uc5800 (view source)

Defined in: uf_assem.h

Overview
uc5800 create component by retrieval

Environment
Internal and External

Note: No Metric/English auto-scaling will be done.

Required License(s)
assemblies

```
int uc5800
(
    const char * cp1,
    const char * cp2,
    const char * cp3,
    double * rp4,
    int ip5,
    tag_t * nr6
)
```

const char *	**cp1**	Input	Component name
const char *	**cp2**	Input	Component part fspec with extensions
const char *	**cp3**	Input	Reference set name
double *	**rp4**	Input	Component information (1 - 9) component orientation matrix (10-12) component origin (13) component scale - obsolete
int	**ip5**	Input	Layer Mode 0 = retrieve on original layers 1 = retrieve on work layer
tag_t *	**nr6**	Output	Component entity ID

---

uc5801 (view source)

Defined in: uf_assem.h

Overview
uc5801 read component data

Environment
Internal and External

Required License(s)
gateway

```
void uc5801
(
    tag_t np1,
    char cr2 [ UF_CFI_MAX_FILE_NAME_BUFSIZE ] ,
    char cr3 [ MAX_FSPEC_BUFSIZE ] ,
    char cr4 [ UF_OBJ_NAME_BUFSIZE ] ,
```

```
    double * rr5,
    int * ir6
)
```

tag_t	**np1**	Input	Component entity id
char	**cr2 [UF_CFI_MAX_FILE_NAME_BUFSIZE]**	Output	Component name
char	**cr3 [MAX_FSPEC_BUFSIZE]**	Output	Component part name
char	**cr4 [UF_OBJ_NAME_BUFSIZE]**	Output	Component reference set name
double *	**rr5**	Output	Component orientation information (1 - 9) component orientation matrix (10-12) component origin (13) component scale - obsolete
int *	**ir6**	Output	Computational date and times (1-2) component entity (3-4) component part note: entity will be current date/time part will be the beginning of NX time

---

uc5802 (view source)

Defined in: uf_assem.h

Overview
uc5802 edit component data

Environment
Internal and External

Required License(s)
gateway

```
void uc5802
(
    tag_t np1,
    const char * cp2,
    const char * cp3,
    double * rp4
)
```

tag_t	**np1**	Input	Component entity id
const char *	**cp2**	Input	Component part name
const char *	**cp3**	Input	Reference set name
double *	**rp4**	Input	Rigid motion information (1-9) rigid motion matrix (10-12) rigid motion translation (13) component scale obsolete

---

uc5810 (view source)

Defined in: uf_assem.h

Overview
Creates a reference set.

Environment
Internal and External

Required License(s)
assemblies

```
int uc5810
(
    const char * cp1,
    double * rp2,
    tag_t np3 [ ] ,
    int ip4,
    tag_t * nr5
)
```

const char *	**cp1**	Input	Reference set name (30 char max)
double *	**rp2**	Input	[0] - [8] - Reference Set Orientation Matrix [9] - [11] - Reference Set Origin
tag_t	**np3 [ ]**	Input	Array of reference set member object identifier's
int	**ip4**	Input	Number Of object identifier's In np3
tag_t *	**nr5**	Output	Reference set object identifier

---

uc5811 (view source)

Defined in: uf_assem.h

Overview
Returns reference set name, orientation matrix, and origin.

Environment
Internal and External

Required License(s)
gateway

```
int uc5811
(
    tag_t np1,
    char cr2 [ UF_OBJ_NAME_BUFSIZE ] ,
    double * rr3
)
```

tag_t	**np1**	Input	Reference set object identifier
char	**cr2 [UF_OBJ_NAME_BUFSIZE]**	Output	
double *	**rr3**	Output	12 element array: [0] - [8] - Reference Set Orientation Matrix [9]-[11] - Reference Set Origin

---

uc5812 (view source)

Defined in: uf_assem.h

Overview
Edits reference set name, origin, and orientation matrix.

Environment
Internal and External

Required License(s)
assemblies

```
int uc5812
(
    tag_t np1,
    const char * cp2,
    double * rp3
)
```

tag_t	**np1**	Input	Reference set object identifier
const char *	**cp2**	Input	Reference set name (UF_OBJ_NAME_NCHARS characters max)
double *	**rp3**	Input	12 element array: [0] - [8] - Reference Set Orientation Matrix [9] - [11] - Reference Set Origin

---

uc5816 (view source)

Defined in: uf_assem.h

Overview
uc5816 create component by filing

Environment
Internal and External

Required License(s)
assemblies

```
int uc5816
(
    const char * cp1,
    const char * cp2,
    const char * cp3,
    double * rp4,
    tag_t * np5,
    int ip6,
    tag_t * na7
)
```

const char *	**cp1**	Input	Component name
const char *	**cp2**	Input	Component part fpsec
const char *	**cp3**	Input	Reference set name 0 = all of part

double *	**rp4**	Input	Component information (1-9) component orientation matrix (10-12) component origin
tag_t *	**np5**	Input	Array of entity id's to add to component
int	**ip6**	Input	Count of entity ids in previous argument
tag_t *	**na7**	Input / Output	Component entity ID Input: 0 = file component <>0 = component eid to refile Output: component entity id if created by filing

---

uc5818 (view source)

Defined in: uf_assem.h

Overview
Do a Where-Used List on a Component Part. This operation only finds component instances in archived parts (on disk).

Return
Error Code:
< 0 = Error
0 = No Error
9 = Option Not Available
21 = No Files Found

Environment
Internal and External

Required License(s)
gateway

```
int uc5818
(
    const char * cp1,
    const char * cp2
)
```

const char *	**cp1**	Input	Component Part Name
const char *	**cp2**	Input	Root Directory To Search (100 char max)

---

uf5804 (view source)

Defined in: uf_assem.h

Overview
uf5804 read component member

Environment
Internal and External

Required License(s)
gateway

```
void uf5804
(
    tag_t * np1,
    int * ip2,
    tag_t * nr3
)
```

tag_t *	**np1**	Input	Component entity id
int *	**ip2**	Input	Member instance to read
tag_t *	**nr3**	Output	Component member entity id

UF_ASSEM_activate_sequence (view source)

Defined in: uf_assem.h

Overview

Activate the specified sequence. The active sequence is the default sequence displayed in the sequence navigator (if running in internal mode).

If this function is called outside the sequencing environment (i.e. UF_ASSEM_initialize_sequence has not been called before) then this will return the last sequence that was active in the NX session.

Environment

Internal and External

History

Initially released in V18.0

Required License(s)

assemblies

```
int UF_ASSEM_activate_sequence
(
    tag_t sequence
)
```

tag_t	**sequence**	Input	The sequence to be activated

UF_ASSEM_add_part_to_assembly (view source)

Defined in: uf_assem.h

Overview

Adds an instance of a part to a parent part. If the part is not loaded, it is loaded into the current session. The instance is added to the parts list according to the setting of the parts_list switch (see UF_ASSEM_ask_assem_options and UF_ASSEM_options_s). The user allocated structure error_status is filled with the names and associated error codes of the loaded parts. The allocated arrays must be freed with UF_free and UF_free_string_array.

If refset_name is a NULL pointer or a zero-length string, the entire part is used. If instance_name is a NULL pointer or a zero-length string, the instance

is unnamed.

Environment

Internal and External

See Also

The following table of parameter values for this function is based on the automobile example
See table

Required License(s)

assemblies

```
int UF_ASSEM_add_part_to_assembly
(
    tag_t parent_part,
    const char * part,
    const char * refset_name,
    const char * instance_name,
    double origin [ 3 ] ,
    double csys_matrix [ 6 ] ,
    int layer,
    tag_t * instance,
    UF_PART_load_status_t * error_status
)
```

tag_t	**parent_part**	Input	tag of part to add instance to
const char *	**part**	Input	name of part to instance. The part name can include a directory path. This can not exceed MAX_FSPEC_NCHARS characters.
const char *	**refset_name**	Input	Name of reference set to use from component parts. The refset_name cannot exceed UF_OBJ_NAME_NCHARS characters, cannot include a directory path, and should not have a file extension.
const char *	**instance_name**	Input	Name of new instance The instance_name cannot exceed UF_OBJ_NAME_NCHARS characters, cannot include a directory path, and should not have a file extension.
double	**origin [3]**	Input	Position in <parent_part> where the instance is to be created
double	**csys_matrix [6]**	Input	Orientation of the instance
int	**layer**	Input	-1 Means use the original layers. 0 Means use the work layer 1-255 Means use the specified layer.
tag_t *	**instance**	Output	Tag of the new instance in the work part
UF_PART_load_status_t *	**error_status**	Output to UF_*free*	User allocted structure consisting of names and associated error codes. The allocated arrays must be freed with UF_free_string_array and UF_free. See the description of the UF_PART_load_status_t structure for details on freeing this structure.

UF_ASSEM_add_ref_set_members (view source)

Defined in: uf_assem.h

Overview
Adds an array of members to the reference set passed in.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
assemblies

```
int UF_ASSEM_add_ref_set_members
(
    tag_t ref_set,
    int member_count,
    tag_t * ref_set_members
)
```

tag_t	**ref_set**	Input	Tag of the reference set to add to.
int	**member_count**	Input	The number of new members to add.
tag_t *	**ref_set_members**	Input	Array of the new members to add.

UF_ASSEM_add_sequencing_view (view source)

Defined in: uf_assem.h

Overview
Add a view to the list of views used to display sequencing.
This should called after UF_ASSEM_initialize_sequencing_keep_layout().
Then user can keep their view layout and views of this layout into additional view set. After that the animation of play a motion sequence will show in these additional views.

Environment
Internal and External

History
Initially released in NX 8.5.1

Required License(s)
assemblies

```
int UF_ASSEM_add_sequencing_view
(
    tag_t view
)
```

tag_t	**view**	Input	The tag of view to add.

UF_ASSEM_add_to_cset (view source)

Defined in: uf_assem.h

Overview

Allows you to add a component to an existing component set. You can add the component at the single level (level = FALSE), where only the given component is added to the set. Or, you can add the component at all levels (level = TRUE), where all the child components of the given component are also implicitly added to the set. This means that if subsequently, the component has components added or removed, the change is also reflected in the component set.

Environment

Internal and External

See Also

UF_ASSEM_create_cset
to create a new, empty component set.
Refer to example

Required License(s)

assemblies

```
int UF_ASSEM_add_to_cset
(
    tag_t cset,
    tag_t component,
    logical level
)
```

tag_t	**cset**	Input	Object identifier of component set
tag_t	**component**	Input	Object identifier of component to be added
logical	**level**	Input	Include child components switch

UF_ASSEM_apply_to_cset (view source)

Defined in: uf_assem.h

Overview

This routine allows you to apply the given function to each member of the given component set and, if relevant, to all the members component children.

Environment

Internal and External

See Also

Refer to example

Required License(s)

assemblies

```
int UF_ASSEM_apply_to_cset
(
    tag_t cset,
    UF_ASSEM_cset_fn_t fn,
    void * app_data
)
```

tag_t	**cset**	Input	Object identifier of component set
UF_ASSEM_cset_fn_t	**fn**	Input	Pointer to required function type
void *	**app_data**	Input / Output	Pointer to any application data or NULL

UF_ASSEM_apply_to_cset_members (view source)

Defined in: uf_assem.h

Overview
Allows you to apply the given function to each, explicit member of the given component set. If you want to apply the function to the children of each member too, see UF_ASSEM_apply_to_cset.

Environment
Internal and External

See Also
UF_ASSEM_apply_to_cset

Required License(s)
assemblies

```
int UF_ASSEM_apply_to_cset_members
(
    tag_t cset,
    UF_ASSEM_cset_fn_t fn,
    void * app_data
)
```

tag_t	**cset**	Input	Object identifier of component set
UF_ASSEM_cset_fn_t	**fn**	Input	Pointer to required function type
void *	**app_data**	Input / Output	Pointer to any application data or NULL

UF_ASSEM_ask_active_arrangement (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_active_arrangement

Returns the Active Assembly Arrangement for a given part. This is the Assembly Arrangement which would take effect for it if that part were made the Displayed Part.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also

UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History
Initially released in NX2.

Required License(s)
gateway

int UF_ASSEM_ask_active_arrangement
(
tag_t part,
tag_t * arrangement
)

tag_t	**part**	Input	The part to query
tag_t *	**arrangement**	Output	The Active Assembly Arrangement for that part

---

UF_ASSEM_ask_active_sequence (view source)

Defined in: uf_assem.h

Overview
Returns the sequence that is currently active in the session.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

int UF_ASSEM_ask_active_sequence
(
tag_t * sequence
)

tag_t *	**sequence**	Output	The active sequence in the session

---

UF_ASSEM_ask_all_comp_cset (view source)

Defined in: uf_assem.h

Overview
Allows you to return the system component set which contains (implicitly) all the components in the given part.

Environment

Internal and External

See Also
Refer to example

Required License(s)
gateway

int UF_ASSEM_ask_all_comp_cset
(
tag_t part,
tag_t * cset
)

tag_t	**part**	Input	Object identifier of a part
tag_t *	**cset**	Output	Object identifier of component set found

---

UF_ASSEM_ask_all_part_occ_children (view source)

Defined in: uf_assem.h

Overview
Asks for a list of all part occurrences under a given part occurrence, including suppressed part occurrence children.

Return
Returns the count of part occurrences in the list.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_all_part_occ_children
(
tag_t part_occur,
tag_t * * child_part_occs
)

tag_t	**part_occur**	Input	Tag of part occurrence to query for children
tag_t * *	**child_part_occs**	Output to UF_*free*	Array of tags of child part occurrences. The allocated list of tags must be freed with UF_free when it is no longer needed.

UF_ASSEM_ask_arrangements_in_part (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_arrangements_in_part

Returns all the Assembly Arrangements in a part.

Return

Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
Refer to example

History
Initially released in NX2.

Required License(s)
gateway

```
int UF_ASSEM_ask_arrangements_in_part
(
    tag_t part,
    int * n_arrangements,
    tag_t * * arrangements
)
```

tag_t	**part**	Input	The part to query
int *	**n_arrangements**	Output	The number of Assembly Arrangements returned
tag_t * *	**arrangements**	Output to UF_*free*	The returned Assembly Arrangements

---

UF_ASSEM_ask_arrays_in_part (view source)

Defined in: uf_assem.h

Overview
Returns a list of the component arrays defined by a given part.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_arrays_in_part
(
    tag_t part_tag,
    int * num_arrays,
    tag_t * * array_tags
)
```

tag_t	**part_tag**	Input	Tag of part being queried.
int *	**num_arrays**	Output	The number of arrays in the part.

tag_t * *	**array_tags**	Output to UF_*free*	Dynamically allocated array of array tags. This must be freed by calling UF_free.

UF_ASSEM_ask_arrays_of_inst (view source)

Defined in: uf_assem.h

Overview
Returns a list of the arrays which contain a given instance.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_arrays_of_inst
(
    tag_t instance,
    int * num_arrays,
    tag_t * * arrays
)
```

tag_t	**instance**	Input	Tag of instance being queried.
int *	**num_arrays**	Output	The number of arrays which contain "instance_tag".
tag_t * *	**arrays**	Output to UF_*free*	Dynamically allocated array of array tags. This must be freed by the caller using UF_free.

UF_ASSEM_ask_assem_options (view source)

Defined in: uf_assem.h

Overview
Returns the current settings of the assembly options into the options structure supplied by the user.

Environment
Internal and External

See Also
Refer to UF_ASSEM_options_t

Required License(s)
gateway

```
int UF_ASSEM_ask_assem_options
(
    UF_ASSEM_options_t * options
)
```

UF_ASSEM_options_t *	**options**	Output	Pointer to an assembly options structure

---

UF_ASSEM_ask_auto_add_new_comps (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_auto_add_new_comps

Ask whether new components will be added to the specified reference set automatically or not.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

History
Initially released in NX3.

Required License(s)
gateway

```
int UF_ASSEM_ask_auto_add_new_comps
(
    tag_t ref_set,
    logical * add_new_comps
)
```

tag_t	**ref_set**	Input	The reference set
logical *	**add_new_comps**	Output	Whether new components are to be added automatically

---

UF_ASSEM_ask_bodies_of_assembly_cut (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_bodies_of_assembly_cut

Ask the target and tool bodies for the assembly cut.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

History
Initially released in NX3.

