XeThru File Formats

Documentation

XeThru Application Note **by Novelda AS**

B - September 25. 2017

Summary

This application note describes the directory structure and different file formats created and used by XeThru software.

Table of Contents

1 Overview

This document describes the files produced by the ModuleConnector library (XeThru SW), used by different tools like XeThru Explorer. ModuleConnector supports recording and reading these files as described in the next section.

A timestamp on the format {YYYYMMDD_hhmmss} is used for all files and directory names representing the time of file creation. The time used here is local time as defined by the platform running the XeThru software.

Abbreviation	Description
YYYY	year
MM	month
DD	day of month
hh	hour
mm	minute
ss	second

1.1 ModuleConnector Recording and Playback API

The Recording and Playback API in ModuleConnector is valid for all interfaces (X2M200, X4M300, X4M200, XEP) without any difference in usage. The API consist of three main classes:

1. **DataRecorder**
2. **DataReader**
3. **DataPlayer**

The **DataRecorder** class is a high level data recorder class. The purpose of the DataRecorder class is to record all data types sent by a XeThru device over serial port or similar. All low-level I/O is handled by the recorder itself with no setup required. Data is stored on disk as specified by this document and can be easily read back using the *DataReader* class. The DataRecorder class generates a meta file that contains information about the recording such as exact timestamps when bytes were written to disk, which formats and data types were included in the data set. In short, everything needed in order to reproduce the exact same byte stream as it occurred during recording. *DataReader* and *DataPlayer* uses this meta file as input argument. DataRecorder also supports advance recording options such as splitting of files and directories.

The **DataReader** class is a high level data reader class. The purpose of the DataReader class it to read disk records stored by the *DataRecorder* class. It uses the meta file generated by the *DataRecorder* class as input argument and thus, from the user's point of view the recording appears as one big file even if the recording may contain several files and folders on disk. Data returned from this class is always aligned on complete data records as specified by this document. One recording may contain several data types, however this class allows for easy filtering and seeking into the data set.

The **DataPlayer** class is a high level data playback class. The purpose of the DataPlayer class is to provide the user with the ability to playback recorded data as if it was coming from a physical device - same data rate, same format. So rather than initialising ModuleConnector with a physical device (serial port), it is possible to construct ModuleConnector with a DataPlayer instance and receive telegrams as one would normally receive from a physical device. For example, CSV data on disk is converted back to its original telegram/binary format before it is dispatched via ModuleConnector. Moreover, it is possible to control the output from the player via functions such as play, pause, stop, set playback rate, set filter. Internally, the DataPlayer class uses the *DataReader* class to read records from disk before it converts them into binary packets / telegrams.

See ModuleConnector API documentation for more details.

1.2 XeThru Explorer Recording and Playback

XeThru Explorer uses ModuleConnector for recording and playback. All recordings produced by XeThru Explorer is therefore compatible with the Recording and Playback API in ModuleConnector. The same applies for Playback, i.e. XeThru Explorer is capable of loading recordings produced by XeThru Explorer or other software using ModuleConnector.

Note however, that XeThru Explorer has no support for XEP and/or file formats specific to that module.

2 File Formats

2.1 Baseband Amplitude/Phase

Filename: xethru_baseband_ap_{YYYYMMDD_hhmmss}.dat

This file contains amplitude / phase baseband data in binary format.

Data output rate is the frame rate.

Name	DataType	Description	Comments
FrameCounter	unsgined integer(32)	A sequential counter from the radar data. Incremented for each data message.	
NumOfBins	unsigned integer(32)	Number of bins in data set.	
BinLength	float	Length in meters between each bin.	
SamplingFrequency	float	Chip sampling frequency in Hz.	
CarrierFrequency	float	Chip carrier frequency in Hz.	
RangeOffset	float	Start of first range bin in meters.	
Power	float array	Array of NumOfBins float values of the signal power.	
Phase	float array	Array of NumOfBins float values of the signal phase.	

Parameters in the baseband amplitude/phase message.

