UF_CAM_ask_auto_blank (view source)

Defined in: uf_cam.h

Overview
Query the type and data of an automatic blank.

Return
Return code :

Environment
Internal and External

See Also
UF_CAM_set_auto_blank

History
Released in V19.0

```
int UF_CAM_ask_auto_blank
(
    tag_t object_tag,
    UF_CAM_blank_geom_type_t * geom_type,
    double offset [ 6 ]
)
```

tag_t	**object_tag**	Input	the operation or geometry group containing blank definition
UF_CAM_blank_geom_type_t *	**geom_type**	Output	type type of blank geometry defined

double	**offset [6]**	Output	For geom_type = UF_CAM_auto_block_type, offset is an array of positive deltas to a minimal box
			aligned with the MCS which contains the specified part geometry.
			Offset[0] = offset along +XM
			Offset[1] = offset along -XM
			Offset[2] = offset along +YM
			Offset[3] = offset along -YM
			Offset[4] = offset along +ZM
			Offset[5] = offset along -ZM
			For geom_type = UF_CAM_offset_from_part, offset is a single positive offset from the specified
			Part geometry.
			Offset[0] = global offset of part geometry
			Offset[1-5] unused

UF_CAM_ask_blank_matl_db_object (view source)

Defined in: uf_cam.h

Overview
This function provides the database object which is currently used to access the Blank Material library.

Environment
Internal and External

History

Released in V16.0

```
int UF_CAM_ask_blank_matl_db_object
(
    UF_CAM_db_object_t * db_obj
)
```

UF_CAM_db_object_t *	**db_obj**	Output	- see function description

---

UF_CAM_ask_cam_preferences (view source)

Defined in: uf_cam.h

Overview
This function provides the current settings of the CAM preferences.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_cam_preferences
(
    UF_CAM_preferences_p_t prefs
)
```

UF_CAM_preferences_p_t	**prefs**	Output	- the current CAM preferences

---

UF_CAM_ask_clear_plane_data (view source)

Defined in: uf_cam_planes.h

Overview
Query the origin and normal of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_clear_plane_data
(
    tag_t object_tag,
    double origin [ 3 ] ,
    double normal [ 3 ]
)
```

tag_t	**object_tag**	Input	the parent object of the plane
double	**origin [3]**	Output	the 3D origin of the plane
double	**normal [3]**	Output	the 3D normal of the plane

UF_CAM_ask_clear_plane_status (view source)

Defined in: uf_cam_planes.h

Overview
Query the status of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_clear_plane_status
(
    tag_t object_tag,
    UF_PARAM_clrplane_status_t * status
)
```

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_clrplane_status_t *	**status**	Output	the status of the plane

---

UF_CAM_ask_clear_plane_tag (view source)

Defined in: uf_cam_planes.h

Overview
Query the tag of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_clear_plane_tag
(
    tag_t object_tag,
    tag_t * target_tag
```

)

tag_t	**object_tag**	Input	the parent object of the plane
tag_t *	**target_tag**	Output	the tag of an UF_xform_type entity representing the clearance plane

---

UF_CAM_ask_clear_plane_usage (view source)

Defined in: uf_cam_planes.h

Overview
Query the usage of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_clear_plane_usage
(
    tag_t object_tag,
    UF_PARAM_clrplane_usage_t * usage
)
```

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_clrplane_usage_t *	**usage**	Output	clearance plane usage

---

UF_CAM_ask_config_file (view source)

Defined in: uf_cam.h

Overview
This function provides the name of the CAM configuration file used in the current CAM Session.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAM_ask_config_file
(
    const char * * cam_config_filename
)
```

const char * *	**cam_config_filename**	Output to UF_*free*	- configuration file name of the current CAM session. The returned string must be freed by calling UF_free.

UF_CAM_ask_cutter_db_object (view source)

Defined in: uf_cam.h

Overview

This function provides the database object which is currently used to access the Cutter library.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_ask_cutter_db_object
(
    UF_CAM_db_object_t * db_obj
)
```

UF_CAM_db_object_t *	**db_obj**	Output	- see function description

UF_CAM_ask_doc_template_name (view source)

Defined in: uf_cam.h

Overview

This function provides the name of the file that stores the list of available Documentation templates. This is determined by the contents of cam_config.dat.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_ask_doc_template_name
(
    const char * * doc_template_filename
)
```

const char * *	**doc_template_filename**	Output to UF_*free*	- see function comments

UF_CAM_ask_f_s_db_object (view source)

Defined in: uf_cam.h

Overview
This function provides the database object which is currently used to access the Feeds and Speeds library.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_f_s_db_object
(
    UF_CAM_db_object_t * db_obj
)
```

UF_CAM_db_object_t *	**db_obj**	Output	- see function description

---

UF_CAM_ask_leastsq_sphere (view source)

Defined in: uf_cam.h

Overview
DESCRIPTION:
This utility function computes the center point and the radius of least square sphere for a given set of points.

Environment
Internal and External

History
Released in NX8.5

```
int UF_CAM_ask_leastsq_sphere
(
    double * * point_coords,
    int point_count,
    double tolerance,
    double sphere_center [ 3 ] ,
    double * sphere_radius
)
```

double * *	**point_coords**	Input	coordinates of array of points for sphere fit. (3 point_count values.)
int	**point_count**	Input	- Number of points
double	**tolerance**	Input	- Distance tolerance
double	**sphere_center [3]**	Output	- Coordinates of sphere center
double *	**sphere_radius**	Output	- Sphere radius

UF_CAM_ask_lower_limit_plane_data (view source)

Defined in: uf_cam_planes.h

Overview
Query the origin and normal of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_lower_limit_plane_data
(
    tag_t object_tag,
    double origin [ 3 ] ,
    double normal [ 3 ]
)
```

tag_t	**object_tag**	Input	the parent object of the plane
double	**origin [3]**	Output	the 3D origin of the plane
double	**normal [3]**	Output	the 3D normal of the plane

UF_CAM_ask_lower_limit_plane_status (view source)

Defined in: uf_cam_planes.h

Overview
Query the status of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_lower_limit_plane_status
(
    tag_t object_tag,
    UF_PARAM_lwplane_status_t * status
)
```

tag_t	**object_tag**	Input	the parent object of the plane

UF_PARAM_lwplane_status_t *	**status**	Output	the status of the plane

---

UF_CAM_ask_lower_limit_plane_tag (view source)

Defined in: uf_cam_planes.h

Overview
Query the tag of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_lower_limit_plane_tag
(
    tag_t object_tag,
    tag_t * target_tag
)
```

tag_t	**object_tag**	Input	the parent object of the plane
tag_t *	**target_tag**	Output	the tag of an UF_xform_type entity representing the lower limit plane

---

UF_CAM_ask_lower_limit_plane_usage (view source)

Defined in: uf_cam_planes.h

Overview
Query the usage of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_ask_lower_limit_plane_usage
(
    tag_t object_tag,
    UF_PARAM_lwplane_usage_t * usage
)
```

tag_t	**object_tag**	Input	the parent object of the plane

UF_PARAM_lwplane_usage_t *	**usage**	Output	lower limit plane usage

---

UF_CAM_ask_mach_tool_db_object (view source)

Defined in: uf_cam.h

Overview
This function provides the database object which is currently used to access the Machine Tool library.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_mach_tool_db_object
(
    UF_CAM_db_object_t * db_obj
)
```

UF_CAM_db_object_t *	**db_obj**	Output	- see function description

---

UF_CAM_ask_opt_template_object (view source)

Defined in: uf_cam.h

Overview
This function provides the object which is used to interface with the current Object Parameter Templates (OPTs). Refer to the UF_CAM_opt_stype_cls_t definition to see the possible OPT subtype classes available.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_opt_template_object
(
    UF_CAM_opt_t * opt_object
)
```

UF_CAM_opt_t *	**opt_object**	Output	- see function description

---

UF_CAM_ask_post_template_name (view source)

Defined in: uf_cam.h

Overview
This function provides the name of the file that stores the list of available POST templates. This is determined by the contents of cam_config.dat.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_post_template_name
(
    const char * * post_template_filename
)
```

const char * *	**post_template_filename**	Output to UF_*free*	- see function comments

---

UF_CAM_ask_tool_matl_db_object (view source)

Defined in: uf_cam.h

Overview
This function provides the database object which is currently used to access the Tool Material library.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_ask_tool_matl_db_object
(
    UF_CAM_db_object_t * db_obj
)
```

UF_CAM_db_object_t *	**db_obj**	Output	- see function description

---

UF_CAM_init_session (view source)

Defined in: uf_cam.h

Overview
This function initializes the current CAM session based upon the contents of the configuration file specified by $UGII_CAM_CONFIG. If a CAM session currently exists it is first unloaded.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_init_session
(
    void
)
```

UF_CAM_is_session_initialized (view source)

Defined in: uf_cam.h

Overview
This function answers whether or not there exists a currently initialized CAM session. A currently initialized CAM session must exist in order to call any other NX CAM User Function except init_session.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_is_session_initialized
(
    logical * answer
)
```

logical *	**answer**	Output	- TRUE if there exists an initialized CAM session; FALSE otherwise.

UF_CAM_opt_add_template_part (view source)

Defined in: uf_cam.h

Overview
This function adds the specified part file as a new subtype to the existing Object Parameter Templates.

Environment
Internal and External

History
Released in nx903

```
int UF_CAM_opt_add_template_part
(
    const char * filespec
)
```

const char *	**filespec**	Input	- the name of the file representing the template part

UF_CAM_opt_add_type (view source)

Defined in: uf_cam.h

Overview

This function adds the specified part file as a new type to the existing Object Parameter Templates. All the subtypes contained in the specified part file are added as Object Parameter Templates subtypes.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_opt_add_type
(
    const char * filespec
)
```

const char *	**filespec**	Input	- the name of the file representing the new type.

UF_CAM_opt_ask_clsf_names (view source)

Defined in: uf_cam.h

Overview

This function provides a list of available CLSF names. They are derived from the CLSF template file that is specified by the configuration file that is used to initialize the CAM session. These names can be used to generate a specified CLSF by calling the apppropriate UF_SETUP function.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_opt_ask_clsf_names
(
    int * count,
    const char * * * names
)
```

int *	**count**	Output	- the number of available clsf names
const char * * *	**names**	Output to UF_*free*	- the available clsf names. The returned array must be freed by calling UF_free_string_array.

UF_CAM_opt_ask_doc_names (view source)

Defined in: uf_cam.h

Overview

This function provides a list of available SHOP DOC names. They are derived from the SHOP DOC template file that is specified by the configuration file that is used to initialize the CAM session. These names can be used to generate a specified documentation format by calling the apppropriate UF_SHOPDOC function.

NOTE: you should use UF_free_string_array to free the returned memory.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_opt_ask_doc_names
(
    int * count,
    const char * * * names
)
```

int *	**count**	Output	- the number of names
const char * * *	**names**	Output to UF_*free*	- the available doc names. The returned array must be freed by calling UF_free_string_array.

UF_CAM_opt_ask_object (view source)

Defined in: uf_cam.h

Overview

This function provides the tag of the NX object which corresponds to the specified Object Parameter Template type and subtype.
Templates.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_opt_ask_object
(
    UF_CAM_opt_stype_cls_t subtype_class,
    const char * type,
    const char * subtype,
    tag_t * param
)
```

UF_CAM_opt_stype_cls_t	**subtype_class**	Input	- the desired subtype class
const char *	**type**	Input	- the type of the object desired
const char *	**subtype**	Input	- the subtype of the object desired
tag_t *	**param**	Output	- the tag of the desired object

UF_CAM_opt_ask_post_names (view source)

Defined in: uf_cam.h

Overview

This function provides a list of available post names. They are derived from the post template file that is specified by the configuration file that is used to initialize the CAM session. These names can be used to generate a specified Post by calling the apppropriate UF_SETUP function.

Environment

Internal and External

History

Released in V16.0

```
int UF_CAM_opt_ask_post_names
(
    int * count,
    const char * * * names
)
```

int *	**count**	Output	- the number of available post names
const char * * *	**names**	Output to UF_*free*	- the available post names. The returned array must be freed by calling UF_free_string_array.

UF_CAM_opt_ask_subtypes (view source)

Defined in: uf_cam.h

Overview

This function provides the names of the available Object Parameter Template subtypes for the specified Object Parameter Type. Only those subtypes which have the specified Object Parameter Template subtype class are returned.

Environment

Internal and External

History

Released in V16.0

int UF_CAM_opt_ask_subtypes
(
const char * opt_type_name,
UF_CAM_opt_stype_cls_t subtype_class,
int * count,
const char * * * subtypes
)

const char *	**opt_type_name**	Input	- the name of the OPT type whose subtypes are desired
UF_CAM_opt_stype_cls_t	**subtype_class**	Input	- the desired subtype class
int *	**count**	Output	- the number of subtypes
const char * * *	**subtypes**	Output to UF_*free*	- the available subtypes. The returned array must be freed by calling UF_free_string_array.

UF_CAM_opt_ask_types (view source)

Defined in: uf_cam.h

Overview
This function provides the names of the available Object Parameter Template types.

Environment
Internal and External

History
Released in V16.0

int UF_CAM_opt_ask_types
(
int * count,
const char * * * type_names
)

int *	**count**	Output	- the number of type names returned
const char * * *	**type_names**	Output to UF_*free*	- the available type names. The returned array must be freed by calling UF_free_string_array.

UF_CAM_PREF_ask_data_type (view source)

Defined in: uf_cam_prefs.h

Overview
This function provides the data type of the specified CAM Preference.

Return
UF_CAM_ERROR_PREFERENCE_NOT_DEFINED - if the specified preference is not defined in the above enum

Environment
Internal and External

History
Released in V19.0

int UF_CAM_PREF_ask_data_type
(
UF_CAM_PREF_t pref,
UF_PARAM_type_t * type
)

UF_CAM_PREF_t	**pref**	Input	- the specific desired preference
UF_PARAM_type_t *	**type**	Output	- the data type of the specified preference - currently only UF_PARAM_TYPE_LOGICAL and UF_PARAM_TYPE_INT are supported

UF_CAM_PREF_ask_integer_value (view source)

Defined in: uf_cam_prefs.h

Overview
This function provides the integer value of the specified CAM Preference.

Return
UF_CAM_ERROR_DATA_NOT_CORRECT_TYPE - if specified preference does not hold integer data

Environment
Internal and External

History
Released in V19.0

int UF_CAM_PREF_ask_integer_value
(
UF_CAM_PREF_t pref,
int * value
)

UF_CAM_PREF_t	**pref**	Input	- the specific desired preference
int *	**value**	Output	- the value of the specified preference

UF_CAM_PREF_ask_logical_value (view source)

Defined in: uf_cam_prefs.h

Overview
This function provides the logical setting of the specified CAM Preference.

Return
UF_CAM_ERROR_DATA_NOT_CORRECT_TYPE - if specified preference does not hold logical data

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_PREF_ask_logical_value
(
    UF_CAM_PREF_t pref,
    logical * value
)
```

UF_CAM_PREF_t	**pref**	Input	- the specific desired preference
logical *	**value**	Output	- the value of the specified preference

---

UF_CAM_PREF_set_integer_value (view source)

Defined in: uf_cam_prefs.h

Overview
This function sets the integer value of the specified CAM Preference.

