓰는 함수입니다. 외부 제어기와 연결된 센서, DIO, AIO, 기타 명령 등을 받아오기 위한 목적이며 현재 사용처는 없지만, 외부 주변 장치 메이커와의 협업을 통해 활용될 예정입니다.

인수

인수명	자료형	기본값	설명
fExternalForceTorque	Float[6]	-	External Force Torque
iExternalDI	unsigned short	-	External Digital Input
iExternalDO	unsigned short	-	External Digital Output
fExternalAnalogInput	float[6]		external analog input(6 channel)
fExternalAnalogOutput	Float[6]		external analog output(6 channel)

리턴

값	설명
LPRT_OUTPUT_DATA_LIST	출력 데이터 리스트 참조.

예제

```

1 float fExternalForceTorque[6] = {100, 100, 100, 100, 100, 100};
2 int iExternalDI = 1;
3 int iExternalDO = 2;
4 float fExternalAnalogInput[6] = {100, 100, 100, 100, 100, 100};
5 float fExternalAnalogOutput[6] = {0, 0, 0, 0, 0, 0};
6
7 Drfl.write_data_rt(fExternalForceTorque, iExternalDI, iExternalDO,
fExternalAnalogInput, fExternalAnalogOutput);

```

5.6 서브 모션 함수

서보 모션 함수는 두산에서 제공하는 모션을 사용하지 않고, 외부 제어기에서 모션/힘 제어 알고리즘을 구현하여 사용할 수 있는 함수입니다. 실시간 외부 제어기 시스템을 구축하고 해당 제어기에서 알고리즘을 구현하면 로봇 제어기는 이를 단순 전달 또는 보간하여 처리하는 개념입니다.

5.6.1 알아두기

- 실시간 외부 제어의 서보 모션 함수(ex. servoj_rt, servol_rt, speedj_rt, speedl_rt)는 외부 제어기에서 실시간 OS 등을 활용하여 실시간 환경을 구축해야 전체 시스템의 실시간성이 보장됩니다. 또한, 중간 프로파일을 외부에서 계산할 수 있는 경우 의미가 있습니다. 실시간 환경을 구축할 수 없거나 중간 프로파일을 외부에서 계산할 수 없는 경우에는 TCP/IP 기반의 기존 api에서 일반 모션의 서보 모션 함수(ex. servoj, servol, speedj, speedl)를 사용하세요. 일반 모션의 서보 모션 함수는 타겟값만 실시간으로 바꿔주면 로봇 제어기에서 중간 프로파일을 계산해 줍니다.

5.6.2 CDRFLE.set_velj_rt

기능

실시간 외부 제어의 서보 모션에서 사용하는 전역 조인트 속도 제한값을 설정합니다.

인수

인수명	자료형	기본값	설명
vel	float[6]	-	조인트 속도 제한값 [deg/s]

● 알아두기

- 서보 모션 함수를 사용해서 움직이는 중의 속도가 설정한 전역 속도값보다 크면 Info를 발생시킵니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float vel[6] = {100, 100, 100, 100, 100, 100};
2 Drfl.set_velj_rt(vel);

```

5.6.3 CDRFLEx.set_accj_rt

기능

실시간 외부 제어의 서보 모션에서 사용하는 전역 조인트 가속도 제한값을 설정합니다.

인수

인수명	자료형	기본값	설명
acc	float[6]	-	조인트 가속도 제한값 [deg/s]

❶ 알아두기

- 서보 모션 함수를 사용해서 움직이는 중의 속도가 설정한 전역 속도값보다 크면 Info를 발생시킵니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float acc[6] = {100, 100, 100, 100, 100, 100};
2 Drfl.set_accj_rt(acc);

```

5.6.4 CDRFLEEx.set_velx_rt

기능

실시간 외부 제어의 서보 모션에서 사용하는 전역 태스크 속도 제한값을 설정합니다.

인수

인수명	자료형	기본값	설명
fTransVel	float	-	태스크 Linear 속도 제한값 [mm/s]
fRotationVel	float	-	태스크 Rotational 속도 제한값 [deg/s]. None(-10000) 값이 입력되면, 태스크 Linear 속도 제한값으로부터 자동 계산됩니다.

알아두기

- 서보 모션 함수를 사용해서 움직이는 중의 태스크 속도가 설정한 전역 태스크 속도값보다 크면 Info 를 발생시킵니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float vel[6] = {100, 100, 100, 100, 100, 100};
2 Drfl.set_velx_rt(vel);

```

5.6.5 CDRFLEx.set_accx_rt

기능

실시간 외부 제어의 서보 모션에서 사용하는 전역 태스크 가속도 제한값을 설정합니다.

인수

인수명	자료형	기본값	설명
fTransAcc	float	-	태스크 Linear 가속도 제한값 [mm/s^2]
fRotationAcc	float	-	태스크 Rotational 가속도 제한값 [deg/s^2]. None(-10000) 값이 입력되면, 태스크 Linear 가속도 제한값으로부터 자동 계산됩니다.

❶ 알아두기

- 서보 모션 함수를 사용해서 움직이는 중의 태스크 속도가 설정한 전역 태스크 가속도값보다 크면 Info를 발생시킵니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float acc[6] = {100, 100, 100, 100, 100, 100};
2 Drfl.set_accx_rt(acc);

```

5.6.6 CDRFLEx.servoj_rt

기능

외부 제어기에서 조인트 위치 제어를 하는 함수입니다.

⚠ 주의

- 현재 **servoj_rt** 명령어는 안정화가 되지 않았습니다. 통신 상황에 따라서 Jerky한 모션이 나올 수 있어서, 타겟 시간(fTargetTime)을 튜닝하여 느린 반응 속도로 사용하거나, speedj_rt 명령어를 이용하여 위치 제어하는 것을 추천합니다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	타겟 조인트 위치 [deg]
fTargetVel	float[6]	-	타겟 조인트 속도 [deg/s]. null값을 입력하면, 입력했던 타겟 조인트 위치 기반으로 자동 계산됩니다.
fTargetAcc	float[6]	-	타겟 조인트 가속도 [deg/s ²]. null값을 입력하면, 입력했던 타겟 조인트 위치 기반으로 자동 계산됩니다.
fTargetTime	float	-	타겟 시간 [s]

ℹ 알아두기

- Asnyc 명령어입니다. 호출된 후 다음 명령줄로 넘어갑니다.
- 내부 프로파일은 fTargetTime에 (fTargetPos, fTargetVel, fTargetAcc)에 도달하도록 보간됩니다.
- fTargetTime <= controller's control period(=1ms) 입력하면 보간없이 바로 해당 위치로 제어합니다.
- 모션 중 다음 명령이 들어오지 않으면, 설정된 전역 가속도 제한값 기준으로 감속합니다.

⚠ 주의

- 현재 **servoj_rt** 명령어가 안정화되지 않아서 Jerky 한 모션이 나올 수 있습니다. 시스템 상황에 따라서 fTargetTime을 통신 주기보다 크게 설정하여 사용하십시오. 현재 버전에서는 1 [ms] 주기로 **servoj_rt** 명령어를 호출하는 기준으로 fTargetTime >= 20 [ms] 보다 크게 설정하여 사용하거나 speedj_rt를 이용하여 위치 제어하는 것을 권장합니다.
- 현재 버전에서는 Operation Speed [%] 와 연동되지 않습니다.

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

1 // main.cpp
2 CDRFLEX drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
8 drfl.set_rt_control_output(version, period, losscount);
9 drfl.start_rt_control();
10
11 float time = 0.0;
12 int count = 0;
13 float ratio = 20.0;
14 float q[NUMBER_OF_JOINT] = {0.0, };
15 float q_dot[NUMBER_OF_JOINT] = {0.0, };
16 float q_d[NUMBER_OF_JOINT] = {0.0, };
17 float q_dot_d[NUMBER_OF_JOINT] = {0.0, };
18 float integral_v_error[NUMBER_OF_JOINT] = {0.0, };
19 while (1)
20 {
21     time=(++count)*period;
22     // get current state
23     memcpy(q, drfl.read_data_rt()->actual_joint_position, sizeof(float)*6)
24     ;
25     memcpy(q_dot, drfl.read_data_rt()->actual_joint_velocity, sizeof(float)
26 *6);
27
28     // make trajectory
29     TrajectoryGenerator(q_d, q_dot_d, q_ddot_d); // Custom Trajectory
30     Generation Function
31
32     // or make target position and target velocity, acceleration can be
33     omitted
34     TrajectoryGenerator(q_d);
35     for (int i=0; i<NUMBER_OF_JOINT; i++) {
36         q_dot_d[i] = -10000;
37         q_ddot_d[i] = -10000;
38     }
39
40     drfl.servoj_rt(q_d, q_dot_d, q_ddot_d, ratio*period); // Currently
41     tuning ratio
42
43     if(time > plan1.time)
44     {

```

