EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

WORKSHOP AGREEMENT

CWA 14050-14

November 2000

ICS 35.200; 35.240.15; 35.240.40

Extensions for Financial Services (XFS) interface specification - Release 3.0 - Part 14: Card Embossing Unit Class Interface

Ref. No CWA 14050-14:2000 E

Rue de Stassart, 36 • B-1050 Bruxelles
Tel : +32 2 550 08 11 • Fax : +32 2 550 08 19

Table of Contents

Foreword 3

1. Introduction 5
   - 1.1 Background to Release 3.0 5
   - 1.2 XFS Service-Specific Programming 5

2. Card Embossing Units 7

3. References 7

4. Info Commands 8
   - 4.1 WFS_INF_CEU_STATUS 8
   - 4.2 WFS_INF_CEU_CAPABILITIES 10
   - 4.3 WFS_INF_CEU_FORM_LIST 11
   - 4.4 WFS_INF_CEU_MEDIA_LIST 11
   - 4.5 WFS_INF_CEU_QUERY_FORM 11
   - 4.6 WFS_INF_CEU_QUERY_MEDIA 12
   - 4.7 WFS_INF_CEU_QUERY_FIELD 13

5. Execute Commands 15
   - 5.1 WFS_CMD_CEU_EMBOSS_CARD 15
   - 5.2 WFS_CMD_CEU_RESET 17

6. Events 18
   - 6.1 WFS_SVRE_CEU_INPUTBINTHRESHOLD 18
   - 6.2 WFS_SVRE_CEU_OUTPUTBINTHRESHOLD 18
   - 6.3 WFS_SVRE_CEU_RETAINBINTHRESHOLD 18
   - 6.4 WFS_EXEE_CEU_FIELDERROR 18
   - 6.5 WFS_EXEE_CEU_FIELDWARNING 19
   - 6.6 WFS_EXEE_CEU_MEDIAREMOVED 19
   - 6.7 WFS_SRVE_CEU_MEDIADETECTED 19
   - 6.8 WFS_EXEE_CEU_EMBOSS_FAILURE 19

7. Embossing Form, Field and Media Definitions 21
   - 7.1 Definition Syntax 21
   - 7.2 Embossing Form and Media Measurements 21
   - 7.3 Embossing Form Definition 22
   - 7.4 Embossing Field Definition 23
   - 7.5 Media Definition 24

8. C-Header file 25

Foreword

This CWA is revision 3.0 of the XFS interface specification.

The move from an XFS 2.0 specification (CWA 13449) to a 3.0 specification has been prompted by a series of factors.

Initially, there has been a technical imperative to extend the scope of the existing specification of the XFS Manager to include new devices, such as the Card Embossing Unit.

Similarly, there has also been pressure, through implementation experience and the advance of the Microsoft technology, to extend the functionality and capabilities of the existing devices covered by the specification.

Finally, it is also clear that our customers and the market are asking for an update to a specification, which is now over 2 years old. Increasing market acceptance and the need to meet this demand is driving the Workshop towards this release.

The clear direction of the CEN/ISSS XFS Workshop, therefore, is the delivery of a new Release 3.0 specification based on a C API. It will be delivered with the promise of the protection of technical investment for existing applications and the design to safeguard future developments.

The CEN/ISSS XFS Workshop gathers suppliers as well as banks and other financial service companies. A list of companies participating in this Workshop and in support of this CWA is available from the CEN/ISSS Secretariat.

This CWA was formally approved by the XFS Workshop meeting on 2000-10-18. The specification is continuously reviewed and commented in the CEN/ISSS Workshop on XFS. It is therefore expected that an update of the specification will be published in due time as a CWA, superseding this revision 3.0.

The CWA is published as a multi-part document, consisting of:

Part 1: Application Programming Interface (API) - Service Provider Interface (SPI); Programmer's Reference

Part 2: Service Classes Definition; Programmer's Reference

Part 3: Printer Device Class Interface - Programmer's Reference

Part 4: Identification Card Device Class Interface - Programmer's Reference

Part 5: Cash Dispenser Device Class Interface - Programmer's Reference

Part 6: PIN Keypad Device Class Interface - Programmer's Reference

Part 7: Check Reader/Scanner Device Class Interface - Programmer's Reference

Part 8: Depository Device Class Interface - Programmer's Reference

Part 9: Text Terminal Unit Device Class Interface - Programmer's Reference

Part 10: Sensors and Indicators Unit Device Class Interface - Programmer's Reference

Part 11: Vendor Dependent Mode Device Class Interface - Programmer's Reference

Part 12: Camera Device Class Interface - Programmer's Reference

Part 13: Alarm Device Class Interface - Programmer's Reference

Part 14: Card Embossing Unit Class Interface - Programmer's Reference

Part 15: Cash In Module Device Class Interface- Programmer's Reference

Part 16: Application Programming Interface (API) - Service Provider Interface (SPI) - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 17: Printer Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 18: Identification Card Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 19: Cash Dispenser Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 20: PIN Keypad Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 21: Depository Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 22: Text Terminal Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 23: Sensors and Indicators Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 24: Camera Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 25: Identification Card Device Class Interface - PC/SC Integration Guidelines

In addition to these Programmer's Reference specifications, the reader of this CWA is also referred to a complementary document, called Release Notes. The Release Notes contain clarifications and explanations on the CWA specifications, which are not requiring functional changes. The current version of the Release Notes is available online from http://www.cenorm.be/isss/Workshop/XFS.

Revision History:

3.00	October 18, 2000	Initial release

1. Introduction

1.1 Background to Release 3.0

The CEN XFS Workshop is a continuation of the Banking Solution Vendors Council workshop and maintains a technical commitment to the Win 32 API. However, the XFS Workshop has extended the franchise of multi vendor software by encouraging the participation of both banks and vendors to take part in the deliberations of the creation of an industry standard. This move towards opening the participation beyond the BSVC's original membership has been very succesful with a current membership level of more than 20 companies.

The fundamental aims of the XFS Workshop are to promote a clear and unambiguous specification for both service providers and application developers. This has been achieved to date by sub groups working electronically and quarterly meetings.

The move from an XFS 2.0 specification to a 3.0 specification has been prompted by a series of factors. Initially, there has been a technical imperative to extend the scope of the existing specification of the XFS Manager to include new devices, such as the Card Embossing Unit.

Similarly, there has also been pressure, through implementation experience and the advance of the Microsoft technology, to extend the functionality and capabilities of the existing devices covered by the specification.

Finally, it is also clear that our customers and the market are asking for an update to a specification, which is now over 2 years old. Increasing market acceptance and the need to meet this demand is driving the Workshop towards this release.

The clear direction of the XFS Workshop, therefore, is the delivery of a new Release 3.0 specification based on a C API. It will be delivered with the promise of the protection of technical investment for existing applications and the design to safeguard future developments.

1.2 XFS Service-Specific Programming

The service classes are defined by their service-specific commands and the associated data structures, error codes, messages, etc. These commands are used to request functions that are specific to one or more classes of service providers, but not all of them, and therefore are not included in the common API for basic or administration functions.

When a service-specific command is common among two or more classes of service providers, the syntax of the command is as similar as possible across all services, since a major objective of the Extensions for Financial Services is to standardize command codes and structures for the broadest variety of services. For example, using the **WFSExecute** function, the commands to read data from various services are as similar as possible to each other in their syntax and data structures.

