AGILE	IASF RM

DOCUMENT TYPE:	**TECHNICAL NOTE**	
TITLE: SAFEE TE pseudo ESA TM packets		
DOCUMENT No.:	AGILE-IAS-TN-019	**No. OF PAGES:** 47
ISSUE No.:	1	**DATE:** 23/10/2003
PREPARED BY:	L. PACCIANI	
CHECKED BY:	M.Trifoglio, A. Bulgarelli, F. Gianotti	
APPROVED BY:	M. FEROCI	**DATE:** 23/10/2003

CHANGE RECORD

Issue	Date	Page	Description of Changes	Release
1	23/10/2003		First issue related to SAFEE TE PSEUDO ESA TM PACKETS	1

Table of Contents

Rules

Introduction

SuperAgile SW team chose to use the existing DISCoS [1] tool to perform data storage and quick look analysis for Test Equipments.

This Software was developed to be detector indipendent. It only requests data in a pseudo ESA TM format, a <START> and a <STOP> telecommand [2] corrisponding to *acquisition start* and *stop* respectively.

> ***RULE 0***
>
> *Data directly produced from SAFEE TE are called "**Raw Data**",*
>
> *Raw Data translated in TM format are called "**TM Packets**" or "**TM Data**".*

The Raw Data format is specified in the annex A.

Raw Data produced from Test Equipment doesn't match ESA TM.

In this document we establish the procedure to translate "**Raw Data**"in **TM Packets**" and the TM format.

Some constraint comes from PacketLib [3] (a library developed to handle packets), that we chose to use for TM data manipulation .

1 SAFEE TE pseudo ESA TM packets

The quantum is a 16 bit Big Endian word,

Packets are almost in agreement with ESA standard for TM packets.

Each packet consists of

1. a fixed lenght Packet Header Field,
2. a Data Field. Its format depends on TM flavour, actually we foresee these TM packets:
   - Housekeeping (<HK>),
   - Scientific Data (<SCI>),
   - Extended Calibration Data (<CALEX>),
   - Configuration (<CONF>),
   - Operator log (<RUNLOG>),
   - Regin/Regout (<REGIO>),
   - Universal Time (<TUT>).

1.1 Packet Header

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
CCOE															
Version Number			Type	DFHF	APID										
SF		Source Sequence Counter													
Packet Lenght															

CCOE: specifies the lenght of the following packet (bytes). This word is not in ESA standard TM format, but is strongly suggested by Massimo Trifoglio and Fulvio Gianotti for compatibility with DISCoS tool. According to existing DISCoS version, the maximum packet lenght is 1024Bytes;

Version Number: 001 binary;

Type: 0 for TM (1 for TC);

DFHF: data field header flag, 1,

APID: Application Process Identifier (APID for SAFEE TE : 1296),

SF: Sequence Flag, 11 binary;

Source Sequence Counter: packet counter (one counter for every APID);

Packet Lenght: Data Field Lenght (bytes) –1.

IASF RM

1.2 Packet Data Field

This field format must match:

1. the incoming "Raw Data" format (based on 32 bit Little Endian Words) ,
2. the ESA Standard (based on 16 bit Big Endian Words),
3. the existing PacketLib (developed by A. Bulgarelli),
4. the existing DISCoS tool.

RULE 1

32bit Little Endian Words from "incoming RAW DATA" are translated into 16 bit Big Endian Words (necessary for pseudo ESA TM) in the following way:

Incoming 32 bits RAW DATA word is divided in 2 words of 16 bit (MSW and LSW). The MSW and LSW are byte-swapped (for Little → Big Endian Conversion).

The byte-swapped MSW is inserted in the packet, then the byte-swapped LSW is inserted.

This algorithm corresponds to a single 32 bit Little Endian Word swapped in a 32 bit Big Endian Word and then inserted in the packet.

In the following, when we refer to 32 bit words in packets, we assume that these words have been translated using *rule 1*.

TM Data Fields are divided in

1. **Data Field Header**,
2. **Source Data Field**.

1. In the **Data Field Header** we put all the informations needed to handle the Source Data Field. To distinguish between different TM packets, in the Data Field Header, we reserve two fields to identify the **Packet Type** and the **Packet SubType**.

   We chose:

   <SCI> : 15/1

   <CALEX> : 15/2

   <HK> : 1/1

   <CONF> : 1/2

   <RUNLOG> : 1/3

   <TUT> : 1/4

   <REGIO> : 1/5

   for SAFEE TE.

2. Usually the **Source Data Field** has an internal structure. Assuming to use the existing PacketLib, we can use:
   - the "PacketLib packet format scheme 1" [3] for <HK> (i.e. *N* Data Blocks with a fixed number of elements each one),
   - the "PacketLib packet format scheme 2" [3] for <SCI>, <CALEX>, <RUNLOG> (i.e. *N* Data Blocks, each Data Block has an header and a variable number of elements),
   - the "PacketLib packet format scheme 3" [3] for <CONF>, <REGIO>, <TUT> (i.e. One Data Block with a fixed number of elements each one).

