ETEL Innovative Motion Control

EDI 40: TECHNICAL NOTES

Table of Contents

1. HISTORICAL

	First release	**Last release**	**DSA family** (DSA, DSB)	**DSC family** (DSC2P, DSCDP, DSCDL, DSCDL_QT, DSCDM, DSCDU, DSCDV, DSCQT, DSPAC2, GPMODULE, DSMAX1, DSMAX2, DSMAX3, DSGAT, DSTEB1, DSTEB3, ACCURET_12)	**AccurET family** (AccurET, UltimET)
EDI 10	1.00 (1998)	1.01d (2000) **Obsolete**	X		
EDI 20	2.00a (22.02.2001)	2.28F (20.07.2010)	X	X	
EDI 30	3.00A (17.03.2009)			X	X
EDI 40	4.00beta (12.11.2012)				X

2. DEVELOPMENT INTERFACE

Windows version of EDI 40 has been built with Visual Studio 2010. From version 4.17beta1, EDI is built with Visual Studio 2015 with Target Platform Version set to 10.0.14393.0. It is recommended to use Visual Studio 2010 or later to develop Windows applications using EDI 40.

3. EDI COMPONENTS

EDI 10: lib10c.dll, etb10c.dll, dmd10c.dll, dsa10c.dll, tra10c.dll

EDI 20: lib10c.dll, ekd10c.dll, etb10c.dll, dmd10c.dll, dsa20c.dll, tra10c.dll, etne10c.dll, etne, etnd

EDI 30: lib20c.dll, ekd20c.dll, ekd20_64c.dll (only needed for Win7 64 bits) etb20c.dll, dmd20c.dll, dsa30c.dll, tra20c.dll, esc10c.dll esd10c.dll etne20c.dll, etn20e, etn20d
Sub-components: assert10c.dll, dex10c.dll
External sub-component: FTBUSUI.dll, FTD2XX.dll, wdapi1021.dll, wdapi1021_32.dll (only needed for Win7 64 bits)

EDI 40: lib40c.dll, ekd40c.dll, ekd40_32c.dll (only needed for Application 32bits on Win7 64 bits) etb40c.dll, dmd40c.dll, dsa40c.dll, tra40c.dll, esc40c.dll (only available for 32-bits application) esd40c.dll etne40c.dll, etn40e, etn40d
Sub-components: assert40c.dll, dex40c.dll
External sub-component: FTBUSUI.dll, FTD2XX.dll, wdapi1110.dll, wdapi1110_32.dll (only needed for 32-bits application on Win7 64 bits)

4. MAIN PROGRAMMING CHANGES WHEN PASSING FROM EDI20 TO EDI30

4.1 All files including EDI header files must include new version of header files:

EDI 20	EDI 30
#include <etne10.h>	#include <etne20.h>
#include <tra10.h>	#include <tra20.h>
#include <dsa20.h>	#include <dsa30.h>
#include <etb10.h>	#include <etb20.h>
#include <dmd10.h>	#include <dmd20.h>
#include <ekd10.h>	#include <ekd20.h>
#include <lib10.h>	#include <lib20.h>
	#include <esc10.h>

4.2 The application must be linked with new version of library files:

EDI 20	EDI 30
-L etne10c.lib	-L etne20c.lib
-L tra10c.lib	-L tra20c.lib
-L dsa20c.lib	-L dsa30c.lib
-L etb10c.lib	-L etb20c.lib
-L dmd10c.lib	-L dmd20c.lib
-L ekd10c.lib	-L ekd20c.lib
-L lib10c.lib	-L lib20c.lib
	-L esc10c.lib
	-L esd10c.lib

4.3 AccurET family has new registers type:

DSC registers	AccurET registers
K: Integer 32 bits parameters	K: Integer 32 bits parameters
	KL: Integer 64 bits parameters
	KF: Float 32 bits parameters
	KD: Float 64 bits parameters
X: Integer 32 bits user registers	X: Integer 32 bits user registers
	XL: Integer 64 bits user registers
	XF: Float 32 bits user registers
	XD: Float 64 bits user registers
M: Integer 32 bits monitoring	M: Integer 32 bits monitoring
	ML: Integer 64 bits monitoring
	MF: Float 32 bits monitoring
	MD: Float 64 bits monitoring
	C: Integer 32 bits common registers
	CL: Integer 64 bits common registers
	CF: Float 32 bits common registers
	CD: Float 64 bits common registers
S: Integer 32 bits sequence-registers	