Required License(s)
gateway

```
int UF_ASSEM_ask_bodies_of_assembly_cut
(
    tag_t frec,
    int * n_target_bodies,
```

```
    tag_t * * target_body_tags,
    int * n_tool_bodies,
    tag_t * * tool_body_tags
)
```

tag_t	**frec**	Input	The assembly cut tag
int *	**n_target_bodies**	Output	Number of target bodies
tag_t * *	**target_body_tags**	Output to UF_*free*	Target bodies
int *	**n_tool_bodies**	Output	Number of tool bodies
tag_t * *	**tool_body_tags**	Output to UF_*free*	Tool bodies

UF_ASSEM_ask_child_of_instance (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the child part of an instance. Returns NULL_TAG if the child part of an instance is unloaded.

Return
Returns tag of child part

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_child_of_instance
(
    tag_t instance
)
```

tag_t	**instance**	Input	Tag of instance to query for child part

---

UF_ASSEM_ask_comp_explosion (view source)

Defined in: uf_assem.h

Overview
Returns the explosion status and explosion transform of this particular part occurrence. A part occurrence also inherits the explosion transforms of its parents. This is intended to supplement UF_ASSEM_ask_component_data used on instances.

Environment
Internal and External

See Also

UF_ASSEM_explode_component
UF_ASSEM_unexplode_component
UF_ASSEM_revert_explode_comp

Required License(s)
gateway

int UF_ASSEM_ask_comp_explosion
(
tag_t explosion,
tag_t component,
UF_ASSEM_expl_status_p_t status,
double transform [4] [4]
)

tag_t	**explosion**	Input	The explosion to query
tag_t	**component**	Input	The component (its part occurrence)
UF_ASSEM_expl_status_p_t	**status**	Output	The exploded status of the part occurrence. This can be one of the following enumerated constants: UF_ASSEM_unexploded UF_ASSEM_exploded UF_ASSEM_revert_exploded
double	**transform [4] [4]**	Output	If status = UF_ASSEM_exploded, a 4x4 non-distorting transform (see UF_ASSEM_ask_component_data for more information).

UF_ASSEM_ask_comp_position (view source)

Defined in: uf_assem.h

Overview
Returns the total absolute transform of the component in the given explosion. This is intended to replace the transform returned by UF_ASSEM_ask_component_data when used on part occurrences.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_comp_position
(
tag_t explosion,
tag_t component,
double transform [4] [4]
)

tag_t	**explosion**	Input	The explosion to query
tag_t	**component**	Input	The component (its part occurrence)
double	**transform [4] [4]**	Output	a 4x4 non-distorting transform (see UF_ASSEM_ask_component_data for more information).

UF_ASSEM_ask_component_data (view source)

Defined in: uf_assem.h

Overview

Returns data about an instance or a part occurrence. The instance_name output is always the name of the instance, even when a part occurrence is passed to the function. Note that the names of the part occurrence and its corresponding instance can be different. The refset_name can be different between a part occurrence and its instance. The full 9 elements of the csys_matrix are returned. The full transformation matrix is also returned. The transformation matrix is the 4x4 matrix that combines the origin and CSYS-matrix into a single matrix.

The format of the 4x4 transform is that the elements:

transform[0][0], transform[0][1], transform[0][2]
transform[1][0], transform[1][1], transform[1][2]
transform[2][0], transform[2][1], transform[2][2]

form a 3x3 orthnormal rotation matrix, the elements :

transform[0][3]
transform[1][3]
transform[2][3]

represent a (X, Y, Z) translation in the units of the part containing the component, element

transform[3][3]

is always set to 1.0, and the remaining elements are always set to 0.0.

CAUTION: If the reference set of the component you are asking for data on is set to "Entire Part", the string returned by UF_ASSEM_ask_component_data for refset_name (third argument) is "None".

Environment

Internal and External

Required License(s)

gateway

```
int UF_ASSEM_ask_component_data
(
    tag_t component,
    char part_name [ MAX_FSPEC_BUFSIZE ] ,
    char refset_name [ UF_OBJ_NAME_BUFSIZE ] ,
    char instance_name [ UF_CFI_MAX_FILE_NAME_BUFSIZE ] ,
    double origin [ 3 ] ,
    double csys_matrix [ 9 ] ,
    double transform [ 4 ] [ 4 ]
)
```

tag_t	**component**	Input	Tag of instance or part occurrence
char	**part_name [MAX_FSPEC_BUFSIZE]**	Output	Name of part
char	**refset_name [UF_OBJ_NAME_BUFSIZE]**	Output	Name of the reference set in use
char	**instance_name [UF_CFI_MAX_FILE_NAME_BUFSIZE]**	Output	name of instance
double	**origin [3]**	Output	Position of component
double	**csys_matrix [9]**	Output	Coordinate System Matrix

double	**transform [4] [4]**	Output	Transformation Matrix

UF_ASSEM_ask_cost_of_sequence (view source)

Defined in: uf_assem.h

Overview
Asks the cost for the given sequence. The time for the sequence is an aggregate of the time for each of the steps in the assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_cost_of_sequence
(
    tag_t sequence,
    double * cost
)
```

tag_t	**sequence**	Input	The tag of the given sequence
double *	**cost**	Output	The cost of the sequence

UF_ASSEM_ask_cost_of_step (view source)

Defined in: uf_assem.h

Overview
Asks the cost of the given step

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_cost_of_step
(
    tag_t step,
    double * cost
)
```

tag_t	**step**	Input	The tag of the given step
double *	**cost**	Output	The cost of the step

UF_ASSEM_ask_current_frame (view source)

Defined in: uf_assem.h

Overview

Get the current playback frame for the specified sequence. This is equivalent to the value displayed in the "current frame" combo box in the Sequencing UI: when the sequence duration is N frames, the current frame is an integer in the inclusive range 0 to N.

Example: a sequence has two motion steps, whose durations are 5 and 7. Frame 0 represents the very beginning of the sequence; frame 5 represents the positions of the components after the first step has completed but before the second step has begun; frame 12 represents the very end of the sequence.

If the specified sequence is not the active sequence, return 0, as this would be the current frame if the sequence were made active.

Environment

Internal and External

History

Initially released in NX 8.0.1

Required License(s)

assemblies

```
int UF_ASSEM_ask_current_frame
(
    tag_t sequence,
    int * current_frame
)
```

tag_t	**sequence**	Input	The tag of the sequence
int *	**current_frame**	Output	The current playback frame of the sequence.

UF_ASSEM_ask_current_step (view source)

Defined in: uf_assem.h

Overview

Returns the current step in the given assembly sequence.
It is now possible that the playback can be stopped in the middle of a sequence step (i.e. a sequence step is still in progress). If this is the case this function will return the sequence step that is currently in progress.

If this function is called outside the sequencing environment then this will return the first step of the given sequence since that is the step that would be played first if that sequence was made active.

Environment

Internal and External

History

Initially released in V18.0

Required License(s)

gateway

```
int UF_ASSEM_ask_current_step
(
    tag_t sequence,
    tag_t * step
)
```

tag_t	**sequence**	Input	Tag of the sequence
tag_t *	**step**	Output	The step that will be played back next

---

UF_ASSEM_ask_default_arrangement (view source)

Defined in: uf_assem.h

Overview

UF_ASSEM_ask_default_arrangement

Returns the Default Assembly Arrangement for a given part. This is the Assembly Arrangement which would used by a new parent assembly if this part were added to that parent as a new component.

Return

Error code - use UF_get_fail_message to obtain the corresponding error message

Environment

This function is supported for both Internal and External execution.

See Also

UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History

Initially released in NX2.

Required License(s)

gateway

```
int UF_ASSEM_ask_default_arrangement
(
    tag_t part,
    tag_t * arrangement
)
```

tag_t	**part**	Input	The part to query
tag_t *	**arrangement**	Output	The Default Assembly Arrangement for that part

---

UF_ASSEM_ask_default_ref_sets (view source)

Defined in: uf_assem.h

Overview
Returns the current list of default reference sets. The names of the reference sets are returned in the order they appear in the load options dialog (and the order in which they are applied).
UF_ASSEM_set_default_ref_sets for setting these defaults.

Environment
Internal and External

See Also
UF_ASSEM_restore_load_options
UF_ASSEM_set_default_ref_sets
Refer to example

History
Original release was in V13.0.

Required License(s)
gateway

```
int UF_ASSEM_ask_default_ref_sets
(
    int * n_ref_sets,
    char * * * default_ref_sets
)
```

int *	**n_ref_sets**	Output	The number of default reference sets.
char * * *	**default_ref_sets**	Output to UF_*free*	An ordered array of reference set names indicating the default reference sets currently in use. This is returned in allocated memory and should be freed using UF_free_string_array.

UF_ASSEM_ask_deformable_definition (view source)

Defined in: uf_assem.h

Environment
Internal and External

History
Initially released in NX2

Required License(s)
gateway

```
int UF_ASSEM_ask_deformable_definition
(
    tag_t part,
    tag_t * deformable_feature
)
```

tag_t	**part**	Input
tag_t *	**deformable_feature**	Output

UF_ASSEM_ask_deformable_definition_data (view source)

Defined in: uf_assem.h

Overview
Inquire routine to obtain the data for the deformable feature.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
gateway

```
int UF_ASSEM_ask_deformable_definition_data
(
    tag_t deformable_feature_tag,
    UF_ASSEM_deform_data_p_t deform_data
)
```

tag_t	**deformable_feature_tag**	Input
UF_ASSEM_deform_data_p_t	**deform_data**	Output

UF_ASSEM_ask_deformed_definition_data (view source)

Defined in: uf_assem.h

Overview
Inquire routine to obtain the data for the deformed feature.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
gateway

```
int UF_ASSEM_ask_deformed_definition_data
(
    tag_t deformed_feature_tag,
    UF_ASSEM_deformed_definition_data_p_t deformed_data
)
```

tag_t	**deformed_feature_tag**	Input
UF_ASSEM_deformed_definition_data_p_t	**deformed_data**	Output

---

UF_ASSEM_ask_displayed_deformation_of_part_occ (view source)

Defined in: uf_assem.h

Overview

Ask for the current deformed feature for the part occurrence in the context of the current display part.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
gateway

int UF_ASSEM_ask_displayed_deformation_of_part_occ
(
tag_t part_occ,
tag_t * deformed_feature
)

tag_t	**part_occ**	Input
tag_t *	**deformed_feature**	Output

UF_ASSEM_ask_exploded_object (view source)

Defined in: uf_assem.h

Overview
In normal circumstances, a dimension or other annotation which is attached to an object, even in an exploded view, stays in the real position of that object. When attaching such annotation to an object in an exploded view, the new_object tag given by this function should be used instead of the normal tag.

Please note that the behavior of an annotation attached to the unexploded object in an exploded view is not guaranteed to be consistent, and indeed may be considered erroneous in some cases. The function is designed to be as resilient as possible, as is demonstrated by the example.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_exploded_object
(
tag_t explosion,
tag_t old_object,
tag_t * new_object
)

tag_t	**explosion**	Input	The tag of the explosion used in the view to which you wish to place the annotation.
tag_t	**old_object**	Input	The normal tag for the object.
tag_t *	**new_object**	Output	The tag of the exploded version, or the original tag in all cases where there is no exploded version.

UF_ASSEM_ask_explosion_vector (view source)

Defined in: uf_assem.h

Overview
Returns the suggested explosion vector for the component (the direction in which it appears to be explodable), derived from mating conditions. An error is returned if the component is not mated or not properly constrained by the mating conditions.

Environment
Internal and External

See Also
UF_ASSEM_create_explosion

Required License(s)
gateway

```
int UF_ASSEM_ask_explosion_vector
(
    tag_t component,
    double vector [ 3 ]
)
```

tag_t	**component**	Input	the tag of a part occurrence not instance
double	**vector [3]**	Output	The suggested explosion vector derived from mating conditions in absolute coordinates.

UF_ASSEM_ask_explosions (view source)

Defined in: uf_assem.h

Overview
Asks for all the explosions in a part file

Environment
Internal and External

History
Initially released into NX3.0

Required License(s)
gateway

```
int UF_ASSEM_ask_explosions
(
    tag_t part_tag,
    int* n_explosions,
    tag_t* * explosion_tags
)
```

tag_t	**part_tag**	Input	The tag of the part for which to get the explosions.
int*	**n_explosions**	Output	Number of explosions.
tag_t* *	**explosion_tags**	Output to UF_*free*	The tags of the explosions in this part. This array must be freed by the caller using UF_free.

UF_ASSEM_ask_hidden_comps (view source)

Defined in: uf_assem.h

Overview
Return an array of the components hidden in the given view.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_hidden_comps
(
    tag_t view,
    tag_p_t* components,
    int* count
)
```

tag_t	**view**	Input	The view you are seeking informaton on
tag_p_t*	**components**	Output to UF_*free*	The components which are erased in the view. This array must be freed by the caller using UF_free.
int*	**count**	Output	The number of component tags in the array.

UF_ASSEM_ask_inst_of_part_occ (view source)

Defined in: uf_assem.h

Overview
Returns the instance tag of a part occurrence.

Return
Instance tag of part occurrence

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_inst_of_part_occ
(
    tag_t part_occur
)
```

tag_t	**part_occur**	Input	Tag of part occurrence to query for instance.

UF_ASSEM_ask_instance_intent (view source)

Defined in: uf_assem.h

Overview
Returns intent data of an instance. The caller must allocate the UF_ASSEM_instance_intent_t structure, and pass the pointer to that structure to this routine. The routine then fills the structure with data, which includes dynamically allocated data.

Environment
Internal and External

See Also
UF_ASSEM_free_instance_intent

Required License(s)
gateway

```
int UF_ASSEM_ask_instance_intent
(
    tag_t instance,
    UF_ASSEM_instance_intent_p_t instance_intent
)
```

tag_t	**instance**	Input	Object identifier of the instance
UF_ASSEM_instance_intent_p_t	**instance_intent**	Output to UF_*free*	Data of the intent. The caller must free data inside this structure by calling UF_ASSEM_free_instance_intent when the structure is no longer needed.

UF_ASSEM_ask_instance_of_name (view source)

Defined in: uf_assem.h

Overview
Return the tag of the instance which has the given name, below the parent part (one level only).

Return
Tag of named instance

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_instance_of_name
(
    tag_t parent_part,
    const char * instance_name
)
```

tag_t	**parent_part**	Input	Tag of parent part
const char *	**instance_name**	Input	Name of instance to obtain tag for; must be no longer than UF_OBJ_NAME_NCHARS characters.

UF_ASSEM_ask_iset_array_data (view source)

Defined in: uf_assem.h

Overview

Returns information about an ISET Component array in the "array_data" structure and a dynamically allocated array of tags.

For rectangular arrays, the components in the list are as follows:

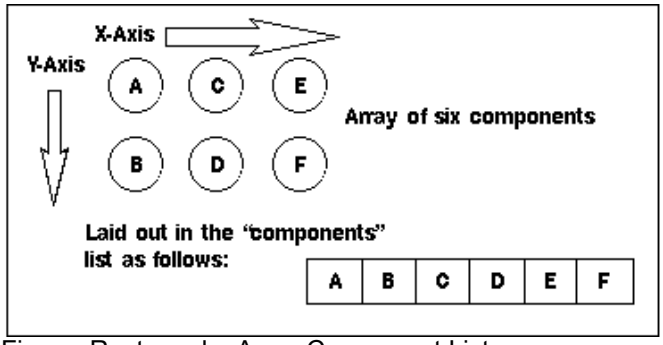


Figure. Rectangular Array Component List

Environment

Internal and External

Required License(s)

gateway

```
int UF_ASSEM_ask_iset_array_data
(
    tag_t array,
    UF_ASSEM_iset_array_data_p_t array_data,
    tag_t * * components
)
```

tag_t	**array**	Input	Tag of array.
UF_ASSEM_iset_array_data_p_t	**array_data**	Output to UF_*free*	Pointer to data structure which is to be filled with array parameters. Use UF_free to deallocate memory.
tag_t * *	**components**	Output to UF_*free*	Tags of components. Use UF_free to deallocate memory.

UF_ASSEM_ask_last_filter (view source)

Defined in: uf_assem.h

Overview

UF_ASSEM_ask_last_filter

Returns the last used filter stored for the given part tag. The output is either a filter tag or a cset tag, but could never be both at the same time. Both output

arguments may return a null tag if there is no filter or component set matching the stored filter name in the given part file.