In general, the specific command set for a service class is defined as the union of the specific capabilities likely to be provided by the developers of the services of that class; thus any particular device will normally support only a subset of the defined command set.

There are three cases in which a service provider may receive a service-specific command that it does not support:

- The requested capability is defined for the class of service providers by the XFS specification, the particular vendor implementation of that service does not support it, and the unsupported capability is ***not*** considered to be fundamental to the service. In this case, the service provider returns a successful completion, but does no operation. An example would be a request from an application to turn on a control indicator on a passbook printer; the service provider recognizes the command, but since the passbook printer it is managing does not include that indicator, the service provider does no operation and returns a successful completion to the application.
- The requested capability is defined for the class of service providers by the XFS specification, the particular vendor implementation of that service does not support it, and the unsupported capability ***is*** considered to be fundamental to the service. In this case, a WFS_ERR_UNSUPP_COMMAND error is returned to the calling

application. An example would be a request from an application to a cash dispenser to dispense coins; the service provider recognizes the command but, since the cash dispenser it is managing dispenses only notes, returns this error.

- The requested capability is ***not*** defined for the class of service providers by the XFS specification. In this case, a WFS_ERR_INVALID_COMMAND error is returned to the calling application.

This design allows implementation of applications that can be used with a range of services that provide differing subsets of the functionalities that are defined for their service class. Applications may use the **WFSGetInfo** and **WFSAsyncGetInfo** commands to inquire about the capabilities of the service they are about to use, and modify their behavior accordingly, or they may use functions and then deal with WFS_ERR_UNSUPP_COMMAND error returns to make decisions as to how to use the service.

2. Card Embossing Units

This section describes the functions provided by a generic card embossing unit (CEU). These descriptions include definitions of the service-specific commands that can be issued, using the **WFSAsyncExecute**, **WFSExecute**, **WFSGetInfo** and **WFSAsyncGetInfo** functions.

Embossing card units are generally viewed by XFS as compound devices with the following capabilities and features:

- Embossing of magnetic stripe card/ smart card.
- Reading/encoding magnetic stripe tracks 1, 2, and 3.
- Reading/writing smart card.
- LCD display/ keypad input.

The XFS services supporting the various embossing card unit components are outlined as follows:

- Embossing of magnetic stripe card/ smart card – Card Embossing Unit (CEU) service.
- Reading/encoding magnetic stripe tracks 1, 2, and 3 – ID Card (IDC) service, however when combined encoding/ embossing is performed the CEU service class is used.
- Reading/writing smart cards - ID Card (IDC) service, however when combined writing smart card/ embossing is performed the CEU service class is used.
- LCD display/ keypad input – Text Terminal Unit (TTU) service.

3. References

1. XFS Application Programming Interface (API)/Service Provider Interface (SPI), Programmer’s Reference Revision 3.00, October 18, 2000

4. Info Commands

4.1 WFS_INF_CEU_STATUS

Description This command reports the full range of information available, including the information that is provided either by the service provider or directly from the device.

Input Param None.

Output Param

```
LPWFSCEUSTATUS      lpStatus;

typedef struct _wfs_ceu_status
   {
   WORD          fwDevice;
   WORD          fwMedia;
   WORD          fwRetainBin;
   WORD          fwOutputBin;
   WORD          fwInputBin;
   USHORT        usTotalCards;
   USHORT        usOutputCards;
   USHORT        usRetainCards;
   LPSTR         lpszExtra;
   } WFSCEUSTATUS, * LPWFSCEUSTATUS;
```

fwDevice
Specifies the state of the ID card device as one of the following flags:

Value	Meaning
WFS_CEU_DEVONLINE	The device is present, powered on and online (i.e., operational, not busy processing a request and not in an error state).
WFS_CEU_DEVOFFLINE	The device is offline (e.g., the operator has taken the device offline by turning a switch or pulling out the device).
WFS_CEU_DEVPOWEROFF	The device is powered off or physically not connected.
WFS_CEU_DEVNODEVICE	There is no device intended to be there; e.g. this type of self service machine does not contain such a device or it is internally not configured.
WFS_CEU_DEVHWERROR	The device is present but inoperable due to a hardware fault that prevents it from being used.
WFS_CEU_DEVUSERERROR	The device is present but a person is preventing proper device operation. The application should suspend the device operation or remove the device from service until the service provider generates a device state change event indicating the condition of the device has changed e.g. the error is removed (WFS_CEU_DEVONLINE) or a permanent error condition has occurred (WFS_CEU_DEVHWERROR).
WFS_CEU_DEVBUSY	The device is busy and unable to process an execute command at this time

fwMedia
Specifies the state of the ID card unit as one of the following flags:

Value	Meaning
WFS_CEU_MEDIAPRESENT	Media is present in the device, not in the entering position and not jammed.
WFS_CEU_MEDIANOTPRESENT	Media is not present in the device and not at the entering position.
WFS_CEU_MEDIAJAMMED	Media is jammed in the device; operator intervention is required.
WFS_CEU_MEDIANOTSUPP	Capability to report media position is not supported by the device.

WFS_CEU_MEDIAUNKNOWN	The media state cannot be determined with the device in its current state (e.g., the value of *fwDevice* is WFS_CEU_DEVNODEVICE, WFS_CEU_DEVPOWEROFF, WFS_CEU_DEVOFFLINE, or WFS_CEU_DEVHWERROR).
WFS_CEU_MEDIAENTERING	Media is at the entry/exit slot.
WFS_CEU_MEDIATOPPER	Topper failure.
WFS_CEU_MEDIAINHOPPER	Card is positioned in input bin.
WFS_CEU_MEDIAOUTHOPPER	Card is positioned in output bin.
WFS_CEU_MEDIAMSRE	Encoding failure.
WFS_CEU_MEDIARETAINED	Card is positioned in retain bin.

fwRetainBin
Specifies the state of the CEU retain bin as one of the following flags:

Value	Meaning
WFS_CEU_RETAINBINOK	The retain bin is not full.
WFS_CEU_RETAINBINFULL	The retain bin is full.
WFS_CEU_RETAINBINHIGH	The retain bin is nearly full.
WFS_CEU_RETAINBINNOTSUPP	The retain bin state can not be reported.

fwOutputBin
Specifies the state of the Embossing unit output bin as one of the flags:

Value	Meaning
WFS_CEU_OUTPUTBINOK	The output bin is not full.
WFS_CEU_OUTPUTBINFULL	The output bin is full.
WFS_CEU_OUTPUTBINHIGH	The output bin is nearly full.
WFS_CEU_OUTPUTNOTSUPP	The output bin state can not be reported.

fwInputBin
Specifies the state of the Embossing unit input bin as one of the flags:

Value	Meaning
WFS_CEU_INPUTBINOK	The input bin is not full.
WFS_CEU_INPUTBINEMPTY	The input bin is empty.
WFS_CEU_INPUTBINLOW	The input bin is nearly empty.
WFS_CEU_INPUTNOTSUPP	The input bin state can not be reported.

usTotalCards
The total number of cards, including those in input bin, output bin, and retain bin.

usOutputCards
The total number of output bin cards.

usRetainCards
The total number of retain bin cards.

lpszExtra
Points to a list of vendor-specific, or any other extended, information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string is null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

4.2 WFS_INF_CEU_CAPABILITIES

Description This command is used to retrieve the capabilities of the Embossing Card Unit.

Input Param None.