T: Integer 32 bits trace registers	T: Integer 32 bits trace registers
	TL: Integer 64 bits trace registers
	TF: Float 32 bits trace registers
	TD: Float 64 bits trace registers
A: Integer 32 bits address-registers	A: Integer 32 bits address-registers
L: Integer 32 bits lookup-table	
	LD: Float 64 bits lookup-tables
E: Integer 32 bits trigger-registers	
	EL: Integer 64 bits trigger-registers
F: Pseudo-float 32 bits user-registers	Replaced by real float 32 bits XF registers

If your application accesses increment-value of registers using generic EDI functions dsa_get_register... or dsa_set_registers... , you must use correct function to access other typs of registers:

	EDI 20	**EDI 30**
Access integer 32 bits registers	dsa_get_register_s	dsa_get_register_s or dsa_get_register_int32_s
	dsa_get_register_a	dsa_get_register_a or dsa_get_register_int32_a
	dsa_set_register_s	dsa_set_register_s or dsa_set_register_int32_s
	dsa_set_register_a	dsa_set_register_a or dsa_set_register_int32_a
	dsa_get_array_s	dsa_get_array_s or dsa_get_array_int32_s
	dsa_set_array_a	dsa_set_array_a or dsa_set_array_int32_a
	dsa_quick_register_request_s	dsa_quick_register_request_s or dsa_quick_register_int32_request_s
	dsa_quick_register_request_a	dsa_quick_register_request_a or dsa_quick_register_int32_request_a
Access integer 64 bits registers	unavailable	dsa_get_register_int64_s
		dsa_get_register_int64_a
		dsa_set_register_int64_s
		dsa_set_register_int64_a
		dsa_get_array_int64_s
		dsa_get_array_int64_a
		dsa_quick_register_int64_request_s
		dsa_quick_register_int64_request_a
Access float 32 bits registers	unavailable	dsa_get_register_float32_s
		dsa_get_register_ float32_a
		dsa_set_register_ float32_s
		dsa_set_register_ float32_a
		dsa_get_array_ float32_s
		dsa_get_array_ float32_a

		dsa_quick_register_ float32_request_s
		dsa_quick_register_ float32_request_a
Access float 64 bits registers	unavailable	dsa_get_register_float64_s
		dsa_get_register_ float64_a
		dsa_set_register_ float64_s
		dsa_set_register_ float64_a
		dsa_get_array_ float64_s
		dsa_get_array_ float64_a
		dsa_quick_register_ float64_request_s
		dsa_quick_register_ float64_request_a

If your application converts increment value of registers into ISO value (or vice-versa) using EDI function dsa_convert_to_iso, you must use correct function to convert into/from right increment type:

	EDI 20	**EDI 30**
Convert integer 32 bits registers	dsa_convert_to_iso	dsa_convert_to_iso or dsa_convert_int32_to_iso
	dsa_convert_from_iso	dsa_convert_from_iso or dsa_convert_int32_from_iso
Convert integer 64 bits registers	unavailable	dsa_convert_int64_to_iso
		dsa_convert_int64_from_iso
Convert float 32 bits registers	unavailable	dsa_convert_float32_to_iso
		dsa_convert_float32_from_iso
Convert float 64 bits registers	unavailable	dsa_convert_float64_to_iso
		dsa_convert_float64_from_iso

The commands on AccurET family can also have parameters of new types. If your application use generic function dsa_execute_command_... to send a command with parameters specified in increments, you must be careful to use the appropriate EDI function. EDI 30 does not contain generic functions for all combination of parameters. So, if you want to send a command with more than 2 parameters or a command with parameters of type integer 64 bits, float 32 bits or float 64 bits, use the generic function dsa_execute_command_x_s.

4.4 TransnET (AccurET-family communication bus) uses a new communication protocol called "ETCOM":

This protocol allows:

- To connect up to 63 axis
- To pass up to 203 integer 32 bits parameters in one record

- To pass integer 32 bits, integer 64 bits, float 32 bits and float 64 bits parameters

Each EDI function having a parameter representing an axis-mask has a corresponding function with a 64-bit axis-mask.

Each EDI function having a parameter representing an axis number limited to 31 has a corresponding function with an axis number limited to 63.