1.2.1 Data Field for <SCI>and <CALEX>

According to the “PacketLib packet format scheme 2” we chose:

1.2.1.1 Data Field Header

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format version															
Number of events in the packet (SCI Data) Number of events in the packet (“Extended Calibration Data)															
Spare															
Used Running Parameters/Commands long words (0-7)															
Running Parameter/Command 1- MSW															
Running Parameter/Command 1 -LSW															
Running Parameter/Command 2 -MSW															
Running Parameter/Command 2 -LSW															
Running Parameter/Command 3 -MSW															
Running Parameter/Command 3 - LSW															
Running Parameter/Command 4 -MSW															
Running Parameter/Command 4 - LSW															
Running Parameter/Command 5 - MSW															
Running Parameter/Command 5 - LSW															
Running Parameter/Command 6 -MSW															
Running Parameter/Command 6 -LSW															
Running Parameter/Command 7 -MSW															
Running Parameter/Command 7 -LSW															

CheckSum Flag: 00 binary, **TBD**;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

We chose:

<SCI> : 15/1

<CALEX> : 15/2

for SAFEE TE and SAIE TE too.

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1, 1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [4]).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

Number of events in the packet: It corresponds to the number of Data Blocks (one block for event) in the packet.

Used Running Parameters/Commands:: this field specifies the number of valid Running Parameters/ Commands.

There is a fixed number of 14 word of 16 bit for Running Parameter/Command Words.

The Raw Data running parameter/command words are Unsigned Long Words (as specified for SAFEE TE Raw Data Format in the annex). These words are translated according to **Rule 1**.

1.2.1.2 <SCI> and <CALEX> Source Data Field

According to "PacketLib Packet Format scheme 2" [3], the Source Data Field is divided in Data Blocks. Each Block has an Header and a variable number of elements of fixed dimension.

There is a variable number of Data Blocks. In our case, there is a Data Block for every event.

1.2.1.2.1 Data Block Header

The meaning of each word of Block header is explained in the annex A.

MSW and LSW are obtained from the **Rule 1**.

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
V785 ADC Data Header MSW															
V785 ADC Data Header LSW															
V785 End of block MSW															
V785 End of block LSW															
SIS3600a (V3) Data Word MSW															
SIS3600a (V3) Data Word LSW															
SIS3600b (V4) Data Word MSW															
SIS3600b (V4) Data Word LSW															

The Block Lenght is obtainable from **V785 ADC Data Header Word**, Five bits of V785 ADC Data Header Word specify the number **(K)** of converted ADC Input Channels. Then the number of **Data**

Block elements is k.

1.2.1.2.2 Data Block Elements

The meaning of each Data Block Element is explained in the annex A.

MSW and LSW are obtained from the **Rule 1**.

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
V785 ADC Data Word 0 , MSW															
V785 ADC Data Word 0 , LSW															
. . .															
V785 ADC Data Word k-1, MSW															
V785 ADC Data Word k-1, LSW															

1.2.2 Data Field for <HK>

According to the “PacketLib packet format scheme 1” we chose:

1.2.2.1 Data Field Header for <HK>

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format Version															
Number of blocks															
spare															
Number of elements for each block															

CheckSum Flag: 00 binary TBD;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

We chose:

<HK> : 1/1

for SAFEE TE.

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [3]).

Number of Blocks: this is the number of HK blocks in the packet (chosen to be 4).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

Number of elements: this is the number of HK elements for each block (chosen to be 32).

1.2.2.2 <HK> Source Data Field

According to “PacketLib Packet Format scheme 1” [3], the Source Data Field is divided in Data Blocks. Each Block has a fixed number of elements of fixed dimension.

There is a variable number of Data Blocks.

In our case, there is a Data Block for every Stream of HouseKeepings.

We chose to have 4 blocks of Housekeepings in a <HK> (N=4).

1.2.2.2.1 Data Block Elements

The meaning of each Data Block Element is explained in the annex A. MSW and LSW are obtained from the **Rule 1**.

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
HK Word 0, MSW															
HKWord 0, LSW															
. . .															
HK Word K-1, MSW															
HK Word K-1, LSW															

1.2.3 Data Field for <REGIO>

According to the “PacketLib packet format scheme 3” we chose:

1.2.3.1 Data Field Header for <REGIO>

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format Version															
Dummy															

CheckSum Flag: 00 binary TBD;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

We chose:

<REGIO> : 1/5

the dummy word has value 1.

for SAFEE TE .

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [3]).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

1.2.3.2 <REGIO> Source Data Field

According to “PacketLib Packet Format scheme 3” [3], the Source Data Field has an unique Data Block. This Data Block has a fixed number of elements of fixed dimension. We have 123 Elements (every elements is composed of two 16 bit words).

1.2.3.2.1 Data Block Elements

The meaning of each Data Block Element is explained in the annex A. MSW and LSW are obtained from the **Rule 1**.

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
REGIO Word 0, MSW															
REGIOWord 0, LSW															
. . .															
REGIO Word 122, MSW															
REGIO Word 122, LSW															

1.2.4 Data Field for <CONF>

According to the “PacketLib packet format scheme 3” we chose:

1.2.4.1 Data Field Header for <CONF>

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format Version															
Dummy															

CheckSum Flag: 00 binary TBD;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

We chose:

<CONF> : 1/3

for SAFEE TE .

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [3]).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

1.2.4.2 <CONF> Source Data Field

According to “PacketLib Packet Format scheme 3” [3], the Source Data Field has an unique Data Block. This Data Block has a fixed number of elements of fixed dimension. We have **N** Elements (each element is composed of two 16 bit words, - **N**=92 for version 0, **N**=94 for version 1, N=103 for version 2).

The meaning of each Data Block Element is explained in the annex A. MSW and LSW are obtained from the **Rule 1**.