```

40     time=0;
41     Drfl.stop(STOP_TYPE_SLOW);
42     break;
43 }
44
45     rt_task_wait_period(NULL); // RTOS function
46 }
```

5.6.7 CDRFLEx.servol_rt

기능

외부 제어기에서 태스크 위치 제어를 하는 함수입니다.

⚠ 주의

- 현재 **servol_rt** 명령어는 안정화가 되지 않았습니다. 통신 상황에 따라서 Jerky한 모션이 나올 수 있어서, 타겟 시간(fTargetTime)을 튜닝하여 느린 반응 속도로 사용하거나, speedl_rt 명령어를 이용하여 위치 제어하는 것을 추천합니다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	타겟 태스크 위치 [mm, deg] (rotation: euler zyz)
fTargetVel	float[6]	-	타겟 태스크 속도. [mm/s, deg/s] null값을 입력하면, 입력했던 타겟 태스크 위치 기반으로 자동 계산됩니다.
fTargetAcc	float[6]	-	타겟 태스크 가속도. [mm/s, deg/s] null값을 입력하면, 입력했던 타겟 태스크 위치 기반으로 자동 계산됩니다.
fTargetTime	float	-	타겟 시간 [s]

ℹ 알아두기

- Asnyc 명령어입니다. 호출된 후 다음 명령줄로 넘어갑니다.
- 내부 프로파일은 fTargetTime 에 (fTargetPos, fTargetVel, fTargetAcc) 에 도달하도록 보간됩니다.
- fTargetTime <= controller's control period(=1ms) 인 경우 보간없이 바로 해당 속도로 제어합니다.
- 모션 중 다음 명령이 들어오지 않으면, 설정된 전역 가속도 제한값 기준으로 감속합니다.

주의

- 현재 `servol_rt` 명령어가 안정화되지 않아서 `Jerky` 한 모션이 나올 수 있습니다. 시스템 상황에 따라서 `fTargetTime`을 통신 주기보다 크게 설정하여 사용하십시오. 현재 버전에서는 1 [ms] 주기로 `servol_rt` 명령어를 호출하는 기준으로 `fTargetTime >= 20 [ms]` 보다 크게 설정하여 사용하거나 `speedl_rt`를 이용하여 위치 제어하는 것을 권장합니다.
- 현재 버전에서는 Operation Speed [%] 와 연동되지 않습니다.
- 현재 버전에서는 힘/강성 제어 함수와 연동되지 않습니다.
- 현재 버전에서 특이점 옵션 중 DR_VAR_VEL 와 연동되지 않습니다. 설정할 경우 자동으로 DR_AVOID 옵션으로 설정됩니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 float fTargetPos[6] = {1500, 3, 100, 0, 0, 0};
2 float fTargetVel[6] = {100, 100, 100, 100, 100, 100};
3 float fTargetAcc[6] = {100, 100, 100, 100, 100, 100};
4 float fTargetTime = 6;
5 Drfl.servol_rt(fTargetPos, fTargetVel, fTargetAcc, fTargetTime);

```

5.6.8 CDRFLEX.speedj_rt

기능

외부 제어기에서 조인트 속도 제어를 하는 함수입니다.

인수

인수명	자료형	기본값	설명
fTargetVel	float[6]	-	타겟 조인트 속도. [deg/s]

인수명	자료형	기본값	설명
fTargetAcc	float[6]	-	타겟 조인트 가속도 [deg/s ²]. null값을 입력하면, 입력했던 타겟 조인트 위치 기반으로 자동 계산됩니다.
fTargetTime	float	-	타겟 시간 [s]

❶ 알아두기

- Async 명령어입니다. 호출된 후 다음 명령줄로 넘어갑니다.
- 내부 프로파일은 fTargetTime에 (fTargetVel, fTargetAcc)에 도달하도록 보간됩니다.
- fTargetTime <= controller's control period(=1ms)인 경우 보간없이 바로 해당 속도로 제어합니다.
- fTargetVel에 도달할 때까지 다음 명령이 들어오지 않으면, 마지막 들어온 속도를 유지합니다.
- 단, 안전을 위해서 1[s] 동안 다음 명령이 들어오지 않으면 Time-Out으로 에러를 발생시키고 정지합니다.
- 모션 중 전역으로 설정된 가속도 제한값을 초과할 경우, 모션을 정지하지는 않고 Info 메시지를 발생시킵니다.

⚠ 주의

- 현재 버전에서는 Operation Speed [%]와 연동되지 않습니다.

리턴

값	설명
0	오류
1	성공

예제

```

1 // main.cpp
2 CDRFLEEx drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
8 drfl.set_rt_control_output(version, period, losscount);
9 drfl.start_rt_control();
10
11 float time = 0.0;

```

```

12 int count = 0;
13 float q[NUMBER_OF_JOINT] = {0.0, };
14 float q_dot[NUMBER_OF_JOINT] = {0.0, };
15 float q_d[NUMBER_OF_JOINT] = {0.0, };
16 float q_dot_d[NUMBER_OF_JOINT] = {0.0, };
17 float vel_d[NUMBER_OF_JOINT] = {0.0, };
18 float acc_d[NUMBER_OF_JOINT] = {0.0, };
19 float kv[NUMBER_OF_JOINT] = {1.0, 1.0, 1.0, 1.0, 1.0, 1.0}; // have to
tune
20 float kp[NUMBER_OF_JOINT] = {1.0, 1.0, 1.0, 1.0, 1.0, 1.0}; // have to
tune
21 float ki[NUMBER_OF_JOINT] = {1.0, 1.0, 1.0, 1.0, 1.0, 1.0}; // have to
tune
22 float integral_v_error[NUMBER_OF_JOINT] = {0.0, };
23 while (1)
{
24     time=(++count)*period;
// get current state
25     memcpy(q, drfl.read_data_rt()->actual_joint_position, sizeof(float)*6)
;
26     memcpy(q_dot, drfl.read_data_rt()->actual_joint_velocity, sizeof(float)
*6);

27     // make trajectory
28     TrajectoryGenerator(q_d, q_dot_d); // Custom Trajectory Generation
Function
29
30     // velocity feedforward + pi controller
31     for (int i=0; i<6; i++) {
32         q_dot_v[i] = kv[i] * (q_d[i] - q[i]);
33         integral_v_error[i] += q_dot_v[i] - q_dot[i];
34         vel_d[i] = q_dot_d[i] + kp[i]*(q_dot_v[i] - q_dot[i]) +
ki[i]*integral_v_error[i];
35         acc_d[i] = -10000;
36     }
37     drfl.speedj_rt(vel_d, acc_d[i], period);
38
39     if(time > plan1.time)
40     {
41         time=0;
42         Drfl.stop(STOP_TYPE_SLOW);
43         break;
44     }
45
46     rt_task_wait_period(NULL); // RTOS function
47 }
48
49
50 }
```

5.6.9 CDRFLEx.speedl_rt

기능

외부 제어기에서 태스크 속도 제어를 하는 함수입니다.

인수

인수명	자료형	기본값	설명
fTargetPos	float[6]	-	타겟 태스크 속도. [mm/s, deg/s]
fTargetVel	float[6]	-	타겟 태스트 가속도. [mm/s, deg/s] null 값을 입력하면, 입력했던 타겟 태스크 위치 기반으로 자동 계산됩니다.
fTargetAcc	float[6]	-	타겟 시간 [s]

● 알아두기

- Asnyc 명령어입니다. 호출된 후 다음 명령줄로 넘어갑니다.
- 내부 프로파일은 fTargetTime 에 (fTargetVel, fTargetAcc) 에 도달하도록 보간됩니다.
- fTargetTime <= controller's control period(=1ms) 인 경우 보간없이 바로 해당 속도로 제어합니다.
- fTargetVel에 도달할 때까지 다음 명령이 들어오지 않으면, 마지막 들어온 속도를 유지합니다.
- 단, 안전을 위해서 1[s] 동안 다음 명령이 들어오지 않으면 Time-Out으로 에러를 발생시키고 정지합니다.
- 모션 중 전역으로 설정된 가속도 제한값을 초과할 경우, 모션을 정지하지는 않고 Info 메시지를 발생시킵니다.

⚠ 주의

- 현재 버전에서는 Operation Speed [%] 와 연동되지 않습니다.
- 현재 버전에서는 힘/강성 제어 함수와 연동되지 않습니다.
- 현재 버전에서 특이점 옵션 중 DR_VAR_VEL 와 연동되지 않습니다. 설정할 경우 자동으로 DR_AVOID 옵션으로 설정됩니다.

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

1 float fTargetPos[6] = {1500, 3, 100, 0, 0, 0};
2 float fTargetVel[6] = {100, 100, 100, 100, 100, 100};
3 float fTargetAcc[6] = {100, 100, 100, 100, 100, 100};
4 float fTargetTime = 6;
5 Drfl.servol_rt(fTargetPos, fTargetVel, fTargetAcc, fTargetTime);

```

5.6.10 CDRFLEx.torque_rt

기능

외부 제어기에서 모터 토크 제어를 하는 함수입니다.

인수

인수명	자료형	기본값	설명
fMotorTor	float[6]	-	타겟 모터 토크 [Nm]
fTargetTime	float	-	타겟 시간 [s]

● 알아두기

- Async 명령어입니다. 호출된 후 다음 명령줄로 넘어갑니다.
- fTargetTime에 fMotorTor이 되도록 제어합니다.
- fTargetTime <= controller's control period(=1ms)인 경우 보간없이 바로 해당 모터 토크로 제어합니다.
- fMotorTor에 도달할 때까지 다음 명령이 들어오지 않으면, 마지막 들어온 토크를 유지합니다.
- 단, 안전을 위해서 1[s] 동안 다음 명령이 들어오지 않으면 Time-Out으로 에러를 발생시키고 정지합니다.

⚠ 주의

- H 시리즈의 경우, 2축에 중력 보상 장치가 있어서 입력한 모터 토크와 중력 보상 장치에서 발생하는 토크가 더해져서 로봇을 움직입니다. 외부 제어기로 출력하는 gravity_torque는 (=link gravity torque - 중력 보상 장치 토크)를 뜻합니다.

리턴

값	설명
0	오류

예제