Each EDI function having a parameter representing an old ETB_REC record has a corresponding function with a parameter representing a ETB_ETCOM record.

The corresponding functions have the prefix "etcom" in their name:

	DSC family functions	**AccurET family functions**
etb	ETB_DIAG	ETB_ETCOM_DIAG
	etb_diag	etb_etcom_diag
	ETB_SDIAG	ETB_ETCOM_SDIAG
	etb_sdiag	etb_etcom_sdiag
	ETB_FDIAG	ETB_ETCOM_FDIAG
	etb_fdiag	etb_etcom_fdiag
	etb_multi_send	etb_etcom_multi_send
	etb_get_drv_present	etb_etcom_get_drv_present
	etb_get_drv_status	etb_etcom_get_drv_status
	etb_get_drv_info	etb_etcom_get_drv_info
	etb_get_ext_info	etb_etcom_get_ext_info
	etb_add_drv_handler	etb_etcom_add_drv_handler
	etb_putm	etb_etcom_putm
	etb_putr	etb_etcom_putm
	etb_getm	etb_etcom_getm
	etb_getr	etb_etcom_getr
	etb_start_download	etb_etcom_start_download
	etb_start_download_file	etb_etcom_start_download_file
	etb_start_upload_file	etb_etcom_start_upload_file
	etb_start_upload	etb_etcom_start_upload
	etb_download_firmware	etb_etcom_download_firmware
dsa	dsa_open_e	dsa_etcom_open_e
	dsa_open_ef	dsa_etcom_open_ef
	dsa_get_etb_axis	dsa_etcom_get_etb_axis
	dsa_create_auto_e	dsa_etcom_create_auto_e
tra	TRA_LINE_WRITER	TRA_ETCOM_LINE_WRITER
	TRA_ISO_CONVERTER	TRA_ETCOM_ISO_CONVERTER
	tra_download_register_stream_e	tra_etcom_download_register_stream_e
	tra_download_register_stream_e2	tra_etcom_download_register_stream_e2
	tra_upload_register_stream_e	tra_etcom_upload_register_stream_e
	tra_send_direct_stream_e	tra_etcom_send_direct_stream_e
	tra_get_axis_mask_e	tra_etcom_get_axis_mask_e

	tra_set_axis_mask_e	tra_etcom_set_axis_mask_e
	tra_translate_rqs_to_ascii_ex	tra_etcom_translate_rqs_to_ascii_ex
	tra_translate_cmd_to_ascii_ex	tra_etcom_translate_cmd_to_ascii_ex
	tra_translate_cmd_from_ascii_ex	tra_etcom_translate_cmd_from_ascii_ex
	tra_set_iso_converter	tra_etcom_set_iso_converter
	tra_get_iso_converter	tra_etcom_get_iso_converter
	tra_set_preference_axis_mask	tra_etcom_set_preference_axis_mask
	tra_get_preference_axis_mask	tra_etcom_get_preference_axis_mask

4.5 Remarks:

DSMAX1, DSMAX2, and DSMAX3 have axis number 31. UltimET has number 63.
- ⇨ If you use DSC-family functions on an AccurET-family device, a 32-bits mask with bit 31 set will access the UltimET.
- ⇨ If you use AccurET-family functions on a DSC-family device, a 64-bits mask with bit 63 set will access the DSMAX.
- ⇨ If you use AccurET-family functions on a DSC-family device, a 64-bits mask with one of the 32-62 bit set will return an error

If you use functions accessing new register's types on DSC family, EDI will return an error.
If you send a raw ETCOM record to a device of DSC family, EDI will try to convert it into the old ETB_REC record. If this is not possible (too-much parameters, new register type, etc), EDI will return an error.

4.6 EDI specific functions:

EDI, especially DSA library, provides a huge amount of specific functions. EDI30 does no more support old DSA and DSB devices. Some specific functions were designed to access DSA/DSB specific registers. If your application calls one of these functions, EDI will return a …_EOBSOLETE error.
These functions are grouped under OBSOLETE group in the HTML documentation.

