RTX2018

Interface Specification

1 Introduction

This document describes the software interface between an application for the RTX2018 system and the DLL that is implementing the functionality and connection to the HW of the system.

1.1 Document History

Ver.	Description	Resp.	Date
0.1	Draft version.	LGJ	25-Feb-2019
0.2	Improved draft revision.	Init	xx-mmm-yyyy
1.0	Released after review by xxx, yyy and zzz.	Init	xx-mmm-yyyy

1.2 References

Number	Document

2 System Description

The system is based on the Rohde & Schwarz CMW100 tester. The CMW100 is basically a fast AD / DA converter. The CMW100 is connected to a PC with some CMW100 driver software. The CMW100 hardware, PC and CMW100 driver is the real test equipment. On top of this the DECT functionality is added. This is done with a DLL, that offer a set of functions to setup the equipment and make the calculations needed for testing a DECT device. The test application uses this DLL.

An overview of the system is shown in Figure 1.

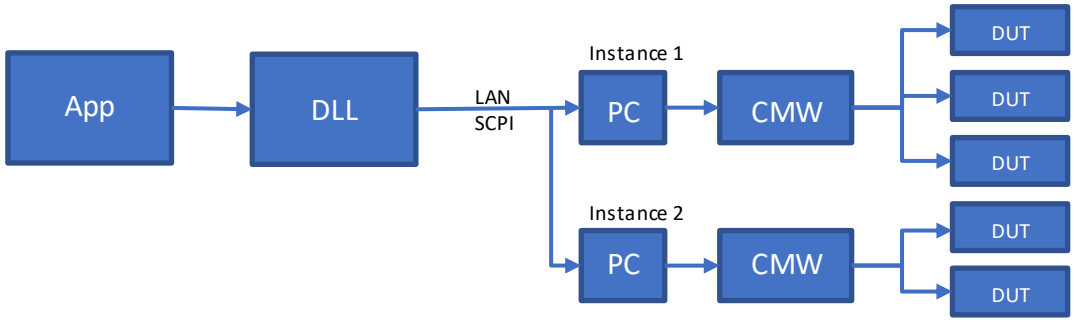

Figure 1: System setup for RTX2018 test equipment

The test application can be programmed in any language that supports use of a Windows DLL.

The test application and DLL can be located and executed on the PC connected to the CMW100 or it can be executed from a remote connection.

3 Format of Interfaces, Functions and Types

Communication with a device is done using an *interface*, which is a collection of functions and types.

Interfaces are documented using the following format:

Interface:	**The name of the interface**
Description:	A description of the interface.

All functions and types following an interface specification belong to that interface, until the end of the document or a new interface is specified.

The communication in this interface is function based.

Function interfaces do not use mails for communication. Typically, these interact with the DLL and do not directly communicate with the target. A function is documented like this:

Callx:	**The name of the function**	
Description:	A description of what the function does	
Return value type:	The type of the return value. This may be a simple type or the name of a composite type, which is documented in the *Types* section in this document. If a composite type is only used as a call return value, it may be documented immediately after the call documentation.	
Return value description:	A description of the return value	
Parameters:		
Type	**Name:**	**Description:**
		Here the types and names of all parameters in the function is described. Types used for parameters are documented elsewhere in this document.

Type definitions are documented using one of two similar formats:

TypeNamex:	**The name of the defined type**
Group:	The kind of type. Typical groups are enumerations, structures, unions, constants, etc.
Description:	A description of the type
Type:	The underlying type, e.g. uint8_t, int32_t, bool etc.
Value:	The value of the type (constants only)

TypeNamex:	**The name of the defined type**
Group:	The kind of type. Typical groups are enumerations, structures, unions, constants, etc.
Description:	A description of the underlying type

Code	**Description**
Code that defines the members of the type	A description of each member.

The following sections document the various interfaces used in this system and their functions, and types.

4 RTX2018 Interface

4.1 Interface Definitions

Interface:	**Cmw**
Description:	This interface allows applications to use and configure the CWM module to implement the RTX2018 functionality. All functions and types in this interface are prefixed with “Cmw”.