Power is calculated using:

$$power(n) = i(n)^2 + q(n)^2$$

If amplitude is desired:

$$amp(n) = \sqrt{power(n)}$$

Phase is calculated using:

$$phase(n) = atan2(\frac{q(n)}{i(n)})$$

where n=[0..NumBins-1], i(n) and q(n) are the 2 channels of the complex baseband signal.

Phase is outputted in radians.

2.2 Baseband I/Q

Filename: xethru_baseband_iq_{YYYYMMDD_hhmmss}.dat

This file contains I/Q baseband data in binary format.

Data output rate is the frame rate.

Parameters in the complex baseband I/Q message.

Name	DataType	Description	Comments
FrameCounter	unsigned integer(32)	A sequential counter from the radar data. Incremented for each data message.	
NumOfBins	unsgined integer(32)	Number of bins in data set.	
BinLength	float	Length in meters between each bin.	
SamplingFrequency	float	Chip sampling frequency in Hz.	
CarrierFrequency	float	Chip carrier frequency in Hz.	
RangeOffset	float	Start of first range bin in meters.	
SigI	float array	Array of NumOfBins float values of the signal I-channel.	
SigQ	float array	Array of NumOfBins float values of the signal Q-channel.	

2.3 Pulse-Doppler Float

Filename: xethru_pulsedoppler_float_{YYYYMMDD_hhmmss}.dat

This file contains pulse doppler data in binary format.

Elements in the pulse doppler file.

Name	DataType	Description	Comments
FrameCounter	unsigned integer(32)	A sequential counter from the radar data. Incremented for each data message.	
MatrixCounter	unsigned integer(32)	Incremental matrix counter.	
RangeIdx	unsigned integer(32)	Range bin index of current doppler vector [0..RangeBins-1]	
RangeBins	unsigned integer(32)	Number of total rangebins in the pulse-doppler output matrix.	
FrequencyCount	unsigned integer(32)	Number of points in frequency axis.	
PulseDopplerInstance	unsigned integer(32)	Selected pulsedoppler type from [0..N-1] where N is number of PDs.	
FPS	float	Output chip framerate [frames per second].	
FPSDecimated	float	Input FPS of this PulseDopplerInstance.	
FrequencyStart	float	Frequency of first value.	
FrequencyStep	float	Difference between each frequency bin.	
Range	float	Absolute range of current frequency array.	
Data[0, FrequencyCount>	float array	Power of pulsedoppler bin.	

2.4 Pulse-Doppler Byte

Filename: xethru_pulsedoppler_byte_{YYYYMMDD_hhmmss}.dat

This file contains pulse doppler data in compressed binary format.

Elements in the pulse doppler file.

Name	DataType	Description	Comments
FrameCounter	unsigned integer(32)	A sequential counter from the radar data. Incremented for each data message.	
MatrixCounter	unsigned integer(32)	Incremental matrix counter.	
RangeIdx	unsigned integer(32)	Range bin index of current doppler vector [0..RangeBins-1].	
RangeBins	unsigned integer(32)	Number of total rangebins in the pulsedoppler output matrix.	
FrequencyCount		Number of points in frequency axis.	

Name	DataType	Description	Comments
	unsigned integer(32)		
PulseDopplerInstance	unsigned integer(32)	Selected pulsedoppler type from [0..N-1] where N is number of PDs.	
ByteStepStart	float	Start of dB compression range.	
ByteStepSize	float	Size of one step in dB.	
FPS	float	Output chip framerate [frames per second].	
FPSDecimated	float	Input FPS of this PulseDopplerInstance.	
FrequencyStart	float	Frequency of first value.	
FrequencyStep	float	Difference between each frequency bin.	
Range	float	Absolute range of current frequency array.	
Data[0, FrequencyCount>	unsigned integer(8) array	Power of pulsedoppler bin (compressed float values).	