These functions are:

	OBSOLETE functions
dmd	dmd dmd_get_enum_range
dsa	dsa dsa_get_cl_input_filter_s
	dsa_get_cl_input_filter_a
	dsa_set_cl_input_filter_s
	dsa_set_cl_input_filter_a
	dsa_get_ref_demand_value_s
	dsa_get_ref_demand_value_a
	dsa_get_apr_input_filter_s
	dsa_get_apr_input_filter_a
	dsa_set_apr_input_filter_s
	dsa_set_apr_input_filter_a
	dsa_get_interrupt_mask_1_s

	dsa_get_interrupt_mask_1_a
	dsa_set_interrupt_mask_1_s
	dsa_get_interrupt_mask_1_a
	dsa_get_daisy_chain_number_s
	dsa_get_daisy_chain_number_a
	dsa_get_interrupt_mask_2_s
	dsa_get_interrupt_mask_2_a
	dsa_set_interrupt_mask_2_s
	dsa_get_interrupt_mask_2_a
	dsa_get_indirect_axis_number_s
	dsa_get_indirect_axis_number_a
	dsa_set_indirect_axis_number_s
	dsa_set_indirect_axis_number_a
	dsa_get_drive_mask_value_s
	dsa_get_drive_mask_value_a
	dsa_get_indirect_register_sidx_s
	dsa_get_indirect_register_sidx_a
	dsa_set_indirect_register_sidx_s
	dsa_set_indirect_register_sidx_a
	dsa_get_irq_drive_status_1_s
	dsa_get_irq_drive_status_1_a
	dsa_get_irq_drive_status_2_s
	dsa_get_irq_drive_status_2_a
	dsa_get_ack_drive_status_1_s
	dsa_get_ack_drive_status_1_a
	dsa_get_ack_drive_status_2_s
	dsa_get_ack_drive_status_2_a
	dsa_get_irq_pending_axis_mask_s
	dsa_get_irq_pending_axis_mask_a
	dsa_get_encoder_hall_1_signal_s
	dsa_get_encoder_hall_1_signal_a
	dsa_get_encoder_hall_2_signal_s
	dsa_get_encoder_hall_2_signal_a
	dsa_get_encoder_hall_3_signal_s
	dsa_get_encoder_hall_3_signal_a
	dsa_get_init_phase_rate_s
	dsa_get_init_phase_rate_a
	dsa_set_init_phase_rate_s
	dsa_set_init_phase_rate_a
	dsa_get_acc_actual_value_s
	dsa_get_acc_actual_value_a
	dsa_get_end_velocity_s
	dsa_get_end_velocity_a
	dsa_set_end_velocity_s
	dsa_set_end_velocity_a
	dsa_get_profile_deceleration_s
	dsa_get_profile_deceleration_a

	dsa_set_profile_deceleration_s
	dsa_set_profile_deceleration_a
	dsa_get_max_profile_velocity_s
	dsa_get_max_profile_velocity_a
	dsa_set_max_profile_velocity_s
	dsa_set_max_profile_velocity_a
	dsa_get_max_acceleration_s
	dsa_get_max_acceleration_a
	dsa_set_max_acceleration_s
	dsa_set_max_acceleration_a
	dsa_get_pl_force_feedback_gain_2_s
	dsa_get_pl_force_feedback_gain_2_a
	dsa_set_pl_force_feedback_gain_2_s
	dsa_set_pl_force_feedback_gain_2_a
	dsa_get_cl_regen_mode_s
	dsa_get_cl_regen_mode_a
	dsa_set_cl_regen_mode_s
	dsa_set_cl_regen_mode_a
	dsa_get_drive_control_mask_s
	dsa_get_drive_control_mask_a
	dsa_get_cl_phase_advance_shift_s
	dsa_get_cl_phase_advance_shift_a
	dsa_set_cl_phase_advance_shift_s
	dsa_set_cl_phase_advance_shift_a
	dsa_set_init_current_rate_s
	dsa_set_init_current_rate_a
	dsa_get_init_current_rate_s
	dsa_get_init_current_rate_a
etb	etb_auto_number

Some specific functions access a different register, depending on accessed device-type. It is so advised to access registers using specific function (if any).
For each specific function, HTML documentation will specify which register is accessed.

4.7 EDI acquisition functions:

The acquisition functionality has been extended on new AccurET family. Thus, a set of new acquisition functions has been implemented.

