

Additel FCINTF Driver (ADT227) Extend Interface Description

Additel FCINTF Driver expands some user-defined interface functions based on the FCINTF specification to address some of the customer's new requirements for ADT227:

- Allowed to delete the specified task;
- Add task directory hierarchy;
- Support uploading all verification results.

4.3 ICalibratorDownload

4.3.5 DeleteTag()

```
HRESULT DeleteTag( [in] long ISessionId,  
                  [in, string] LPCWSTR szwLocationInCalibrator,  
                  [out]FCISTATUS* pStatus);
```

Description

The client can call this function to delete the specified task (Tag). The driver uses this function to delete the qualified tasks in the calibrator, including the result data already generated in the task.

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
szwLocationInCalibrator	Identifies the tag in the calibrator. This can be any value, but it must be the same value returned by the driver during the download for this tag.
pStatus	Receives status to return to calling function.

4.3.6 AddFolder ()

```
HRESULT AddFolder( [in] long ISessionId,  
                  [in, string] LPCWSTR szwFolderName,  
                  [in, string] LPCWSTR szwParentId,  
                  [out, string] LPWSTR* pszwGroupId,  
                  [out]FCISTATUS* pStatus);
```

Description

The client calls this function to add a folder to the calibrator. The driver will add a folder to the specified location of the calibrator. However, it should be noted that if the client adds a folder with the same name to the same location multiple times, the returned *GroupId* content may be incorrect, and the client should avoid the above problem.

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
szwFolderName	Folder name.

szwParentId	Uniquely identifies the folder in the calibrator. This is a GUID string, If parent folder is the root directory, the string can be empty or "00000000-0000-0000-0000-000000000000"
pszwGroupId	Identifies the folder in the calibrator. It is a GUID string and must not be empty,example "f899c87a-db73-4ff8-80cf-07928c4f6950"
pStatus	Receives status to return to calling function.

Comment

pszwGroupId is an [out] string, so the driver and client must observe the proper memory allocation and deletion procedures (see 2.7). The driver must initialize this pointer to NULL, then use *CoTaskMemAlloc()* to allocate memory. The client must call *CoTaskMemFree()* on any non-null returned value.

4.3.7 DeleteFolder ()

```
HRESULT DeleteFolder(      [in] long ISessionId,
                           [in, string]  LPCWSTR szwGroupId,
                           [out]FCISTATUS* pStatus);
```

Description

The client calls this function to delete the specified folder. The driver will delete the folder and all files in the folder, including tasks and subfolders. After the deletion, the files in the folder cannot be recovered. Before the client performs this operation, make sure that you no longer need any files in this folder.

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
szwGroupId	Uniquely identifies the folder in the calibrator. This is a GUID string, and must not be empty,example "f899c87a-db73-4ff8-80cf-07928c4f6950";and root folder id "00000000-0000-0000-0000-000000000000" is not allowed.
pStatus	Receives status to return to calling function.

4.3.8 GetFolder ()

```
HRESULT GetFolder( [in] long ISessionId,
                   [in, string]  LPCWSTR szwFolderName,
                   [in, string]  LPCWSTR szwParentId,
                   [out, string] LPWSTR* pszwGroupId,
                   [out]FCISTATUS* pStatus);
```

Description

The client calls this function to obtain the ID of the folder (the unique ID in the folder calibrator).

Parameters

Parameters	Description
------------	-------------

ISessionId	Driver assigned number that identifies the current session.
szwFolderName	Folder name.
szwParentId	Uniquely identifies the folder in the calibrator. This is a GUID string, If parent folder is the root directory, the string can be empty or "00000000-0000-0000-0000-000000000000"
pszwGroupId	Identifies the folder in the calibrator. It is a GUID string and must not be empty, example "f899c87a-db73-4ff8-80cf-07928c4f6950". The driver will assign the same value to pszwGroupId in the AddFolder () function (see 4.3.6)
pStatus	Receives status to return to calling function.

Comment

pszwGroupId is an [out] string, so the driver and client must observe the proper memory allocation and deletion procedures (see 2.7). The driver must initialize this pointer to NULL, then use *CoTaskMemAlloc()* to allocate memory. The client must call *CoTaskMemFree()* on any non-null returned value.

