Description of

JR/Graupner SPCM Data Format

Shaul Eizikovich, Michel Kuenemann

Introduction

This document describes the SPCM data format employed by JR/Graupner. The information is based on work by *Tsutomu Seki*[2] and enhanced by reverse-engineering of many patterns recorded from a Graupner MX 16s transmitter.

It is assumed that the reader of this document has a basic understanding of RC transmitters and receivers. This document aim is to help this reader to better understand the SPCM format and enable him to create hardware or software devices that correctly interpret SPCM signals.

Overview

SPCM format supports eight 10-bit **real-time** channels and two 8-bit **extra** channels. For every real-time channel, the data is sent every 44 or 22 milliseconds as will be explained. The extra channels data is sent every 44 to 220 milliseconds, according to need.
In addition, the format supports transmission of per-channel Fail-Safe (FS) data.

Real-Time channels

There are eight real-time channels marked CH1 to CH8. Every channel carries information from the transmitter to the receiver in the form of an unsigned integer in the range of 0 to 1023 (10 bits). By default, the 10-bit **absolute** channel value is transmitted every 44 milliseconds. If the channel value changes halfway between transmissions – the transmitter sends a **relative** channel value 22 milliseconds after the previous absolute value has been transmitted.

Absolute Channel Value

The channel value reflects the transmitter's stick (or equivalent) position. Minimum position will yield a value of 0 while maximum position will yield a high value of 1023. Hence, the channel absolute value is fully described by 10-bits.

Minimum position (0) is equivalent to a PWM (Pulse Width Modulation) pulse of 900µS or to a servo motor throw of -60°. Maximum position (1023) is equivalent to a PPM pulse of 2100µS or to a servo motor throw of +60°.

Relative Channel Value

Halfway between two consecutive transmissions of a channel's Absolute Value, the transmitter evaluates the current value of the channel. If this value has changed, the transmitter may transmit this new current value in the following manner:

1. The value is *relative* to the previous transmission of the corresponding Absolute Value. The transmitter calculates the absolute difference between the current channel value and the last transmitted Absolute Value. We call this *absolute* difference the channel **Delta** value.
2. The value has *polarity*. If the new value is bigger than the last transmitted Absolute Value then we define the Delta value as **Positive**. If the new value is smaller than the last transmitted Absolute Value then we define the Delta value as **Negative**.
3. The full range of Delta values is split into 16 smaller **Ranges**. The Ranges are of different

sizes. Every Delta value falls into exactly one Range. Every range holds one or more Delta values. The lowest Range is Range$_0$. The upper Range is Range$_{15}$. See Table 4 for details.

4. The transmitter passes the new current value by transmitting the corresponding Range Index (0 to 15) and the Polarity.

Extra Channels

There are two extra channels marked CH9 & CH10. These channels cannot be used for missions that time-critical or require top accuracy such as aircraft navigation or throttle. Rather, they can be used for controlling flaps, landing gear, bomb dropping etcetera. Each extra channel carries information from the transmitter to the receiver in the form of an unsigned integer in the range of 0 to 255 (8 bits). For the sake of simplicity, we shall assume these channels to also be 10-bit channels where the two lower bits are always 0, hence ranging from 0 to 1020 with granularity of 4. When unchanging, the absolute channel value is sent approximately every 220 milliseconds. When the value changes, it is sent approximately every 44 milliseconds.

Fail-Safe Values (FS)

In case the receiver receives corrupted data or in case the transmission is interrupted, the receiver may take one of the following actions:

- For each channel, Retain the current channel values (FS disabled).
- For each channel, move to a pre-loaded values. These values are the fail-safe (FS) values.

SPCM enables transmission of per-channel FS values for the eight Real-Time channels CH1 to CH8. As these are 10-bit channels, the Fail-safe value for a given channel is also a 10-bit value. However, only the upper eight bits of the Fail-safe value are transmitted. The lower 2 bits are assumed to always be zero.

Special Case: Fail-Safe value of 0 (all ten bits are 0) means *FS disabled*.

Physical Frame

Structure

The basic data structure used by SPCM to carry data is the *Frame*. Each frame begins with a special synchronization sequence, followed by payload data and ends with padding. The frame duration is constant (Approximately 44 Milliseconds). The number of pulses it contains may vary.

Pulse

The length of a pulse, with the exception of the *synchronization* pulse, may be any integer between 2 and 14 *ticks*. A tick is defined as a period of 165 μSeconds. The length of the *synchronization* pulse is 2.5 ticks (412.5 μSeconds). Pulse-polarity is not taken into account.

Pulse Decoding

A pulse of N ticks is decoded into the binary value of '1' followed by N-1 '0's. (e.g. A pulse of 5 ticks is decoded into binary '10000'). The pulse polarity is ignored.