EDI 20	**EDI 30**
dsa_create_acquisition	dsa_create_acquisition
dsa_destroy_acquisition	dsa_destroy_acquisition

dsa_is_valid_acquisition	dsa_is_valid_acquisition
dsa_acquisition_config_trace	dsa_acquisition_config_trace
dsa_acquisition_config_trigger	dsa_acquisition_config_immediate_trigger dsa_acquisition_config_begin_of_movement_trigger dsa_acquisition_config_end_of_movement_trigger dsa_acquisition_config_position_trigger dsa_acquisition_config_position_int64_trigger dsa_acquisition_config_trace_idx_trigger dsa_acquisition_config_trace_idx_int32_trigger dsa_acquisition_config_trace_idx_int64_trigger dsa_acquisition_config_trace_idx_float32_trigger dsa_acquisition_config_trace_idx_float64_trigger dsa_acquisition_config_register_trigger dsa_acquisition_config_register_int32_trigger dsa_acquisition_config_register_int64_trigger dsa_acquisition_config_register_float32_trigger dsa_acquisition_config_register_float64_trigger dsa_acquisition_config_int32_bit_field_state_trigger dsa_acquisition_config_int64_bit_field_state_trigger dsa_acquisition_config_int32_bit_field_change_trigger dsa_acquisition_config_int64_bit_field_change_trigger
dsa_acquisition_config_frequency	dsa_acquisition_config_frequency
dsa_acquisition_get_real_total_time	dsa_acquisition_get_real_total_time
dsa_acquisition_get_trace_real_nb_points	dsa_acquisition_get_trace_real_nb_points
dsa_acquisition_upload_trace	dsa_acquisition_upload_trace
dsa_acquisition_upload_inctrace	dsa_acquisition_upload_int32_trace dsa_acquisition_upload_int64_trace dsa_acquisition_upload_float32_trace dsa_acquisition_upload_float64_trace
dsa_acquisition_stop_acquire	dsa_acquisition_stop_acquire
dsa_acquisition_reserve	dsa_acquisition_reserve
dsa_acquisition_unreserve	dsa_acquisition_unreserve
dsa_acquisition_set_name	dsa_acquisition_set_name
dsa_acquisition_unreserve_all	dsa_acquisition_unreserve_all
dsa_acquisition_is_reserved	dsa_acquisition_is_reserved
	dsa_acquisition_get_time_limits

4.8 EDI sequence download/upload functions:

The sequence philosophy has considerably changed between DSC family and AccurET family.

DSC-family sequence download/upload functions:

On DSC-family products, the sequences were interpreted by the device itself.

Technically, the sequence was edited by the user using the ETEL Sequence language. After that, using the EDI TRA library, this text-sequence was translated into a set of ETB_REC record and these records were downloaded into the device and stored into S-registers.
The upload of sequences was also done by TRA library, which is able to translate ETB_REC records into text-sequence.

EDI TRA functions used	
Create sequence traductor connected to device	tra_create_sequence_traductor_e
Attaches an ISO converter (optional)	tra_set_iso_converter
Download the text-sequence into device	tra_download_sequence_stream_e
Upload the sequence from device	tra_upload_sequence_stream_e
Destroy the traductor	tra_destroy

AccurET-family sequence download/upload functions:

Only available under WINDOWS
On AccurET-family products, the sequences are first compiled into SHARC machine code and the result downloaded into the device. The device is so able to execute directly this machine-code.
Technically, the sequence is edited by the user using the new ETEL C-Sequence language. (This language is similar to C). After that using a new EDI-ESC library, this C-sequence is compiled into SHARC machine-code and downloaded into the device. The source code-itself is also downloaded into the device, allowing the user to re-upload the sequence present in the device.

EDI ESC functions used	
Create compiler	esc_create
Attaches an ISO converter (required)	esc_set_iso_converter
Compiles and download sequence from a text-file into device OR Compiles and download sequence from a buffer into device	esc_download_cseq_file OR esc_download_cseq_buffer
Upload sequence from a device into a text-file OR Upload sequence from a device into a buffer	esc_upload_cseq_file OR esc_upload_cseq_buffer
Destroy compiler	esc_destroy

5. MAIN PROGRAMMING CHANGES WHEN PASSING FROM EDI30 TO EDI40

5.1 All files including EDI header files must include new version of header files:

EDI 30	EDI 40
#include <etne20.h>	#include <etne40.h>
#include <tra20.h>	#include <tra40.h>
#include <dsa30.h>	#include <dsa40.h>
#include <etb20.h>	#include <etb40.h>
#include <dmd20.h>	#include <dmd40.h>
#include <ekd20.h>	#include <ekd40.h>
#include <lib20.h>	#include <lib40.h>
#include <esc10.h>	