If a filter tag is returned, you may use UF_FLTR_evaluate_filter for each of the part occurrences in your assembly to check whether it needs to be loaded or not. If a cset tag is returned, you may use UF_PART_open_cset to open that component set directly.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

History
Initially released in NX2

Required License(s)
gateway

```
int UF_ASSEM_ask_last_filter
(
    tag_t part,
    tag_t * filter_tag,
    tag_t * cset_tag
)
```

tag_t	**part**	Input	Tag of the input part
tag_t *	**filter_tag**	Output	Tag of last filter used to open the part
tag_t *	**cset_tag**	Output	Tag of last cset used to open the part

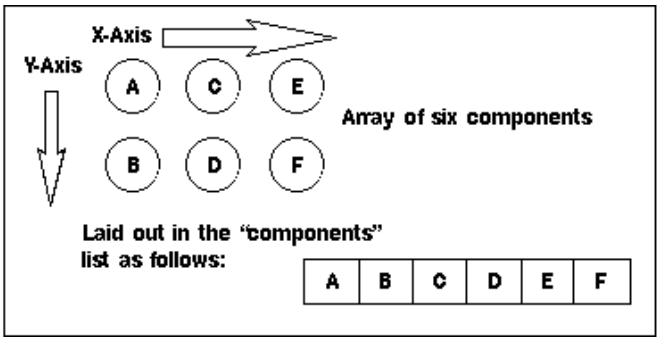
---

UF_ASSEM_ask_mc_array_data (view source)

Defined in: uf_assem.h

Overview
Returns information about a Master Component array in the "array_data" structure and a dynamically allocated array of tags.

For 2D linear arrays, the component list is laid out as follows:



Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_mc_array_data
(
    tag_t array,
    UF_ASSEM_mc_array_data_p_t array_data,
    tag_t * * components
)
```

tag_t	**array**	Input	Tag of array.
UF_ASSEM_mc_array_data_p_t	**array_data**	Output	Pointer to data structure which is to be filled with array parameters. The caller is responsible for freeing the array_name member by calling UF_free.
tag_t * *	**components**	Output to UF_*free*	Tags of components. Use UF_free to deallocate memory.

UF_ASSEM_ask_name_of_arrangement (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_name_of_arrangement

Returns the name of the given Assembly Arrangement.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History
Initially released in NX2.

Required License(s)
gateway

```
int UF_ASSEM_ask_name_of_arrangement
(
    tag_t arrangement,
    char * * name
)
```

tag_t	**arrangement**	Input	The Assembly Arrangement to query
char * *	**name**	Output to UF_*free*	The name of the given Assembly Arrangement

UF_ASSEM_ask_occs_of_entity (view source)

Defined in: uf_assem.h

Overview
Asks for all of the occurrences of object in all part occurrences in the session.

Return
Returns the count of the object occurrences.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
int UF_ASSEM_ask_occs_of_entity
(
    tag_t object,
    tag_t * * occurrences
)
```

tag_t	**object**	Input	Tag of object to query for occurrences
tag_t * *	**occurrences**	Output to UF_*free*	Array of tags of occurrences. The occurrences array must be freed with UF_free when it is no longer needed.

UF_ASSEM_ask_occs_of_part (view source)

Defined in: uf_assem.h

Overview
Asks for tags of all part occurrences of "part" under the assembly of "parent_part."

Return
Returns the count of part occurrences.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
int UF_ASSEM_ask_occs_of_part
(
    tag_t parent_part,
    tag_t part,
    tag_t * * part_occs
```

)

tag_t	**parent_part**	Input	Tag of parent part to query. If parent_part is NULL, uses the current displayed part.
tag_t	**part**	Input	Tag of part with occurrences
tag_t * *	**part_occs**	Output to UF_*free*	Array of part occurrence tags. The part_occs array must be freed with UF_free when it is no longer needed.

UF_ASSEM_ask_orientation (view source)

Defined in: uf_assem.h

Overview
Returns the origin and orientation of a reference set.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_ASSEM_ask_orientation
(
    tag_t ref_set_tag,
    double origin [ 3 ] ,
    double orientation [ 9 ]
)
```

tag_t	**ref_set_tag**	Input	Reference set tag whose orientation and origin are wanted.
double	**origin [3]**	Output	Array to store the origin of the reference set in.
double	**orientation [9]**	Output	Array to store the reference set's orientation in.

UF_ASSEM_ask_parent_component (view source)

Defined in: uf_assem.h

Overview
Return the tag of the part_occurrence, or v9 component, containing the occurrence. The occurrence can be either an entity occurrence, v9 component or a part occurrence.

Environment
Internal and External

See Also
example

Required License(s)
gateway

```
int UF_ASSEM_ask_parent_component
(
    tag_t occur,
    tag_t * parent
)
```

tag_t	**occur**	Input	Occurrence object to be accessed
tag_t *	**parent**	Output	Parent object returned (or NULL_TAG if there is not a parent object)

UF_ASSEM_ask_parent_of_instance (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the parent part of an instance.

Return
Parent part tag

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example and refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_parent_of_instance
(
    tag_t instance
)
```

tag_t	**instance**	Input	Tag of instance to query for parent

UF_ASSEM_ask_part_name_of_child (view source)

Defined in: uf_assem.h

Overview
Returns the part name of the child part of an instance. If the child part is loaded, this function is equivalent to UF_PART_ask_part_name. If the child part is unloaded, then this function returns the filename of the child part using the current search options. This file name may then be given to UF_PART_open, or UF_PART_open_quiet, to open the child part. Alternatively, you can use the function UF_ASSEM_ensure_child_loaded to load child part in the context of the current assembly.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_part_name_of_child
(
    tag_t instance,
    char part_fspec [ MAX_FSPEC_BUFSIZE ]
)
```

tag_t	**instance**	Input	Tag of instance to query for child part
char	**part_fspec [MAX_FSPEC_BUFSIZE]**	Output	Part name of child part of instance

---

UF_ASSEM_ask_part_occ_children (view source)

Defined in: uf_assem.h

Overview
Asks for a list of part occurrences under a given part occurrence.

Return
Returns the count of part occurrences in the list.

Environment
Internal and External. Please note that this routine will not return part occurrence children that are suppressed. Suppressed part occurrences are, by default, not visible in the assembly. To get a list of all part occurrence children, including suppressed ones, please use UF_ASSEM_ask_all_part_occ_children.

See Also
The following table of parameter values for this function is based on the automobile example
Refer to table

Required License(s)
gateway

```
int UF_ASSEM_ask_part_occ_children
(
    tag_t part_occur,
    tag_t * * child_part_occs
)
```

tag_t	**part_occur**	Input	Tag of part occurrence to query for children
tag_t * *	**child_part_occs**	Output to UF_*free*	Array of tags of child part occurrences. The allocated list of tags must be freed with UF_free when it is no longer needed.

UF_ASSEM_ask_part_occ_of_inst (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the part occurrence corresponding to the instance and the parent part occurrence. A NULL_TAG passed as the "parent_part_occ" specifies the displayed part.

Return

Returns tag of part occurrence

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_part_occ_of_inst
(
    tag_t parent_part_occ,
    tag_t instance
)
```

tag_t	**parent_part_occ**	Input	Tag of parent part occurrence
tag_t	**instance**	Input	Tag of instance

UF_ASSEM_ask_part_occ_suppress_state (view source)

Defined in: uf_assem.h

Overview
Determines if the given part occurrence is suppressed, or if one of the part occurrences that contains the given part occurrence is suppressed. Part occurrences that are suppressed, or that have one of their parents suppressed, are, by default, not visible in the assembly.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_part_occ_suppress_state
(
    tag_t part_occ,
    logical * suppressed,
    logical * parent_suppressed
)
```

tag_t	**part_occ**	Input	The tag of the part occurrence to query.
logical *	**suppressed**	Output	True if the given part occurrence is suppressed.
logical *	**parent_suppressed**	Output	True if any parent part occurrence of the given part occurrence is suppressed. May be NULL if the caller is not interested in determining the suppress state of the part occurrence

---

UF_ASSEM_ask_part_occs_of_inst (view source)

Defined in: uf_assem.h

Overview
Asks for all of the part occurrences associated with an instance.

Return
Returns the count of part occurrences. When a -1 is returned then the tag is not valid or another error occurred. Zero means there were no occurrences of the instance.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
int UF_ASSEM_ask_part_occs_of_inst
(
    tag_t instance,
    tag_t * * part_occs
)
```

tag_t	**instance**	Input	Tag of instance to query for part occurrences
tag_t * *	**part_occs**	Output to UF_*free*	Allocated array of tags of part occurrences. The part_occs array must be freed with UF_free when it is no longer needed.

UF_ASSEM_ask_part_occurrence (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the part_occurrence containing the "occurrence." "occurrence" can be either an object occurrence or a part occurrence.

Return
Returns tag of part occurrence.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_part_occurrence
(
    tag_t occurrence
)
```

tag_t	**occurrence**	Input	Tag of object/part occurrence

UF_ASSEM_ask_part_occurrence_of_step (view source)

Defined in: uf_assem.h

Overview
Returns the part occurrences impacted by given sequence step

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_part_occurrence_of_step
(
    tag_t step,
    int * num_part_occs,
    tag_t * * part_occs
)
```

tag_t	**step**	Input	The step to be queried.
int *	**num_part_occs**	Output	Number of part occurrences referenced by the step.
tag_t * *	**part_occs**	Output to UF_*free*	The part occurrences referenced by the step. The array must be UF_free'd

UF_ASSEM_ask_prototype_of_occ (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the prototype object if "occurrence" is an object occurrence tag. Returns the part tag of the part if occurrence is a part occurrence tag. Returns NULL_TAG if the prototype object of the occurrence is unloaded. Returns NULL_TAG if a call is made with a part occurrence of a part which is not loaded.

Before a body/face/edge is promoted, its occurrence points to the prototype in the component. After a body/face/edge is promoted, a new prototype is created and the occurrence is changed so that it points to the new prototype (promoted prototype). The promoted prototype and the promoted body are one and the same. Therefore, if you query a promoted body for its prototype with UF_ASSEM_ask_prototype_of_occ, it returns the tag of the promoted body.

You can go from a base body/face/edge to the corresponding promoted body/face/edge using UF_MODL_prom_map_object_up. Similarly, you can go from a promoted body/face/edge to the corresponding base body/face/edge using UF_MODL_prom_map_object_down. Both routines work with prototypes only.

Return
Returns the tag of the prototype

Environment
Internal and External

See Also
UF_MODL_prom_map_object_up
UF_MODL_prom_map_object_down

The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

tag_t UF_ASSEM_ask_prototype_of_occ
(
tag_t occurrence
)

tag_t	**occurrence**	Input	Tag of part occurrence or object occurrence

---

UF_ASSEM_ask_ref_set_data (view source)

Defined in: uf_assem.h

Overview
Returns all the information associated with a reference set. The array of reference set members requires freeing by the caller.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

int UF_ASSEM_ask_ref_set_data
(
tag_t ref_set,
char ref_set_name [UF_OBJ_NAME_BUFSIZE] ,
double origin [3] ,
double matrix [9] ,
int * num_members,
tag_p_t * members
)

tag_t	**ref_set**	Input	Reference set tag whose data is required.
char	**ref_set_name [UF_OBJ_NAME_BUFSIZE]**	Output	The reference set's name.
double	**origin [3]**	Output	Array containing the reference set's origin.
double	**matrix [9]**	Output	Array containing the reference set's orientation.
int *	**num_members**	Output	Number of elements in the reference set.
tag_p_t *	**members**	Output to UF_*free*	Array of elements in the reference set. Use UF_free to deallocate memory when done.

UF_ASSEM_ask_ref_set_members (view source)

Defined in: uf_assem.h

Overview
Returns an allocated array of the members of a reference set.

Starting with NX7.5, this function now returns the lightweight facets along with the solids for the reference set.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_ASSEM_ask_ref_set_members
(
    tag_t ref_set,
    int * ret_count,
    tag_p_t * members
)
```

tag_t	**ref_set**	Input	Reference set tag whose members are wanted.
int *	**ret_count**	Output	The number of members of the reference set.
tag_p_t *	**members**	Output to UF_*free*	Array of the members of the reference set. Use UF_free to deallocate memory when done.

UF_ASSEM_ask_ref_sets (view source)

Defined in: uf_assem.h

Overview
Returns all the reference sets to which the specified reference set member belongs. This array must be freed by the caller.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_ASSEM_ask_ref_sets
(
    tag_t ref_set_member,
    int * num_ref_sets,
    tag_p_t * ref_sets
)
```

tag_t	**ref_set_member**	Input	Tag whose reference sets are wanted.

int *	**num_ref_sets**	Output	The number of reference sets that member is in.
tag_p_t *	**ref_sets**	Output to UF_*free*	Array of the reference sets to which the member belongs. Use UF_free to deallocate memory.

UF_ASSEM_ask_root_part_occ (view source)

Defined in: uf_assem.h

Overview
Returns the tag of the root part occurrence, given the tag of a part. This part occurrence is the top of the part occurrence tree for the part. You can use this function with UF_ASSEM_ask_part_occ_children to traverse the part occurrence tree for a part.

Return
Tag of root part occurrence. Returns a NULL_TAG if there is no root part occurrence (e.g. part with no components).

Environment
Internal and External

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_root_part_occ
(
    tag_t part
)
```

tag_t	**part**	Input	Tag of part

UF_ASSEM_ask_save_trueshape (view source)

Defined in: uf_assem.h

Overview
Returns a logical indicating whether the saving of true shape data is currently enabled.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_save_trueshape
(
    logical * save_trueshape_data
)
```

logical *	**save_trueshape_data**	Output	Logical indicating whether the saving of true shape data is currently enabled. (TRUE = save true shape data, FALSE = do not save true shape data)

UF_ASSEM_ask_search_directories (view source)

Defined in: uf_assem.h

Overview
Ask for the list of the search directories.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_search_directories
(
    int * count,
    char * * * dir_list,
    logical * * sub_dir
)
```

int *	**count**	Output	The number of directories in the list.
char * * *	**dir_list**	Output to UF_*free*	A pointer to an allocated array of "count" pointers to allocated character strings containing the directory names. The array of character strings must be freed with UF_free_string_array.
logical * *	**sub_dir**	Output to UF_*free*	A pointer to an allocated array of "count" logicals. Each logical is TRUE if the subdirectories of the corresponding directory should be searched, or FALSE if the subdirectories should not be searched. The array must be freed with UF_free.

UF_ASSEM_ask_sequence_description (view source)

Defined in: uf_assem.h

Overview
Asks the description assigned to the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_sequence_description
(
    tag_t sequence,
    char * * desc
)
```

tag_t	**sequence**	Input	The tag of the given sequence
char * *	**desc**	Output to UF_*free*	The description of the sequence, null terminated. The returned string must be UF_free'd

UF_ASSEM_ask_sequence_duration (view source)

Defined in: uf_assem.h

Overview
Asks the total duration in playback frames for an entire sequence. The duration is defined as (last frame number) - (first frame number), so a sequence whose duration is N has frame numbers in the inclusive range 0 to N.

Environment
Internal and External

History
Initially released in NX 8.0.1

Required License(s)
assemblies

```
int UF_ASSEM_ask_sequence_duration
(
    tag_t sequence,
    int * duration
)
```

tag_t	**sequence**	Input	The tag of the sequence
int *	**duration**	Output	The duration, in playback frames, of the sequence.

UF_ASSEM_ask_sequence_name (view source)

Defined in: uf_assem.h

Overview
Asks the name of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_sequence_name
(
    tag_t sequence,
    char * * name
)
```

tag_t	**sequence**	Input	The tag of the given sequence
char * *	**name**	Output to UF_*free*	The name of the sequence, null terminated. The returned string must be UF_free'd

UF_ASSEM_ask_sequence_type (view source)

Defined in: uf_assem.h

Overview
Returns the type of sequence.