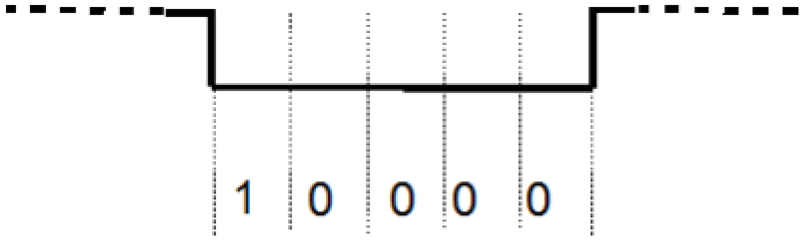

Figure 1: Five-Tick Pulse

The payload data begins one tick after the *synchronization* pulse and is 256 tick long. It is interpreted so that the most significant bit (Bit 255) is the first to arrive and its least significant bit (Bit 0) is the last to arrive. The payload data is sliced into 32 Octets of data, where the first Octet to arrive is Octet 0 and the last Octet to arrive is Octet 31.

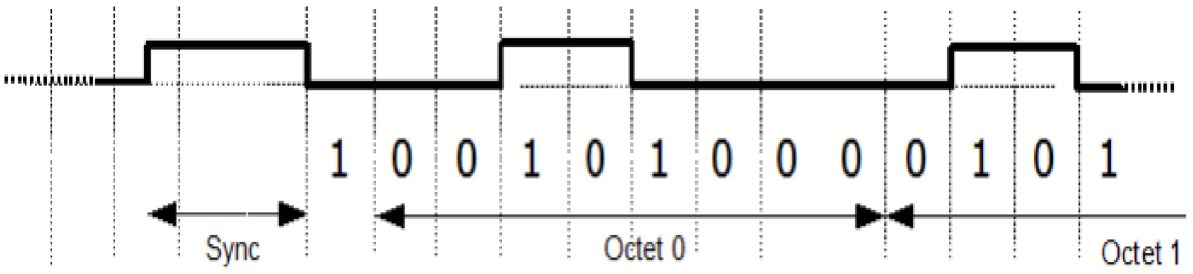

Figure 2: Frame Sync Pulse followed by first Octet

All pulses arriving after the payload and before the next synchronization pulse are considered as padding, hence ignored.

The total duration of a frame is constant and equals to 262 ticks, synchronization pulse excluded. A frame contains 32 Octets of 8 ticks of data followed by 6 padding ticks.

Logical Frame

Decoding

Every payload data on the physical frame corresponds to a logical frame carrying encoded data. Every Octet of the payload data is decoded in-place into a five-bit *Quintet* according to the following table:

Octet Value (Hex.)	Quintet Value (Hex.)	Quintet Value (Bin.)
1	00	00000
2	10	10000
4	11	10001
5	1A	11010
8	15	10101
9	0A	01010
~~0B~~ 0A	08	01000
10	17	10111
11	0E	01110
12	0C	01100
14	04	00100
15	1B	11011
20	13	10011
21	0F	01111
22	0D	01101
24	05	00101
25	19	11001
28	07	00111
29	1D	11101
2A	1F	11111
40	12	10010
41	0B	01011
42	09	01001
44	01	00001
45	18	11000
48	03	00011
49	1C	11100
4A	1E	11110
50	02	00010
51	14	10100
52	16	10110
54	06	00110

Table 1: Octet to Quintet conversion table

Structure

Every frame is constructed of 32 Quintets of logical data. The data is organized as follows:

1	**Ch 2**		3	**Ch 4**		A		5	**Ch 6**		7	**Ch 8**		B		**Ch 1**		2	**Ch 3**		4	C		**Ch 5**		6	**Ch 7**		8	D	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Figure 3: Logical Frame structure

The *Absolute Channel Value* is marked **bold** (**Ch 1 ... Ch 8**).
The corresponding *Auxiliary* data is underscored (1...8).
Four *control* fields marked with grey background (A...D).

Absolute Channel Value

Channel value is an unsigned ten-bit integer ranging from 0 to 1023. The Absolute 10-bit value of every one of the eight channels appears once in every frame, in a constant location. Since a new frame is transmitted every 44 milliseconds, the absolute channel data is also transmitted every 44 milliseconds.

Auxiliary Channel data

The Auxiliary channel data is a Quintet transmitted half-time between two Absolute channel values. The interpretation of this Quintet can be one of the following:

1. Extra channel data.
2. Fail Safe (FS) data.
3. Relative Channel data.

Control Field

There are four 10-bit *Control Fields. Each one* controls the six quintets preceding it. Its two most significant bits, bits 9 & 8, control the meaning of the two auxiliary channel data fields preceding the control field:

Ctrl Field	bit	Affected Aux Field
A	9	Aux 1
A	8	Aux 3
B	9	Aux 5
B	8	Aux 7
C	9	Aux 2
C	8	Aux 4
D	9	Aux 6
D	8	Aux 8

Table 2: Control Field - Bits 8 & 9

For example, Bit 9 (MSB) of Ctrl C, also marked as **C[9]**, controls the meaning of field Aux 2. In other words, C[9] is the control bit is Aux2.
The full meaning of an Aux field is fully determined by its control bit **and** its own bit 4 (MSB) as explained in the following table:

Ctrl bit	Aux MSB	Meaning of Aux field
0	-	Relative Channel Data
1	0	Extra Channel Data
1	1	Fail Safe Data (FS)

Table 3: Aux Field

Relative Channel Value

The LSB of the Auxiliary Quintet carries the *Polarity* of the Relative Channel Data:
AUX[0] = 0: Positive (add Delta value)
AUX[0] = 1: Negative (subtract Delta value)
The remaining four higher bits of Auxiliary Quintet carries the Range Index as previously explained:

Index	Range of Deltas
0	3 .. 4
1	5
2	6 .. 7
3	8 .. 10
4	11
5	12 .. 15
6	16 .. 19
7	20.. 24
8	25.. 31
9	32.. 43
10	44.. ?
11	?
12	?
13	?
14	?
15	?