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CONF Word 0, MSW															
CONF Word 0, LSW															
. . .															
CONF Word **N**-1, MSW															
CONF Word **N**-1, LSW															

1.2.5 Data Field for <RUNLOG>

Each “Run Log” packet contains a row of “General Log”. “General Log” contains:

- the log fields coming from the “SAFEE TE GUI” [5] (concerning the init, start, stop of the Run);
- the “Run Log” with the comments written by the operator concerning the run (on the “Run Log” field of the SAFEE TE GUI);
- the “Session Log” with the comments written by the operator concerning the session (on the “Session Log” field of the SAFEE TE GUI);
- the log fields coming from the “SAFEE TE Handler” (concerning the reason of Run Stop, or kill, but still under test);
- The “Command Log” (with the commands[1] given to the TE);

According to “PacketLib Packet Format scheme 2” [3], we chose:

1.2.5.1 Data Field Header for <RUNLOG>

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format Version															
Number of blocks in the packet															

CheckSum Flag: 00 binary TBD;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

[1] The “Command Log” format is the following: there is a row with the command given to the TE (see [5] for the “commands format”), then other 6 rows follow, specifying:

- The PC Time since 00:00 UTC january 1,1970 expressed in
  - seconds (first row)
  - milliseconds (second row),
- The PC Time in hh:mm:ss,
- the last event number,
- the 24 LSB of the SAFEE TE internal time (1μs step size),
- the 24 MSB of the SAFEE TE internal time.

We chose:

<RUNLOG> : 1/3

for SAFEE TE.

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [3]).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

Number of blocks in the packet: this is the number of blocks contained in the packet (fixed to 1).

1.2.5.2 <RUNLOG> Source Data Field

According to "PacketLib Packet Format scheme 2" [3], the Source Data Field contains a variable number of blocks. We fixed the number of blocks to 1. This Block has a variable number of elements of fixed dimension. One element is an unsigned short word.

1.2.5.2.1 <RUNLOG> Data Block Header

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Number of characters of the row															
Index of Log row															

Number of characters of the row: this is the number of characters contained in the "Log Row" including the termination character.

Index of Log row: is the index of the row in the log file to be build (and the index of <RUNLOG>).

1.2.5.2.2 <RUNLOG> Data Block Elements

The "Log Row" could be even or odd bytes long. The number of elements is calculated on the basis of the "**Number of characters of the row**". If this number is odd, the last element contains a spare byte.

If m is the “**Number of characters of the row”** and m is odd, we have:

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Character 1								Character 2							
Character 3								Character 4							
. . .															
Character m								Spare							

If m is the “**Number of characters of the row”** and m is even, we have:

MSB LSB

0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
Character 1								Character 2							
Character 3								Character 4							
. . .															
Character m-1								Character m							

1.2.6 Data Field for <TUT>

This packet contains the "Run start" time. This information is given by the PC serving the acquisition. The accuracy of the time of the run start, estimated in this way, is around some tenth of millisecond. Moreover the PC time is not synchronized with other instruments or clocks (for example http://www.boulder.nist.gov/timefreq/cesium/parcs.htm).

1.2.6.1 Data Field Header for <TUT>

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Spare						Checksum flag		Packet Type				Packet SubType			
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Time Tag millisecond (16 bit unsigned)															
Packet Format Version															
Block Lenght (number of unsigned short words)															

CheckSum Flag: 00 binary TBD;

Packet TYPE/SUBTYPE: they specify the packet type/subtype.

We chose:

<TUT> : 1/4

for SAFEE TE and SAIE TE too.

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the creation of the packet (see [3]).

Packet Format Version: this field is filled with the current version for the format of the packet identified with APID/Type/SubType.

Block Lenght: this is the number of unsigned short words contained in the block, this number is fixed to 4.

1.2.6.2 <TUT> Source Data Field

According to "PacketLib Packet Format scheme 3" [3], the Source Data Field contains a single block.

This Block is 4 unsigned short words long.

MSB LSB

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Time Tag second (32 bit integer signed MSB)															
Time Tag second (32 bit integer signed LSB)															
Spare															
Time Tag millisecond (16 bit unsigned)															

Time Tag: CPU System Time expressed as second and millisecond since 00:00 UTC january 1,1970 (i.e. as given by the C function ftime) corresponding to the "Run Start" in the DAQ-PC reference system (see [3]).

ANNEX A

a. “CALEX” and “SCI” Data Words (from incoming Raw Data)

The quantum is a 32 bit Little Endian word.

There are **3 version of data** for CALEX and SCI.

The most general CALEX and SCI Data topology is shown below:

MSB LSB

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="32">.
.
.
.
.</td></tr>
<tr><td colspan="32">Running Parameter/Command Word</td></tr>
<tr><td colspan="32">Running Parameter/Command Word</td></tr>
<tr><td colspan="32">Running Parameter/Command Word</td></tr>
<tr><td colspan="32">Event j first word</td></tr>
<tr><td colspan="32">.
.
.</td></tr>
<tr><td colspan="32">Event j+m last word</td></tr>
<tr><td colspan="32">Running Parameter/Command Word</td></tr>
<tr><td colspan="32">.
.
.
.
.</td></tr>
</table>

A variable number of words specifying running parameters/commands (i.e. the position of a source for SCI, or the amplitude of the pulser for CALEX) are followed by a variable number of events (each event is described by a variable number of long words).

The sintax of each word is explained in the following.

a.a Raw Data Words Format

They are 32 bit Little Endian Words.

A three bit field (bit 24, 25, 26 - "**DATA WORD IDENTIFIER"** (**DWI**)) identifies words coming from V785 ADC and words for Running Parameters/Commands.