Decompressed float value is calculated using:

*float = powf(10.0f, (ByteStepStart + byte * ByteStepSize) / 10.0f)*

2.5 Noise Map Float

Filename: xethru_noisemap_float_{YYYYMMDD_hhmmss}.dat

This file contains noise map data in binary format.

Elements in the noise map file.

Name	DataType	Description	Comments
FrameCounter	unsigned integer(32)	A sequential counter from the radar data. Incremented for each data message.	
MatrixCounter	unsigned integer(32)	Incremental matrix counter.	
RangeIdx	unsigned integer(32)	Range bin index of current doppler vector [0..RangeBins-1].	
RangeBins	unsigned integer(32)	Number of total rangebins in the pulsedoppler output matrix.	
FrequencyCount	unsigned integer(32)	Number of points in frequency axis.	
PulseDopplerInstance	unsigned integer(32)	Selected pulsedoppler type from [0..N-1] where N is number of PDs.	

Name	DataType	Description	Comments
FPS	float	Output chip framerate [frames per second].	
FPSDecimated	float	Input FPS of this PulseDopplerInstance.	
FrequencyStart	float	Frequency of first value.	
FrequencyStep	float	Difference between each frequency bin.	
Range	float	Absolute range of current frequency array.	
Data[0, FrequencyCount>	float array	Power of pulsedoppler bin.	

2.6 Noise Map Byte

Filename: xethru_noisemap_byte_{YYYYMMDD_hhmmss}.dat

This file contains noise map data in compressed binary format, i.e. the data array contains compressed float values.

Elements in the noise map file.

Name	DataType	Description	Comments
FrameCounter	unsigned integer(32)	A sequential counter from the radar data. Incremented for each data message.	
MatrixCounter	unsigned integer(32)	Incremental matrix counter.	
RangeIdx	unsigned integer(32)	Range bin index of current doppler vector [0..RangeBins-1].	
RangeBins	unsigned integer(32)	Number of total rangebins in the pulsedoppler output matrix.	
FrequencyCount	unsigned integer(32)	Number of points in frequency axis.	
PulseDopplerInstance	unsigned integer(32)	Selected pulsedoppler type from [0..N-1] where N is number of PDs.	
ByteStepStart	float	Start of dB compression range.	
ByteStepSize	float	Size of one step in dB.	
FPS	float	Output chip framerate [frames per second].	
FPSDecimated	float	Input FPS of this PulseDopplerInstance.	
FrequencyStart	float	Frequency of first value.	
FrequencyStep	float	Difference between each frequency bin.	
Range	float		

Name	DataType	Description	Comments
		Absolute range of current frequency array.	
Data[0, FrequencyCount>	unsigned integer(8) array	Power of pulsedoppler bin (compressed float values).	

Decompressed float value is calculated using:

*float = powf(10.0f, (ByteStepStart + byte * ByteStepSize) / 10.0f)*

2.7 Generic Float Data

Filename: xethru_datafloat_{YYYYMMDD_hhmmss}.dat

This file contains generic float data in binary format.

Parameters in the data float file.

Name	DataType	Description	Comments
ContentId	unsigned integer (32)	Generic Content ID, depending on application.	
Info	unsigned integer (32)	Generic Info. E.g. framecounter for XEP.	
Length	unsigned integer (32)	Number of float values in data set.	
Data	float	Array of Length float values.	

2.8 Generic Byte Data

Filename: xethru_databyte_{YYYYMMDD_hhmmss}.dat

This file contains generic byte data in binary format.

Parameters in the data byte file.

Name	DataType	Description	Comments
ContentId	unsigned integer (32)	Generic Content ID, depending on application	
Info	unsigned integer (32)	Generic Info. E.g. framecounter for XEP.	
Length	unsigned integer (32)	Number of byte values in data set.	
Data	byte	Array of Length byte values.	

2.9 Generic String Data

Filename: xethru_datastring_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list with generic string data.

Parameters in the data string file.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Added by host Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timezone difference) Example: 1997-07-16T20:40:30.045+01:00
ContentId	integer	Generic content ID, depending on application.	
Info	integer	Generic info, depending on application.	
Message	string	String data.	