Output Param `LPWFSCEUCAPS lpCaps;`

```
typedef struct _wfs_ceu_caps
   {
   WORD          wClass;
   BOOL          bCompound;
   BOOL          bCompareMagneticStripe;
   BOOL          bMagneticStripeRead;
   BOOL          bMagneticStripeWrite;
   BOOL          bChipIO;
   WORD          wChipProtocol;
   LPSTR         lpszExtra;
   } WFSCEUCAPS, * LPWFSCEUCAPS;
```

wClass
Specifies the logical service class; value is WFS_SERVICE_CLASS_CEU.

bCompound
Specifies whether the logical device is part of a compound physical device and is either TRUE or FALSE.

bCompareMagneticStripe
Indicates whether CEU has capability of comparing magnetic stripe contents (TRUE) as a prerequisite for an encoding or embossing operation.

bMagneticStripeRead
Indicates whether CEU has magnetic stripe reading capability and is either TRUE or FALSE.

bMagneticStripeWrite
Indicates whether CEU has magnetic stripe writing capability and is either TRUE or FALSE.

bChipIO
Indicates whether CEU has smart card updating capability and is either TRUE or FALSE.

wChipProtocol
Specifies the chip card protocols that are supported by the service provider as a combination of the following flags:

Value	Meaning
WFS_CEU_NOTSUPP	The CEU card unit can not handle chip cards.
WFS_CEU_CHIPT0	The CEU card unit can handle the T=0 protocol.
WFS_CEU_CHIPT1	The CEU card unit can handle the T=1 protocol.
WFS_CEU_CHIPT2	The CEU card unit can handle the T=2 protocol.
WFS_CEU_CHIPT3	The CEU card unit can handle the T=3 protocol.
WFS_CEU_CHIPT4	The CEU card unit can handle the T=4 protocol.
WFS_CEU_CHIPT5	The CEU card unit can handle the T=5 protocol.
WFS_CEU_CHIPT6	The CEU card unit can handle the T=6 protocol.
WFS_CEU_CHIPT7	The CEU card unit can handle the T=7 protocol.
WFS_CEU_CHIPT8	The CEU card unit can handle the T=8 protocol.
WFS_CEU_CHIPT9	The CEU card unit can handle the T=9 protocol.
WFS_CEU_CHIPT10	The CEU card unit can handle the T=10 protocol.
WFS_CEU_CHIPT11	The CEU card unit can handle the T=11 protocol.
WFS_CEU_CHIPT12	The CEU card unit can handle the T=12 protocol.
WFS_CEU_CHIPT13	The CEU card unit can handle the T=13 protocol.
WFS_CEU_CHIPT14	The CEU card unit can handle the T=14 protocol.
WFS_CEU_CHIPT15	The CEU card unit can handle the T=15 protocol.

lpszExtra
Points to a list of vendor-specific, or any other extended information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string is null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

4.3 WFS_INF_CEU_FORM_LIST

Description This command is used to retrieve the list of forms available on the device.

Input Param None.

Output Param
```
LPSTR         lpszFormList;
```

lpszFormList
Pointer to a list of null-terminated form names, with the final name terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments None.

4.4 WFS_INF_CEU_MEDIA_LIST

Description This command is used to retrieve the list of media definitions available on the device.

Input Param None.

Output Param
```
LPSTR         lpszMediaList;
```

lpszMediaList
Pointer to a list of null-terminated media names, with the final name terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments None.

4.5 WFS_INF_CEU_QUERY_FORM

Description This command is used to retrieve details of the definition of a specified CEU form. The WFS_INF_CEU_QUERY_FORM does not currently contain any form attributes, however is retained for future expansion.

Input Param
```
LPSTR         lpszFormName;
```

lpszFormName
Points to the null-terminated form name on which to retrieve details.

Output Param
```
LPWFSCEUFORM lpForm;
```

```
typedef struct _wfs_ceu_form
   {
      LPSTR     lpszFormName;
      LPSTR     lpszFields;
      } WFSCEUFORM, * LPWFSCEUFORM;
```

lpszFormName
Specifies the null-terminated name of the form.

lpszFields
Pointer to a list of null-terminated field names, with the final name terminating with two null characters.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_CEU_FORMNOTFOUND	The specified form cannot be found.
WFS_ERR_CEU_FORMINVALID	The specified form is invalid.

Comments None.

4.6 WFS_INF_CEU_QUERY_MEDIA

Description This command is used to retrieve details of the definition of a specified media.

Input Param

```
LPSTR          lpszMediaName;
```

lpszMediaName
Pointer to the null-terminated media name about which to retrieve details.

Output Param

```
LPWFSCEUFRMMEDIA    lpFormMedia;

  typedef struct _wfs_ceu_frm_media
      {
      WORD       fwMediaType;
      WORD       wBase;
      WORD       wUnitX;
      WORD       wUnitY;
      WORD       wSizeWidth;
      WORD       wSizeHeight;
      WORD       wEmbossAreaX;
      WORD       wEmbossAreaY;
      WORD       wEmbossAreaWidth;
      WORD       wEmbossAreaHeight;
      WORD       wRestrictedAreaX;
      WORD       wRestrictedAreaY;
      WORD       wRestrictedAreaWidth;
      WORD       wRestrictedAreaHeight;
      } WFSCEUFRMMEDIA, * LPWFSCEUFRMMEDIA;
```

fwMediaType
Specifies the type of media as one of the following flags:

Value	Meaning
WFS_CEU_MEDIAECARD	Embossible card media.

wBase
Specifies the base unit of measurement of the form and can be one of the following:

Value	Meaning
WFS_CEU_INCH	The base unit is inches.
WFS_CEU_MM	The base unit is millimeters.
WFS_CEU_ROWCOLUMN	The base unit is rows and columns.

wUnitX
Specifies the horizontal resolution of the base units as a fraction of the *wBase* value. For example, a value of 16 applied to the base unit WFS_CEU_INCH means that the base horizontal resolution is 1/16".

wUnitY
Specifies the vertical resolution of the base units as a fraction of the *wBase* value. For example, a value of 10 applied to the base unit WFS_CEU_MM means that the base vertical resolution is 0.1 mm.

wSizeWidth
Specifies the width of the media in terms of the base horizontal resolution.

wSizeHeight
Specifies the height of the media in terms of the base vertical resolution.

wEmbossAreaX
Specifies the horizontal offset of the Card Emboss area relative to the top left corner of the media in terms of the base horizontal resolution.

wEmbossAreaY
Specifies the vertical offset of the Card Emboss area relative to the top left corner of the media in terms of the base vertical resolution.

wEmbossAreaWidth
Specifies the Card Emboss area width of the media in terms of the base horizontal resolution.

WEmbossAreaHeight
Specifies the Card Emboss area height of the media in terms of the base vertical resolution.

wRestrictedAreaX
Specifies the horizontal offset of the restricted area relative to the top left corner of the media in terms of the base horizontal resolution.

wRestrictedAreaY
Specifies the vertical offset of the restricted area relative to the top left corner of the media in terms of the base vertical resolution.

wRestrictedAreaWidth
Specifies the restricted area width of the media in terms of the base horizontal resolution.

wRestrictedAreaHeight
Specifies the restricted area height of the media in terms of the base vertical resolution.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_CEU_MEDIANOTFOUND	The specified media definition cannot be found.
WFS_ERR_CEU_MEDIAINVALID	The specified media definition is invalid.