Return
UF_CAM_ERROR_DATA_NOT_CORRECT_TYPE - if specified preference does not hold integer data

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_PREF_set_integer_value
(
    UF_CAM_PREF_t pref,
    int value
)
```

UF_CAM_PREF_t	**pref**	Input	- the specific desired preference
int	**value**	Input	- the value of the specified preference

UF_CAM_PREF_set_logical_value (view source)

Defined in: uf_cam_prefs.h

Overview

This function sets the logical setting of the specified CAM Preference.

Return

UF_CAM_ERROR_DATA_NOT_CORRECT_TYPE - if specified preference does not hold logical data

Environment

Internal and External

History

Released in V19.0

```
int UF_CAM_PREF_set_logical_value
(
    UF_CAM_PREF_t pref,
    logical value
)
```

UF_CAM_PREF_t	**pref**	Input	- the specific desired preference
logical	**value**	Input	- the value of the specified preference

---

UF_CAM_PREPRO_init_module (view source)

Defined in: uf_cam_prepro.h

Overview

Initializes the required environment for this module.

Environment

Internal and External

```
int UF_CAM_PREPRO_init_module
(
    void
)
```

UF_CAM_PREPRO_mark_model_as_cam (view source)

Defined in: uf_cam_prepro.h

Overview

This function will mark the facet model as a model that can be used for CAM purposes. This will inform the CAM preprocessors that the prepro information is available and that the user intends to use it to represent the corresponding geometry.

Environment

Internal and External

See Also

See the sample program sample program

int UF_CAM_PREPRO_mark_model_as_cam
(
    tag_t model
)

tag_t	**model**	Input	The facet model created or updated using the interface described in uf_facet.h

---

UF_CAM_reinit_opt (view source)

Defined in: uf_cam.h

Overview
This function reinitializes the Object Parameter Templates based upon the contents of the specified template file.

Environment
Internal and External

History
Released in V16.0

int UF_CAM_reinit_opt
(
    const char * template_filename
)

const char *	**template_filename**	Input	- see function description

---

UF_CAM_reinit_session (view source)

Defined in: uf_cam.h

Overview
This function initializes the current CAM session based upon the contents of the specified configuration file. If a CAM session currently exists it is first unloaded.

Environment
Internal and External

History
Released in V16.0

int UF_CAM_reinit_session
(
    const char * config_file
)

const char *	**config_file**	Input	- the name of the configuration file to use to initialize the CAM session.

UF_CAM_set_auto_blank (view source)

Defined in: uf_cam.h

Overview
Define the type and data of an automatic blank.

Return
Return code :

Environment
Internal and External

See Also
UF_CAM_ask_auto_blank

History
Released in V19.0

```
int UF_CAM_set_auto_blank
(
    tag_t object_tag,
    UF_CAM_blank_geom_type_t geom_type,
    double offset [ 6 ]
)
```

tag_t	**object_tag**	Input	the operation or geometry group containing blank definition
UF_CAM_blank_geom_type_t	**geom_type**	Input	type type of blank geometry defined

double	**offset [6]**	Input	For geom_type = UF_CAM_auto_block_type, offset is an array of positive deltas to a minimal box
			aligned with the MCS which contains the specified part geometry.
			Offset[0] = offset along +XM
			Offset[1] = offset along -XM
			Offset[2] = offset along +YM
			Offset[3] = offset along -YM
			Offset[4] = offset along +ZM
			Offset[5] = offset along -ZM
			For geom_type = UF_CAM_offset_from_part, offset is a single positive offset from the specified
			Part geometry.
			Offset[0] = global offset of part geometry
			Offset[1-5] unused

UF_CAM_set_cam_preferences (view source)

Defined in: uf_cam.h

Overview
This function sets the current settings of the CAM preferences.

Environment
Internal and External

History
Released in V16.0

```
int UF_CAM_set_cam_preferences
(
    UF_CAM_preferences_p_t prefs
)
```

UF_CAM_preferences_p_t	**prefs**	Input	- the values to use to set the current CAM preferences settings.

UF_CAM_set_clear_plane_data (view source)

Defined in: uf_cam_planes.h

Overview
Define/edit the origin and normal of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_clear_plane_data
(
    tag_t object_tag,
    double origin [ 3 ] ,
    double normal [ 3 ]
)
```

tag_t	**object_tag**	Input	the parent object of the plane
double	**origin [3]**	Input	the 3D origin of the plane
double	**normal [3]**	Input	the 3D normal of the plane

UF_CAM_set_clear_plane_status (view source)

Defined in: uf_cam_planes.h

Overview
Set the status of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

int UF_CAM_set_clear_plane_status
(
tag_t object_tag,
UF_PARAM_clrplane_status_t status
)

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_clrplane_status_t	**status**	Input	the status of the plane

---

UF_CAM_set_clear_plane_tag (view source)

Defined in: uf_cam_planes.h

Overview
Set the tag of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

int UF_CAM_set_clear_plane_tag
(
tag_t object_tag,
tag_t target_tag
)

tag_t	**object_tag**	Input	the parent object of the plane
tag_t	**target_tag**	Input	the tag of an UF_xform_type entity representing the clearance plane

---

UF_CAM_set_clear_plane_usage (view source)

Defined in: uf_cam_planes.h

Overview
Set the usage of a clearance plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_clear_plane_usage
(
    tag_t object_tag,
    UF_PARAM_clrplane_usage_t usage
)
```

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_clrplane_usage_t	**usage**	Input	clearance plane usage

---

UF_CAM_set_lower_limit_plane_data (view source)

Defined in: uf_cam_planes.h

Overview
Define/edit the origin and normal of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_lower_limit_plane_data
(
    tag_t object_tag,
    double origin [ 3 ] ,
    double normal [ 3 ]
)
```

tag_t	**object_tag**	Input	the parent object of the plane
double	**origin [3]**	Input	the 3D origin of the plane
double	**normal [3]**	Input	the 3D normal of the plane

UF_CAM_set_lower_limit_plane_status (view source)

Defined in: uf_cam_planes.h

Overview
Set the status of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_lower_limit_plane_status
(
    tag_t object_tag,
    UF_PARAM_lwplane_status_t status
)
```

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_lwplane_status_t	**status**	Input	the status of the plane

---

UF_CAM_set_lower_limit_plane_tag (view source)

Defined in: uf_cam_planes.h

Overview
Set the tag of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_lower_limit_plane_tag
(
    tag_t object_tag,
    tag_t target_tag
)
```

tag_t	**object_tag**	Input	the parent object of the plane
tag_t	**target_tag**	Input	the tag of an UF_xform_type entity representing the lower limit plane

---

UF_CAM_set_lower_limit_plane_usage (view source)

Defined in: uf_cam_planes.h

Overview
Set the usage of a lower limit plane

Return
Return code :

Environment
Internal and External

History
Released in V19.0

```
int UF_CAM_set_lower_limit_plane_usage
(
    tag_t object_tag,
    UF_PARAM_lwplane_usage_t usage
)
```

tag_t	**object_tag**	Input	the parent object of the plane
UF_PARAM_lwplane_usage_t	**usage**	Input	lower limit plane usage

UF_CAM_set_material (view source)

Defined in: uf_cam.h

Overview
This function sets the material type for the input object.

Environment
Internal and External

History
Released in NX6

```
int UF_CAM_set_material
(
    tag_t object_tag,
    char * libref
)
```

tag_t	**object_tag**	Input	Tag to input object (cutter or geom group)
char *	**libref**	Input	library reference to desired material

UF_CAM_update_list_object_customization (view source)

Defined in: uf_cam.h

Overview
This function provids the functionality to update the customization information of a list of objects to be the same as the template type and subtype from which it was created.

Environment
Internal or External

History

Relaeased in NX3

int UF_CAM_update_list_object_customization
(
tag_t * object_tags
)

tag_t *	**object_tags**	Input	The tags of the objects for which the customization should be updated based on template type and subtype

---

UF_CAM_update_single_object_customization (view source)

Defined in: uf_cam.h

Overview
This function provids the functionality to update the customization information of an object to be the same as the template type and subtype from which it was created.

Environment
Internal or External

History
Relaeased in NX3

int UF_CAM_update_single_object_customization
(
tag_t object_tag
)

tag_t	**object_tag**	Input	The tag of the object for which the customization should be updated based on template type and subtype

---

UF_CAM_wizard_ask_current_object (view source)

Defined in: uf_ui_param.h

Overview
This function queries the current object the manufacturing wizard is working with. If there is no current object in the wizard process the function will return NULL. The current object is returned through the output parameter 'param_tag'

Environment
Internal

History
Released in NX4

int UF_CAM_wizard_ask_current_object
(
    tag_t * param_tag
)

tag_t *	**param_tag**	Output	see above

UF_CAM_wizard_set_current_object (view source)

Defined in: uf_ui_param.h

Overview
This function sets the current object the manufacturing wizard
should work with. The object must exists in the current work part and
must be a valid manufacturing object.

Environment
Internal

History
Released in NX4

int UF_CAM_wizard_set_current_object
(
    tag_t param_tag
)

tag_t	**param_tag**	Input	see above

UF_CAMBND_append_bnd_from_curve (view source)

Defined in: uf_cambnd.h

Overview
Appends a single boundary created from edges or curves to the object.

NOTE:
The pointer to the UF_CAMBND_app_data_t structure in the boundary_data
structure must either be NULL or a structure allocated and initialized
by the user.

Environment
Internal and External

History
Released in V18.0

int UF_CAMBND_append_bnd_from_curve
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    int count,
    tag_t * curves,

```
    UF_CAMBND_boundary_data_p_t boundary_data,
    UF_CAMBND_app_data_p_t * app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
int	**count**	Input	the count of edges/curves
tag_t *	**curves**	Input	count the edge/curve tags from which a boundary will be created
UF_CAMBND_boundary_data_p_t	**boundary_data**	Input	the boundary data
UF_CAMBND_app_data_p_t *	**app_data**	Input	count the application data for each member

---

UF_CAMBND_append_bnd_from_face (view source)

Defined in: uf_cambnd.h

Overview

Appends one or more boundaries that are created from the face to the object.

NOTES:
The face must be planar.

The pointer to the UF_CAMBND_app_data_t structure in the boundary_data
structure must either be NULL or a structure allocated and initialized
by the user.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMBND_append_bnd_from_face
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    tag_t face,
    UF_CAMBND_boundary_data_p_t boundary_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
tag_t	**face**	Input	the face tag from which the boundary will be created
UF_CAMBND_boundary_data_p_t	**boundary_data**	Input	the boundary data

UF_CAMBND_append_item_ude (view source)

Defined in: uf_cambnd.h

Overview
Appends a boundary member user defined event.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_append_item_ude
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type,
    char * ude_name,
    UF_CAMBND_UDE_t * ude,
    logical * response
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events.
char *	**ude_name**	Input	the name of the user defined event
UF_CAMBND_UDE_t *	**ude**	Output	the object of the user deined events
logical *	**response**	Output	the response. success = TRUE, fail = FALSE.

UF_CAMBND_ask_boundaries (view source)

Defined in: uf_cambnd.h

Overview
Gets the list of boundaries of geometry type from the object.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMBND_ask_boundaries
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    int * count,
    UF_CAMBND_boundary_t * * boundaries
```

)

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
int *	**count**	Output	the count of boundaries
UF_CAMBND_boundary_t * *	**boundaries**	Output to UF_*free*	the list boundary items it must be freed using UF_free

---

UF_CAMBND_ask_boundary_app_data (view source)

Defined in: uf_cambnd.h

Overview
Gets the application data of the boundary.

The memory for app_data must be allocated by the user.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMBND_ask_boundary_app_data
(
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_app_data_t * app_data
)
```

UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_app_data_t *	**app_data**	Output	the application data

---

UF_CAMBND_ask_boundary_data (view source)

Defined in: uf_cambnd.h

Overview
Gets the boundary data without the application data of the boundary.

The memory for boundary_data must be allocated by the user.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMBND_ask_boundary_data
(
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_boundary_data_t * boundary_data
)
```

UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_boundary_data_t *	**boundary_data**	Output	the boundary data

---

UF_CAMBND_ask_boundary_group_data (view source)

Defined in: uf_cambnd.h

Overview
Gets the group data of the boundary.

The memory for group_data must be allocated by the user.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_ask_boundary_group_data
(
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_group_data_t * group_data
)
```

UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_group_data_t *	**group_data**	Output	the boundary group data

---

UF_CAMBND_ask_boundary_items (view source)

Defined in: uf_cambnd.h

Overview
Gets the members of the boundary.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMBND_ask_boundary_items
(
    UF_CAMBND_boundary_t boundary,
    int * count,
    UF_CAMBND_item_t * * items
```

)

UF_CAMBND_boundary_t	**boundary**	Input	the boundary
int *	**count**	Output	the count of boundary members
UF_CAMBND_item_t * *	**items**	Output to UF_*free*	the list of boundary members it must be freed using UF_free

UF_CAMBND_ask_item_app_data (view source)

Defined in: uf_cambnd.h

Overview

Gets the application data of the member.

The memory for app_data must be allocated by the user.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMBND_ask_item_app_data
(
    UF_CAMBND_item_t item,
    UF_CAMBND_app_data_t * app_data
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_app_data_t *	**app_data**	Output	the application data of the boundary member

UF_CAMBND_ask_item_entity (view source)

Defined in: uf_cambnd.h

Overview

Gets the application data of the member.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMBND_ask_item_entity
(
    UF_CAMBND_item_t item,
    tag_t * entity
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
tag_t *	**entity**	Output	the geometry tag of the boundary member

UF_CAMBND_ask_item_group_data (view source)

Defined in: uf_cambnd.h

Overview
Gets the group data of the boundary member.

The memory for group_data must be allocated by the user.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_ask_item_group_data
(
    UF_CAMBND_item_t item,
    UF_CAMBND_group_data_t * group_data
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_group_data_t *	**group_data**	Output	the group data of the boundary member

UF_CAMBND_ask_item_udes (view source)

Defined in: uf_cambnd.h

Overview
Asks the boundary member user defined events.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_ask_item_udes
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type,
    int * num_udes,
    UF_CAMBND_UDE_t * * udes
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events. Either Start or End.
int *	**num_udes**	Output	the count of the user defined events
UF_CAMBND_UDE_t * *	**udes**	Output to UF_*free*	the list of the user defined events

---

UF_CAMBND_can_accept_item_ude (view source)

Defined in: uf_cambnd.h

Overview
Determine whether the boundary member can be set the user defined events.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_can_accept_item_ude
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type,
    char * ude_name,
    logical * response
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events.
char *	**ude_name**	Input	the name of the user defined event
logical *	**response**	Output	the response. Can be set = TRUE, can not be set = FALSE

UF_CAMBND_delete_all_item_udes (view source)

Defined in: uf_cambnd.h

Overview
Deletes all boundary member user defined events.

Environment

Internal and External

History

Released in V19.0

```
int UF_CAMBND_delete_all_item_udes
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events. Either Start or End.

UF_CAMBND_delete_boundaries (view source)

Defined in: uf_cambnd.h

Overview

Deletes all the boundaries of geometry type from the object.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMBND_delete_boundaries
(
    tag_t object_tag,
    UF_CAM_geom_type_t type
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary

UF_CAMBND_delete_boundary (view source)

Defined in: uf_cambnd.h

Overview

Deletes a boundary of the boundary type from the object.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMBND_delete_boundary
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_CAMBND_boundary_t boundary
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_CAMBND_boundary_t	**boundary**	Input	the item to be deleted

---

UF_CAMBND_delete_item_ude (view source)

Defined in: uf_cambnd.h

Overview
Deletes a boundary member user defined event.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_delete_item_ude
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type,
    UF_CAMBND_UDE_t ude
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events.
UF_CAMBND_UDE_t	**ude**	Input	the object of the user deined events

---

UF_CAMBND_is_inherited (view source)

Defined in: uf_cambnd.h

Overview
Sets the group data of the boundary member.

Environment
Internal and External

History
Released in V19.0

int UF_CAMBND_is_inherited
```
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    logical * response
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
logical *	**response**	Output	the response, inherited = TRUE, not inherited = FALSE

UF_CAMBND_set_boundary_app_data (view source)

Defined in: uf_cambnd.h

Overview
Sets the boundary application data.

Environment
Internal and External

History
Released in V18.0

int UF_CAMBND_set_boundary_app_data
```
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_app_data_p_t app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_app_data_p_t	**app_data**	Input	the application data

UF_CAMBND_set_boundary_group_data (view source)

Defined in: uf_cambnd.h

Overview
Sets the boundary group data.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_set_boundary_group_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_group_data_p_t group_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_group_data_p_t	**group_data**	Input	the boundary group data

---

UF_CAMBND_set_boundary_plane (view source)

Defined in: uf_cambnd.h

Overview
Sets the boundary plane.

Environment
Internal and External

History
Released in NX4.0

```
int UF_CAMBND_set_boundary_plane
(
    UF_CAMBND_boundary_t boundary,
    double bnd_origin [ 3 ] ,
    double bnd_matrix [ 9 ]
)
```

UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
double	**bnd_origin [3]**	Input	the plane origin (WCS)
double	**bnd_matrix [9]**	Input	the plane matrix

UF_CAMBND_set_item_app_data (view source)

Defined in: uf_cambnd.h

Overview
Sets the application data of the member.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMBND_set_item_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_item_t item,
    UF_CAMBND_app_data_p_t app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_app_data_p_t	**app_data**	Input	the application data of the boundary member

---

UF_CAMBND_set_item_group_data (view source)

Defined in: uf_cambnd.h

Overview
Sets the group data of the boundary member.

Environment
Internal and External

History
Released in V19.0

```
int UF_CAMBND_set_item_group_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_CAMBND_boundary_t boundary,
    UF_CAMBND_item_t item,
    UF_CAMBND_group_data_p_t group_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_CAMBND_boundary_t	**boundary**	Input	the boundary item
UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_group_data_p_t	**group_data**	Input	the group data of the boundary member

UF_CAMBND_WEDM_append_item_ude (view source)

Defined in: uf_cambnd.h

Overview
Appends a WEDM boundary member user defined event.