```

1 // main.cpp
2 CDRFLEx drfl;
3 drfl.connect_rt_control();
4 string version = "v1.0";
5 float period = 0.001; // 1 msec
6 int losscount = 4;
7 Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
8 drfl.set_rt_control_output(version, period, losscount);
9 drfl.start_rt_control();
10
11 float time = 0.0;
12 int count = 0;
13 float q[NUMBER_OF_JOINT] = {0.0, };
14 float q_dot[NUMBER_OF_JOINT] = {0.0, };
15 float q_d[NUMBER_OF_JOINT] = {0.0, };
16 float q_dot_d[NUMBER_OF_JOINT] = {0.0, };
17 float trq_g[NUMBER_OF_JOINT] = {0.0, };
18 float trq_d[NUMBER_OF_JOINT] = {0.0, };
19 float kp[NUMBER_OF_JOINT] = {1.0, 1.0, 1.0, 1.0, 1.0, 1.0}; // have to
tune
20 float kd[NUMBER_OF_JOINT] = {1.0, 1.0, 1.0, 1.0, 1.0, 1.0}; // have to
tune
21 while (1)
22 {
23     time=(++count)*period;
24     // get current state
25
26     memcpy(q, drfl.read_data_rt()->actual_joint_position, sizeof(float)*6)
;
27     memcpy(q_dot, drfl.read_data_rt()->actual_joint_velocity, sizeof(float)
*6);
28     memcpy(trq_g, drfl.read_data_rt()->gravity_torque, sizeof(float)*6);
29
30     // make trajectory
31     TrajectoryGenerator(q_d, q_dot_d); // Custom Trajectory Generation
Function
32
33     // gravity compensation + pd control
34     for (int i=0; i<6; i++) {
35         trq_d[i] = trq_g[i] + kp[i]*(q_d[i] - q[i]) + kd[i]*(q_dot_d[i] -
q_dot[i]);

```

```
36     }
37     drfl.torque_rt(trq_d, 0);
38
39     if(time > plan1.time)
40     {
41         time=0;
42         Drfl.stop(STOP_TYPE_SLOW);
43         break;
44     }
45
46     rt_task_wait_period(NULL); // RTOS function
47 }
```

6 어플리케이션 명령어(Application Command)

6.1 용접(Welding)

i 용접 기능은 V2.x 제어기에만 지원합니다. #define DRCF_VERSION = 2 지정 후 DRFL 헤더를 임포트 해주세요.

```
#define DRCF_VERSION 2
#include <DRFLEx.h>
```

- [set_on_monitoring_welding_data\(p. 333\)](#)
- [set_on_monitoring_analog_welding_data\(p. 334\)](#)
- [set_on_monitoring_digital_welding_data\(p. 335\)](#)
- [app_weld_weave_cond_trapezoidal\(p. 338\)](#)
- [app_weld_weave_cond_zigzag\(p. 339\)](#)
- [app_weld_weave_cond_circular\(p. 341\)](#)
- [app_weld_weave_cond_sinusoidal\(p. 342\)](#)
- [app_weld_enable_analog\(p. 343\)](#)
- [app_weld_set_weld_cond_analog\(p. 347\)](#)
- [app_weld_adj_welding_cond_analog\(p. 348\)](#)
- [app_weld_set_interface_eip_m2r_process\(p. 350\)](#)
- [app_weld_set_interface_eip_r2m_mode\(p. 353\)](#)
- [app_weld_set_interface_eip_r2m_process\(p. 355\)](#)
- [app_weld_set_interface_eip_r2m_test\(p. 357\)](#)
- [app_weld_set_interface_eip_r2m_condition\(p. 359\)](#)
- [app_weld_set_interface_eip_r2m_option\(p. 361\)](#)
- [app_weld_set_interface_eip_m2r_process2\(p. 363\)](#)
- [app_weld_set_interface_eip_m2r_monitoring\(p. 366\)](#)
- [app_weld_set_interface_eip_m2r_other\(p. 368\)](#)
- [app_weld_reset_interface\(p. 370\)](#)
- [app_weld_enable_digital\(p. 371\)](#)
- [app_weld_set_weld_cond_digital\(p. 371\)](#)
- [app_weld_adj_welding_cond_digital\(p. 373\)](#)
- [measure_welding_tcp\(p. 374\)](#)
- [set_welding_cockpit_setting\(p. 375\)](#)
- [set_digital_welding_monitoring_mode\(p. 376\)](#)
- [app_weld_adj_motion_offset\(p. 377\)](#)
- [set_welding_cockpit_setting_time_setting\(p. 378\)](#)

6.1.1 set_on_monitoring_welding_data

기능

하위 제어기에서 상위 제어기로 요청한 용접 상태 정보를 전달하기 위하여 전송한다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringWeldingDataCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

void MyWeldingDataCallback(const LPROBOT_WELDING_DATA pData)
{
    std::cout << "Adj Available: " << static_cast<int>(pData->_iAdjAvail) <<
    std::endl;
    std::cout << "Target Voltage: " << pData->_fTargetVol << std::endl;
    std::cout << "Target Current: " << pData->_fTargetCur << std::endl;
    std::cout << "Target Velocity: " << pData->_fTargetVel << std::endl;
    std::cout << "Actual Voltage: " << pData->_fActualVol << std::endl;
    std::cout << "Actual Current: " << pData->_fActualCur << std::endl;
    std::cout << "Offset Y: " << pData->_fOffsetY << std::endl;
    std::cout << "Offset Z: " << pData->_fOffsetZ << std::endl;
    std::cout << "Arc On: " << static_cast<int>(pData->_iArcOnDO) << std::endl;
    std::cout << "Gas On: " << static_cast<int>(pData->_iGasOnDO) << std::endl;
    std::cout << "Inching Plus: " << static_cast<int>(pData->_iInchPDO) <<
    std::endl;
    std::cout << "Inching Minus: " << static_cast<int>(pData->_iInchNPO) <<
    std::endl;
    std::cout << "Status: " << static_cast<int>(pData->_iStatus) << std::endl;
}

int main()
{
    Drfl.set_on_monitoring_welding_data(MyWeldingDataCallback);
}

```

6.1.2 set_on_monitoring_analog_welding_data

기능

하위 제어기에서 상위 제어기로 요청한 아날로그 용접 상태 정보를 전달하기 위하여 전송한다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringAnalogWelding DataCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

void AnalogWeldingDataCallback(const LPROBOT_ALALOG_WELDING_DATA pData)
{
    std::cout << "Adj Available: " << static_cast<int>(pData->_iAdjAvail) <<
std::endl;
    std::cout << "Target Voltage: " << pData->_fTargetVol << std::endl;
    std::cout << "Target Current: " << pData->_fTargetCur << std::endl;
    std::cout << "Target Velocity: " << pData->_fTargetVel << std::endl;
    std::cout << "Actual Voltage: " << pData->_fActualVol << std::endl;
    std::cout << "Actual Current: " << pData->_fActualCur << std::endl;
    std::cout << "Offset Y: " << pData->_fOffsetY << std::endl;
    std::cout << "Offset Z: " << pData->_fOffsetZ << std::endl;
    std::cout << "Arc On: " << static_cast<int>(pData->_iArcOnDO) << std::endl;
    std::cout << "Gas On: " << static_cast<int>(pData->_iGasOnDO) << std::endl;
    std::cout << "Inching Plus: " << static_cast<int>(pData->_iInchPDO) <<
std::endl;
    std::cout << "Inching Minus: " << static_cast<int>(pData->_iInchNPO) <<
std::endl;
    std::cout << "Status: " << static_cast<int>(pData->_iStatus) << std::endl;
    std::cout << "Blow Out: " << static_cast<int>(pData->_iBlowOut) << std::endl;
    std::cout << "Feeding Velocity: " << pData->_iFeedingVel << std::endl;
}

int main()
{
    Drfl.set_on_monitoring_analog_welding_data(AnalogWeldingDataCallback);
}

```

6.1.3 set_on_monitoring_digital_welding_data

기능

하위 제어기에서 상위 제어기로 요청한 디지털 용접 상태 정보를 전달하기 위하여 전송한다.

인수

인수명	자료형	기본값	설명
pCallbackFunc	TOnMonitoringDigitalWeldingDataCB	-	콜백 함수 정의 참조

리턴

없음

예제