2.10 Respiration

Filename: xethru_respiration_{YYYYMMDD_hhmmss}.csv

(XeThruExplorer legacy: xethru_log_Respiration_XethruX2M200_{YYYYMMDD_hhmmss}.csv)

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

The output rate is the frame rate and contains basic respiration data and breathing pattern data.

Parameters in the sleep frame message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timezone difference) Example: 1997-07-16T20:40:30.045 +01:00
State	integer	This represents the state of the sensor module.	Values given by Respiration/Sleep profile.
RPM	integer	Respiration rate per minute.	

Name	DataType	Description	Comments
ObjectDistance	float	Distance to the tracked object.	
ObjectMovement	float	Movement of the tracked object (breathing pattern).	
SignalQuality	integer	Value representing the signal quality.	Value from 0 (low) to 10 (high).

2.11 Respiration Moving List

Filename: xethru_respiration_movinglist_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

A new respiration moving list message is outputted every second. It contains individual movement information in range intervals down to 5cm and target detection lists giving information like size and speed of individual targets.

Parameters in the respiration moving list message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timezone difference) Example: 1997-07-16T20:40:30.045+01:00
Counter	integer	A sequential counter from the radar data. Incremented for each data message.	
MovementIntervalCount	integer	Number of items in the MovementSlowItems list and MovementFastItems list. Deterministic, depending on detection zone.	
MovementSlowItems	float	List of movement slow (refers to slow pulsedoppler matrix) values for all range intervals. Values from 0-100.	Length of list is **MovemenIntervalCount** , format: [1, 2, N]
MovementFastItems	float	List of movement fast (refers to fast pulsedoppler matrix) values for all range intervals. Values from 0-100.	Length of list is **MovementIntervalCount** , format: [1, 2, N]

2.12 Respiration Detection List

Filename: xethru_respiration_detectionlist_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

A new respiration detection list message is outputted every second. It contains individual movement information in range intervals down to 5cm and target detection lists giving information like size and speed of individual targets.

Parameters in the respiration detection list message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:s zzzTZD (loc time + timezone difference) Example: 19 07-16T20:40 30.045+01:0
Counter	integer	A sequential counter from the radar data. Incremented for each data message.	
DetectionCount	integer	Number of detections observed, listed in DetectionDistanceItems, DetectionRadarCrossSectionItems and DetectionVelocityItems lists	
DetectionDistanceItems	float	List of distance in meters to all moving targets in the detection zone.	Length of lis **DetectionC** , format: [1, 2 N]
DetectionRadarCrossSectionItems	float	List of radar cross section in cm^2 to all moving targets in the detection zone.	Length of lis **DetectionC** , format: [1, 2 N]
DetectionVelocityItems	float	List of radial velocity in m/s to all moving targets in the detection zone.	Length of lis **DetectionC** , format: [1, 2 N]

2.13 Sleep

Filename: xethru_sleep_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

A new sleep data message is outputted every second. It contains respiration and movement data that for instance can be used in a sleep analysis context.

Parameters in the extended sleep message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timezone difference) Example: 1997-07-16T20:40:30.045+01:00
FrameCounter	integer	A sequential counter from the radar data. Incremented for each data message.	
SensorState	integer	This represents the state of the sensor module.	Values given by Respiration /Sleep profile.
RespirationRate	float	Respiration rate (respirations per minute / RPM). Valid when SensorState is Breathing.	Valid only when SensorState is Breathing. 0 otherwise.
Distance	float	Gives the distance to the subject (which the sensor is currently locked on to) from the sensor.	Valid only when SensorState is Breathing. 0 otherwise.
SignalQuality	integer	Quality measure of the signal quality, describing the signal-to-noise ratio of the current respiration lock. Value from 0 to 10, 0=low -> 10=high.	Valid only when SensorState is Breathing.