Comments None.

4.7 WFS_INF_CEU_QUERY_FIELD

Description This function is used to retrieve details on the definition of a single or all fields on a specified form.

Input Param `LPWFSCEUQUERYFIELD lpQueryField;`

```
typedef struct _wfs_ceu_query_field
    {
    LPSTR           lpszFormName;
    LPSTR           lpszFieldName;
    } WFSCEUQUERYFIELD, * LPWFSCEUQUERYFIELD;
```

lpszFormName
Points to the null-terminated form name.

lpszFieldName
Points to the null-terminated name of the field about which to retrieve details. If this value is NULL, then retrieve details for all fields on the form.

Output Param `LPWFSFRMFIELD *     lppFields;`

lppFields
Pointer to a null-terminated array of pointers to field definition structures:

```
typedef struct _wfs_ceu_frm_field
    {
    LPSTR     lpszFieldName;
    WORD      fwType;
    WORD      fwClass;
    LPSTR     lpszInitialValue;
    LPSTR     lpszFormat;
    } WFSCEUFRMFIELD, * LPWFSCEUFRMFIELD;
```

lpszFieldName
Pointer to the null-terminated field name.

fwType
Specifies the type of field and can be one of the following:

Value	Meaning
WFS_CEU_FIELDTEXT	A text field.
WFS_CEU_FIELDOCR	An Optical Character Recognition (OCR) field.

fwClass
Specifies the class of the field and can be one of the following:

Value	Meaning
WFS_CEU_CLASSSTATIC	The field data cannot be set by the application.
WFS_CEU_CLASSOPTIONAL	The field data can be set by the application.
WFS_CEU_CLASSREQUIRED	The field data must be set by the application.

lpszInitialValue
The initial value of the field when the field is written as output.

lpszFormat
Format string as defined in the form for this field.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_CEU_FORMNOTFOUND	The specified form cannot be found.
WFS_ERR_CEU_FIELDNOTFOUND	The specified field cannot be found.

5. Execute Commands

5.1 WFS_CMD_CEU_EMBOSS_CARD

Description This command is used to emboss an identification card by merging the supplied variable field data with the defined form and field data specified in the form. Optionally the magnetic stripe can be read and verified before being encoded, or a smart card can be updated.

The ATR of the chip must be obtained before issuing this command by issuing the ID Card class WFS_CMD_IDC_READ_RAW_DATA command.

Input Param

```
LPWFSCEUEMBOSSCARD  lpEmbossCard;
```

```
typedef struct _wfs_ceu_emboss_card
   {
   LPSTR     lpszFormName;
   LPSTR     lpszMediaName;
   LPSTR     lpszFields;
   LPSTR     lpszCompareFormIOFormName;
   LPSTR     lpszCompareFormIOTrackData;
   LPSTR     lpszFormIOFormName;
   LPSTR     lpszFormIOTrackData;
   WORD      wChipProtocol;
   ULONG     ulChipDataLength;
   LPBYTE    lpbChipData;
   } WFSCEUEMBOSSCARD, * LPWFSCEUEMBOSSCARD;
```

lpszFormName
Pointer to the null-terminated form name.

lpszMediaName
Pointer to the null-terminated media name.

lpszFields
Pointer to a series of "<FieldName>=<FieldValue>" strings, where each string is null-terminated with the final string terminating with two null characters. If the field is an index field, then the syntax of the string is instead "<FieldName>**[**<index>**]**=<FieldValue>", where <index> specifies the zero-based element of the index field.

lpszCompareFormIOFormName
lpszCompareFormIOFormName and lpszCompareFormIOTrackData are used collectively when the contents of the magnetic stripe are being read and verified before the card is embossed or the magnetic stripe is encoded. Points to the name of the magnetic stripe form to be used, as defined in the IDC service class.

lpszCompareFormIOTrackData
Points to the data to be used in the form.

lpszFormIOFormName
lpszFormIOFormName and lpszFormIOTrackData are used collectively when the magnetic stripe is being encoded (after a successful magnetic stripe compare operation) and during the emboss operation. Points to the name of the form to be used, as defined in the IDC service class.

lpszFormIOTrackData
Points to the data to be used in the form.

wChipProtocol
wChipProtocol, ulChipDataLength, and lpbChipData are used collectively when the smart card is being updated during the emboss operation. If this parameter equals zero then the smart card should not be updated during the emboss operation. Possible other values are:

Value	Meaning
WFS_CEU_CHIPT0	Use the T=0 protocol to communicate with the chip.
WFS_CEU_CHIPT1	Use the T=1 protocol to communicate with the chip.
WFS_CEU_CHIPT2	Use the T=2 protocol to communicate with the chip.
WFS_CEU_CHIPT3	Use the T=3 protocol to communicate with the chip.
WFS_CEU_CHIPT4	Use the T=4 protocol to communicate with the chip.

WFS_CEU_CHIPT5	Use the T=5 protocol to communicate with the chip.
WFS_CEU_CHIPT6	Use the T=6 protocol to communicate with the chip.
WFS_CEU_CHIPT7	Use the T=7 protocol to communicate with the chip.
WFS_CEU_CHIPT8	Use the T=8 protocol to communicate with the chip.
WFS_CEU_CHIPT9	Use the T=9 protocol to communicate with the chip.
WFS_CEU_CHIPT10	Use the T=10 protocol to communicate with the chip.
WFS_CEU_CHIPT11	Use the T=11 protocol to communicate with the chip.
WFS_CEU_CHIPT12	Use the T=12 protocol to communicate with the chip.
WFS_CEU_CHIPT13	Use the T=13 protocol to communicate with the chip.
WFS_CEU_CHIPT14	Use the T=14 protocol to communicate with the chip.
WFS_CEU_CHIPT15	Use the T=15 protocol to communicate with the chip.

ulChipDataLength
Specifies the length of the following field *lpbChipData*.

lpbChipData
Points to the data sent to the chip.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_CEU_FORMNOTFOUND	The specified form definition cannot be found.
WFS_ERR_CEU_FORMINVALID	The specified form definition is invalid.
WFS_ERR_CEU_MEDIANOTFOUND	The specified media definition cannot be found.
WFS_ERR_CEU_MEDIAINVALID	The specified media definition is invalid.
WFS_ERR_CEU_NOMEDIA	There is no card inside the device.
WFS_ERR_CEU_MEDIAOVERFLOW	The form overflowed the media.
WFS_ERR_CEU_IDC_FORMNOTFOUND	The specified IDC form definition cannot be found.
WFS_ERR_CEU_IDC_FORMINVALID	The specified IDC form definition is invalid.
WFS_ERR_CEU_INVALIDDATA	An error occurred while communicating with the chip.
WFS_ERR_CEU_PROTOCOLNOTSUPP	The protocol used was not supported by the service provider.
WFS_ERR_CEU_ATRNOTOBTAINED	The ATR was not obtained by issuing the IDC class WFS_CMD_CEU_READ_RAW_DATA command.
WFS_ERR_CEU_FIELDSPECFAILURE	The syntax of the *lpszFields* member is invalid.
WFS_ERR_CEU_FIELDERROR	An error occurred while processing a field, causing termination of the emboss request. An execute event WFS_EXEE_CEU_FIELDERROR is posted with the details.
WFS_ERR_CEU_EMBOSSFAILURE	A failure has occurred during Emboss processing. A service event WFS_EXEE_CEU_EMBOSS_FAILURE is posted with details

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_SRVE_CEU_INPUTBINTHRESHOLD	Input bin is nearly empty.
WFS_SRVE_CEU_OUTPUTBINTHRESHOLD	Output bin is nearly full.
WFS_SRVE_CEU_RETAINBINTHRESHOLD	Retain bin is nearly full.
WFS_EXEE_CEU_EMBOSS_FAILURE	A card embossing failure has occurred.
WFS_EXEE_CEU_FIELDERROR	A fatal error occurred while processing a field.
WFS_EXEE_CEU_FIELDWARNING	A non-fatal error occured while processing a field.