```

void DigitalWeldingDataCallback(const LPROBOT_DIGITAL_WELDING_DATA pData)
{
    std::cout << "Adj Available: " << static_cast<int>(pData->_iAdjAvail) <<
    std::endl;
    std::cout << "Target Voltage: " << pData->_fTargetVol << std::endl;
    std::cout << "Target Current: " << pData->_fTargetCur << std::endl;
    std::cout << "Target Velocity: " << pData->_fTargetVel << std::endl;
    std::cout << "Actual Voltage: " << pData->_fActualVol << std::endl;
    std::cout << "Actual Current: " << pData->_fActualCur << std::endl;
    std::cout << "Offset Y: " << pData->_fOffsetY << std::endl;
    std::cout << "Offset Z: " << pData->_fOffsetZ << std::endl;
    std::cout << "Arc On: " << static_cast<int>(pData->_iArcOnDO) << std::endl;
    std::cout << "Gas On: " << static_cast<int>(pData->_iGasOnDO) << std::endl;
    std::cout << "Inching Plus: " << static_cast<int>(pData->_iInchPDO) <<
    std::endl;
    std::cout << "Inching Minus: " << static_cast<int>(pData->_iInchNPO) <<
    std::endl;
    std::cout << "Status: " << static_cast<int>(pData->_iStatus) << std::endl;
    std::cout << "Blow Out: " << static_cast<int>(pData->_iBlowOut) << std::endl;
    std::cout << "Feeding Velocity: " << pData->_fFeedingVel << std::endl;
    std::cout << "Actual Feeding Velocity: " << pData->_fActualFeedingVel <<
    std::endl;
    std::cout << "Error Number: " << pData->_iErrorNumber << std::endl;
    std::cout << "Wire Stick: " << pData->_fWireStick << std::endl;
    std::cout << "Error: " << pData->_iError << std::endl;
    std::cout << "Option 1: " << pData->_fOption1 << std::endl;
    std::cout << "Option 2: " << pData->_fOption2 << std::endl;
    std::cout << "Option 3: " << pData->_fOption3 << std::endl;
    std::cout << "Option 4: " << pData->_fOption4 << std::endl;
    std::cout << "Option 5: " << pData->_fOption5 << std::endl;
    std::cout << "Option 6: " << pData->_fOption6 << std::endl;
    std::cout << "Option 7: " << pData->_fOption7 << std::endl;
    std::cout << "Option 8: " << pData->_fOption8 << std::endl;
    std::cout << "Option 9: " << pData->_fOption9 << std::endl;
    std::cout << "Option 10: " << pData->_fOption10 << std::endl;
    std::cout << "Current Flow: " << static_cast<int>(pData->_iCurrentFlow) <<
    std::endl;
    std::cout << "Process Active: " << static_cast<int>(pData->_iProcessActive)
    << std::endl;
    std::cout << "Machinery Ready: " << static_cast<int>(pData->_iMachineryReady)
    << std::endl;
    std::cout << "Voltage Correction: " << pData->_fVoltageCorrection <<
    std::endl;
    std::cout << "Dynamic Correction: " << pData->_fDynamicCorrection <<
    std::endl;
}

```

```
int main()
{
    Drfl.set_on_monitoring_digital_welding_data(DigitalWeldingDataCallback);
}
```

6.1.4 app_weld_weave_cond_trapezoidal

기능

사다리꼴 위빙조건을 설정합니다. 위빙조건은 용접기능 활성화(app_weld_enable_analog()) / app_weld_enable_digital())부터 비활성화(app_weld_disable_analog() / app_weld_disable_digital())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 위빙조건은 용접경로의 진행방향을 위빙x축으로, 위빙x축에서 TCP-z방향을 벡터곱(cross-product)한 방향을 위빙y축으로 정의하는 위빙좌표계에서 정의됩니다. 좌표계 및 위빙 설정 인자는 아래 그림을 참조하세요. 하나의 용접구간 내에서는 하나의 위빙조건만 허용되며 용접 중 app_weld_adj_welding_cond_analog() / app_weld_set_weld_cond_digital() 명령어를 통해 옵셋 또는 위빙폭을 조정하거나, 티칭펜던트의 용접조건 조정 popup에서 (전압/전류/속도 및) 옵셋을 조정할 수 있습니다. 다만 티칭펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET상태 (즉, app_weld_set_weld_cond_analog() / app_weld_set_weld_cond_digital()으로 지정한 용접조건 설정) 인 경우에만 가능합니다.

인수

인수명	자료형	기본값	설명
wv_offset	float[2]	[0, 0]	[0] 위빙좌표계-y방향 옵셋 (mm) [1] 위빙좌표계-z방향 옵셋 (mm)
wv_angle	float	0	위빙좌표계-x축기준 위빙평면 회전각 (deg)
wv_param	list(float[10])	[0, 1.5, 0, -1.5, 0.3, 0.1, 0.3, 0.3, 0.1, 0.3]	[0] 위빙점1-x (mm) [1] 위빙점1-y (mm) [2] 위빙점2-x (mm)[3] 위빙점2-y (mm) [4] 위빙점1 → 2시간(sec)[5] 위빙점1 → 2가감속시간(sec)[6] 위빙점1 드웰시간(sec)[7] 위빙점2 → 1시간(sec)[8] 위빙점2 → 1가감속시간(sec)[9] 위빙점2 드웰시간(sec)

리턴

값	설명
0	오류
1	성공

예제

```
CONFIG_TRAPEZOID_WEAVERS_SETTING weaving_trap;

weaving_trap._fOffsetY = 0;
weaving_trap._fOffsetZ = 0;
weaving_trap._fGradient = 0;
weaving_trap._fwPT1[0] = 0;
weaving_trap._fwPT1[1] = 3;
weaving_trap._fwPT2[0] = 0;
weaving_trap._fwPT2[1] = -3;
weaving_trap._fwT1 = 0.3;
weaving_trap._fwTAcc1 = 0.1;
weaving_trap._fwTTD1 = 0.2;
weaving_trap._fwT2 = 0.3;
weaving_trap._fwTAcc2 = 0.1;
weaving_trap._fwTTD2 = 0.2;

Drfl.app_weld_weave_cond_trapezoidal(weaving_trap);
```

6.1.5 app_weld_weave_cond_zigzag

기능

지그재그 위빙조건을 설정합니다. 위빙조건은 용접기능 활성화(app_weld_enable_analog())/app_weld_enable_digital()부터 비활성화(app_weld_disable_analog()/app_weld_disable_digital())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 위빙조건은 용접경로의 진행방향을 위빙x축으로, 위빙x축에서 TCP-z방향을 벡터곱(cross-product)한 방향을 위빙y축으로 하는 위빙좌표계에서 정의됩니다. 좌표계 및 위빙 설정 인자는 아래 그림을 참조하세요. 하나의 용접구간 내에서는 하나의 위빙조건만 허용되며 용접 중 app_weld_adj_welding_cond_analog() / app_weld_set_weld_cond_digital() 명령어를 통해 옵셋 또는 위빙폭을 조정하거나, 티칭펜던트의 용접조건 조정 popup에서 (전압/전류/속도 및) 옵셋을 조정할 수 있습니다. 다만 티칭펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET상태 (즉, app_weld_set_weld_cond_analog() / app_weld_set_weld_cond_digital()으로 지정한 용접조건 설정) 인 경우에만 가능합니다.

인수

인수명	자료형	기본값	설명
fOffsetY	float	0.0	위빙 좌표계의 Y 방향 옵셋 (mm)
fOffsetZ	float	0.0	위빙 좌표계의 Z 방향 옵셋 (mm)
fGradient	float	0.0	위빙 좌표계의 X 축 기준 위빙 평면 회전각 (deg)
fWeavingWidth	float	5.0	위빙 폭 (mm)
fWeavingCycle	float	0.7	위빙 주기 (sec)

리턴

값	설명
0	오류
1	성공

예제

```

float fOffsetY = 0.0;
float fOffsetZ = 0.0;
float fGradient = 0.0;
float fWeavingWidth = 5.0;
float fWeavingCycle = 0.7;

Drfl.app_weld_weave_cond_zigzag(fOffsetY, fOffsetZ, fGradient, fWeavingWidth,
fWeavingCycle);

```

6.1.6 app_weld_weave_cond_circular

기능

원형 위빙조건을 설정합니다. 위빙조건은 용접기능 활성화(app_weld_enable_analog()) / app_weld_enable_digital()부터 비활성화(app_weld_disable_analog() / app_weld_disable_digital())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 위빙조건은 용접경로의 진행방향을 위빙x축으로, 위빙x축에서 TCP-z방향을 벡터곱(cross-product)한 방향을 위빙y축으로 하는 위빙좌표계에서 정의됩니다. 좌표계 및 위빙 설정 인자는 아래 그림을 참조하세요. 하나의 용접구간 내에서는 하나의 위빙조건만 허용되며 용접 중 app_weld_adj_welding_cond_analog() / app_weld_set_weld_cond_digital() 명령어를 통해 옵셋 또는 위빙폭을 조정하거나, 티칭펜던트의 용접조건 조정 popup에서 (전압/전류/속도 및) 옵셋을 조정할 수 있습니다. 다만 티칭펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET상태(즉, app_weld_set_weld_cond_analog() / app_weld_set_weld_cond_digital()으로 지정한 용접조건 설정)인 경우에만 가능합니다.

인수

인수명	자료형	기본값	설명
fOffsetY	float	0.0	위빙 좌표계의 Y 방향 옵셋 (mm)
fOffsetZ	float	0.0	위빙 좌표계의 Z 방향 옵셋 (mm)
fGradient	float	0.0	위빙 좌표계의 X 축 기준 위빙 평면 회전각 (deg)
fwWdt	float [2]	{3.0, 3.0}	[0] 위빙 폭1 (mm) [1] 위빙 폭2 (mm)
fwT	float [2]	{0.3, 0.3}	[0] 위빙 주기1 (sec) [1] 위빙 주기2 (sec)

리턴

값	설명
0	오류
1	성공

예제

```
float fOffsetY = 0.0;
float fOffsetZ = 0.0;
float fGradient = 0.0;
float fwWdt[2] = {3.0, 3.0};
float fwT[2] = {0.3, 0.3};