DWI for "**Running Parameters/Commands**" word is 001 binary,

DWI for "**V785 ADC Data Header**" word is 010 binary,

DWI for "**V785 ADC DATA**" word is 000 binary,

DWI for "**V785 End of Block**" word is 100 binary.

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

Data Words coming from **SIS3600** modules have no **DWI** but they follow the "V785 End of Block" word.

This format guarantees the Direct Data Access on Raw Data.

a.a.a Available Running Parameters/Commands

Generic Format for Running Parameters/Commands:

MSB LSB

31	*30*	*29*	*28*	*27*	*26*	*25*	*24*	*23*	*22*	*21*	*20*	*19*	*18*	*17*	*16*	*15*	*14*	*13*	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
Parameter/ Command ID					**0**	**0**	**1**	Parameter/ Command																							

Parameter/Command ID: Specifies the running parameter. Available values are:

XA Channel Under Test:

0	0	0	0	0	**0**	**0**	**1**	0	0	0	0	0	0	0	0	0	0	Channel	XA	D.C.	SAFEE

Channel: is the XA channel under test (0-127 channels, 7 bit),

XA: identifies the XA chip inside a D.C. (1-3) to be tested (2 bit),

D.C.: identifies the daisy chain inside a SAFEE to be tested (3 bit),

SAFEE: identifies the SAFEE to be tested (2 bit).

Pulse:

0	0	0	0	1	**0**	**0**	**1**	0	0	0	MPV901P Raw Setting	D.C.	SAFEE

SAFEE: identifies the SAFEE where pulses are sent (2 bit),

D.C.: identifies the daisy chain inside a SAFEE where pulses are sent (2 bit) ,

MPV901P Raw Setting: specifies the DAC amplitude for the pulse (16 bit). The nominal pulse height ($\mathbf{Pulse}_{mV}$) applied to each SAFEE channel is obtained from the Raw value ($\mathbf{Pulse}_{raw}$) using the following expression:

$$Pulse_{mV} = \frac{1.22}{200} \times Pulse_{raw}$$

Set SAFEE DAC:

0	0	0	1	0	**0**	**0**	**1**	0	0	0	0	0	0	0	0	DAC Value	DAC	AD8842 id	SAFEE

SAFEE: identifies the SAFEE were SAFEE DAC is located (2 bit),

AD8842 id: identifies the chip inside the SAFEE where SAFEE DAC channel is located (2 bit),

DAC: specifies the DAC channel inside the chip that is configured (4 bit).

DAC Value: specifies the DAC amplitude (8 bit).

Single Event Data Words for “SCI” and “CALEX Data (a single event of the SAFEE Test Equipment) contains:

1. a group of variable number of words for the ADC module (V785 from CAEN) description of the event,
2. two words for digital information (we use two SIS3600 modules from SBS).

The V785 words are explained in V785 manual available from the CAEN web page http://www.caen.it/nuclear.

An event in the output buffer of this module is described with:

1. a **V785 Data Header** word (containing a five bit field with the number **(K)** of input channel over a certain threshold),
2. **K** words of **"V785 Data"** (containing the **"Pulse Height Amplitude"** of the input channel specified in the **"Input Channel ID"** field),
3. a **"V785 end of block"** used as trailer. It contains the **"Gate Counter"** field, this counter could be programmed in two way:
   - to count only gates arriving when the ADC module is not busy,
   - to count all gates for the ADC module.

Here follows the V785 **"Input Channel ID"** assigment for the signals coming from SAFEE:

Input Channel ID	*SAFEE Signal*
0	MGO-1 (MGO for D.C. 1)
1	Out-1 (Energy for D.C. 1)
2	D2-1 (Addr. High for D.C. 1)
3	D1-1 (Addr. Low for D.C. 1)
4	MGO-2 (MGO for D.C. 2)
5	Out-2 (Energy for D.C.2)
6	D2-2 (Addr. High for D.C. 2)
7	D1-2 (Addr. Low for D.C. 2)
8	MGO-3 (MGO for D.C. 3)
9	Out-3 (Energy for D.C. 3)
10	D2-3 (Addr. High for D.C. 3)
11	D1-3 (Addr. Low for D.C. 3)
12	MGO-4 (MGO for D.C. 4)
13	Out-4 (Energy for D.C. 4)
14	D2-4 (Addr. High for D.C. 4)
15	D1-4 (Addr. Low for D.C. 4)

In Raw data Version 0 and 2 the following two 32 bits fields are filled with

SIS3600 data words that contain two digital words of 32 bits acquired for every event.

In DAQ source code this modules are called SIS3600a and SIS3600b, while in TE hardware they are called V3 and V4.

The information stored in this two long words are:

The whole **“Good Event” Counter**,

the 4 LSb of **“Event Counter**” (acquired by both SIS3600 modules to check modules syncronization).

the **“Time Counter” (with 1us resolution),**

the **M.G.O. Pattern**: each bit is “1” if the corrisponding D.C. gives a valid trigger signal for that event.

In Raw data Version 1 the following two 32 bits fields are filled with

PC DAQ Time Tag: DAQ CPU System Time of the event, expressed as second and microsecond since 00:00 UTC january 1,1970 (see [3]).