WFS_EXEE_CEU_MEDIAREMOVED	This event is generated when a card is removed before completion of a write operation.

Comments None

5.2 WFS_CMD_CEU_RESET

Description Sends a service reset to the service provider. Any media found in the device will be captured into the specified bin (depending on hardware). The WFS_SRVE_CEU_MEDIADETECTED event will indicate that media was found in the device on Reset and will indicate the position and status of the media following completion of the command.

Input Param LPWORD lpwCeuMediaControl;

Specifies the action that should be done if media is detected during the Reset operation, as one of the following values

Value	Meaning
WFS_CEU_CTRLTOINPUTBIN	Any media detected should be moved to the input bin.
WFS_CEU_CTRLTOOUTPUTBIN	Any media detected should be moved to the output bin.
WFS_CEU_CTRLTORETAINBIN	Any media detected should be moved to the retain bin.

Output Param None.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_SRVE_CEU_OUTPUTBINTHRESHOLD	Output bin is nearly full.
WFS_SRVE_CEU_RETAINBINTHRESHOLD	Retain bin is nearly full.
WFS_SRVE_CEU_MEDIADETECTED	Media was detected in the device during a reset.

Comments This command is used by an application control program to cause a device to reset itself to a known good condition.

6. Events

6.1 WFS_SVRE_CEU_INPUTBINTHRESHOLD

Description This service event specifies that the input bin holding the input cards is nearly empty, requiring operator intervention soon.

Event Param
```
LPWORD        lpwInputBin;
```

lpfwInputBin
Specifies the state of the CEU unit input bin as one of the following flags:

Value	Meaning
WFS_CEU_INPUTBINOK	The input bin of the CEU unit is full.
WFS_CEU_INPUTBINLOW	The input bin of the CEU unit is low.
WFS_CEU_INPUTBINEMPTY	The input bin of the CEU unit is empty.

6.2 WFS_SVRE_CEU_OUTPUTBINTHRESHOLD

Description This service event specifies that the output bin holding embossed cards is nearly full, requiring operator intervention soon.

Event Param
```
LPWORD        lpwOutputBin;
```

lpfwOutputBin
Specifies the state of the CEU unit output bin as one of the following flags:

Value	Meaning
WFS_CEU_OUTPUTBINOK	The output bin of the CEU unit was emptied.
WFS_CEU_OUTPUTBINFULL	The output bin of the CEU unit is full.
WFS_CEU_OUTPUTBINHIGH	The output bin of the CEU unit is nearly full.

6.3 WFS_SVRE_CEU_RETAINBINTHRESHOLD

Description This service event specifies that the retain bin is nearly full, requiring operator intervention soon.

Event Param
```
LPWORD        lpwRetainBin;
```

lpfwRetainBin
Specifies the state of the ID card unit retain bin as one of the following flags:

Value	Meaning
WFS_CEU_RETAINBINOK	The retain bin of the CEU unit was emptied.
WFS_CEU_RETAINBINFULL	The retain bin of the CEU unit is full.
WFS_CEU_RETAINBINHIGH	The retain bin of the CEU unit is nearly full.

6.4 WFS_EXEE_CEU_FIELDERROR

Description This event specifies that a fatal error has occurred while processing a field.

Event Param
```
LPWFSCEUFIELDFAIL   lpFieldFail;

typedef struct _wfs_ceu_field_failure
   {
   LPSTR        lpszFormName;
   LPSTR        lpszFieldName;
   WORD         wFailure;
   } WFSCEUFIELDFAIL, * LPWFSCEUFIELDFAIL;
```

lpszFormName
Points to the null-terminated form name.
lpszFieldName
Points to the null-terminated field name.

wFailure
Specifies the type of failure and can be one of the following:

Value	Meaning
WFS_CEU_FIELDREQUIRED	The specified field ***must*** be supplied by the application.
WFS_CEU_FIELDSTATICOVWR	The specified field is static and thus ***cannot*** be overwritten by the application.
WFS_CEU_FIELDOVERFLOW	The value supplied for the specified fields is too long.
WFS_CEU_FIELDNOTFOUND	The specified field does not exist.
WFS_CEU_FIELDNOTREAD	The specified field is not an input field.
WFS_CEU_FIELDNOTWRITE	An attempt was made to write to an input field.
WFS_CEU_FIELDHWERROR	The specified field uses special hardware (e.g., OCR) and an error occurred.
WFS_CEU_FIELDTYPENOTSUPPORTED	The form field type is not supported with device.

Comments None.

6.5 WFS_EXEE_CEU_FIELDWARNING

Description This event is used to specify that a non-fatal error has occurred while processing a field.

Event Param `LPWFSPTRFIELDFAIL    lpFieldFail;`

as defined in the section describing WFS_EXEE_CEU_FIELDERROR.

Comments None.

6.6 WFS_EXEE_CEU_MEDIAREMOVED

Description This event is generated when a card is removed before completion of a write operation.

Event Param None.

Comments None.

6.7 WFS_SRVE_CEU_MEDIADETECTED

Description This event is generated when a media is detected in the device during a reset operation.

Event Param WORD wPosition;

w*Position*
Specifies the media position after the reset operation, as one of the following values:

Value	Meaning
WFS_CEU_MEDIARETAINED	The media was successfully retained during the reset operation.
WFS_CEU_MEDIAREMOVED	The media was removed during the reset operation.
WFS_CEU_MEDIAJAMMED	The media is jammed in the device.
WFS_CEU_MEDIAUNKNOWN	The media is in an unknown position.

6.8 WFS_EXEE_CEU_EMBOSS_FAILURE

Description This service event is used to specify that an error has occurred during processing of a WFS_CMD_CEU_EMBOSS_CARD execute command.

Event Param LPWORD lpwEmbossFailure;

Specified as one of the following flags:

Value	Meaning
WFS_CEU_STEPPER_ERROR	Stepper hardware error.
WFS_CEU_TOPPER_FOIL_BREAK	Topper foil has broken.
WFS_CEU_CARD_FEED_ERROR	Card feed failure.
WFS_CEU_MAGNETIC_STRIPE_ERROR	Magnetic stripe read/write error.
WFS_CEU_RETAIN_BIN_FULL	Retain bin is full.
WFS_CEU_OUTPUT_BIN_FULL	Output bin is full.
WFS_CEU_COVER_OPEN	Device cover is open.
WFS_CEU_TOPPER_JAM	Topper has jammed.
WFS_CEU_STACKER_ERROR	Stacker error either inside device or in output bin.
WFS_CEU_SYSTEM_ERROR	Unknown system error.
WFS_CEU_OCR_ERROR	OCR unit failure.
WFS_CEU_EMBOSS_LIMITS_EXCEEDED	Embossing limits exceeded.
WFS_CEU_COMMUNICATIONS_FAILURE	Communications failure.
WFS_CEU_DATA_FORMAT_ERROR	Communications data format error.
WFS_CEU_BUFFER_OVERRUN	Buffer overrun.
WFS_CEU_PRE_ENCODE_READ_ERROR	Pre-encode read error.
WFS_CEU_PRE_ENCODE_DATA_MATCH_ERROR	Data has failed to compare during pre-encode data match step.
WFS_CEU_INPUT_BIN_EMPTY	Input bin is empty.
WFS_CEU_DEVICE_BUSY	Device is busy, unable to emboss card.