Name	DataType	Description	Comments
			0 otherwise.
MovementSlow	float	First movement metric which captures the larger movements. It is given as a percentage(0-100). Higher the percentage larger the movement.	
MovementFast	float	Second movement metric which also captures the larger movements. It is represented as a percentage (0-100). Higher the percentage larger the movement. This metric is more responsive than the MovementSlow. It captures the movements faster than the former.	

2.14 Presence Single

Filename: xethru_presence_single_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

A new oresence single message is outputted every second. It contains presence information about the target closest to the radar.

Parameters in the presence single message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timez difference) Example: 1997-07-16T20:40:30.045+01:00
Counter	integer	A sequential counter from the radar data. Incremented for each data message.	
PresenceState	integer	This represents the state of the sensor module.	Values given by Presence profile.
Distance	float	Distance to where presence is detected.	N/A for presenceState = XTS_VAL_PRESENCE_PRESENCESTATE_NO_PRES

Name	DataType	Description	Comments
Direction	integer	Direction of detected object. 0=stationary, 1=towards sensor, 2=away from sensor.	N/A for presenceState = XTS_VAL_PRESENCE_PRESENCESTATE_NO_PRES
SignalQuality	integer	A measure of the signal quality giving presence detection. Typically used to identify if the sensor is positioned correctly. Value from 0 to 10 where 0=low and 10=high.	Valid only for presenceState = XTS_VAL_PRESENCE_PRESENCESTATE_PRESENC

2.15 Presence Moving List

Filename: xethru_presence_movinglist_{YYYYMMDD_hhmmss}.csv

This file is a semicolon separated list file and contains a header with meta information related to when and how the recording was performed.

A new presence moving list message is outputted every second. It contains individual movement information in range intervals down to 5cm and target detection lists giving information like size and speed of individual targets.

Parameters in the presence moving list message.

Name	DataType	Description	Comments
TimeStamp	string	Real time corresponding to the captured data.	Standard format: YYYY-MM-DDThh:mm:ss.zzzTZD (local time + timezone difference) Example: 1997-07-16T2 40:30.045+01:00
Counter	integer	A sequential counter from the radar data. Incremented for each data message.	
PresenceState	integer		

Name	DataType	Description	Comments
		This represents the state of the sensor module.	Values given by Presence profile.
MovementIntervalCount	integer	Number of items in the MovementSlowItem list and MovementFastItem list. Deterministic, depending on detection zone.	
DetectionCount	integer	Number of detections observed, listed in DetectionDistance, DetectionRadarCrossSection and DetectionVelocity lists.	
MovementSlowItem	float	List of movement slow (refers to slow pulsedoppler matrix) values for all range intervals. Values from 0-100.	Length of list is **MovemenIntervalCoun** , format: [1, 2, N]
MovementFastItem	float	List of movement fast (refers to fast pulsedoppler matrix) values for all range intervals. Values from 0-100.	Length of list is **MovementIntervalCou** , format: [1, 2, N]
DetectionDistance	float	List of distance in meters to all moving targets in the detection zone.	Length of list is **DetectionCount**, format: [1, 2, N]
DetectionRadarCrossSection	float	List of radar cross section in cm^2 to all moving targets in the detection zone.	Length of list is **DetectionCount**, format: [1, 2, N]
DetectionVelocity	float	List of radial velocity in m/s to all moving targets in the detection zone.	Length of list is **DetectionCount**, format: [1, 2, N]

3 Document History

Rev.	Release date	Change description
A	2016-May-27	Initial release
B	2017-September-20	Added new file formats and overview of recording and playback API

4 Disclaimer

The information in this document is provided in connection with Novelda products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Novelda products. EXCEPT AS SET FORTH IN THE NOVELDA TERMS AND CONDITIONS OF SALES LOCATED ON THE NOVELDA WEBSITE, NOVELDA ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL NOVELDA BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF NOVELDA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Novelda makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Novelda does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Novelda products are not suitable for, and shall not be used in, automotive applications. Novelda products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.