Drfl.app_weld_weave_cond_circular(fOffsetY, fOffsetZ, fGradient, fwWdt, fwT);
```

6.1.7 app_weld_weave_cond_sinusoidal

기능

사인 위빙조건을 설정합니다. 위빙조건은 용접기능 활성화 (app_weld_enable_analog()) / app_weld_enable_digital()부터 비활성화(app_weld_disable_analog()/app_weld_disable_digital())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 위빙조건은 용접경로의 진행방향을 위빙x축으로, 위빙x축에서 TCP-z방향을 벡터곱(cross-product)한 방향을 위빙y축으로 하는 위빙좌표계에서 정의됩니다. 좌표계 및 위빙 설정 인자는 아래 그림을 참조하세요. 하나의 용접구간 내에서는 하나의 위빙조건만 허용되며 용접 중 app_weld_adj_welding_cond_analog()/app_weld_set_weld_cond_digital() 명령어를 통해 옵셋 또는 위빙폭을 조정하거나, 티칭펜던트의 용접조건 조정 popup에서 (전압/전류/속도 및) 옵셋을 조정할 수 있습니다. 다만 티칭펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET상태 (즉, app_weld_set_weld_cond_analog()/app_weld_set_weld_cond_digital()으로 지정한 용접조건 설정) 인 경우에만 가능합니다.

인수

인수명	자료형	기본값	설명
fOffsetY	float	0.0	위빙 좌표계의 Y 방향 옵셋 (mm)
fOffsetZ	float	0.0	위빙 좌표계의 Z 방향 옵셋 (mm)
fGradient	float	0.0	위빙 좌표계의 X 축 기준 위빙 평면 회전각 (deg)
fWeavingWidth	float	3.0	위빙 폭1 (mm)
fWeavingCycle	float	0.6	위빙 주기1 (sec)

리턴

값	설명
0	오류
1	성공

예제

```

float fOffsetY = 0.0;
float fOffsetZ = 0.0;
float fGradient = 0.0;
float fWeavingWidth= 3.0;
float fWeavingCycle= 0.6;

Drfl.app_weld_weave_cond_sinusoidal(fOffsetY, fOffsetZ, fGradient, fWeavingWidth,
fWeavingCycle);

```

6.1.8 app_weld_enable_analog

기능

아날로그 용접기능을 활성화합니다. 아날로그 입출력 및 디지털 신호출력 방식으로 연결 가능한 용접기의 연결 및 환경정보를 입력 인자로 입력합니다.

대상 용접기는 아날로그 인터페이스 방식을 지원하여 연결된 제어기의 아날로그 출력 채널로부터 목표전류 및 목표 전압지령을 입력 받을 수 있어야 합니다. 물리적으로 연결된 아날로그 채널의 채널번호(1 또는 2)와 출력모드(전류/전압)를 ch_v_out, ch_f_out에 설정하세요. 제어기의 아날로그 입/출력 범위는 전압모드 설정 시 0~10V, 전류모드 설정 시 4~20mA입니다. 각 채널별 설정모드 및 출력범위가 용접기의 입력사양 및 범위와 호환되도록 하세요. (예를 들어 용접기의 목표값 입력 범위가 0~10V라고 하면, 제어기의 출력채널을 전압모드(0~10V 출력범위)로 설정하는 것이 적절합니다. 또 다른 예로 용접기의 입력채널 사양이 2~15V라고 하면, 대응되는 제어기의 아날로그 채널은 전류모드 (4~20mA 출력범위)로 설정한 후 출력라인에 75옴의 저항을 연결하여 3~15V 범위의 전압을 출력할 수 있도록 연결합니다. 이 경우 제어기로 설정할 수 없는 2V~3V사이의 범위는 지령을 줄 수 없게됩니다.) 가능한 한 용접기에서 요구되는 입력범위를 많이 포함할 수 있도록 설정하는 것이 좋습니다.

제어기의 아날로그 출력 최대 및 최소 범위와 용접기의 출력 최대 및 최소 범위를 spec_v_out, spec_f_out에 설정 합니다.

- spec_v_out/spec_f_out의 첫째 항목 = WO_min
- spec_v_out/spec_f_out의 둘째 항목 = CO_min

- spec_v_out/spec_f_out의 셋째 항목 = WO_max
- spec_v_out/spec_f_out의 네째 항목 = CO_max

여기서, WO_min, WO_max는 용접기의 최소, 최대 출력사양이며, CO_min, CO_max는 각각 WO_min, WO_max에 대응되는 제어기의 아날로그 출력 값입니다.

- i** 용접기에서 출력되는 용접전류는 와이어 피딩속도는 물론 모재의 재질, 용접와이어의 재질/종류/토출길이, 용접전압등에 의해 변동되며 이는 용접기 또는 별도로 장착한 전류센서를 연결하여 확인하여야 합니다. 용접중인 전압/전류측정값을 확인하기 위해 아날로그 출력방식의 용접기 또는 별도의 센서를 연결하여야 합니다. 이에 대응하는 제어기의 아날로그 입력 채널번호 및 입력모드를 ch_v_in, ch_c_in에 설정합니다. 제어기의 아날로그 입력 최대 및 최소 범위와 센서의 측정 최대 및 최소 범위를 spec_v_in,, spec_c_in에 설정 합니다.

- spec_v_in/spec_c_in의 첫째 항목 = SO_min
- spec_v_in/spec_c_in의 둘째 항목 = CI_min
- spec_v_in/spec_c_in의 셋째 항목 = SO_max
- spec_v_in/spec_c_in의 네째 항목 = CI_max

여기서, SO_min, SO_max는 각각 센서의 최소, 최대 측정 값이며, CI_min, CI_max는 각각 SO_min, SO_max에 대응되는 제어기의 입력값입니다.

디지털 접점방식으로 용접기와 연결되는 ARC-ON/OFF(용접출력신호-시작/종료), GAS-ON/OFF(가스출력신호-시작/종료), INCHING-Forward-ON/OFF(정방향와이어송급신호-시작/종료), INCHING-Backward-ON/OFF(역방향와이어송급신호-시작/종료), BlowOut-ON/OFF(토치청소가스출력신호-시작/종료) 채널번호를 설정하십시오. ARC-ON/OFF 신호 외의 신호출력은 용접기의 해당기능 지원여부에 따라서 선택적으로 입력하십시오.

인수

인수명	자료형	기본값	설명
pConfiganalogweldinginterface	CONFIG_ANALOG_WELDING_INTERFACE	=	아날로그 용접 인터페이스 구조체

리턴

값	설명
0	오류

값	설명
1	성공

예제

```
// CONFIG_ANALOG_WELDING_INTERFACE 구조체 초기화
CONFIG_ANALOG_WELDING_INTERFACE config;
config._bMode = 1; // Start mode

// Target Voltage 설정
config._tTargetVoltage._iChannel = 1;
config._tTargetVoltage._iChannelType = 1; // Voltage
config._tTargetVoltage._iRealMinOut = 0.0;
config._tTargetVoltage._iMinOut = 0.0;
config._tTargetVoltage._iRealMaxOut = 200.0;
config._tTargetVoltage._iMaxOut = 200.0;

// Feeding Speed 설정
config._tFeedingSpeed._iChannel = 1;
config._tFeedingSpeed._iChannelType = 0; // Current
config._tFeedingSpeed._iRealMinOut = 0.0;
config._tFeedingSpeed._iMinOut = 0.0;
config._tFeedingSpeed._iRealMaxOut = 150.0;
config._tFeedingSpeed._iMaxOut = 150.0;

// Welding Voltage 설정
config._tWeldingVoltage._iChannel = 1;
config._tWeldingVoltage._iChannelType = 1; // Voltage
config._tWeldingVoltage._iRealMinOut = 0.0;
config._tWeldingVoltage._iMinOut = 0.0;
config._tWeldingVoltage._iRealMaxOut = 200.0;
config._tWeldingVoltage._iMaxOut = 200.0;

// Welding Current 설정
config._tWeldingCurrent._iChannel = 1;
config._tWeldingCurrent._iChannelType = 0; // Current
config._tWeldingCurrent._iRealMinOut = 0.0;
config._tWeldingCurrent._iMinOut = 0.0;
config._tWeldingCurrent._iRealMaxOut = 150.0;
config._tWeldingCurrent._iMaxOut = 150.0;

// 기타 설정
config._iArcOnDO = 1;
config._iGasOnDO = 1;
config._iInchPDO = 1;
config._iInchNDO = 1;
config._iBlowOutValue = 1;

// app_weld_enable_analog 함수 호출
Drfl.app_weld_enable_analog(config);
```

6.1.9 app_weld_set_weld_cond_analog

기능

아날로그 용접조건을 설정합니다. 용접조건은 용접기능 활성화 (app_weld_enable_analog()) 부터 비활성화 (app_weld_disable_analog())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 용접조건 중 용접공정변수(weld_proc_param)는 용접의 시작/종료 시의 가스/조건유지시간 등의 상세한 조건을 나타냅니다. 아래의 그림을 참조하여 입력하세요. 하나의 용접구간 내에서는 하나의 용접조건만 허용되며 용접 중 app_weld_adj_welding_cond_analog() 명령어를 통해 용접조건을 조정하거나, 티칭펜던트의 용접조건 조정 popup에서 전압/피딩속도/속도 (및 위빙옵셋)를 조정할 수 있습니다. 다만 티칭펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET상태 (즉, app_weld_set_weld_cond_analog()으로 지정한 용접조건 설정) 인 경우에만 가능합니다.

- !** 용접기에서 출력되는 용접전류는 와이어 피딩속도는 물론 모재의 재질, 용접와이어의 재질/종류/토출길이, 용접전압등에 의해 변동되며 이는 용접기 또는 별도로 장착한 전류센서를 연결하여 확인하여야 합니다.

인수

인수명	자료형	기본값	설명
pConfigAnalogWeldingSetting	CONFIG_ANALOG_WELDING_SETTING	=	아날로그 용접 세팅 구조체

리턴

값	설명
0	오류
1	성공

예제

