RADIOCOMMUNICATION BUREAU	EPFDLIMITSAPI Library	**Ref** : EPFDLIMITSAPI-IE-UM
		Version : 4.2
		Author : V. PHILAVONG
	Dept./Div. : IAP/SAS	**Date** : 2023-08-24

EPFDLIMITSAPI Library

User Manual

Abstract

- Contains the description of the library's functions you can use to programmatically read the EPFD_Limits databases.

Participants

- SAS division

Point of contact

- Virasouk.philavong@itu.int

International Telecommunication Union

Place des Nations CH-1211 Geneva 20 Switzerland www.itu.int

CHAPTER 1 INTRODUCTION 4

CHAPTER 2 DATA TYPES AND STRUCTURES 5

CHAPTER 3 FUNCTIONS DIRECTORY 8

LIMITS DATA MANIPULATION FUNCTIONS 9

EPFD_Limits_Extract *10*

EPFD_Limits_OpenConnectionToDb *13*

EPFD_Limits_CloseConnectionToDb *13*

EPFD_Limits_EmptyStEpfdExams *13*

EPFD_Limits_CalculationByLatitude *14*

EPFD_Limits_GetServiceCodeFromClassOfStation *14*

CHAPTER 4 USING EPFDLIMITSAPI 15

4.1 ADVANCED TOPICS 15

4.2 SAMPLES 15

DOCUMENT CHANGE RECORD

Issue	Date	Change Description
V1	2007-06-08	Template release – V. Philavong
V2	2007-08-15	Added *ss_alt_max* in *stEpfdExamsInfo*
V3	2010-08-03	Modified EPFD_limits database (completely reviewed) Modified data structure *stEpfdExamsInfo, stEpfdLimits* to fit with new EPFD_limits Added new input parameter to **EPFD_Limits_Extract** function: *inBandwdth* (Frequency band width)
V3.1	2010-08-06	Added new input parameter to **EPFD_Limits_Extract** function: *inDateOfReceipt* (date of receipt)
V3.2	2010-08-31	Added new function **EPFD_Limits_GetServCodeFromClassOfStation**
V3.3	2011-08-17	Changed all variables declared as float to double
V3.4	2011-11-25	Added new function **EPFD_Limits_Extract_By_Examination**
V3.5	2015-08-25	Bug fix in function: *Epfd_LimitsAccess::CalulationEpfdLimitByLatitude* The calculation at latitude -90.0 returns NOVALUEDOUBLE
V4.0	2016-11-30	Modified data structure *stEpfdExamsInfo* to include the variable *rf_pattern_rr*
V4.1	2016-12-19	Updated Epfd_st.h (WRC15 SRS V8): increase the size of the sat_name field from 20 to 30.
V4.2	2023-08-24	64-bit version release

]

Chapter 1 INTRODUCTION

The **EPFDLIMITSAPI** library will give your applications the possibility to access and extract data stored in the EPFD_Limits databases. It does that through an interface defined by a set of C-style functions that are not supposed to change regardless of the format of the database accessed.

This manual gives application developers general as well as detailed information about **EPFDLIMITSAPI** functions, data types.

Therefore, it is divided into three main parts:

Chapter 2. **DATA TYPES AND STRUCTURES**

Chapter 3. **FUNCTIONS DIRECTORY**

Chapter 4. **USING EPFDLIMITSAPI**

Chapter 2 DATA TYPES AND STRUCTURES

The data types, data structures and constants are defined in the file Epfd_st.h and the prototypes of functions are defined in the file EpfdLimitsApi.h, distributed both with the library. Their contents are listed below.

```
/*****************************************************************************************************/
/* The EPFDLIMITSAPI data types, data structures and constants in Epfd_st.h file
*/
/*****************************************************************************************************/

#ifndef _EPFD_ST_
#define _EPFD_ST_

//No value
#define SCEPFD_NOVALUEFLOAT           -99999.f       /**<No value float*/
#define SCEPFD_NOVALUEDOUBLE          -99999.        /**<No value double*/
#define SCEPFD_NOVALUEULONG           0xFFFFFFFF     /**<No value unsigned long*/
#define SCEPFD_NOVALUELONG            0xFFFFFFFF     /**<No value long*/
#define SCEPFD_NOVALUEPTR             NULL           /**<No value pointer*/
#define SCEPFD_NOVALUESTR             ""             /**<No value string*/

//Arrays dimensions & lengths
#define SCEPFD_SATNAME_LEN            31      /**<Size of satellite name code WRC15-SRS_V8*/
#define SCEPFD_ADM_LEN                4       /**<Size of administration code*/

#define SCEPFD_ITURGN_LEN             4       /**<Size of itu region code*/
#define SCEPFD_SERVICE_LEN            6       /**<Size of service code*/
#define SCEPFD_STNCLS_LEN             3       /**<Size of class of station code*/
#define SCEPFD_LINKDIR_LEN            5       /**<Size of link direction code*/
#define SCEPFD_REFRR_LEN              51      /**<Size of reference provision or AR22 table name*/
#define SCEPFD_METHOD_LEN             51      /**<Size for calculation method*/
#define SCEPFD_ANT_PATTERN_CODE_LEN   13      /**<Size of antenna pattern code*/
#define SCEPFD_ANT_PATTERN_TEXT_LEN   51      /**<Size of antenna pattern text*/



typedef enum  {
        EPFD_OKAY = 0,
        EPFD_EC_UNKNOWN = -1,
        EPFD_EC_MASK_XML = -2,        /**<Value of input parameters in Masks store function is
different from values in XML file.In this case check if ntc_id, adm and ntwk_name are correct*/
        EPFD_EC_XML = -3,             /**<Problem with XML file*/
        EPFD_WC_UNKNOWN = 1
} stEpfdStatus ;       /**<Return codes and defined status*/


typedef enum   {
        EPFD_DN = 1,                  /**<Down*/
        EPFD_UP = 2,                  /**<Up*/
        EPFD_IS = 3                   /**<Inter-satellite*/
} stEpfdLinkDirectType;                       /**<Direction of calculation types*/

typedef struct{
        char  stn_cls[SCEPFD_STNCLS_LEN] ;    /**<Class of station where the Epfd limit is applied:
                                              'EC','ED', 'ER', 'EK', 'ET', 'TC', 'TD' or 'EB', 'EV',
                                              'UB', 'UV'*/
} stEpfdStnCls;        /**<Class of station codes*/

typedef struct{
        char  service[SCEPFD_SERVICE_LEN] ;   /**<Service where the Epfd limit is applied: 'FSS' or
                                              'BSS'*/
} stEpfdServCode;      /**<Sevice codes*/

typedef enum   {
        EPFD_A22 = 1,                 /**<Analysis Hard limit Article 22*/
        EPFD_97A = 2,                 /**<Analysis 9.7 A LES Triggers */
        EPFD_97B = 3                  /**<Analysis 9.7 B NGSO Triggers*/
} stEpfdExamCode;                     /**<Examination code*/

typedef struct {
        double epfd ;          /**<Equivalent power flux-density [dB(W/m²)]*/
        double time_percent ;  /**<Percentage of time during which epfd may not be exceeded*/
```

```
        double adjust_fact ;   /**<Only for special case additional Article 22, RR22.5C4, RR22.5C8 */

        double rf_beam_wdth ;  /**<Only for special case EPFD up and EPFD intersatellite Article 22,
TABLE 22-2, TABLE 22-3). Reference antenna beamwidth (degree)*/

        double rf_diam ;       /**<Only for special case Article 22, TABLE 22-4A. Reference antenna
diameter (m)*/

        double es_long_min ;   /**<Lowest longitude of earth station in GSO system [degrees]*/
        double es_long_max ;   /**<Highest longitude of earth station in GSO system [degrees]*/
        double es_lat_min ;    /**<Lowest latitude of earth station in GSO system [degrees]*/
        double es_lat_max ;    /**<Highest latitude of earth station in GSO system [degrees]*/
        double es_min_gain ;   /**<minimum of GSO earth station antenna maximum isotropic gain
[dBi]*/
        double es_min_elev ;   /**<minimum elevation angle [degrees]*/
        double gso_bss_long ;  /**<broadcasting-satellite service at specific longitude with
elevation angles greater than 5 [degrees]*/
        double gso_max_inclin ;       /**<Only for Operational limit tables. Maximum value allowed
for Orbital inclination of the geostationary satellite (degrees)*/

        double ngso_min_alt ;
        double ngso_max_alt ;

        double es_G2T_min ;           /**<minimum of rapport between the maximum antenna gain (G)
and the lowest total receiving system noise temperature (K)*/