Comments None.

7. Embossing Form, Field and Media Definitions

This section outlines the format of the embossing definitions of forms and the fields within them.

7.1 Definition Syntax

The syntactic rules for form, field and media definitions are as follows:

• White space	space, tab
• Line continuation	backslash (\)
• Line termination	CR, LF, CR/LF; line termination ends a “keyword section” (a keyword and its value[s])
• Keywords	must be all upper case
• Names	(field/media/font names) any case; case is preserved; service providers are case sensitive
• Strings	all strings must be enclosed in double quote characters ("); to include a double quote in a string, “escape” with a forward slash (/")
• Comments	start with two forward slashes (//), end at line termination

Other notes:

- If a keyword is present, all its values must be specified; default values are used only if the keyword is absent.
- Values that are character strings are marked with asterisks in the definitions below, and must be quoted as specified above.

7.2 Embossing Form and Media Measurements

The UNIT keyword sections of the form and media definitions specify the base horizontal and vertical resolution as follows:

- the *base* value specifies the base unit of measurement
- the x and y values specify the horizontal and vertical resolution as fractions of the base value (e.g., an *x* value of 10 and a base value of MM means that the base horizontal resolution is 0.1mm).

The base resolutions thus defined by the UNIT keyword section of the ***form*** definition are used as the units of the form definition keyword sections:

- SIZE (*width* and *height* values)
- ALIGNMENT (*xoffset* and *yoffset* values)

and of the field definition keyword sections:

- POSITION (*x* and *y* values)
- SIZE (*width* and *height* values)

The base resolutions thus defined by the UNIT keyword section of the ***media*** definition are used as the units of the media definition keyword sections:

- SIZE (*width* and *height* values)
- EMBOSSAREA (*x*, *y*, *width* and *height* values)
- RESTRICTED (*x*, *y*, *width* and *height* values)

7.3 Embossing Form Definition

XFSFORM		*formname*	
BEGIN			
(required)	**UNIT**	*base,*	Base resolution unit for form definition MM INCH ROWCOLUMN
		x,	Horizontal base unit fraction
		y	Vertical base unit fraction
(required)	**SIZE**	width,	Width of form
		height	Height of form
	ALIGNMENT	alignment,	Alignment of the form on the physical medium: TOPLEFT (default) TOPRIGHT BOTTOMLEFT BOTTOMRIGHT
		xoffset,	Horizontal offset relative to the horizontal alignment specified by alignment. Always specified as a positive value (i.e., if aligned to the right side of the medium, means offset the form to the left). (default = 0)
		yoffset	Vertical offset relative to the vertical alignment specified by alignment. Always specified as a positive value (i.e., if aligned to the bottom of the medium, means offset the form upward). (default = 0)
	VERSION	*major,*	Major version number
		minor,	Minor version number
		date,*	Creation/modification date
		*author**	Author of form
(required)	**LANGUAGE**	*languageID*	Language used in this form – a 16 bit value (LANGID) which is a combination of a primary (10 bits) and a secondary (6 bits) language ID (This is the standard language ID in the Win32 API; standard macros support construction and decomposition of this composite ID)
	COPYRIGHT	*copyright**	Copyright entry
	TITLE	*title**	Title of form
	COMMENT	*comment**	Comment section
	USERPROMPT	*prompt**	Prompt string for user interaction
	[XFSFIELD	*fieldname*	One field definition (as defined in the next section) for each field in the form

	BEGIN
	. . .
	END]		

END			

7.4 Embossing Field Definition

XFSFIELD		*fieldname*	
BEGIN			
(required)	**POSITION**	*x,* *y*	Horizontal position (relative to left or right side of form, depending upon HPOSITION keyword) Vertical position (relative to top or bottom of form, depending upon VPOSITION keyword)
	HPOSITION		Horizontal field positioning relative to: LEFT (default) RIGHT
	VPOSITION		Vertical field positioning relative to: TOP BOTTOM (default)
	SIDE		Side of card: FRONT (default) BACK
(required)	**SIZE**	*width,* *height*	Field width Field height
	TYPE	*fieldtype*	Type of field: TEXT (default) OCR
	CLASS	*class*	Field class OPTIONAL (default) STATIC REQUIRED
	CASE	*case*	Convert field contents to NOCHANGE (default) UPPER LOWER
	HORIZONTAL	*justify*	Horizontal alignment of field contents LEFT (default) RIGHT CENTER JUSTIFY
	VERTICAL	*justify*	Vertical alignment of field contents BOTTOM (default) CENTER TOP
font definition information	**FONT**	*fontname**	Font name; in some cases this predefines the following parameters:
	POINTSIZE	*pointsize*	Point size
	CPI	*cpi*	Characters per inch
	LPI	*lpi*	Lines per inch
	FORMAT	*formatstring**	This is an application defined input field describing how the application should format the data. This may be interpreted by the service provider.
	INITIALVALUE	*value**	Initial value
END			

7.5 Media Definition

The media definition determines those characteristics that result from the combination of a particular media type together with a particular vendor's identification card or smart card. The aim is to make it easy to move forms between different vendor’s identification cards or smart cards which might have different constraints on how they handle a specific media type. It is the service provider's responsibility to ensure that the form definition does not specify the embossing of any fields that conflict with the media definition. An example of such a conflict might be that the form definition asks for a field to be embossed in an area that the media definition defines as a restricted area, such as on the chip of a smart card.

XFSMEDIA		medianame*	
BEGIN			
	TYPE	type	Predefined media types are: EMBOSSCARD
(required)	**UNIT**	base, x, y,	Base resolution unit for media definition MM INCH ROWCOLUMN Horizontal base unit fraction Vertical base unit fraction
(required)	**SIZE**	width, height	Width of physical media Height of physical media
	EMBOSSAREA	x, y, width, height	Embossing area relative to top left corner of physical media (default = physical size of media)
	RESTRICTED	x, y, width, height	Restricted area relative to to top left corner of physical media (default = no restricted area)
END			