5.2 The application must be linked with new version of library files:

EDI 30	EDI 40
-L etne20c.lib	-L etne40c.lib
-L tra20c.lib	-L tra40c.lib
-L dsa30c.lib	-L dsa40c.lib
-L etb20c.lib	-L etb40c.lib
-L dmd20c.lib	-L dmd40c.lib
-L ekd20c.lib	-L ekd40c.lib
-L lib20c.lib	-L lib40c.lib
-L esd10c.lib	-L esd40c.lib
-L esc10c.lib	-L esc40c.lib (available for 32-bits application only)
	-L emp40c.lib (from EDI-4.10A only)

5.3 EDI obsolete functions

5.3.1 Register and functionality remove

EDI, especially DSA library, provides a huge amount of specific functions. EDI40 does no more support old DSA, DSB and DSC devices. Some specific functions were designed to access DSA/DSB/DSC specific registers. These functions are no more available:

OBSOLETE functions due to register or functionality remove
dsa_set_trace_mode_mvt_s/a
dsa_set_trace_mode_pos_s/a
dsa_set_trace_mode_dev_s/a
dsa_set_trace_mode_iso_s /a
dsa_set_trace_mode_immediate_s/a
dsa_trace_acquisition_s/a
dsa_ipol_reset_s/a
dsa_wait_window_user_channel_s/a
dsa_wait_movement_user_channel_s/a
dsa_wait_time_user_channel_s/a

dsa_wait_position_user_channel_s/a
dsa_wait_sgn_register_greater_user_channel_s/a
dsa_wait_sgn_register_lower_user_channel_s/a
dsa_wait_bit_set_user_channel_s/a
dsa_wait_bit_clear_user_channel_s/a
dsa_get_rtm_mon
dsa_init_rtm_fct
dsa_start_rtm
dsa_stop_rtm
dsa_edit_sequence_s/a
dsa_exit_sequence_s/a
dsa_can_command_1_s/a
dsa_can_command_2_s/a
dsa_get_ebl_baudrate_s/a
dsa_get_drive_fuse_checking_s/a
dsa_get_motor_temp_checking_s/a
dsa_get_interrupt_mask_1_s/a
dsa_get_interrupt_mask_2_s/a
dsa_get_indirect_axis_number_s/a
dsa_get_indirect_register_sidx_s/a
dsa_get_daisy_chain_number_s/a
dsa_get_drive_mask_value_s/a
dsa_get_irq_drive_status_1_s/a
dsa_get_irq_drive_status_2_s/a
dsa_get_ack_drive_status_1_s/a
dsa_get_ack_drive_status_2_s/a
dsa_get_irq_pending_axis_mask_s/a
dsa_get_encoder_phase_3_factor_s/a
dsa_get_encoder_phase_3_offset_s/a
dsa_get_encoder_index_signal_s/a
dsa_get_encoder_hall_1_signal_s/a
dsa_get_encoder_hall_2_signal_s/a
dsa_get_encoder_hall_3_signal_s/a
dsa_get_apr_input_filter_s/a
dsa_get_cl_regen_mode_s/a
dsa_get_pdr_step_value_s/a
dsa_get_ctrl_shift_factor_s/a
dsa_get_ref_demand_value_s/a
dsa_get_drive_control_mask_s/a
dsa_get_cl_output_filter_s/a
dsa_get_cl_input_filter_s/a
dsa_get_cl_phase_advance_factor_s/a
dsa_get_cl_phase_advance_shift_s/a
dsa_get_pl_speed_feedfwd_gain_s/a
dsa_get_pl_force_feedback_gain_1_s/a
dsa_get_pl_force_feedback_gain_2_s/a
dsa_get_pl_output_filter_s/a