Environment
Internal and External

History
Initially released in NX 3.0

Required License(s)
gateway

```
int UF_ASSEM_ask_sequence_type
(
    tag_t sequence,
    int * seq_type
)
```

tag_t	**sequence**	Input	The tag of the sequence
int *	**seq_type**	Output	The sequence type. One of the constants UF_ASSEM_ASSEMBLE_SEQUENCE UF_ASSEM_DISASSEMBLE_SEQUENCE UF_ASSEM_OPERATIONAL_SEQUENCE UF_ASSEM_UNKNOWN_SEQUENCE_TYPE

UF_ASSEM_ask_sequences_in_part (view source)

Defined in: uf_assem.h

Overview
Asks the sequences in the given part.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_sequences_in_part
(
    tag_t part,
    int * num_sequences,
    tag_t * * sequences
)
```

tag_t	**part**	Input	The tag of the given part
int *	**num_sequences**	Output	Number of sequences

tag_t * *	**sequences**	Output to UF_*free*	The tags of the sequences. The user must free this array using UF_free when done using the data.

UF_ASSEM_ask_stable_id_of_instance (view source)

Defined in: uf_assem.h

Overview

Returns the stable id of the given instance.

If the instance has a stable id then this routine will return a character pointer to an allocated copy of the stable id otherwise it will return NULL.

The stable id is a stable revision independent identifier for the parent child relationships between assemblies and their components.

This function is primarily for use in managed mode (TCIN) where the stable id is assigned when the instance is first created.

The character pointer should be freed by the user by calling UF_free.

Environment

Internal and External

Required License(s)

gateway

```
int UF_ASSEM_ask_stable_id_of_instance
(
    tag_t instance_tag,
    char * * stable_id
)
```

tag_t	**instance_tag**	Input	Tag of the instance.
char * *	**stable_id**	Output to UF_*free*	Stable Id of the instance. This pointer should be freed by the user by calling UF_free.

UF_ASSEM_ask_step_duration (view source)

Defined in: uf_assem.h

Overview

Asks the duration in playback frames of the given step. The duration is defined as (last frame number) - (first frame number).

Example: if frame 6 represents the positions of the components just before a motion step, and frame 15 represents their positions just after the step, the duration of the step is 9.

Environment

Internal and External

History

Initially released in NX 8.0.1

Required License(s)

assemblies

```
int UF_ASSEM_ask_step_duration
(
    tag_t step,
    int * duration
)
```

tag_t	**step**	Input	The tag of the step
int *	**duration**	Output	The duration, in frames, of the step.

UF_ASSEM_ask_step_element_durations (view source)

Defined in: uf_assem.h

Overview
Asks the durations, measured in playback frames, of the step elements making up a given step. The returned array is NULL if the step has no elements.

Environment
Internal and External

History
Initially released in NX 8.0.1

Required License(s)
assemblies

```
int UF_ASSEM_ask_step_element_durations
(
    tag_t step,
    int * num_durations,
    int * * durations
)
```

tag_t	**step**	Input	The tag of the step
int *	**num_durations**	Output	Number of returned durations
int * *	**durations**	Output to UF_*free*	The durations of the step elements. The user must free this array using UF_free when done using the data.

UF_ASSEM_ask_step_increment (view source)

Defined in: uf_assem.h

Overview
Asks the step increment of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)

gateway

int UF_ASSEM_ask_step_increment
(
tag_t sequence,
int * increment
)

tag_t	**sequence**	Input	The tag of the given sequence
int *	**increment**	Output	The step increment for the sequence

---

UF_ASSEM_ask_step_number (view source)

Defined in: uf_assem.h

Overview
Asks a step number for the given step. The step number indicates the position of the step in the sequence, and is automatically generated and maintained by the sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

int UF_ASSEM_ask_step_number
(
tag_t step,
int * ask_step_number
)

tag_t	**step**	Input	The tag of the step
int *	**ask_step_number**	Output	Step number of the given step

---

UF_ASSEM_ask_step_type (view source)

Defined in: uf_assem.h

Overview
Asks the type of the given step.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_step_type
(
    tag_t step,
    int * step_type
)
```

tag_t	**step**	Input	The tag of the step
int *	**step_type**	Output	The step type. One of the constants UF_ASSEM_ASSEMBLE_STEP UF_ASSEM_DISASSEMBLE_STEP UF_ASSEM_CAMERA_STEP UF_ASSEM_MOTION_STEP UF_ASSEM_SNAPSHOT_STEP

UF_ASSEM_ask_steps (view source)

Defined in: uf_assem.h

Overview
Asks the steps of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_steps
(
    tag_t sequence,
    int * num_steps,
    tag_t * * steps
)
```

tag_t	**sequence**	Input	The tag of the sequence
int *	**num_steps**	Output	Number of steps
tag_t * *	**steps**	Output to UF_*free*	The tags of the steps. The user must free this array using UF_free when done using the data.

UF_ASSEM_ask_steps_of_part_occ (view source)

Defined in: uf_assem.h

Overview
Returns the steps in the given sequence that reference the specified part occurrence

Environment
Internal and External

History

Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_steps_of_part_occ
(
    tag_t sequence,
    tag_t part_occ,
    int * num_steps,
    tag_t * * steps
)
```

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	The part occurrence whose steps are required
int *	**num_steps**	Output	Number of steps that reference the part occurrence
tag_t * *	**steps**	Output to UF_*free*	The steps that reference the given part occurrence. The array must be UF_free'd

---

UF_ASSEM_ask_suppress_state (view source)

Defined in: uf_assem.h

Overview
Determines if the given instance is suppressed.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_suppress_state
(
    tag_t instance,
    logical * suppressed
)
```

tag_t	**instance**	Input	The tag of the instance to query.
logical *	**suppressed**	Output	True if the given instance is suppressed.

UF_ASSEM_ask_suppression_exp (view source)

Defined in: uf_assem.h

Overview
Returns the expression a given instance is dependent on for suppress state. This will return NULL_TAG if the given instance has no suppression expression.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ask_suppression_exp
(
    tag_t instance,
    tag_t * exp
)
```

tag_t	**instance**	Input	The tag of the instance to get suppression expression for.
tag_t *	**exp**	Output	Expression used to control suppression for the given instance.

UF_ASSEM_ask_time_of_sequence (view source)

Defined in: uf_assem.h

Overview
Asks the time for the given sequence. The time for the sequence
is an aggregate of the time for each of the steps in the assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_time_of_sequence
(
    tag_t sequence,
    double * time
)
```

tag_t	**sequence**	Input	The tag of the given sequence
double *	**time**	Output	The time taken for the sequence

UF_ASSEM_ask_time_of_step (view source)

Defined in: uf_assem.h

Overview
Asks the time taken for the given step.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

int UF_ASSEM_ask_time_of_step
(
tag_t step,
double * time
)

tag_t	**step**	Input	The tag of the given step
double *	**time**	Output	The time taken for the step

UF_ASSEM_ask_transform_of_occ (view source)

Defined in: uf_assem.h

Overview
Asks the 4x4 transform matrix of an object occurrence or part occurrence.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_transform_of_occ
(
tag_t occurrence,
double transform [4] [4]
)

tag_t	**occurrence**	Input	Tag of occurrence to query for transform matrix.
double	**transform [4] [4]**	Output	Transformation matrix for occurrence

---

UF_ASSEM_ask_type_of_array (view source)

Defined in: uf_assem.h

Overview
Returns the type of an array. The system uses one of the defined constants in UF_ASSEM_MC_ARRAY or UF_ASSEM_ISET_ARRAY.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_type_of_array
(
tag_t array,
int * type
)

tag_t	**array**	Input	Tag of array being queried.
int *	**type**	Output	Type of array.

UF_ASSEM_ask_unprocessed_partoccs (view source)

Defined in: uf_assem.h

Overview
Given an assembly sequence tag, this function will return the components that have not yet been processed in the assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_ask_unprocessed_partoccs
(
    tag_t sequence,
    int * num_unprocessed_partoccs,
    tag_t * * unprocessed_partoccs
)
```

tag_t	**sequence**	Input	The tag of the given sequence
int *	**num_unprocessed_partoccs**	Output	Number of unprocessed part occurrences
tag_t * *	**unprocessed_partoccs**	Output to UF_*free*	Tags of unprocessed part occurrences. The array must be freed using UF_free

---

UF_ASSEM_ask_used_arrangement (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_ask_used_arrangement

Returns the Used Assembly Arrangement for a given component. This is the Assembly Arrangement which is used to represent that component in the parent part of that component.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_set_used_arrangement

UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History
Initially released in NX2.

Required License(s)
gateway

int UF_ASSEM_ask_used_arrangement
(
tag_t component,
tag_t * arrangement
)

tag_t	**component**	Input	The component to query
tag_t *	**arrangement**	Output	The Used Assembly Arrangement for that component

---

UF_ASSEM_ask_view_explosion (view source)

Defined in: uf_assem.h

Overview
Get an explosion used in a view. Outputs the tag of the explosion which is displayed in the view, or a NULL_TAG if the view is displaying the real assembly.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_ask_view_explosion
(
tag_t view,
tag_t * explosion
)

tag_t	**view**	Input	The tag of the view about which to enquire
tag_t *	**explosion**	Output	The explosion used in the view, or NULL_TAG if the view is showing the real assembly positions (has no explosion associated with it).

UF_ASSEM_ask_work_occurrence (view source)

Defined in: uf_assem.h

Overview
Returns the work occurrence of the work part.

Return

Returns the tag of the work occurrence. Returns a NULL_TAG if there is no current work part, or if the work part occurrence is the displayed part.

Environment
Internal and External

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_work_occurrence
(
    void
)
```

UF_ASSEM_ask_work_part (view source)

Defined in: uf_assem.h

Overview
Returns the part tag of the current work part.

Return
Returns the tag of the current work part, else returns a NULL_TAG if there is no current work part.

Environment
Internal and External

Required License(s)
gateway

```
tag_t UF_ASSEM_ask_work_part
(
    void
)
```

UF_ASSEM_capture_arrangement_from_current_sequence (view source)

Defined in: uf_assem.h

Overview
Captures an arrangement of the current sequence playback positions and assemble/disassemble states. The disassemble states will be represented as part occurrence specific suppressions.

Any positioning or suppression will be done as assembly level overrides (where the assembly is the part owning the sequence).

There are situations where an arrangement can be captured, but with some warnings. The warnings will be returned in the list of warnings. The warnings are standard NX error codes. Their description can be obtained with UF_get_fail_message. The nature of the warning may indicate that not all positions could be represented in the arrangement (usually a consequence of mating conditions and assembly constraints that are active in the assembly while mating conditions and assembly constraints had been disabled in the sequence). Similarly, it may not always be possible to suppress the required components on the assembly level since in sequencing you can "disassemble" a subassembly's geometry without "disassembling" all its child components.

Sequencing must have initialized before this function can be called.

Environment
Internal and External

See Also
UF_ASSEM_capture_arrangement_from_current_sequence_extended
UF_ASSEM_initialize_sequencing

History
Initially released in NX5.0

Required License(s)
gateway

```
int UF_ASSEM_capture_arrangement_from_current_sequence
(
    const char * arrangement_name,
    tag_t* arrangement,
    int* warnings_count,
    int* * warnings
)
```

const char *	**arrangement_name**	Input	The name of the arrangement
tag_t*	**arrangement**	Output	The created arrangement
int*	**warnings_count**	Output	The number of warnings
int* *	**warnings**	Output to UF_*free*	List of warning codes. Must be freed with UF_free

---

UF_ASSEM_capture_arrangement_from_current_sequence_extended (view source)

Defined in: uf_assem.h

Overview
Captures an arrangement of the current sequence playback positions and assemble/disassemble states. The disassemble states will be represented as part occurrence specific suppressions.

Any positioning or suppression will be done as assembly level overrides (where the assembly is the part owning the sequence).

There are situations where an arrangement can be captured, but with some warnings. The warnings will be returned in the list of warnings. The warnings are standard NX error codes. Their description can be obtained with UF_get_fail_message. The nature of the warning may indicate that not all positions could be represented in the arrangement (usually a consequence of mating constraints that are active in the assembly while mating constraints had been disabled in the sequence). Optionally, mating constraints may be ignored during arrangement capture if desired. Similarly, it may not always be possible to suppress the required components on the assembly level since in sequencing you can "disassemble" a subassemlby's geometry without "disassembling" all its child components.

Sequencing must have initialized before this function can be called.

Environment
Internal and External

See Also
UF_ASSEM_capture_arrangement_from_current_sequence
UF_ASSEM_initialize_sequencing

History
Initially released in NX8.5

Required License(s)
gateway

int UF_ASSEM_capture_arrangement_from_current_sequence_extended
(
const char * arrangement_name,
logical ignore_constraints,
tag_t* arrangement,
int* warnings_count,
int* * warnings
)

const char *	**arrangement_name**	Input	The name of the arrangement
logical	**ignore_constraints**	Input	Whether to ignore assembly constraints
tag_t*	**arrangement**	Output	The created arrangement
int*	**warnings_count**	Output	The number of warnings
int* *	**warnings**	Output to UF_*free*	List of warning codes. Must be freed with UF_free

---

UF_ASSEM_check_array_status (view source)

Defined in: uf_assem.h

Overview
Performs an update validity check on a component array. If the array can be updated, then this function returns 0. If the array check fails, then this function returns an error code. Use UF_get_fail_message to capture the error string. Some possible errors that can occur are as follows:
"Array is suppressed."
"Array template is unloaded."
"Array master is unloaded."
"Array template is not mated to a valid feature instance."
"Array size expression is invalid."

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_check_array_status
(
tag_t array
)

tag_t	**array**	Input	Tag of array to check.

UF_ASSEM_convert_prev16_aligns (view source)

Defined in: uf_assem.h

Overview

UF_ASSEM_convert_prev16_aligns

Converts the specified PreV16 Align constraints to Distance constraints. This function will attempt to adjust the offset distance of the constraint to maintain the existing behaviour of the constraint - in some cases it might negate the offset expression. A list of messages is returned which describes any reasons why constraints could not be converted or could not update afterwards.

Return

Error code - use UF_get_fail_message to obtain the corresponding error message

Environment

This function is supported for both Internal and External execution.

See Also

UF_ASSEM_find_prev16_aligns_to_check
UF_ASSEM_free_prev16_aligns

History

Initially released in NX3.0.4 MP1.

Required License(s)

gateway

```
int UF_ASSEM_convert_prev16_aligns
(
    int n_aligns_to_convert,
    const UF_ASSEM_prev16_align_t * aligns_to_convert,
    int * n_messages,
    UF_ASSEM_prev16_align_t * * messages
)
```

int	**n_aligns_to_convert**	Input	The number of constraints to convert
const UF_ASSEM_prev16_align_t *	**aligns_to_convert**	Input	The constraints to convert (array of length n_aligns_to_convert)
int *	**n_messages**	Output	The number of messages returned.
UF_ASSEM_prev16_align_t * *	**messages**	Output to UF_*free*	The returned messages (of length n_messages) Freed by UF_ASSEM_free_prev16_aligns

---

UF_ASSEM_copy_explosion (view source)

Defined in: uf_assem.h

Overview

Copies the explosion transforms from one (sub)assembly to an equivalent (sub)assembly. The destination may be in the same explosion as the source, or it may be another explosion in the same part, or it may be an explosion in a different part altogether.

Environment

Internal and External

Required License(s)
assemblies

int UF_ASSEM_copy_explosion
(
tag_t source_explosion,
tag_t source_component,
tag_t destination_explosion,
tag_t destination_component
)

tag_t	**source_explosion**	Input	The tag of the explosion from which to copy transforms.
tag_t	**source_component**	Input	Either NULL_TAG, in which case the whole explosion is copied, or a component (part occurrence) tag indicating the portion of the explosion to be copied.
tag_t	**destination_explosion**	Input	The tag of the explosion which the source explosion should be copied into.
tag_t	**destination_component**	Input	Either NULL_TAG, in which case the whole explosion is over-written, or a component (part occurrence) tag indicating a portion of the explosion to be over-written (in either case, the tag must represent the same componenrt part as the source_component argument).

UF_ASSEM_count_ents_in_part_occ (view source)

Defined in: uf_assem.h

Overview
Count the number of entity occurrences associated with a part occurrence.

Return
Number of object occurrences

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

int UF_ASSEM_count_ents_in_part_occ
(
tag_t part_occur
)

tag_t	**part_occur**	Input	Tag of part occurrence

UF_ASSEM_count_objs_in_comp (view source)

Defined in: uf_assem.h

Overview
Given a part occurrence tag or v9 component type as input argument this routine gets the count of members for a given component.

Environment
Internal and External

See Also
example

Required License(s)
gateway

```
int UF_ASSEM_count_objs_in_comp
(
    tag_t comp_tag,
    int * returned_count
)
```

tag_t	**comp_tag**	Input	Source component
int *	**returned_count**	Output	Count of members in source component

---

UF_ASSEM_count_ref_sets_in (view source)

Defined in: uf_assem.h

Overview
Counts the number of links from an object to the reference sets that contain it.

Return
The number of links from the object to the reference sets that contain it. This value is >= 0.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_count_ref_sets_in
(
    tag_t object
)
```

tag_t	**object**	Input	Object Identifier (can be an occurrence)

UF_ASSEM_create_assembly_cut (view source)

Defined in: uf_assem.h

Overview

UF_ASSEM_create_assembly_cut

Create an assembly cut feature in a part.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

History
Initially released in NX3.