Environment
Internal and External

History
Released in NX6.0

```
int UF_CAMBND_WEDM_append_item_ude
(
    UF_CAMBND_item_t item,
    int pass_num,
    UF_CAMBND_UDE_set_type_t set_type,
    char * ude_name,
    UF_CAMBND_UDE_t * ude,
    logical * response
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
int	**pass_num**	Input	Pass number
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events.
char *	**ude_name**	Input	the name of the user defined event
UF_CAMBND_UDE_t *	**ude**	Output	the object of the user deined events
logical *	**response**	Output	the response. success = TRUE, fail = FALSE.

UF_CAMBND_WEDM_ask_item_udes (view source)

Defined in: uf_cambnd.h

Overview
Asks the boundary member user defined events for WEDM geometry.

Environment
Internal and External

History
Released in NX6.0

```
int UF_CAMBND_WEDM_ask_item_udes
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type,
    int pass_num,
    int * num_udes,
    UF_CAMBND_UDE_t * * udes
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events. Either Start or End.
int	**pass_num**	Input	pass number for which the events are to be known
int *	**num_udes**	Output	the count of the user defined events
UF_CAMBND_UDE_t * *	**udes**	Output	the list of the user defined events

---

UF_CAMBND_WEDM_delete_all_item_udes (view source)

Defined in: uf_cambnd.h

Overview
Deletes all WEDM boundary member user defined events.

Environment
Internal and External

History
Released in NX6.0

```
int UF_CAMBND_WEDM_delete_all_item_udes
(
    UF_CAMBND_item_t item,
    UF_CAMBND_UDE_set_type_t set_type
)
```

UF_CAMBND_item_t	**item**	Input	the WEDM boundary member

UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events. Either Start or End.

UF_CAMBND_WEDM_delete_item_ude (view source)

Defined in: uf_cambnd.h

Overview
Deletes a Wedm boundary member user defined event.

Environment
Internal and External

History
Released in NX6.0

```
int UF_CAMBND_WEDM_delete_item_ude
(
    UF_CAMBND_item_t item,
    int pass_num,
    UF_CAMBND_UDE_set_type_t set_type,
    UF_CAMBND_UDE_t ude
)
```

UF_CAMBND_item_t	**item**	Input	the boundary member
int	**pass_num**	Input	pass number for which events are specified
UF_CAMBND_UDE_set_type_t	**set_type**	Input	the type of the user defined events.
UF_CAMBND_UDE_t	**ude**	Input	the object of the user deined events

UF_CAMGEOM_append_custom_points (view source)

Defined in: uf_camgeom.h

Overview
Appends a list of custom points to the operation.

The data used varies by operation type:
Planar Milling:
Predrill Engage Points: A single depth value (may be 0.0).
Cut Region Start Points: Upper Depth and Lower Depth values (may be 0.0).
Cavity Milling:
Predrill Engage Points: A single depth value (may be 0.0).
Cut Region Start Points: Upper Depth and Lower Depth values (may be 0.0).
Face Milling:
Predrill Engage Points: No depth used.
Cut Region Start Points: No depth used.
Surface Contouring:
Predrill Engage Points: Not valid.

Cut Region Start Points: No depth used.

Environment

Internal and External

History

Released in V18.0

int UF_CAMGEOM_append_custom_points
(
tag_t object_tag,
UF_CAMGEOM_custom_point_type_t point_type,
int count,
UF_CAMGEOM_custom_point_p_t * point_data
)

tag_t	**object_tag**	Input	the target operation
UF_CAMGEOM_custom_point_type_t	**point_type**	Input	the type of the point
int	**count**	Input	the count of point entities
UF_CAMGEOM_custom_point_p_t *	**point_data**	Input	list of point data pointers

UF_CAMGEOM_append_items (view source)

Defined in: uf_camgeom.h

Overview

Appends a list of geometry entities to the object .

The allowed types of entities are the solid body, solid sheet body, face,
and curve for the most objects and their geometry types, except the following:

Part Geometry of Facing Operation allows only solid body and solid sheet body.
Cut Area allows only solid sheet body and face.
Trim Geometry is not allowed.

If the entity list contains any invalid entities, an error code is returned
and no geometry entity is appended.

Environment

Internal and External

History

Released in V18.0

int UF_CAMGEOM_append_items
(
tag_t object_tag,
UF_CAM_geom_type_t geometry_type,
int count,
tag_t * entity_list,
UF_CAMGEOM_app_data_p_t * app_data
)

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry
int	**count**	Input	the count of geometry entities
tag_t *	**entity_list**	Input	count the list of geometry entities
UF_CAMGEOM_app_data_p_t *	**app_data**	Input	count the list of app_data pointers for each geometry entities

---

UF_CAMGEOM_ask_collector_items (view source)

Defined in: uf_camgeom.h

Overview

Gets all the geometry items of a collector in the geometry list and returns entity occurrences of the underlying geometry.

NOTE: Any app data for the items held by the collector is stored with the collector and must be accessed using its UF_CAMGEOM_item_t.

Environment

Internal and External

History

Released in NX8.0.1

```
int UF_CAMGEOM_ask_collector_items
(
    UF_CAMGEOM_item_t object_tag,
    int * count,
    tag_t * * items
)
```

UF_CAMGEOM_item_t	**object_tag**	Input	geometry item containing a collector
int *	**count**	Output	the count of geometry items
tag_t * *	**items**	Output to UF_*free*	the list of geometry item tags. it must be freed using UF_free

---

UF_CAMGEOM_ask_custom_points (view source)

Defined in: uf_camgeom.h

Overview

Gets the list of custom points and their app_data for the operation.

The data used varies by operation type:
Planar Milling:

Predrill Engage Points: A single depth value (may be 0.0).
Cut Region Start Points: Upper Depth and Lower Depth values (may be 0.0).
Cavity Milling:
Predrill Engage Points: A single depth value (may be 0.0).
Cut Region Start Points: Upper Depth and Lower Depth values (may be 0.0).
Face Milling:
Predrill Engage Points: Not used.
Cut Region Start Points: Not used
Surface Contouring:
Predrill Engage Points: Not valid.
Cut Region Start Points: No depth used.

NOTE:
Space for the point data list will be allocated by this function.
It must be freed by the calling program using UF_free on each entry in
the list and then the list pointer itself when the information
is no longer needed.

Environment
Internal and External

History
Released in NX3.0

```
int UF_CAMGEOM_ask_custom_points
(
    tag_t object_tag,
    UF_CAMGEOM_custom_point_type_t point_type,
    int * count,
    UF_CAMGEOM_custom_point_p_t * * point_data
)
```

tag_t	**object_tag**	Input	the target operation
UF_CAMGEOM_custom_point_type_t	**point_type**	Input	the type of the point
int *	**count**	Output	the count of point entities
UF_CAMGEOM_custom_point_p_t * *	**point_data**	Output to UF_*free*	list of point data pointers for each point entity

UF_CAMGEOM_ask_geom_provider (view source)

Defined in: uf_camgeom.h

Overview
Gets the object providing the specified geometry type to the input object.

Environment
Internal and External

Note: Input object may be either an operation or geometry group

History
Released in NX3.0

```
int UF_CAMGEOM_ask_geom_provider
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    tag_t * provider_tag
)
```

tag_t	**object_tag**	Input	the object being provided geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry
tag_t *	**provider_tag**	Output	tag of the object providing the geometry to object

---

UF_CAMGEOM_ask_item_app_data (view source)

Defined in: uf_camgeom.h

Overview

Gets the application data of the item.

The memory for application data must be allocated by the user.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMGEOM_ask_item_app_data
(
    UF_CAMGEOM_item_t item,
    UF_CAMGEOM_app_data_t * app_data
)
```

UF_CAMGEOM_item_t	**item**	Input	the geometry item
UF_CAMGEOM_app_data_t *	**app_data**	Output	the application data of the item

---

UF_CAMGEOM_ask_item_entity (view source)

Defined in: uf_camgeom.h

Overview

Gets the geometry entity of the item.

Environment

Internal and External

History

Released in V18.0

```
int UF_CAMGEOM_ask_item_entity
(
    UF_CAMGEOM_item_t item,
    tag_t * entity
)
```

UF_CAMGEOM_item_t	**item**	Input	the geometry item
tag_t *	**entity**	Output	the item entity

UF_CAMGEOM_ask_item_maxmin (view source)

Defined in: uf_camgeom.h

Overview
Returns the parametric max-min box for a face geom_item

Environment
Internal and External

History
Released in NX3.0

```
int UF_CAMGEOM_ask_item_maxmin
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    tag_t entity,
    double * maxmin
)
```

tag_t	**object_tag**	Input	the parent object
UF_CAM_geom_type_t	**geometry_type**	Input	Part, drive, check, etc.
tag_t	**entity**	Input	entity to be evaluated
double *	**maxmin**	Output	u,v maxmin box of face

UF_CAMGEOM_ask_items (view source)

Defined in: uf_camgeom.h

Overview
Gets all the geometry items of given type from the object.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMGEOM_ask_items
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    int * count,
    UF_CAMGEOM_item_t * * items
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry
int *	**count**	Output	the count of geometry items
UF_CAMGEOM_item_t * *	**items**	Output to UF_*free*	the list of geometry items it must be freed using UF_free

UF_CAMGEOM_delete_custom_points (view source)

Defined in: uf_camgeom.h

Overview
Deletes a list of custom points from the operation.

Environment
Internal and External

History
Released in NX3.0

```
int UF_CAMGEOM_delete_custom_points
(
    tag_t object_tag,
    UF_CAMGEOM_custom_point_type_t point_type
)
```

tag_t	**object_tag**	Input	the target operation
UF_CAMGEOM_custom_point_type_t	**point_type**	Input	the type of the point

UF_CAMGEOM_delete_geometry (view source)

Defined in: uf_camgeom.h

Overview
Deletes all the geometry items of the given type from the object.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMGEOM_delete_geometry
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry

---

UF_CAMGEOM_delete_item (view source)

Defined in: uf_camgeom.h

Overview
Deletes the item from the geometry of the given type.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMGEOM_delete_item
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    UF_CAMGEOM_item_t item
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry
UF_CAMGEOM_item_t	**item**	Input	the item to be deleted

UF_CAMGEOM_eval_surface (view source)

Defined in: uf_camgeom.h

Overview
Evaluates a surface at a input parameter position.

Environment
Internal and External

History
Released in NX3.0

```
int UF_CAMGEOM_eval_surface
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    tag_t entity,
    double uv [ 2 ] ,
    UF_MODL_SRF_VALUE_p_t srf_value
)
```

tag_t	**object_tag**	Input	the parent object
UF_CAM_geom_type_t	**geometry_type**	Input	Part, drive, check, etc.
tag_t	**entity**	Input	entity to be evaluated
double	**uv [2]**	Input	u,v parameter position
UF_MODL_SRF_VALUE_p_t	**srf_value**	Output	evaluation data structure

---

UF_CAMGEOM_set_item_app_data (view source)

Defined in: uf_camgeom.h

Overview
Sets the application data of the item.

Environment
Internal and External

History
Released in V18.0

```
int UF_CAMGEOM_set_item_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t geometry_type,
    UF_CAMGEOM_item_t item,
    UF_CAMGEOM_app_data_p_t app_data
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAM_geom_type_t	**geometry_type**	Input	the type of the geometry
UF_CAMGEOM_item_t	**item**	Input	the geometry item
UF_CAMGEOM_app_data_p_t	**app_data**	Input	the application data of the item

---

UF_CAMTEXT_append_items (view source)

Defined in: uf_camtext.h

Overview

Appends a list of geometry entities to the object .

The allowed types of entities are drafting note and drafting label objects.
If the entity list contains any invalid entities, an error code is returned
and no geometry entity is appended.

Environment

Internal and External

History

Released in NX2.0

```
int UF_CAMTEXT_append_items
(
    tag_t object_tag,
    int count,
    tag_t * entity_list
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
int	**count**	Input	the count of geometry entities
tag_t *	**entity_list**	Input	the list of geometry entities

UF_CAMTEXT_ask_item_entity (view source)

Defined in: uf_camtext.h

Overview

Gets the geometry entity of the item.

Environment

Internal and External

History

Released in NX2.0

```
int UF_CAMTEXT_ask_item_entity
(
    UF_CAMTEXT_item_t item,
    tag_t * entity
)
```

UF_CAMTEXT_item_t	**item**	Input	the geometry item
tag_t *	**entity**	Output	the item entity

UF_CAMTEXT_ask_items (view source)

Defined in: uf_camtext.h

Overview
Gets all the text geometry items from the object.

Environment
Internal and External

History
Released in NX2.0

```
int UF_CAMTEXT_ask_items
(
    tag_t object_tag,
    int * count,
    UF_CAMTEXT_item_t * * items
)
```

tag_t	**object_tag**	Input	the parent object of the geometry
int *	**count**	Output	the count of geometry items
UF_CAMTEXT_item_t * *	**items**	Output to UF_*free*	the list of geometry items it must be freed using UF_free

---

UF_CAMTEXT_delete_geometry (view source)

Defined in: uf_camtext.h

Overview
Deletes all the text geometry items from the object.

Environment
Internal and External

History
Released in NX2.0

```
int UF_CAMTEXT_delete_geometry
(
    tag_t object_tag
)
```

tag_t	**object_tag**	Input	the parent object of the geometry

UF_CAMTEXT_delete_item (view source)

Defined in: uf_camtext.h

Overview
Deletes one text geometry item from the object.

Environment

Internal and External

History

Released in NX2.0

int UF_CAMTEXT_delete_item
(
    tag_t object_tag,
    UF_CAMTEXT_item_t item
)

tag_t	**object_tag**	Input	the parent object of the geometry
UF_CAMTEXT_item_t	**item**	Input	the item to be deleted

UF_CLSF_import (view source)

Defined in: uf_clsf.h

Overview

Function Name: UF_CLSF_import

Function Description:
This function imports the CLS tool paths into the part specified

int UF_CLSF_import
(
    tag_t part_tag,
    char * clsf_name
)

tag_t	**part_tag**	Input	the NX part tag
char *	**clsf_name**	Input	the name of the cls file from which the tool paths are imported

---

UF_CUT_LEVELS_add_levels_using_geom (view source)

Defined in: uf_cut_levels.h

Overview

Adds cut levels defined by face and/or point tags to the list of cut levels.

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if a change could not be made.

Will return UF_CAM_ERROR_INVALID_CUT_LEVEL_ENTITY if a geometry tag did not specify a valid face or point.

The range type, as specified by UF_PARAM_CUTLEV_RANGE_TYPE, should not be UF_PARAM_clv_range_single as levels can not be added to this range type. The error UF_CAM_ERROR_SINGLE_RANGE_NOT_CHANGED will be returned if this happens.

Environment
Internal or External

History
Released in NX3

```
int UF_CUT_LEVELS_add_levels_using_geom
(
    tag_t operation_tag,
    int num_to_add,
    tag_t * geom_tags,
    double max_depth_per_cut,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
int	**num_to_add**	Input	The number of face and/or point tags being added. Value must be non negative.
tag_t *	**geom_tags**	Input	An array of face and/or point tags defining the new cut level. If faces are not horizontal (with respect to the tool axis), or even flat, then the heighest z level of the face will be used for the cut level.
double	**max_depth_per_cut**	Input	The max depth per cut to be used for subdividing the range above the new cut levels. A value of zero indicates that there is no limit. Value must be non negative.
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the addition. If NULL is passed in, then the argument is ignored.

UF_CUT_LEVELS_add_levels_using_z (view source)

Defined in: uf_cut_levels.h

Overview
Adds cut levels defined by explicit z-levels

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if a change could not be made.

The range type, as specified by UF_PARAM_CUTLEV_RANGE_TYPE, should not be UF_PARAM_clv_range_single as levels can not be added to this range type. The error UF_CAM_ERROR_SINGLE_RANGE_NOT_CHANGED will be returned if this happens.