4.2 Enumeration Definitions

TypeName:	**Cmw_ErrorType_t**
Group:	Enumeration
Description:	This type defines the legal error types in the system

Code	**Description**
CMW_ERR_NO_ERROR = 0	No error
CMW_ERR_RANGE = 1	A value was outside the legal range
CMW_ERR_UNDERDRIVEN = 2	Input-level too high to perform valid measurements
CMW_ERR_OVERDRIVEN = 3	Input-level too low to perform valid measurements
CMW_ERR_UNABLE_TO_CONNECT_CMW = 4	Unable to make connection to the instrument
CMW_ERR_TIMEOUT = 5	The operation timed out
CMW_ERR_NOT_IMPLEMENTED = 6	The requested operation is not implemented
CMW_ERR_NO_DATA_AWAILABLE=7	No signal to analyze
CMW_ERR_EVAL_WINDOW_VIOLATION=8	Eval-window do not fit in capture window
CMW_ERR_FILE_NOT_FOUND=9	
CMW_ERR_NO_SYNC_FOUND=10	Not able to find sync in signal
CMW_ERR_NO_LICENSE_AWAILABLE=11	
CMW_ERR_INVALID_LICENSEFILE=12	
CMW_ERR_LICENSE_EXPIRED=13	
CMW_ERR_UNKNOWN = 14	The request or subcommand was unknown
CMW_ERR_NO_REPLY = 0x80	The device did not respond to the DLL in time

TypeName:	Cmw_StatusType_t
Group:	Enumeration
Description:	This type defines the status modes in the system

Code	Description
CMW_STATUS_CONNECTED_TO_CMW = 1	
CMW_NO_SIGNAL = 2	
CMW_STATUS_OK = 0	Everything is OK

TypeName:	Cmw_AnaModType_t
Group:	Enumeration
Description:	This type defines the used modulation type in the system

Code	Description
GFSK = 0	
PI2_DBPSK = 1	
PI4_DQPSK = 2	
PI8_D8PSK = 3	

TypeName:	Cmw_AnaRefPoint_t
Group:	Enumeration
Description:	This type defines the reference point modes for analyzing

Code	Description
B0 = 0	Used if B0 is available
TrigPoint = 1	Used if B0 not defined
All = 2	All sampled data will be analyzed.

TypeName:	Cmw_State_t
Group:	Enumeration
Description:	This type defines the On/Off state of a feature

Code	Description
Off = 0	Feature is disabled
On = 1	Feature is enabled

TypeName:	Cmw_GenPortType_t
Group:	Enumeration
Description:	This type defines the port numbering for the generator

Code	Description
Port1 = 0x01	Value for port 1
Port2 = 0x02	Value for port 2
Port3 = 0x04	Value for port 3
Port4 = 0x08	Value for port 4
Port5 = 0x10	Value for port 5
Port6 = 0x20	Value for port 6
Port7 = 0x40	Value for port 7
Port8 = 0x80	Value for port 8

4.3 Type Definitions

TypeName:	Cmw_AnaMeasDataRetType_t
Group:	Struct
Description:	This type defines the calculated results based on one set of samples.

Code	Description
double PhaseOffset;	[Rad]. Initial relative phase orientation in demodulated signal. Only useful for debugging purposes.
double FreqOffset;	[Hz] Frequency offset of the analyzed signal.
uint64_t SampleOffset;	[Samples].
uint64_t B0SymbolOffset;	[Symbols] Number of symbol times from start of capture window only location of the b0. Symbol.
uint64_t B0SampleOffset;	Samples. Sample number of the capture window where the center of b0 is.
int64_t MinFreqDeviation;	[Hz] Frequency deviation of the FM demodulated signal. Only valid result if modulation is GFSK.
int64_t MaxFreqDeviation;	[Hz] Frequency deviation of the FM demodulated signal. Only valid result if modulation is GFSK.
double EVM;	[dB]. RMS EVM of the signal. Only valid for PI2_DBPSK, PI4_DQPSK, PI8_D8PSK,
double EVMPeak;	[dB]. Peak EVM of the signal. Only valid for PI2_DBPSK, PI4_DQPSK, PI8_D8PSK,
double Power;	[dBm]. TX power level of the signal.
Cmw_ErrorType_t Error;	

TypeName:	Cmw_BlockDataType_t
Group:	Struct
Description:	This type defines data that controls copying a big data block in smaller chunks.

Code	Description
uint32_t Start;	The index of the next chunk in the big block
uint8_t Size;	The size of the chunk
double Time[100];	Sample time
double IData[100];	The I data of the chunk
double QData[100];	The Q data of the chunk
bool Done;	If true, this chunk is the last one
Cmw_ErrorType_t Error;	

TypeName:	Cmw_GeneralRetType_t
Group:	Struct
Description:	General return type used by functions where no data is returned, but only the error code.

Code	Description
Cmw_ErrorType_t Error;	

TypeName:	Cmw_InitRetType_t
Group:	Struct
Description:	Return type for the initialization call.