Table 4: Range of Deltas

Fail-safe (FS) Data

Decoding

Given the Fail-safe 10-bit value:

FS_9	FS_8	FS_7	FS_6	FS_5	FS_4	FS_3	FS_2	0	0

Figure 4: Fail-safe 10-bit value

The Auxiliary Quintet is decoded as follows:

AUX[4]:	'1'.
AUX[3-2]:	Index
AUX[1-0]:	Partial FS value according to the index:

Index		FS Value	
0	0	FS_3	FS_2
0	1	FS_5	FS_4
1	0	FS_7	FS_6
1	1	FS_9	FS_8

Table 5: Aux Field when used to pass Fail-safe values

Discussion

In order for the transmitter to pass an updated FS value to the receiver, there is a need for four frames, each frame carrying two bits of the eight upper-bits of the FS value. This does not seem to be a problem since this value very rarely changes, and this is not usually done during flight.

The fact the the two LSB are always zero reduces the channel granularity from 1024 states to 255 states (8-bit data minus 1 state dedicated to 'disable FS'). This does not seem to be a problem since this value is used for saving the model rather than for manoeuvres.

The two *Index* bits fully specify the function of the two lower bits of the Auxiliary quintet. Therefore, the transmission of the data can be done randomly.

High Channel Data

Decoding

Channel 9 is passed over Aux1 (or Aux5) and Aux3 (or Aux7).
The upper nibble of the eight-bit value is the lower nibble of Aux1 (or Aux5).
The lower nibble of the eight-bit value is the lower nibble of Aux3 (or Aux7).

Channel 10 is passed over Aux2 (or Aux6) and Aux4 (or Aux8).
The upper nibble of the eight-bit value is the lower nibble of Aux2 (or Aux6).
The lower nibble of the eight-bit value is the lower nibble of Aux4 (or Aux8).

Discussion

Since the High Channels (CH9 & CH10) are multiplexed with other data (FS & Relative Channel Values) on the same location - they cannot be relied upon for transmission of real-time data. In addition, the fact that they are 8-bit channels further make them unattractive for regular use.
Normally, the High Channel data and the Fail-Safe data are passed alternately on Aux1, Aux3, Aux 2 and Aux4.
However, in case that one of the lower channels (1 to 4) is changing, the corresponding Aux field is taken over to transfer Relative Channel values. In such a case the corresponding High Channel nibble will be passed on the backup Aux field:

	Primary	Backup
Channel 9 upper nibble	Aux 1	Aux 5
Channel 9 lower nibble	Aux 3	Aux 7
Channel 10 upper nibble	Aux 2	Aux 6
Channel 10 lower nibble	Aux 4	Aux 8

Table 6: Channels 9 & 10

Error Detection

The lower eight bits of every control field (A,B,C,D) is used for error detection in the respective preceding 32 bits.

Implementation

At any given time, the receiver attributes an absolute channel-value for each channel. This value may be the actual absolute value transmitted by the transmitter, a calculated value or a Fail-Safe value.
The meaning of this value can interpreted as the deflection of the servo arm:
At mid-point of the servo arm is defined as 0%
Historically, a movement of 40° from the mid-point is defined as 100% movement. Clockwise movement (servo arm seen from above) is defined as positive and counter-clockwise movement is defined as negative.
Graupner systems allow a full scale servo travel of (+/-150 %) corresponding to an arm deflection of (+/- 60°).
An absolute value of 0 corresponds to +150% and an absolute value of 1023 corresponds to -150%.
The following formula allows to convert channel position values (0 to 1023) into servo deflection:

```
servo (%) = (300 / 1023) * position + 150
```

Following formula changes servo stroke (%) into channel position:

```
position = (1023/300) * servo + 511.5
```

Acknowledgements

This document is based on the work of *Tsutomu Seki* – the author of SmartPropo.
We have also much learned from the work done with the `Futaba PCM 1024Z format by W. Pasman.`

References

[1] PCM1024Z format: What's Known? By W. Pasman.
http://www.cg.its.tudelft.nl/~wouter/publications/pasman03k.pdf
[2] Sekiriki (2002). SmartPropo: The RC to PC Audio Interface.
http://www.sekiriki.jp/smartpropo/index.html
[3] SmartPropoPlus Software by Shaul Eizikovich
http://www.smartpropoplus.com