        double es_emi_bdwidth_min ;   /**<GSO earth station emission bandwidth [MHz]*/


} stEpfdLimits ;       /**<Epfd limits bloc structure*/

typedef struct {
        unsigned long array_size ;            /**<Size of the Epfd limits array*/
        stEpfdLimits *epfdLimits;             /**<Array of epfd limits*/
} stEpfdLimitsArray ;  /**<Epfd limit array*/

typedef struct {
        char ref_rr[SCEPFD_REFRR_LEN] ;       /**<Reference table name or provision*/
        char rule_type[SCEPFD_METHOD_LEN] ;   /**<Rule type which is a reference methodology used for
the calculation of EPFD: 'full', 'operational' or 'additional' */
        unsigned long valid_from ;            /**<Date from which complete coordination/notification
information has been received by BR*/
        unsigned long valid_to ;              /**<Last date when applicable*/
        unsigned long mask_id ;               /**<ID number of EPFD examination*/
        unsigned long mask_argmt ;            /**<Mask argument: {0,1,2,3,4,5,6}*/
        double freq_min ;                     /**<Frequency minimum where the Epfd limit is applied
(GHz)*/
        double freq_max ;                     /**<Frequency maximum where the Epfd limit is applied
(GHz)*/
        char  rr_region[SCEPFD_ITURGN_LEN] ;  /**<ITU region where the Epfd limit is applied: 'XR1',
'XR2' or 'XR3'*/
        char  rr_service[SCEPFD_SERVICE_LEN] ;/**<Service where the Epfd limit is applied: 'FSS'or
'BSS'*/
        char  link_direction[SCEPFD_LINKDIR_LEN] ;   /**<Link direction where the Epfd limit is
applied: 'down', 'up' or 'is'*/
        double rf_diam ;                              /**<Reference antenna diameter (m)*/
        double rf_band_wdth ;                         /**<Reference bandwidth (kHz)*/
        char  rf_pattern[SCEPFD_ANT_PATTERN_CODE_LEN] ;      /**<Reference radiation pattern*/
        char  rf_pattern_rr[SCEPFD_ANT_PATTERN_TEXT_LEN] ;   /**<Reference radiation pattern name in
Radio Regulation*/

} stEpfdExamsInfo ;           /**<Examination info*/

/**<Epfd datas for examination */
typedef struct {
        stEpfdExamsInfo examsInfo ;                  /**<Examination info*/
        stEpfdLimitsArray limits ;                   /**<Epfd limits array*/
} stEpfdExams ;


#endif //_EPFD_ST_
```

```
/*****************************************************************************************************/
/* The EPFDLIMITSAPI functions in the file EpfdLimitsApi.h
*/
/*****************************************************************************************************/

#ifdef __cplusplus
extern "C"
{
#endif

#include "Epfd_st.h"


// The following ifdef block is the standard way of creating macros which make exporting
// from a DLL simpler. All files within this DLL are compiled with the EPFDLIMITSAPI_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any project
// that uses this DLL. This way any other project whose source files include this file see
// EPFDLIMITSAPI_API functions as being imported from a DLL, wheras this DLL sees symbols
// defined with this macro as being exported.
#ifdef EPFDLIMITSAPI_EXPORTS
#define EPFDLIMITSAPI_API __declspec(dllexport)
#else
#define EPFDLIMITSAPI_API __declspec(dllimport)
#endif

////////////////////////////////////////////////////////////////////////////////////////////
/////////////////////////////////////////AR22 LIMITS//////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////
/*!
 *      \brief  Open connection to AR22 Limits database
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_OpenConnectionToDb ( const char*
inEpfdLimitsDbPath);

/*!
 *      \brief  Close connection to AR22 Limits database
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_CloseConnectionToDb ();

/*!
 *      \brief  Extract EPFD limits from database
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_Extract(stEpfdLinkDirectType inCalcType,
       stEpfdServCode inServCode, double inFreqAssgn, double inFreqBandwdth,
       unsigned long inDateOfReceipt, stEpfdExams** outExamsArray, int* outExamsCount);

/*!
 *      \brief  Empty ExamsArray
 */
EPFDLIMITSAPI_API void WINAPI EPFD_Limits_EmptyStEpfdExams(stEpfdExams* inExamsArray,
                                                           int inExamsCount);

/*!
 *      \brief  Calculate Epfd limit in function of latitude according to the AR22 footnotes 22.5C2
and 22.5C8
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_CalculationByLatitude(double inLatitude,
                                                                        double& outEpfdLimitValue)
;

/*!
 *      \brief  Get Service code from class of station
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_GetServCodeFromClassOfStation(stEpfdStnCls
inStnCls, stEpfdServCode& outServCode);

/*!
 *      \brief  Extract EPFD limits by examination code from database
 */
EPFDLIMITSAPI_API stEpfdStatus WINAPI EPFD_Limits_Extract_By_Examination(stEpfdExamCode
inExaminationCode, stEpfdLinkDirectType inLinkDirectType, stEpfdServCode inServCode,
double inFreqAssgn, double inFreqBandwdth, double inOperatingHeight,
unsigned long inDateOfReceipt,stEpfdExams** outExamsArray, int* outExamsCount) ;

#ifdef __cplusplus
}
#endif
```