4.3.9 EnterFolder ()

```
HRESULT EnterFolder(    [in] long ISessionId,
                        [in, string]    LPCWSTR szwGroupId,
                        [out]FCISTATUS* pStatus);
```

Description

The client calls this function to specify the download directory of the task. When the client executes the *DownloadTag* function, the driver downloads the task to the currently specified directory. If you need to change the download location of the task, just call this function again and enter the target directory. When the client calls the *FinishDownloading* or *AbortDownload* function, the task download directory will be restored to the root directory.

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
szwGroupId	Uniquely identifies the folder in the calibrator. This is a GUID string, If folder is the root directory, the string can be empty or "00000000-0000-0000-0000-000000000000"
pStatus	Receives status to return to calling function.

4.3.10 GetSubFolder()

```
HRESULT GetSubFolder( [in] long ISessionId,
                      [in, string]    LPCWSTR szwParentId,
                      [out]FCI_FOLDER pGroupIds[],
                      [in]    int nSize,
                      [out]int* pnCount,
                      [out]FCISTATUS* pStatus);
```

Description

The client calls this function to obtain all the subfolder information under the specified

directory. When the client executes the GetSubFolder function, the driver will find all subfolders under the szwParentId directory and store the data in the array pGroupIds. The client can specify the number of folders allowed to be returned each time. The client must call this function repeatedly until pStatus indicates that there is no more data to upload (FCI_END_OF_upload).

The uploaded folder data does not have an implied order, and there is no persistent history record of folder information in the driver.

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
szwParentId	Uniquely identifies the folder in the calibrator. This is a GUID string, If folder is the root directory, the string can be empty or "00000000-0000-0000-0000-000000000000"
pGroupIds[]	Array to receive subfolder information
nSize	Maximum capacity of pGroupIds array
pnCount	Receive the number of subfolders and return to the calling function. No more than nSize
pStatus	Receives status to return to calling function.

Comment

pGroupIds is an array, the client provides a pointer to the array and specifies the maximum size of the array nSize. Pointer in structure FCI_FOLDER: The driver must initialize this pointer to NULL, then use CoTaskMemAlloc() to allocate memory. The client must call CoTaskMemFree() on any non-null returned value.

This function returns FCI_END_OF_UPLOAD when there are no more folders to upload.

4.3.11 GetFolderTags()

```

HRESULT GetFolderTags( [in] long ISessionId,
                      [in, string] LPCWSTR szwParentId,
                      [out]FCI_TAG pTags[],
                      [in] int nSize,
                      [out]int* pnCount,
                      [out]FCISTATUS* pStatus);

```

Description

The client calls this function to obtain the label brief information (LocationInCalibrator, ParentID, DeviceTag, DeviceSerialNum) under the specified directory. When the client executes the GetFolderTags function, the driver will find all the tasks under the szwParentId directory (tasks created by FCINTF Driver) and store the data in the array pTags. The client can specify the number of tags allowed to be returned each time. The client must call this function repeatedly until pStatus indicates that there is no more data to upload (FCI_END_OF_upload).

The uploaded folder data does not have an implied order, and there is no persistent history record of folder information in the driver.

Parameters

Parameters	Description
------------	-------------

ISessionId	Driver assigned number that identifies the current session.
szwParentId	Uniquely identifies the folder in the calibrator. This is a GUID string, If folder is the root directory, the string can be empty or "00000000-0000-0000-0000-000000000000"
pTags []	Array to receive tag information
nSize	Maximum capacity of pTags array
pnCount	Receive the number of subfolders and return to the calling function. No more than nSize
pStatus	Receives status to return to calling function.

Comment

pTags is an array, the client provides a pointer to the array and specifies the maximum size of the array nSize. Pointer in structure FCI_TAG: The driver must initialize this pointer to NULL, then use CoTaskMemAlloc() to allocate memory. The client must call CoTaskMemFree() on any non-null returned value.

This function returns FCI_END_OF_UPLOAD when there are no more data to upload.