Data Version 0:

MSB LSB

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="32">V785 ADC Data header:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>1</td><td>0</td><td colspan="8">dummy</td><td>0</td><td>0</td><td colspan="6">Converted input channels</td><td colspan="8">dummy</td></tr>
<tr><td colspan="32">V785 ADC data word 1:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="6">Input channel ID</td><td colspan="2">dummy</td><td>UN</td><td>OV</td><td colspan="12">Pulse Height Amplitude</td></tr>
<tr><td colspan="32">V785 ADC data word 2,
.
.
V785 ADC data word k.

V785 End of Block:</td></tr>
<tr><td colspan="5">dummy</td><td>1</td><td>0</td><td>0</td><td colspan="24">Gate counter (all events or good events only)</td></tr>
<tr><td colspan="32">SIS3600a (VME module V3) data (inverted logic):</td></tr>
<tr><td colspan="24">Board 3 Event Counter
bit 24.. .... bit 1</td><td colspan="4">MGO Pattern
D.C.4 … D.C. 1</td><td colspan="4">Board 4 Time Counter
(1us step)
bit 29 ... bit 32</td></tr>
<tr><td colspan="32">SIS3600b (VME module V4) data (inverted logic):</td></tr>
<tr><td colspan="12">Board 4 Time Counter
(1us step)
bit 17 .... bit 28</td><td colspan="4">Board 3 Event Counter
bit 1 .... bit 4</td><td colspan="16">Board 4 Time Counter
(1us step)
bit 1 .... bit 16</td></tr>
</table>

Data Version 1:

MSB LSB

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="32">V785 ADC Data header:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>1</td><td>0</td><td colspan="8">dummy</td><td>0</td><td>0</td><td colspan="6">Converted input channels</td><td colspan="8">dummy</td></tr>
<tr><td colspan="32">V785 ADC data word 1:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="6">Input channel ID</td><td colspan="2">dummy</td><td>UN</td><td>OV</td><td colspan="12">Pulse Height Amplitude</td></tr>
<tr><td colspan="32">V785 ADC data word 2,
.
.
V785 ADC data word k.

V785 End of Block:</td></tr>
<tr><td colspan="5">dummy</td><td>1</td><td>0</td><td>0</td><td colspan="24">Gate counter (all events or good events only)</td></tr>
<tr><td colspan="32">DAQ PC Time for the event, word a:</td></tr>
<tr><td colspan="32">DAQ PC time tag seconds (32 bit integer signed MSB)
bit 32... .... bit 1</td></tr>
<tr><td colspan="32">DAQ PC Time for the event, word b:</td></tr>
<tr><td colspan="32">DAQ PC time tag microseconds (32 bit integer signed LSB)
bit 32... .... bit 1</td></tr>
</table>

Data Version 2:

MSB LSB

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="32">V785 ADC Data header:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>1</td><td>0</td><td colspan="8">dummy</td><td>0</td><td>0</td><td colspan="6">Converted input channels</td><td colspan="8">dummy</td></tr>
<tr><td colspan="32">V785 ADC data word 1:</td></tr>
<tr><td colspan="5">dummy</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="6">Input channel ID</td><td colspan="2">dummy</td><td>UN</td><td>OV</td><td colspan="12">Pulse Height Amplitude</td></tr>
<tr><td colspan="32">V785 ADC data word 2,
.
.
V785 ADC data word k.

V785 End of Block:</td></tr>
<tr><td colspan="5">dummy</td><td>1</td><td>0</td><td>0</td><td colspan="24">Gate counter (all events or good events only)</td></tr>
<tr><td colspan="32">SIS3600a (VME module V3) data (inverted logic):</td></tr>
<tr><td colspan="24">Board 3 Event Counter
bit 24... .... bit 1</td><td colspan="4">MGO Pattern
D.C.4 … D.C. 1</td><td colspan="4">Board 4 Time Counter
(1us step)
bit 32 ... bit 29</td></tr>
<tr><td colspan="32">SIS3600b (VME module V4) data (inverted logic):</td></tr>
<tr><td colspan="4">Board 3 Event Counter
bit 4 .... bit 1</td><td colspan="28">Board 4 Time Counter
(1us step)
bit 28 .... bit 1</td></tr>
</table>

b. “Housekeeping” Data (from incoming Raw Data)

HK data are blocks of 32 elements (16 words for 16 Counters and 16 words for 16 Voltmeter).

Data are sequential, a field of 8 bits (**Element ID**) guarantees against data asynchronization.

MSB LSB

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

DATA WORDS:

block j:

Element ID (0)	Counter 0
Element ID (1)	Counter 1
Element ID (2)	Counter 2
Element ID (3)	Counter 3
Element ID (4)	Counter 4
Element ID (5)	Counter 5
Element ID (6)	Counter 6
Element ID (7)	Counter 7
Element ID (8)	Counter 8
Element ID (9)	Counter 9
Element ID (10)	Counter 10
Element ID (11)	Counter 11
Element ID (12)	Counter 12
Element ID (13)	Counter 13
Element ID (14)	Counter 14
Element ID (15)	Counter 15
Element ID (16)	Nominal d.d.p for voltmeter 0 (mV)
Element ID (17)	Nominal d.d.p for voltmeter 1 (mV)
Element ID (18)	Nominal d.d.p for voltmeter 2 (mV)
Element ID (19)	Nominal d.d.p for voltmeter 3 (mV)
Element ID (20)	Nominal d.d.p for voltmeter 4 (mV)
Element ID (21)	Nominal d.d.p for voltmeter 5 (mV)
Element ID (22)	Nominal d.d.p for voltmeter 6 (mV)
Element ID (23)	Nominal d.d.p for voltmeter 7 (mV)
Element ID (24)	Nominal d.d.p for voltmeter 8 (mV)
Element ID (25)	Nominal d.d.p for voltmeter 9 (mV)
Element ID (26)	Nominal d.d.p for voltmeter 10 (mV)
Element ID (27)	Nominal d.d.p for voltmeter 11 (mV)

MSB LSB

Element ID (28)	Nominal d.d.p for voltmeter 12 (mV)
Element ID (29)	Nominal d.d.p for voltmeter 13 (mV)
Element ID (30)	Nominal d.d.p for voltmeter 14 (mV)
Element ID (31)	Nominal d.d.p for voltmeter 15 (mV)

Block j+1,

.