Chapter 3 FUNCTIONS DIRECTORY

This chapter contains the list of functions from the **EPFDLIMITSAPI** that are available to the application developer. It includes functions allowing to extract data stored in EPFD_Limits.mdb database.

The names of functions are beginning with **EPFD_Limits**_ and following by a description of what they use for. For example: **EPFD_Limits_Extract** (use for extracting data from EPFD_Limits database)**, EPFD_Limits_OpenConnectionToDb** (use for open the connection to EPFD_Limits database)**, etc**

The general scheme for the presentation is as follows:

Function: **[Function Name]**

Syntax: **[Function Syntax]**

Purpose: **[Function's Purpose]**

Parameters: **[Description Of The Function's Input Parameters]**

Return value: **[Description Of The Value Returned By The Function]**

Remarks: **[Additional Information]**

See Also: **[Cross Reference To Other Related Items]**

LIMITS DATA MANIPULATION FUNCTIONS

Here are the list of functions that can be called from EPFDLIMITSAPI

- **EPFD_Limits_Extract**
- **EPFD_Limits_Extract_By_Examination**
- **EPFD_Limits_OpenConnectionToDb**
- **EPFD_Limits_CloseConnectionToDb**
- **EPFD_Limits_EmptyStEpfdExams**
- **EPFD_Limits_CalculationByLatitude**
- **EPFD_Limits_GetServiceCodeFromClassOfStation**

The descriptions, in detail, of each function are presented here below:

Function: **EPFD_Limits_Extract**

Syntax: **stEpfdStatus EPFD_Limits_Extract (**stEpfdLinkDirectType *inCalcType*,
stEpfdServCode *inServCode***,**
double *inFreqAssgn*,
double *inFreqBandwdth,*
unsigned long *inDateOfReceipt,*
stEpfdExams** *outExamsArray,*
int* *outExamsCount***);**

Purpose: This function extracts EPFD limits from EPFD_limits database.