```
// CONFIG_ANALOG_WELDING_SETTING 구조체 초기화
CONFIG_ANALOG_WELDING_SETTING config;
config._iVirtualWelding = 0; // Welding mode
config._fTargetVoltage = 200.0; // 목표 전압 (V)
config._fTargetCurrent = 150.0; // 목표 전류 (A)
config._fTargetVel = 10.0; // 목표 속도 (mm/sec)
config._fMinVel = 10.0; // 최소 속도 (mm/sec)
config._fMaxVel = 100.0; // 최대 속도 (mm/sec)
config._tDetail._fRs = 0.5; // ratio start
config._tDetail._fTss = 0.3; // 보호가스방출시간
config._tDetail._fTas = 2.0; // 시작전류시간
config._tDetail._fTwc = 1.0; // 용접조건변경시간
config._tDetail._fRf = 0.7; // ratio finish
config._tDetail._fTaf = 0.4; // 종료전류시간
config._tDetail._fTsf = 0.7; // 종료보호가스방출시간
config._tDetail._fStartVoltage = 0.6; // 시작 전압 조건
config._tDetail._fEndVoltage = 1.5; // 종료 전압 조건
config._fTargetFeedingSpeed = 0.0; // 목표 공급 속도

// app_weld_set_weld_cond_analog 함수 호출
bool result = app_weld_set_weld_cond_analog(config);
```

6.1.10 app_weld_adj_welding_cond_analog

기능

아날로그 용접 중 용접 및 위빙조건을 조정합니다. 일반적으로 연속된 경로에서 구간별로 용접조건을 변경하고자 할 때 모션명령어(move(), movec(), moveb(), movesx()) 호출 직전에 사용합니다. 본 명령어로 조정인자를 입력한 경우 해당하는 용접 및 위빙조건이 조정되며 이 때에는 TP의 용접모니터링 정보창에서 용접/위빙조건을 실시간으로 조정 할 수 없습니다. 조정조건에서 본조건(app_weld_set_weld_cond_analog()) 및 app_weld_weave_cond_trapezoidal() 등으로 설정한 용접/위빙조건)으로 복귀하려면 flag_reset=1로 실행하세요. flag_reset=1 설정 시 TP에서 실시간으로 조정한 최종 조건으로 복귀되며(실시간으로 조정불가능한 위빙폭의 비(wv_width_ratio)는 1로 변경됩니다.) TP에서 용접조건을 실시간으로 조정할 수 있습니다.

인수

인수명	자료형	기본값	설명
bRealTime	unsigned char		0: present 1: realtime

인수명	자료형	기본값	설명
bResetFlag	unsigned char	-	0 : 조정값 적용 1 : 기준목표 (app_weld_set_weld_cond_analog()) 값 적용
fTargetVol	float	-	목표전압 (V)
fFeedingVel	float	-	피딩속도 (m/min)
fTargetVel	float	-	목표속도 (mm/sec) • 티치펜던트의 입력단위와 다른 것에 유의할 것(Cm/min)
fOffsetY	float	-	위빙좌표계-y방향 옵셋 (mm)
fOffsetZ	float	-	위빙좌표계-z방향 옵셋 (mm)
fWidthRate	float	-	변경위빙폭/설정위빙폭 의 비 (0~2)

리턴

값	설명
0	오류
1	성공

예제

```
// app_weld_set_weld_cond_analog 함수 호출
unsigned char bRealTime = 0;
unsigned char bResetFlag = 0;
float fTargetVol = 0;
float fFeedingVel = 0;
float fTargetVel = 0;
float fOffsetY = 0;
float fOffsetZ = 0;
float fWidthRate = 0;
bool result = Drfl.app_weld_adj_welding_cond_analog(bRealTime, bResetFlag,
fTargetVol, fFeedingVel, fTargetVel, fOffsetY, fOffsetZ, fWidthRate);
```

6.1.11 app_weld_set_interface_eip_m2r_process

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 용접기에서 로봇제어기로 보내는 통신 데이터 중, 용접 수행을 위한 제어기와 용접기간의 연동 신호의 인터페이스를 설정할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

알아두기

1. EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.
app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()
2. 로봇 모션 시작은 용접기로 부터의 current_flow 신호와 연동이 되나 main_current 항목 설정 시 해당 신호와 연동이 됩니다.
3. 로봇 모션 종료는 용접기로 부터의 current_flow 신호와 연동이 되나 process_active 항목 설정 시 해당 신호와 연동이 됩니다.

인수

인수명	자료형	기본값	설명
_tCurrentFlow	아래참고	아래참고	용접전류 발생중 (용접기별사양)
_tProcessActive			용접공정 활성화 (용접기별사양)
_tMainCurrent			본용접전류 발생중 (용접기별사양)
_tMachineReady			용접 대기중 (용접기별사양)
_tCommReady			통신 대기중 (용접기별사양)

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255

인수명	자료형	기본값	설명
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```

CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2 process2Data;
process2Data._tCurrentFlow = {0,};
process2Data._tProcessActive = {0,};
process2Data._tMainCurrent = {0,};
process2Data._tMachineReady = {0,};
process2Data._tCommReady = {0,};
// app_weld_set_interface_eip_m2r_process2 함수 호출
bool result = Drfl.app_weld_set_interface_eip_m2r_process2(process2Data);

```

6.1.12 app_weld_set_interface_eip_r2m_mode

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 용접모드와 관련한 인터페이스를 설정합니다. 추가적으로 필요한 모드 선택 기능은 옵션 항목(wm_opt1)을 통해 추가할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
 app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
 app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
 app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterfacemode	CONFIG_DIGITAL_WELDING_INTERFACE_MODE	-	구조체 정의 참조

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값

인수명	자료형	기본값	설명
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
// EIP R2M 모드 데이터 설정 구조체 생성 및 값 설정
CONFIG_DIGITAL_WELDING_INTERFACE_MODE modeData;
modeData._tWeldingMode = {0,};
modeData._tT2TSpecial = {0,};
modeData._tPulseMode = {0,};
modeData._tWMod1 = {0,};

// app_weld_set_interface_eip_r2m_mode 함수 호출
bool result = Drfl.app_weld_set_interface_eip_r2m_mode(modeData);
```

6.1.13 app_weld_set_interface_eip_r2m_process

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 용접 수행을 위해 로봇제어기와 용접기 간 연동 신호에 대한 인터페이스를 설정합니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

i 알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다. `app_weld_set_interface_eip_r2m_process()`,
`app_weld_set_interface_eip_r2m_mode()`, `app_weld_set_interface_eip_r2m_test()`,
`app_weld_set_interface_eip_r2m_condition()`, `app_weld_set_interface_eip_r2m_option()`,
`app_weld_set_interface_eip_m2r_process()`, `app_weld_set_interface_eip_m2r_monitoring()`,
`app_weld_set_interface_eip_m2r_other()`

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterfaceprocess	CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS	-	구조체 정의 및 아래 참조
인수명	자료형	기본값	설명
welding_start	아래참조	아래참조	용접시작 명령 (용접기별 사양)
robot_ready			로봇 상태 (용접기별 사양)
error_reset			용접기 발생 에러 초기화 (용접기별 사양)

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1

인수명	자료형	기본값	설명
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS processSetting;
processSetting._tWeldingStart = {0,};
processSetting._tRobotReady = {0,};
processSetting._tErrorReset = {0,};
// app_weld_set_interface_eip_r2m_process 함수 호출
Drfl.app_weld_set_interface_eip_r2m_process(processSetting);
```

6.1.14 app_weld_set_interface_eip_r2m_test

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 용접모드와 관련한 인터페이스를 설정합니다. 추가적으로 필요한 모드 선택 기능은 옵션 항목(wm_opt1)을 통해 추가할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다. EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 테스트 신호 설정과 관련한 인터페이스를 설정합니다. 추가적인 테스트 신호 관련 설정은 옵션 항목(ts_opt1, ts_opt2)을 통해 추가할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다

i 알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

```
app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()
```

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterface test	CONFIG_DIGITAL_WELDING_INTERFACE_TEST	없음	디지털 용접 인터페이스 테스트 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```

CONFIG_DIGITAL_WELDING_INTERFACE_TEST testData;
testData._tGasTest = {0,};
testData._tInchingP = {0,};
testData._tInchingM = {0,};
testData._tBlowOutTorch = {0,};
testData._tSimulation = {0,};
testData._tTSopt1 = {0,};
testData._tTSopt2 = {0,};

// app_weld_set_interface_eip_r2m_test 함수 호출
bool result = Drfl.app_weld_set_interface_eip_r2m_test(testData);

```

6.1.15 app_weld_set_interface_eip_r2m_condition

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 용접모드와 관련한 인터페이스를 설정합니다. 추가적으로 필요한 모드 선택 기능은 옵션 항목(wm_opt1)을 통해 추가할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다. EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 테스트 신호 설정과 관련한 인터페이스를 설정합니다. 추가적인 테스트 신호 관련 설정은 옵션 항목(ts_opt1, ts_opt2)을 통해 추가할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

i 알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
 app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
 app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
 app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()

인수

수명	자료형	기본값	설명
pConfigdigitalweldinginterfacecondition	CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION	없음	디지털 용접 인터페이스 조건 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION conditionData;
conditionData._tJobNumber = {0,};
conditionData._tSynegicID = {0,};
conditionData._tWireFeedSpeed = {0,};
conditionData._tArcLengthCorrection = {0,};
conditionData._tDynamicCorrection = {0,};

// app_weld_set_interface_eip_r2m_condition 함수 호출
bool result = Drfl.app_weld_set_interface_eip_r2m_condition(conditionData);