Required License(s)
assemblies

```
int UF_ASSEM_create_assembly_cut
(
    tag_t part,
    logical blank_tool_bodies,
    int n_target_body_occs,
    tag_t * target_body_occs,
    int n_tool_bodies,
    tag_t * tool_bodies,
    tag_t * acut_tag
)
```

tag_t	**part**	Input	The part in which the assembly cut is created
logical	**blank_tool_bodies**	Input	To keep tool bodies displayed after the cut
int	**n_target_body_occs**	Input	The number of body occurrences
tag_t *	**target_body_occs**	Input	The target body occurrences
int	**n_tool_bodies**	Input	The number of tools
tag_t *	**tool_bodies**	Input	The tool bodies or body occurrences
tag_t *	**acut_tag**	Output	The assembly cut feature tag

UF_ASSEM_create_component_part (view source)

Defined in: uf_assem.h

Overview
Create a new part, moves selected objects to it, then adds an instance of it to the parent part. Any other transferrable objects upon which the given objects depend are also moved into the component.

The instance will be added to the parts-list according to the setting of the global switch, see UF_ASSEM_ask_assem_options.

CAUTION: Object occurrences and part occurrences cannot be moved into a component.

If an object depends upon another object which is not transferable, then that object is not moved into the component and a return value of UF_PART_warn_objects_not_copied is returned. The operation continues though, and any other requested transferable objects are still moved into the component. This situation can arise because a drafting object is not transferred if its associated geometry is not

transferred.

If any other error occurs, the operation does not succeed and the appropriate error code is returned.

Environment
Internal and External

See Also
UF_ASSEM_ask_assem_options
Refer to example

History
V11.0 modified to retain feature parameters in the transferred objects.

Required License(s)
assemblies

```
int UF_ASSEM_create_component_part
(
    tag_t parent_part,
    const char * new_part_name,
    const char * refset_name,
    const char * instance_name,
    int units,
    int layer,
    double origin [ 3 ] ,
    double csys_matrix [ 6 ] ,
    int n_objects,
    tag_t * objects,
    tag_t * instance
)
```

tag_t	**parent_part**	Input	Tag of parent part
const char *	**new_part_name**	Input	Name of new component part
const char *	**refset_name**	Input	Name of reference set; must be no longer than UF_OBJ_NAME_NCHARS characters.
const char *	**instance_name**	Input	Name of instance to add to parent part; must be no longer than UF_OBJ_NAME_NCHARS characters.
int	**units**	Input	1 = MM 2 = Inches
int	**layer**	Input	-1 = original 0 = use work layer 1-255 = use specified layer
double	**origin [3]**	Input	Position in parent part where the instance is to be created.
double	**csys_matrix [6]**	Input	Orientation of the instance
int	**n_objects**	Input	Number of objects in the "objects" array.
tag_t *	**objects**	Input	Pointer to an array of tags of objects that should be moved to the new component part.
tag_t *	**instance**	Output	Tag of instance

UF_ASSEM_create_constrained_iset_array (view source)

Defined in: uf_assem.h

Overview
Creates an ISET array, based on the parameters in the "array_data" structure. The template component must already be constrained to one element of the ISET. This differs from UF_ASSEM_create_iset_array in that it works with assembly constraints rather than mating conditions.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_create_constrained_iset_array
(
UF_ASSEM_iset_array_data_p_t array_data,
tag_t * array
)

UF_ASSEM_iset_array_data_p_t	**array_data**	Input	Data structure containing array parameters.
tag_t *	**array**	Output	Tag of newly created array.

UF_ASSEM_create_cset (view source)

Defined in: uf_assem.h

Overview
Creates a component set in the given part with the given name which can not be NULL. The output of this routine is the tag of the newly created, empty component set.

Environment
Internal and External

See Also
See UF_ASSEM_add_to_cset to populate a component set.
Refer to example

Required License(s)
assemblies

int UF_ASSEM_create_cset
(
tag_t part,
const char * name,
tag_t * object
)

tag_t	**part**	Input	Part where component set is created
const char *	**name**	Input	Name to be given to component set
tag_t *	**object**	Output	Object identifier of created component set

---

UF_ASSEM_create_deformable_part (view source)

Defined in: uf_assem.h

Overview
Creates a deformable definition for the part occurrence given in the current work part.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
assemblies

```
int UF_ASSEM_create_deformable_part
(
    UF_ASSEM_deform_data_p_t data,
    tag_t * deformable_feature
)
```

UF_ASSEM_deform_data_p_t	**data**	Input	
tag_t *	**deformable_feature**	Output	

UF_ASSEM_create_explosion (view source)

Defined in: uf_assem.h

Overview
Creates an explosion in the specified display part; the components in the assembly are initially at their real assembly positions, and the explosion is initially not used in any views.

Environment
Internal and External

See Also
Refer to example

Required License(s)
assemblies

```
int UF_ASSEM_create_explosion
(
    tag_t display_part_tag,
    const char* explosion_name,
    tag_p_t explosion_tag
)
```

tag_t	**display_part_tag**	Input	The tag of the display part in which to create the explosion
const char*	**explosion_name**	Input	A name for the explosion; this must be a valid NX name value, as well as not being used by any existing explosion in the display part.
tag_p_t	**explosion_tag**	Output	The tag of the created explosion (or NULL_TAG on error)

UF_ASSEM_create_iset_array (view source)

Defined in: uf_assem.h

Overview
Creates an ISET array, based on the parameters in the "array_data" structure. The template component must already be mated to one element of the ISET.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_create_iset_array
(
UF_ASSEM_iset_array_data_p_t array_data,
tag_t * array
)

UF_ASSEM_iset_array_data_p_t	**array_data**	Input	Data structure containing array parameters.
tag_t *	**array**	Output	Tag of newly created array.

UF_ASSEM_create_mc_array (view source)

Defined in: uf_assem.h

Overview
Creates a Master Component array in the work part, based on the parameters in the "array_data" structure.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_create_mc_array
(
UF_ASSEM_mc_array_data_p_t array_data,
tag_t * array
)

UF_ASSEM_mc_array_data_p_t	**array_data**	Input	Data structure containing array parameters.
tag_t *	**array**	Output	Tag of newly created array.

UF_ASSEM_create_ref_set (view source)

Defined in: uf_assem.h

Overview
Creates a reference set at the coordinates specified, with the given name, orientation, and members.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
assemblies

```
int UF_ASSEM_create_ref_set
(
    const char * ref_set_name,
    double origin [ 3 ] ,
    double matrix [ 9 ] ,
    tag_t * ref_set_members,
    int num_members,
    tag_t * ref_set_tag
)
```

const char *	**ref_set_name**	Input	Name of the new reference set. Must be no longer than UF_OBJ_NAME_NCHARS characters.
double	**origin [3]**	Input	The origin coordinates of the reference set.
double	**matrix [9]**	Input	The orientation matrix of the reference set.
tag_t *	**ref_set_members**	Input	num_members Array of members of the new reference set.
int	**num_members**	Input	The number of members in the members array.
tag_t *	**ref_set_tag**	Output	The tag of the newly created reference set.

UF_ASSEM_create_sequence (view source)

Defined in: uf_assem.h

Overview
Creates an empty assembly sequence.
UF_ASSEM_initialize_sequencing must be called before calling this function.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_create_sequence
(
    char * name,
    tag_t part,
    tag_t * sequence
)
```

char *	**name**	Input	The name of the sequence
tag_t	**part**	Input	The part to create the sequence in
tag_t *	**sequence**	Output	The tag of the newly created sequence

---

UF_ASSEM_create_step (view source)

Defined in: uf_assem.h

Overview
Creates a step in the given sequence.

A part occurrence may have more than one step in the sequence, if it is to be assembled or disassembled more than once.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_create_step
(
    tag_t sequence,
    tag_t part_occurrence,
    int step_type,
    double time,
    double cost,
    char * description,
    tag_t insert_at_step,
    tag_t * step
)
```

tag_t	**sequence**	Input	The sequence in which the step needs to be created
tag_t	**part_occurrence**	Input	The part occurrence being sequenced
int	**step_type**	Input	The type of the step. One of the constants UF_ASSEM_ASSEMBLE_STEP or UF_ASSEM_DISASSEMBLE_STEP only
double	**time**	Input	The time of the step. The time is a user defined number.
double	**cost**	Input	The cost of the step. The time is a user defined number.
char *	**description**	Input	The description of the step. May be NULL, if the user does not want to assign a description at this point.
tag_t	**insert_at_step**	Input	The step that this step is to be inserted after. May be NULL_TAG, in which case the new step is added at the beginning of the sequence

tag_t *	**step**	Output	The tag of the newly created step. May be NULL_TAG if no step was created - an error code describing the reason is returned.

UF_ASSEM_create_typed_sequence (view source)

Defined in: uf_assem.h

Overview
Creates a typed sequence.

UF_ASSEM_initialize_sequencing must be called before calling this function.

Environment
Internal and External

History
Initially released in NX 3.0

Required License(s)
assemblies

```
int UF_ASSEM_create_typed_sequence
(
    const char * name,
    int sequence_type,
    tag_t part,
    tag_t * sequence
)
```

const char *	**name**	Input	The name of the sequence
int	**sequence_type**	Input	The type of sequence to be created. One of the constants UF_ASSEM_ASSEMBLE_SEQUENCE UF_ASSEM_DISASSEMBLE_SEQUENCE UF_ASSEM_OPERATIONAL_SEQUENCE
tag_t	**part**	Input	The part to create the sequence in
tag_t *	**sequence**	Output	The tag of the newly created sequence

---

UF_ASSEM_cycle_ents_in_part_occ (view source)

Defined in: uf_assem.h

Overview
Cycles the object occurrences in a part occurrence.

Return
Returns object occurrence tag. Returns NULL_TAG after last object cycles.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .

Refer to table

Required License(s)
gateway

tag_t UF_ASSEM_cycle_ents_in_part_occ
(
tag_t part_occur,
tag_t object_occur
)

tag_t	**part_occur**	Input	Tag of part occurrence
tag_t	**object_occur**	Input	Tag of object occurrence. Pass the previous object tag in to the routine (starting with NULL_TAG to initialize), and the routine cycles the next object tag.

UF_ASSEM_cycle_inst_of_part (view source)

Defined in: uf_assem.h

Overview
Cycles the instances in a part which is the parent of the instances.

Return
Returns the next instance tag. Returns a NULL_TAG after the last object cycles through.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

tag_t UF_ASSEM_cycle_inst_of_part
(
tag_t part,
tag_t instance
)

tag_t	**part**	Input	Tag of parent part to instances
tag_t	**instance**	Input	Pass the previous instance tag into the routine (starting with NULL_TAG to initialize), and the routine cycles to the next instance tag.

UF_ASSEM_cycle_objs_in_comp (view source)

Defined in: uf_assem.h

Overview

Cycles the occurrenced members in the source component. Start cycle with a NULL_TAG in member and cycle ends when NULL_TAG is returned in member.

Environment
Internal and External

See Also
Please see example

Required License(s)
gateway

```
int UF_ASSEM_cycle_objs_in_comp
(
    tag_t component,
    tag_t * member
)
```

tag_t	**component**	Input	Source component
tag_t *	**member**	Input / Output	Current member from source component

---

UF_ASSEM_deform_part (view source)

Defined in: uf_assem.h

Overview
Creates a deformed part occurrence in the current work part.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
assemblies

```
int UF_ASSEM_deform_part
(
    UF_ASSEM_deform_part_data_p_t deform_data,
    UF_ASSEM_deform_part_warnings_p_t deform_warnings
)
```

UF_ASSEM_deform_part_data_p_t	**deform_data**	Input / Output
UF_ASSEM_deform_part_warnings_p_t	**deform_warnings**	Output

UF_ASSEM_delete_array (view source)

Defined in: uf_assem.h

Overview
Deletes an array. Either removes all of the components except the array template or only removes the array and leaves all of the components. Components which are master components of other

arrays cannot be deleted.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_delete_array
(
tag_t array,
logical delete_all
)

tag_t	**array**	Input	Tag of array being deleted.
logical	**delete_all**	Input	Delete all flag: TRUE = Deletes all components in the array except the array template. FALSE = Deletes the array but not the components in the array.

UF_ASSEM_delete_explosion (view source)

Defined in: uf_assem.h

Overview
Deletes the specified explosion; the explosion may not be used in any views when this is done. All data concerning the explosion of individual components is lost.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_delete_explosion
(
tag_t explosion
)

tag_t	**explosion**	Input	The tag of the explosion to delete.

UF_ASSEM_delete_sequence (view source)

Defined in: uf_assem.h

Overview
Deletes an assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)

assemblies

int UF_ASSEM_delete_sequence
(
tag_t sequence
)

tag_t	**sequence**	Input	The tag of the given sequence

UF_ASSEM_delete_step (view source)

Defined in: uf_assem.h

Overview
Deletes a step from its assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

int UF_ASSEM_delete_step
(
tag_t step
)

tag_t	**step**	Input	The tag of the given step

UF_ASSEM_edit_assembly_cut (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_edit_assembly_cut

Edit an assembly cut feature in a part.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

History
Initially released in NX3.

Required License(s)
assemblies

```
int UF_ASSEM_edit_assembly_cut
(
    tag_t frec_tag,
    int n_target_bodies,
    tag_t * target_body_tags,
    int n_tool_bodies,
    tag_t * tool_body_tags,
    logical blank_tool_bodies
)
```

tag_t	**frec_tag**	Input	The assembly cut tag
int	**n_target_bodies**	Input	Number of target bodies
tag_t *	**target_body_tags**	Input	Target body occurrences
int	**n_tool_bodies**	Input	Number of tool bodies
tag_t *	**tool_body_tags**	Input	Tool body occurrences
logical	**blank_tool_bodies**	Input	Blank the display of the tool bodies

---

UF_ASSEM_edit_iset_array (view source)

Defined in: uf_assem.h

Overview

Edits a ISET Component array, by applying the parameters in the "array_data" structure.

CAUTION: Neither the dimensions nor the feature_iset tag can be edited. To change the size of the array, you should edit the feature ISET that the array is based on, by using UF_MODL_ask_circular_iset_parms or UF_MODL_ask_linear_iset_parms to find the expressions that control the ISET, and then edit these expressions.

Environment

Internal and External

See Also

UF_MODL_ask_circular_iset_parms
UF_MODL_ask_linear_iset_parms

Required License(s)

assemblies

```
int UF_ASSEM_edit_iset_array
(
    tag_t array,
    UF_ASSEM_iset_array_data_p_t array_data
)
```

tag_t	**array**	Input	Tag of array to be edited.
UF_ASSEM_iset_array_data_p_t	**array_data**	Input	Data structure containing new array parameters.

UF_ASSEM_edit_mc_array (view source)

Defined in: uf_assem.h

Overview
Edits a Master Component array, by applying the parameters in the "array_data" structure.

CAUTION: to edit the size and offset of the array, it is only necessary to edit the relevant expressions.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_edit_mc_array
(
    tag_t array,
    UF_ASSEM_mc_array_data_p_t array_data
)
```

tag_t	**array**	Input	Tag of array to be edited.
UF_ASSEM_mc_array_data_p_t	**array_data**	Input	Data structure containing new array parameters.

UF_ASSEM_edit_ref_set_data (view source)

Defined in: uf_assem.h

Overview
Modifies the specified reference set with a new origin and/or orientation. To change the name of the reference set, use the UF_OBJ_set_name routine.

Environment
Internal and External

See Also
UF_OBJ_set_name

History
Original release was in V14.0.

Required License(s)
assemblies

```
int UF_ASSEM_edit_ref_set_data
(
    tag_t ref_set_tag,
    double origin [ 3 ] ,
    double matrix [ 9 ]
)
```

tag_t	**ref_set_tag**	Input	Reference set tag whose orientation and origin are to be changed.
double	**origin [3]**	Input	Array of coordinates for the reference set's new origin.

double	**matrix [9]**	Input	Matrix Array for the reference set's new orientation.

UF_ASSEM_ensure_child_loaded (view source)

Defined in: uf_assem.h

Overview
Ensures that the child part of an instance is loaded. If the load of a child fails, then the load_status is filled out as it is for other load functions.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_ensure_child_loaded
(
    tag_t instance,
    UF_PART_load_status_t * load_status
)
```

tag_t	**instance**	Input	Tag of parent part to instances
UF_PART_load_status_t *	**load_status**	Output to UF_*free*	See the UF_PART_load_status_t structure definition. The allocated structure is filled with the names and associated error codes of any parts that did not load correctly. The allocated arrays must be freed with UF_free_string_array and UF_free.