dsa_get_pl_speed_filter_s/a
dsa_get_ttl_special_filter_s/a
dsa_get_init_current_rate_s/a
dsa_get_init_phase_rate_s/a
dsa_get_end_velocity_s/a
dsa_get_profile_deceleration_s/a
dsa_get_min_position_range_limit_s/a
dsa_get_max_profile_velocity_s/a
dsa_get_max_acceleration_s/a
dsa_get_acc_actual_value_s/a
dsa_get_io_error_event_mask_s/a
dsa_get_syncro_input_mask_s/a
dsa_get_syncro_input_value_s/a
dsa_get_syncro_output_mask_s/a
dsa_get_syncro_output_value_s/a
dsa_get_x_analog_gain_s/a
dsa_get_x_analog_offset_s/a
dsa_get_can_feedback_1_s/a
dsa_get_can_feedback_2_s/a
dsa_get_mon_dest_index_s/a
dsa_get_mon_gain_s/a
dsa_get_mon_offset_s/a
dsa_get_trigger_map_offset_s/a
dsa_get_trigger_map_size_s/a
dsa_get_trigger_io_mask_s/a
dsa_get_trigger_irq_mask_s/a
dsa_get_realtime_enabled_global_s/a
dsa_get_realtime_valid_mask_s/a
dsa_get_realtime_enabled_mask_s/a
dsa_get_realtime_pending_mask_s/a
dsa_set_ebl_baudrate_s/a
dsa_set_drive_fuse_checking_s/a
dsa_set_motor_temp_checking_s/a
dsa_set_interrupt_mask_1_s/a
dsa_set_interrupt_mask_2_s/a
dsa_set_indirect_axis_number_s/a
dsa_set_indirect_register_sidx_s/a
dsa_set_encoder_phase_3_factor_s/a
dsa_set_encoder_phase_3_offset_s/a
dsa_set_apr_input_filter_s/a
dsa_set_cl_regen_mode_s/a
dsa_set_pdr_step_value_s/a
dsa_set_ctrl_shift_factor_s/a
dsa_set_cl_output_filter_s/a
dsa_set_cl_input_filter_s/a
dsa_set_cl_phase_advance_factor_s/a
dsa_set_cl_phase_advance_shift_s/a

dsa_set_pl_speed_feedfwd_gain_s/a
dsa_set_pl_force_feedback_gain_1_s/a
dsa_set_pl_force_feedback_gain_2_s/a
dsa_set_pl_output_filter_s/a
dsa_set_pl_speed_filter_s/a
dsa_set_ttl_special_filter_s/a
dsa_set_init_current_rate_s/a
dsa_set_init_phase_rate_s/a
dsa_set_end_velocity_s/a
dsa_set_profile_deceleration_s/a
dsa_set_min_position_range_limit_s/a
dsa_set_max_profile_velocity_s/a
dsa_set_max_acceleration_s/a
dsa_set_io_error_event_mask_s/a
dsa_set_syncro_input_mask_s/a
dsa_set_syncro_input_value_s/a
dsa_set_syncro_output_mask_s/a
dsa_set_syncro_output_value_s/a
dsa_set_x_analog_gain_s/a
dsa_set_x_analog_offset_s/a
dsa_set_mon_dest_index_s/a
dsa_set_mon_gain_s/a
dsa_set_mon_offset_s/a
dsa_set_trigger_map_offset_s/a
dsa_set_trigger_map_size_s/a
dsa_set_trigger_io_mask_s/a
dsa_set_trigger_irq_mask_s/a
dsa_set_realtime_enabled_global_s/a
dsa_set_realtime_valid_mask_s/a
dsa_set_realtime_pending_mask_s/a
etb_get_baudrate
etb_multi_send
etb_start_rtm
etb_stop_rtm
etb_init_rtm_fct
etb_get_rtm_mon
etb_link_error
etb_irq_watchdog
etb_get_bus_counters
etb_add_rt_handler
etb_remove_rt_handler
etb_auto_number
etb_activate_download
etb_start_download_file
etb_start_upload_file
etb_etcom_multi_send
etb_etcom_start_download_file

etb_etcom_start_upload_file

5.3.2 Axis addressing

DSC and AccurET families differ concerning axis addressing:

- DSC family allows 32 axes, where DSMAX has number 31.
- AccurET family allows 64 axes, where UltimET has number 63.

EDI functions with parameters allowing axis addressing have been doubled in EDI3 (Example: dsa_get_etb_axis was dedicated to DSC family and returned an axis number 0 and 31, while dsa_etcom_get_etb_axis was dedicated to AccurET family and returned an axis number 0 and 63). As EDI4 does no more support DSC, the DSC dedicated functions addressing axis have been removed:

OBSOLETE functions due to DSC-family axis addressing
dsa_open_e
dsa_open_ef
dsa_get_etb_axis
dsa_create_auto_e
etb_get_baudrate
etb_multi_send
etb_get_drv_present
etb_get_drv_status
etb_get_drv_info
etb_get_ext_info
etb_diag
etb_sdiag
etb_fdiag
etb_putm
etb_putr
etb_getm
etb_getr
etb_start_download
etb_start_upload
etb_download_firmware
tra_upload_register_stream_e
tra_download_register_stream_e
tra_download_register_stream_e2
tra_upload_limited_register_stream_e
tra_send_direct_stream_e
tra_get_axis_mask_e
tra_set_axis_mask_e
tra_set_preference_axis_mask
tra_get_preference_axis_mask

5.3.3 Record accessing

DSC and AccurET families differ concerning the record used on the communication protocol:
- DSC family uses ETBREC fixed size record.
- AccurET family uses ETCOM variable length record.

As EDI3.xx supports both DSC and ETEL products, some functions allowing management of records have been implemented. The functions for the old ETBREC record have been removed:

OBSOLETE functions due to DSC-family ETBREC record
tra_translate_cmd_to_ascii
tra_translate_cmd_to_ascii_ex
tra_translate_cmd_from_ascii
tra_translate_cmd_from_ascii_ex
tra_translate_rqs_to_ascii
tra_translate_rqs_to_ascii_ex
tra_get_iso_converter
tra_set_iso_converter
etb_etcom_to_recs
etb_recs_to_etcom

5.3.4 NON ANSI functions

Some non ANSI functions have been removed. These are the functions which return a whole structure content. These are especially the functions allowing the initialisation of a structure:

OBSOLETE NON ANSI functions
dsa_init_status
dsa_init_info
dsa_init_x_info
dsa_init_vector
dsa_init_vector_typ
dsa_init_rtm
etb_init_drv_info
etb_init_timeouts
etb_init_ext_info
etb_init_drv_status
etb_init_rec_param
etb_init_counters
etb_init_bus_status
etb_init_svr_info
etb_activate_status
etb_init_master_info

The call to these functions can be replaced by:

Example:
DSA_STATUS status = dsa_init_status();
replaced by
DSA_STATUS status = {sizeof(DSA_STATUS)};

5.3.5 Functions managing special ETEL devices

A special ETEL device called "gp_module" was developed. This device was a special "TEB" device. All functions accessing this device have been removed:

OBSOLETE « GP_MODULE » special device functions
dsa_create_gp_module
dsa_create_gp_module_group
dsa_is_valid_gp_module
dsa_is_valid_gp_module_group
dsa_is_valid_gp_module_base

5.3.6 Special functions accessing "DSMAX"

Some functions accessing DSMAX must be used in EDI3 to access UltimET as well. These functions has been renamed into "...*MASTER*...", providing a more generic name.
To allow portability between EDI3 and EDI4, the old functions name still exists, but will be removed in further EDI version:

OBSOLETE « DSMAX » functions	
dsa_create_dsmax	dsa_create_master
dsa_create_ dsmax _group	dsa_create_ master _group
dsa_is_valid_ dsmax	dsa_is_valid_master
dsa_is_valid_dsmax _group	dsa_is_valid_master _group
dsa_is_valid_dsmax_base	dsa_is_valid_master_base
dsa_set_dsmax	dsa_set_master
dsa_get_dsmax	dsa_get_master

OBSOLETE « DSMAX » objects	
DSA_DSMAX	DSA_MASTER
DSA_DSMAX_GROUP	DSA_MASTER_GROUP
DSA_DSMAX_BASE	DSA_MASTER_BASE

5.3.7 Function allowing old ETEL sequence translation/download/upload

The ETEL sequences on DSC-family were interpreted by the drive. These sequences were saved into S registers of the drive. The functions, allowing translation, download and upload of such sequences into S registers, have been removed:

OBSOLETE ETEL sequence translation/download/upload
tra_is_valid_sequence_traductor
tra_create_sequence_traductor_o
tra_create_sequence_traductor_o2
tra_clear_sequence_drive_map
tra_setup_sequence_drive_map
tra_etcom_setup_sequence_drive_map
tra_is_valid_sequence_traductor_e
tra_create_sequence_traductor_e
tra_download_sequence_stream_e
tra_upload_sequence_stream_e
tra_get_sequence_line_e

5.4 EDI 64bits unavailable functionality

Inside EDI, a dll allowing new ETEL sequence compilation was provided on EDI3.xx. This dll is no more available on EDI 64 bits native dll.
Compilation of ETEL Sequence must be done with a 32-bits application as ComET.