23.0 Telemetry Writer Templates (36-53232.03 01)

23.1 Purpose

The purpose of the collection of telemetry packet writer classes is to format bit-packed fields into telemetry packet buffers to be sent out of the instrument. These classes are produced by a code-generator. This section describes the template for these classes.

The details of the telemetry packet formats are shown in the "ACIS Instrument Program and Command List," MIT 36-01410 (IP&CL).

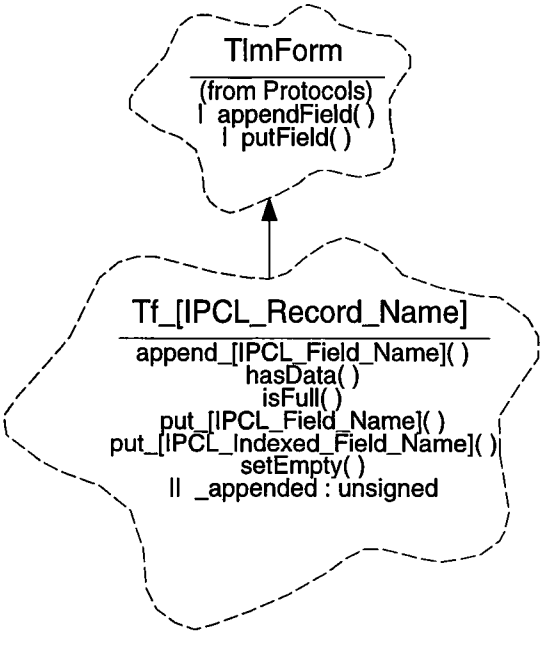
23.2 Uses

Use 1:: Write a particular field into a telemetry packet buffer
Use 2:: Append elements to the end of a telemetry packet buffer
Use 3:: Determine the number of words in a telemetry packet buffer
Use 4:: Determine if a variable length telemetry packet buffer is full
Use 5:: Determine if a variable length telemetry packet buffer has any data to send

23.3 Organization

All generated telemetry writers are a subclass of ***Protocols***::**TlmForm** (see Section TBD). The name of a given telemetry packet writer class is the name of the record type, as listed in the IP&CL, with spaces replaced with underscores, and prepended with the name **Tf_**.

FIGURE 99. Telemetry Packet Writer Relationships



Protocols*::TlmForm** - This class is a member of the ***Protocols class category and represents a generic telemetry packet formatter. It provides functions used by its sub-classes to determine the xxx. This class is described in more detail in Section TBD.

Tf_[IPCL_Record_Name] - This represents a generic template for the various types of telemetry packet formatter classes, where "**[IPCL_Record_Name]**" is the name of the telemetry packet record definition within the IP&CL database (spaces within the record name are replaced with "_" in the class name). Each class definition provides a member function which returns the number of words in the packet (`getWordCount`) and one member function for each field defined within the record but are not part of a variable length array (`put_[IPCL_Field_Name]`, `put_[IPCL_Indexed_Field_Name]`). For fields which are part of a fixed length array, this member function takes a single index argument, indicating which element of the array to read. If a field defines a variable length array at the end of a packet, the class contains a member function which appends all elements of the field to the end of the packet (`append_[IPCL_Field_Name]`). If the field's data type is another record, the function arguments consist of the fields within the defined record. If a telemetry packet is variable in length, the class contains member functions which indicate if the packet has had any data appended to it (`hasData`), and whether or not the packet has room for any more data (`isFull`).

23.4 Writer Design Issues

23.4.1 Assumptions

This section lists some of the assumptions made by the command packet and parameter block reader classes.

- Except for fields defining structures, no single field is longer than 32-bits
- No telemetry packet contains more than 1 variable length array
- Variable length arrays are always at the end of a telemetry packet buffer

23.4.2 Put Field Access Functions

Two approaches were considered when building the code-generator. One is to have the code-generator produce already customized code, which optimally writes fields to the packet's output buffer. The other approach has the code-generator producing standard code which relies on the compiler to optimize of the inline expansions of `putField()` member function calls. The current design takes the second approach.

The general form for every `put_[IPCL_Field_Name]` function consists of the following:

```
putField(value, BITOFFSET, BITWIDTH, BITMASK, index, ARRAYWIDTH);
return;
```

Where *value* is the value to store into the telemetry buffer, BITOFFSET is the bit position of the field from the start of the packet or block. If the field is within an array, BITOFFSET is the offset to the field within the first array element. BITWIDTH is the number of bits in the field and must be less than or equal to 32. BITMASK is a right-justified mask of the field (1's correspond to bits in the field). If the field is within an array, *index* is an argument which selects which array element to access and ARRAYWIDTH is the number of bits within one element of the array. If the field is not within an array, *index* and ARRAYWIDTH will be zero set to 0 by the code-generator.