```

6.1.16 app_weld_set_interface_eip_r2m_option

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 로봇제어기에서 용접기로 보내는 통신 데이터 중, 기본으로 제공하는 설정 항목들 (app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(), app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition()) 이외에 필요한 기능들을 해당 명령어를 통해 추가적으로 설정할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

```

app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()

```

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterfacecondition	CONFIG_DIGITAL_WELDING_INTERFACE_CONDITION	없음	디지털 용접 인터페이스 조건 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nCommDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
// EIP R2M 옵션 데이터 설정 구조체 생성 및 값 설정
CONFIG_DIGITAL_WELDING_INTERFACE_OPTION optionData;
optionData._tOption1 = {0,};
optionData._tOption2 = {0,};
optionData._tOption3 = {0,};
optionData._tOption4 = {0,};
optionData._tOption5 = {0,};
optionData._tOption6 = {0,};
optionData._tOption7 = {0,};
optionData._tOption8 = {0,};
optionData._tOption9 = {0,};
optionData._tOption10 = {0,};
optionData._tOption11 = {0,};
optionData._tOption12 = {0,};
optionData._tOption13 = {0,};
optionData._tOption14 = {0,};
optionData._tOption15 = {0,};

// app_weld_set_interface_eip_r2m_option 함수 호출
bool result = Drfl.app_weld_set_interface_eip_r2m_option(optionData);
```

6.1.17 app_weld_set_interface_eip_m2r_process2

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 용접기에서 로봇제어기로 보내는 통신 데이터 중, 용접 수행을 위한 제어기와 용접기간의 연동 신호의 인터페이스를 설정할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

알아두기

1. EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.
`app_weld_set_interface_eip_r2m_process()`, `app_weld_set_interface_eip_r2m_mode()`,
`app_weld_set_interface_eip_r2m_test()`, `app_weld_set_interface_eip_r2m_condition()`,
`app_weld_set_interface_eip_r2m_option()`, `app_weld_set_interface_eip_m2r_process()`,
`app_weld_set_interface_eip_m2r_monitoring()`, `app_weld_set_interface_eip_m2r_other()`
2. 로봇 모션 시작은 용접기로 부터의 `current_flow` 신호와 연동이 되나 `main_current` 항목 설정 시 해당 신호와 연동이 됩니다.
3. 로봇 모션 종료는 용접기로 부터의 `current_flow` 신호와 연동이 되나 `process_active` 항목 설정 시 해당 신호와 연동이 됩니다.

인수

인수명	자료형	기본값	설명
<code>pConfigdigitalweldinginterfaceprocess2</code>	<code>CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2</code>	없음	디지털 용접 인터페이스 프로세스2 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
<code>_bEnable</code>	<code>unsigned char</code>	없음	미사용: 0 사용: 1
<code>_nDataType</code>	<code>unsigned char</code>	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
<code>_nPositionalNumber</code>	<code>unsigned char</code>	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
<code>_fMinData</code>	<code>float</code>	없음	데이터 최소값
<code>_fMaxData</code>	<code>float</code>	없음	데이터 최대값
<code>_nByteOffset</code>	<code>unsigned char</code>	없음	통신 데이터 위치(byte): 1~255
<code>_nBitOffset</code>	<code>unsigned char</code>	없음	통신 데이터 위치(bit): 1~255

인수명	자료형	기본값	설명
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
// EIP M2R 프로세스2 데이터 설정 구조체 생성 및 값 설정
CONFIG_DIGITAL_WELDING_INTERFACE_PROCESS2 process2Data;
process2Data._tCurrentFlow = {0,};
process2Data._tProcessActive = {0,};
process2Data._tMainCurrent = {0,};
process2Data._tMachineReady = {0,};
process2Data._tCommReady = {0,};
// app_weld_set_interface_eip_m2r_process2 함수 호출
bool result = Drfl.app_weld_set_interface_eip_m2r_process2(process2Data);
```

6.1.18 app_weld_set_interface_eip_m2r_monitoring

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 용접기에서 로봇제어기로 보내는 통신 데이터 중, 용접기 상태 모니터링과 관련한 인터페이스를 설정합니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다.

i 알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
 app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
 app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
 app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterface monitoring	CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING	없음	디지털 용접 인터페이스 모니터링 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255

인수명	자료형	기본값	설명
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
// EIP M2R 모니터링 데이터 설정 구조체 생성 및 값 설정
CONFIG_DIGITAL_WELDING_INTERFACE_MONITORING monitoringData;
monitoringData._tWeldingVoltage = {0,};
monitoringData._tWeldingCurrent = {0,};
monitoringData._tWireFeedSpeed = {0,};
monitoringData._tWireStick = {0,};
monitoringData._tError = {0,};
monitoringData._tErrorNumber = {0,};

// app_weld_set_interface_eip_m2r_monitoring 함수 호출
bool result = Drfl.app_weld_set_interface_eip_m2r_monitoring(monitoringData);
```

6.1.19 app_weld_set_interface_eip_m2r_other

기능

EtherNet/IP 통신을 지원하는 용접기를 사용하기 위한 통신 인터페이스 설정을 합니다. 용접기에서 로봇제어기로 보내는 통신 데이터 중, 기본으로 제공하는 설정 항목들 (app_weld_set_interface_eip_m2r_process(), app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()) 이외에 필요한 기능의 인터페이스 설정을 해당 명령어를 통해 추가적으로 수행할 수 있습니다. 아래 설정값과 관련된 세부 사항은 해당 용접기의 통신 시그널 데이터시트를 참고하여 기입하기 바랍니다

i 알아두기

EtherNet/IP 통신방식 용접기를 이용한 정상적인 용접기능 수행을 위해서는 8종의 인터페이스 설정 명령어의 설정이 모두 완료되어야 합니다.

app_weld_set_interface_eip_r2m_process(), app_weld_set_interface_eip_r2m_mode(),
 app_weld_set_interface_eip_r2m_test(), app_weld_set_interface_eip_r2m_condition(),
 app_weld_set_interface_eip_r2m_option(), app_weld_set_interface_eip_m2r_process(),
 app_weld_set_interface_eip_m2r_monitoring(), app_weld_set_interface_eip_m2r_other()

인수

인수명	자료형	기본값	설명
pConfigdigitalweldinginterface other	CONFIG_DIGITAL_WELDING_INTERFACE_OTHER	없음	디지털 용접 인터페이스 기타 설정 구조체

인수 자료형, 기본값, 설명은 아래와 동일

인수명	자료형	기본값	설명
_bEnable	unsigned char	없음	미사용: 0 사용: 1
_nDataType	unsigned char	없음	데이터 타입(on/off: 0, 선택: 1, 값: 2)
_nPositionalNumber	unsigned char	없음	데이터 자릿수(1: 0, 0.1: 1, 0.01: 2)
_fMinData	float	없음	데이터 최소값
_fMaxData	float	없음	데이터 최대값

인수명	자료형	기본값	설명
_nByteOffset	unsigned char	없음	통신 데이터 위치(byte): 1~255
_nBitOffset	unsigned char	없음	통신 데이터 위치(bit): 1~255
_nComnDataType	unsigned char	없음	데이터 사이즈 1-bit(disable Low): 0 1-bit(disable High): 1 2-bit: 2 4-bit: 3 8-bit(byte): 4 15-bit: 5 16-bit(short): 6 32-bit(int): 7
_nMaxDigitSize	unsigned char	없음	유효한 데이터 사이즈 값(bit)

리턴

값	설명
0	오류
1	성공

예제

```
// EIP M2R 기타 데이터 설정 구조체 생성 및 값 설정
CONFIG_DIGITAL_WELDING_INTERFACE_OTHER otherData;
otherData._tOption1 = {0,};
otherData._tOption2 = {0,};
otherData._tOption3 = {0,};
otherData._tOption4 = {0,};
otherData._tOption5 = {0,};
otherData._tOption6 = {0,};
otherData._tOption7 = {0,};
otherData._tOption8 = {0,};
otherData._tOption9 = {0,};
otherData._tOption10 = {0,};

// app_weld_set_interface_eip_m2r_other 함수 호출
bool result = Drfl.app_weld_set_interface_eip_m2r_other(otherData);
```

6.1.20 app_weld_reset_interface

기능

이 함수는 용접 인터페이스를 초기 상태로 리셋합니다.

인수

없음

인수명	자료형	기본값	설명
bEnable	unsigned char	없음	(1: 리셋)

리턴

값	설명
0	오류
1	성공

예제

```
Drfl.app_weld_reset_interface(1);
```

6.1.21 app_weld_enable_digital

기능

통신 인터페이스 방식 용접기능을 활성화합니다. EtherNet/IP 인터페이스만 지원합니다.

인수

인수명	자료형	기본값	설명
bMode	unsigned char	없음	디지털 용접 모드 설정 (0: 비활성화)

리턴

값	설명
0	오류
1	성공

예제