.

Block j+N-1.

Actually we have the following channel map:

Counter 0 is for **"OR MGO"**,

Counter 1 is for **"MGO D.C. 1"**,

Counter 2 is for **"MGO D.C. 2"**,

Counter 3 is for **"MGO D.C. 3"**,

Counter 4 is for **"MGO D.C. 4"**,

Counter 5 is for **"GOOD EVENT"**,

Counter 6 is for **"TIME LSW** (i.e. the 24 less significative bits),

Counter 7 is for **"TIME MSW"** (i.e. The 24 most significative bits).

All other counters are unused .

Voltmeter words are the nominal Voltage (mV) of 16 sources (actually they are unused).

c. “REGIO” Data (from incoming Raw Data)

“REGIO” data include a Long Word specifying the **Daisy Chain** where we have sent the **Register IN** and that has sent us its **Register OUT**.

There are 61 Long Words for **Register IN** and 61 Long Words for **Register OUT**.

Register IN (and **Register OUT**) can be descripted using 243 bytes also.

In the following table we give the raw data format for **REGIO** Data.

MSB LSB

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

Daisy Chain Identifier:

Spare	D.C. id

Register IN Long Word 0:

Byte 3	Byte 2	Byte 1	Byte 0

.

.

.

.

Register IN Long Word 60:

spare	Byte 242	Byte 241	Byte 240

Register OUT Long Word 0:

Byte 3	Byte 2	Byte 1	Byte 0

.

.

.

.

Register OUT Long Word 60:

spare	Byte 242	Byte 241	Byte 240

Byte 0 contains the first 8 bit of the **Register IN**.

The Raw Data Format follows the iussue given from hardware implementation of the **Register IN**.

The bit 0 of the **Register IN** is mapped in bit 7 of the Byte 0,

The bit 1 of the **Register IN** is mapped in bit 6 of the Byte 0,

…

The bit 7 of the **Register IN** is mapped in bit 0 of the Byte 0,

The bit 8 of the **Register IN** is mapped in bit 7 of the Byte 1,

The bit 9 of the **Register IN** is mapped in bit 6 of the Byte 1,

…

The bit 15 of the **Register IN** is mapped in bit 0 of the Byte 1,

The bit 16 of the **Register IN** is mapped in bit 7 of the Byte 2,

And so on.

The same Raw Data Format is used for **Register OUT**.

d. "CONF" Data (from incoming Raw Data)

a.a Configuration "Version 0"

In the following table we give the raw data format for **CONF** Data, version 0.

MSB LSB

31	*30*	*29*	*28*	*27*	*26*	*25*	*24*	*23*	*22*	*21*	*20*	*19*	*18*	*17*	*16*	*15*	*14*	*13*	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
Conf Words:																															
Run Code																															
Ascii Raw Run ID (run mode, date, Run No) – Byte 0 - 3																															
. . .																															
Ascii Raw Run ID (run mode, date, Run No) – Byte 28 – 31																															

MSB	LSB

Store HK?
HK Sampling Period
Get Register OUT?
V785 Enable Mask (*)
V785 ADC input channel 0 lower Threshold (mv)
. . .
V785 ADC input channel 31 lower Threshold (mv)
SAFEE DAC 1 of CHIP 1 raw setting
. . .
SAFEE DAC 8 of CHIP 1 raw setting
SAFEE DAC 1 of CHIP 2 raw setting
. . .
SAFEE DAC 8 of CHIP 2 raw setting
SAFEE DAC 1 of CHIP 3 raw setting
. . .
SAFEE DAC 8 of CHIP 3 raw setting
Store Normal Data? (*)
Minimum Operations Period (s) (*)
Max events to acquire (*)
Max acquisition time (s) (*)
Begin scan from XA channel (**)
Begin Scan From XA (**)
Begin Scan From D.C. (**)
Begin Scan From SAFEE (**)
End Scan at XA channel (**)
End Scan at XA (**)
End Scan at D.C. (**)
End Scan at SAFEE (**)
Lower Pulse Height (**)
Pulse Step Size (***)
Number of Pulse Steps (***)
Number of Pulses inside one Pulse Train (**)
Number of Trains with the same Pulse Amplitude (**)
SAFEE DAC Lower Value (****)
SAFEE DAC Step Size (****)

MSB LSB

SAFEE DAC Number of Steps (****)
Fine Threshold Lower Value (*****)
Fine Threshold Step Size (*****)
Fine Threshold Number of Steps (*****)

a.b Configuration “Version 1”

In the following table we give the raw data format for **CONF** Data, version 1.