Parameters:

inLinkDirectType	stEpfdLinkDirectType	The input parameter for direction of link, it must be one of a code defined in the enumerator stEpfdLinkDirectType.
inServCode	stEpfdServCode	The input parameter service code defined in the header file Epfd_st.h.
inFreqAssgn	double	The input parameter assigned frequency. The value must be in [GHz].
inFreqBandwdth	double	The input parameter bandwidth of the frequency band, in [kHz].
inDateOfReceipt	unsigned long	The input parameter date of receipt (Format: YYYYMMDD).
outExamsArray	stEpfdExams**	The output parameter contains a list of stEpfdExams. Each stEpfdExams corresponds to a possible case that can be found in a table of limits in the RR AR22.
outExamCount	int*	The output parameter contains a number of stEpfdExams structure in *outExamsArray* (it corresponds to the number of possible cases)

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Remarks: The definition of **stEpfdLinkDirectType**, **stEpfdServCode**, **stEpfdExams** and **stEpfdStatus** are in EPFD_st.h file.

Before calling **EPFD_Limits_Extract**, the function **EPFD_Limits_OpenConnectionToDb** must be called first to establish the connection to **EPFD_Limits** database, then to close the database use **EPFD_Limits_CloseConnectionToDb**.

Note that after calling **EPFD_Limits_Extract**, the user is also responsible for deleting the allocated memory of the pointer to the structure stEpfdLimits and stEpfdExams. It can be done by calling the function **EPFD_Limits_EmptyStEpfdExams**.

Function: **EPFD_Limits_Extract_By_Examination**

Syntax: **stEpfdStatus EPFD_Limits_Extract (**stEpfdExamCode *inExaminationCode*,
stEpfdLinkDirectType *inCalcType*,
stEpfdServCode *inServCode***,**
double *inFreqAssgn*,
double *inFreqBandwdth,*
double *inOperatingHeight*,
unsigned long *inDateOfReceipt,*
stEpfdExams** *outExamsArray,*
int* *outExamsCount***);**

Purpose: This function extracts EPFD limits from EPFD_limits database.

Parameters:

inExaminationCode	stEpfdExamCode	The input parameter for the type of examination, i it must be one of a code defined in the enumerator stEpfdExamCode.
inLinkDirectType	stEpfdLinkDirectType	The input parameter for direction of link, it must be one of a code defined in the enumerator stEpfdLinkDirectType.
inServCode	stEpfdServCode	The input parameter service code defined in the header file Epfd_st.h.
inFreqAssgn	double	The input parameter assigned frequency. The value must be in [GHz].
inFreqBandwdth	double	The input parameter bandwidth of the frequency band, in [kHz].
inOperatingHeight	double	The input parameter non-GSO satellite operating altitude [Km]
inDateOfReceipt	unsigned long	The input parameter date of receipt (Format: YYYYMMDD).
outExamsArray	stEpfdExams**	The output parameter contains a list of stEpfdExams. Each stEpfdExams corresponds to a possible case that can be found in a table of limits in the RR AR22.
outExamCount	int*	The output parameter contains a number of stEpfdExams structure in *outExamsArray* (it corresponds to the number of possible cases)

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Remarks:

The definition of **stEpfdExamCode**, **stEpfdLinkDirectType**, **stEpfdServCode**, **stEpfdExams** and **stEpfdStatus** are in EPFD_st.h file.

Before calling **EPFD_Limits_Extract**, or **EPFD_Limits_Extract_By_Examination**, the function **EPFD_Limits_OpenConnectionToDb** must be called first to establish the connection to **EPFD_Limits** database, then to close the database use **EPFD_Limits_CloseConnectionToDb**.

Note that after calling **EPFD_Limits_Extract** or **EPFD_Limits_Extract_By_Examination**, the user is also responsible for deleting the allocated memory of the pointer to the structure stEpfdLimits and stEpfdExams. It can be done by calling the function **EPFD_Limits_EmptyStEpfdExams**.