Code	Description
uint8_t Instance;	Instance number to be used in rest of the interface
char DevName[256];	Name, serial number and version of the attached instrument
char DLLName[256];	Name and version of the attached instrument
Cmw_ErrorType_t Error;	

TypeName:	Cmw_GetPowerRetType_t
Group:	Struct
Description:	Return type for get power call

Code	Description
double Power;	Average power in measurement window
Cmw_ErrorType_t Error;	

4.4 Function Definitions

Call:	Cmw_Init
Description:	Initialize the Cmw interface
Return value type:	Cmw_InitRetType_t
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE CMW_ERR_UNABLE_TO_CONNECT_CMW

Parameters:		
Type	Name	Description
char*	SocketServerIP	IP-addres of socket server
char*	SocketServerName	DNS name of socket server
uint16_t	Port	IP port of socket server

Call:	Cmw_Close
Description:	Closes connection to CMW
Return value type:	Cmw_GeneralRetType_t
Return value description:	CMW_ERR_NO_ERROR

Parameters:		
Type	Name	Description
uint8_t	Instance	

Call:	Cmw_GetSystemInfo	
Description:	Reads information about the tester and DLL	
Return value type:	Cmw_InitRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE CMW_ERR_UNABLE_TO_CONNECT_CMW	
Parameters:		
Type	Name	Description
uint8_t	Instance	

Call:	Cmw_SetAnaFrequency	
Description:	Set frequency for the analyzer. Range: 0 – 6 GHz Default: 1 GHz	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint64_t	Freq	[Hz] Frequency to be set

Call:	Cmw_SetGenFrequency	
Description:	Set frequency for the generator. Range: 0 – 6 GHz Default: 1 GHz	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint64_t	Freq	[Hz] Frequency to be set

Call:	Cmw_SetGenPower	
Description:	Set generator power Range: -134 – -15 dBm Default_ -134 dBm	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	**Name**	**Description**
uint8_t	Instance	
double	Power	[dBm] Generatorpower to be set

Call:	Cmw_LoadGeneratorModFile	
Description:	Specify the path and filename of the file to be for modulating the generator	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_TIMEOUT CMW_ERR_FILE_NOT_FOUND	
Parameters:		
Type	**Name**	**Description**
uint8_t	Instance	
const char*	ModulationFilePath	Path of the file to be loaded into the CMW. (“C:\RTX2018\”)
const char*	ModulationFile	Filename of the file to be loaded into the CMW.

Call:	Cmw_SetGeneratorPort		
Description:	Sets the output port for the generator. More than one port can be active at same time. Default: 0		
Return value type:	Cmw_GeneralRetType_t		
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_TIMEOUT CMW_ERR_RANGE		
Parameters:			
Type		**Name**	**Description**
uint8_t		Instance	
Cmw_GenPortType_t		Port	Sets the ports on and off. (0b00000100 set port 3 on)

Call:	Cmw_SetGeneratorState		
Description:	Sets the state of the generator. This command controls all the ports. Default: Off		
Return value type:	Cmw_GeneralRetType_t		
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_TIMEOUT		
Parameters:			
Type		**Name**	**Description**
uint8_t		Instance	
Cmw_State_t		State	Sets the generator state to On or Off

Call:	Cmw_SetAnaModulationType	
Description:	Sets the modulation type of the analyzer Default: PI2_DBPSK	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
Cmw_AnaModType_t	ModulationType	

Call:	Cmw_SetAnaCaptureWindow	
Description:	Defines the window where the signal is sampled. GateDelay: Range: -1000 – -10 us Default: -40; GateTime: Range: 10 – 1000 us Default: 400 us	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
int64_t	GateDelay	[us] Start of capture time relative to trigger point.
uint64_t	GateTime	[us] Length of capture time.

Call:	Cmw_SetAnaAnalyzerWindow		
Description:	Defines the part the capture window used to analyze signal. RefPoint: Default: All GateDalay: Range: -1000 – 1000 us Default: 0; GateTime: Range: 10 – 1000 us Default: 350 us		
Return value type:	Cmw_GeneralRetType_t		
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE		
Parameters:			
Type		Name	Description
uint8_t		Instance	
Cmw_AnaRefPoint_t		RefPoint	
int64_t		GateDelay	[us]
uint64_t		GateTime	[us]

Call:	Cmw_SetAnaTrigOffset		
Description:	Defines the offset to expected power to trig a measurement sequence Range: -50 – 0 dB Default: 0 dB		
Return value type:	Cmw_GeneralRetType_t		
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE		
Parameters:			
Type		Name	Description
uint8_t		Instance	
double		TrigOffset	dB

Call:	Cmw_SetAnaTrigGap	
Description:	Defines the time where power has to be below trigger offset to allow new trigger. Range: 0 – 1000 us (if 0 us, the first coming trigger event is used.) Default: 10 us	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint64_t	TrigGap	us