The code-generator produces constants for all parameters to `putField()` except *value* and *index*. Given an appropriately written `putField()` member function, the compiler's optimizer can very efficient code when it is expanded.

23.4.3 Append Field Access Functions

The approach used to append items to the end of a telemetry packet is similar to that taken to write a value into the middle of the packet buffer, except that the append_[IPCL_Field_Name] functions may append entire records to the end of the buffer.

The general form for each append_[IPCL_Field_Name] function consists of one call to appendField() for each subfield in the structure being appended. The subfields are appended in the order they appear in the appended structure.

23.5 Class Tf_[IPCL_Record_Name]

Documentation:

This is a template for all telemetry packet writer classes generated from data contained with the ACIS Instrument Program and Command List database. The generated classes are responsible for providing functions which write or append fields to a telemetry packet buffer.

Export Control: Public

Cardinality: n

Hierarchy:

Superclasses: **`TlmForm`**

Public Interface:

Operations:

```
append_[IPCL_Field_Name]()
hasData()
isFull()
put_[IPCL_Field_Name]()
put_[IPCL_Indexed_Field_Name]()
setEmpty()
```

Private Interface:

Has-A Relationships:

`unsigned` *_appended*: This instance variable is defined in all variable length and indicates the number of elements which have been appended to the end of the packet buffer.

Concurrency: Guarded

Persistence: Transient

23.5.1 append_[IPCL_Field_Name]()

Public member of: **`Tf_[IPCL_Record_Name]`**

Return Class: **`void`**

Arguments:

`unsigned or int` *`subfield1`*
`unsigned or int` *`subfield2`*
`...`

Documentation:

This function appends the elements of the indicated field to the end of the telemetry packet buffer. Each passed argument corrresponds to each sub-field of the appended structure.

Semantics:

For each passed argument, call `appendField()`, passing the value to store, the bit-offset of the field in the first array element, the bit-width of the field, the index of the structure being appended to and the bit-width of a single structure element. Once the element has been appended, increment *`_appended`*.

Concurrency: Guarded

23.5.2 hasData()

Public member of: **Tf_[IPCL_Record_Name]**

Return Class: **Boolean**

Documentation:

This function is provided by all variable length telemetry packets and indicates whether or not data has been appended to the packet. If the packet has had data appended, the function returns *BoolTrue*. If not, it returns *BoolFalse*.

Semantics:

Return *BoolTrue* if *_appended* is not 0, otherwise return *BoolFalse*.

Concurrency: Guarded

23.5.3 isFull()

Public member of: **Tf_[IPCL_Record_Name]**

Return Class: **Boolean**

Documentation:

This function is provided by all variable length telemetry packets and indicates whether the caller can append 1 more element to the buffer.

Semantics:

Compute the number of bits occupied if the buffer had 1 more element appended.

```
offset = ARRAYOFFSET + (_appended + 1) * ARRAYWIDTH
```

If it is beyond the capacity of the buffer, return *BoolFalse*, otherwise return *BoolTrue*.

Concurrency: Guarded

23.5.4 put_[IPCL_Field_Name]()

Public member of: **`Tf_[IPCL_Record_Name]`**

Return Class: **`void`**

Arguments:

`unsigned or int` *`value`*

Documentation:

This function writes the passed *`value`* into the telemetry packet buffer to the indicated bit-field. If the field is **`unsigned`**, the value argument is an unsigned number. If the field is **`signed`**, then value is passed as a signed integer.

Semantics:

Call the parent function **`TlmForm`**::`putField()`, passing the *`value`*, the constant *`bitoffset`*, *`bitwidth`*, and *`bitmask`* as arguments, Pass 0 for the *`index`*, and 0 for the *`arraywidth`*.

Concurrency: Guarded

23.5.5 put_[IPCL_Indexed_Field_Name]()

Public member of: **Tf_[IPCL_Record_Name]**

Return Class: **void**

Arguments:

unsigned or int *value*
unsigned *index*

Documentation:

This function writes the passed *value* into the telemetry packet buffer to the indicated array field. If the field is **unsigned**, the *value* argument is an unsigned number. If the field is **signed**, then value is passed as an signed integer. *Index* indicates into which array element to store the value.

Semantics:

Call the parent function **TlmForm**::putField(), passing the *value*, the constant *bitoffset*, *bitwidth*, and *bitmask* as arguments, Pass the provided *index* argument, and the constant for the *arraywidth*.

Concurrency: Guarded

23.5.6 setEmpty()

Public member of: **Tf_[IPCL_Record_Name]**

Return Class: **void**

Documentation:

This function is placed in all variable length classes. When called, the function resets the count of number of elements current appended to the telemetry packet buffer.

Semantics:

_appended = 0

Concurrency: Guarded