XML – IRIG 106 Chapter 10 mapping

Note: This is based on the RCC document 123-16 "IRIG 106 Chapter 10 Programmers Handbook" appendix P with additional references to download sources for the samples. The original document can be accessed from http://www.wsmr.army.mil/RCCsite/Pages/Publications.aspx for those who have access to this server. Others can try this mirror: http://www.irig106.org/wiki/rcc_mirror

This document also describes extended features added by specification version 0.3.2 by Data Bus Tools. These features are not official and not guaranteed to be added to the future public RCC specification. However the official documents normally get updated every four years and we want to keep this new technology growing. So far additions are made in a way that XML files according the official format are still valid in the new format but not always vice versa. We will provide the changes to any new RCC releases and so far the RCC specification is 100 % from Data Bus Tools.

1 Introduction

This specification is a reply to the RCC change request CR031.

The intention of this specification is to have a way to define IRIG 106 Chapter 10 files for testing purposes in a readable XML format. This would allow tools to convert between XML and CH10 back and forth. The focus of XML is readability and not memory efficiency or processing efficiency. Therefore the area of application for this specification is the generation of smaller test files (although nothing technically prevents having large files).

The conversion from XML to CH10 opens these opportunities for example:

- Setup of test data for the verification of CH10 analysis software;
- Preparation of exactly specified data sets for replay.

The conversion from CH10 to XML opens these opportunities for example:

- CH10 files (for example from recorders under development) could be converted to XML for closer inspection and to verify possible corrections (by converting the manually corrected XML file back to CH10);
- Specific error cases can be created for test cases by converting real files to XML, injecting the error and converting back for replay.

It is anticipated that with the use of this new feature new use cases will come up that will require changes to the specification. It is in a usable state but still under active development. For current information check www.databustools.de/IRIG106/XMLCH10mapping.html

2 Changes

See the XML schema document (XSD) file for a brief change history.

3 Concepts

This specification provides an XML schema that defines many but not all restrictions for valid files. Having validated XML files is the precondition to any further conversion.

The XML files will basically be a sequence of CH10 packets, each defined within one XML **Packet** tag.

Within one Packet tag, optional secondary headers and data type specific content can be placed.

3.1 Raw data

Many places in the file allow the placement of raw data, for example for the creation of undefined packet types or to create errors. If anything cannot be defined in a structured way by the use of XML tags and attributes, it can still be defined by defining the raw data directly.

3.2 Endianness

CH10 files use little endian to encode attributes spanning over several bytes. Therefore a checksum attribute defined as 0x12345678 shall be stored in the byte order 78 56 34 12. If any of the XML attributes separates the data in groups however, then little endian byte order shall only be used within each group. For example if some raw data is defined as 16-bit hex words like “1234 5678”, then the data shall be stored in the order 34 12 78 56 because the values form two groups and little endian only applies within the group 1234 and the group 5678.

3.3 Auto values

Often attributes offer “auto-values”. This means that the user is not providing these values but the conversion tool is supposed to do this. Often there will be three options:

- Automatic calculation by conversion tool
- Automatic calculation by conversion tool, modified somehow by users choice (offset)
- Fixed value (user provided content)

An example would be a checksum value, for which following situations could occur

- The user omitted the value: The calculation will be automatic
- The user provided a relative value like +5 or -12: The result of the automatic calculation will be modified by +5 or -12 to create an invalid checksum.
- The user provided a fixed value: The fixed value will be used

Auto-values require an explicit definition of what they relate to. An auto-value that is calculating the number of MIL-STD 1553 messages within one packet for example requires explicit definition of the MIL-STD 1553 messages via XML tags. Definitions via a block of raw data will not count.

If you use a fixed value somewhere remember you need to update it manually when you change other data.

3.4 Hexadecimal vs decimal

There are attributes that require the data to be defined in hexadecimal and others that require decimal data. Most attributes accept both. In this case, hexadecimal data shall be prefixed by “0x”.

3.5 Flag words

Data structures like flag words (like packet flags or CSDW) will be available to define as a whole or with its sub elements. In general the definition for the flag word will be used as a base that is then bitwise modified by the more specific sub elements within the flag word that are explicitly defined in XML.

Example:

PacketFlags=”0xC4” RTCSyncError=”True”

The packet flags of 0xC4 indicate that secondary headers are present and used as IEEE1588 intra-packet time. The RTCSyncError additionally sets the bit to indicate an RTC sync error so the stored flag word becomes 0xE4.

3.6 RTC

Packet RTCs have several options to define which are described in the XSD file. Here some examples:

1234	RTC = 1234
0xFF	RTC = 255
p+120	RTC is 120 ticks behind the previous packet
c+120	RTC is 120 ticks behind the previous packet on the same channel

3.7 Channel defaults

You can define default values for packet flags, data type and data type version per channel. You can also change these defaults at different places in the file. You can override this in individual packets.

3.8 Separate time stamps

The tag SeparateTimeStamp can be used to create Intra Packet Time Stamps (IPTS) when defining raw data. It can be placed inside packets where raw data can be defined and can define an absolute or packet relative time stamp.

3.9 Various

All data that is not further specified shall be 0.

Some attributes just have one possible value. For example the RTCSyncError from the previous example can just be “true”. Instead of setting it to false, the attribute must be omitted.

The XML schema (XSD file) contains some additional documentation for some elements.

4 Specific data types

4.1 TMATS

The TMATS data can be provided directly or as include file.

If you directly provide TMATS data you can use the ASCII tag. Note that you must make sure that the characters you use do not conflict with the XML syntax. Also as a result of the XML

specification itself all line breaks will be converted to a single line feed (ASCII code 0x0A). Also XML prohibits the 0x00 ASCII code inside the data.

4.2 Time

Time packets can be either defined relative to the last time packet on the same channel or you can provide an absolute time definition.

4.3 Real data types

Currently ARINC 429, MIL-STD 1553, UART and CAN bus data can be defined in a structured way. Other data types can always be provided as raw data inside the packet.

5 Sample files

There are seven sample files in XML and converted CH10 format and can be downloaded from www.databustools.de/IRIG106/XMLCH10mapping.html :

Arinc429 / Mil1553 / UART	These show how data for ARINC 429, MIL-STD-1553 and UART can be defined in a structured way.
CANBus	Demonstrates how data can be defined as raw data if the data type is not yet supported by XML. The same CAN bus data is defined as: - Raw data - Structured definition - Definition in DATaRec4 format (Zodiac Data systems CH10 extension)
NMEA0183	Shows an NMEA 0183 GPS definition using structured UART message definition.
RTCWrap	Shows how specific elements of packet decoding can be tested. Here the RTC wraps from the highest value to 0 and you can check how your CH10 software reacts on it (this is no unrealistic case since RTC has no defined starting value).
mappingFeatures	Shows many other features of the XML to CH10 mapping. The created file contains a lot of intentional errors like arbitrary data inserted between packets, checksum and sequence errors etc. It also shows things like relative packet time definition and other concepts explained above. This sample uses an external TMATS file. You must download it and place it in the same directory or adapt the path.