MSB LSB

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Conf Words:																															
Run Code																															
Ascii Raw Run ID (run mode, date, Run No) – Byte 0 - 3																															
. . .																															
Ascii Raw Run ID (run mode, date, Run No) – Byte 28 – 31																															
Store HK?																															
HK Sampling Period																															
Get Register OUT?																															
V785 Enable Mask (*)																															
V785 ADC input channel 0 lower Threshold (mv)																															
. . .																															
V785 ADC input channel 31 lower Threshold (mv)																															
SAFEE DAC 1 of CHIP 1 raw setting																															
. . .																															
SAFEE DAC 8 of CHIP 1 raw setting																															
SAFEE DAC 1 of CHIP 2 raw setting																															
. . .																															
SAFEE DAC 8 of CHIP 2 raw setting																															
SAFEE DAC 1 of CHIP 3 raw setting																															

MSB LSB

. . .
SAFEE DAC 8 of CHIP 3 raw setting
Store Normal Data? (*)
Minimum Operations Period (s) (*)
Max events to acquire (*)
Max acquisition time (s) (*)
Begin scan from XA channel (**)
Begin Scan From XA (**)
Begin Scan From D.C. (**)
Begin Scan From SAFEE (**)
End Scan at XA channel (**)
End Scan at XA (**)
End Scan at D.C. (**)
End Scan at SAFEE (**)
Lower Pulse Height (**)
Pulse Step Size (***)
Number of Pulse Steps (***)
Number of Pulses inside one Pulse Train (**)
Number of Trains with the same Pulse Amplitude (**)
Train Period (μs) (**)
Pulse Period (μs) (**)
SAFEE DAC Lower Value (****)
SAFEE DAC Step Size (****)
SAFEE DAC Number of Steps (****)
Fine Threshold Lower Value (*****)
Fine Threshold Step Size (*****)
Fine Threshold Number of Steps (*****)

a.c Configuration “Version 2”

In the following table we give the raw data format for **CONF** Data, version 2.

MSB LSB

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="32">Conf Words:</td></tr>
<tr><td colspan="32">Safee Code</td></tr>
<tr><td colspan="32">Run Code</td></tr>
<tr><td colspan="32">Ascii Raw Run ID (run mode, date, Run No) – Byte 0 – 3</td></tr>
<tr><td colspan="32">. . .</td></tr>
<tr><td colspan="32">Ascii Raw Run ID (run mode, date, Run No) – Byte 28 – 31</td></tr>
<tr><td colspan="32">Store HK?</td></tr>
<tr><td colspan="32">HK Sampling Period</td></tr>
<tr><td colspan="32">Get Register OUT?</td></tr>
<tr><td colspan="32">V785 Enable Mask (*)</td></tr>
<tr><td colspan="32">V785 ADC input channel 0 lower Threshold (mv)</td></tr>
<tr><td colspan="32">. . .</td></tr>
<tr><td colspan="32">V785 ADC input channel 31 lower Threshold (mv)</td></tr>
<tr><td colspan="32">SAFEE DAC 1 of CHIP 1 raw setting</td></tr>
<tr><td colspan="32">. . .</td></tr>
<tr><td colspan="32">SAFEE DAC 8 of CHIP 1 raw setting</td></tr>
<tr><td colspan="32">SAFEE DAC 1 of CHIP 2 raw setting</td></tr>
<tr><td colspan="32">. . .</td></tr>
<tr><td colspan="32">SAFEE DAC 8 of CHIP 2 raw setting</td></tr>
<tr><td colspan="32">SAFEE DAC 1 of CHIP 3 raw setting</td></tr>
<tr><td colspan="32">. . .</td></tr>
<tr><td colspan="32">SAFEE DAC 8 of CHIP 3 raw setting</td></tr>
<tr><td colspan="32">SAFEE DAC 1 of CHIP 4 raw setting</td></tr>
<tr><td colspan="32">. .</td></tr>
</table>

MSB LSB

.
SAFEE DAC 8 of CHIP 4 raw setting
Max events to acquire (*)
Max acquisition time (s) (*)
Begin scan from XA channel (**)
Begin Scan From XA (**)
Begin Scan From D.C. (**)
Begin Scan From SAFEE (**)
End Scan at XA channel (**)
End Scan at XA (**)
End Scan at D.C. (**)
End Scan at SAFEE (**)
Lower Pulse Height (**)
Pulse Step Size (***)
Number of Pulse Steps (***)
Number of Pulses inside one Pulse Train (**)
Number of Trains with the same Pulse Amplitude (**)
Train Period (μs) (**)
Pulse Period (μs) (**)
SAFEE DAC Lower Value (****)
SAFEE DAC Step Size (****)
SAFEE DAC Number of Steps (****)
Fine Threshold Lower Value (*****)
Fine Threshold Step Size (*****)
Fine Threshold Number of Steps (*****)

(*) This field contains a valid value only for “Normal” Run.

(**) This field contains a valid value for “PULSE”, “CALIB”, “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN”.

(***) This field contains a valid value for “CALIB”, “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN”.

(****) This field contains a valid value for “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN”.

(*****) This field contains a valid value only for “FINE THR SCAN”.