Environment
Internal or External

History
Released in NX3

```
int UF_CUT_LEVELS_add_levels_using_z
(
    tag_t operation_tag,
    int num_to_add,
    double * z_levels,
    double max_depth_per_cut,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
int	**num_to_add**	Input	The number of z-levels being added. Value must be non negative.
double *	**z_levels**	Input	Array of z distances, along the tool axis, of the cut level from the origin.
double	**max_depth_per_cut**	Input	The max depth per cut to be used for subdividing the range above the new cut levels. A value of zero indicates that there is no limit. Value must be non negative.
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the addition. If NULL is passed in, then the argument is ignored.

UF_CUT_LEVELS_ask_level (view source)

Defined in: uf_cut_levels.h

Overview

Returns the data for the specified cut level.

Note this returns the data from the UF_CUT_LEVELS_t object which is not associative to the part. Use UF_CUT_LEVELS_load to ensure you have an up to date version of the data.

Environment

Internal or External

History

Released in NX3

```
int UF_CUT_LEVELS_ask_level
(
    UF_CUT_LEVELS_t * cut_levels,
    int index,
    UF_CUT_LEVEL_single_t * * level_data_ptr_addr
)
```

UF_CUT_LEVELS_t *	**cut_levels**	Input	The loaded cut levels data structure.

int	**index**	Input	The index of the cut level the data is required for (0 being the top level).
UF_CUT_LEVEL_single_t * *	**level_data_ptr_addr**	Output	Address of the pointer to the data of the requested single level. The address must not be NULL on input. The data includes: - The tag of the face or point entity used to define the current cut level. This can be NULL if the level was defined by an explicit z level. - The z distance of the cut level down from the top cut level. All values will be non-negative. - The max depth per cut value to be used for the range between this cut level and the one above it.

UF_CUT_LEVELS_ask_top_off_level (view source)

Defined in: uf_cut_levels.h

Overview

Returns the data for the specified top off level

Note this returns the data from the UF_CUT_LEVELS_t object which is not associative to the part. Use UF_CUT_LEVELS_load to ensure you have an up to date version of the data.

Environment

Internal or External

History

Released in NX3

```
int UF_CUT_LEVELS_ask_top_off_level
(
    UF_CUT_LEVELS_t * cut_levels,
    int index,
    UF_CUT_LEVEL_single_t * * level_data_ptr_addr
)
```

UF_CUT_LEVELS_t *	**cut_levels**	Input	The loaded cut levels data structure.
int	**index**	Input	The index of the top off level the data is required for (0 being the top level).
UF_CUT_LEVEL_single_t * *	**level_data_ptr_addr**	Output	Address of the pointer to the data of the requested single level. The address must not be NULL on input. The data includes: - The tag of the face or point entity used to define the current cut level. This can be NULL if the level was defined by an explicit z level. - The z distance of the cut level down from the top cut level. All values will be non-negative. - The max depth per cut value to be used

			for the range between this cut level and the one above it.

UF_CUT_LEVELS_delete_level (view source)

Defined in: uf_cut_levels.h

Overview

Deletes the specified cut level.

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if change could not be made.

The range type, as specified by UF_PARAM_CUTLEV_RANGE_TYPE, should not be UF_PARAM_clv_range_single as levels can not be deleted with this range type.
The error UF_CAM_ERROR_SINGLE_RANGE_NOT_CHANGED will be returned if this happens.

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if deletion could not be made.

Environment

Internal or External

History

Released in NX3

```
int UF_CUT_LEVELS_delete_level
(
    tag_t operation_tag,
    int delete_level,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
int	**delete_level**	Input	The index of the cut level to be deleted (0 being the top level).
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the deletion. If NULL is passed in, then the argument is ignored.

UF_CUT_LEVELS_edit_level_using_geom (view source)

Defined in: uf_cut_levels.h

Overview

Edits an existing cut level using a face or point. Any levels between the old and new location will be removed.

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if change could not be made.

Will return UF_CAM_ERROR_INVALID_CUT_LEVEL_ENTITY if a geometry tag did not specify a valid face or point.

The range type, as specified by UF_PARAM_CUTLEV_RANGE_TYPE, should not be UF_PARAM_clv_range_single as levels can not be added to this range type.

Environment

Internal or External

History

Released in NX3

```
int UF_CUT_LEVELS_edit_level_using_geom
(
    tag_t operation_tag,
    int edit_level,
    tag_t geom_tag,
    double max_depth_per_cut,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
int	**edit_level**	Input	The index of the level to be edited (0 being the top level).
tag_t	**geom_tag**	Input	The tag of the face or point defining the new cut level.
double	**max_depth_per_cut**	Input	The max depth per cut to be used for subdividing the range above the new cut level.
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the addition. If NULL is passed in, then the argument is ignored.

UF_CUT_LEVELS_edit_level_using_z (view source)

Defined in: uf_cut_levels.h

Overview

Edits an existing cut level defined using an explicit z-level. Any levels between the old and new location will be removed.

Will return UF_CAM_ERROR_CUT_LEVEL_CHANGE_NOT_MADE if change could not be made.

Environment

Internal or External

History

Released in NX3

```
int UF_CUT_LEVELS_edit_level_using_z
(
    tag_t operation_tag,
    int edit_level,
    double z_level,
    double max_depth_per_cut,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
int	**edit_level**	Input	The index of the level to be edited (0 being the top level).
double	**z_level**	Input	The z distance, along the tool axis, of the cut level from the origin.
double	**max_depth_per_cut**	Input	The max depth per cut to be used for subdividing the range above the new cut level.
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the addition. If NULL is passed in, then the argument is ignored.

UF_CUT_LEVELS_free (view source)

Defined in: uf_cut_levels.h

Overview
Frees the memory associated with a cut levels structure when the user has finished with it.

Environment
Internal or External

See Also
UF_CUT_LEVELS_load

History
Released in NX3

```
int UF_CUT_LEVELS_free
(
    UF_CUT_LEVELS_t * * cut_levels_ptr_addr
)
```

UF_CUT_LEVELS_t * *	**cut_levels_ptr_addr**	Output to UF_*free*	The data structure create by the call to UF_CUT_LEVELS_load. The memory associated with this structure is freed, and a NULL pointer is returned.

UF_CUT_LEVELS_load (view source)

Defined in: uf_cut_levels.h

Overview

Loads the current cut levels for the specified operation into the data
structure UF_CUT_LEVELS_t.

Note that this creates a copy of the current cut levels, and will not
be associative to the part.

Note that top off levels are stored separately from the other cut levels
and that top off levels can not be changed. They are only available if the
UF_PARAM_CUTLEV_RANGE_TYPE_INDEX parameter is set to UF_PARAM_clv_range_single.
Top off levels are set at levels where there are horizontal (relative to
the tool axis) regions of the part which face upwards. There is no top off
level when it coincides with a cut level.

Will return UF_CAM_ERROR_CUT_LEVELS_NOT_SUPPORTED if cut levels are not
supported for the specified operation.

Environment

Internal or External

See Also

UF_CUT_LEVELS_free

History

Released in NX3

```
int UF_CUT_LEVELS_load
(
    tag_t operation_tag,
    UF_CUT_LEVELS_t * * cut_levels_ptr_addr
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
UF_CUT_LEVELS_t * *	**cut_levels_ptr_addr**	Output to UF_*free*	A new data structure containing all of the cut levels for the specified operation. The pointer to the structure must be NULL on input, or it must be a valid pointer created using a previous call to UF_CUT_LEVELS_load. The pointer must be freed by UF_CUT_LEVELS_free.

UF_CUT_LEVELS_reset_to_default (view source)

Defined in: uf_cut_levels.h

Overview

Resets the cut levels to their default.

Environment

Internal or External

History
Released in NX3

```
int UF_CUT_LEVELS_reset_to_default
(
    tag_t operation_tag
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.

UF_CUT_LEVELS_set_range_type (view source)

Defined in: uf_cut_levels.h

Overview
Set the range type to that specified. This sets the parameter UF_PARAM_CUTLEV_RANGE_TYPE_INDEX, which is otherwise read only, and updates the cut levels data structure.

Environment
Internal or External

History
Released in NX3

```
int UF_CUT_LEVELS_set_range_type
(
    tag_t operation_tag,
    UF_PARAM_clv_range_type_t range_type,
    UF_CUT_LEVELS_t * cut_levels
)
```

tag_t	**operation_tag**	Input	The tag of the operation for which the cut levels are defined.
UF_PARAM_clv_range_type_t	**range_type**	Input	The range type to be set.
UF_CUT_LEVELS_t *	**cut_levels**	Output to UF_*free*	If the data structure created by a call to UF_CUT_LEVELS_load is passed in, then it will be updated to reflect the changes from the new range type. If NULL is passed in, then the argument is ignored.

UF_CUTTER_ask_holder_data (view source)

Defined in: uf_cutter.h

Overview

Query a cutter for the data of a tool cylindrical section holder

Return

UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment

Internal and External

History

Released in NX2.0

```
int UF_CUTTER_ask_holder_data
(
    tag_t object_tag,
    int * count,
    UF_CUTTER_holder_section_t * * * data
)
```

tag_t	**object_tag**	Input	the tag of the cutter
int *	**count**	Output	number of holder sections
UF_CUTTER_holder_section_t * * *	**data**	Output to UF_*free*	count array of structure pointers for each cylindrical section

---

UF_CUTTER_ask_section_count (view source)

Defined in: uf_cutter.h

Overview

Query a cutter for the number of cylindrical sections defining the holder

Return

UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment

Internal and External

History

Released in NX2.0

```
int UF_CUTTER_ask_section_count
(
    tag_t object_tag,
    int * count
)
```

tag_t	**object_tag**	Input	the tag of the cutter
int *	**count**	Output	the number of cylindrical sections

UF_CUTTER_ask_tracking_point_count (view source)

Defined in: uf_cutter.h

Overview
Query the number of tracking points in the Cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment
Internal and External

See Also
UF_CUTTER_ask_tracking_point_data

History
Released in NX2.0

```
int UF_CUTTER_ask_tracking_point_count
(
    tag_t object_tag,
    int * count
)
```

tag_t	**object_tag**	Input	the parent object of the points
int *	**count**	Output	the number of tracking points

UF_CUTTER_ask_tracking_point_data (view source)

Defined in: uf_cutter.h

Overview
Query the tracking point data for a cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment
Internal and External

History
Released in NX2.0

```
int UF_CUTTER_ask_tracking_point_data
(
    tag_t object_tag,
    int * count,
    UF_CUTTER_tracking_point_data_t * * * data
)
```

tag_t	**object_tag**	Input	the parent of the point

int *	**count**	Output	number of tracking points
UF_CUTTER_tracking_point_data_t * * *	**data**	Output to UF_*free*	count the data for the points

---

UF_CUTTER_ask_turn_tracking_point_data (view source)

Defined in: uf_cutter.h

Overview
Query the turn tracking point data for a cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment
Internal and External

History
Released in NX5.0

```
int UF_CUTTER_ask_turn_tracking_point_data
(
    tag_t object_tag,
    int * count,
    UF_CUTTER_turn_tracking_point_data_t * * * data
)
```

tag_t	**object_tag**	Input	the parent of the point
int *	**count**	Output	number of tracking points
UF_CUTTER_turn_tracking_point_data_t * * *	**data**	Output to UF_*free*	count the data for the points

---

UF_CUTTER_ask_type_and_subtype (view source)

Defined in: uf_cutter.h

```
int UF_CUTTER_ask_type_and_subtype
(
    tag_t object_id,
    int * type,
    int * subtype
)
```

tag_t	**object_id**	Input	Object identifier of CUTTER
int *	**type**	Output	CUTTER Type

int *	**subtype**	Output	CUTTER Subtype

UF_CUTTER_create (view source)

Defined in: uf_cutter.h

Overview
This function creates a Cutter based upon the Cutter template object specified. All parameters of the newly created Cutter are derived from the specified Cutter template object.

Environment
Internal and External

History
Released in V16.0

```
int UF_CUTTER_create
(
    char * type_name,
    char * subtype_name,
    tag_t * new_object
)
```

char *	**type_name**	Input	- the template type name of the desired Cutter template object.
char *	**subtype_name**	Input	- the template subtype name of the desired Cutter template object.
tag_t *	**new_object**	Output	- the tag of the newly created Cutter object

UF_CUTTER_create_holder_section (view source)

Defined in: uf_cutter.h

Overview
Create a new holder section and append it to the holder definition

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter
UF_CAM_ERROR_INVALID_RADIUS
UF_CAM_ERROR_INVALID_DIAMETER
UF_CAM_ERROR_INVALID_LENGTH
UF_CAM_ERROR_INVALID_TAPER
The input structure contains invalid data

Environment
Internal and External

History
Released in NX2.0

```
int UF_CUTTER_create_holder_section
(
    tag_t object_tag,
    UF_CUTTER_holder_section_t * data
)
```

tag_t	**object_tag**	Input	the tag of the cutter
UF_CUTTER_holder_section_t *	**data**	Input	the data of the section

---

UF_CUTTER_create_tracking_point (view source)

Defined in: uf_cutter.h

Overview
Create a new tracking point and add it to the input Cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment
Internal and External

History
Released in NX2.0

```
int UF_CUTTER_create_tracking_point
(
    tag_t object_tag,
    UF_CUTTER_tracking_point_data_t * data
)
```

tag_t	**object_tag**	Input	the parent of the point
UF_CUTTER_tracking_point_data_t *	**data**	Input	the data of the point

---

UF_CUTTER_create_turn_tracking_point (view source)

Defined in: uf_cutter.h

Overview
Create a new turn tracking point and add it to the input Cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment
Internal and External

History
Released in NX5.0

int UF_CUTTER_create_turn_tracking_point
(
  tag_t object_tag,
  UF_CUTTER_turn_tracking_point_data_t * data
)

tag_t	**object_tag**	Input	the parent of the point
UF_CUTTER_turn_tracking_point_data_t *	**data**	Input	the data of the point

---

UF_CUTTER_delete_holder_section (view source)

Defined in: uf_cutter.h

Overview
Delete a specific tool cylindrical section

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter
UF_CAM_ERROR_INVALID_INDEX
No holder section with this index exists in the cutter.

Environment
Internal and External

History
Released in NX2.0

int UF_CUTTER_delete_holder_section
(
  tag_t object_tag,
  int index
)

tag_t	**object_tag**	Input	the tag of the cutter
int	**index**	Input	index to desired section to delete

UF_CUTTER_delete_tracking_point (view source)

Defined in: uf_cutter.h

Overview
Delete a specified tracking point from the specified cutter.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment

Internal and External

History

Released in NX2.0

```
int UF_CUTTER_delete_tracking_point
(
    tag_t cutter_tag,
    int index
)
```

tag_t	**cutter_tag**	Input	the parent cutter of the point
int	**index**	Input	index of tracking point in parent (from 0 to the number of tracking points -1)

UF_CUTTER_edit_holder_section (view source)

Defined in: uf_cutter.h

Overview

Modify the data for a specific tool cylindrical section

Return

UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter

Environment

Internal and External

History

Released in NX2.0

```
int UF_CUTTER_edit_holder_section
(
    tag_t object_tag,
    int index,
    UF_CUTTER_holder_section_t * data
)
```

tag_t	**object_tag**	Input	the tag of the cutter
int	**index**	Input	index to desired section to modify
UF_CUTTER_holder_section_t *	**data**	Input	modified section data

---

UF_CUTTER_retrieve (view source)

Defined in: uf_cutter.h

Overview

This function retrieves a Cutter from the current Cutter Library (as
defined in cam_config.dat) and creates an NX Cutter Object based upon

the values received from the library.

Environment
Internal and External

History
Released in V16.0

```
int UF_CUTTER_retrieve
(
    const char * libref,
    tag_t * tool_tag
)
```

const char *	**libref**	Input	- the library reference of the desired cutter. Can be gotten from a record set or an existing NX object
tag_t *	**tool_tag**	Output	- the NX object created as a result of the retrieval

---

UF_CUTTER_set_tracking_point_data (view source)

Defined in: uf_cutter.h

Overview
Modify the tracking point data for a specified point.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter
UF_CAM_ERROR_INVALID_INDEX
No tracking point with this index exists in the cutter.

Environment
Internal and External

History
Released in NX2.0

```
int UF_CUTTER_set_tracking_point_data
(
    tag_t object_tag,
    int index,
    UF_CUTTER_tracking_point_data_t * data
)
```

tag_t	**object_tag**	Input	the parent object of the point
int	**index**	Input	index of tracking point in parent (from 0 to the number of tracking points - 1)
UF_CUTTER_tracking_point_data_t *	**data**	Input	the modified data of the point

---

UF_CUTTER_set_turn_tracking_point_data (view source)

Defined in: uf_cutter.h

Overview
Modify the turn tracking point data for a specified point.