UF_ASSEM_eval_instance_intent (view source)

Defined in: uf_assem.h

Overview
Evaluates intent data of an instance and optionally applies the result.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_eval_instance_intent
(
    tag_t * instance,
    logical apply_result,
    UF_ASSEM_instance_status_p_t instance_status
)
```

tag_t *	**instance**	Input / Output	Object identifier of the instance. Note that the instance tag is not currently modified; but it may be modified in the future (hence, its designation as an I/O parameter).

logical	**apply_result**	Input	Apply result flag.
UF_ASSEM_instance_status_p_t	**instance_status**	Output	Status of the instance.

UF_ASSEM_explode_component (view source)

Defined in: uf_assem.h

Overview
Explode a component by a given transform. This is combined with exploded position of the component's parent and the component's normal transform to give the resultant exploded position.

Environment
Internal and External

See Also
See UF_ASSEM_create_explosion.

Required License(s)
assemblies

```
int UF_ASSEM_explode_component
(
    tag_t explosion,
    tag_t part_occurrence,
    double transform [ 4 ] [ 4 ]
)
```

tag_t	**explosion**	Input	The explosion in which to modify the component
tag_t	**part_occurrence**	Input	the tag of a part occurrence not instance
double	**transform [4] [4]**	Input	a 4x4 non-distorting transform (see UF_ASSEM_ask_component_data for more information).

UF_ASSEM_find_immed_old_comps (view source)

Defined in: uf_assem.h

Overview
Ask for a list of immediate old style components. Only components of the given part are returned. This routine may be used to determine if the given part contains pre-V10 assemblies that may be upgraded (i.e., if n_immediate_components > 0 then the part contains pre-V10 components).

CAUTION: "Immediate" components are those direct components in the work part and does not include components of a subassembly of the work part. "Old" signifies that the components are pre-V10 style components.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_find_immed_old_comps
(
tag_t part,
tag_t * * immediate_components,
int * n_immediate_components
)

tag_t	**part**	Input	Tag of part to query for old components.
tag_t * *	**immediate_components**	Output to UF_*free*	Allocated array of tags of immediate old style components. This array must be freed by calling UF_free.
int *	**n_immediate_components**	Output	number of pre-v10 components

---

UF_ASSEM_find_occurrence (view source)

Defined in: uf_assem.h

Overview
Return the tag of the object occurrence in the given part, corresponding to the object_prototype.

Return
Returns tag of object occurrence or NULL_TAG if object is not found.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

tag_t UF_ASSEM_find_occurrence
(
tag_t part_occur,
tag_t object_prototype
)

tag_t	**part_occur**	Input	Tag of part occurrence
tag_t	**object_prototype**	Input	Tag of object prototype

UF_ASSEM_find_prev16_aligns_to_check (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_find_prev16_aligns_to_check

Finds PreV16 Align constraints which need converting to Distance constraints

so their behaviour will remain stable.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_convert_prev16_aligns
UF_ASSEM_free_prev16_aligns

History
Initially released in NX3.0.4 MP1.

Required License(s)
gateway

```
int UF_ASSEM_find_prev16_aligns_to_check
(
    tag_t part_tag,
    logical recurse,
    int * n_aligns_to_check,
    UF_ASSEM_prev16_align_t * * aligns_to_check
)
```

tag_t	**part_tag**	Input	The part in which to search
logical	**recurse**	Input	Whether to search in all components (true) or just in this part (false)
int *	**n_aligns_to_check**	Output	The number of found constraints
UF_ASSEM_prev16_align_t * *	**aligns_to_check**	Output to UF_*free*	The found constraints (array of length n_aligns_to_check) Freed by UF_ASSEM_free_prev16_aligns

---

UF_ASSEM_free_deform_warnings_data (view source)

Defined in: uf_assem.h

Overview
Free the deform part warnings structure

Environment
Internal and External

History
Initially released in NX2

Required License(s)
assemblies

```
int UF_ASSEM_free_deform_warnings_data
(
    UF_ASSEM_deform_part_warnings_p_t warnings
)
```

UF_ASSEM_deform_part_warnings_p_t	**warnings**	Input	

UF_ASSEM_free_instance_intent (view source)

Defined in: uf_assem.h

Overview
Frees intent data returned from UF_ASSEM_ask_instance_intent.

Environment
Internal and External

See Also
UF_ASSEM_ask_instance_intent

Required License(s)
assemblies

int UF_ASSEM_free_instance_intent
(
UF_ASSEM_instance_intent_p_t instance_intent
)

UF_ASSEM_instance_intent_p_t	**instance_intent**	Input	Data of the intent.

UF_ASSEM_free_prev16_aligns (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_free_prev16_aligns

Frees the memory consumed by the array of UF_ASSEM_prev16_align_t objects
returned by UF_ASSEM_find_prev16_aligns_to_check and UF_ASSEM_convert_prev16_aligns.
The memory for the array will be freed along with all data referenced by all
members of the array.

Return
Error code - use UF_get_fail_message to obtain the
corresponding error message

Environment
This function is supported for both Internal and
External execution.

See Also
UF_ASSEM_find_prev16_aligns_to_check
UF_ASSEM_convert_prev16_aligns

History
Initially released in NX3.0.4 MP1.

Required License(s)
gateway

int UF_ASSEM_free_prev16_aligns
(
int n_aligns,
UF_ASSEM_prev16_align_t * aligns
)

int	**n_aligns**	Input	The number of entries in the aligns array
UF_ASSEM_prev16_align_t *	**aligns**	Input	The array to be freed

---

UF_ASSEM_get_occ_in_work_occ (view source)

Defined in: uf_assem.h

Overview
Return the part occurrence tag in the context of the work occurrence

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_get_occ_in_work_occ
(
    tag_t part_occ,
    tag_t * occ_in_work
)
```

tag_t	**part_occ**	Input	
tag_t *	**occ_in_work**	Output	

UF_ASSEM_get_ref_set_inst (view source)

Defined in: uf_assem.h

Overview
Returns the object identifier of the reference set (specified by reference set "number") that contains the object. The object can be an occurrence.

Return
Object Identifier of the numbered reference set that contains the object. NULL_TAG is returned when the object does not belong to any reference sets or if it has less than number links.

Environment
Internal and External

Required License(s)
gateway

```
tag_t UF_ASSEM_get_ref_set_inst
(
    tag_t object,
    int number
)
```

tag_t	**object**	Input	Object Identifier (object can be an occurrence)

int	**number**	Input	The number of the reference set to which the object belongs (e.g. 1 is the first reference set it belongs to, 2 is the second reference set, etc.).

UF_ASSEM_hide_component (view source)

Defined in: uf_assem.h

Overview
Hide the given component in the given view. The effect is analagous to moving the component to a layer which is invisible in the view.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_hide_component
(
    tag_t component,
    tag_t view
)
```

tag_t	**component**	Input	the tag of a part occurrence not instance
tag_t	**view**	Input	The view from which to hide the component

---

UF_ASSEM_ignore_part_occ (view source)

Defined in: uf_assem.h

Overview
Ignores the given part occurrence in the sequence.

Ignored part occurrences are not marked processed, and are no longer considered valid candidates for assembly or disassembly.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_ignore_part_occ
(
    tag_t sequence,
    tag_t part_occ
)
```

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	The part occurrence to be ignored

UF_ASSEM_init_deform_part_data (view source)

Defined in: uf_assem.h

Overview
Initializes a deformed part data structure.

Environment
Internal and External

History
Initially released in NX2

Required License(s)
assemblies

```
int UF_ASSEM_init_deform_part_data
(
    UF_ASSEM_deform_part_data_p_t deform_part
)
```

UF_ASSEM_deform_part_data_p_t	**deform_part**	Input

UF_ASSEM_initialize_sequencing (view source)

Defined in: uf_assem.h

Overview
Initializes the sequencing environment.
Must be called before any sequencing playback type Open API function can be called.
Note that the user interface layout may change upon initializing the sequencing task environment.

Environment
Internal and External

History
Initially released in NX 3.0

Required License(s)
assemblies

```
int UF_ASSEM_initialize_sequencing
(
    void
)
```

UF_ASSEM_initialize_sequencing_keep_layout (view source)

Defined in: uf_assem.h

Overview

Initializes the sequencing environment.
Must be called before any sequencing playback type Open API function can be called.
User can keep their current layout when active a sequence.

Environment
Internal and External

History
Initially released in NX 8.5.1

Required License(s)
assemblies

int UF_ASSEM_initialize_sequencing_keep_layout
(
void
)

UF_ASSEM_is_component_ngc (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_is_component_ngc

Queries whether the input component is a non-geometric component (NGC)
or not.

Return
Returns TRUE if component_tag is a NGC, FALSE otherwise.

Environment
This function is supported for both Internal and
External execution.

History
Initially released in NX6.0.

Required License(s)
gateway

logical UF_ASSEM_is_component_ngc
(
tag_t component_tag
)

tag_t	**component_tag**	Input	the occurence or instance tag

UF_ASSEM_is_ignored (view source)

Defined in: uf_assem.h

Overview
Determines if the given part occurrence is marked to be ignored in the
given sequence

Environment

Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_is_ignored
(
    tag_t sequence,
    tag_t part_occ,
    logical * ignored
)
```

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	Tag of the part occurrence
logical *	**ignored**	Output	True if the part occurrence is ignored in the sequence, false otherwise.

UF_ASSEM_is_member_of_cset (view source)

Defined in: uf_assem.h

Overview
Allows you to lookup the given component in the given component set.

Environment
Internal and External

See Also
UF_ASSEM_ask_all_comp_cset
UF_ASSEM_apply_to_cset
Refer to example

Required License(s)
gateway

```
int UF_ASSEM_is_member_of_cset
(
    tag_t cset,
    tag_t component,
    logical * result
)
```

tag_t	**cset**	Input	Object identifier of a component set
tag_t	**component**	Input	Object identifier of component to be looked up
logical *	**result**	Output	TRUE if component found, otherwise FALSE

UF_ASSEM_is_occurrence (view source)

Defined in: uf_assem.h

Overview
Determines whether an object is an object/part occurrence or a prototype object.

Return
TRUE = if object occurrence or a part occurrence.
FALSE = if object is a prototype object.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

```
logical UF_ASSEM_is_occurrence
(
    tag_t entity
)
```

tag_t	**entity**	Input	Tag of object

UF_ASSEM_is_part_deformable (view source)

Defined in: uf_assem.h

Environment
Internal and External

History
Initially released in NX2

Required License(s)
gateway

```
logical UF_ASSEM_is_part_deformable
(
    tag_t part
)
```

tag_t	**part**	Input	

UF_ASSEM_is_part_occurrence (view source)

Defined in: uf_assem.h

Overview
Determines whether a given tag is a part occurrence.

Return
TRUE = if tag is a part occurrence.
FALSE = if tag is not a part occurrence.

Environment
Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

logical UF_ASSEM_is_part_occurrence
(
tag_t occurrence
)

tag_t	**occurrence**	Input	Tag of occurrence

UF_ASSEM_is_preassembled (view source)

Defined in: uf_assem.h

Overview
Determines if the given part occurrence is preassembled in the given sequence

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

int UF_ASSEM_is_preassembled
(
tag_t sequence,
tag_t part_occ,
logical * preassembled
)

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	Tag of the part occurrence
logical *	**preassembled**	Output	True if the part occurrence is preassembled in the sequence, false otherwise.

UF_ASSEM_is_ref_set_member (view source)

Defined in: uf_assem.h

Overview

Queries whether the specified tag is a member of any reference sets.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_ASSEM_is_ref_set_member
(
    tag_t potential_member,
    logical * member_flag
)
```

tag_t	**potential_member**	Input	Tag to test for reference set membership.
logical *	**member_flag**	Output	TRUE if object is in a reference set.

UF_ASSEM_make_current_step (view source)

Defined in: uf_assem.h

Overview
Plays back the sequence up to the given step. The given step is not itself played back - it becomes the step that will be played back next.

UF_ASSEM_initialize_sequencing must be called before calling this function.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_make_current_step
(
    tag_t step
)
```

tag_t	**step**	Input	Tag of the step that will be made current

---

UF_ASSEM_move_step (view source)

Defined in: uf_assem.h

Overview
Moves a step to a different location in it's sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

int UF_ASSEM_move_step
(
tag_t step_to_be_moved,
tag_t insert_at_step
)

tag_t	**step_to_be_moved**	Input	The tag of the step being moved
tag_t	**insert_at_step**	Input	The step to be inserted before. May be null in which case the step is moved to the end of the sequence

---

UF_ASSEM_occ_is_in_work_part (view source)

Defined in: uf_assem.h

Overview
Checks if the part occurrence is in the current work part.

Environment
Internal and External

Required License(s)
gateway

int UF_ASSEM_occ_is_in_work_part
(
tag_t part_occ,
logical * is_in_work
)

tag_t	**part_occ**	Input	The tag of the part occurrence to check.
logical *	**is_in_work**	Output	TRUE if the part occurrence is in the current work part, FALSE if it is not.

UF_ASSEM_part_is_descendant (view source)

Defined in: uf_assem.h

Overview
Queries whether a descendent part is a descendent of a parent part. The descendent does not have to be a direct child of the parent. This routine returns a value of TRUE if the same part is given to both arguments.

Return
Returns TRUE if descendent_part is a descendent of parent_part, FALSE otherwise.

Environment

Internal and External

See Also
The following table of parameter values for this function is based on the automobile example .
Refer to table

Required License(s)
gateway

logical UF_ASSEM_part_is_descendant
(
tag_t parent_part,
tag_t descendent_part
)

tag_t	**parent_part**	Input	Tag of parent part
tag_t	**descendent_part**	Input	Tag of descendent part

UF_ASSEM_playback_animate_to_frame (view source)

Defined in: uf_assem.h

Overview
Playback a frame-by-frame animation of the specified sequence, from the current frame to a target frame. The specified sequence will become the active sequence.

The meaning of the target_frame parameter is equivalent to the "current frame" combo box in the Sequencing UI. When the sequence duration is N frames, the target_frame parameter may range from 0 to N inclusive.

The animation speed can be increased by skipping frames. The frame_skip must be >= 0; when the frame_skip is 0, every frame is drawn.

Environment
Internal and External. Please note that in v18, playback is only implemented as a change of display state. The playback functions are probably meaningful only in the internal user function environment.

UF_ASSEM_initialize_sequencing must be called before calling this function.

See Also
UF_ASSEM_ask_current_frame

History
Initially released in NX 8.0.1

Required License(s)
assemblies

int UF_ASSEM_playback_animate_to_frame
(
tag_t sequence,
int target_frame,
int frame_skip
)

tag_t	**sequence**	Input	The sequence to be played back

int	**target_frame**	Input	the frame to stop at
int	**frame_skip**	Input	how many frames to advance at a time

---

UF_ASSEM_playback_seek_to_frame (view source)

Defined in: uf_assem.h

Overview

Seek to a specified playback frame in the given sequence. The sequence will be advanced to the target frame immediately, without any playback animation. The specified sequence will become the active sequence.

The meaning of the target_frame parameter is equivalent to the "current frame" combo box in the Sequencing UI. When the sequence duration is N frames, the target_frame parameter may range from 0 to N inclusive.

Environment

Internal and External

See Also

UF_ASSEM_ask_current_frame

History

Initially released in NX 8.0.1

Required License(s)

assemblies

```
int UF_ASSEM_playback_seek_to_frame
(
    tag_t sequence,
    int target_frame
)
```

tag_t	**sequence**	Input	The tag of the sequence
int	**target_frame**	Input	the frame to stop at

UF_ASSEM_playback_sequence (view source)

Defined in: uf_assem.h

Overview

Playback the given assembly sequence, using the playback_command.
The specified sequence will become the active sequence.
The playback command is one of the enumerated constants

UF_ASSEM_seq_step_forward: step forward until next changed frame
UF_ASSEM_seq_step_backward: step backward until next changed frame
UF_ASSEM_seq_play_forward: fast-forward to end of sequence
UF_ASSEM_seq_play_backward: rewind to start of sequence

Each sequence step has a duration, measured in frames. The step_forward and step_backward modes of this function advance the sequence frame-by-frame, with one exception: frames where no motion occurs are skipped.

(For example: assemble and disassemble steps may have durations larger than one frame, but this function will advance through their entire duration with

one call. Motion steps, on the other hand, will be stepped through frame-by-frame, because components move between every frame.)

If "Stop Before Collision" has been enabled in the NX UI, then the step_forward/step_backward modes will not advance to the next frame if this would cause a collision.

To detect how many frames the sequence has been advanced after a call to this function, call UF_ASSEM_ask_current_frame before and after playback and compare the values returned.

Environment
Internal and External. Please note that in v18, playback is only implemented as a change of display state. The playback functions are probably meaningful only in the internal user function environment.

UF_ASSEM_initialize_sequencing must be called before calling this function.

See Also
UF_ASSEM_ask_sequence_duration
UF_ASSEM_ask_step_duration
UF_ASSEM_ask_step_element_durations
UF_ASSEM_ask_current_frame
UF_ASSEM_playback_seek_to_frame
UF_ASSEM_playback_animate_to_frame

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_playback_sequence
(
    tag_t sequence,
    int playback_command
)
```

tag_t	**sequence**	Input	The sequence to be played back
int	**playback_command**	Input	One of the enumerated constants UF_ASSEM_seq_step_forward

---

UF_ASSEM_preassemble_partocc (view source)

Defined in: uf_assem.h

Overview
Preassembles the given part occurrence in the sequence. Part occurrences must be preassembled (or assembled in the sequence) before they can be considered for disassembly.