Function: **EPFD_Limits_OpenConnectionToDb**

Syntax: **stEpfdStatus EPFD_Limits_OpenConnectionToDb (**const char* *inEpfdLimitsDbPath***);**

Purpose: This function opens a connection to Epfd_Limits database specified by its filename. If file is not found an error is generated.

Parameters:

inEpfdLimitsDbPath	const char*	The full path of Epfd_Limits database (i.e the location of the database file + the database file).

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Function: **EPFD_Limits_CloseConnectionToDb**

Syntax: **stEpfdStatus EPFD_Limits_CloseConnectionToDb ( );**

Purpose: The function closes open connection to Epfd_Limit database.

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Function: **EPFD_Limits_EmptyStEpfdExams**

Syntax: **void EPFD_Limits_EmptyStEpfdExams (**stEpfdExams* *inExamsArray,*
int *inExamsCount***);**

Purpose: This function deletes the allocated memory inside the structure stEpfdExams.

Parameters:

inExamsArray	stEpfdExams*	List of structure stEpfdExams to be deleted .
inExamCount	int*	Number of stEpfdExams structure to be deleted.

Return value: Void

Function: **EPFD_Limits_CalculationByLatitude**

Syntax: **stEpfdStatus EPFD_Limits_CalculationByLatitude (**double *inLatitude,*
double& *outEpfdLimitvalue***);**

Purpose: This function calculates the single-entry epfd↓ limits in function of the latitude according to the addition table specified in the footnote 22.5C.4 and 22.5C.8 of RR-AR22.

Parameters:

inLatitude	double	The input parameter Latitude value
outEpfdLimitValue	double&	The output parameter Epfd↓ limit calculated at that given latitude

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Function: **EPFD_Limits_GetServiceCodeFromClassOfStation**

Syntax: **stEpfdStatus EPFD_Limits_GetServiceCodeFromClassOfStation (**stEpfdStnCls *inStnCls*,
stEpfdServCode& *outServCode***);**

Purpose: This function returns the service code of the given class of station.

Parameters:

inStnCls	stEpfdStnCls	The input structure for class of station defined in the header file Epfd_st.h
outServCode	stEpfdServCode &	The output structure for service code defined in the header file Epfd_st.h

Return value: On success, this function returns a EPFD_OKAY defined in **stEpfdStatus** in a value ≥0. Otherwise, it returns an error value (<0).

Remarks: **EPFD_Limits_CalculationByLatitude** and **EPFD_Limits_GetServiceCodeFromClassOfStation** must be used in between **EPFD_Limits_OpenConnectionToDb** and **EPFD_Limits_CloseConnectionToDb** calls.

Chapter 4 Using EPFDLIMITSAPI

4.1 ADVANCED TOPICS

This section deals with a number of issues (calling conventions, passing parameters, ...) that the programmer needs to know in order to properly use **EPFDLIMITSAPI**, especially when he uses another programming language than **C** to interface the main program with **EPFDLIMITSAPI**.

- In its current version, **EPFDLIMITSAPI** was developed using **Microsoft Visual C++ 6.0™** and its target operating environment is Windows XP.
- When you are using the **EPFDLIMITSAPI** library from a C/C++ program you should *#include* the header file **Epfd_st.h** and **EpfdLimitsApi.h**. It contains the declaration of all functions, constants and return codes.
- Regarding **Calling Conventions**, it should to be noted that all **EPFDLIMITSAPI** functions are declared to be **WINAPI** (which is actually a *#define* for **__stdcall**).
- To run an application using the **EPFDLIMITSAPI** library you will also need some other DLLs. These DLLs will be provided in an EPFD Package.

4.2 SAMPLES

The **limits.cpp** file contains a small C++ console application code showing the use of some features of the **EPFDLIMITSAPI** library. It has been developed using the Microsoft Visual C++ 6.0 compiler.