Return
UF_CAM_ERROR_TAG_NOT_CORRECT_TYPE
The input tag is not a cutter
UF_CAM_ERROR_INVALID_INDEX
No turn tracking point with this index exists in the cutter.

Environment
Internal and External

History
Released in NX2.0

```
int UF_CUTTER_set_turn_tracking_point_data
(
    tag_t object_tag,
    int index,
    UF_CUTTER_turn_tracking_point_data_t * data
)
```

tag_t	**object_tag**	Input	the parent object of the point
int	**index**	Input	index of tracking point in parent (from 0 to the number of tracking points - 1)
UF_CUTTER_turn_tracking_point_data_t *	**data**	Input	the modified data of the point

---

UF_CUTTER_update_from_lib (view source)

Defined in: uf_cutter.h

Overview
This function updates the data of an already existing Cutter from the current Cutter Library (as defined in cam_config.dat).

Environment
Internal and External

History
Released in V18.0

```
int UF_CUTTER_update_from_lib
(
    tag_t tool_tag
)
```

tag_t	**tool_tag**	Input	- the NX object which shall be updated

UF_FBM_GEOM_ask_accessibility_vectors (view source)

Defined in: uf_fbm_geom.h

Overview
Return the list of accessibility vectors associated to the feature of the FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_accessibility_vectors
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t representative_feature,
    int * count,
    tag_t * * smart_vectors
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**representative_feature**	Input	A representative feature object of the group
int *	**count**	Output	The number of accessibility vectors
tag_t * *	**smart_vectors**	Output to UF_*free*	The array of accessibility vectors. Memory has to be freed calling UF_free

UF_FBM_GEOM_ask_available_criteria (view source)

Defined in: uf_fbm_geom.h

Overview
Return all the available criteria of the FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_available_criteria
(
    tag_t fbm_geom_tag,
    int * count,
    char * * * criteria_list
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int *	**count**	Output	The number of criteria

char * * *	**criteria_list**	Output to UF_*free*	The criteria that are available for classification. Memory is allocated by this function and has to be freed by calling UF_free_string_array

UF_FBM_GEOM_ask_double_of_criteria (view source)

Defined in: uf_fbm_geom.h

Overview

Return the value of a double type of criteria for a
specific feature object that is present in the FBM_GEOM group.
If the feature object is not present in the FBM_GEOM group,
an error will be returned. Also, if the criteria type does
not match, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_double_of_criteria
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t ncfeat_object,
    char * criterion,
    double * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**ncfeat_object**	Input	The ncfeat_obj object for which the criteria value has to be evaluated
char *	**criterion**	Input	The criterion for which the value is requested
double *	**value**	Output	The double value

UF_FBM_GEOM_ask_double_value_of_classified_crit (view source)

Defined in: uf_fbm_geom.h

Overview

Return the double value of a criterion for the set indicated by the
classified_set index . If the type of criterion is not a double, an error
will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_double_value_of_classified_crit
(
    tag_t fbm_geom_tag,
    char * criterion,
    UF_FBM_GEOM_classified_crit_t classified_set_list,
    int classified_set_index,
    double * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**criterion**	Input	The criteria for which the value is asked for
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The result of the classification
int	**classified_set_index**	Input	The index of the set which should be used
double *	**value**	Output	The value of the criterion

UF_FBM_GEOM_ask_feature_entities (view source)

Defined in: uf_fbm_geom.h

Overview
Return the list of entities associated to the feature of the FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_feature_entities
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t representative_feature,
    int * count,
    tag_t * * entities
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**representative_feature**	Input	A representative feature object of the group
int *	**count**	Output	The number of entities of the feature object
tag_t * *	**entities**	Output to UF_*free*	The array of entities of the feature object. Memory has to be freed by calling UF_free

UF_FBM_GEOM_ask_feature_name (view source)

Defined in: uf_fbm_geom.h

Overview
Return the name of the feature in the FBM_GEOM group

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_feature_name
(
    tag_t fbm_geom_tag,
    char * * feature_name
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char * *	**feature_name**	Output to UF_*free*	Name of the feature on which the fbm_geom group is applied. The memory must be freed by calling UF_free

---

UF_FBM_GEOM_ask_features (view source)

Defined in: uf_fbm_geom.h

Overview
Return the feature objects of FBM_GEOM group

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_features
(
    tag_t fbm_geom_tag,
    int * count,
    UF_NCFEAT_t * * ncfeat_objs
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int *	**count**	Output	The number of feature objects returned
UF_NCFEAT_t * *	**ncfeat_objs**	Output to UF_*free*	The array of ncfeat objects. The memory for the objects allocated by this function and must be freed by calling UF_free

---

UF_FBM_GEOM_ask_int_value_of_classified_crit (view source)

Defined in: uf_fbm_geom.h

Overview

Return the integer value of a criterion for the set indicated by the classified_set index . If the type of criterion is not an integer, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_int_value_of_classified_crit
(
    tag_t fbm_geom_tag,
    char * criterion,
    UF_FBM_GEOM_classified_crit_t classified_set_list,
    int classified_set_index,
    int * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**criterion**	Input	The criteria for which the value is asked for
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The result of the classification
int	**classified_set_index**	Input	The index of the set which should be used
int *	**value**	Output	The value of the criterion

UF_FBM_GEOM_ask_integer_of_criteria (view source)

Defined in: uf_fbm_geom.h

Overview

Return the value of a integer type of criteria for a
specific feature object that is present in the FBM_GEOM group.
If the feature object is not present in the FBM_GEOM group,
an error will be returned. Also, if the criteria type does
not match, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_integer_of_criteria
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t ncfeat_obj,
    char * criterion,
    int * value
```

)

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**ncfeat_obj**	Input	The ncfeat_obj object for which the criteria value has to be evaluated
char *	**criterion**	Input	The criterion for which the value is requested
int *	**value**	Output	The integer value

UF_FBM_GEOM_ask_list_of_feature_names (view source)

Defined in: uf_fbm_geom.h

Overview
Return all the names of the features that are valid for the FBM_GEOM group

History
Released in V19.0

```
int UF_FBM_GEOM_ask_list_of_feature_names
(
    tag_t fbm_geom_tag,
    int * count,
    char * * * feature_names
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int *	**count**	Output	The number of feature names returned
char * * *	**feature_names**	Output to UF_*free*	The array of the feature names. The memory for the names allocated by this function and must be freed by calling UF_free_string_array.

UF_FBM_GEOM_ask_logical_of_criteria (view source)

Defined in: uf_fbm_geom.h

Overview
Return the value of a logical type of criteria for a specific feature
object that is present in the FBM_GEOM group.
If the feature object is not present in the FBM_GEOM group, an error
will be returned. Also, if the criteria type does not match, an error
will be returned.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_ask_logical_of_criteria
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t ncfeat_obj,
    char * criterion,
    logical * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**ncfeat_obj**	Input	The ncfeat_obj object for which the criteria value has to be evaluated
char *	**criterion**	Input	The criterion for which the value is requested
logical *	**value**	Output	The logical value

UF_FBM_GEOM_ask_logical_value_of_classified_crit (view source)

Defined in: uf_fbm_geom.h

Overview

Return the logical value of a criterion for the set indicated by the classified_set index . If the type of criterion is not a logical, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_logical_value_of_classified_crit
(
    tag_t fbm_geom_tag,
    char * criterion,
    UF_FBM_GEOM_classified_crit_t classified_set_list,
    int classified_set_index,
    logical * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**criterion**	Input	The criteria for which the value is asked for
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The result of the classification
int	**classified_set_index**	Input	The index of the set which should be used
logical *	**value**	Output	The value of the criterion

UF_FBM_GEOM_ask_representative_features (view source)

Defined in: uf_fbm_geom.h

Overview

Return the feature objects that represent the all other feature objects
in FBM_GEOM group.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_representative_features
(
    tag_t fbm_geom_tag,
    int * count,
    UF_NCFEAT_t * * rep_feature_list
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int *	**count**	Output	The number of representative features in this group
UF_NCFEAT_t * *	**rep_feature_list**	Output to UF_*free*	The array of representative features in this group. The memory is allocated by this function and has to be freed by calling UF_free

---

UF_FBM_GEOM_ask_string_of_criteria (view source)

Defined in: uf_fbm_geom.h

Overview

Return the value of a string type of criteria for a
specific feature object that is present in the FBM_GEOM group.
If the feature object is not present in the FBM_GEOM group,
an error will be returned. Also, if the criteria type does
not match, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_string_of_criteria
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t ncfeat_obj,
    char * criterion,
    char * * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**ncfeat_obj**	Input	The ncfeat_obj object for which the criteria value has to be evaluated

char *	**criterion**	Input	The criterion whose value has to be evaluated
char * *	**value**	Output to UF_*free*	The string value. Memory has to be freed by calling UF_free

---

UF_FBM_GEOM_ask_string_value_of_classified_crit (view source)

Defined in: uf_fbm_geom.h

Overview

Return the string value of a criterion for the set indicated by the classified_set index . If the type of criterion is not a string, an error will be returned.

Environment

Internal and External

History

Released in V19.0

```
int UF_FBM_GEOM_ask_string_value_of_classified_crit
(
    tag_t fbm_geom_tag,
    char * criterion,
    UF_FBM_GEOM_classified_crit_t classified_set_list,
    int classified_set_index,
    char * * value
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**criterion**	Input	The criteria for which the value is asked for
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The result of the classification
int	**classified_set_index**	Input	The index of the set which should be used
char * *	**value**	Output to UF_*free*	The value of the criterion. Memory has to be freed by calling UF_free

---

UF_FBM_GEOM_ask_type_of_criterion (view source)

Defined in: uf_fbm_geom.h

Overview

Return the data type of a criterion of the FBM_GEOM group.

Environment

Internal and External

History

Released in V19.0

int UF_FBM_GEOM_ask_type_of_criterion
(
tag_t fbm_geom_tag,
char * criterion,
UF_FBM_GEOM_crit_value_type_p_t type
)

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**criterion**	Input	The criterion for which the type has to be returned
UF_FBM_GEOM_crit_value_type_p_t	**type**	Output	The type of the criterion

UF_FBM_GEOM_ask_used_criteria (view source)

Defined in: uf_fbm_geom.h

Overview
Return the criteria that has been specified to be used for classification of the feature objects in the FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

int UF_FBM_GEOM_ask_used_criteria
(
tag_t fbm_geom_tag,
int * count,
char * * * used_criteria_list
)

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int *	**count**	Output	The number of criteria
char * * *	**used_criteria_list**	Output to UF_*free*	The criteria that will be used for classification. Memory is allocated by this function and has to be freed by calling UF_free_string_array

UF_FBM_GEOM_classify_by_criteria (view source)

Defined in: uf_fbm_geom.h

Overview

Return the result of the classification based on the given criteria.

Environment
Internal and External

History
Released in V19.0

int UF_FBM_GEOM_classify_by_criteria
(
tag_t fbm_geom_tag,
int num_of_criteria,
char * * criteria,
int * num_of_classified_sets,
UF_FBM_GEOM_classified_crit_t * classified_set_list
)

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
int	**num_of_criteria**	Input	The number of criteria that have to be considered
char * *	**criteria**	Input	The criteria to be considered
int *	**num_of_classified_sets**	Output	The count of criteria sets possible
UF_FBM_GEOM_classified_crit_t *	**classified_set_list**	Output to UF_*free*	The object containing the results of the classification. This has to be freed calling the function UF_FBM_GEOM_free_classified_set_list

UF_FBM_GEOM_create (view source)

Defined in: uf_fbm_geom.h

Overview
Create FBM_GEOM groups from the features in the part geometry of the
parent group. Only those features in the part geometry that are also in
the selected list of the Machining Feature Manager are used to create groups.
Refer to uf_mfm.h for details of selected features in the Machining Feature Manager.

Environment
Internal and External

History
Released in NX4

int UF_FBM_GEOM_create
(
char * type,
char * subtype,
tag_t parent_geom,
tag_t * new_object
)

char *	**type**	Input	The template type

char *	**subtype**	Input	The template sub-type
tag_t	**parent_geom**	Input	The parent geometry group of new feature groups
tag_t *	**new_object**	Output	The first new feature group object

---

UF_FBM_GEOM_free_classified_set_list (view source)

Defined in: uf_fbm_geom.h

Overview
Free the classification object.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_free_classified_set_list
(
    tag_t fbm_geom_tag,
    UF_FBM_GEOM_classified_crit_t classified_set_list
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The object that has to be freed

---

UF_FBM_GEOM_remove_accessibility_vectors (view source)

Defined in: uf_fbm_geom.h

Overview
Remove all the accessibility vectors associated to the feature of FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_remove_accessibility_vectors
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t representative_feature
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest

UF_NCFEAT_t	**representative_feature**	Input	A representative feature object of the group

---

UF_FBM_GEOM_remove_feature (view source)

Defined in: uf_fbm_geom.h

Overview
Remove the feature object from the FBM_GEOM group. If the feature object is not in the FBM_GEOM group, then an error is returned

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_remove_feature
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t feature
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**feature**	Input	The feature object to be removed from the fbm_geom group

---

UF_FBM_GEOM_set_accessibility_vectors (view source)

Defined in: uf_fbm_geom.h

Overview
Set the list of accessibity vectors associated to the feature of FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_set_accessibility_vectors
(
    tag_t fbm_geom_tag,
    UF_NCFEAT_t representative_feature,
    int count,
    tag_t * smart_vectors
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_NCFEAT_t	**representative_feature**	Input	A representative feature object of the group

int	**count**	Input	The number of accessibility vectors
tag_t *	**smart_vectors**	Input	count The array of accessibility vectors

UF_FBM_GEOM_set_classified_features (view source)

Defined in: uf_fbm_geom.h

Overview
Set the classified set of feature objects in FBM_GEOM group.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_set_classified_features
(
    tag_t fbm_geom_tag,
    UF_FBM_GEOM_classified_crit_t classified_set_list,
    int classified_set_index
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
UF_FBM_GEOM_classified_crit_t	**classified_set_list**	Input	The result of the classification
int	**classified_set_index**	Input	The index of the set which should be used

UF_FBM_GEOM_set_feature_name (view source)

Defined in: uf_fbm_geom.h

Overview
Set the name of the feature in the FBM_GEOM group and create the feature objects from the name.

Environment
Internal and External

History
Released in V19.0

```
int UF_FBM_GEOM_set_feature_name
(
    tag_t fbm_geom_tag,
    char * feature_name
)
```

tag_t	**fbm_geom_tag**	Input	The tag of the fbm_geom group of interest
char *	**feature_name**	Input	One of the names that is returned by the call to UF_FBM_GEOM_ask_list_of_feature_names

---

UF_HMOP_ask_hole_axis (view source)

Defined in: uf_hmop.h

Overview
Return Hole Axis of the representative feature object.
The representative feature is obtained from the geometry
parent of the operation(which should be of the type FBM_geom).

Environment
Internal and External

History
Released in V19.0

```
int UF_HMOP_ask_hole_axis
(
    tag_t hmop_tag,
    UF_NCFEAT_t rep_feature,
    tag_t * smart_vector
)
```

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_vector**	Output	Hole Axis of rep_feature.

UF_HMOP_ask_hole_depth (view source)

Defined in: uf_hmop.h

Overview
Return Hole Depth of of the representative feature object.
The representative feature must be obtained from the geometry
parent of the operation(which should be of the type FBM_geom).

Environment
Internal and External

History
Released in V19.0

```
int UF_HMOP_ask_hole_depth
(
    tag_t hmop_tag,
    UF_NCFEAT_t rep_feature,
    tag_t * smart_point
)
```

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_point**	Output	Hole Depth of rep_feature.

UF_HMOP_ask_hole_top (view source)

Defined in: uf_hmop.h

Overview

Return Hole Top of the representative feature object.
The representative feature is obtained from the geometry
parent of the operation(which should be of the type FBM_geom).

Environment

Internal and External

History

Released in V19.0

```
int UF_HMOP_ask_hole_top
(
    tag_t hmop_tag,
    UF_NCFEAT_t rep_feature,
    tag_t * smart_point
)
```

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_point**	Output	Hole Top of rep_feature.

UF_HMOP_set_hole_axis (view source)

Defined in: uf_hmop.h

Overview

Set Hole Axis to the representative feature object. The Hole Axis tag
must be associated to an entity of the representative feature object.
Otherwise, it will return an error.
The representative feature must be obtained from the geometry
parent of the operation(which should be of the type FBM_geom).

Environment

Internal and External

History

Released in V19.0