The preassembled set can be used to define the initial state of the assembly sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_preassemble_partocc
(
    tag_t sequence,
    tag_t part_occ
)
```

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	The part occurrence to be preassembled

---

UF_ASSEM_register_animation_callback (view source)

Defined in: uf_assem.h

Overview

Register the customer-supplied callback function .
This callback function will called at end of each frame when play a motion sequence synchronously.
eg.UF_ASSEM_playback_animate_to_frame.
User also can register a user data with this callback function. This user data will be passed as parameter when this callback function called.

Environment

Internal and External

History

Initially released in NX 8.5.1

Required License(s)

assemblies

```
int UF_ASSEM_register_animation_callback
(
    UF_ASSEM_animation_callback_f_t callback,
    void * user_data
)
```

UF_ASSEM_animation_callback_f_t	**callback**	Input	The user callback function.
void *	**user_data**	Input	The user data.

UF_ASSEM_remove_from_cset (view source)

Defined in: uf_assem.h

Overview

Allows you to remove a component from an existing component set.
It is not an error to attempt to remove a component from a set that is not a member of that set.

Environment

Internal and External

See Also

See UF_ASSEM_add_to_cset
to add a component to an existing component set.
Refer to example

Required License(s)

assemblies

int UF_ASSEM_remove_from_cset
(
tag_t cset,
tag_t component
)

tag_t	**cset**	Input	Object identifier of component set
tag_t	**component**	Input	Object identifier of component to be removed

---

UF_ASSEM_remove_ignored (view source)

Defined in: uf_assem.h

Overview
Removes the given part occurrence from the ignored set of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

int UF_ASSEM_remove_ignored
(
tag_t sequence,
tag_t part_occ
)

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	The part occurrence to be removed from the ignored set

---

UF_ASSEM_remove_instance (view source)

Defined in: uf_assem.h

Overview
Removes an instance from an assembly part.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_remove_instance
(
tag_t instance

```
)
```

tag_t	**instance**	Input	Tag of instance to remove.

UF_ASSEM_remove_preassembled (view source)

Defined in: uf_assem.h

Overview
Removes the given part occurrence from the preassembled set of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_remove_preassembled
(
    tag_t sequence,
    tag_t part_occ
)
```

tag_t	**sequence**	Input	The tag of the given sequence
tag_t	**part_occ**	Input	The part occurrence to be removed from the preassembled set

---

UF_ASSEM_remove_ref_set_members (view source)

Defined in: uf_assem.h

Overview
Removes an array of members from the specified reference set.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
assemblies

```
int UF_ASSEM_remove_ref_set_members
(
    tag_t ref_set,
    int member_count,
    tag_t * ref_set_members
)
```

tag_t	**ref_set**	Input	Tag of the reference set to remove members from.

int	**member_count**	Input	The number of members to remove
tag_t *	**ref_set_members**	Input	Array of the members to remove.

UF_ASSEM_remove_sequencing_view (view source)

Defined in: uf_assem.h

Overview
Remove a view from the list of views used to display sequencing.
This should called after user called the UF_ASSEM_add_sequencing_view() that added views.

Environment
Internal and External

History
Initially released in NX 8.5.1

Required License(s)
assemblies

```
int UF_ASSEM_remove_sequencing_view
(
    tag_t view
)
```

tag_t	**view**	Input	The tag of view to remove.

UF_ASSEM_rename_instance (view source)

Defined in: uf_assem.h

Overview
Renames an instance from an assembly part. Note that this changes the instance name, but not the name of the corresponding part occurrence and not the object name referred to by UF_OBJ_ask_name.

Environment
Internal and External

History
V15.0 change: This function was modified to return an integer error code which can be passed into UF_get_fail_message.

Required License(s)
assemblies

```
int UF_ASSEM_rename_instance
(
    tag_t instance,
    const char * new_name
)
```

tag_t	**instance**	Input	Tag of instance to rename.

const char *	**new_name**	Input	New name for instance; must be no longer than UF_OBJ_NAME_NCHARS characters.

UF_ASSEM_replace_refset (view source)

Defined in: uf_assem.h

Overview

Replace the reference set used by one or more instances or part occurrences held in the target_tags array. If an instance is specified, all part occurrences of that instance will use the new reference set. This routine only works for immediate children of the work part.

If new_refset_name is a NULL pointer or a zero-length string, the entire part is used. If it is the string "Empty", then none of the part is displayed.

Environment

Internal and External

Required License(s)

assemblies

```
int UF_ASSEM_replace_refset
(
    int count,
    tag_t * target_tags,
    const char * new_refset_name
)
```

int	**count**	Input	Count of part occurrences
tag_t *	**target_tags**	Input	count Array of instance and/or part occurrence tags
const char *	**new_refset_name**	Input	Name of new reference set; must be no longer than UF_OBJ_NAME_NCHARS characters.

UF_ASSEM_reposition_instance (view source)

Defined in: uf_assem.h

Overview

Reposition an existing instance to a new position/orientation.

Environment

Internal and External

Required License(s)

assemblies

```
int UF_ASSEM_reposition_instance
(
    tag_t instance,
    double new_origin [ 3 ] ,
    double new_csys_matrix [ 6 ]
)
```

tag_t	**instance**	Input	tag of instance to reposition
double	**new_origin [3]**	Input	New orientation of the instance
double	**new_csys_matrix [6]**	Input	New coordinate system matrix

---

UF_ASSEM_reposition_part_occurrence (view source)

Defined in: uf_assem.h

Overview
Transforms the given part occurrence by the given delta transform, in the context of the owning-assembly of the part occurrence. The option is used to control what happens if the given part occ can't be transformed (because the required explicit override does not exist).

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
Internal and External

History
Initially released in NX3.

Required License(s)
assemblies

```
int UF_ASSEM_reposition_part_occurrence
(
    tag_t part_occ,
    double xform [ 4 ] [ 4 ] ,
    UF_ASSEM_level_option_t option
)
```

tag_t	**part_occ**	Input	The tag of the part occurrence you wish to transform.
double	**xform [4] [4]**	Input	A 4x4 delta transform matrix in terms of the displayed part.
UF_ASSEM_level_option_t	**option**	Input	If the given part occurrence can't be transformed, the option determines what happens next. UF_ASSEM_use_strict_level - the operation will fail and return UF_ASSEM_err_operation_requires_override UF_ASSEM_use_existing_level - the operation will instead transform the corresponding part occurrence at the nearest level down in the assembly where an appropriate override exists. If none exists then the instance will be transformed. UF_ASSEM_establish_override - the operation will create an override in the part of the given part occurrence and then perform the transform.

UF_ASSEM_restore_load_options (view source)

Defined in: uf_assem.h

Overview
Restores the load options as recorded in the specified load options file. The file should be in the format of the standard load_options.def file, although there is no requirement to name the file "load_options.def".

If NULL is passed in place of the specified file, this function restores the `load_options.def' file in the current directory if it exists.
If the file passed in does not contain a complete set of load options settings, the ones that are specified are changed and the others are left unchanged.

Return
Return code:
0 = No error
1 = The file passed in does not exist or is unreadable
2 = The file passed is not a valid load options file
n = Error code

Environment
Internal and External

See Also
UF_ASSEM_ask_assem_options
UF_ASSEM_set_assem_options
Refer to example

Required License(s)
gateway

```
int UF_ASSEM_restore_load_options
(
    const char * load_options_file
)
```

const char *	**load_options_file**	Input	The load options file that should be restored.

UF_ASSEM_restore_work_part_context_quietly (view source)

Defined in: uf_assem.h

Overview
Restores the previous work part context as returned by UF_ASSEM_set_work_part_context_quietly and frees previous_work_part_context. After a call to this routine, previous_work_part_context will be NULL.

Environment
Internal and External

See Also
UF_ASSEM_set_work_part_context_quietly

History
NX11.0

Required License(s)
gateway

int UF_ASSEM_restore_work_part_context_quietly
(
UF_ASSEM_work_part_context_p_t* previous_work_part_context
)

UF_ASSEM_work_part_context_p_t*	**previous_work_part_context**	Input / Output	The work part context returned by UF_ASSEM_set_work_part_context_quietly. Will automatically be freed by this function.

UF_ASSEM_revert_explode_comp (view source)

Defined in: uf_assem.h

Overview

Explodes a component back to its normal assembly position. In general, exploding a component causes its own components to move with it. This function sets the component such that it always (and associatively) retains its real position.

Exploding or unexploding the component loses this setting, and it then explodes from the explosion position given by its parents in the assembly.

Environment

Internal and External

Required License(s)

assemblies

int UF_ASSEM_revert_explode_comp
(
tag_t explosion,
tag_t component
)

tag_t	**explosion**	Input	The explosion in which to modify the component
tag_t	**component**	Input	the tag of a part occurrence not instance

---

UF_ASSEM_set_active_arrangement (view source)

Defined in: uf_assem.h

Overview

UF_ASSEM_set_active_arrangement

Sets which is the Active Assembly Arrangement in its part. (The part is deduced from the given arrangement.) This is the Assembly Arrangement which would take effect for it if that part were made the Displayed Part. If it is currently the displayed part, then the component positions will be updated immediately as appropriate.

See Also

UF_ASSEM_ask_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part

Refer to example

History
Initially released in NX2.

Required License(s)
assemblies

int UF_ASSEM_set_active_arrangement
(
tag_t arrangement
)

tag_t	**arrangement**	Input	The new Active Assembly Arrangement

---

UF_ASSEM_set_assem_options (view source)

Defined in: uf_assem.h

Overview
Set the current settings of the assembly options. To assure that your program continues to work correctly with future versions of Open API, we recommend that you call UF_ASSEM_ask_assem_options to initialize your options structure before setting the values you desire to set/modify. New options may be added to the UF_ASSEM_options_s structure in the future.

Environment
Internal and External

See Also
Refer to UF_ASSEM_options_t

Required License(s)
gateway

int UF_ASSEM_set_assem_options
(
UF_ASSEM_options_t * options
)

UF_ASSEM_options_t *	**options**	Input	Pointer to an assembly options structure

---

UF_ASSEM_set_auto_add_new_comps (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_set_auto_add_new_comps

Set whether new components will be added to the specified reference set automatically or not.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment

This function is supported for both Internal and External execution.

History
Initially released in NX3.

Required License(s)
assemblies

int UF_ASSEM_set_auto_add_new_comps
(
tag_t ref_set,
logical add_new_comps
)

tag_t	**ref_set**	Input	The reference set
logical	**add_new_comps**	Input	Whether new components are to be added automatically

---

UF_ASSEM_set_cost_of_step (view source)

Defined in: uf_assem.h

Overview
Assign a cost to the given step

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

int UF_ASSEM_set_cost_of_step
(
tag_t step,
double assign_cost
)

tag_t	**step**	Input	The tag of the given step
double	**assign_cost**	Input	The cost to be assigned to the step

---

UF_ASSEM_set_default_arrangement (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_set_default_arrangement

Set the Default Assembly Arrangement for its part. (The part is deduced from the given arrangement.) This is the Assembly Arrangement which would used by a new parent assembly if this part were added to that parent as a new component. No components are

repositioned as a result of calling this function.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_set_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History
Initially released in NX2.

Required License(s)
assemblies

```
int UF_ASSEM_set_default_arrangement
(
    tag_t arrangement
)
```

tag_t	**arrangement**	Input	The new Default Assembly Arrangement for its part

---

UF_ASSEM_set_default_ref_sets (view source)

Defined in: uf_assem.h

Overview
Applies the specified reference sets as the defaults in the order supplied. The default reference sets must have "As Saved", "Entire Part" and "Empty" in the list. If any of these are not specified in the supplied list, they are added to the end of the list in that order.

Environment
Internal and External

See Also
UF_ASSEM_ask_default_ref_sets

History
Original release was in V13.0.

Required License(s)
gateway

```
int UF_ASSEM_set_default_ref_sets
(
    int n_ref_sets,
    char * * default_ref_sets
)
```

int	**n_ref_sets**	Input	The number of default reference sets being set.

char * *	**default_ref_sets**	Input	An ordered array of reference set names indicating the default reference sets.

UF_ASSEM_set_instance_intent (view source)

Defined in: uf_assem.h

Overview
Assigns intent data to an instance.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_set_instance_intent
(
    tag_t instance,
    UF_ASSEM_instance_intent_p_t instance_intent
)
```

tag_t	**instance**	Input	Object identifier of the instance
UF_ASSEM_instance_intent_p_t	**instance_intent**	Input	Data of the intent.

UF_ASSEM_set_ref_set_by_cset (view source)

Defined in: uf_assem.h

Overview
Changes the reference set representing each component to a reference set of the given name. If no reference set exists with that name then the component is left as it was.

Environment
Internal and External

See Also
Refer to example

Required License(s)
gateway

```
int UF_ASSEM_set_ref_set_by_cset
(
    tag_t cset,
    const char * cname
)
```

tag_t	**cset**	Input	Set of components to set reference set for
const char *	**cname**	Input	Name of reference set to represent each component by. The entire part reference set can be set by entering "None" as the reference set name.

UF_ASSEM_set_save_trueshape (view source)

Defined in: uf_assem.h

Overview
Set the save true shape data option to the input value

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_set_save_trueshape
(
    logical save_trueshape_data
)
```

logical	**save_trueshape_data**	Input	Flag indicating whether true shape data should be saved. (TRUE = save true shape data, FALSE = do not save true shape data)

UF_ASSEM_set_search_directories (view source)

Defined in: uf_assem.h

Overview
Sets the list of the search directories. You must call
UF_ASSEM_set_assem_options to enable the use of search
directories.
This function may return the following non-standard error codes:
1 if at least one of the directories was bad
2 if all the directories were bad
All other error codes can be processed using UF_get_fail_message.

Environment
Internal and External

See Also
UF_ASSEM_set_assem_options

Required License(s)
gateway

```
int UF_ASSEM_set_search_directories
(
    int count,
    char * * dir_list,
    logical sub_dir [ ]
)
```

int	**count**	Input	The number of directories in the list.
char * *	**dir_list**	Input	An array of "count" pointers to character strings containing the directory names.
logical	**sub_dir [ ]**	Input	An array of "count" logicals. Each logical is TRUE if the subdirectories of the corresponding directory

	should be searched, or FALSE if the subdirectories should not be searched.