```
// 디지털 용접 모드 설정
unsigned char bMode = 0; // 0: 디지털 용접 모드 비활성화

// app_weld_disable_digital 함수 호출
bool result = Drfl.app_weld_disable_digital(bMode);
```

6.1.22 app_weld_set_weld_cond_digital

기능

통신방식 용접기의 용접조건을 설정합니다. 용접조건은 용접기능 활성화 (app_weld_enable_digital()) 부터 비활성화(app_weld_disable_digital())로 정의한 용접구간 내에서만 유효하며 이외에 실행 시 에러가 발생합니다. 용접조

건으로 설정할 수 있는 항목은 다음 명령어(app_weld_set_interface_eip_r2m_mode(), app_weld_set_interface_eip_r2m_condition(), app_weld_set_interface_eip_r2m_option())에 해당하는 용접기와 통신 인터페이스 설정을 완료한 항목에 대해서만 가능합니다.

하나의 용접구간 내에서는 하나의 용접조건만 허용되며 용접 중 app_weld_adj_welding_cond_digital() 명령어를 통해 용접조건을 조정하거나, 티치펜던트의 용접조건 조정 popup에서 전압조정/동적인자조정/피딩속도/속도(및 위빙옵셋)를 조정할 수 있습니다. 다만 티치펜던트에서의 용접조건 조정은 명령어를 통한 용접조건 조정상태가 RESET 상태(즉, app_weld_set_weld_cond_digital()으로 지정한 용접조건 설정)인 경우에만 가능합니다.

알아두기

1. 전압조정(voltage correction) : 아크 길이를 조정 합니다.
2. 동적인자조정(dynamic correction) : 아크 특성을 조정 합니다

인수

인수명	자료형	기본값	설명
pConfigdigitalweldingcondition	CONFIG_DIGITAL_WELDING_CONDITION	없음	디지털 용접 조건 설정 구조체

리턴

값	설명
0	오류
1	성공

예제

```

CONFIG_DIGITAL_WELDING_ADJUST adjustData;
adjustData._bRealTime = 1; // 실시간 조정 여부 (1: 실시간)
adjustData._bResetFlag = 0; // 리셋 플래그 (0: 현재 값)
adjustData._fTargetVel = 110.0; // 목표 속도 설정
adjustData._fOffsetY = 12.0; // 위빙 Y 오프셋 설정
adjustData._fOffsetZ = 6.0; // 위빙 Z 오프셋 설정
adjustData._fWidthRate = 1.2; // 위빙 폭 비율 설정
adjustData._fDynamicCor = 0.8; // 동적 보정 값 설정
adjustData._fVoltageCor = 22.0; // 전압 보정 값 설정
adjustData._nJobNumber = 1; // 작업 번호 설정
adjustData._nSynergicID = 1; // 시너직 ID 설정

// app_weld_adj_weld_cond_digital 함수 호출
bool result = Drfl.app_weld_adj_weld_cond_digital(adjustData);

```

6.1.23 app_weld_adj_welding_cond_digital

기능

통신방식 용접기를 이용한 용접 중 용접 및 위빙조건을 조정합니다. 일반적으로 연속된 경로에서 구간별로 용접조건을 변경하고자 할 때 모션명령어(movel(), movec(), moveb(), movesx()) 호출 직전에 사용합니다. 본 명령어로 조정 인자를 입력한 경우 해당하는 용접 및 위빙조건이 조정되며 이 때에는 TP의 용접모니터링 정보창에서 용접/위빙조건을 실시간으로 조정할 수 없습니다. 조정조건에서 본조건(app_weld_set_weld_cond_digital() 및 app_weld_weave_cond_trapezoidal() 등으로 설정한 용접/위빙조건)으로 복귀하려면 flag_reset=1로 실행하세요. flag_reset=1 설정 시 TP에서 실시간으로 조정한 최종 조건으로 복귀되며(실시간으로 조정불가능한 위빙폭의 비(wv_width_ratio)는 1로 변경됩니다.) TP에서 용접조건을 실시간으로 조정할 수 있습니다.

인수

인수명	자료형	기본값	설명
pConfigdigitalweldingadjust	CONFIG_DIGITAL_WELDING_ADJUST	없음	디지털 용접 조건 조정 설정 구조체

리턴

값	설명
0	오류

값	설명
1	성공

예제

```

CONFIG_DIGITAL_WELDING_ADJUST adjustData;
adjustData._bRealTime = 1; // 실시간 조정 여부 (1: 실시간)
adjustData._bResetFlag = 0; // 리셋 플래그 (0: 현재 값)
adjustData._fTargetVel = 110.0; // 목표 속도 설정
adjustData._fOffsetY = 12.0; // 위빙 Y 오프셋 설정
adjustData._fOffsetZ = 6.0; // 위빙 Z 오프셋 설정
adjustData._fWidthRate = 1.2; // 위빙 폭 비율 설정
adjustData._fDynamicCor = 0.8; // 동적 보정 값 설정
adjustData._fVoltageCor = 22.0; // 전압 보정 값 설정
adjustData._nJobNumber = 1; // 작업 번호 설정
adjustData._nSynergicID = 1; // 시너지 ID 설정

// app_weld_adj_welding_cond_digital 함수 호출 (오타 수정)
bool result = Drfl.app_weld_adj_welding_cond_digital(adjustData);

```

6.1.24 measure_welding_tcp

기능

이 함수는 지정된 측정 모드와 스틱 아웃 값을 사용하여 용접 TCP를 측정합니다. 측정을 위해 로봇은 fTargetPos 배열에 지정된 9개의 목표 위치로 이동합니다. 측정 결과는 LPMEASURE_TCP_RESPONSE 구조체에 저장됩니다. 이 구조체는 TCP 위치 (X, Y, Z, RX, RY, RZ) 및 측정 상태 정보를 포함합니다.

인수

인수명	자료형	기본값	설명
iMode	unsigned char	없음	측정 모드 (0: 4점법, 1: 6점법)
fStickout	float	없음	스틱 아웃 값 (mm)
fTargetPos	float[9][NUMBER_OF_JOINT]	없음	목표 위치 (9개의 조인트에 대한 X, Y, Z 좌표)

리턴

리턴값	자료형	설명
result	LPMEASURE_TCP_RESPONSE	TCP 측정 결과를 나타내는 구조체

예제

```
// TCP 측정 모드 설정
unsigned char iMode = 0; // 측정 모드 (0: 4점법, 1: 6점법)

// 스틱 아웃 값 설정
float fStickout = 50.0; // 스틱 아웃 값 (mm)

// 목표 위치 설정 (9개의 조인트에 대한 X, Y, Z 좌표)
float fTargetPos[9][NUMBER_OF_JOINT] = {
    {0.0, 0.0, 0.0}, // 1번 조인트 위치
    {100.0, 0.0, 0.0}, // 2번 조인트 위치
    {100.0, 100.0, 0.0}, // 3번 조인트 위치
    {0.0, 100.0, 0.0}, // 4번 조인트 위치
    // ... 나머지 조인트 위치
};

// TCP 측정 결과를 저장할 구조체 선언
LPMEASURE_TCP_RESPONSE result;

// measure_welding_tcp 함수 호출
result = Drfl.measure_welding_tcp(iMode, fStickout, fTargetPos);

if (result._iResult == 0) {
    printf("용접 TCP 측정 성공\n");
    printf("TCP X: %f, Y: %f, Z: %f, RX: %f, RY: %f, RZ: %f\n",
        result._fTCPPos[0], result._fTCPPos[1], result._fTCPPos[2],
        result._fTCPPos[3], result._fTCPPos[4], result._fTCPPos[5]);
} else {
    printf("용접 TCP 측정 실패 (오류 코드: %d)\n", result._iResult);
}
```

6.1.25 set_welding_cockpit_setting

기능

이 함수는 택 용접 모드, 택 용접 반복 횟수, 택 용접 시간, 택 용접 간격 시간 등의 정보를 설정합니다.

인수

인수명	자료형	기본값	설명
cDataType	unsigned char	없음	데이터 타입 (0: On/Off, 1: Value)
fData	float	없음	출력 데이터 값

리턴

값	설명
0	오류
1	성공

예제

```
// 데이터 타입 설정
unsigned char cDataType = 0; // 0: On/Off, 1: Value

// 출력 데이터 설정
float fData = 1.0; // 출력 데이터 값 (cDataType에 따라 의미가 다름)

// set_digital_welding_signal_output 함수 호출
bool result = Drfl.set_digital_welding_signal_output(cDataType, fData);
```

6.1.26 set_digital_welding_monitoring_mode

기능

이 함수는 디지털 용접 모니터링 모드를 활성화 또는 비활성화합니다.

인수

인수명	자료형	기본값	설명
bEnable	unsigned char	없음	모니터링 모드 활성화 여부 (1: 활성화, 0: 비활성화)

리턴

값	설명
0	오류
1	성공

예제

```
// 디지털 용접 모니터링 모드 활성화 여부 설정
unsigned char bEnable = 1; // 1: 활성화, 0: 비활성화

// set_digital_welding_monitoring_mode 함수 호출
bool result = Drfl.set_digital_welding_monitoring_mode(bEnable);
```

6.1.27 app_weld_adj_motion_offset

기능

이 함수는 로봇의 모션 경로에 Y, Z 오프셋을 적용합니다.

인수

인수명	자료형	기본값	설명
fOffsetY	float	없음	Y 오프셋 (mm)
fOffsetZ	float	없음	Z 오프셋 (mm)

리턴

값	설명
0	오류
1	성공

예제

```
float fOffsetY = 2.0; // Y 오프셋 (mm)
float fOffsetZ = 3.0; // Z 오프셋 (mm)

// app_weld_adj_motion_offset 함수 호출
bool result = Drfl.app_weld_adj_motion_offset(fOffsetY, fOffsetZ);
```

6.1.28 set_welding_cockpit_setting_time_setting

기능

이 함수는 용접 Cockpit의 시간 설정 값을 변경합니다. 시간 설정 값은 초 단위로 입력해야 합니다.

인수

인수명	자료형	기본값	설명
time	int	없음	시간 설정 값 (단위: s)

리턴

없음

예제

```
// 시간 설정 값 (단위: s)
int time = 5; // 5s

// set_welding_cockpit_setting_time_setting 함수 호출
Drfl.set_welding_cockpit_setting_time_setting(time);
```