```
int UF_HMOP_set_hole_axis
(
    tag_t hmop_tag,
    UF_NCFEAT_t rep_feature,
    tag_t * smart_vector
)
```

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_vector**	Input	Smart vector associated to one of the entities of rep_feature.

UF_HMOP_set_hole_depth (view source)

Defined in: uf_hmop.h

Overview

Set Hole Depth to the representative feature object. The Hole Depth tag must be associated to an entity of the representative feature object. Otherwise, it will return an error.
The representative feature must be obtained from the geometry parent of the operation(which should be of the type FBM_geom).

Environment

Internal and External

History

Released in V19.0

```
int UF_HMOP_set_hole_depth
(
    tag_t hmop_tag,
    UF_NCFEAT_t rep_feature,
    tag_t * smart_point
)
```

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_point**	Input	Smart point associated to one of the entities of rep_feature.

UF_HMOP_set_hole_top (view source)

Defined in: uf_hmop.h

Overview

Set Hole Top to the representative feature object. The Hole Top tag must be associated to an entity of the representative feature object.
Otherwise, it will return an error.

The representative feature is obtained from the geometry
parent of the operation (which should be of the type FBM_geom).

Environment

Internal and External

History

Released in V19.0

int UF_HMOP_set_hole_top
(
tag_t hmop_tag,
UF_NCFEAT_t rep_feature,
tag_t * smart_point
)

tag_t	**hmop_tag**	Input	The tag of the hole making operation of interest
UF_NCFEAT_t	**rep_feature**	Input	A representative feature.
tag_t *	**smart_point**	Input	Smart point associated to one of the entities of rep_feature.

UF_MCT_replace_machine (view source)

Defined in: uf_lib_cam.h

Overview

This function replaces the current Machine Tool with a specified machine
from the Machine Library (as defined in cam_config.dat) and creates an
NX Machine Tool Object based upon the values received from the library.

Environment

Internal and External

History

Released in NX3.0

int UF_MCT_replace_machine
(
char * libref,
int retrieve_opt,
UF_NCMCT_setup_replace_mode_t mounting_opt,
tag_t * ncmct_tag
)

char *	**libref**	Input	- the library reference of the desired machine tool. Can be gotten from a record set or an existing NX object
int	**retrieve_opt**	Input	- Flag indicating whether tool pocket data is to be retrieved with new machine: 0 = No 1 = Yes
UF_NCMCT_setup_replace_mode_t	**mounting_opt**	Input	- Flag indicating how the machine is to be located with respect to the part if there is a part file associated with the machine: 0 = No mounting. Machine is loaded as

			defined. 1 = Mount the machine onto the workpiece. 2 = Mount the workpiece onto the machine.
tag_t *	**ncmct_tag**	Output	- the NX object created as a result of the retrieval

---

UF_MCT_retrieve (view source)

Defined in: uf_lib_cam.h

Overview
This function retrieves a Machine Tool from the current Machine Tool Library (as defined in cam_config.dat) and creates an NX Machine Tool Object based upon the values received from the library.

Environment
Internal and External

History
Released in V16.0

```
int UF_MCT_retrieve
(
    const char * libref,
    tag_t * ncmct_tag
)
```

const char *	**libref**	Input	- the library reference of the desired machine tool. Can be gotten from a record set or an existing NX object
tag_t *	**ncmct_tag**	Output	- the NX object created as a result of the retrieval

---

UF_MFM_ask_attribute_type (view source)

Defined in: uf_mfm.h

Overview
Return the type of the attribute.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_attribute_type
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    UF_MFM_attr_value_type_p_t type
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
UF_MFM_attr_value_type_p_t	**type**	Output	The attribute type

---

UF_MFM_ask_attributes (view source)

Defined in: uf_mfm.h

Overview
Return a list attribute names of the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_attributes
(
    UF_NCFEAT_t machining_feature,
    int * count,
    char * * * attribute_names
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
int *	**count**	Output	The number of attributes
char * * *	**attribute_names**	Output to UF_*free*	The array of attribute names. The memory is allocated by this function and must be freed by calling UF_free to the array and for each name string

---

UF_MFM_ask_candidate_machining_feature_types (view source)

Defined in: uf_mfm.h

Overview
Return a list of types of machining feature candidates in the bodies.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_candidate_machining_feature_types
(
    int body_count,
    tag_t * body_list,
    int * type_count,
    char * * * candidate_type_names
)
```

int	**body_count**	Input	The number of bodies
tag_t *	**body_list**	Input	The list of bodies
int *	**type_count**	Output	The number of candidate types
char * * *	**candidate_type_names**	Output to UF_*free*	The array of candidate feature type names. The memory is allocated by this function and must be freed by calling UF_free to the array and for each name string

---

UF_MFM_ask_double_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview

Return the original and overridden values of the double attribute.
Two values are equal if the attribute is not overridden.

Environment

Internal and External

History

Released in NX3.0

```
    int UF_MFM_ask_double_value_of_attribute
    (
        UF_NCFEAT_t machining_feature,
        char * attribute,
        double * original_value,
        double * overridden_value
    )
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
double *	**original_value**	Output	The original double value
double *	**overridden_value**	Output	The current double value

---

UF_MFM_ask_feature_name (view source)

Defined in: uf_mfm.h

Overview

Return the name of the machining feature.

Environment

Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_feature_name
(
    UF_NCFEAT_t machining_feature,
    char * * feature_name
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char * *	**feature_name**	Output to UF_*free*	The Feature name of the machining feature. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_ask_feature_type (view source)

Defined in: uf_mfm.h

Overview
Return feature type of the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_feature_type
(
    UF_NCFEAT_t machining_feature,
    char * * feature_type_name
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char * *	**feature_type_name**	Output to UF_*free*	Feature type name of the machining feature. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_ask_geometry_groups (view source)

Defined in: uf_mfm.h

Overview
Return a list of geometry groups of the machining feature.
These geometry groups contain the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_geometry_groups
(
    UF_NCFEAT_t machining_feature,
    int * count,
    tag_t * * geometry_groups
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
int *	**count**	Output	The number of geometry groups that the machining feature belongs to
tag_t * *	**geometry_groups**	Output to UF_*free*	The array of geometry groups that machining feature belongs to. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_ask_integer_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview
Return the original and overridden values of the integer attribute.
Two values are equal if the attribute is not overridden.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_integer_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    int * original_value,
    int * overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
int *	**original_value**	Output	The original integer value
int *	**overridden_value**	Output	The current integer value

UF_MFM_ask_list_of_faces (view source)

Defined in: uf_mfm.h

Overview
Get list of faces of specified machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_list_of_faces
(
    UF_NCFEAT_t machining_feature,
    int * count,
    tag_t * * face_list
)
```

UF_NCFEAT_t	**machining_feature**	Input	Machining feature
int *	**count**	Output	number of faces
tag_t * *	**face_list**	Output to UF_*free*	face list

UF_MFM_ask_logical_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview
Return the original and overridden values of the logical attribute.
Two values are equal if the attribute is not overridden.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_logical_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    logical * original_value,
    logical * overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
logical *	**original_value**	Output	The original logical value
logical *	**overridden_value**	Output	The current logical value

---

UF_MFM_ask_machined_status (view source)

Defined in: uf_mfm.h

Overview
Return the status of the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_machined_status
(
    UF_NCFEAT_t machining_feature,
    tag_t geometry_group,
    UF_MFM_machined_status_t * status
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
tag_t	**geometry_group**	Input	The gometry group the machining feature
UF_MFM_machined_status_t *	**status**	Output	The machined status

---

UF_MFM_ask_machining_feature_types (view source)

Defined in: uf_mfm.h

Overview
Return a list of the types of machining features that exist in the part.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_machining_feature_types
(
    tag_t part_tag,
    int * count,
    char * * * feature_type_names
)
```

tag_t	**part_tag**	Input	The part tag
int *	**count**	Output	The number of machining feature types
char * * *	**feature_type_names**	Output to UF_*free*	The array of machining feature names. The memory is allocated by this function and must be freed by calling UF_free to the array and for each name string

UF_MFM_ask_machining_features_of_part (view source)

Defined in: uf_mfm.h

Overview
Return a list of machining features that exist in the part.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_machining_features_of_part
(
    tag_t part_tag,
    int * count,
    UF_NCFEAT_t * * machining_features
)
```

tag_t	**part_tag**	Input	The part tag
int *	**count**	Output	The number of machining features in the part
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining features in the part. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_ask_machining_features_of_type (view source)

Defined in: uf_mfm.h

Overview
Return a list of machining features of the type in the part.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_machining_features_of_type
(
    tag_t part_tag,
    char * type_name,
    int * count,
    UF_NCFEAT_t * * machining_features
)
```

tag_t	**part_tag**	Input	The part tag
char *	**type_name**	Input	The feature type name

int *	**count**	Output	The number of machining features
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining features. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_ask_overridden_status (view source)

Defined in: uf_mfm.h

Overview
Return the overridden status of the machining feature.
The status is TRUE is any attribute or the name is set by the user.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_overridden_status
(
    UF_NCFEAT_t machining_feature,
    logical * overridden_status
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
logical *	**overridden_status**	Output	The overridden status

UF_MFM_ask_selected_fea_list (view source)

Defined in: uf_mfm.h

Overview
Ask the selected feature list.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_selected_fea_list
(
    UF_NCFEAT_t * * machining_features,
    int * count
)
```

UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	selected features

int *	**count**	Output	: number of selected features

UF_MFM_ask_source_type (view source)

Defined in: uf_mfm.h

Overview
Return the source type of the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_source_type
(
    UF_NCFEAT_t machining_feature,
    UF_MFM_source_type_t * source
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
UF_MFM_source_type_t *	**source**	Output	The source type

UF_MFM_ask_string_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview
Return the original and overridden values of a string attribute.
Two values are equal if the attribute is not overridden.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_ask_string_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    char * * original_value,
    char * * overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name

char * *	**original_value**	Output to UF_*free*	The original string value. The memory is allocated by this function and must be freed by calling UF_free.
char * *	**overridden_value**	Output to UF_*free*	The overridden string value. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_clean_selected_fea_list (view source)

Defined in: uf_mfm.h

Overview
Clean the memory of selected feature list.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_clean_selected_fea_list
(

)
```

UF_MFM_create_machining_feature (view source)

Defined in: uf_mfm.h

Overview
Create a machining feature of UF_MFM_source_type_recognized_feature type
from a list of faces without setting any feature attributes.
To set the attributes, call UF_MFM_set_double_ug_attribute.

Environment
Internal and External

History
Released in NX4.0

```
int UF_MFM_create_machining_feature
(
    char * feature_type,
    int count,
    tag_t * face_list,
    UF_NCFEAT_t * machining_feature
)
```

char *	**feature_type**	Input	The type of machining feature

int	**count**	Input	The number of faces for a machining feature
tag_t *	**face_list**	Input	The list of faces for a maching feature
UF_NCFEAT_t *	**machining_feature**	Output	A new machining feature

---

UF_MFM_create_machining_features_from_modeling_features (view source)

Defined in: uf_mfm.h

Overview
Create machining features from NX modeling features of the machining feature types in the bodies.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_create_machining_features_from_modeling_features
(
    int body_count,
    tag_t * body_list,
    int type_count,
    char * * feature_types,
    int * count,
    UF_NCFEAT_t * * machining_features
)
```

int	**body_count**	Input	The number of bodies
tag_t *	**body_list**	Input	The list of bodies
int	**type_count**	Input	The number of feature types
char * *	**feature_types**	Input	The list of feature type names
int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_recognized_features (view source)

Defined in: uf_mfm.h

Overview
Create machining features from recognized features in the bodies.

Environment

Internal and External

History

Released in NX3.0

int UF_MFM_create_machining_features_from_recognized_features
(
int body_count,
tag_t * body_list,
int * count,
UF_NCFEAT_t * * machining_features
)

int	**body_count**	Input	The number of bodies
tag_t *	**body_list**	Input	The list of bodies
int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_tagged_arcs (view source)

Defined in: uf_mfm.h

Overview

Create machining features from tagged arcs in the current part.

Environment

Internal and External

History

Released in NX3.0

int UF_MFM_create_machining_features_from_tagged_arcs
(
int * count,
UF_NCFEAT_t * * machining_features
)

int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_tagged_edges (view source)

Defined in: uf_mfm.h

Overview

Create machining features from tagged edges in the bodies.

Environment
Internal and External

History
Released in NX3.0

**int UF_MFM_create_machining_features_from_tagged_edges
(
    int body_count,
    tag_t * body_list,
    int * count,
    UF_NCFEAT_t * * machining_features
)**

int	**body_count**	Input	The number of bodies
tag_t *	**body_list**	Input	The list of bodies
int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_tagged_faces (view source)

Defined in: uf_mfm.h

Overview
Create machining features from tagged faces in the bodies.

Environment
Internal and External

History
Released in NX3.0

**int UF_MFM_create_machining_features_from_tagged_faces
(
    int body_count,
    tag_t * body_list,
    int * count,
    UF_NCFEAT_t * * machining_features
)**

int	**body_count**	Input	The number of bodies
tag_t *	**body_list**	Input	The list of bodies
int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_tagged_points (view source)

Defined in: uf_mfm.h

Overview
Create machining features from tagged points the current part.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_create_machining_features_from_tagged_points
(
    int * count_of_machining_features,
    UF_NCFEAT_t * * machining_features
)
```

int *	**count_of_machining_features**	Output	The number of machining features
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining features. The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_create_machining_features_from_user_defined_features (view source)

Defined in: uf_mfm.h

Overview
Create machining features from NX user defined features (UDF) of the machining feature types in the bodies.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_create_machining_features_from_user_defined_features
(
    int body_count,
    tag_t * body_list,
    int type_count,
    char * * feature_types,
    int * count,
    UF_NCFEAT_t * * machining_features
)
```

int	**body_count**	Input	The number of bodies

tag_t *	**body_list**	Input	The list of bodies
int	**type_count**	Input	The number of feature types
char * *	**feature_types**	Input	The list of feature type names
int *	**count**	Output	The of machining feature
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	The array of machining fetures The memory is allocated by this function and must be freed by calling UF_free.

UF_MFM_delete_machining_features (view source)

Defined in: uf_mfm.h

Overview
Delete machining features.
If any item in the list is not a machining feature, an error is returned and
the machining features will not be deleted.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_delete_machining_features
(
    int count,
    UF_NCFEAT_t * machining_features
)
```

int	**count**	Input	The number of machining features to be deleted
UF_NCFEAT_t *	**machining_features**	Input	The array of machining features to be deleted

UF_MFM_has_selected_fea_list (view source)

Defined in: uf_mfm.h

Overview
Ask whether there are selected features in the list.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_has_selected_fea_list
(
        logical * result
    )
```

logical *	**result**	Output	: whether there is selected feature list

---

UF_MFM_recognize_holes (view source)

Defined in: uf_mfm.h

Overview

Recognize hole features of the types and from the solid bodies provided in the input argument.

Environment

Internal and External

History

Released in NX4.0

```
int UF_MFM_recognize_holes
(
    tag_t * body_list,
    int body_count,
    char * * type_list,
    int type_count,
    UF_MFM_recognize_options_p_t options,
    int * feature_count,
    UF_NCFEAT_t * * machining_features
)
```

tag_t *	**body_list**	Input	list of solid bodies
int	**body_count**	Input	number of solid bodies
char * *	**type_list**	Input	list of hole types to be recognized NULL will recognize all types in feature definition file
int	**type_count**	Input	number of holes types to be recognized Zero will recognize all types in feature definition file
UF_MFM_recognize_options_p_t	**options**	Input	Options used during recognition
int *	**feature_count**	Output	number of recognized machining holes
UF_NCFEAT_t * *	**machining_features**	Output to UF_*free*	list of recognized machining features

---

UF_MFM_set_double_ug_attribute (view source)

Defined in: uf_mfm.h

Overview

Set NX double attribute in the specified machining feature, if the attribute already exists, then reset the attribute value; otherwise add attribute.

Environment

Internal and External

History

Released in NX3.0

```
int UF_MFM_set_double_ug_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    double value
)
```

UF_NCFEAT_t	**machining_feature**	Input	Machining feature
char *	**attribute**	Input	attribute name
double	**value**	Input	attribute value

UF_MFM_set_double_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview

Set the value of the double attribute.

