**Telemetry layout user's guide**

This document describes a process that will automatically generate the layout of telemetry packets. It is intended to generate tables that can be used by FSW to form packets and to generate the telemetry dictionary packet definition from the same input.

A series of packets will be defined, each identified by one or more "cycle numbers", expected to be minor cycle numbers within a major cycle where the sequence repeats. Each packet may also include the specification of the channel to use in transmitting the packet. For 1553 this might be the subaddress, in general the interpretation of the channel is up to the FSW.

The packet may start with an explicitly defined packet header followed by various field values. The field values can be various sizes as appropriate for each.

There are 2 products of the process:

- C++ code to allow telemetry packets to be formed. The C++ API and usage are described in TlmPacketDefBase.hpp. This header file is manually coded and does not change. The C++ implementation file for this class is auto-generated based on packet(s) defined as described in this document. The API allows telemetry packets to be populated by inserting values and to be transmitted, either as a byte array or as one or more FF3BufType objects. The API also allows individual fields to be extracted from an existing packet. The latter is intended to aid test and verification.

- An HTML document describing the packets

Each data field in the packet includes an identifier that may be used to determine the correct value to insert in the field. The identifier is arbitrary and up to the implementer but an ISF telemetry database identifier used to fetch the value from the telemetry database may be used for those fields that are populated from the telemetry database. Each field also has a size in bits to use in the output telemetry packet. Fields appear in the packet in the order that ther are defined in the input.

The input to the process is a tab-delimited CSV file described below. A convenient way to generate this format is by maintaining the data in a spreadsheet and exporting to CSV but the actual input is the CSV file which may be generated in any way desired..

The first column in the CSV file is a keyword giving type of the row. Keywords are case-insensitive. The keywords can be:

> **<u>Cycle</u>**: Indicates that this is the start of a new packet and gives the minor cycle number(s) for this packet. This must be the first keyword beginning the definition of a packet.

> Channel: Gives the channel where this packet is to be sent. For 1553 this might be a subaddress, otherwis it may map to an interface.

> Identifier: Gives an identifier to this packet. A header field may be defined that will automatically be set to this value.

**Header**: The remainder of the parameters on this line allow a packet header field to be defined. This is repeated for each header field. Header fields may be defined using explicit header fields although this is not required. The API includes methods to populate time, ID and sequence number in explicit header fields.

**Item**: Specifies an individual telemetry item, this line appears once for each telemetry item in the packet. The packet will be formed containing items in the order that they are specified.

**Reserved**: Specifies a reserved field (unused space).

**Align**: Aligns to the next multiple of the specified number of bits. Equivalent to doing a reserve for that number of bits. If already aligned, does nothing.

**Constant**: Specifies a field that contains a constant value.

**Comment**: Allows column headers or other comments to be included in the spreadsheet. This record is ignored in downstream processing.

For all records, the sixth column is a description /comment field used in documentation. The remainder of the row is dependent on the keyword. Blank lines are ignored.

For all records that require a field size to be specified, the size is in the fifth column.

## Cycle record

The second column is a non-negative integer giving the cycle number(s) for production of this packet. Cycle number is in the range 0 to N – 1 where N is the number of minor cycles in a major cycle. Multiple cycles may be specified for a packet as a comma-separated list.

A cycle record must always be the first record in a packet definition.

## Channel record

The second column is a non-negative integer giving the channel associated with this packet.

## Identifier record

Second column: A text name (mnemonic) for this packet. Used in documentation. No spaces and must contain only characters legal in file names.

Third column: an integer >= 0 giving an identifier for this packet.

The sixth column would normally contain a description of the packet to be used in documentation.

If there is a header record defining an ID field, it will be automatically populated with this value.

## Header record

The second column is a reserved word identifying the specific header field being defined.  Header records occur once for each field in the header. The fields occur adjacent and in the order specified. As for other records, the header records contain a comment in the sixth column. All header fields are optional. The telemetry items start immediately after the header with no alignment automatically done.

Header record types (second column):

> **ID**: This identifies a field to receive the numerical ID given in the identifier record for this packet. The fifth column is the number of bits in the ID field. This field will default to the value specified in the identifier record. The default may be overridden with the setHeaderId method in the base class.

> **Time**: This identifies a field to contain the time. A time is a 48 bit field with 32 bits of seconds followed by 16 bits of subseconds.  The subseconds are in units of 1/65536 of a second.

> Note: This may change in future releases as additional time formats are supported. In the API this is passed as an Fw::Time object.  The base time field is not used.

> **Sequence**: This defines a header field to contain a packet sequence number.  The fifth column is the size in bits of this field.

> **Field**: This specifies a field that may be populated as desired with mission-specific data. The third column is an identifier for this field as an arbitrary integer. The fifth column is the number of bits in the field.

## Item record

Second column: Text name (mnemonic) of this telemetry item.  Must not contain spaces and must not start with a numeric digit.

Third column: A numeric identifier used to identify the data that goes into this field. It may be a telemetry ID used retrieve this  item from the telemetry database or any other value that is understood by the code that forms the packet.

Fourth column: The data type of the item as returned from the telemetry database.  The data type is used only for documentation and may be any descriptive string.

Fifth column: Number of bits in the telemetry packet to use for storage of this item.  If the item to be stored is longer than this, the lowest addressed bits are used.

## Reserve record

The fifth column is the number of bits to be reserved.

## Align record

The second column is the desired alignment.  Must be 8, 16, 32 or 64.

## Constant record

Second column: Text name (mnemonic) of this telemetry item.  Must not contain spaces and must not start with a numeric digit.

Third column: The data type for the constant. This is used only in documentation.

Fourth column: The value for the constant field.

Fifth column: The number of bits in the telemetry packet to use for storage of this field.

## **Comment record**

Content is ignored, may be anything to increase readability