Call:	Cmw_SetAnaExpPowLevel	
Description:	Sets the powerlevel at which it is expected that the DUT transmits. Range: -60 – 50 dBm Default: 0	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
double	PowLevel	[dBm]

Call:	Cmw_SetAnalyzerPort	
Description:	Sets the RF port o the CMW to be used for the analyzer. Only one port can be active. Range: 1 – 8 Default: 1	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint8_t	Port	

Call:	Cmw_SetAnaSymbolRate	
Description:	Sets the symbol rate of the sampled signal to be analyzed. The DLL operates with an oversampling factor of 16. Range: 0 – 2MHz Default: 1152000 Hz	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint64_t	SymbolRate	[Hz]

Call:	Cmw_SetSyncWord	
Description:	Defines the length and content of the Synchronization word. Syncword: Default: 0x51975555 NumberOfBitsInSyncword: Range: 0 – 64 Default: 32	
Return value type:	Cmw_GeneralRetType_t	
Return value description:	CMW_ERR_NO_ERROR CMW_ERR_RANGE	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint64_t	SyncWord	The syncword that the DLL uses to determines the position of bit 0.
uint64_t	NumberOfBitsInSyncword	Number of bits in synchronization word. Max value: 64

Call:	Cmw_AnaGetMeasData	
Description:	Capture and analyze the signal. Calculated values are reported in the return structure. If signal is captured, it can be re-analyzed if UpdateSignal is low.	
Return value type:	Cmw_AnaMeasDataRetType_t	
Return value description:		
Parameters:		
Type	Name	Description
uint8_t	Instance	
bool	UpdateSignal	Used to control when to capture new signal.

Call:	Cmw_AnaGetPower	
Description:	Capture and calculate power of captured signal	
Return value type:	Cmw_GetPowerRetType_t	
Return value description:	Power is returned in dBm	
Parameters:		
Type	Name	Description
uint8_t	Instance	
bool	UpdateSignal	Used to control when to capture new signal

Call:	Cmw_AnaSetSignalDataBlock	
Description:	Download signal to be analyzed from the client. By loading a signal with this function and afterwards analyzing it with "UpdateSignal" equal to 0, it is possible to analyze stored signals. Mainly used for debug purposes.	
Return value type:	Cmw_GeneralRetType_t	
Return value description:		
Parameters:		
Type	Name	Description
uint8_t	Instance	
Cmw_BlockDataType_t	SignalBlock	Chunk of data. Max 100 records per transaction.
uint16_t	Blockno	The index of the chunk. Must start from 0 and increment by one for each call.

Call:	Cmw_AnaGetDemodDataBlock	
Description:	Upload one big block of data to the client	
Return value type:	Cmw_BlockDataType_t	
Return value description:	The next chunk of data	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint16_t	Blockno	The index of the chunk. Must start from 0 and increment by one for each call.

Call:	Cmw_AnaGetPowerDataBlock	
Description:	Upload one big block of data to the client	
Return value type:	Cmw_BlockDataType_t	
Return value description:	The next chunk of data	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint16_t	Blockno	The index of the chunk. Must start from 0 and increment by one for each call.

Call:	Cmw_AnaGetIQDataBlock	
Description:	Upload one big block of data to the client	
Return value type:	Cmw_BlockDataType_t	
Return value description:	The next chunk of data	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint16_t	Blockno	The index of the chunk. Must start from 0 and increment by one for each call.

Call:	Cmw_AnaGetDemodulatedBitsBlock	
Description:	Upload one big block of data to the client.	
Return value type:	Cmw_BlockDataType_t	
Return value description:	The next chunk of data	
Parameters:		
Type	Name	Description
uint8_t	Instance	
uint16_t	Blockno	The index of the chunk. Must start from 0 and increment by one for each call.

5 Read Back of Large Data Amounts

As this DLL is intended to also be used with "Managed memory" programming tools as Python and .Net the DLL does not use pointers to memory locations for return large data sets.

As there is a limited amount of data size available for the return structure, the data will need to be read back using several calls to the function to obtain the data in chunks.

This method is valid for the following functions:

- Cmw_AnaGetDemodulatedBitsBlock
- Cmw_AnaGetIQDataBlock
- Cmw_AnaGetPowerDataBlock
- Cmw_AnaGetDemodDataBlock
- Cmw_AnaSetSignalDataBlock

The first call to one of these functions in populated with a Blockno of 0.

The return structure for this function call includes both part of the data (I and Q data) together with the length of the data.

Besides the number for the next chunk is included. This can be used as calling argument for the next call the function.

When the is no more data available, the "Done" field is asserted.