Environment

Internal and External

History

Released in NX3.0

```
int UF_MFM_set_double_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    double overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
double	**overridden_value**	Input	The overridden value

UF_MFM_set_feature_name (view source)

Defined in: uf_mfm.h

Overview
Set the name of the machining feature.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_set_feature_name
(
    UF_NCFEAT_t machining_feature,
    char * feature_name
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**feature_name**	Input	New feature name

UF_MFM_set_int_ug_attribute (view source)

Defined in: uf_mfm.h

Overview
Set NX integer attribute in the specified machining feature, if the attribute already exists, then reset the attribute value; otherwise add attribute.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_set_int_ug_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    int value
)
```

UF_NCFEAT_t	**machining_feature**	Input	Machining feature
char *	**attribute**	Input	attribute name
int	**value**	Input	attribute value

UF_MFM_set_integer_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview

Set the value of the integer attribute.

Environment

Internal and External

History

Released in NX3.0

```
int UF_MFM_set_integer_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    int overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
int	**overridden_value**	Input	The overridden value

UF_MFM_set_logical_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview

Set the value of the logical attribute.

Environment

Internal and External

History

Released in NX3.0

```
int UF_MFM_set_logical_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    logical overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
logical	**overridden_value**	Input	The overridden value

UF_MFM_set_selected_fea_list (view source)

Defined in: uf_mfm.h

Overview
Set the selected feature list.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_set_selected_fea_list
(
    UF_NCFEAT_t * machining_features,
    int count
)
```

UF_NCFEAT_t *	**machining_features**	Input	selected features
int	**count**	Input	number of selected features

UF_MFM_set_string_ug_attribute (view source)

Defined in: uf_mfm.h

Overview
Set NX string attribute in the specified machining feature, if the attribute already exists, then reset the attribute value; otherwise add attribute.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_set_string_ug_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    char * value
)
```

UF_NCFEAT_t	**machining_feature**	Input	Machining feature
char *	**attribute**	Input	attribute name
char *	**value**	Input	attribute value

UF_MFM_set_string_value_of_attribute (view source)

Defined in: uf_mfm.h

Overview
Set the value of the string attribute.

Environment
Internal and External

History
Released in NX3.0

```
int UF_MFM_set_string_value_of_attribute
(
    UF_NCFEAT_t machining_feature,
    char * attribute,
    char * overridden_value
)
```

UF_NCFEAT_t	**machining_feature**	Input	The pointer of the machining feature
char *	**attribute**	Input	The attribute name
char *	**overridden_value**	Input	The overridden value

UF_OPER_ask_name (view source)

Defined in: uf_oper_spec.h

Overview
This function copies the Operation name associated with "oper_id" to the memory
pointed at "op_name". This memory must be the same as UF_OPER_OPNAME_BUFSIZE bytes.

Environment
Internal and External

```
int UF_OPER_ask_name
(
    UF_OPER_id_t oper_id,
    char op_name [ UF_OPER_OPNAME_BUFSIZE ]
)
```

UF_OPER_id_t	**oper_id**	Input	Operation identifier
char	**op_name [UF_OPER_OPNAME_BUFSIZE]**	Output	Pointer to at least UF_OPER_OPNAME_BUFSIZE bytes

---

UF_OPER_ask_oper (view source)

Defined in: uf_oper_spec.h

Overview
Returns the UF_oper_id_t that corresponds to the UF_CAM_exit_id_t passed to the
User Exit from NX.

Environment
Internal and External

```
int UF_OPER_ask_oper
(
    UF_CAM_exit_id_t exit_id,
    UF_OPER_id_t * oper_id
)
```

UF_CAM_exit_id_t	**exit_id**	Input	exit_id passed to user exit from NX
UF_OPER_id_t *	**oper_id**	Output	Operation Identifier

UF_OPER_ask_path (view source)

Defined in: uf_oper_spec.h

Overview
Returns the path identifier associated with "oper_id".

Environment
Internal and External

```
int UF_OPER_ask_path
(
    UF_OPER_id_t oper_id,
    UF_PATH_id_t * path_id
)
```

UF_OPER_id_t	**oper_id**	Input	Operation identifier
UF_PATH_id_t *	**path_id**	Output	Path identifier of this oper's path

UF_OPRBND_append_item_ude (view source)

Defined in: uf_oprbnd.h

Overview
Append the user defined events of the inherited member.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_append_item_ude
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_UDE_set_type_t set_type,
    char * ude_name,
    UF_OPRBND_UDE_t * ude,
    logical * response
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_UDE_set_type_t	**set_type**	Input	the user defined event (ude) types, either Start or End
char *	**ude_name**	Input	the ude name
UF_OPRBND_UDE_t *	**ude**	Output	the ude object
logical *	**response**	Output	the response. Success = TRUE, fail = FALSE

UF_OPRBND_ask_boundary_app_data (view source)

Defined in: uf_oprbnd.h

Overview

Gets the inherited application data of the boundary.

The memory for app_data must be allocated by the user.

Environment

Internal and External

History

Released in V19.0

```
int UF_OPRBND_ask_boundary_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_app_data_t * app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_app_data_t *	**app_data**	Output	the application data

UF_OPRBND_ask_item_app_data (view source)

Defined in: uf_oprbnd.h

Overview

Gets the application data of the inherited member.

The memory for app_data must be allocated by the user.

Environment

Internal and External

History

Released in V19.0

```
int UF_OPRBND_ask_item_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_app_data_t * app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_app_data_t *	**app_data**	Output	the application data of the boundary member

UF_OPRBND_ask_item_udes (view source)

Defined in: uf_oprbnd.h

Overview

Gets the user defined events of the inherited member.

Environment

Internal and External

History

Released in V19.0

```
int UF_OPRBND_ask_item_udes
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_UDE_set_type_t set_type,
    int * num_udes,
    UF_OPRBND_UDE_t * * udes
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_UDE_set_type_t	**set_type**	Input	the user defined event (ude) types, either Start or End
int *	**num_udes**	Output	the number of user defined events
UF_OPRBND_UDE_t * *	**udes**	Output	the ude object

UF_OPRBND_can_accept_item_ude (view source)

Defined in: uf_oprbnd.h

Overview
Check if the user defined event can be accepted.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_can_accept_item_ude
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_UDE_set_type_t set_type,
    char * ude_name,
    logical * response
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_UDE_set_type_t	**set_type**	Input	the user defined event (ude) types, either Start or End
char *	**ude_name**	Input	the ude name

logical *	**response**	Output	the response. Can be accpeted = TRUE, can not be accpeted = FALSE

UF_OPRBND_delete_all_item_udes (view source)

Defined in: uf_oprbnd.h

Overview
Deletes all of the user defined events of the inherited member.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_delete_all_item_udes
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_UDE_set_type_t set_type
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_UDE_set_type_t	**set_type**	Input	the user defined event (ude) types, either Start or End

---

UF_OPRBND_delete_item_ude (view source)

Defined in: uf_oprbnd.h

Overview
Delete the user defined event of the inherited member.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_delete_item_ude
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_UDE_set_type_t set_type,
    UF_OPRBND_UDE_t ude
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_UDE_set_type_t	**set_type**	Input	the user defined event (ude) types, either Start or End
UF_OPRBND_UDE_t	**ude**	Input	the ude object

UF_OPRBND_set_boundary_app_data (view source)

Defined in: uf_oprbnd.h

Overview
Sets the inherited boundary application data.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_set_boundary_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_app_data_p_t app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_app_data_p_t	**app_data**	Input	the application data

---

UF_OPRBND_set_item_app_data (view source)

Defined in: uf_oprbnd.h

Overview
Sets the application data of the inherited member.

Environment
Internal and External

History
Released in V19.0

```
int UF_OPRBND_set_item_app_data
(
    tag_t object_tag,
    UF_CAM_geom_type_t type,
    UF_OPRBND_boundary_t boundary,
    UF_OPRBND_item_t item,
    UF_OPRBND_app_data_p_t app_data
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
UF_CAM_geom_type_t	**type**	Input	the type of the boundary
UF_OPRBND_boundary_t	**boundary**	Input	the boundary
UF_OPRBND_item_t	**item**	Input	the boundary member
UF_OPRBND_app_data_p_t	**app_data**	Input	the application data of the boundary member

---

UF_TURN_ask_cut_region_of_index (view source)

Defined in: uf_turn.h

Overview
In some cases there may be multiple non-connected cut regions
representing the remaining material that can be cut by the currently
active turning tool with respect to the operation referenced and part
and blank geometry specified.
Function 'UF_TURN_ask_cut_region_of_index' finds out, whether a cut
region with positive integer index 'index_of_cut_region' exists, and if
so, returns its cross-sectional area and a Cut Region Selection Point
(in coordinates described relative to ACS) that uniquely identifies
the indexed cut region.

The user must configure tool, geometry data and operation parameters
prior to calling this function.
It will return the status value 0 if cut regions could be detected,
and an error code, if no cut regions have been found.

Return
Return code :
= 0 : successful - admissible cut regions could be found
> 0 : failing error number
< 0 : failing error number

Environment
Internal and External

History
created in V19.0.

```
int UF_TURN_ask_cut_region_of_index
(
    tag_t oper_tag,
    int index_of_cut_region,
    double * area_of_cut_region,
    double selection_point_for_cut_region [ 3 ] ,
    logical * cut_region_exists,
    char * * message
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which cut regions are to be found
int	**index_of_cut_region**	Input	Index of Cut Region to query
double *	**area_of_cut_region**	Output	Cross-sectional area of Cut Region with index (if existing, 0 else)
double	**selection_point_for_cut_region [3]**	Output	Returns coordinates of cut region selection point relative to ACS uniquely identifying the indexed Cut Region
logical *	**cut_region_exists**	Output	True, if cut region having 'index_of_cut_region' exists
char * *	**message**	Output	Informational message both for successful detection as for error case

UF_TURN_ask_cut_regions_exist (view source)

Defined in: uf_turn.h

Overview
This routine performs a scan for cut regions representing the remaining material that can be cut by the currently active turning tool with respect to the operation referenced and part and blank geometry specified. The user must configure tool, geometry data and operation parameters prior to calling this function.
It will return the status value 0 if cut regions could be detected, and an error code, if no cut regions have been found.

Return
Return code :
= 0 : successful - admissible cut regions could be detected
> 0 : failing error number
< 0 : failing error number

Environment
Internal and External

History
created in V19.0

```
int UF_TURN_ask_cut_regions_exist
(
    tag_t oper_tag,
    int * number_of_cut_regions_found,
    double * total_area_of_cut_regions_found,
    UF_TURN_cut_regions_location_p_t cut_regions_location,
    char * * message
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which cut regions are to be found
int *	**number_of_cut_regions_found**	Output	Number of cut regions found or 0, if none found
double *	**total_area_of_cut_regions_found**	Output	Sums cross-sectional area of cut regions found
UF_TURN_cut_regions_location_p_t	**cut_regions_location**	Output	Reports location of cut regions found relative to centerline
char * *	**message**	Output	Informational message both for successful detection as for error case

UF_TURN_create_blank_from_boundary (view source)

Defined in: uf_turn.h

Overview

This routine creates a turning blank from a single boundary from edges/curves (type UF_PARAM_turn_workpiece_type_curves).
The count of edges/curves, the edge/curve tags with boundary and application data and stock values for radial, face and equidistant stock are input parameters for this function.

Return

Return code :
= 0 : successful - turn blank could be created
> 0 : failing error number
< 0 : failing error number

Environment

Internal and External

History

Released in NX2

```
int UF_TURN_create_blank_from_boundary
(
    tag_t object_tag,
    int count,
    tag_t * curves,
    UF_CAMBND_boundary_data_p_t boundary_data,
    UF_CAMBND_app_data_p_t * app_data,
    double stock_equi,
    double stock_face,
    double stock_radial
)
```

tag_t	**object_tag**	Input	the parent object of the boundary
int	**count**	Input	the count of edges/curves
tag_t *	**curves**	Input	the edge/curve tags from which a boundary will be created
UF_CAMBND_boundary_data_p_t	**boundary_data**	Input	the boundary data
UF_CAMBND_app_data_p_t *	**app_data**	Input	the application data for each member
double	**stock_equi**	Input	the equidistant stock for the blank boundary
double	**stock_face**	Input	the face stock for the blank boundary
double	**stock_radial**	Input	the radial stock for the blank boundary

---

UF_TURN_create_parametric_blank (view source)

Defined in: uf_turn.h

Overview

This routine creates a turning blank of type cylinder or tube
(UF_PARAM_turn_workpiece_type_cylinder or UF_PARAM_turn_workpiece_type_tube).
Direction of the blank (UF_PARAM_turn_workpiece_direction_towards_head_stock
or UF_PARAM_turn_workpiece_direction_from_head_stock), mounting point for
positioning, length, outer and additional inner diameter for tube must be
specified as input parameters to this function.

Return

Return code :
= 0 : successful - turn blank could be created
> 0 : failing error number
< 0 : failing error number

Environment

Internal and External

History

Released in NX2

```
int UF_TURN_create_parametric_blank
(
    tag_t object_tag,
    UF_PARAM_turn_workpiece_type_t workpiece_type,
    UF_PARAM_turn_workpiece_direction_t direction,
    tag_t mounting_point,
    double length,
    double outer_diameter,
    double inner_diameter
)
```

tag_t	**object_tag**	Input	the parent object of the blank
UF_PARAM_turn_workpiece_type_t	**workpiece_type**	Input	the type of the blank

UF_PARAM_turn_workpiece_direction_t	**direction**	Input	the direction of the parametric blank
tag_t	**mounting_point**	Input	the mounting point of the parametric blank
double	**length**	Input	the length of the parametric blank
double	**outer_diameter**	Input	the diameter of the parametric blank
double	**inner_diameter**	Input	the inner diameter of the tube blank

---

UF_TURN_ipw_box (view source)

Defined in: uf_turn.h

Overview

This routine computes a box around the curve resulting from the cross-sectioned in-process workpiece.
It provides:
- origin, length and diameter of the workpiece
- two points defining lower left and upper right corner of the box described in 2D "workplane coordinates" relative to the spindle coordinate system as well as
- the four corner points of the box in 3D coordinates relative to ACS.

Return

Return code :
= 0 : successful - min max workpiece box could be created
> 0 : failing error number
< 0 : failing error number

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_ipw_box
(
    tag_t oper_tag,
    double * length,
    double * diameter,
    double bottom_left_in_plane [ 2 ] ,
    double top_right_in_plane [ 2 ] ,
    double bottom_left_pnt3 [ 3 ] ,
    double bottom_right_pnt3 [ 3 ] ,
    double top_left_pnt3 [ 3 ] ,
    double top_right_pnt3 [ 3 ] ,
    char * * message
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which min max box of IPW has to be created
double *	**length**	Output	length of the workpiece
double *	**diameter**	Output	diameter of the workpiece
double	**bottom_left_in_plane [2]**	Output	bottom left position of workpiece in plane

double	**top_right_in_plane [2]**	Output	top right position of workpiece in plane
double	**bottom_left_pnt3 [3]**	Output	bottom left position of workpiece
double	**bottom_right_pnt3 [3]**	Output	bottom right position of workpiece
double	**top_left_pnt3 [3]**	Output	top left position of workpiece
double	**top_right_pnt3 [3]**	Output	top right position of workpiece
char * *	**message**	Output	Informational message both for successful detection as for error case

---

UF_TURN_map_angle_from_wcs (view source)

Defined in: uf_turn.h

Overview
This function converts an angle relative to WCS coordinates into an angle relative to the spindle orientated coordinate system.