8. C-Header file

```
/******************************************************************************
*                                                                             *
* xfsceu.h    XFS – Card Embossing Unit (CEU) definitions                     *
*                                                                             *
*             Version 3.00  (10/18/00)                                        *
*                                                                             *
******************************************************************************/

#ifndef __INC_XFSCEU__H
#define __INC_XFSCEU__H

#ifdef __cplusplus
extern "C" {
#endif

#include <xfsapi.h>

/*   be aware of alignment   */
#pragma pack(push,1)


/* values of WFSCEUCAPS.wClass */

#define     WFS_SERVICE_CLASS_CEU               (12)
#define     WFS_SERVICE_CLASS_NAME_CEU          "CEU"
#define     WFS_SERVICE_CLASS_VERSION_CEU       0x0003

#define     CEU_SERVICE_OFFSET                  (WFS_SERVICE_CLASS_CEU * 100)

/* CEU Info Commands */

#define     WFS_INF_CEU_STATUS                  (CEU_SERVICE_OFFSET + 1)
#define     WFS_INF_CEU_CAPABILITIES            (CEU_SERVICE_OFFSET + 2)
#define     WFS_INF_CEU_FORM_LIST               (CEU_SERVICE_OFFSET + 3)
#define     WFS_INF_CEU_QUERY_FORM              (CEU_SERVICE_OFFSET + 4)
#define     WFS_INF_CEU_MEDIA_LIST              (CEU_SERVICE_OFFSET + 5)
#define     WFS_INF_CEU_QUERY_MEDIA             (CEU_SERVICE_OFFSET + 6)
#define     WFS_INF_CEU_QUERY_FIELD             (CEU_SERVICE_OFFSET + 7)

/* CEU Execute Commands */

#define     WFS_CMD_CEU_EMBOSS_CARD             (CEU_SERVICE_OFFSET + 1)
#define     WFS_CMD_CEU_RESET                   (CEU_SERVICE_OFFSET + 2)


/* CEU Messages */

#define     WFS_SRVE_CEU_INPUTBINTHRESHOLD      (CEU_SERVICE_OFFSET + 1)
#define     WFS_SRVE_CEU_OUTPUTBINTHRESHOLD     (CEU_SERVICE_OFFSET + 2)
#define     WFS_SRVE_CEU_RETAINBINTHRESHOLD     (CEU_SERVICE_OFFSET + 3)
#define     WFS_EXEE_CEU_FIELDERROR             (CEU_SERVICE_OFFSET + 4)
#define     WFS_EXEE_CEU_FIELDWARNING           (CEU_SERVICE_OFFSET + 5)
#define     WFS_EXEE_CEU_EMBOSS_FAILURE         (CEU_SERVICE_OFFSET + 6)

/* The following value is only defined for the WFS_SRVE_CEU_MEDIADETECTED */
#define     WFS_SRVE_CEU_MEDIAREMOVED           (CEU_SERVICE_OFFSET + 7)

#define     WFS_SRVE_CEU_MEDIADETECTED          (CEU_SERVICE_OFFSET + 8)

/* values of WFSCEUSTATUS.fwDevice */
#define     WFS_CEU_DEVONLINE                   WFS_STAT_DEVONLINE
#define     WFS_CEU_DEVOFFLINE                  WFS_STAT_DEVOFFLINE
#define     WFS_CEU_DEVPOWEROFF                 WFS_STAT_DEVPOWEROFF
#define     WFS_CEU_DEVNODEVICE                 WFS_STAT_DEVNODEVICE
#define     WFS_CEU_DEVHWERROR                  WFS_STAT_DEVHWERROR
#define     WFS_CEU_DEVUSERERROR                WFS_STAT_DEVUSERERROR
#define     WFS_CEU_DEVBUSY                     WFS_STAT_DEVBUSY

/* values of WFSCEUSTATUS.fwMedia  */

#define     WFS_CEU_MEDIAPRESENT                (1)
```

```
#define     WFS_CEU_MEDIANOTPRESENT             (2)
#define     WFS_CEU_MEDIAJAMMED                 (3)
#define     WFS_CEU_MEDIANOTSUPP                (4)
#define     WFS_CEU_MEDIAUNKNOWN                (5)
#define     WFS_CEU_MEDIAENTERING               (6)
#define     WFS_CEU_MEDIATOPPER                 (7)
#define     WFS_CEU_MEDIAINHOPPER               (8)
#define     WFS_CEU_MEDIAOUTHOPPER              (9)
#define     WFS_CEU_MEDIAMSRE                   (10)
#define     WFS_CEU_MEDIARETAINED               (11)
#define     WFS_CEU_MEDIAREMOVED                (12)

/* values of WFSCEUSTATUS.fwRetainBin  */

#define     WFS_CEU_RETAINBINOK                 (1)
#define     WFS_CEU_RETAINBINFULL               (2)
#define     WFS_CEU_RETAINBINHIGH               (3)
#define     WFS_CEU_RETAINBINNOTSUPP            (4)

/* values of WFSCEUSTATUS.fwOutputBin  */

#define     WFS_CEU_OUTPUTBINOK                 (1)
#define     WFS_CEU_OUTPUTBINFULL               (2)
#define     WFS_CEU_OUTPUTBINHIGH               (3)
#define     WFS_CEU_OUTPUTNOTSUPP               (4)

/* values of WFSCEUSTATUS.fwInputBin  */

#define     WFS_CEU_INPUTBINOK                  (1)
#define     WFS_CEU_INPUTBINEMPTY               (2)
#define     WFS_CEU_INPUTBINLOW                 (3)
#define     WFS_CEU_INPUTNOTSUPP                (4)

/* values of _wfs_ceu_frm_header.wBase , wfs_ceu_frm_media.wBase  */

#define     WFS_CEU_INCH                        (1)
#define     WFS_CEU_MM                          (2)
#define     WFS_CEU_ROWCOLUMN                   (3)

/* values of _wfs_ceu_frm_header.wAlignment */

#define     WFS_CEU_TOPLEFT                     (1)
#define     WFS_CEU_TOPRIGHT                    (2)
#define     WFS_CEU_BOTTOMLEFT                  (3)
#define     WFS_CEU_BOTTOMRIGHT                 (4)

/* values of _wfs_ceu_frm_media.fwMediaType  */

#define     WFS_CEU_MEDIAECARD                  (1)

/* values of _wfs_ceu_frm_field.fwType  */

#define     WFS_CEU_FIELDTEXT                   (1)
#define     WFS_CEU_FIELDOCR                    (2)

/* values of _wfs_ceu_frm_field.fwClass  */

#define     WFS_CEU_CLASSSTATIC                 (1)
#define     WFS_CEU_CLASSOPTIONAL               (2)
#define     WFS_CEU_CLASSREQUIRED               (3)

/* values WFSCEUFIELDFAIL.wFailure */

#define     WFS_CEU_FIELDREQUIRED               (1)
#define     WFS_CEU_FIELDSTATICOVWR             (2)
#define     WFS_CEU_FIELDOVERFLOW               (3)
#define     WFS_CEU_FIELDNOTFOUND               (4)
#define     WFS_CEU_FIELDNOTREAD                (5)
#define     WFS_CEU_FIELDNOTWRITE               (6)
#define     WFS_CEU_FIELDHWERROR                (7)
#define     WFS_CEU_FIELDTYPENOTSUPPORTED       (8)

/* values of WFSCEUEMBOSSCARD.fwChipProtocols */
```

```
#define     WFS_CEU_NOTSUPP                     0x0000
#define     WFS_CEU_CHIPT0                      0x0001
#define     WFS_CEU_CHIPT1                      0x0002