Safee Code: this field contains the Code for the SAFEE under test, as given in the following table:

Safee CODE	Safee under test

0	Safee from IDEAS
1	SAFEE from ILFA, “SEM 1”
2	SAFEE from ILFA, “SEM 2”
3	SAFEE from ILFA, “bread board”
4	SAFEE from ILFA with 32 dac channels

Run Code: this field contains the Run Code as given in the following table:

Run CODE	RUN Mode
1	Normal Run
2	Pulse Run
3	CALIB Run
4	THR SCAN
5	PREBIAS Scan
6	VFS Scan
7	VFP Scan
8	TRIGWBIAS Scan
9	RESWBIAS Scan
10	FINE THR Scan
11	STRBIAS Scan
12	SHABIAS Scan
13	VTHRBIAS Scan
14	OTABIAS Scan
15	TRIGDELBIAS Scan
16	IBDAC Scan

Ascii Raw Run ID: This is a 32 bytes string to be used to correlate the files produced by DISCoS and by Raw Data First Acquisition.
Store HK?: Only the bit 0 of this field is used. It is 0 if the Housekeeping are not Stored, 1 otherwise.
HK Sampling Period: This field specifies the Housekeeping sampling period (in s).
Get Register OUT: Only the bit 0 of this field is used. It is 0 if the Register OUT are not Stored, 1 otherwise.
V785 Enable Mask: bit “j” of this field is filled with 0 if the V785 ADC input channel “j” is not enabled, 1 otherwise. This field is meaningful only for “NORMAL” Run Mode.
V785 input channel “j” lower Threshold: this filed specifies the lower threshold (in mV) of the channel “j” of the V785 ADC module.
SAFEE DAC “j” of CHIP “i” raw setting: This field specifies the raw configuration of the DAC “j” inside CHIP “i”.

Store Normal Data?: Only the bit 0 of this field is used. It is 0 if the "NORMAL" Data are not Stored, 1 otherwise. This field is meaningful for "NORMAL" Run only.
Minimum Operations Period: This field specifies the time duration (in s) of the "Minimum Operations Period" of the "Normal" Runs. This field is meaningful for "NORMAL" Run only.
Max Events to acquire: This is the max number of events that the Test Equipment is programmed to acquire (in the "Normal" Runs). This field is meaningful for "NORMAL" Run only.
Max Acquisition Time: This is the duration of the Normal Run (in s). This field is meaningful for "NORMAL" Run only.
Begin Scan From XA Channel, **Begin Scan From XA**, **Begin Scan from D.C.**, **Begin Scan From SAFEE**: These fields specify the coordinates of the first channel for a "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs. These fields are meaningful for "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs.

End Scan at XA Channel, **End Scan at XA**, **End Scan at D.C.**, **End Scan at SAFEE**: These fields specify the coordinates of the last channel for a "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs. These fields are meaningful for "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs.

Lower Pulse Height: Specifies the Lower Pulse Amplitude (in mV) in every run were the pulser is used. For "PULSE" Run it Specifies the Pulse Amplitude (in mV). This field is meaningful for "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs.

Pulse Step Size: Specifies the Pulse Step Size (in mV) in every run were the pulser is used for a pulse amplitude scan. This field is meaningful for "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs.

Number of Pulse Steps: Specifies the number of Pulse Steps in every run were the pulser is used for a pulse amplitude scan. This field is meaningful for "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS SCAN", "FINE THR SCAN" runs.

Number of Pulses inside one Pulse Train: Specifies how many pulses with a given Amplitude are sent by the pulser inside one "Pulse Train". This field is meaningful for "PULSE", "CALIB", "THR SCAN", "PREBIAS SCAN", "VFS SCAN", "VFP SCAN", "TRIGWBIAS SCAN", "RESWBIAS

SCAN”, “FINE THR SCAN” runs.

Number of Trains with the same Pulse Amplitude: Specifies how many “Pulse Trains” with a given “Pulse Amplitude” are sent by the pulser. This field is meaningful for “PULSE”, “CALIB”, “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN” runs.

Train Period (µs): Specifies the nominal time interval between two Pulse Trains.

Pulse Period (µs): Specifies the nominal time interval between two Pulses inside the same Pulse Train.

SAFEE DAC Lower Value: This field specifies the lower “SAFEE DAC raw value” in every run were there is a scan of the voltage supplied by some SAFEE DAC. This field is meaningful for “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN” runs.

SAFEE DAC Step Size: This field specifies the step size of the “SAFEE DAC” scan in every run were there is a scan of the voltage supplied by some SAFEE DAC. This field is meaningful for “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN” runs.

SAFEE DAC Number of Steps: This field specifies the number of steps of the “SAFEE DAC” scan in every run were there is a scan of the voltage supplied by some SAFEE DAC. This field is meaningful for “THR SCAN”, “PREBIAS SCAN”, “VFS SCAN”, “VFP SCAN”, “TRIGWBIAS SCAN”, “RESWBIAS SCAN”, “FINE THR SCAN” runs.

Fine Threshold Lower Value: This field specifies the lower value of “Fine Threshold” in every run were there is a scan of the XA internal DACs. This field is meaningful for “FINE THR SCAN” run.

Fine Threshold Step Size: This field specifies the step size of the “Fine Threshold” scan in every run were there is a scan of the XA internal DACs. This field is meaningful for “FINE THR SCAN” run.

Fine Threshold Number of Steps: This field specifies the number of steps of the “Fine Threshold” scan in every run were there is a scan of the XA internal DACs. This field is meaningful for “FINE THR SCAN” run.

Bibliography

1: M. Trifoglio et al, astro-ph/0101122, 9 January 2001
2: M. Trifoglio et al, AGILE-ITE-SR-001, January/2002
3: A. Bulgarelli, Data Field Handled by PacketLib, 8/5/2
4: M. Trifoglio et al, AGILE-ITE-SR-003, April/2002
5: L. Pacciani, Guida d'uso all'interfaccia grafica del software di acquisizione per il test equipment della SAFEE, November/2002