Return
angle relative to the spindle coordinate system

Environment
Internal and External

History
created in V19.0

```
double UF_TURN_map_angle_from_wcs
(
    tag_t oper_tag,
    double wcs_angle
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which angle should be mapped
double	**wcs_angle**	Input	angle relative to WCS coordinates

UF_TURN_map_angle_to_wcs (view source)

Defined in: uf_turn.h

Overview
This function converts an angle relative to the spindle coordinate system into an angle relative to WCS coordinates.

Return
angle relative to WCS coordinates

Environment

Internal and External

History

created in V19.0

```
double UF_TURN_map_angle_to_wcs
(
    tag_t oper_tag,
    double scs_angle
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which angle should be mapped
double	**scs_angle**	Input	angle relative to spindle coordinate system

---

UF_TURN_map_pnt2_from_wcs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 2D point relative to WCS coordinates
into a point relative to the spindle orientated coordinate
system.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_pnt2_from_wcs
(
    tag_t oper_tag,
    double wcs_pnt2 [ 2 ] ,
    double scs_pnt2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 2D point should be mapped
double	**wcs_pnt2 [2]**	Input	2D point relative to WCS coordinates
double	**scs_pnt2 [2]**	Output	2D point relative to spindle coordinate system

---

UF_TURN_map_pnt2_to_acs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 2D point relative to the spindle coordinate
system into a 3D point relative to ACS coordinates.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_pnt2_to_acs
(
    tag_t oper_tag,
    double scs_pnt2 [ 2 ] ,
    double acs_pnt3 [ 3 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 3D point should be mapped
double	**scs_pnt2 [2]**	Input	2D point relative to spindle coordinate system
double	**acs_pnt3 [3]**	Output	3D point relative to ACS coordinates

---

UF_TURN_map_pnt2_to_wcs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 2D point relative to the spindle coordinate
system into a point relative to WCS coordinates.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_pnt2_to_wcs
(
    tag_t oper_tag,
    double scs_pnt2 [ 2 ] ,
    double wcs_pnt2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 2D point should be mapped
double	**scs_pnt2 [2]**	Input	2D point relative to spindle coordinate system
double	**wcs_pnt2 [2]**	Output	2D point relative to ACS coordinates

---

UF_TURN_map_pnt3_from_acs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 3D point relative to ACS coordinates
into a 2D point relative to the spindle orientated coordinate system.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_pnt3_from_acs
(
    tag_t oper_tag,
    double acs_pnt3 [ 3 ] ,
    double scs_pnt2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 3D point should be mapped
double	**acs_pnt3 [3]**	Input	3D point relative to ACS coordinates
double	**scs_pnt2 [2]**	Output	2D point relative to spindle coordinate system

---

UF_TURN_map_tooltrackingpoint_from_wcs (view source)

Defined in: uf_turn.h

Overview

This function converts a tool tracking point relative to WCS coordinates into a tool tracking point relative to the spindle orientated coordinate system.

Return

tool tracking point relative to the spindle coordinate system

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_tooltrackingpoint_from_wcs
(
    tag_t oper_tag,
    int wcs_tooltrackingpoint
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which the tool tracking point should be mapped
int	**wcs_tooltrackingpoint**	Input	tool tracking point relative to WCS coordinates

---

UF_TURN_map_tooltrackingpoint_to_wcs (view source)

Defined in: uf_turn.h

Overview

This function converts a tool tracking point relative to the spindle coordinate system into a tool tracking point relative to WCS coordinates.

Return

tool tracking point relative to WCS coordinates

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_tooltrackingpoint_to_wcs
(
    tag_t oper_tag,
    int scs_tooltrackingpoint
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which the tool tracking point should be mapped
int	**scs_tooltrackingpoint**	Input	tool tracking point relative to spindle coordinate system

---

UF_TURN_map_vec2_from_wcs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 2D vector relative to WCS coordinates into a 2D vector relative to the spindle orientated coordinate system.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_vec2_from_wcs
(
    tag_t oper_tag,
    double wcs_vec2 [ 2 ] ,
    double scs_vec2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 2D vector should be mapped
double	**wcs_vec2 [2]**	Input	2D vector relative to WCS coordinates
double	**scs_vec2 [2]**	Output	2D vector relative to spindle coordinate system

UF_TURN_map_vec2_to_acs (view source)

Defined in: uf_turn.h

Overview
This routine converts a 2D vector relative to the spindle coordinate system into a 3D vector relative to ACS coordinates.

Environment
Internal and External

History
created in V19.0

```
int UF_TURN_map_vec2_to_acs
(
    tag_t oper_tag,
    double scs_vec2 [ 2 ] ,
    double acs_vec3 [ 3 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 3D vector should be mapped
double	**scs_vec2 [2]**	Input	2D vector relative to spindle coordinate system
double	**acs_vec3 [3]**	Output	3D vector relative to ACS coordinates

UF_TURN_map_vec2_to_wcs (view source)

Defined in: uf_turn.h

Overview
This routine converts a 2D vector relative to the spindle coordinate system into a 2D vector relative to WCS coordinates.

Environment
Internal and External

History
created in V19.0

```
int UF_TURN_map_vec2_to_wcs
(
    tag_t oper_tag,
    double scs_vec2 [ 2 ] ,
    double wcs_vec2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 2D vector should be mapped
double	**scs_vec2 [2]**	Input	2D vector relative to spindle coordinate system
double	**wcs_vec2 [2]**	Output	2D vector relative to WCS coordinates

UF_TURN_map_vec3_from_acs (view source)

Defined in: uf_turn.h

Overview

This routine converts a 3D vector relative to ACS coordinates
into a 2D vector relative to the spindle coordinate system.

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_map_vec3_from_acs
(
    tag_t oper_tag,
    double acs_vec3 [ 3 ] ,
    double scs_vec2 [ 2 ]
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which 3D vector should be mapped
double	**acs_vec3 [3]**	Input	3D vector relative to ACS coordinates
double	**scs_vec2 [2]**	Output	2D vector relative to spindle coordinate system

UF_TURN_save_spinning_ipw_as_part (view source)

Defined in: uf_turn.h

Overview

This routine saves the "spinning shape" of the In-Process Workpiece
(IPW) for the operation referenced. The IPW of an operation represents
the status of the workpiece immediately after the operation has been
cut. The IPW can be thought of assuming its "spinning shape"
whenever the turning machine tool's spindle that holds the workpiece
is rotating.
Function 'UF_TURN_save_spinning_ipw_as_part' will return a zero if
the function is successful, otherwise it returns an error number.

Return

Return code :
= 0 : successful
> 0 : failing error number
< 0 : failing error number

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_save_spinning_ipw_as_part
(
    tag_t oper_tag,
    char * filename,
    char * * message
)
```

tag_t	**oper_tag**	Input	Tag of the operation for which cut regions are to be detected
char *	**filename**	Input	Name of partfile to save IPW to
char * *	**message**	Output	Informational message both for case where IPW was successfully saved to disk as for error case

---

UF_TURN_teachmode_create_subop (view source)

Defined in: uf_turn.h

Overview

This routine creates a teachmode suboperation letting you define the type of suboperation you want to create and finally adds the newly created teachmode suboperation to the list of suboperations contained in the given teachmode operation.

Return

Return code :
= 0 : successful - suboperation could be created
> 0 : failing error number
< 0 : failing error number

Environment

Internal and External

History

created in V19.0

```
int UF_TURN_teachmode_create_subop
(
    tag_t oper_tag,
    UF_PARAM_ttmopr_subop_type_t subop_type,
    tag_t * subop_tag,
    char * * message
)
```

tag_t	**oper_tag**	Input	Tag of the teachmode operation for which suboperations should be added
UF_PARAM_ttmopr_subop_type_t	**subop_type**	Input	Type of the suboperation
tag_t *	**subop_tag**	Output	Tag of the created suboperation
char * *	**message**	Output	Informational message both for successful detection as for error case

---

UF_UDE_ask_boolean (view source)

Defined in: uf_ude.h

Overview

This function returns in 'value' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object specified by 'ude_obj'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_boolean
(
    UF_UDE_t ude_obj,
    char * param_name,
    logical * value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
logical *	**value**	Output	see above

UF_UDE_ask_double (view source)

Defined in: uf_ude.h

Overview

This function returns in 'value' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object specified by 'ude_obj'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_double
(
    UF_UDE_t ude_obj,
    char * param_name,
    double * value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above

double *	**value**	Output	see above

UF_UDE_ask_integer (view source)

Defined in: uf_ude.h

Overview
This function returns in 'value' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object specified by 'ude_obj'.

Environment
Internal and External

History
Originally released in V18.0

```
int UF_UDE_ask_integer
(
    UF_UDE_t ude_obj,
    char * param_name,
    int * value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
int *	**value**	Output	see above

UF_UDE_ask_name (view source)

Defined in: uf_ude.h

Overview
This function returns the name of the User Defined Machine Control Event object 'ude_obj' in 'ude_name'.

Environment
Internal and External

History
Originally released in V18.0

```
int UF_UDE_ask_name
(
    UF_UDE_t ude_object,
    char * * ude_name
)
```

UF_UDE_t	**ude_object**	Input	see above

char * *	**ude_name**	Output to UF_*free*	see above NOTE: The memory allocated for ude_name has to be freed by calling UF_free on ude_name

---

UF_UDE_ask_param_toggle (view source)

Defined in: uf_ude.h

Overview
This function returns in 'toggle' the toggle status of the parameter specified
by 'param_name' in the User Defined Machine Control Event object 'ude_obj'.

Environment
Internal and External

History
Originally released in NX3

```
int UF_UDE_ask_param_toggle
(
    UF_UDE_t ude_obj,
    char * param_name,
    UF_UDE_param_toggle_t * toggle
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
UF_UDE_param_toggle_t *	**toggle**	Output	see above

UF_UDE_ask_param_type (view source)

Defined in: uf_ude.h

Overview
This function returns type of a parameter of name 'param_name' in the User
Defined Machine Control Event object 'ude_obj'.

Environment
Internal and External

History
Originally released in V18.0

```
int UF_UDE_ask_param_type
(
    UF_UDE_t ude_obj,
    char * param_name,
    UF_UDE_param_type_t * param_type
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
UF_UDE_param_type_t *	**param_type**	Output	Type of the parameter

---

UF_UDE_ask_params (view source)

Defined in: uf_ude.h

Overview

This function returns the number and names of parameters of the User Defined Machine Control Event object 'ude_obj'.

NOTE: The returned array must be freed by calling UF_free_string_array.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_params
(
    UF_UDE_t ude_obj,
    int * number_of_params,
    char * * * param_names
)
```

UF_UDE_t	**ude_obj**	Input	see above
int *	**number_of_params**	Output	Number of parameters
char * * *	**param_names**	Output to UF_*free*	Names of the parameters The returned array must be freed by calling UF_free_string_array.

UF_UDE_ask_point (view source)

Defined in: uf_ude.h

Overview

This function returns in 'smart_point_tag' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object 'ude_obj'.

NOTE: The 'smart_point_tag' is the tag of a smart point

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_point
(
    UF_UDE_t ude_obj,
    char * param_name,
    tag_t * smart_point_tag
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
tag_t *	**smart_point_tag**	Output	see above

UF_UDE_ask_string (view source)

Defined in: uf_ude.h

Overview

This function returns in 'value' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object specified by 'ude_obj'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_string
(
    UF_UDE_t ude_obj,
    char * param_name,
    char * * value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
char * *	**value**	Output to UF_*free*	see above. This should be freed with a call to UF_free on value

---

UF_UDE_ask_vector (view source)

Defined in: uf_ude.h

Overview

This function returns in 'smart_vector_tag' the value of the parameter specified by 'param_name'. It is the value of this parameter that is currently being used by the object 'ude_obj'.

NOTE: The 'smart_vector_tag' is the tag of a smart vector

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_ask_vector
(
    UF_UDE_t ude__obj,
    char * param_name,
    tag_t * smart_vector_tag
)
```

UF_UDE_t	**ude__obj**	Input	see above
char *	**param_name**	Input	see above
tag_t *	**smart_vector_tag**	Output	see above

UF_UDE_is_param_optional (view source)

Defined in: uf_ude.h

Overview

This function returns TRUE in 'response' if the parameter of the User
Defined Machine Control Event object 'ude_obj' is optional and FALSE if not

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_is_param_optional
(
    UF_UDE_t ude_obj,
    char * param_name,
    logical * response
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
logical *	**response**	Input	see above

UF_UDE_set_boolean (view source)

Defined in: uf_ude.h

Overview
This function assigns the value 'value' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

Environment
Internal and External

History
Originally released in V18.0

int UF_UDE_set_boolean
(
UF_UDE_t ude_obj,
char * param_name,
logical param_value
)

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
logical	**param_value**	Input	see above

UF_UDE_set_double (view source)

Defined in: uf_ude.h

Overview
This function assigns the value 'value' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

Environment
Internal and External

History
Originally released in V18.0

int UF_UDE_set_double
(
UF_UDE_t ude_obj,
char * param_name,
double value
)

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
double	**value**	Input	see above

UF_UDE_set_integer (view source)

Defined in: uf_ude.h

Overview

This function assigns the value 'value' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_set_integer
(
    UF_UDE_t ude_obj,
    char * param_name,
    int value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
int	**value**	Input	see above

UF_UDE_set_param_toggle (view source)

Defined in: uf_ude.h

Overview

This function sets the parameter of name 'param_name' in the User Defined Machine Control Event object 'ude_obj' to be active or inactive as speicifed by the value of 'toggle'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_set_param_toggle
(
    UF_UDE_t ude_obj,
    char * param_name,
    UF_UDE_param_toggle_t toggle
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
UF_UDE_param_toggle_t	**toggle**	Input	see above

UF_UDE_set_point (view source)

Defined in: uf_ude.h

Overview

This function assigns the value 'smart_point_tag' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

NOTE: The tag that is passed to this function has to be the tag of a smart point

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_set_point
(
    UF_UDE_t ude_obj,
    char * param_name,
    tag_t smart_point_tag
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
tag_t	**smart_point_tag**	Input	see above

UF_UDE_set_string (view source)

Defined in: uf_ude.h

Overview

This function assigns the value 'value' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_set_string
(
    UF_UDE_t ude_obj,
    char * param_name,
    char * value
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
char *	**value**	Input	see above

UF_UDE_set_vector (view source)

Defined in: uf_ude.h

Overview

This function assigns the value 'smart_vector_tag' to the parameter specified by 'param_name' for the object specified by 'ude_obj'.

NOTE: The tag that is passed to this function has to be the tag of a smart vector (also known as a "smart direction" object

Environment

Internal and External

History

Originally released in V18.0

```
int UF_UDE_set_vector
(
    UF_UDE_t ude_obj,
    char * param_name,
    tag_t smart_vector_tag
)
```

UF_UDE_t	**ude_obj**	Input	see above
char *	**param_name**	Input	see above
tag_t	**smart_vector_tag**	Input	see above

UF_UI_ONT_ask_selected_nodes (view source)

Defined in: uf_ui_ont.h

Overview

This function returns the number and the tags of the selected nodes in the active view of the Operation Navigation Tool(ONT).

Environment

Internal

History

Originally released in V16.0

```
int UF_UI_ONT_ask_selected_nodes
(
    int * count,
    tag_t * * objects
)
```

int *	**count**	Output	- Number of selected nodes

tag_t * *	**objects**	Output to UF_*free*	- the tags of the selected nodes. The returned array must be freed by calling UF_free.

UF_UI_ONT_ask_view (view source)

Defined in: uf_ui_ont.h

Overview

This function returns the current view mode of the ONT

Return

= 0 Successful
other failing error number

Environment

Internal

History

Originally released in NX3

```
int UF_UI_ONT_ask_view
(
    UF_UI_ONT_tree_mode_t * view
)
```

UF_UI_ONT_tree_mode_t *	**view**	Output

UF_UI_ONT_collapse_view (view source)

Defined in: uf_ui_ont.h

Overview

This function collapses all nodes of the current view

Return

= 0 Successful
other failing error number

Environment

Internal

History

Originally released in NX3

```
int UF_UI_ONT_collapse_view
(
    void
)
```

UF_UI_ONT_expand_view (view source)

Defined in: uf_ui_ont.h

Overview
This function expands all nodes of the current view

Return
= 0 Successful
other failing error number

Environment
Internal

History
Originally released in NX3

```
int UF_UI_ONT_expand_view
(
    void
)
```

UF_UI_ONT_refresh (view source)

Defined in: uf_ui_ont.h

Overview
This function refreshes the operation navigator.

Return
= 0 Successful
other failing error number

Environment
Internal

History
Originally released in V19.0

```
int UF_UI_ONT_refresh
(
    void
)
```

UF_UI_ONT_switch_view (view source)

Defined in: uf_ui_ont.h

Overview
This function changes the view of the ONT to the specified view

Return

= 0 Successful
other failing error number

Environment
Internal

History
Originally released in NX3

```
int UF_UI_ONT_switch_view
(
    UF_UI_ONT_tree_mode_t view
)
```

UF_UI_ONT_tree_mode_t	**view**	Input

UF_UI_PARAM_edit_object (view source)

Defined in: uf_ui_param.h

Overview
This function displays the dialog for the object of the input "obj_tag".
The obj_tag has to be CAM object. The dialog that is displayed will be the default dialog for the object. In the interactive session editing a CAM object from the Operation Navigator causes the dialog of the object to be brought up. Calling this function will have the same effect.

Environment
Internal

History
Originally released in V16.0

```
int UF_UI_PARAM_edit_object
(
    tag_t obj_tag,
    int * dialog_response
)
```

tag_t	**obj_tag**	Input	- tag of the UF_PARAM object for which the dialog is requested
int *	**dialog_response**	Output	- Response from the dialog of the object Possible values UF_UI_OK UF_UI_APPLY UF_UI_BACK UF_UI_CANCEL