4.4 ICalibratorUpload

4.4.5 UploadNextResult ()

```
HRESULT UploadNextResult ( [in] long ISessionId,
                          [out, string] LPWSTR *pszwLocationInCalibrator,
                          [out, string] LPWSTR *pszwDeviceTag,
                          [out, string] LPWSTR *pszwDeviceSerialNum,
                          [out, string] LPWSTR *pszwTechnician,
                          [out, string] LPWSTR *pszwServiceNote,
                          [out] int *pnTemperatureStd, // IPTS_68 or ITS_90
                          [out] FCICOMMDEF *pCommDef,
                          [out] FCIRESULT *pAsFound,
                          [out] FCIRESULT *pAsLeft,
                          [out] FCISTATUS *pStatus );
```

Description

This function is basically the same as *UploadNextTag*, except that only one test result data is uploaded each time(asFound or asLeft). To ensure all available results are uploaded, the client must repeatedly call this function until the *pStatus* indicates that there are no more tags to upload (FCI_END_OF_UPLOAD).

Parameters

Parameters	Description
ISessionId	Driver assigned number that identifies the current session.
pszwLocationInCalibrator	Identifies the tag in the calibrator. This can be any value, but it must be the same value returned by the driver during the download for this tag.
pszwDeviceTag	Receives the device tag name or NULL.
pszwDeviceSerialNum	Receives the device serial number or NULL.
pszwTechnician	Receives the technician's name or NULL.

pszwServiceNote	Receives the service note or NULL.
pnTemperatureStd	IPTS-68 or ITS-90
pCommDef	Points to communication definition.
pAsFound	Points to as found results
pAsLeft	Points to as left results.
pStatus	Receives status to return to calling function.

Comment

Refer to 4.4.2 UploadNextTag.

5. Appendix A - FCINTF.IDL File

Add data structure:

```
typedef struct tagFCI_FOLDER
{
    [string] LPWSTR szwID;
    [string] LPWSTR szwName;
    [string] LPWSTR szwParentID;
} FCI_FOLDER;

typedef struct tagFCI_TAG
{
    [string] LPWSTR szwParentID;
    [string] LPWSTR szwLocationInCalibrator;
    [string] LPWSTR szwDeviceTag;
    [string] LPWSTR szwDeviceSerialNum;
} FCI_TAG;
```

ICalibratorDownload add interface:

```
HRESULT DeleteTag( [in] long ISessionId,
                  [in, string] LPCWSTR szwLocationInCalibrator,
                  [out] FCISTATUS* pStatus);

HRESULT AddFolder( [in] long ISessionId,
                  [in, string] LPCWSTR szwFolderName,
                  [in, string] LPCWSTR szwParentId,
                  [out, string] LPWSTR* pszwGroupId,
                  [out] FCISTATUS* pStatus);

HRESULT DeleteFolder( [in] long ISessionId,
                     [in, string] LPCWSTR szwGroupId,
                     [out] FCISTATUS* pStatus);

HRESULT GetFolder( [in] long ISessionId,
                  [in, string] LPCWSTR szwFolderName,
                  [in, string] LPCWSTR szwParentId,
                  [out, string] LPWSTR* pszwGroupId,
```

```

[out]    FCISTATUS* pStatus);
HRESULT EnterFolder( [in] long ISessionId,
                    [in, string]    LPCWSTR szwGroupId,
                    [out]    FCISTATUS* pStatus);
HRESULT GetSubFolder( [in] long ISessionId,
                    [in, string]    LPCWSTR szwParentId,
                    [out]    FCI_FOLDER pGroupIds[],
                    [in]    int nSize,
                    [out]    int* pnCount,
                    [out]    FCISTATUS* pStatus);
HRESULT GetFolderTags( [in] long ISessionId,
                    [in, string]    LPCWSTR szwParentId,
                    [out]    FCI_TAG pTags[],
                    [in]    int nSize,
                    [out]    int* pnCount,
                    [out]    FCISTATUS* pStatus);

```

ICalibratorUpload add interface:

```

HRESULT UploadNextResult ( [in] long ISessionId,
                        [out, string] LPWSTR *pszwLocationInCalibrator,
                        [out, string] LPWSTR *pszwDeviceTag,
                        [out, string] LPWSTR *pszwDeviceSerialNum,
                        [out, string] LPWSTR *pszwTechnician,
                        [out, string] LPWSTR *pszwServiceNote,
                        [out] int *pnTemperatureStd, // IPTS_68 or ITS_90
                        [out] FCICOMMDEF *pCommDef,
                        [out] FCIRESULT *pAsFound,
                        [out] FCIRESULT *pAsLeft,
                        [out] FCISTATUS *pStatus );

```