UF_ASSEM_set_sequence_description (view source)

Defined in: uf_assem.h

Overview
Sets the description of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_set_sequence_description
(
    tag_t sequence,
    char * desc
)
```

tag_t	**sequence**	Input	The tag of the given sequence
char *	**desc**	Input	The description to be assigned to the sequence

---

UF_ASSEM_set_sequence_name (view source)

Defined in: uf_assem.h

Overview
Sets the name of the given sequence.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
gateway

```
int UF_ASSEM_set_sequence_name
(
    tag_t sequence,
    char * name
)
```

tag_t	**sequence**	Input	The tag of the given sequence
char *	**name**	Input	The name of the sequence

UF_ASSEM_set_step_increment (view source)

Defined in: uf_assem.h

Overview
Sets the step increment of the given sequence. By default, the step increment for a newly created sequence is 10.

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_set_step_increment
(
    tag_t sequence,
    int increment
)
```

tag_t	**sequence**	Input	The tag of the given sequence
int	**increment**	Input	The step increment for the sequence

UF_ASSEM_set_suppression_exp (view source)

Defined in: uf_assem.h

Overview
Assigns an expression to control the suppress state of the given instance.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_set_suppression_exp
(
    tag_t instance,
    const char * exp_string,
    tag_t * exp_tag
)
```

tag_t	**instance**	Input	The tag of the instance to assign the suppression expression for.
const char *	**exp_string**	Input	Expression string to use. If the expression already exists in the instance's part, then that expression will be used. You can use UF_MODL_ask_exp_tag_string to get the expression string for an existing expression.
tag_t *	**exp_tag**	Output	Expression tag for the given expression string.

UF_ASSEM_set_time_of_step (view source)

Defined in: uf_assem.h

Overview
Assign a time to the given step

Environment
Internal and External

History
Initially released in V18.0

Required License(s)
assemblies

```
int UF_ASSEM_set_time_of_step
(
    tag_t step,
    double assign_time
)
```

tag_t	**step**	Input	The tag of the given step
double	**assign_time**	Input	The time to be assigned to the step

---

UF_ASSEM_set_used_arrangement (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_set_used_arrangement

Sets the Used Assembly Arrangement for a given component. This is the Assembly Arrangement which is used to represent that component in the parent part of that component. The positions of the components beneath the given component are updated as appropriate.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_ask_active_arrangement
UF_ASSEM_set_active_arrangement
UF_ASSEM_ask_used_arrangement
UF_ASSEM_ask_default_arrangement
UF_ASSEM_set_default_arrangement
UF_ASSEM_ask_name_of_arrangement
UF_ASSEM_ask_arrangements_in_part
Refer to example

History

Initially released in NX2.

Required License(s)
assemblies

int UF_ASSEM_set_used_arrangement
(
tag_t component,
tag_t arrangement
)

tag_t	**component**	Input	The component to modify
tag_t	**arrangement**	Input	The new Used Assembly Arrangement for that Component

---

UF_ASSEM_set_view_explosion (view source)

Defined in: uf_assem.h

Overview
Sets the explosion used in a view; the view may already have an explosion in which case it is replaced. Drafting view-dependent geometry such as dimensions and silhouettes is appropriately adjusted or marked for update. "explosion" may be NULL_TAG, in which case the view is reset to having no explosion if one was in use.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_set_view_explosion
(
tag_t view,
tag_t explosion
)

tag_t	**view**	Input	The view which should be associated with (and therefore display) the given explosion.
tag_t	**explosion**	Input	The explosion to use.

UF_ASSEM_set_work_occurrence (view source)

Defined in: uf_assem.h

Overview
Set the work part and work occurrence to part_occur. This routine sets just the work part, not the displayed part. If you set the work part to a partially loaded part, the system finishes loading the part.

Environment
Internal and External

Required License(s)
gateway

```
int UF_ASSEM_set_work_occurrence
(
    tag_t part_occur
)
```

tag_t	**part_occur**	Input	Tag of part occurrence

UF_ASSEM_set_work_part (view source)

Defined in: uf_assem.h

Overview

Sets the work part to a different part. If part is NULL_TAG, the work part is set to the displayed part. This routine sets just the work part, not the displayed part. This routine only modifies the work part in an assembly. If not working in an assembly, use UF_PART_set_display_part. If there is more than one occurrence in the displayed part, the first one is chosen. If you set the work part to a partially loaded part, the system finishes loading the part.

Environment

Internal and External

History

In V16.0 this function was changed to no longer perform an update. If an update is required, the calling program must now specifically call UF_MODL_update.

Required License(s)

gateway

```
int UF_ASSEM_set_work_part
(
    tag_t part
)
```

tag_t	**part**	Input	Tag of part to be set as work part.

UF_ASSEM_set_work_part_context_quietly (view source)

Defined in: uf_assem.h

Overview

Sets the work part to a different part quietly. Setting the work part quietly sets the work part without updating the display or any other parts of the NX user interface in an interactive NX session.
By doing this, objects can be created in other than the current work part (which most Open API routines currently rely on).
This routine is primarily designed to temporarily change the work part to a component in the current displayed assembly without updating NX display or NX user interface to reflect the new work part. Another use case is to open a part without displaying by calling UF_PART_open_quiet, and then setting it as the work part quietly with this routine.
This routine will return an error if the given part is not already fully loaded.

This routine should be used with great care and the previous work part context should be restored immediately with UF_ASSEM_restore_work_part_context_quietly, using the previous_work_part_context output by this routine.

Between the call to this function and UF_ASSEM_restore_work_part_context_quietly it is not expected that the display part changes.
This routine does not expect the root and work part that are active at the time it is called to become invalid before UF_ASSEM_restore_work_part_context_quietly is called.

A call to UF_ASSEM_restore_work_part_context_quietly will free the output previous_work_part_context.

Warning:
This routine can be used when necessary to perform operations in parts which are not the work part. The function should always be called in conjunction with UF_ASSEM_restore_work_part_context_quietly. Call UF_ASSEM_set_work_part_context_quietly before the desired operation is performed and then call UF_ASSEM_restore_work_part_context_quietly immediately after the operation has been performed.
Actions that cannot be performed interactively are not supported with this routine.

Avoid using this function with the Advanced Simulation application, when a drawing sheet is displayed, or any application that does not support situations where the work part is different from the display part.

Environment
Internal and External

See Also
UF_ASSEM_restore_work_part_context_quietly
UF_PART_open_quiet

History
NX11.0

Required License(s)
gateway

int UF_ASSEM_set_work_part_context_quietly
(
tag_t part_tag,
UF_ASSEM_work_part_context_p_t* previous_work_part_context
)

tag_t	**part_tag**	Input	The tag of the part to become the new work part
UF_ASSEM_work_part_context_p_t*	**previous_work_part_context**	Output to UF_*free*	The previous work part context at the time this routine was called (NX should be reset to this immediately after completing the context sensitive operations.)

UF_ASSEM_show_component (view source)

Defined in: uf_assem.h

Overview
Unhide the given component in the given view.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_show_component
(

```
    tag_t component,
    tag_t view
)
```

tag_t	**component**	Input	the tag of a part occurrence not instance
tag_t	**view**	Input	The view in which to un-hide the component

---

UF_ASSEM_substitute_component (view source)

Defined in: uf_assem.h

Overview

Replaces the part used by an instance with a new part. The instance to be replaced must be an instance of the work part. This operation is equivalent to deleting the instance and creating a new one so that the old instance tag is no longer valid after a call to this routine. Instead, the tag of the new instance is returned.

Environment

Internal and External

See Also

See UF_ASSEM_use_alternate if you want to maintain mating conditions.

Required License(s)

assemblies

```
int UF_ASSEM_substitute_component
(
    tag_t * instance,
    const char * new_part_version,
    const char * new_comp_name,
    const char * new_refset_name,
    int layer,
    UF_PART_load_status_t * load_status
)
```

tag_t *	**instance**	Input / Output	Pointer to tag of instance
const char *	**new_part_version**	Input	Name of new component part version
const char *	**new_comp_name**	Input	Name of new component; must be no longer than UF_OBJ_NAME_NCHARS characters.
const char *	**new_refset_name**	Input	Name of new reference set; must be no longer than UF_OBJ_NAME_NCHARS characters.
int	**layer**	Input	-1 = original 0 = use work layer 1-255 = use specified layer
UF_PART_load_status_t *	**load_status**	Output to UF_*free*	See the UF_PART_load_status_t structure definition. The allocated structure is filled with the names and associated error codes of any parts that did not load correctly. The allocated arrays must be freed with UF_free_string_array and UF_free.

UF_ASSEM_suppress_array (view source)

Defined in: uf_assem.h

Overview
Suppress a component array so that it no longer updates. Note that the appearance of the array does not change.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_suppress_array
(
    tag_t array
)
```

tag_t	**array**	Input	Tag of array to be suppressed.

UF_ASSEM_suppress_instances (view source)

Defined in: uf_assem.h

Overview
Suppresses the given instances. This function returns the first error code found in the failures array, or 0 if there were no errors.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_suppress_instances
(
    int n_instances,
    tag_t instances [ ] ,
    int failures [ ]
)
```

int	**n_instances**	Input	The number of instances to suppress.
tag_t	**instances [ ]**	Input	The tags of the instances to suppress.
int	**failures [ ]**	Output	Failure codes for each instance (0 if no failure)

UF_ASSEM_terminate_sequencing (view source)

Defined in: uf_assem.h

Overview
Terminates the sequencing environment.
Must be called when no further sequencing operations are intended.

Environment
Internal and External

History
Initially released in NX 3.0

Required License(s)
assemblies

int UF_ASSEM_terminate_sequencing
(
void
)

UF_ASSEM_unexplode_component (view source)

Defined in: uf_assem.h

Overview
Unexplodes a component by setting the explosion transform of a component to NULL. The new position of the component in the explosion is the position of its parent plus its normal assembly transform.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_unexplode_component
(
tag_t explosion,
tag_t part_occurrence
)

tag_t	**explosion**	Input	The explosion in which to modify the component
tag_t	**part_occurrence**	Input	The component (its part occurrence)

UF_ASSEM_unset_suppression_exp (view source)

Defined in: uf_assem.h

Overview
Removes the dependency on an expression for the given instance's suppress state.

Environment
Internal and External

Required License(s)
assemblies

int UF_ASSEM_unset_suppression_exp
(

```
    tag_t instance
)
```

tag_t	**instance**	Input	The tag of the instance to remove suppression by expression from.

UF_ASSEM_unsuppress_array (view source)

Defined in: uf_assem.h

Overview
Unsuppress a component array so that it updates.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_unsuppress_array
(
    tag_t array
)
```

tag_t	**array**	Input	Tag of array to unsuppress.

UF_ASSEM_unsuppress_instances (view source)

Defined in: uf_assem.h

Overview
Unsuppresses the given instances. This function returns the first error code found in the failures array, or 0 if there were no errors.

Environment
Internal and External

Required License(s)
assemblies

```
int UF_ASSEM_unsuppress_instances
(
    int n_instances,
    tag_t instances [ ] ,
    int failures [ ]
)
```

int	**n_instances**	Input	The number of instances to suppress.
tag_t	**instances [ ]**	Input	The tags of the instances to suppress.
int	**failures [ ]**	Output	Failure codes for each instance (0 if no failure)

UF_ASSEM_update_component_group (view source)

Defined in: uf_assem.h

Overview
UF_ASSEM_update_component_group

Updates the input component group if present on the input part.

Return
Error code - use UF_get_fail_message to obtain the corresponding error message

Environment
This function is supported for both Internal and External execution.

See Also
UF_ASSEM_add_to_cset
UF_ASSEM_remove_from_cset

History
Initially released in NX8.0.

Required License(s)
assemblies

```
int UF_ASSEM_update_component_group
(
    const tag_t part_tag,
    const char* component_group_name,
    logical do_update_structure
)
```

const tag_t	**part_tag**	Input	part whose component group is to be updated
const char*	**component_group_name**	Input	name of component group which is to be updated
logical	**do_update_structure**	Input	whether to update the structure before updating the component group

---

UF_ASSEM_upgrade_to_instances (view source)

Defined in: uf_assem.h

Overview
Upgrade the components in an assembly in a part to use instances. If upgrade fails, then upgrade_status contains failure codes for each component that failed to upgrade.

Return
Returns TRUE if successful; otherwise returns FALSE.

Environment
Internal and External

See Also
See UF_ASSEM_upgrade_status_t

Required License(s)
assemblies

```
logical UF_ASSEM_upgrade_to_instances
(
    tag_t part,
    int n_components,
    tag_t * components,
    logical recurse,
    logical create_component,
    UF_ASSEM_upgrade_status_t * upgrade_status
)
```

tag_t	**part**	Input	Tag of part containing components to be upgraded
int	**n_components**	Input	Number of components in the "components" array
tag_t *	**components**	Input	n_components Array of old style components to be upgraded
logical	**recurse**	Input	TRUE = recurse thru listed components upgrading their subassemblies; FALSE = only upgrade components listed
logical	**create_component**	Input	FALSE = look for v10 part in memory and disk and update with part if found; otherwise fail to upgrade component TRUE = Try above, if part not found create part in current directory and move geometry to it
UF_ASSEM_upgrade_status_t *	**upgrade_status**	Output to UF_*free*	Contains failure code for each component that failed to upgrade. Structure arrays must be freed with UF_free.

UF_ASSEM_use_alternate (view source)

Defined in: uf_assem.h

Overview

Replaces the part used by an instance with a new part. The instance to be modified must be an instance of the work part.

Assuming that the old and new parts have been defined as alternates, the substitution maintains any mating conditions which involve the old part. See the Alternates section of the Assemblies User Manual.

This function can be called in exactly the same way as UF_ASSEM_substitute_component except that it is not possible to change the component's layer using this option. Note, however, that the instance tag is not changed by this function.

Environment

Internal and External

See Also

See UF_ASSEM_substitute_component
if maintaining mating conditions is not required.

Required License(s)

assemblies

```
int UF_ASSEM_use_alternate
(
```

```
    tag_t* instance,
    const char * new_part,
    const char * new_comp_name,
    const char * new_refset_name,
    UF_PART_load_status_t * load_status
)
```

tag_t*	**instance**	Input	Pointer to tag of instance
const char *	**new_part**	Input	Name of the new part to use in place of the old part
const char *	**new_comp_name**	Input	New name of the component; must be no longer than UF_OBJ_NAME_NCHARS characters.
const char *	**new_refset_name**	Input	Name of reference set to use; must be no longer than UF_OBJ_NAME_NCHARS characters.
UF_PART_load_status_t *	**load_status**	Output to UF_*free*	See the UF_PART_load_status_t structure definition. The allocated structure is filled with the names and associated error codes of any parts that did not load correctly. The allocated arrays must be freed with UF_free_string_array and UF_free.

UF_ASSEM_where_is_part_used (view source)

Defined in: uf_assem.h

Overview

Asks for the list of parent part tags that have the requested part as an immediate child in the current session. A parent part appears in the returned array once for each time it is a parent of the given part. So given wheel in auto you would see axle twice.

CAUTION: Do not confuse this command with the "where-used" function.

Return

Return the number of parent parts found.

Environment

Internal and External

See Also

The following table of parameter values for this function is based on the automobile example
Refer to table

Required License(s)

gateway

```
int UF_ASSEM_where_is_part_used
(
    tag_t part,
    tag_t * * parent_parts
)
```

tag_t	**part**	Input	Tag of child part for query
tag_t * *	**parent_parts**	Output to UF_*free*	Allocated array of tags of parent parts. The parent_parts array must be freed with UF_free when it is no longer needed.

UF_ASSEM_where_used_report (view source)

Defined in: uf_assem.h

Overview

Displays a where used report to the listing device for the given component filename. A list of files and corresponding error codes are allocated and returned in the load_status structure for any files that could not be processed successfully.

This function will not work in NXManager mode. An error is not returned and there will be no results returned in load_status structure.

Environment

Internal and External

See Also

UF_PART_load_status_t

History

V15.0 change: This function was modified to return an integer error code which can be passed into UF_get_fail_message.

Required License(s)

gateway

```
int UF_ASSEM_where_used_report
(
    const char * comp_name,
    const char * dir,
    int search_opt,
    logical do_all_levels,
    UF_PART_load_status_t * load_status
)
```

const char *	**comp_name**	Input	Component file name. This should be the simple name without a directory path.
const char *	**dir**	Input	specified directory if search_opt = 2
int	**search_opt**	Input	search options 0 = use search directories 1 = use component directory 2 = use <dir>
logical	**do_all_levels**	Input	report all levels of the assembly tree in which a component is found
UF_PART_load_status_t *	**load_status**	Output to UF_*free*	See the UF_PART_load_status_t structure definition. The allocated structure is filled with the names and associated error codes of any parts that did not load correctly. The allocated arrays must be freed with UF_free_string_array and UF_free.