#define     WFS_CEU_CHIPT2                      0x0004
#define     WFS_CEU_CHIPT3                      0x0008
#define     WFS_CEU_CHIPT4                      0x0010
#define     WFS_CEU_CHIPT5                      0x0020
#define     WFS_CEU_CHIPT6                      0x0040
#define     WFS_CEU_CHIPT7                      0x0080
#define     WFS_CEU_CHIPT8                      0x0100
#define     WFS_CEU_CHIPT9                      0x0200
#define     WFS_CEU_CHIPT10                     0x0400
#define     WFS_CEU_CHIPT11                     0x0800
#define     WFS_CEU_CHIPT12                     0x1000
#define     WFS_CEU_CHIPT13                     0x2000
#define     WFS_CEU_CHIPT14                     0x4000
#define     WFS_CEU_CHIPT15                     0x8000

/* WFS_EXEE_CEU_EMBOSS_FAILURE  Flags  */

#define     WFS_CEU_STEPPER_ERROR               (1)
#define     WFS_CEU_TOPPER_FOIL_BREAK           (2)
#define     WFS_CEU_CARD_FEED_ERROR             (3)
#define     WFS_CEU_MAGNETIC_STRIPE_ERROR       (4)
#define     WFS_CEU_RETAIN_BIN_FULL             (5)
#define     WFS_CEU_OUTPUT_BIN_FULL             (6)
#define     WFS_CEU_COVER_OPEN                  (7)
#define     WFS_CEU_TOPPER_JAM                  (8)
#define     WFS_CEU_STACKER_ERROR               (9)
#define     WFS_CEU_SYSTEM_ERROR                (10)
#define     WFS_CEU_OCR_ERROR                   (11)
#define     WFS_CEU_EMBOSS_LIMITS_EXCEEDED      (12)
#define     WFS_CEU_COMMUNICATIONS_FAILURE      (13)
#define     WFS_CEU_DATA_FORMAT_ERROR           (14)
#define     WFS_CEU_BUFFER_OVERRUN              (15)
#define     WFS_CEU_PRE_ENCODE_READ_ERROR       (16)
#define     WFS_CEU_PRE_ENCODE_DATA_MATCH_ERROR (17)
#define     WFS_CEU_INPUT_BIN_EMPTY             (18)
#define     WFS_CEU_DEVICE_BUSY                 (19)

/* values of lpwCeuMediacontrol paramater of WFS_CMD_CEU_RESET command */
#define     WFS_CEU_CTRLTOINPUTBIN              (1)
#define     WFS_CEU_CTRLTOOUTPUTBIN             (2)
#define     WFS_CEU_CTRLTORETAINBIN             (3)

/* WOSA/XFS CEU Errors */

#define WFS_ERR_CEU_FORMNOTFOUND                (-(CEU_SERVICE_OFFSET + 1))
#define WFS_ERR_CEU_FORMINVALID                 (-(CEU_SERVICE_OFFSET + 2))
#define WFS_ERR_CEU_MEDIANOTFOUND               (-(CEU_SERVICE_OFFSET + 3))
#define WFS_ERR_CEU_MEDIAINVALID                (-(CEU_SERVICE_OFFSET + 4))
#define WFS_ERR_CEU_NOMEDIA                     (-(CEU_SERVICE_OFFSET + 5))
#define WFS_ERR_CEU_MEDIAOVERFLOW               (-(CEU_SERVICE_OFFSET + 6))
#define WFS_ERR_CEU_IDC_FORMNOTFOUND            (-(CEU_SERVICE_OFFSET + 7))
#define WFS_ERR_CEU_IDC_FORMINVALID             (-(CEU_SERVICE_OFFSET + 8))
#define WFS_ERR_CEU_INVALIDDATA                 (-(CEU_SERVICE_OFFSET + 9))
#define WFS_ERR_CEU_PROTOCOLNOTSUPP             (-(CEU_SERVICE_OFFSET + 10))
#define WFS_ERR_CEU_ATRNOTOBTAINED              (-(CEU_SERVICE_OFFSET + 11))
#define WFS_ERR_CEU_FIELDSPECFAILURE            (-(CEU_SERVICE_OFFSET + 12))
#define WFS_ERR_CEU_FIELDERROR                  (-(CEU_SERVICE_OFFSET + 13))
#define WFS_ERR_CEU_EMBOSSFAILURE               (-(CEU_SERVICE_OFFSET + 14))
#define WFS_ERR_CEU_FIELDNOTFOUND               (-(CEU_SERVICE_OFFSET + 15))


/*=================================================================*/
/* CEU Info Command Structures and variables */
/*=================================================================*/

typedef struct _wfs_ceu_status
{
    WORD      fwDevice;
    WORD      fwMedia;
    WORD      fwRetainBin;
    WORD      fwOutputBin;
    WORD      fwInputBin;
    USHORT    usTotalCards;
    USHORT    usOutputCards;
    USHORT    usRetainCards;
    LPSTR     lpszExtra;
} WFSCEUSTATUS, * LPWFSCEUSTATUS;


typedef struct _wfs_ceu_caps
{
    WORD      wClass;
    BOOL      bCompound;
    BOOL      bCompareMagneticStripe;
    BOOL      bMagneticStripeRead;
    BOOL      bMagneticStripeWrite;
    BOOL      bChipIO;
    WORD      wChipProtocol;
    LPSTR     lpszExtra;
} WFSCEUCAPS, * LPWFSCEUCAPS;


typedef struct _wfs_ceu_form
{
    LPSTR     lpszFormName;
    LPSTR     lpszFields;
} WFSCEUFORM, * LPWFSCEUFORM;


typedef struct _wfs_ceu_frm_media
{
    WORD      fwMediaType;
    WORD      wBase;
    WORD      wUnitX;
    WORD      wUnitY;
    WORD      wSizeWidth;
    WORD      wSizeHeight;
    WORD      wEmbossAreaX;
    WORD      wEmbossAreaY;
    WORD      wEmbossAreaWidth;
    WORD      wEmbossAreaHeight;
    WORD      wRestrictedAreaX;
    WORD      wRestrictedAreaY;
    WORD      wRestrictedAreaWidth;
    WORD      wRestrictedAreaHeight;
} WFSCEUFRMMEDIA, * LPWFSCEUFRMMEDIA;

typedef struct _wfs_ceu_query_field
{
    LPSTR     lpszFormName;
    LPSTR     lpszFieldName;
} WFSCEUQUERYFIELD, * LPWFSCEUQUERYFIELD;

typedef struct _wfs_ceu_frm_field
{
    LPSTR     lpszFieldName;
    WORD      fwType;
    WORD      fwClass;
    LPSTR     lpszInitialValue;
    LPSTR     lpszFormat;
} WFSCEUFRMFIELD, * LPWFSCEUFRMFIELD;

/*=================================================================*/
/* CEU Execute Command Structures */
/*=================================================================*/

typedef struct _wfs_ceu_emboss_card
{
    LPSTR     lpszFormName;
    LPSTR     lpszMediaName;
    LPSTR     lpszFields;
    LPSTR     lpszCompareFormIOFormName;
    LPSTR     lpszCompareFormIOTrackData;
    LPSTR     lpszFormIOFormName;
    LPSTR     lpszFormIOTrackData;
    WORD      wChipProtocol;
    ULONG     ulChipDataLength;
    LPBYTE    lpbChipData;
} WFSCEUEMBOSSCARD, * LPWFSCEUEMBOSSCARD;


/*=================================================================*/
/* CEU Message Structures */
/*=================================================================*/

typedef struct _wfs_ceu_field_failure
{
    LPSTR     lpszFormName;
    LPSTR     lpszFieldName;
    WORD      wFailure;
} WFSCEUFIELDFAIL, * LPWFSCEUFIELDFAIL;

#ifdef __cplusplus
}
#endif

//restore alignment
#pragma pack (pop)

#endif  /* __INC_XFSCEU__H */
```